



# **RHV Plugin**

**Bacula Systems Documentation**

---

# Contents

<b>1</b>	<b>Scope</b>	<b>3</b>
<b>2</b>	<b>Red Hat Virtualization Technology</b>	<b>4</b>
2.1	Ovirt . . . . .	4
2.2	Architecture . . . . .	4
2.3	Template . . . . .	7
2.4	Storage Domain . . . . .	7
2.5	Data Warehouse . . . . .	8
2.6	Networking . . . . .	8
2.7	Red Hat Virtualization APIs . . . . .	8
2.8	Connection Modes . . . . .	8
2.9	More Information . . . . .	8
<b>3</b>	<b>Architecture and Design</b>	<b>9</b>
3.1	Backup Process . . . . .	9
3.2	Restore Process . . . . .	15
<b>4</b>	<b>Features</b>	<b>15</b>
4.1	Backup Features . . . . .	15
4.2	Restore Features . . . . .	16
<b>5</b>	<b>Limitations</b>	<b>16</b>
<b>6</b>	<b>Compatibility and Requirements</b>	<b>17</b>
6.1	Compatibility . . . . .	17
6.2	Requirements . . . . .	17
6.3	Use of computer resources . . . . .	18
<b>7</b>	<b>Configuration</b>	<b>18</b>
7.1	Plugin Installation . . . . .	18
7.2	Certificate and Truststore . . . . .	19
7.3	User Permissions . . . . .	21
<b>8</b>	<b>Plugin Parameters</b>	<b>22</b>
8.1	General Plugin Parameters . . . . .	23
8.2	Tuning Parameters . . . . .	24
8.3	Backup-Specific Parameters . . . . .	25
8.4	Restore-Specific Parameters . . . . .	26
8.5	System-Specific Parameters . . . . .	27
8.6	Configuration File . . . . .	27
<b>9</b>	<b>Fileset Configuration</b>	<b>27</b>
9.1	Backup Configuration Examples . . . . .	27
<b>10</b>	<b>Restore</b>	<b>30</b>
10.1	File-Level Restore . . . . .	34
<b>11</b>	<b>Diagnostic Operations</b>	<b>35</b>
11.1	Querying RHV . . . . .	35
11.2	System operations . . . . .	38
11.3	Transferring Data . . . . .	39
<b>12</b>	<b>Troubleshooting</b>	<b>40</b>

12.1 Bacula environment . . . . .	40
12.2 RHV environment . . . . .	40
12.3 javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException . . . . .	41
12.4 VDSM hostTest command GetCapabilitiesVDS failed: Vds timeout occurred . . . . .	43
<b>13 Future Development Directions</b>	<b>43</b>
<b>14 RHV Single Item Restore</b>	<b>43</b>
14.1 Features Summary . . . . .	44
14.2 Installation . . . . .	44
14.3 Notes about the “bacula” Account on RHEL . . . . .	45
14.4 Fuse FileSystem . . . . .	46
14.5 Samba SMB Shares . . . . .	46
14.6 Configuration . . . . .	47
14.7 Restore Scenario With Text Console Interface . . . . .	47
14.8 Notes . . . . .	49
14.9 Limitations . . . . .	49

## Contents

---

### Note

You can download this article as a [PDF](#)

### Enterprise

#### Bacula Enterprise Only

This solution is **only** available for Bacula Enterprise. For subscription inquiries, please reach out to [sales@baculasystems.com](mailto:sales@baculasystems.com).

#### Important

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

## 1 Scope

IT organizations are constantly being challenged to deliver high quality solutions with reduced total cost of ownership. Virtualization technologies are an important and widely used element of the resulting strategies and accordingly found in most data centers.

Naturally, reliable solutions to protect the underlying data are a critical topic. This chapter presents the Bacula Enterprise plugin and strategy to protect Red Hat Virtualization environments.

The plugin provides several ways to backup virtual machines, image level with a full backup or proxy backup, incremental or differential backup, with a set of options to select different backup sets. To later, recovery them with a set of options to customize the restore process.

## 2 Red Hat Virtualization Technology

Red Hat Virtualization (RHV) is a full virtualization platform capable of managing the full set of features that make up a virtualized data center: Hosts, networks, local, remote, or distributed storage. Management, accounting and monitoring are done through an API-based Web interface, which implements users and roles to implement access restrictions.

### 2.1 Ovirt

RHV is based on the open source Ovirt virtualization platform. Ovirt is available for Linux distributions related to RHEL. Ovirt is under active development, and RHEL integrates releases they consider mature. Therefore, Ovirt itself can typically exist in newer versions than what is part of RHV.

### 2.2 Architecture

RHV is composed of the following elements:

#### **RHV Manager**

is the service offering management through a web interface and APIs.

#### **Virtualization Hosts**

are the hypervisors providing their resources to the virtual machines. They are managed through the RHV Manager, and are based on the KVM system.

Two kinds of hosts are available:

#### **RHEL hypervisors**

being standard RHEL server systems

#### **oVirt nodes**

are minimal servers which are distributed as ISO images to install a lean virtualization platform solely for RHV usage.

There are 2 ways to deploy the RHV Manager: Standalone, with the RHV Manager running on a host outside of the virtualized environment, which does not provide native high availability, but is easily deployed and managed, and self-hosted, where RHV Manager runs inside the virtualized environment, in a dedicated VM.

The latter approach can be a highly available service by making use of the `ovirt-ha-agent` and `virt-ha-broker` services.

In figure *RHV Host Architecture*, the architecture of a virtualization host is shown. The stack of technologies employed by a host to provide a VM consists of the following elements:

#### **VDSM**

is the host agent service running on every virtualization host to provide the hypervisor service to the RHV system. This services listens on the TCP port 54321 on the network.

#### **libvirt**

is the toolkit that manages virtual machines.

#### **QEMU**

is the multi-platform emulator which provides emulation of full systems, including CPU capabilities not available on the underlying system.

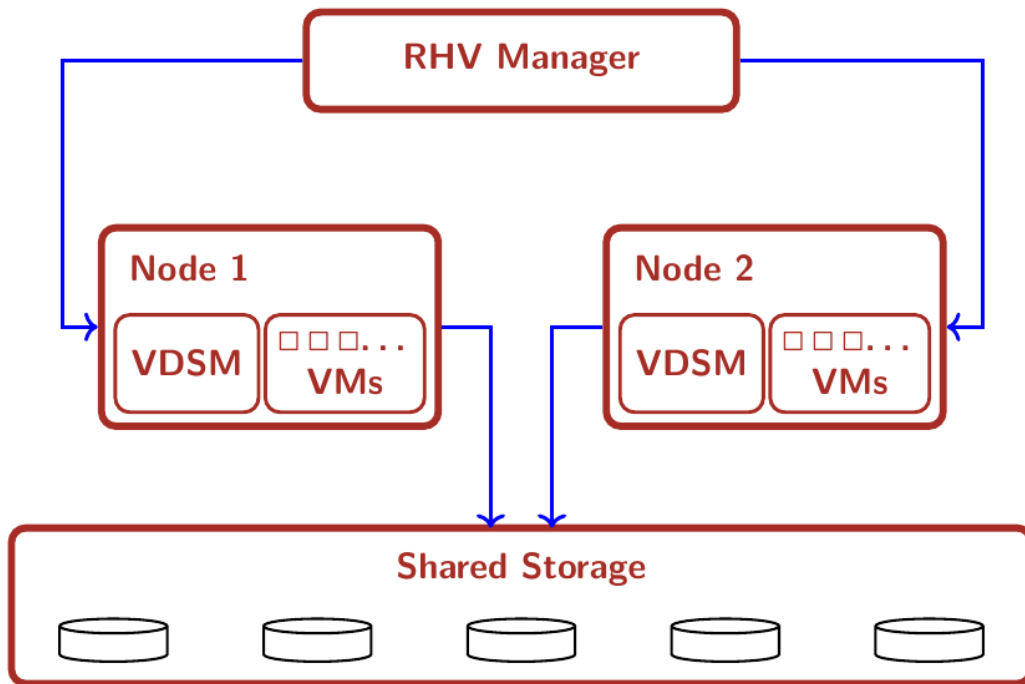


Fig. 1: General Architecture (Standalone mode)

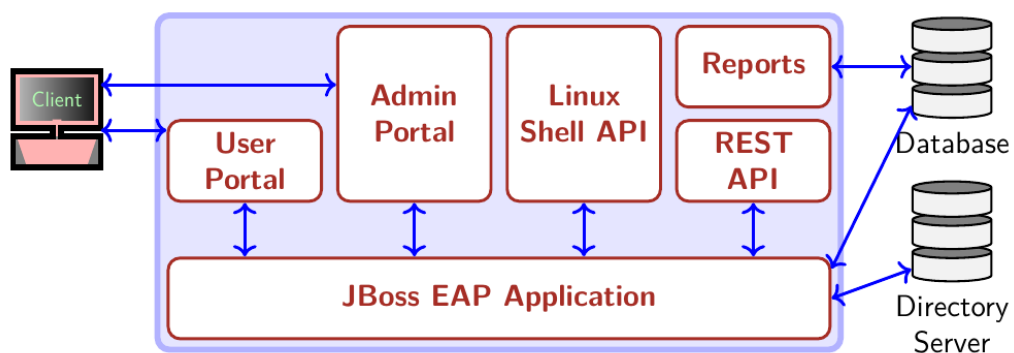


Fig. 2: RHV Manager Architecture

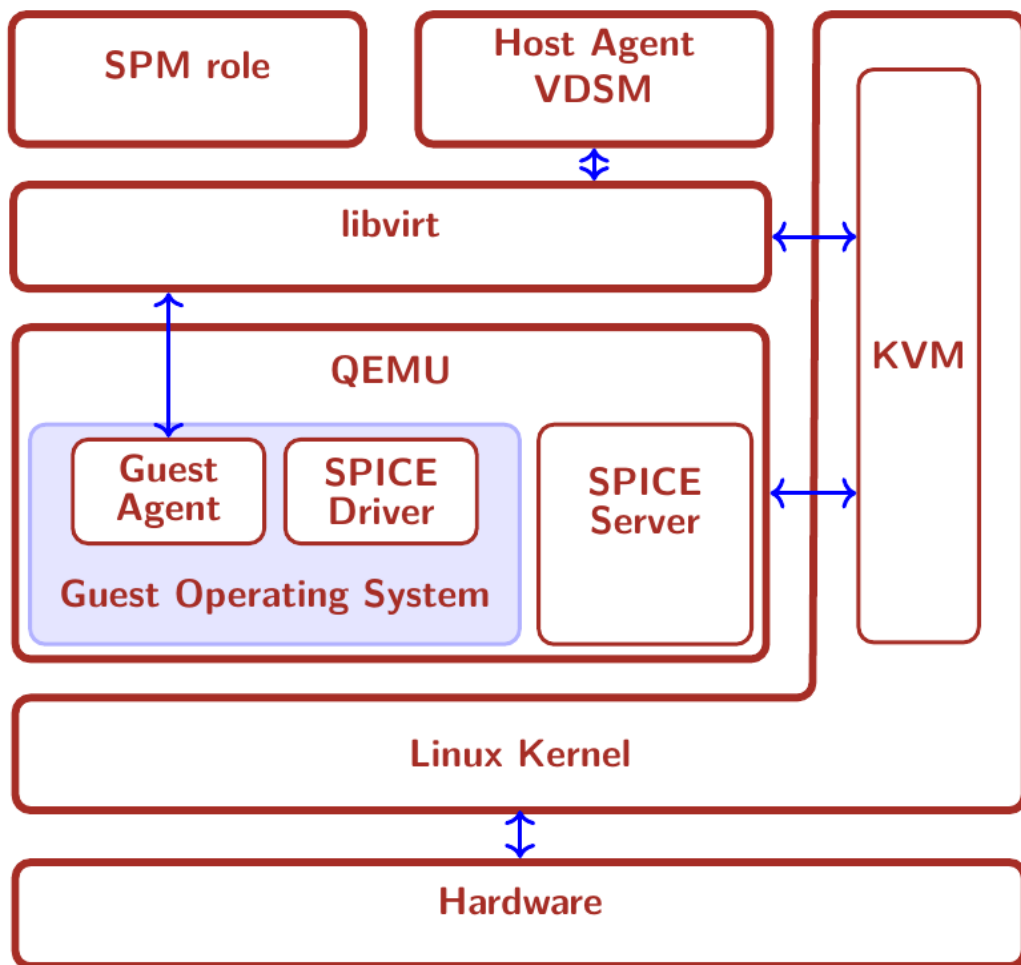


Fig. 3: RHV Host Architecture

**KVM**

is the kernel module which provides the system integration component to QEMU. It allows to execute guest VMs in user space, ensuring full segregation of VMs from each other and the host system.

**SPICE**

is the interface between virtualized user-facing components and remote interfaces, providing input, display, and sound services to remote interfaces (the “viewer” program).

Other important concepts to understand a RHV system are

**Datacenter**

is the most high-level group of virtualization systems, the VMs managed in it, and all the resources available.

**Cluster**

refers to a group of virtualization hosts (sharing networks, storage, and some other infrastructure properties). A Cluster is always part of a distinct Datacenter, and a Datacenter can consist of more than one Cluster.

**Storage Domain**

indicates a logical entity providing storage capacity to Virtual Machines of a Datacenter.

**Storage Pool Manager**

is the role of a certain host in the datacenter which creates, manages, and removes virtual disk images.

**Template**

is the base configuration of vms. This configuration can be from the architecture of the processor that will use the vm to the disks attached to it.

**Host**

is a physical server on which the virtual machines run. Alternatively, host is also called hypervisor. Both RHEL Virtualization Hypervisors and RHEL hosts interact with the rest of the virtualized environment in the same way.

## 2.3 Template

The templates have the ability to create virtual machines quickly and conveniently. To create a template we need a vm to copy its configuration, its networks and content of its disks.

The templates have a specific configuration and unique content on the disks. Once the template is created the content of the disks is immutable and RHV does not allow to add or remove disks. However, it is possible to modify, add or modify the networks attached to the template.

## 2.4 Storage Domain

Storage Domains are distinct storage subsystems used to host different kinds of information.

Two Storage Domains are used by RHV:

**The Data Domain**

stores virtual hard disks of VMs and templates in a Data Center. Data Domains are exclusive to one Data Center.

Data Domains can be backed by different storage attachment technologies, such as NFS, Fibre Channel, FCoE, GlusterFS, Ceph, iSCSI, and any (local) POSIX-compliant file system.

**The ISO Domain**

hosts images of media such as CDs and DVDs to deploy software on VMs. There can only be one

ISO Domain in a Data Center, an ISO Domain can be shared among Data Centers, and NFS is the only available backing store for this type of storage domain.

## 2.5 Data Warehouse

RHV employs a database of historical data where all the management and auditing data is stored. This function is provided by the `ovirt_engine_history` service. It makes use of the PostgreSQL database for storing its data.

## 2.6 Networking

RHV allows the definition of different logical networks to isolate traffic types and paths. These networks are created, maintained, and destroyed by the RHV Manager, which also handles VLANs, routing, and firewalling.

Usually, there will be several networks in use in a RHV Datacenter. Some pre-defined network types are available to segregate different types of traffic: A management network which should be used exclusively by RHV management communications, VM networks for use by the virtual machines, storage networks for example for iSCSI or NFS traffic, and storage migration networks.

Networks for dedicated purposes can be created as needed, and they can be bound to specific network hardware (and VLANs) available on the virtualization hosts.

## 2.7 Red Hat Virtualization APIs

RHV provides different APIs to access its functionality:

- Shell: 4.1 4.2
- REST API: 4.1 4.2 4.3
- SDKs (based on the REST API)
  - Java: 4.1 4.2 4.3
  - Python: 4.1 4.2 4.3
  - Ruby: 4.1 4.2 4.3

The Bacula Enterprise Red Hat Virtualization Plugin is based on the Java SDK 4.3

## 2.8 Connection Modes

RHV allows 2 different authentication schemes:

- OAuth Authentication
- HTTP Basic Authentication

The plugin supports both [connection](#) and [authentication modes](#).

## 2.9 More Information

The information of this chapter is based upon the official RHEL Documentation available in the following links:

### General Documentation

[redhat.com/.../red-hat-virtualization/](https://redhat.com/.../red-hat-virtualization/)

### Version Information

[redhat.com/.../updates/rhev](https://redhat.com/.../updates/rhev)

**oVirt**

<https://ovirt.org/>

## 3 Architecture and Design

The Red Hat Virtualization Plugin is a File Daemon plugin. It may be installed on a machine inside or outside of your RHV environment. Is based on a Java SDK, therefore is non-machine dependent and is compatible with any Operating System where the Bacula File Daemon can run and the Java Virtual Machine is available.

### 3.1 Backup Process

The backup process allows you to perform in two different environments: external or internal. The external process allows full clone backups or snapshots, and incremental or differential backups. The internal process allows full backups to be performed faster than the external processes.

#### **Backup from RHV environment external machine**

The next backups methods (clone, snapshot, template) perform a download of the VM's disks through the API. These methods are slower than the 'Proxy VM' method, but they do not depend on a virtual machine in the RHV environment.

The general backup process of a virtual machine is as follows (check also Figure *External backup diagram*):

1. Check for the existence of the virtual machine to protect and the compatibility of the RHV environment with the chosen backup method (check Section *Limitations*).
2. If it is not a template backup:
  - Check the snapshots:
    - If the last backup was incomplete and the generated snapshot was not removed successfully.
    - If apply a full backup in a virtual machine with incremental snapshots.
  - Launch a snapshot.
3. If it is a clone backup:
  - Clone the virtual machine using the created snapshot.
  - Download configuration of the original virtual machine and cloned machine (XML).
  - Download all disks of cloned machine.
4. If it is a template backup:
  - Download configuration of template (XML).
  - Download all disks of template.
5. If it is a snapshot backup:
  - Download configuration of virtual machine (XML).
  - Download all virtual machine snapshot disks.
6. If it is a backup clone:
  - If necessary, remove clone machine.
7. If it is not a template backup and **cbt is off**:

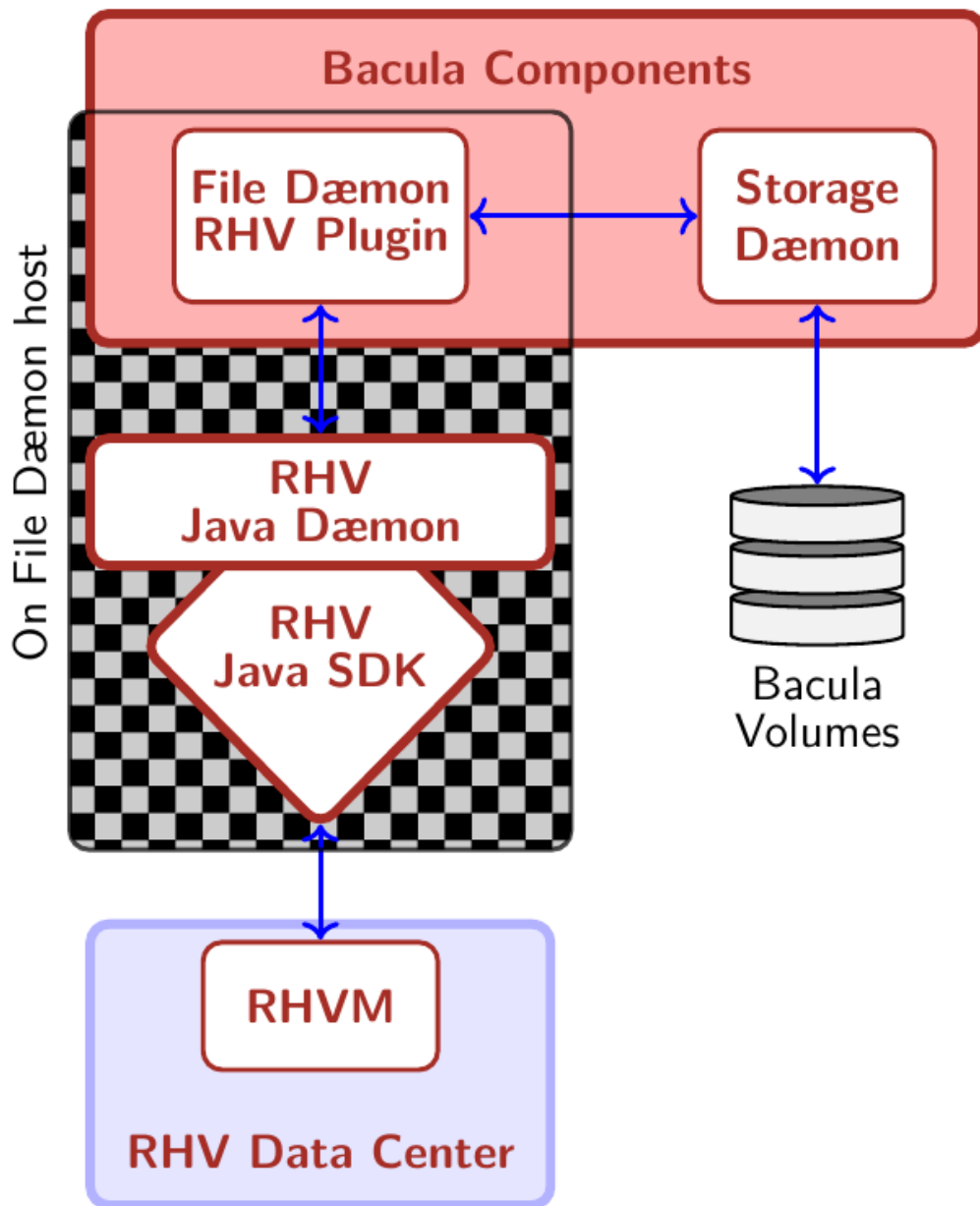


Fig. 4: Plugin Architecture

- Remove snapshot.

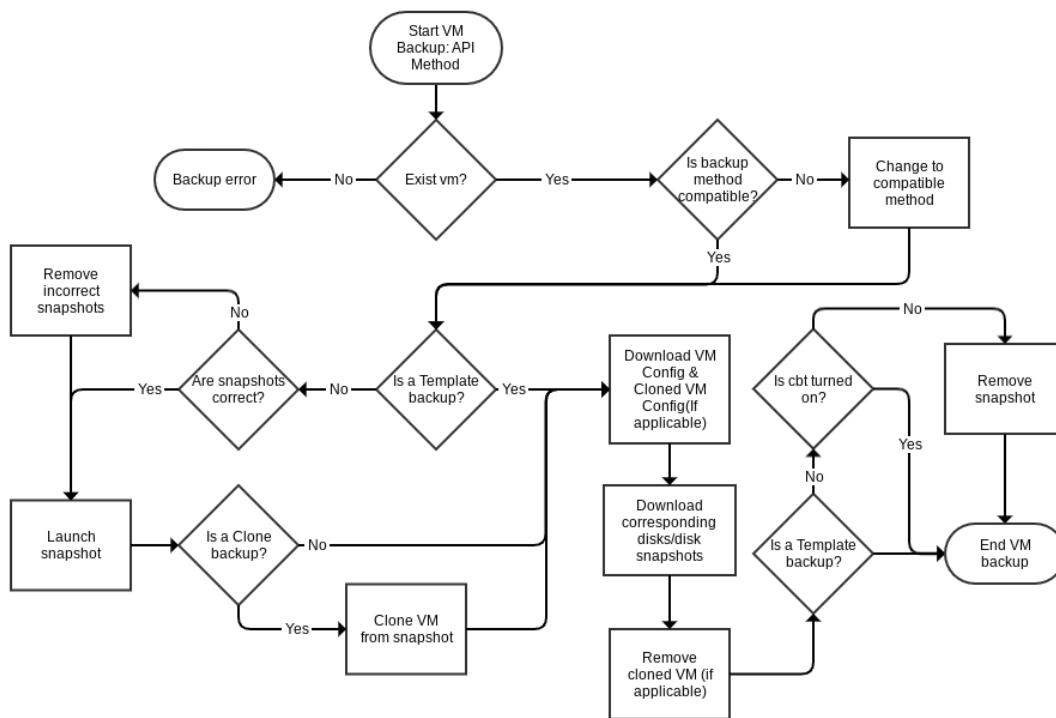


Fig. 5: External backup diagram

## Incremental and Differential Backups

### Incremental Backup Process

An incremental backup is one in which successive copies of the data will contain only the portion of the disk(s) that has changed since the previous backup was performed. When a full recovery is needed, the restoration process will need the last full backup plus all the incremental backups up to the point of restoration. Incremental backups are often desirable as they reduce storage space usage, and are quicker to perform than full or differential backups.

The method to perform incremental backups used by this plugin is as follows:

- CBT function is activated in the fileset.
- The first FULL will leave the snapshot, which it created during backup, in the machine.
- Each Incremental will perform a new snapshot, take the data from that new snapshot (which will contain the difference between the current state and the snapshot immediately before) and eliminate the snapshot created by the previous Incremental backup.

The disadvantages of this process are the following:

- A longer time is required during a restore since the Full and all Incrementals will need to be read.
- It is recommended to perform closed cycles that are not very long with the following sequence: 1 FULL + X INCREMENTAL, with 'X' being the number of backups in the chain. This recommendation is due to the fact that in each iteration that performs an incremental backup the snapshot that

is maintained in the virtual machine is larger, and therefore the operations carried out with it will take longer and longer, and more data will be stored. (See Table *A normal incremental backup*).

### Differential Backup Process

A differential backup is a type of data backup which only backs up the difference in the data since the last full backup. The rationale in this is that since changes to data are generally few compared to the entire amount of data in the data repository, the amount of time required to complete the backup will be smaller than if a full backup was performed every time that the organization or data owner wishes to back up changes since the last full backup. Another advantage, at least as compared to the incremental backup method of data backup, is that at data restoration time, at most two backup media are ever needed to restore all the data (The last Full + the last Differential). This simplifies data restores as well as increases the likelihood of shortening data restoration time.

The method to perform incremental backups used by this plugin is as follows:

- CBT function is activated in the fileset.
- The first FULL will leave the snapshot, which it created during backup on the machine. This full snapshot will be present in all differential backup processes until a new full is performed.
- Each differential will make a new snapshot. It will take the data of that new snapshot (which will contain the difference between the Full and the current state) and eliminate the snapshot at the end.

The advantages of this level of backup over INCREMENTAL are:

- Faster restorations.
- Unlimited iteration of differential backups without influencing backup performance.

In Figure **fig:incr-diff-backup-diagram** you can see 3 marks with the letters A, B and C. These marks correspond with:

1. Check previous backups with internal RhvCatalog stored in `/opt/bacula/working/rhv/catalog/{server_parameter}/{vm_id}.json`. The function of this catalog is to control the backups of each VM to check if they finish correctly. If this catalog entry doesn't exist for the VM, the plugin creates it with a **FULL** backup.
2. Incremental/Differential backups need a chain of snapshots. The base snapshot is the snapshot of the last Full backup done by the RHV plugin for a virtual machine. The description of the snapshots created by the RHV plugin is `backup_{jobId}_{jobName}`.
3. If the snapshot chain is interrupted by any snapshot created outside of the RHV plugin, the RHV plugin must make a **COMPLETE** backup. The RHV plugin will always keep foreign snapshots.

### Use cases in snapshot maintenance

Table 1: A normal incremental backup

	T1	T2	T3	T4	T5
Type Backup	F	I	I	I	I
Create snapshot	S1	S2	S3	S4	S5
Snapshots in VM	S1	S2	S3	S4	S5

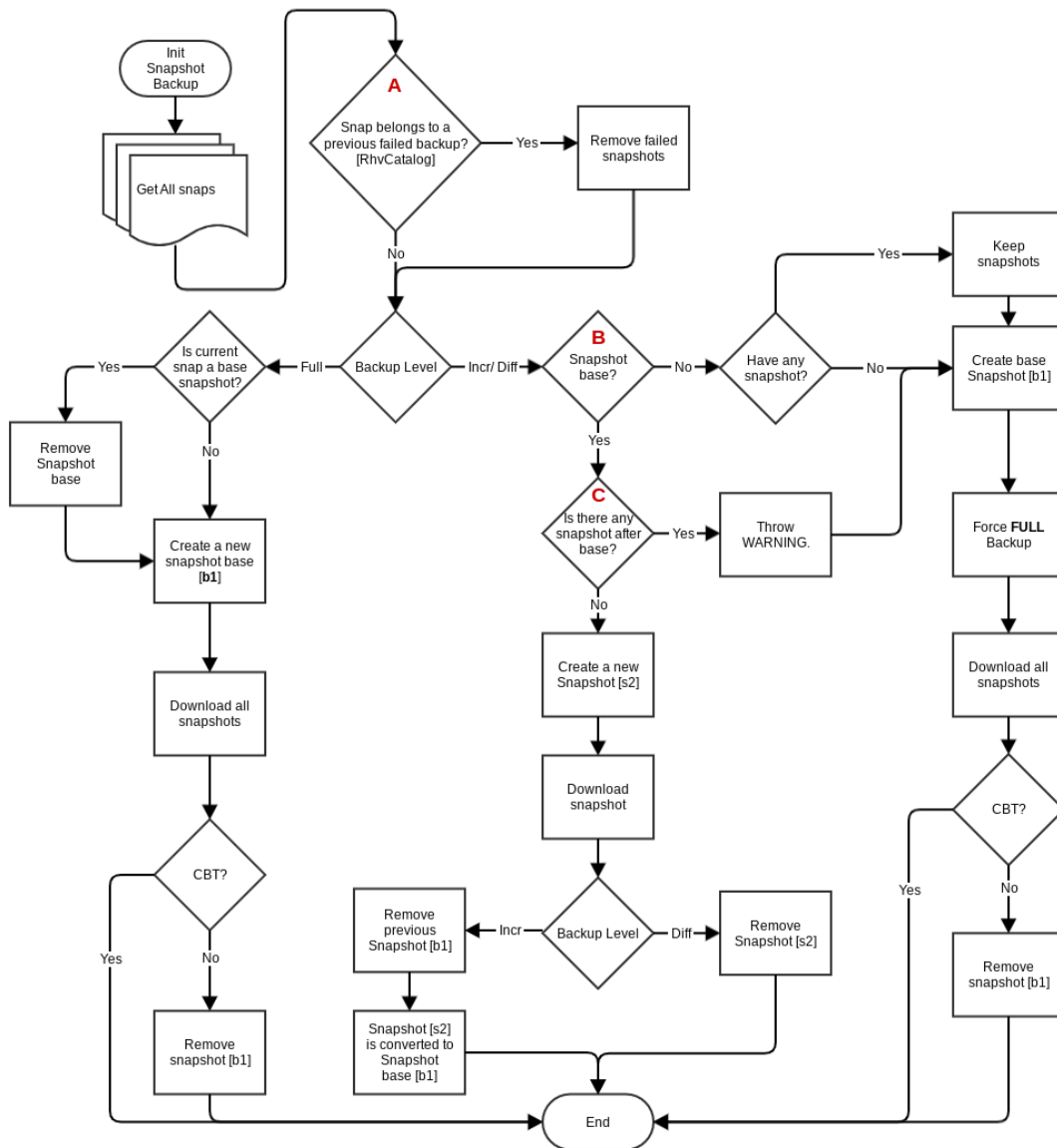


Fig. 6: Incremental/Differential backup diagram

Table 2: A normal differential backup

	T1	T2	T3	T4	T5
Type Backup	F	D	D	D	D
Create snapshot	S1	S2	S3	S4	S5
Snapshots in VM	S1	S1	S1	S1	S1

Table 3: A change of type backup

	T1	T2	T3	T4	T5
Type Backup	F	D	D	F	I
Create snapshot	S1	S2	S3	S4	S5
Snapshots in VM	S1	S1	S1	S4	S5

Table 4: An external snapshot in T1

	T1	T2	T3	T4	T5
Type Backup		F	I	I	I
Create snapshot	E1	S2	S3	S4	S5
Snapshots in VM	E1	E1 + S2	E1 + S3	E1 + S4	E1 + S5

Table 5: An external snapshot between backups

	T1	T2	T3	T4	T5
Type Backup	F	I		I *	I
Create snapshot	S1	S2	E3	S4	S5
Snapshots in VM	S1	S2	S2 + E3	S2 + E3 + S4	S2 + E3 + S5

I \*: Internally, the RHV plugin will do a FULL backup and log a warning.

### Backup from RHV environment internal machine

A proxyVM backup is a method where backup is done through a special VM inside the RHV environment. Disks belonging to VMs that need to be backed up will be attached to this special VM and backed up via this proxyVM. This method only allows FULL backups, but is much faster than the other REST API based methods.

The backup process consists of binding to each of the virtual machine disks to perform the backup, backing up of the contents of the disks, then unbinding from them. The binding process is made through a snapshot and an attach operation of the snapshot to the proxy VM.

### Backup Cache

This feature keeps VM backups in different storage domains to be used for future restores. This functionality allows faster restores of virtual machines, but requires more space available in the storage domains.

### Quiescing VMs

If the guest agent for RHV is installed in a VM, the system will be quiesced automatically and transparently when a snapshot is created. Red Hat's documentation at [redhat.com/.../Live\\_Snapshots\\_in\\_Red\\_Hat...html](https://redhat.com/.../Live_Snapshots_in_Red_Hat...html) provides full information.

## 3.2 Restore Process

The general restore process of a virtual machine consists of

1. Create a new VM using an XML configuration.
2. Create all of the VM's disks.
3. Create every snapshot, if necessary.
4. Upload disks.
5. Create the VM's network interfaces.
6. Create the template from the virtual machine, if necessary.
7. Switch on the virtual machine if necessary.

The disk upload operations are performed using the ImageTransfer service of oVirt.

## 4 Features

The following lists present the general features of this plugin.

### 4.1 Backup Features

- Full image-level backup.
- Incremental backup.
- Differential backup.
- Proxy backup.
- Agentless backup.
- RHV snapshot integration to create a snapshot to back up, and always delete this snapshot when the backup finishes, when the plugin finishes its work, or when the next backup is initiated if a prior backup error occurred.
- Snapshot consistency through quiescing VMs for backup.
- Backup storage cache in Red Hat Virtualization Manager (Where we keep cloned VMs).
- Individual VM Backups.
- VM derivated of template Backups.
- Individual Template Backups.
- Optional VM configuration only backups.
- Disk exclusion.
- Selection of VMs to back up by tags, regular expression matching on name, or per any structure RHV *Querying RHV*.
- Backup of virtual machines in a “running”, “paused” or “shut off” state.
- VM exclusion.
- Password obfuscation.
- Failed backup control:
  - Find and remove previous failed backups / snapshots with every execution.

- Backup cancellation and connection loss controls to try to remove snapshots or clones of current backup.
- Control of snapshots by an internal catalog.

## 4.2 Restore Features

- Restore to original or different location in RHV environment (Cluster or Storage Domain).
- Replace original VM or create a new VM with a new name.
- Restore templates.
- Change destination VM name.
- Local restore of disk images and VM XML configuration to allow manual processing.
- Exclude disks from restore.
- Turn VM on after restore, or leave turned off.
- Restore disk configuration:
  - Virtual disk interface (ide, virtio, virtio\_scsi, spart).
  - “Active” state of disk.
  - “Boot disk” flag.
  - Template to name restored disks.
  - Name list of restored disks.
- Restore NIC configurations (providing MAC address list).

## 5 Limitations

The plugin currently does not support the following features:

- The backup snapshot method is not compatible with RHVM4.1.
- Virtual Machines in Suspended status cannot be backed up. This is oVirt/RHV limitation as do a snapshot is not allowed with that state.
- The Incremental/Differential backup is only compatible with snapshot backups.
- The Incremental/Differential backup is compatible with RHV Plugin version  $\geq 4.0.0$ .
- The proxy backup is compatible only with Bacula Enterprise version  $\geq 12.4.0$ .
- Virtual Full backups.
- Using an external OAuth / SSO service different from the one provided directly by RHV.
- Stop or Resume incomplete Jobs. For this to be possible with the guarantee of consistent backups, snapshots of virtual disks and cloned VMs would need to be kept.
- Optimized backup with thin disks. The Ovirt API doesn't provide any information about thin disk usage, except with thin disks located in iSCSI storage.
- Restores from a backup sequences based on a Full and Differential and/or Incremental snapshots of VMs that contain at least one disk in a **Block Based Storage Domain** are possible only with RHV versions 4.4.5 or newer.

- The `restart` command has limitations with plugins, as it initiates the Job from scratch rather than continuing it. Bacula determines whether a Job is restarted or continued, but using the `restart` command will result in a new Job.

## 6 Compatibility and Requirements

### 6.1 Compatibility

The plugin is currently developed exclusively for RHEL Virtualization 4.1 and above environments (RHEL 7-based). It has been tested only on those platforms. Other virtualization platforms based on oVirt, and sufficiently similar to the RHV platform, may also be suitable.

**Bacula Systems** will add compatible platforms to this list when interoperability has been demonstrated through its development and QA processes. Please contact Bacula Systems Support if you have questions about the currently supported RHV versions.

### 6.2 Requirements

#### Bacula

The plugin works with version 10.1 or newer of Bacula Enterprise with **non Accurate mode** (The **Accurate** Job directive must be disabled).

#### Java

The server hosting the Bacula File Daemon must have the Java Virtual Machine version 8 or newer installed.

#### Red Hat Virtualization Considerations

##### *Disk Download*

In order to allow correct disk image downloads during the backup processes, and due to an existing bug of RHV, the following configuration changes must be made in RHV:

*For RHV 4.1 (Compatible with RHV 4.2)*

Connect to the RHV Manager host using `ssh`, then

```
$ su - postgres
$ psql -U postgres -d engine
# Get the existing value for future reference
psql (12.3)
Type "help" for help.

postgres=# SELECT * FROM vdc_options WHERE
option_name='ImageTransferClientTicketValidityInSeconds';
UPDATE vdc_options SET option_value=999999
WHERE option_name='ImageTransferClientTicketValidityInSeconds';
```

The value of `option_value` represents the validity time in seconds of the token to access disk images for download or upload. We recommend to set the life time to be virtually unlimited with the value 999999 in the above query.

##### *RHV 4.2*

Connect to the RHV Manager host using `ssh`, then

```
# Get the existing value for future reference
engine-config --get=ImageTransferClientTicketValidityInSeconds
# Set value (we recommend 999999 seconds, virtually unlimited)
engine-config --set=ImageTransferClientTicketValidityInSeconds=999999
```

### Storage Space

If RHV version 4.2 is used, additional backup storage space is not required, but if RHV version 4.1 is used, or the parameter 'apply clone' is set, as discussed in section *Backup Process*, backups need to perform a complete clone of a snapshot of each virtual machine being backed up. This clone is removed after a backup or at the start of a subsequent backup in case the previous one failed without the plugin having a chance to clean up. Therefore it is necessary to have sufficient storage space in the Storage Domain the backup runs in. It is also important to consider how many backups are going to be run in parallel and reserve enough space for those temporary clones.

## 6.3 Use of computer resources

### Proxy Backup

The process will run in a virtual machine that it is placed inside the same RHV environment we are protecting. It will be necessary to deploy this VM which contains the Bacula agent and proxy VM. This virtual machine has some minimal requirements:

- **OS:** There is no specific OS requirement and this mode should work with any updated OS. However, it is recommended to use Debian 10 (Buster), as it is the only one where proxy mode has been tested extensively.
- **RAM:** 2 GB.
- **Network:** Access to logical network RHV.
- **Storage:** 4GB.
- **Bacula Enterprise version:** 12.4.0.
- **Java Virtual Machine:** 8.

Calculations have been made that determine that the machine with the minimum requirements can process up to 20 backups in parallel. It will be necessary to add 1 GB of RAM for every 10 backups in parallel.

### API Backup

This process will execute in an external machine, and it has been calculated that 1 GB of available RAM is needed for every 10 parallel backups for this process.

## 7 Configuration

### 7.1 Plugin Installation

The **Bacula Enterprise** RHV plugin is distributed as a standard package for every supported operating system.

After installing the package, a few aspects must be considered: The Plugin Directory directive of the File Daemon resource in `/opt/bacula/etc/bacula-fd.conf` has to point to where the `rhv-fd.so` plugin is installed. The usual Bacula plugin directory is `/opt/bacula/plugins`. Thus, the configuration file `bacula-fd.conf` should look like in the following example:

```
FileDaemon {
  Name = bacula-fd
  Plugin Directory = /opt/bacula/plugins
  ...
}
```

## 7.2 Certificate and Truststore

There are two ways to create the truststore that allows the Bacula plugin to connect to RHV in secure mode.

### Automatic

#### *Call plugin*

The plugin has an option to create the truststore automatically. To create the truststore run the following command:

```
java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --server=myrhv.com --
↳operation=system
  --create_truststore=true --truststore_file=/opt/bacula/working/rhv_
↳truststore
  --truststore_password=changeit
```

Example command to create the truststore

```
user@host:~/\ $ java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --server=
↳{server}
  --operation=system --create_truststore=true --truststore_file={truststore_
↳path}
  --truststore_password={truststore_password}
```

#### *Interactive script*

There is an interactive script in `/opt/bacula/scripts/rhv_config.sh`. When the script is executed it requests parameters such as: server, truststore path, truststore password, alias in truststore and keytool Java path. Only the parameter 'server' is required. The others parameters by default are:

- Truststore path: `/opt/bacula/etc/rhv.cacerts`
- Truststore password: `changeit`
- Truststore internal alias: `rhvPluginX<randomNumber(1-100000)>`
- Path to Java's keytool: `/usr/bin/keytool`

Example of a script execution:

```
[root@rhv-client tmp]# /opt/bacula/scripts/rhv_config.sh
Welcome wizard to create TrustStore File
Enter FQDM Red Hat Virtualization Manager:rhv-mgmt.local.lan
Path truststore (/opt/bacula/etc/rhv.cacerts):
Password truststore (changeit):
Alias (rhvPluginX89803):
Path keytool ('/usr/bin/keytool'):
```

(continues on next page)

(continued from previous page)

```
Resume:
Server: rhv-mgmt.local.lan
Path Truststore: /opt/bacula/etc/rhv.cacerts
Pass Truststore: default
Alias: rhvPluginX89803
Keytool: /usr/bin/keytool

Are you sure? [no] yes
Truststore generate successfully
[root@rhv-client tmp]#
```

## Manual

To generate the truststore that allows the Bacula plugin to connect to RHV, the RHV server certificate needs to be downloaded and added to the existing Java truststore or a new one used by the plugin.

In the following examples, the address will be used; it needs to be replaced with the proper one.

To install the RHV server certificate on the File Daemon host, the certificate can be prepared in different ways:

- Download using http, as in

```
# beware - line broken below!
curl -o rhvm.cer 'http://rhv.example.com/ovirt-engine/services/
pki-resource?resource=ca-certificate&format=X509-PEM-CA'
```

- Edit a new certificate file and paste the needed content:
  1. Log in to the Bacula File Daemon host.
  2. Use ssh to connect to the RHV Manager.
  3. `cat /etc/pki/ovirt-engine/ca.pem`
  4. Copy the contents from the “BEGIN CERTIFICATE” line, up to and including the “END CERTIFICATE” one.
  5. Log out from the RHV host.
  6. Edit a new text file: For example `vim rhvm.cer`
  7. Paste the copied text. (With vim in a typical Linux terminal session, the key strokes would be “i shift-ctrl-v escape : x”.)

Once the certificate file, called `rhvm.cer` in the example above, is available, it be added to a Java truststore file. The truststore path and password are general required parameters of this plugin.

It is possible to use an existing Java truststore (including the local default file), or to create a new one. For this example, a new one is created:

```
keytool -import -alias "mirhvm.crt_truststore" \
-file rhvm.cer -keystore rhvm.truststore
# A prompt will ask for a password
```

To use the existing, default Java truststore, the command would be as follows:

```
keytool -import -alias "mirhvm.crt_truststore" \
-file rhvm.cer -keystore $JAVA_HOME/jre/lib/security/cacerts
# This will ask for a password, default value is: changeit
```

**Note**

By default, the Java truststore path is: \$JAVA\_HOME/jre/lib/security/cacerts

Please refer to the relevant Java documentation for an explanation of the `keytool` command, or Java truststore in general.

**Note**

The full path to the truststore file used, as well as the password, are needed for further plugin configuration steps, as well as for its operation.

**Without Certificate**

The plugin also allows a connection without certificate. It is discouraged to use this mode in production environments. To run the plugin in this mode, the plugin must be execute with the option “insecure”, explained in *Plugin Parameters*.

### 7.3 User Permissions

It is possible to use the RHV “admin” user for the plugin. But, in general, IT organizations should prefer using a specific backup user account with more restricted permissions. In order to have full functionality of this plugin, the user account that will be used must have the following RHV permissions:

Table 6: RHV User Permissions

Category	Required setting
RHV Special Roles	ExternalEventsCreator UserVmManager
System, <i>Configure System</i>	Login Permissions
Template, <i>Provisioning Operations</i>	Import/Export
VM	Run VM
VM, <i>Provisioning Operations</i>	Edit properties Create Create Instance Delete Edit Storage Edit properties
Disk, <i>Provisioning Operations</i>	Create Delete
Disk, <i>Provisioning Operations, Edit Storage</i>	Attach Manipulate SCSI I/O Privileges

## 8 Plugin Parameters

Below all options and parameters that can be used in a Fileset plugin line with the RHV plugin are described.

Options which have default values don't need to be defined as long as their default behavior is desirable.

### Note

To create obfuscated passwords, the Bacula administrator should execute the command `@encode password` in `bconsole` and copy the result, or execute it directly via the plugin. This application is explained in section *Plugin System*.

### Warning

When orthogonal VM selection schemes are combined, the plugin only will back them up once. For example: **target\_datacenter=dc1** and **target\_virtualmachine=vm1,vm2** where **vm1 is in dc1**, and **vm2 is in dc2**, the plugin will backup **dc1** and **vm2**, and **vm1 only backup once**.

## 8.1 General Plugin Parameters

Parameter	Require	Default	Description	Example
<b>server</b>	Yes		DNS name of RHV~Manager (the server parameter requires a domain name, not an URI, nor are IP addresses allowed).	<code>server=rhv.example.com</code>
<b>use_image</b>	No	false	Indicates whether the Manager's proxy or the host itself will be used to retrieve disk data. This requires that the hosts be network-accessible by the bacula-fd.	<code>use_imageio_in_mng_node=true</code>
<b>truststore_file</b>	Yes		File name of truststore	<code>truststore\_file=/opt/bacula/etc/rhv-trust</code>
<b>truststore_pass</b>	No	changeit	Truststore password	<code>truststore\_password=changeit</code>
<b>truststore_hpassword</b>	No	changeit	Obfuscated truststore password, use either the obfuscated or plain password	<code>truststore_hpassword=NTUy0jU0NDpRR1xa</code>
<b>insecure</b>	No	false	Mode insecure, without truststore. Not recommended in production environment	<code>insecure=yes</code>
<b>user</b>	No	admin	RHV user account to authenticate as	<code>user=bacula</code>
<b>password</b>	Yes		User password	<code>password=rhvpass123</code>
<b>hpassword</b>	Yes		Obfuscated user password	<code>hpassword=NTUy0jU0NDpRR1xaR0VJWwA</code>
<b>auth</b>	No	http	Authentication type, http or oauth	<code>auth=oauth</code>
<b>config_file</b>	No		Path to configuration file	<code>config_file=/opt/bacula/etc/rhv.conf</code>
<b>profile</b>	No	internal	Connection user profile	<code>profile=internal</code>
<b>abort_on_</b>	No	1	Abort the backup in case of any problem with disks implied or previous backup files (0 or 1)	<code>abort_on_error=0</code>
<b>output</b>	No	json	Helper program output format, one of json, bacula, console	<code>output=console</code>
<b>log</b>	No	./rhv-plugin	Log to specified file	<code>log=/tmp/rhvbkup.log</code>
<b>debug</b>	No	0	Enable debug output: 0 (informational), 1 (debug), 2 (deep debug)	<code>debug=1</code>

## 8.2 Tuning Parameters

Parameter	Re- quired	De- fault	Description	Example
<code>system_num_tries_connection</code>	No	5	Number of failed attempts in http requests to RHV/oVirt server before failing	<code>system_num_tries_connection=10</code>
<code>system_interval_request</code>	No	120	Interval between http request to RHV/oVirt server (seconds)	<code>system_interval_request=360</code>
<code>system_timeout_request</code>	No	120	Timeout for http requests done to RHV/oVirt server (seconds)	<code>system_timeout_request=360</code>

### 8.3 Backup-Specific Parameters

Parameter	Require	Default	Description	Example
<code>target_datacenter</code>	No		Back up VMs from Data Center	<code>target_datacenter=production</code>
<code>target_cluster</code>	No		Back up VMs from specified Cluster	<code>target_cluster=webfarm</code>
<code>target_tag</code>	No		Back up VMs with specified tag	<code>target_tag=bkup</code>
<code>target_virtualmachine</code>	No		Comma-separated list of VM names to back up	<code>target_virtualmachine=web1,web2,db1</code>
<code>target_path</code>	No		Backup all vms of children path. Support comma-separated list.	<code>target_path=/datacenters/Default/clusters/Default/,/tags/tag1/</code>
<code>target_template</code>	No		Comma-separated list of Template names to back up	<code>target_template=template1,temp2,temp3</code>
<code>target_exclude_disks</code>	No		<b>In testing</b> , comma-separated list of disk ids to exclude	<code>target_exclude_disks=65ccb526-30c0-4b3b874e8c-dddb-4e1e-a4fc-3cdf0b76ec1c</code>
<code>target_exclude_vms</code>	No		Comma-separated list of VM names to exclude	<code>target_exclude_vms=web,db2</code>
<code>target_configuration_only</code>	No	false	Back up only VM configuration, not disk images, true or false	<code>target_configuration_only=true</code>
<code>clone_storage</code>	No	false	Enable clone mode backups	<code>clone_storage=on</code>
<code>restorecache_retention</code>	No	0	Number of clones to keep in RHVM	<code>restorecache_retention=3</code>
<code>restorecache_storage_domain</code>	No		Target storage to keep backup clone	<code>restorecache_storage_domain=dataMaster</code>
<code>cbt</code>	No	false	Active incremental or differential backup, available since Bacula Enterprise version 12.0.2 or newer	<code>cbt=true</code>
<code>proxy_vm</code>	No		Virtual machine to proxy backup	<code>proxy_vm=ProxyVmBackup</code>
<code>proxy_block</code>	No	true	Safely attach/detach disks in proxy backup	<code>proxy_block=true</code>
<code>persist_memory_state</code>	No	false	Save the memory state (or not) when taking a live snapshot for the backup. During a live snapshot with memory creation, the VM will be locked to prevent it from being changed. When the memory dump is being saved, the VM will be switched to the 'paused' state. Available since Bacula Enterprise 14.0.5. previously, version memory was persisted by default.	<code>persist_memory_state=true</code>

## 8.4 Restore-Specific Parameters

### Note

These are usually set interactively during restore, **not** in a plugin line in a file set.

The correct compatibility [vm, cluster, storageDomain] is the responsibility of the administrator.

Parameter	Require	Default	Description	Example
<b>vm_n</b>	No	suffix	Name of restored VM, suffix and original have special meaning (see chapter <i>Restore</i> for details)	vm_name=restoredVM
<b>template_</b>	No	suffix	Name of restored Template, suffix and original have special meaning (see chapter <i>Restore</i> for details)	template_name=restoredTemplate
<b>vm_d</b>	No		Comma-separated list of disks to restore (skip all others)	vm_disks=3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e
<b>vm_e:</b>	No		<b>In testing</b> , comma-separated list of disks to exclude	vm_exclude_disks=3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e
<b>cluster</b>	No	original	Cluster to restore to, uses value from backup	cluster=secondary
<b>storage_d</b>	No	original	Storage Domain name to restore to: For specific cases that affect only one disk, you should use the id of the disk (you can use the restore prompt) delimited by ':' and the value. For the general case, no id or ':' should be used. Separate each case by ','.	storage_domain=3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e:primary
<b>vm_fc</b>	No	false	Overwrite existing target VM, true or false	vm_force_overwrite=true
<b>vm_s</b>	No	false	Turn on restored VM, true or false	vm_switchon=true
<b>remove_</b>	No	true	In template restore or vm restore with create template, remove vm restored after create template (see chapter <i>Restore</i> for details)	remove_base_vm=false
<b>disk_i</b>	No	original	Interface types for restored disks; comma-separated list of original, ide, virtio, virtio_scsi, spart	disk_interface=3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e:ide
<b>disk_:</b>	No	original	Comma-separated list of original, true, false flags to make disks active	disk_active=true, 3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e
<b>disk_ </b>	No	original-footref	Comma-separated list of original, true, false flags to make disks bootable	disk_boot=true, 3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e
<b>disk_ </b>	No	original	Comma-separated list of names for restored disks	disk_names=3b874e8c-dddb-4e1e-a4fc-3cdf0b7c1e1e:restOfDisks
<b>nics</b>	No	new	Comma-separated list (one entry per virtual NIC) of MAC addresses or original, new	nics=C0:BA:C0:7A:CA:FE, original

## 8.5 System-Specific Parameters

Parameter	Re-require	De-fault	Description	Example
<code>create_truststore</code>	No	false	Create truststore of server RHVM (only use with Java Daemon), see <i>Plugin System</i> for details	<code>create_truststore=true</code>
<code>system_num_tries_connection</code>	No	5	Number failed connection tries before throw an Error more important, see section <i>System operations</i> for details}	<code>system_num_tries_connection=10</code>
<code>system_interval_request</code>	No	120	Time(seconds) between failed try and other try	<code>system_interval_request=60</code>
<code>system_timeout_request</code>	No	120	Timeout(seconds) to request server	<code>system_timeout_request=60</code>

## 8.6 Configuration File

The plugin parameter `config_file` allows for the creation of a file which can be used to provide all necessary options instead of putting them into the Fileset's RHV plugin line. These configuration files are plain text files containing one setting per line where the name of the setting and its value are separated by an equals sign "=". An example configuration file could look like this:

```
server=rhv.example.com
user=admin
truststore_file=/opt/bacula/etc/rhvm.truststore
target_datacenters=myDatacenter
operation=backup
```

If a parameter is mentioned more than once, the last instance is used.

## 9 Fileset Configuration

A Fileset to back up RHV VMs will usually contain one `plugin=rhv:...` line, and may contain `File=...` directives.

It is recommended to have only a single RHV plugin line per Fileset, which will ensure that no unpredictable behaviour in case of errors such as failure to proceed with one plugin call, but success with all others.

**Bacula Systems** strongly advises against the use of other plugins in Filesets which use the RHV plugin.

### 9.1 Backup Configuration Examples

The minimal Fileset. This Fileset will backup all vms in RHV environment:

```
Fileset {
  Name = minimalFileset
  Include {
    Plugin = "rhv: server=rhv.example.com insecure=true password=rhypass123"
  }
}
```

The minimal Fileset using proxy backup method, using the *PROXY\_BACKUP\_VM*. This Fileset will backup all VMs in the RHV environment (includes proxy backup VM):

```
Fileset {
  Name = minimalFilesetProxy
  Include {
    Plugin = "rhv: server=rhv.example.com insecure=true password=rhvpas123_
↪proxy_vm=PROXY_BACKUP_VM"
  }
}
```

The minimal backup Job. This Job uses the minimal Fileset above, so it will backup all VMs in the RHV environment:

```
Job {
  Name = minimalJob
  Type = Backup
  Level = Full
  Write Bootstrap = "/var/bacula/working/minimalJob.bsr"
  Accurate = no
  Client = Client1
  Maximum Concurrent Jobs = 1
  Allow Duplicate Jobs = no
  Fileset = minimalFilesetProxy
  Storage = Storage1
  Pool = Pool1
  Messages = Standard
  Enabled = yes
}
```

Backing up only “vmExample” can be achieved like this:

```
Fileset {
  Name = ExampleFileset0
  Include {
    Options {
      Signature = MD5
    }
    Plugin = "rhv: server=rhv.example.com insecure=true password=rhvpas123_
↪target_virtualmachine=vmExample"
  }
}
```

A backup of the VM named “vm1” and all vms that the datacenter “dc1” contains, excluding the disks with ids 65ccb526-30c0-4beb-b348-4395e70d6e32 and 3b874e8c-ddbb-4e1e-a4fc-3cdf0b76ec1c, would be configured like this:

```
Fileset {
  Name = ExampleFileset1
  Include {
    Options {
      Signature = MD5
    }
    Plugin = "rhv:server=rhv.example.com
```

(continues on next page)

(continued from previous page)

```
truststore_file=/usr/lib/jvm/java-1.8.0-openjdk-amd64/jre/lib/security/
↪cacerts/rhvm.truststore
  auth=http user=admin password=rhvpas123 target_virtualmachine=vm1↪
↪target_datacenter=dc1
  target_exclude_disks=65ccb526-30c0-4beb-b348-4395e70d6e32,3b874e8c-dddb-
↪4e1e-a4fc-3cdf0b76ec1c"
}
}
```

The backup of all VMs with names starting with “vm”, excluding “vm1”:

```
Fileset {
  Name = ExampleFileset2
  Include {
    Options {
      Signature = MD5
    }
    Plugin = "rhv:server=rhv.example.com truststore_password=changeit↪
↪auth=http profile=internal
  truststore_file=/usr/lib/jvm/java-1.8.0-openjdk-amd64/jre/lib/security/
↪cacerts/rhvm.truststore
  user=admin password=rhvpas123 target_virtualmachine=vm* target_exclude_
↪vms=vm1"
  }
}
```

The backup of all templates with names starting with “template”:

```
Fileset {
  Name = ExampleFileset3
  Include {
    Options {
      Signature = MD5
    }
    Plugin = "rhv:server=rhv.example.com truststore_password=changeit↪
↪auth=http profile=internal
  truststore_file=/usr/lib/jvm/java-1.8.0-openjdk-amd64/jre/lib/security/
↪cacerts/rhvm.truststore
  user=admin password=rhvpas123 target_template=template*"
  }
}
```

Backing up only “vm1” with backup proxy method, using the vm *PROXY\_BACKUP\_VM*:

```
Fileset {
  Name = ExampleProxyFileset
  Include {
    Options {
      Signature = MD5
    }
    Plugin = "rhv: server=rhv.example.com insecure=true password=rhvpas123↪
↪target_virtualmachine=vm1 proxy_vm=PROXY_BACKUP_VM"
}
```

(continues on next page)

(continued from previous page)

```
}  
}
```

## 10 Restore

Restores of backups done using the RHV plugin are initiated like restores of any file-based backups. The File Daemon used for the restore, indicated by the Client selected as restore target, must have the RHV plugin installed in order to allow restores to RHV directly.

From version 3.0.0, the plugin allows restores of templates. This integration is compatible with restore of vms. And also it allows create a template from a restored vm, directly. Only the appropriate parameters should be used.

From version 4.0.0 (**Bacula Enterprise** version 12.4.0), the plugin allows restores of incremental and differential backups, and proxy backups.

If the `where` parameter of a restore of a RHV plugin backup is set to indicate the original location, the restore will be passed directly to RHV Manager. An example would be a `bconsole` command line such as:

```
restore job=one-RHV-VM where=/  

```

If the restore target location is set to anything else, the selected data will be restored to the path indicated. VM or TEMPLATE configuration will be in an XML file, and virtual disk images will be stored in the format applicable, depending on format at backup time and restore options set. This operation is not necessary use the truststore file.

When selecting VMs or TEMPLATES to restore, these will be represented as sub-directories in a common root directory `@rhv/`. Only one VM or one TEMPLATE (one subdirectory) should be restored at a time, and for restores to RHV directly (as opposed to a plain file), the whole subdirectory needs to be marked.

The treatment between templates and vms is done transparently to the user. Therefore, the user should know if it is a job with a template backup or vm backup. The only way to know if a backup was from a vm or a template is by looking at the name of a file inside the directory. This file will be named as: “`nameVm|nameTemplate_vm|template_BaculaRhvBackup.xml`”. Such as:

```
exampleVm_vm_BaculaRhvBackup.xml  
exampleTemplate_template_BaculaRhvBackup.xml
```

The options that were used during a backup job are stored in Bacula’s catalog database and they will be applied during a restore session. To provide full flexibility, these options can be modified, and some options specific to restores are also available. These options are documented, along with all other plugin options, in table *Restore-Specific Parameters*.

Note that some options will be used to pass lists of values to the plugin. Such lists consist of comma-separated values. If a list does not contain enough values to be applied to all related objects, the default value as shown in table *Restore-Specific Parameters* is used for the missing ones.

Often, special values are available to cause certain behaviour; such options are explained below, in detail.

When using `bconsole`, the options can be modified using the option 13 (“Plugin Opts”) of the “modify” menu.

An example `bconsole` restore session of a RHV backup job that saved a single VM, with one disk won’t be active but all of them yes, is shown below:

```

Connecting to Director 127.0.0.1:8101
1000 OK: 103 127.0.0.1-dir Version: 9.0.6 (20 November 2017)
Enter a period to cancel a command.
restore jobid=926 Client=RHV-fd where=
Using Catalog "MyCatalog"
You have selected the following JobId: 926
Building directory tree for JobId(s) 926 ...
6 files inserted into the tree.
You are now entering file selection mode where you add (mark) and
remove (unmark) files to be restored. No files are initially added, unless
you used the "all" keyword on the command line.
Enter "done" to leave this mode.
cwd is: /
$ cd @rhv/
cwd is: /@rhv/
$ ls
vmTest/
$ cd vmTest/
cwd is: /@rhv/vmTest/
$ ls
DiskSnap_513901a8-095e-4bbc-a6d3-b082e91fe64c_151736db-dba5-491b-af39-
↪537914f4a176_0.img
DiskSnap_513901a8-095e-4bbc-a6d3-b082e91fe64c_e3d2062a-d463-4271-9c8a-
↪ff85a8d73d3c_1.img
DiskSnap_58efd04d-48e8-4ab6-9d9c-42247798fd31_151736db-dba5-491b-af39-
↪537914f4a176_4.img
DiskSnap_58efd04d-48e8-4ab6-9d9c-42247798fd31_e3d2062a-d463-4271-9c8a-
↪ff85a8d73d3c_5.img
DiskSnap_d382cd50-f052-4621-950a-4692200012e7_151736db-dba5-491b-af39-
↪537914f4a176_2.img
DiskSnap_d382cd50-f052-4621-950a-4692200012e7_e3d2062a-d463-4271-9c8a-
↪ff85a8d73d3c_3.img
vmTest_attachs_BaculaRhvBackup.xml
vmTest_config_BaculaRhvBackup.json
vmTest_disks_BaculaRhvBackup.xml
vmTest_dsnaps_BaculaRhvBackup.xml
vmTest_nics_BaculaRhvBackup.xml
vmTest_snapshots_BaculaRhvBackup.xml
vmTest_vm_BaculaRhvBackup.xml
$ cd ..
cwd is: /@rhv/
$ mark vmTest
13 files marked.
$ done
$
Bootstrap records written to
/opt/bacula/working/bacula-dir.restore.6.bsr
The Job will require the following (*=>InChanger):
Volume(s)                Storage(s)                SD Device(s)
=====
poolrhv-0415              RestoreStorage
Volumes marked with "*" are in the Autochanger.

```

(continues on next page)

(continued from previous page)

```
6 files selected to be restored.
Using Catalog "MyCatalog"
Run Restore job
JobName:          RestoreFiles
Bootstrap:        /opt/bacula/working/bacula-dir.restore.6.bsr
Where:
Replace:          Always
Fileset:          Full Set
Backup Client:    RHV-fd
Restore Client:   RHV-fd
Storage:          File
When:             2018-05-22 10:32:51
Catalog:          MyCatalog
Priority:          10
Plugin Options:   *None*
OK to run? (yes/mod/no): mod
Parameters to modify:
1: Level
2: Storage
3: Job
4: Fileset
5: Restore Client
6: When
7: Priority
8: Bootstrap
9: Where
10: File Relocation
11: Replace
12: JobId
13: Plugin Options
Select parameter to modify (1-13): 13
Automatically selected : rhv:"abort_on_error=0"
"server=cubietruck1.dtic.ua.es"
"truststore_file=/opt/bacula/etc/rhvm.truststore"
"truststore_password=sosecret!" "auth=http" "profile=internal"
"user=admin" "password=alsogood" "target_virtualmachine=vmB3"
Plugin Restore Options
server:           *None*                (*None*)
truststore_file:  *None*                (*None*)
truststore_password: *None*            (*None*)
system_num_tries_connection: *None*          (*None*)
system_interval_request: *None*          (*None*)
system_timeout_request: *None*          (*None*)
auth:             *None*                (*None*)
profile:          *None*                (*None*)
user:            *None*                (*None*)
password:        *None*                (*None*)
vm_disks:        *None*                (*None*)
vm_exclude_disks: *None*                (*None*)
cluster:         *None*                (*None*)
storage_domain:  *None*                (*None*)
vm_name:         *None*                (*None*)
```

(continues on next page)

(continued from previous page)

```
vm_force_overwrite: *None* (*None*)
vm_switchon: *None* (*None*)
disk_interface: *None* (*None*)
disk_active: *None* (*None*)
disk_boot: *None* (*None*)
nics: *None* (*None*)
disk_names: *None* (*None*)
template_name: *None* (*None*)
remove_base_vm: *None* (*None*)
log: *None* (*None*)
debug: *None* (*None*)
debug:Use above plugin configuration? (yes/mod/no): mod
You have the following choices:
1: server (Restore server)
2: truststore_file (Restore truststore file)
3: truststore_password (Restore truststore password)
4: system_num_tries_connection (Number failed tries before abort)
5: system_interval_request (Time (s) between failed tries)
6: system_timeout_request (Timeout(s) to request server)
7: auth (Mode authentication(http|auth2))
8: profile (Profile connection user)
9: user (Restore user name)
10: password (Password user)
11: vm_disks (List id disks to restore)
12: vm_exclude_disks (Exclude id disks in restore)
13: cluster (Target restore cluster)
14: storage_domain (Target restore domain)
15: vm_name (New name Vm restore)
16: vm_force_overwrite (Force overwrite if exist vm)
17: vm_switchon (Turn on vm when restore finished)
18: disk_interface (List interface disk)
19: disk_active (List if want active disks)
20: disk_boot (List if want bootable disks)
21: nics (List restore MACS)
22: disk_names (List disks' names)
23: template_name (New name Template restore)
24: remove_base_vm (Remove base vm of template)
25: log (Log path in restore phase)
26: debug (Level debug in restore phase)
Select parameter to modify (1-24): 15
Please enter a value for vm_name: TestBaculaActive
Plugin Restore Options
server: *None* (*None*)
...
storage_domain: *None* (*None*)
vm_name: TestBaculaActive (*None*)
vm_force_overwrite: *None* (*None*)
vm_switchon: *None* (*None*)
disk_interface: *None* (*None*)
disk_names: *None* (*None*)
Use above plugin configuration? (yes/mod/no): mod
You have the following choices:
```

(continues on next page)

(continued from previous page)

```
1: server (Restore server)
...
16: vm_force_overwrite (Force overwrite if exist vm)
17: vm_switchon (Turn on vm when restore finished)
18: disk_interface (List interface disk)
19: disk_active (List if want active disks)
20: disk_boot (List if want bootable disks)
21: nics (List restore MACS)
22: disk_names (List disks' names)
23: template_name (New name Template restore)
24: remove_base_vm (Remove base vm of template)
25: log (Log path in restore phase)
26: debug (Level debug in restore phase)
Select parameter to modify (1-24): 16
Please enter a value for vm_force_overwrite: true
Plugin Restore Options
server:          *None*          (*None*)
...
storage_domain: *None*          (*None*)
vm_name:         TestBaculaActive (*None*)
vm_force_overwrite: true          (*None*)
vm_switchon:    *None*          (*None*)
disk_interface: *None*          (*None*)
disk_active:    *None*          (*None*)
disk_boot:      *None*          (*None*)
nics:           *None*          (*None*)
disk_names:     *None*          (*None*)
template_name:  *None*          (*None*)
remove_base_vm: *None*          (*None*)
log:            *None*          (*None*)
debug:         *None*          (*None*)
Use above plugin configuration? (yes/mod/no): yes
Run Restore job
...
OK to run? (yes/mod/no): yes
Job queued. JobId=935
```

## 10.1 File-Level Restore

It is possible to perform file listing and extraction functions over the virtual disk images that are restored on the Bacula client's file system (as opposite to restore directly to the RHV manager) by utilizing the set of tools developed and maintained as a part of a libguestfs project (<http://libguestfs.org>).

Prerequisites are to have libguestfs tools installed on the host that the virtual disk image file is going to be restored on and to have the appropriate daemon started.

Example of commands that should be run in order to meet these prerequisites are, in the case of RHEL7 host:

```
yum install libguestfs-tools
```

```
systemctl start libvirtd
```

In order to be able to list the contents of the virtual disk image, the disk image itself needs to be restored on the filesystem of the local host and command `virt-ls` executed.

Example of the command usage (where the virtual disk has been restored to the file `/tmp/bacula-restore/vm1-cos7/Disk1-249226e-c38c-44f0-a121.img` in order to list the contents of the `/` directory):

```
# virt-ls -R -a /tmp/bacula-restore/vm1-cos7/Disk1-249226e-c38c-44f0-a121.img ↵  
↵ /
```

In order to be able to extract the contents of the virtual disk image, the disk image itself needs to be restored on the filesystem of the local host and command `virt-copy-out` executed.

Example of the command usage (where the virtual disk has been restored to the file `/tmp/bacula-restore/vm1-cos7/Disk1-249226e-c38c-44f0-a121.img` in order to extract the contents of the `/home/` directory to the `/tmp` directory):

```
# virt-copy-out -a /tmp/bacula-restore/vm1-cos7/Disk1-249226e-c38c-44f0-a121.  
↵img \  
/home/ /tmp
```

### Extra notes

- In RHV, inactive disks are ignored when a clone is made from a snapshot.
- In a snapshot backup, if there is a inactive disks without any snapshots, the content of this won't be backed. But the disk will be created, without content, in the restore.
- The templates always use the disk format “thick-provisioned”, although the format disks of base vm was “thin-provisioned”. Can not modify templates' disks.
- The disks out of the template behave like disks of independent vms.
- The backup of virtual machines derived from independent templates behaves like independent virtual machines.
- The backup of vms derivated of dependent templates, only the snapshots of the machine will be backed up, without those of the template.
- The networks of the templates can be modified(name, connected, etc..) after the template was created. So if the network name is changed in restores from previous backups to this change as result will be more networks, as many networks as have been modified.

## 11 Diagnostic Operations

For diagnostic purposes, it is possible to interact with the RHV plugin java program on the shell to list information it retrieves from the RHV system as well as to actually transfer data.

### 11.1 Querying RHV

Assuming the Java environment is properly configured, the general approach is as follows:

```
java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --server=<RHVM-DNS> \  
--truststore_file=<path-to-truststore> --truststore_password=<password> \  
--auth=<auth> --profile=<profile> \  
--user=<user-name> --password=<password> \  
--operation=list --path=<RHVM structure>
```

Results will be returned in JSON format.

The required options indicated above are directly representing the plugin options as described in table *Plugin Parameters*.

The following list shows all the available RHVM structure, by default “/vms”:

```
/
├── api/
│   ├── version/
│   └── compatibility/cluster_storage/"name_cluster"/"name_storage_domain"
├── clusters/"name"/
│   ├── hosts -> ../hosts
│   ├── vms -> ../vms
│   └── templates -> ../templates
├── datacenters/"name"/
│   ├── clusters -> ../clusters
│   ├── disks -> ../disks
│   ├── hosts -> ../hosts
│   ├── storages -> ../storages
│   ├── vms -> ../vms
│   └── templates -> ../templates/
├── disks/"name"/
├── hosts/"name"/
│   ├── storages -> ../storages/
│   └── vms -> ../vms
├── tags/"name"/
│   └── vms -> ../vms
├── vms/"name"/
│   ├── disks -> ../disks
│   └── snapshots
├── storages/"name"/
│   ├── disks -> ../disks
│   ├── vms -> ../vms
│   └── templates -> ../templates
└── templates/"name"/
    ├── disks -> ../disks
    └── vms -> ../vms
```

It is also possible to list RHV information using using the `.ls`-command of `bconsole`. Below, some examples are provided.

The meaning of the required and optional parameters are the same as shown in table *Plugin Parameters* and above for the query functionality.

Note also that `bconsole` commands need to be entered without any line breaks.

Verifying compatible backups of VMs than they starts with `vm1*`:

```
*.ls client=RHV-fd plugin="rhv:server=myrhv.com \
insecure=true password=rhvPass123" path=/vms/vm1*
```

(continues on next page)

(continued from previous page)

```
#Answer:
# -rw-r----- 0 root root 0 1970-01-01 01:00:00 4f5f9abd-f407-4da6-ba39-
↪272aed709c80->vm1\
                                [vm] [BACKUP_CLONE, BACKUP_SNAP]
# -rw-r----- 0 root root 0 1970-01-01 01:00:00 a8bf6b18-a116-4067-b884-
↪172777fec446->vm11\
                                [vm] [BACKUP_SNAP]
```

Listing all disks attached to “vm1”:

```
.ls client=RHV-fd plugin="rhv:server=myrhv.com \
insecure=true user=bacula password=alsogood" path=/vms/vm1/disks/

# Answer:
#-rw-r----- 0 root root 0 1970-01-01 01:00:00 c43a1865-13ba-483e-b38b-
↪6b929d6b4653->\
                                Preassigned_Disk1 [disk]
#-rw-r----- 0 root root 1 1970-01-01 01:00:00 b0b349ea-72d3-405f-9429-
↪c1cb009f30ce->\
                                Preassigned_Disk2 [disk]
#-rw-r----- 0 root root 2 1970-01-01 01:00:00 3f993fb1-3157-4916-aad1-
↪2f323a3cdd4f->\
                                Preassigned_Disk3 [disk]
#-rw-r----- 0 root root 3 1970-01-01 01:00:00 e9ac9bd8-920f-4eb8-b4d0-
↪b405d675b16f->\
                                Preassigned_Disk4 [disk]
```

To list all datacenters:

```
.ls client=RHV-fd plugin="rhv:server=RHV.example.com \
insecure=true user=bacula password=alsogood" path=/datacenters/

# Answers:
# -rw-r----- 0 root root 0 1970-01-01 01:00:00 3e54cf31-7a9a-46ad-8a8d-
↪7843b6d7a145->\
                                dcTest [datacenter]
# -rw-r----- 0 root root 1 1970-01-01 01:00:00 4cdcc978-73af-11e8-835d-
↪b827eb7e5e47->\
                                Default [datacenter]
```

To list all options in datacenter named “dcTest”:

```
.ls client=RHV-fd plugin="rhv:server=RHV.example.com \
insecure=true user=bacula password=alsogood" path=/datacenters/dcTest/

# Answer:
# -rw-r----- 0 root root 0 1970-01-01 01:00:00 clusters
# -rw-r----- 0 root root 1 1970-01-01 01:00:00 disks
# -rw-r----- 0 root root 2 1970-01-01 01:00:00 hosts
# -rw-r----- 0 root root 3 1970-01-01 01:00:00 storages
# -rw-r----- 0 root root 4 1970-01-01 01:00:00 templates
# -rw-r----- 0 root root 5 1970-01-01 01:00:00 vms
```

Check if the Cluster “clusterTest” is compatible with the Storage Domain “iSCSIStorage”

```
.ls client=RHV-fd plugin="rhv:server=RHV.example.com \  
user=bacula password=alsogood" insecure=true \  
path=/api/compatibility/clone_storage/clusterTest[Name_cluster]/  
↪ iSCSIStorage[Name_storage]  
# Answer:  
# -rw-r----- 0 root root 0 1970-01-01 01:00:00 true
```

Get version of RHVM:

```
.ls client=RHV-fd plugin="rhv:server=RHV.example.com \  
user=bacula password=alsogood insecure=true" path=/api/version  
# Answer:  
# -rw-r----- 0 root root 0 1970-01-01 01:00:00 4.2.0
```

## 11.2 System operations

### Plugin System

These operations are designed to offer some functionalities without affecting the RHVM’s state. Among these are check connection with RHVM, create truststore automatically and encode password using same pattern as Bacula. The possible parameters are defined in table *Plugin Parameters*

#### Warning

Call plugin directly!

The plugin generates a new truststore:

```
java -jar rhv-backup.jar --truststore_file=/opt/bacula/etc/rhvOwn.truststore \  
--truststore_password=sosecret --log=./log_debugff.log --server=myrhvm.com \  
--hpassword=Mjk1M2QwRnPCncKT --operation=system --create_truststore=true  
# Result: Truststore file in /opt/bacula/etc/rhvOwn.truststore with password_  
↪ sosecret
```

#### Warning

Advanced use only

These allow control the disk request timeout, the number tries before reporting an error and more.

```
java -jar rhv-backup.jar --log=./log_debugff.log --server=myrhv.com \  
--hpassword=Mjk1M2QwRnPCncKT --operation=list --path=/datacenters/ \  
--num_tries_connection=10 --system_timeout_request=120
```

```
Fileset {  
  Name = ExampleFileset3  
  Include {  
    Options {  
      Signature = MD5
```

(continues on next page)

(continued from previous page)

```
}
Plugin = "rhv:server=rhv.example.com insecure=true
system_num_tries_connection=100 system_interval_request=1200 system_
↪timeout_request=2000
user=admin password=rhvpas123 target_template=template*"
}
}
```

## 11.3 Transferring Data

The Java program interacting with RHV can also be used to transfer data between the RHV infrastructure, in which case additional parameters are available and the `operation=list` parameter-value-pair is not used.

In this mode, the following general considerations apply:

- Either backup, restore, or list mode must be chosen by using appropriate options. Modes can not be mixed.
- Options have to start with double dashes, as in `--server`.
- The `--output` defines the output format and should be explicitly set to either `console` or `json`.
- These modes are not fully supported, but exist mostly to assist in trouble shooting with **Bacula Systems** support team. As such, they are not fully documented and should not be used for regular operations or maintenance of a RHV environment.

Some examples are provided below.

Backup of datacenter “OneDatacenter” configuration:

```
java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --output=console --debug=1 \
--server=rhv.example.com \
--truststore_file=/opt/bacula/etc/rhvm.truststore \
--truststore_password=sosecure --auth=http --profile=internal \
--user=bacula --password=alsogood \
--operation=backup \
--target_datacenter=OneDatacenter \
--target_configuration_only=true \
--path=/tmp/rhvPlugins/
```

Backup of all Virtual Machines whose names match the regular expression “vm.\*”, but excluding “vm1”. Note the quoting of the asterisk character to avoid the shell expanding it:

```
java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --output=console --debug=1 \
--server=rhv.example.com \
--truststore_file=/opt/bacula/etc/rhvm.truststore \
--truststore_password=sosecure --auth=http --profile=internal \
--user=bacula --password=alsogood \
--operation=backup \
--target_path=/vms/vm.* \
--target_exclude_vms=vm1 \
--path=/tmp/rhvPlugins/
```

Restoring the Virtual Machine “vm1”:

```
java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --output=console --debug=1 \
--server=rhv.example.com \
--truststore_file=/opt/bacula/etc/rhvm.truststore \
--truststore_password=sosecure --auth=http --profile=internal \
--user=bacula --password=alsogood \
--operation=restore \
--vm=vm1 \
--path=/tmp/rhvPlugins/
```

Restoring a VM, renaming disks:

```
java -jar /opt/bacula/lib/bacula-rhv-plugin.jar --output=console --debug=1 \
--server=rhv.example.com \
--truststore_file=/opt/bacula/etc/rhvm.truststore \
--truststore_password=sosecure --auth=http --profile=internal \
--user=bacula --password=alsogood \
--operation=restore --restore_vm=vm3 \
--vm_disk_names=65ccb526-30c0-4beb-b348-4395e70d6e32:vm3_restore_disks1,vm3_
↪restore_disks_general
--path=/tmp/rhvPlugins/
```

## 12 Troubleshooting

### 12.1 Bacula environment

This plugin stores the logs depending of the number of paralel executions it has. In other words, if the operations are only performed synchronously, it will always be sotred in the same file, under the suffix of '-0.log'. However, when a backup is running and another backup(or a restore) is started, the first backup will continue to store the log in the same file(under suffix of '-0.log'), but the one just started will be stored in the file under the suffix of '-1.log'.

The plugin indicates the log file in a message in job log. By default, the location of log file will be: */opt/bacula/working/rhv/rhv-debug-X.log*, but the user can modify this default value in file */opt/bacula/rhv\_backend*, or in each job with param *log*.

It's recommended that the log file always be in */opt/bacula/working/rhv/*.

### 12.2 RHV environment

The RHV Manager can return errors in some situations. The RHV REST API return code and messages do not always precisely reflect the cause of the problem. Given that general HTTP codes indicating rather broad error conditions are used, the plugin can not always react in a perfect manner. Instead, the plugin returns all available information about errors and warnings, but in many situations it will be necessary to check RHV logs to understand the cause of an operational error.

The most informative logs of RHV are usually *vdc-log.txt* and */var/log/ovirt-engine/engine.log* (for RHV Manager) and */var/log/vdsm/vdsm.log* (for virtualization hosts) since those reflect the main flow of the application and if any error occurs, it will generally be tracked in those logs.

Once the exact component that failed is known, it is possible to check the specific log file. The full list of RHV logs is described at [redhat.com/.../17587#RHEV\\_Logs\\_Overview](https://redhat.com/.../17587#RHEV_Logs_Overview)

Below we present some common errors and associated actions to resolve them.

#### Failed Transfer

If a disconnection happens when a disk image is being transferred by the ImageTransferService, the virtual disk remains in the state “Transferring via API” even if the backup job has terminated. This state blocks snapshot deletion. If a backup has suffered from non recoverable connection loss to the RHV Manager and this state remains for a virtual disk, a manual action on the RHV Manager can be performed to unblock the situation:

```
# log in to the RHV Manager shell, as root user
su postgres
psql -U engine -d engine
SELECT command_id FROM image_transfers;
# Get the command_id that is aborted
DELETE FROM image_transfers WHERE command_id='<UUID from above>';
# The state 'Transferring via API' will change to 'Locked' (2).
# Query the images table for locked images
SELECT image_group_id FROM images WHERE imagestatus = '2';
# Identify which one needs to be unlocked
UPDATE images SET imagestatus = '1' WHERE imagestatus = '2';
```

This will make the virtual disk inoperative, but it will allow either delete or commits the snapshots created previously.

In the RHV GUI, navigate to “Virtual Machines”, select the faulty one, select “Snapshots”, and perform a preview of a stable snapshot. Afterwards, “Commit” the action.

The plugin will automatically clear this situation with the following backup. This manual intervention is only necessary if there is no option of running a subsequent backup.

#### **DSM <host> command ExtendImageTicketVDS Failed**

The full message provided would be similar to

```
DSM <host> command ExtendImageTicketVDS failed: Image daemon request
failed: u'
status=404, code=404,
title=Not Found,
explanation=The resource could not be found.,
detail=No such ticket: a035901e-c3d6-4033-b0fa-8e127d422ccf,
reason=Not Found'
```

This error can be solved manually in the same way as the previous problem, and the plugin will also clear this situation with the following backup.

### **12.3 javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException**

Here, the full error message would look like

```
javax.net.ssl.SSLHandshakeException:sun.security.validator.ValidatorException:
  PKIX path building failed:
    sun.security.provider.certpath.SunCertPathBuilderException:
      unable to find valid certification path to requested target
at
sun.security.ssl.Alerts.getSSLException(Alerts.java:192)
at
sun.security.ssl.SSLSocketImpl.fatal(SSLSocketImpl.java:1959)
at
sun.security.ssl.Handshaker.fatalSE(Handshaker.java:302)
```

(continues on next page)

(continued from previous page)

```
at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:296)
...
```

This error is related to the certificates used by RHV. A possible solution is shown here: [stackoverflow.com/.../pkix-path-building-failed](https://stackoverflow.com/.../pkix-path-building-failed).

#### **org.apache.http.NoHttpResponseException: <RHVM>:443 failed to respond**

The full error is

```
org.apache.http.NoHttpResponseException: rhv.example.com:443 failed to respond
org.ovirt.engine.sdk4.Error
org.ovirt.engine.sdk4.Error: Failed to send request at
org.ovirt.engine.sdk4.internal.HttpConnection.send(HttpConnection.java:213)
at org.ovirt.engine.sdk4.internal.services.
  EventsServiceImpl$AddRequestImpl.send(EventsServiceImpl.java:83)
...
```

This error happens when no certificate is present. The solution can be found in the SSLHandshakeException problem resolution.

#### **HTTP Response Code is 409**

The full error is:

```
org.ovirt.engine.sdk4.Error: HTTP response code is "409".
  HTTP response message is "Conflict".
at org.ovirt.engine.sdk4.internal.services.ServiceImpl.
  throwError(ServiceImpl.java:113)
at org.ovirt.engine.sdk4.internal.services.ServiceImpl.
  checkFault(ServiceImpl.java:40)
```

This can happen if a snapshot could not correctly get deleted.

As explained in section **ch:fixlockedsnapshot**, this can be manually resolved by committing a snapshot preview in the RHV GUI.

The plugin will solve this situation manually with the following backup. A manual intervention is only necessary if there is no option of running that following backup.

#### **HTTP response is 401. HTTP response message is “Unauthorized”**

This error happens when RHVM credentials are incorrect.

#### **HTTP 503 Code at Disk Download**

This happens if the connection between RHV Manager and RHV virtualization host daemons is not working correctly. Usually this indicates a firewalling issue.

The solution is to ensure that the firewall service (firewalld) is active and correctly configured. For RHV operations, two ports—54321 for vdsmd and 54322 for ovirt-imageio— need to be accessible. To ensure this, execute the follow commands on RHV virtualization hosts:

```
systemctl start firewalld
firewall-cmd --permanent --add-port=54321/tcp
firewall-cmd --permanent --add-port=54322/tcp
```

To check if this command has been effective:

```
ssystemctl restart firewalld
firewall-cmd --list-ports
```

The ports 54321/tcp and 54322/tcp must appear in the list of open ports.

In a production environment, it is reasonable to limit access to the daemons affected only from trusted hosts or network segments. Please refer to RHEL documentation of the `firewall-cmd` program, which is available using `man firewall-cmd`.

## 12.4 VDSM hostTest command GetCapabilitiesVDS failed: Vds timeout occurred

This error can happen if a host is not registered. It shouldn't impact operations on other hosts.

To fix this situation it is usually necessary to confirm the host has been rebooted or shutdown and put it into maintenance mode after.

### Can't create any snapshot in a specific vm, but the vm can run

The origin of this error is unknown, this case is reported [here](#). The simple explanation is that an attempted snapshot removal (which we have no logs of), corrupted the chain.

#### See also

Best Practices for Hypervisors

## 13 Future Development Directions

Some features are planned for further development of this plugin:

- Backup and restore of RHV Manager Configuration and database.
- Implement an option to control Storage Domain overcommitment.

## 14 RHV Single Item Restore

Enterprise

#### Bacula Enterprise Only

This solution is **only** available for Bacula Enterprise. For subscription inquiries, please reach out to [sales@baculasystems.com](mailto:sales@baculasystems.com).

#### Important

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

This article presents how to use the RHV Single File Restore feature with **Bacula Enterprise** and the RHV Plugin.

## 14.1 Features Summary

The **Bacula Enterprise** RHV Single File Restore provides the following main features:

- Console interface and BWeb integration
- Support only for Full level jobs
- Support for Linux filesystems (ext3, ext4, btrfs, lvm, xfs)
- Support for Windows filesystems (FAT, NTFS)

### RHV SIR is available starting with Bacula Enterprise 12.2

This document will present solutions for **Bacula Enterprise** 12.2 and later, which are not applicable to prior versions. The RHV Single File Restore has been tested and is supported on RHEL 7.x, RHEL 8.x, Ubuntu Xenial, Debian Stretch and Debian Buster. SELinux is currently not supported.

### Warning

Only Full level backups are currently supported with RHV Single Item Restore.

## 14.2 Installation

Packages for the RHV Single File Restore plugin are available for supported platforms. Please contact Bacula Systems to get them.

Download the plugin package to your **Storage Daemon** server and then install using the package manager like so:

```
rpm -ivh bacula-enterprise-single-item-restore*.rpm
```

The package manager will ensure that your **Bacula Enterprise** version is compatible with the RHV Single File Restore plugin and will install dependencies. On RHEL, it will be needed to install `perl-JSON` package from **rpmforge** and the `libguestfs-winsupport` package.

### Note

On RHEL 7/8.x, it is necessary to install a custom version of the `libguestfs` packages from our repository to support NTFS devices. Those should not be updated with a newer version from official repositories. The YUM package manager has plugins to prevent package updates, try **yum-plugin-versionlock** or **yum-plugin-priorities**.

Additionally, the `ntfs-3g` package from the EPEL repository is needed for NTFS support. To install the EPEL repository, please follow the official instructions on the EPEL website to install the “epel-release” package here:

<https://fedoraproject.org/wiki/EPEL>

### Note

On RHEL 8.X and 9.x, you must have the the AppStream repository enabled to install the perl-File-Copy. The perl-File-Copy module is a dependency required by the bacula-enterprise-single-item-restore package.

Since Bacula Enterprise 16.0.13.

```
# cat /etc/yum.repos.d/dag.repo
[dag]
name = Bacula - RPMFORGE - DAG
baseurl = https://www.baculasystems.com/dl/DAG/rhel7-64
enabled = 1
protect = 0
gpgcheck = 0

# cat /etc/yum.repos.d/baculasystems.repo
[baculasystems-single_item_restore]
name = Bacula Enterprise - SIR
baseurl = https://www.baculasystems.com/dl/<xxx>/rpms/single-item-restore/
↪<version>/rhel7-64/
enabled = 1
protect = 0
gpgcheck = 0
```

Note: This last repository is required on RHEL7:

```
[baculasystems-dag-guestfish]
name = Bacula Enterprise - DAG for Guestfish
baseurl = https://www.baculasystems.com/dl/DAG/rhel7-64/guestfish/
enabled = 1
protect = 0
gpgcheck = 0
```

```
# yum install bacula-enterprise-single-item-restore
```

If BWeb Management Suite is used:

```
# service bweb restart
```

### 14.3 Notes about the “bacula” Account on RHEL

All commands in this document use the “bacula” unix account to run.

On RHEL, the Unix “bacula” account is locked by default. It means that it’s not possible by default to execute a command such as “su - bacula”.

It is possible to unlock the “bacula” account, or to use “sudo -u bacula” to execute commands. For example:

```
bacula@storage# /opt/bacula/bin/bconsole
```

Can be run from the root account using the following command:

```
root@storage# sudo -u bacula /opt/bacula/bin/bconsole
```

It is also possible to start a shell session using

```
root@storage# sudo -u bacula /bin/bash
```

Or unlock the “bacula” unix account and use “su -” with a command such as:

```
root@storage# chsh -s /bin/bash bacula
root@storage# su - bacula
bacula@storage# whoami
bacula
```

## 14.4 Fuse FileSystem

If a restore session is not properly cleaned up, some directories might still be mounted with the Bacula Fuse FileSystem.

```
baculaafs on /opt/bacula/working/cat-ro type fuse.baculaafs (ro,user=bacula)
backend0 on /opt/bacula/working/test-vm-0 type fuse.backend0 (ro,user=bacula)
/dev/fuse on /opt/bacula/working/test-vm type fuse (rw,nosuid,nodev,
↪user=bacula)
```

It is possible to unmount directories with the `fusermount -u` command.

```
bacula@storage# fusermount -z -u /opt/bacula/working/26
bacula@storage# fusermount -z -u /opt/bacula/working/test-vm-0
bacula@storage# fusermount -z -u /opt/bacula/working/test-vm
```

## 14.5 Samba SMB Shares

The **Bacula Enterprise** RHV Single File Restore plugin can automatically set up Samba SMB shares from the console program or the BWeb Management Suite.

To enable Samba SMB network shares, installing and configuring the “samba” package is mandatory. To configure the `/etc/samba/smb.conf` file correctly, you need to run `install-single-item-restore.sh` script.

```
root@storage# /opt/bacula/scripts/install-single-item-restore.sh install
Do you want to initialise Samba smb.conf [yes/No]: yes
Choose a Workgroup [BACULA]:

root@storage# cat /etc/samba/smb.conf
[global]
workgroup = BACULA
include = /etc/samba/conf.d/all
```

At this point, it is possible to modify `/etc/samba/smb.conf` to add your own configuration directives.

Network share descriptions will be stored in the directory `/etc/samba/conf.d`. It is possible to create and customize the template used by Bacula to generate configuration files.

```
root@storage# cat /etc/samba/conf.d/custom.tpl
[__share__]
```

(continues on next page)

(continued from previous page)

```
path = __path__
follow symlinks = yes
wide links = yes
writable = yes
```

## 14.6 Configuration

On the **Storage Daemon** host server, the `bconsole` program should be configured properly to let the “bacula” user connect to the Director with `/opt/bacula/etc/bconsole.conf`.

```
bacula@storage# /opt/bacula/bin/bconsole
Connecting to Director mydir-dir:9101
1000 OK: 10002 mydir-dir Version: 12.8.0
Enter a period to cancel a command.
* version
mydir-dir Version: 12.8.0 x86_64-redhat-linux-gnu
* quit
```

The package contains a script to test the connection with the Director and to test if the system can mount the *Bacula Virtual File System* properly.

```
bacula@storage# /opt/bacula/scripts/install-single-item-restore.sh check
I: Try to restart the script with sudo...
I: Found catalog MyCatalog
I: bacula-fused started on /tmp/bee-bfuse.XXXXX
I: MyCatalog found
I: 10 Client(s) found
I: /tmp/bee-bfuse.XXXXX unmounted
I: bacula-fused (rw) started on /tmp/bee-bfuse.XXXXX
I: MyCatalog found
I: 10 Client(s) found
I: /tmp/bee-bfuse.XXXXX unmounted
OK: All tests are good.
```

The *Bacula Virtual File System* is not designed to be used by end users to browse or restore files directly. If you try to access and browse the mount point, you may not see any files or files may have strange permissions, ownerships and sizes and will be inaccessible even to the root user.

## 14.7 Restore Scenario With Text Console Interface

The RHV Single File Restore plugin provides a simple console program that provides access to files inside VMs.

```
bacula@storage# /opt/bacula/bin/mount-vm
Automatically Selected Catalog: MyCatalog

Client list:
1: 127.0.0.1-fd
2: win2008-fd
3: rhel7-fd
Select a Client: 1
Selected Client: 127.0.0.1-fd
```

(continues on next page)

(continued from previous page)

```
Job list:
1: RHVSERVER.2021-02-15_19.12.51_34
2: RHVSERVER.2021-02-16_12.12.29_39
3: RHVSERVER.2021-02-16_12.37.54_03
4: RHVSERVER.2021-02-16_14.23.47_03
5: RHVSERVER.2021-02-16_15.45.32_03
6: RHVSERVER.2021-02-16_17.00.47_52
Select a Job: 6
Selected RHVSERVER.2021-02-16_17.00.47_52

Virtual Machine:
1: squeeze2
2: win2008
3: rhel7
4: sir-test-vm
Select a Virtual Machine: 4
Selected sir-test-vm

Actions list:
1: Mount guest filesystem locally
2: Export guest filesystem through SMB
3: Cleanup
Select a Actions: 1
Selected Mount guest filesystem locally

I: Files are available under /opt/bacula/working/mount-vm-6434/disks/sir-test-
↪vm
I: Press enter to finish and cleanup the session
```

In this step, the virtual machine filesystem is mounted locally (in the example above, files are available under `/opt/bacula/working/mount-vm-6434/disks/sir-test-vm`. It is possible to browse directories and copy files (with `cp`, `scp`, `ftp`) as with a standard filesystem from another terminal session with the Unix “root” and “bacula” accounts. If you need to use another Unix account to operate on files, use the “-o allow\_other” option when starting the `mount-vm` script.

```
bacula@storage# ls /opt/bacula/working/mount-vm-6434/disks/sir-test-vm
bin  dev  home      lib          media  opt   root  selinux  sys  usr  ↵
↪vmlinuz
boot  etc  initrd.img  lost+found  mnt    proc  sbin  srv      tmp  var
```

To clean up the session, just press “Enter” in the terminal session where the `mount-vm` script was started.

It is possible to limit the Job list with the following command line options:

- `-s=<days>` Limit the job list to the last *days*
- `-l=<number>` Limit the job list to the last *number* entries
- `-f=<filter>` Specify an advanced filter based on the Job name, the Fileset name or the JobId

```
# Limit the job output to the last 100 jobs
bacula@storage# /opt/bacula/bin/mount-vm -l 100
```

(continues on next page)

(continued from previous page)

```
# Limit the job output to the last 30 days
bacula@storage# /opt/bacula/bin/mount-vm -s 30

# Limit the job output to jobs that start with ``MyRHVServer''
bacula@storage# /opt/bacula/bin/mount-vm -f 'jobname=MyRHVServer*'

# BAD USAGE for the filter option, it will search for a job named
↳ ``MyRHVServer''
bacula@storage# /opt/bacula/bin/mount-vm -f 'jobname=MyRHVServer'

# Limit the job output to jobs that start with ``MyRHVServer''
# and that use the Fileset Test1
bacula@storage# /opt/bacula/bin/mount-vm -f 'jobname=MyRHVServer*
↳ fileset=Test1'

# Limit the job to the jobid XX
bacula@storage# /opt/bacula/bin/mount-vm -f jobid=XX
```

In some cases, the device detection doesn't work properly. It is possible to use the `-m` option to mount recognized disks in a simple way. The option is automatically set when only one disk is selected during the restore.

```
bacula@storage# /opt/bacula/bin/mount-vm -m
```

## 14.8 Notes

### Cache Directory

To speed up future RHV Single File restore sessions, some files that are generated during a restore session are kept in a cache directory.

```
bacula@storage# ls /opt/bacula/working/mount-cache
sir-test-vm-0.bmp  sir-test-vm-2.bmp  MyCatalog-2.idx  MyCatalog-5.idx  ↳
↳ MyCatalog-8.idx
sir-test-vm-1.bmp  sir-test-vm.profile  MyCatalog-4.idx  MyCatalog-6.idx  ↳
↳ MyCatalog-9.idx
```

It is possible to remove files in the cache after some time; they will be re-generated if needed.

### Support

The `install-single-item-restore.sh` script can collect traces automatically when a `mount-vm` session is running.

```
root@storage# /opt/bacula/scripts/install-single-item-restore.sh support
```

## 14.9 Limitations

- The RHV Single File Restore feature uses the Bacula BVFS interface to list files and directories. The Bacula BVFS interface is known to have some performance issues with MySQL catalog backend due to internal MySQL limitations with indexes on TEXT columns. For RHV Single Item Restore there should not be too much impact on performances (the backup structure is usually quite small) but we advise using the PostgreSQL backend for the best experience.

- The RHV Single File Restore performance may vary depending on various factors. For example, Bacula will have to read more data if the Volume was created with a large number of concurrent jobs.
- The Storage Daemon where the RHV Single File Restore is installed should have a CPU with the VT-x/EPT extensions. If these extensions are not available, the performance will be degraded. (From 20s to 10mins in our lab).
- RHEL 7/8 does not support mounting NTFS disks with the libguestfs provided with their system. To mount Microsoft NTFS disks on RHEL 7/8, it is required to install a patched version of the libguestfs packages. Please see notes in the “Installation” section of this document for more information.
- The RHV Single File Restore is compatible with *file based* devices (cloud, dedup, aligned, file, etc..). Tape devices are not supported.