Openstack Virtual Machine Plugin

Bacula Systems Documentation

Contents

Contents

Important: Remember to read the Best Practices chapter common for all of our hypervisor plugins.

The following article aims at presenting the reader with information about the Bacula Enterprise Openstack VM Plugin. Through subchapters, more in-depth information can be found about the following topics:

1 Scope

The **Bacula Enterprise Openstack-VM Plugin** currently supports the following platforms:

- 2023.2 Bobcat
- 2023.1 Antelope

See also:

- *Features*
- *Architecture*
- *Installation*
- *Configuration*
- *Operations*
- *Backup and Restore Strategies*
- *Troubleshooting*
- *Limitations*

Go back to the *main Openstack VM page*.

2 Features

The main feature of Bacula Enterprise Openstack VM Plugin is to offer Full block level backup and restore of instance volume(s).

See also:

- *Scope*
- *Architecture*
- *Installation*
- *Configuration*
- *Operations*
- *Backup and Restore Strategies*
- *Troubleshooting*
- *Limitations*

Go back to the *main Openstack VM page*.

3 Architecture

Bacula Enterprise Openstack-VM Plugin is a Bacula File Daemon plugin built over Openstack Cinder-Backup service.

All information is obtained using a custom implementation of a Cinder-Backup driver feeding data from Openstack to Bacula or the other way around.

Below, there is a simplified vision of the architecture of this plugin within a generic **Bacula Enterprise** deployment:

3.1 Cinder Bacula Driver Backup

During volume backup operations, for every file to backup, the bacula driver will:

- Keep track of backup file name
- Snapshot volume
- Create a FIFO (named pipe) from Openstack to Bacula
- Send relevant command to Bacula to synchronize the named pipe
- Return opened FIFO for Cinder to write into.

Once all files are backed up, the process must be stopped by:

- Closing the named pipe
- Gathering logs from Bacula
- Checking the list of backup files and job statuses
- Deleting the named pipe.

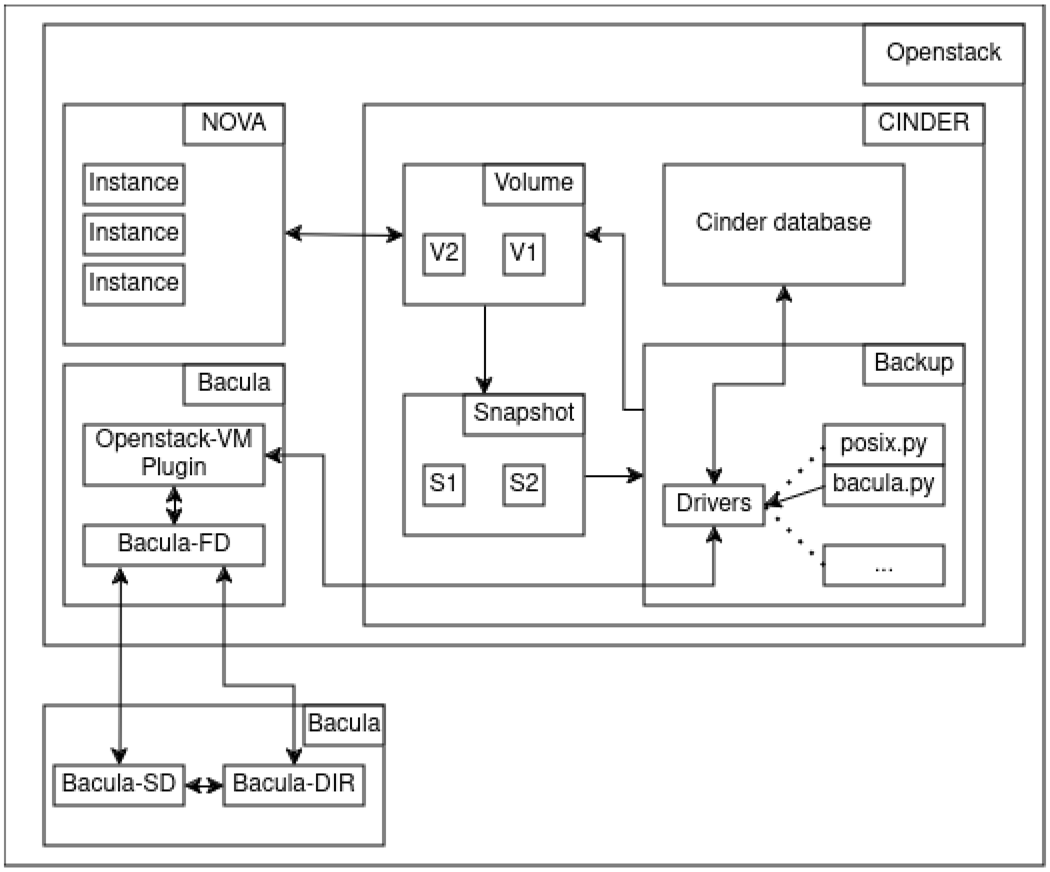


Fig. 1: Openstack-VM Plugin Architecture

3.2 Cinder Bacula Driver Restore

During volume restore operation, the bacula driver will communicate with Bacula through two different channels.

The first instance will handle the restore procedure by performing the process analog to backup apart for the fact the Cinder will read into the named pipe.

The closing process is also analog to the backup process.

The second instance will provide Cinder with the list of restored files to compare with its own file list by:

- Opening the named pipe to Bacula.
- Gathering the backup job file list.
- Returning curated output to Cinder for control.

3.3 Encrypted Volume Support

Volumes encrypted with `LUKS` are supported by the Cinder driver API. However, the encryption keys usually managed by the Openstack Barbican service should be backed up separately following the Openstack backup procedure.

https://docs.openstack.org/operations-guide/ops-backup-recovery.html

See also:

- *Scope*
- *Features*
- *Installation*
- *Configuration*
- *Operations*
- *Backup and Restore Strategies*
- *Troubleshooting*
- *Limitations*

Go back to the *main Openstack VM page*.

4 Installation

This article describes how to install Bacula Enterprise Openstack VM Plugin.

The installation process consists of two parts.

Note: Bacula Enterprise Openstack Plugin must be installed on Openstack host machine.

First, the installation of the bacula-enterprise-openstack-vm plugin with the BIM tool.

Second, configure the plugin as described in *Configuration*.

Third. by running the install script located at `/opt/bacula/scripts/install-openstack-vm.sh` two times and adjusting the Bacula director configuration.

- First time with the *configure* option `root@user:~# /opt/bacula/scripts/install-openstack-vm.sh configure`

- Second time with the *install* option `root@user:~# /opt/bacula/scripts/install-openstack-vm.sh install`
- At this point a configuration sample located at `/opt/bacula/openstack/bacula-dir.conf.sample` is created. Inside this file, there is a configuration example that should be adjusted and added to `/opt/bacula/etc/bacula-dir.conf`.
- The install script can be run the third time with the *test* option with `root@user:~# /opt/bacula/scripts/install-openstack-vm.sh test` to check if the installation is correct.

Note: If the Bacula director already has a *Client* resource, the Client in `bacula-dir.conf.sample` should be ignored as the Client resource should not be duplicated.

4.1 Steps

Here is an example how the install script should be used.

1. Run:

```
root@user:~# /opt/bacula/scripts/install-openstack-vm.sh configure

Enter the unix Openstack account name [stack]:

Enter the Bacula Director Name [stackdev-dir]:

Enter the Bacula Director Address [stackdev]:

Enter the Bacula Director Port [9101]:

Enter the Bacula FileDaemon name [stackdev-fd]:

INFO: Creating configuration template for the Director
      /opt/bacula/openstack/bacula-dir.conf.sample will help you to setup
      a Job with the Bacula Enterprise Openstack Plugin.

      The template can be included in your Director configuration and
      you need to review all items marked as "might need to be adjusted"

root@user:~# /opt/bacula/scripts/install-openstack-vm.sh install

Enter the unix Openstack account name: [stack]


Enter path to cinder drivers folder or automatically search system for it

'/opt/stack/cinder/cinder/backup/drivers/bacula.py' -> '/opt/bacula/share/bacula.py'
```

2. Once the director configuration is updated, run:

```
root@stackdev:/opt/bacula# scripts/install-openstack-vm.sh test

Enter the unix Openstack account name: [stack]

```

(continues on next page)

(continued from previous page)

```
1000 OK: 10002 stackdev-dir Version: 18.0.2 (05 March 2024)
INFO: Connection to the Director OK
INFO: Connection from the Director to the Client OK
INFO: Plugin installed correctly
INFO: Job found on the Director
INFO: FileSet configured on the Director
INFO: RestoreJob found on the Director
INFO: Test job finished ok
```

4.2 Result

Openstack VM Plugin is installed.

See also:

- *Scope*
- *Features*
- *Architecture*
- *Configuration*
- *Operations*
- *Backup and Restore Strategies*
- *Troubleshooting*
- *Limitations*

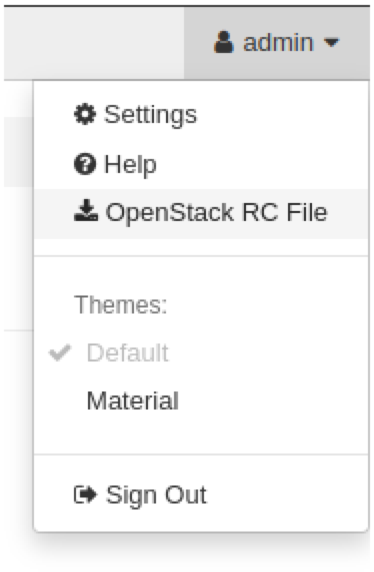
Go back to the *main Openstack VM page*.

5 Configuration

The following article presents the configuration of the plugin.

1. Download the `admin_openrc` file.

   Downloading `admin_openrc.sh` script can be done through the Openstack dashboard. To do so, the user can click on the `OpenStack RC File` menu item located at the top right of the dashboard.



2. Inside the `admin_openrc.sh` file comment or replace both `echo "Please enter your OpenStack Password for project $OS_PROJECT_NAME as user $OS_USERNAME: "` and `read -sr OS_PASSWORD_INPUT` with `OS_PASSWORD_INPUT=<password>` like in the example below.

```
admin-openrc.sh - Bash
#!/usr/bin/env bash
# To use an OpenStack cloud you need to authenticate against the Identity
# service named keystone, which returns a **Token** and **Service Catalog**.
# The catalog contains the endpoints for all services the user/tenant has
# access to - such as Compute, Image Service, Identity, Object Storage, Block
# Storage, and Networking (code-named nova, glance, keystone, swift,
# cinder, and neutron).
#
# *NOTE*: Using the 3 *Identity API* does not necessarily mean any other
# OpenStack API is version 3. For example, your cloud provider may implement
# Image API v1.1, Block Storage API v2, and Compute API v2.0. OS_AUTH_URL is
# only for the Identity API served through keystone.
export OS_AUTH_URL=http://10.0.255.255/identity
# With the addition of Keystone we have standardized on the term **project**
# as the entity that owns the resources.
export OS_PROJECT_ID=abcdefghijklmnopqrstuvwxyz012345
export OS_PROJECT_NAME="admin"
export OS_USER_DOMAIN_NAME="Default"
if [ -z "$OS_USER_DOMAIN_NAME" ]; then unset OS_USER_DOMAIN_NAME; fi
export OS_PROJECT_DOMAIN_ID="default"
if [ -z "$OS_PROJECT_DOMAIN_ID" ]; then unset OS_PROJECT_DOMAIN_ID; fi
# unset v2.0 items in case set
unset OS_TENANT_ID
unset OS_TENANT_NAME
# In addition to the owning entity (tenant), OpenStack stores the entity
# performing the action as the **user**.
export OS_USERNAME="admin"
# With Keystone you pass the keystone password.

# The two next lines are the one that need to be commented out or deleted
```

(continues on next page)

(continued from previous page)

```
# echo "Please enter your OpenStack Password for project $OS_PROJECT_NAME as user $OS_
↪USERNAME: "
# read -sr OS_PASSWORD_INPUT

# Add this line with your Openstack password
OS_PASSWORD_INPUT=<password>

export OS_PASSWORD=$OS_PASSWORD_INPUT
# If your configuration has multiple regions, we set that information here.
# OS_REGION_NAME is optional and only valid in certain environments.
export OS_REGION_NAME="RegionOne"
# Don't leave a blank variable, unset it if it was empty
if [ -z "$OS_REGION_NAME" ]; then unset OS_REGION_NAME; fi
export OS_INTERFACE=public
export OS_IDENTITY_API_VERSION=3
```

3. Copy or symlink the file into `/opt/bacula/etc/admin_openrc.sh` or provide the relevant value for `admin_openrc` plugin parameter.
4. Advise backup service to use the Cinder bacula driver.

   To do so, Cinder configuration file located by default at `/etc/cinder/cinder.conf` has to be edited.
5. Inside the `[DEFAULT]` group, the line `backup_driver = cinder.backup.drivers.bacula.BaculaBackupDriver` need to be added.

```
[DEFAULT]
...
backup_driver = cinder.backup.drivers.bacula.BaculaBackupDriver
...
```

See also:

- *Scope*
- *Features*
- *Architecture*
- *Installation*
- *Operations*
- *Backup and Restore Strategies*
- *Troubleshooting*
- *Limitations*

Go back to the *main Openstack VM page*.

6 Operations

The following article describes details regarding backup, restore or query operations with Bacula Enterprise Openstack VM Plugin.

6.1 Bacula Enterprise Openstack Procedures

The Bacula Enterprise Openstack plugin has its own set of procedures to interact with the Openstack environment.

The user should only interact with procedures that contains the keyword `execute` in their name with the exception of the `openstack-vm-query`.

- `openstack-vm-execute-backup` to instance's volume(s) backup.
- `openstack-vm-execute-restore` to instance's volume(s) restore.
- `openstack-vm-execute-interactive-delete` to delete backups, snapshots or volumes.
- `openstack-vm-query` to get more information about Openstack resources.

All these procedures have their own dedicated chapter in this Operation section.

Backup in Openstack VM

Execute backup for a specific instance by running the `/opt/bacula/bin/openstack-vm-execute-backup` procedure with relevant parameters.

Parameters

- `-B <instance-name>` - If this parameter is set, the procedure will list the backup related for this instance.
- `-b <instance-name>` - If this parameter is set, the procedure will try to backup all volume of an instance named <instance-name>.
- `-c <admin-openrc>` - Path to `admin-openrc.sh` file. Default value is `/opt/bacula/etc/admin-openrc.sh`.
- `-i` - If this parameter is set, the backup will be incremental.
- `-t <tools>` - Path to Openstack procedures. Default value is `/opt/bacula/bin/`.
- `-v <instance-id>` - ID of the instance to backup.
- `-w <wainting-time>` - Waiting time between two poll operations. Default value is `5`.
- `-h` - Display help.

Note: Option `-v` has precedence over option `-b`.

Example

Backup of all the volumes attached to a specific instance, using the instance ID:

Note: To get either the ID or the name of a specific instance, the query procedure can be used with `/opt/bacula/bin/openstack-vm-query -l`. The ID can be found under the ID column.

First get the relevant instance id:

```
root@stackdev:/opt/bacula# bin/openstack-vm-query -l
+--------------------------------------+----------------+--------+----------------------
↪-----------------------------------------------------------+---------------------------+--
↪--------+
| ID                                   | Name           | Status | Networks            ␣
↪                                                            | Image                     |␣
↪Flavor   |
+--------------------------------------+----------------+--------+----------------------
↪-----------------------------------------------------------+---------------------------+--
↪--------+
| instance_ID                          | instance_name  | ACTIVE | private=00.0.0.0,␣
↪1111:1111:1111:0:1111:1111:1111:1111; shared=111.111.111.111 | N/A (booted from␣
↪volume) | m1.micro |
+--------------------------------------+-------+--------+-------------------------------
↪-------------------------------------------------+-------------------------+----------+
```

Then issue backup creation operation:

```
root@openstack-bck:~# /opt/bacula/bin/openstack-vm-execute-backup -v instance_ID
Backup of VM=<instance_ID>  INCREMENTAL=False       ADMIN_OPENRC=/opt/bacula/etc/admin-
↪openrc.sh    SCRIPTS=/opt/bacula/bin/
I: Found 2 volumes to backup for <instance_ID>
I: Backing up <volume_ID>
D: Issue snapshot
I: Snapshot done
I: Backup status=creating
[...]
I: Backup status=creating
I: Backup status=creating
I: Backup status=available
I: Done proceeds to next
I: Issue delete command for snapshot=<snapshot_ID>
I: Backing up <volume_ID>
D: Issue snapshot
D: Snapshot creating ...
I: Snapshot done
I: Backup status=creating
[...]
I: Backup status=creating
I: Backup status=creating
I: Backup status=available
I: Done proceeds to next
I: Issue delete command for snapshot=<snapshot_ID>
I: No more volumes to process END OF BACKUP
```

Backup of all the volumes attached to a specific instance, using the instance name:

```
root@host:/opt/bacula# /opt/bacula/bin/openstack-vm-execute-backup -b instance_name
Backup of VM=<instance_ID>  INCREMENTAL=False       ADMIN_OPENRC=/opt/bacula/etc/admin-
↪openrc.sh    SCRIPTS=/opt/bacula/bin/
I: Found 2 volumes to backup for <instance_ID>
I: Backing up <volume_ID>
D: Issue snapshot
I: Snapshot done
I: Backup status=creating
[...]
I: Backup status=creating
I: Backup status=creating
I: Backup status=available
I: Done proceeds to next
I: Issue delete command for snapshot=<snapshot_ID>
I: Backing up <volume_ID>
D: Issue snapshot
D: Snapshot creating ...
I: Snapshot done
I: Backup status=creating
[...]
I: Backup status=creating
I: Backup status=creating
I: Backup status=available
I: Done proceeds to next
I: Issue delete command for snapshot=<snapshot_ID>
I: No more volumes to process END OF BACKUP
```

When using the -b or -v options, to backup an OpenStack instance, there will be one backup jobid in Bacula for each volume attached to the instance. Also, there will be one backup in the OpenStack server for each volume.

```
root@host:/opt/bacula# /opt/bacula/bin/openstack-vm-query -b
+---------------+----------------------+------------------------------------------+------
↪-----+------+-------------+
| ID            | Name                 | Description                              |␣
↪Status    | Size | Incremental |
+---------------+----------------------+------------------------------------------+------
↪-----+------+-------------+
| <backup1_ID>> | <backup1_name>       | Backup done by Bacula Enterprise         |␣
↪available |   10 | False       |
|                                      | INSTANCE=<instance_name> DATE=<datetime> |     ␣
↪     |      |             |
|                                      |                                          |     ␣
↪     |      |             |
| <backup2_ID>> | <backup2_name>       | Backup done by Bacula Enterprise         |␣
↪available |    5 | False       |
|                                      | INSTANCE=<instance_name> DATE=<datetime> |     ␣
↪     |      |             |
+---------------+----------------------+------------------------------------------+------
↪-----+------+-------------+
```

Also, in the Catalog, two jobids are created:

```
|     xx | job.openstack-bck-fd.openstack-vm | 2024-01-01 12:00:00 | B     | I      |       ␣
↪ 6 | 12,345,678 | T          |
|     XX | job.openstack-bck-fd.openstack-vm | 2024-01-01 12:00:05 | B     | I      |       ␣
↪ 9 | 90,123,456 | T          |
```

Backup Job Example with a RunScript Block

As mentioned earlier, it is possible to define a backup job to trigger the openstack-vm-execute-backup program to execute the backup in the OpenStack server.

The RunScript block below triggers the openstack-vm-execute-backup program to backup all the volumes attached to the *MyInstance* instance in the OpenStack server, having the *openstack-bck-fd* bacula client installed:

```
Job {
  Name = OpenStack-test-job
  JobDefs = BackupsToDisk
  FileSet = None
  Client = openstack-bck-fd
  RunScript {
    Command = "/opt/bacula/bin/openstack-vm-execute-backup -b MyInstance"
    RunsOnClient = yes
    RunsWhen = Before
  }
}

FileSet {
  Name = None
  EnableVSS = no
}
```

See also:

- *Restore in Openstack VM*
- *Query*
- *Interactive Delete*

Go back to the *main Openstack VM page*.

Restore in Openstack VM

Restore volumes attached to a previous backup or a specific volume by running the `/opt/bacula/bin/openstack-vm-execute-restore` procedure with relevant parameters.

Note: It is not possible to perform a restore of an OpenStack instance volume(s) by using the bconsole restore command or a Bacula Graphical Interface. The `/opt/bacula/bin/openstack-vm-execute-restore` program available in the OpenStack server must be used for restores.

Parameters

- `-b <backup_id>` - ID of a specific volume backup to restore.
- `-c <admin-openrc>` - Path to modified admin-openrc.sh Default value is `/opt/bacula/etc/admin-openrc.sh`.
- `-n <backup_name>` - If this parameter is set, the volume with this name will be restored.
- `-t <tools>` - Path to Openstack procedure. Default value is `/opt/bacula/bin/`.
- `-v <instance_id>` - ID of the instance to restore.
- `-w <waitingTime>` - Waiting time in seconds between two completion check. Default value is `5`.
- `-h` - Display help.

Example

To restore all the volumes from an instance.

Note: To get the ID of a specific instance, the query procedure can be used with `/opt/bacula/bin/openstack-vm-query -l`. The ID can be found under the ID column.

In case the virtual machine was deleted beforehand Cinder backups created by the plugin will have the original virtual machine ID as a name. To access the list of backup the query procedure can be used with `/opt/bacula/bin/openstack-vm-query -b`.

Restore using the instance ID. Get the instance_ID using the `/opt/bacula/bin/openstack-vm-query -l` command:

```
root@stackdev:/opt/bacula# bin/openstack-vm-query -l
+--------------------------------------+----------------+--------+-----------------------
↪---------------------------------------------------------+--------------------------+--
↪--------+
| ID                                   | Name           | Status | Networks             ␣
↪                                                           | Image                    |␣
↪Flavor   |
+--------------------------------------+----------------+--------+-----------------------
↪---------------------------------------------------------+--------------------------+--
↪--------+
| instance_ID                          | instance_name  | ACTIVE | private=00.0.0.0,␣
↪1111:1111:1111:0:1111:1111:1111:1111; shared=111.111.111.111 | N/A (booted from␣
↪volume) | m1.micro |
+--------------------------------------+-------+--------+--------------------------------
↪------------------------------------------------+--------------------------+----------+
```

Then issue the restore command using the instance_ID value:

```
root@stackdev:~# /opt/bacula/bin/openstack-vm-execute-restore -v <instance_ID>
Restore of INSTANCE_ID=<instance_ID>                SCRIPT_PATH=/opt/bacula/bin/   ␣
↪ADMIN_OPENRC=/opt/bacula/etc/admin-openrc.sh
I: 1 backup to restore
I: Restoring <backup1_ID>
I: Volume restoration in progress=restoring-backup
```

(continues on next page)

(continued from previous page)

```
I: Restored volume found with ID=<restored_volume_ID>
I: Volume restoration in progress=restoring-backup

...

I: Volume restoration in progress=restoring-backup
I: Volume restoration in progress=available
I: Done moving on to next
I: No more backup to restore END
```

Once this procedure is done, the volumes will be in an *available* status, and you will need to either create a new instance and attach the restored volumes, or to attach the restored volumes to an existent instance using the openstack CLI or with the GUI from the dashboard.

It is also possible to restore a single volume from a backup id by using the `-b` option.

Get the backup_ID using the `/opt/bacula/bin/openstack-vm-query -b` command:

```
root@host:/opt/bacula# /opt/bacula/bin/openstack-vm-query -b
+---------------+----------------------+-------------------------------------------+------
↪-----+------+-------------+
| ID            | Name                 | Description                               |␣
↪Status    | Size | Incremental |
+---------------+----------------------+-------------------------------------------+------
↪-----+------+-------------+
| <backup1_ID> | <backup1_name>       | Backup done by Bacula Enterprise          |␣
↪available |   10 | False       |
|                                      | INSTANCE=<instance_name> DATE=<datetime> |      ␣
↪     |      |             |
|                                      |                                           |      ␣
↪     |      |             |
| <backup2_ID> | <backup2_name>       | Backup done by Bacula Enterprise          |␣
↪available |    5 | False       |
|                                      | INSTANCE=<instance_name> DATE=<datetime> |      ␣
↪     |      |             |
+---------------+----------------------+-------------------------------------------+------
↪-----+------+-------------+
```

Then issue the restore command using the backup ID value, for example:

```
root@stackdev:~# /opt/bacula/bin/openstack-vm-execute-restore -b <backup1_ID>
Restore of INSTANCE_ID=<instance_ID>                SCRIPT_PATH=/opt/bacula/bin/   ␣
↪ADMIN_OPENRC=/opt/bacula/etc/admin-openrc.sh
I: 1 backup to restore
I: Restoring <backup1_ID>
I: Volume restoration in progress=restoring-backup
I: Restored volume found with ID=<restored_volume_ID>
I: Volume restoration in progress=restoring-backup

...

I: Volume restoration in progress=restoring-backup
I: Volume restoration in progress=available
```

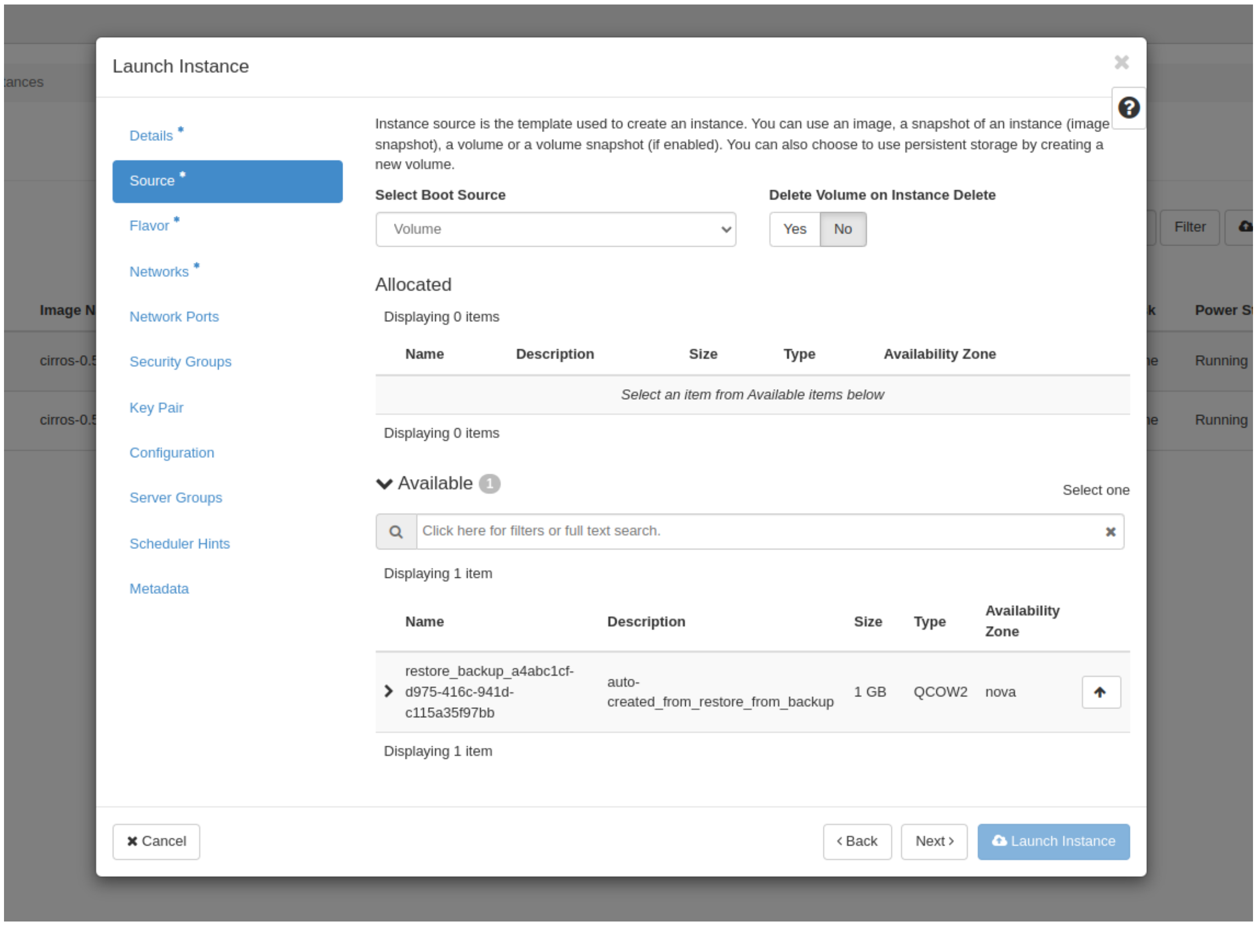
(continues on next page)

(continued from previous page)

```
I: Done moving on to next
I: No more backup to restore END
```

Create instance from GUI with restored disk

- Under the Instances overview select the `Launch Instance` menu.
- In the source menu select `Volume` from `Select Boot Source`
- If the disk containing the operating system has to be restored. Under the `Available` section select the newly restored disk by hitting the up arrow sign on the right



Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.

Select Boot Source: Volume

Delete Volume on Instance Delete: Yes [✓ No]

Allocated

Displaying 0 items

Name	Description	Size	Type	Availability Zone
Select an item from Available items below				

Displaying 0 items

Available 1 — Select one

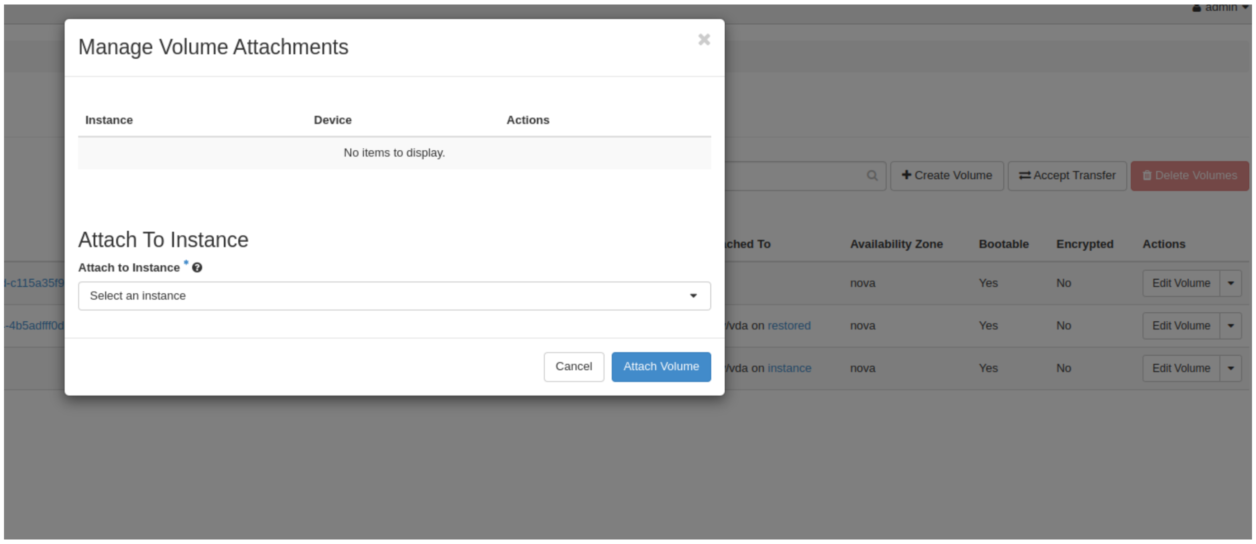
Click here for filters or full text search.

Displaying 1 item

Name	Description	Size	Type	Availability Zone
restore_backup_a4abc1cf-d975-416c-941d-c115a35f97bb	auto-created_from_restore_from_backup	1 GB	QCOW2	nova

Displaying 1 item

- Setup all other parameters, preferably with the same flavor as the backup instance.
- Launch instance
- Manually attach restored data disks to newly created instance via `Volumes` menu
- On the restored disk the `Edit volume` menu contains a `Manage Volume Attachments` section
- In the `Attach to instance` sub-menu select the relevant instance to attach the disk to



See also:

- *Backup in Openstack VM*
- *Query*
- *Interactive Delete*

Go back to the *main Openstack VM page*.

Query

Display different information about backup, snapshot, instance and/or volumes by running the `/opt/bacula/bin/openstack-vm-query` with relevant parameters.

Parameters

- `-b` Lists backups
- `-c <admin_openrc>` Path to admin-openrc.sh
- `-f <format>` Format the output in one of the following format: json, table, value, yaml
- `-l` Lists instances
- `-p` Check if `Cinder-backup` is running
- `-q` Check if `Cinder` module is installed
- `-s` List snapshots
- `-v` Lists volumes
- `-V` Verbose output for `-p` and `-q` options
- `-h` Display help

Example

The query procedure is used to list different resources in a defined format.

`/opt/bacula/bin/openstack-vm-query` is the base command.

To list the instances available in the OpenStack server:

To list the backups performed:

Listing volumes and backups in a json format would result in:

See also:

- *Backup in Openstack VM*
- *Restore in Openstack VM*
- *Interactive Delete*

Go back to the *main Openstack VM page*.

Interactive Delete

It is possible to delete backup(s), snapshot(s) or volume(s) by running the `/opt/bacula/bin/openstack-vm-execute-interactive-delete` procedure with relevant parameters

Note: The delete operation is sent through Openstack API, and it has the force flag activated by default.

Parameters

- `-b <backup-ID>`: For interactive backup deletion if no ID is specified the procedure will go through all backups asking for deletion.
- `-c <admin-openrc>` Path to modified admin-openrc.sh DEFAULT=/opt/bacula/admin-openrc.sh.
- `-s <snapshot-ID>` For interactive snapshot deletion if no ID is specified the procedure will go through all snapshots asking for deletion.
- `-t <tools>` Path to openstack-vm-scripts DEFAULT=/opt/bacula/bin/
- `-v <volume-ID>` For interactive volume deletion if no ID is specified the procedure will go through all volumes asking for deletion.
- `-h` Display help

Example

The options of this procedure are analog to other operations, but used for backup/snapshot/volume deletion instead.

Interactive delete of backups would be `/opt/bacula/bin/openstack-vm-execute-interactive-delete -b`

With an output looking like this:

```
root@openstack-bck:~# /opt/bacula/bin/openstack-vm-execute-interactive-delete -b
INTERACTIVE BACKUP DELETE START

Would you like to delete BACKUP
ID=<backup1_ID>
NAME=<backup1_name>
DESCRIPTION=Backup done by Bacula Enterprise INSTANCE=<instance_name> DATE=Tue Mar 19␣
↪16:34:03 UTC 2024

Start deletion [y]es / [N]o ?y
Delete command sent


Would you like to delete BACKUP
ID=<backup2_ID>
NAME=<backup2_name>
DESCRIPTION=Backup done by Bacula Enterprise INSTANCE=<instance_name> DATE=Tue Mar 19␣
↪16:34:03 UTC 2024

Start deletion [y]es / [N]o ?y
Delete command sent

...

INTERACTIVE BACKUP DELETE FINISHED
```

See also:

- *Backup in Openstack VM*
- *Restore in Openstack VM*
- *Interactive Delete*

Go back to the *main Openstack VM page*.

Warning: If the user wants to interact with other procedures than the ones listed above it must be done with extreme caution and contact Bacula Enterprise support beforehand.

When running a backup or a restore using the OpenStack VM Plugin, the plugin procedures will need to contact the Bacula Enterprise Director in order to run backup and restore jobs. This communication involves the Cinder Bacula Driver, and the bconsole program.

The Bacula console *bconsole* will be installed along with the OpenStack VM Plugin, and the console should be able to connect to your Director and have access to the local Client, the backup Job and other Bacula resources. These requirements are explained in the *Installation* section.

When using the OpenStack VM plugin, it is not possible to manually start a backup or restore using a Bacula Enterprise User Interface, bconsole or BWeb.

Backups or restores must be initiated by using the OpenStack VM plugin set of procedures *openstack-vm-**. However, it is always possible to run a regular backup of your OpenStack server, and then, use a RunScript block to trigger the execution of the OpenStack VM plugin backup procedure. This will allow you to have a regular schedule defined to backup your OpenStack server, along with Instances volumes. An example of a backup job using a RunScript block to trigger an OpenStack instance volumes backup is described in the *Backup in Openstack VM* section.

See also:

- *Scope*
- *Features*
- *Architecture*
- *Installation*
- *Configuration*
- *Backup and Restore Strategies*
- *Troubleshooting*
- *Limitations*

Go back to the *main Openstack VM page*.

7 Backup and Restore Strategies

7.1 Installing Bacula Client on Each Guest

This strategy works by installing a Bacula Enterprise File Daemon on every virtual machine as if they were regular, physical clients. In order to optimize the I/O usage of Openstack, the user will use Bacula's Schedules, Priorities, and Maximum Concurrent Jobs to spread backup jobs over the backup window. Since all VMs could use the same storage on the Openstack hypervisor, running all backup jobs at the same time could create a bottleneck on the disk/network subsystem since Bacula will walk through all filesystems to open/read/close/stat files.

Installing the Bacula Enterprise File Daemon on each virtual machine permits to manage virtual servers like physical servers and also to use all Bacula Enterprise's features such as:

- Quick restores of individual files
- Checksum of individual files for Virus and Spyware detection
- Verify Jobs
- File/Directory exclusion (such as swap or temporary files)
- File level compression
- Accurate backups.

7.2 Cinder Driver Backup with Openstack Plugin

With the Cinder driver strategy, the Bacula Enterprise Openstack-VM will save all Openstack volume`s at the raw level, in the Openstack context.

Bacula's Openstack-VM plugin will read and save the content of Openstack instance using Cinder backup API.

Cinder allows the user to integrate various storage solutions into the Openstack cloud. It does this by providing a stable interface for hardware providers to write drivers that allow the usage of Cinder volumes backup capabilities.

See also:

- *Scope*
- *Features*
- *Architecture*

- *Installation*
- *Configuration*
- *Operations*
- *Troubleshooting*
- *Limitations*

Go back to the *main Openstack VM page*.

8 Troubleshooting

This article presents recommended solutions for common issues that may arise while using the Openstack VM Plugin.

- `D: cannot unpack non-iterable VolumeBackupsRestore object`

At restore time the restore volume command might output the following message `D: cannot unpack non-iterable VolumeBackupsRestore object`. This issue shouldn't impact the restore process and it can be ignored.

- `W: Openstack returned too many values to unpack (expected 2)`

At restore time Openstack might output a warning `W: Openstack returned too many values to unpack (expected 2)`. Restore should go through regardless and not be impacted by the message. This issue also happens when using the OpenStack CLI, and a bug report has been reported to the Openstack team.

See also:

- *Scope*
- *Features*
- *Architecture*
- *Installation*
- *Configuration*
- *Operations*
- *Backup and Restore Strategies*
- *Limitations*

Go back to the *main Openstack VM page*.

9 Limitations

The following article presents limitations of Openstack VM Plugin.

- after a restore-procedure only the volumes are restored. The specific restored instance must be manually

restored and by attaching the relevant volumes to a new instance.

- currently, only full level instance(s) volume(s) backups are possible.

See also:

- *Scope*
- *Features*
- *Architecture*

- *Installation*
- *Configuration*
- *Operations*
- *Backup and Restore Strategies*
- *Troubleshooting*

Go back to the *main Openstack VM page*.