Proxmox Plugin

Bacula Systems Documentation

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

Contents

Contents

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

This document aims at presenting the reader with information about the **Bacula Enterprise Proxmox Plugin**. The document briefly describes the target technology of the plugin, defines the scope of its operations, and presents its main features, including clusters.

1 Scope

The Proxmox Plugin was originally introduced with Bacula Enterprise version 10.0.

Our plugin is designed to work seamlessly with the latest releases of Proxmox VE. If you encounter any problems with your Proxmox VE version, reach out to Bacula Systems Support for assistance.

See also:

- Go to *Proxmox Features*
- Go to *Proxmox Installation*
- Go to *Proxmox Configuration*
- Go to *Proxmox Operations*
- Go to *Proxmox Backup and Restore Strategies*
- Go to *Proxmox Troubleshooting*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

2 Features

In order to deliver high quality solutions with reduced total cost of ownership, Bacula Enterprise represents the opportunity to do exactly this, by offering a particularly broad range of features. Its two Proxmox modules are just some of these many features.

Bacula has two different plugins for Proxmox. The first is a Proxmox module where FULL backups are supported, and both virtual machines and LXC containers can be backed up very efficiently. The second is a QEMU module, where FULL and INCREMENTAL backups of QEMU virtual machines are supported. DIFFERENTIAL backups are not supported yet. This module does not backup LXC containers.

Both modules provide fast, effective virtual machine bare metal recovery including QEMU and LXC guests.

Detailed list of features of the Proxmox module:

- Snapshot-based online backup of any guest VM including QEMU and LXC guests
- Full image-level backup
- Ability to restore complete virtual machine image
- Ability to restore QEMU VM archive (.vma) to an alternate directory
- Ability to restore LXC VM archive (.tar) and configuration to an alternate directory
- Ability to scan a *Proxmox cluster* to automatically create a backup configuration for each virtual machine

See also:

- Go back to *Proxmox Scope*
- Go to *Proxmox Installation*
- Go to *Proxmox Configuration*
- Go to *Proxmox Operations*
- Go to *Proxmox Backup and Restore Strategies*
- Go to *Proxmox Troubleshooting*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

3 Backup and Restore Strategies

3.1 Image Backup

With the image backup level strategy, the Bacula Enterprise Proxmox Plugin will save the Client disks as raw images for QEMU VMs and as a `tar` archives for LXC VMs, in the Proxmox context.

For this to work, it is not needed a Bacula File Daemon in each guest VM. The Bacula Proxmox Plugin will contact the Proxmox hypervisor to read and save the contents of virtual machine disks using snapshots (default behavior) and dump them using the Proxmox API.

Bacula does not need to walk through the Client filesystems to open, read, close and `stat` files, so it consumes less resources on the Proxmox infrastructure than a file level backup on each guest machine would. On the other hand, Bacula will also read and save useless data such as swap files or temporary files.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

The Proxmox Plugin will save not only the disk images of the guest VM, but also guest VM configurations which allows for very easy guest VM restores.

3.2 Installing Bacula Client on Each Guest

With this first strategy, you do not use the Bacula Enterprise Proxmox Plugin, but instead install a Bacula Enterprise File Daemon inside every virtual machine as if they were normal physical clients. In order to optimize the I/O usage on your Proxmox system, you will use Bacula's Schedules, Priorities, and Job concurrency settings to spread backup jobs throughout the backup window. Since all guest VMs could use the same storage on the Proxmox hypervisor, running all your backup jobs at the same time can create a bottleneck on the disk/network subsystem.

Installing a Bacula Enterprise File Daemon in each virtual machine permits you to manage your 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 or Directory exclusion (such as swap or temporary files).
- File level compression.
- Accurate backups.

See also:

- Go back to *Proxmox Scope*
- Go back to *Proxmox Features*
- Go back to *Proxmox Installation*
- Go back to *Proxmox Configuration*
- Go back to *Proxmox Operations*
- Go to *Proxmox Troubleshooting*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

4 Installation

This article describes how to install Bacula Enterprise Proxmox Plugin.

The Bacula File Daemon and its Proxmox Plugin need to be installed on the host of the Proxmox hypervisor which runs the guest VMs that are to be backed up. Proxmox uses a customized Debian distribution, so the Bacula Enterprise File Daemon for that platform has to be used.

Installation of the Bacula Enterprise Proxmox Plugin is most easily done by adding the repository file suitable for the existing subscription and the Debian version (the distributions package manager configuration) that Proxmox is built upon.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

4.1 Prerequisites

The **Plugin Directory** directive of the **File Daemon** resource in `/opt/bacula/etc/bacula-fd.conf` must point to where the `proxmox-fd.so` plugin file is installed. The standard Bacula plugin directory is `/opt/bacula/plugins`.

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

4.2 Steps

Installation example:

1. Insert the following content to the `/etc/apt/sources.list.d/bacula.list`:

```
#Bacula Enterprise
deb https://www.baculasystems.com/dl/@customer-string@/debs/bin/@version@/stretch-64/␣
↪stretch main
deb https://www.baculasystems.com/dl/@customer-string@/debs/proxmox/@version@/stretch-64/
↪ stretch proxmox
```

2. Run `apt-get update`.

```
apt-get update
```

3. Then, to install the Plugin, run: `apt-get install bacula-enterprise-proxmox-plugin`

```
apt-get install bacula-enterprise-proxmox-plugin
```

Manual installation of the packages can be done after downloading the right files from the Bacula Systems provided download area, and then using the package manager to install.

See also:

- Go back to *Proxmox Scope*
- Go back to *Proxmox Features*
- Go to *Proxmox Configuration*
- Go to *Proxmox Operations*
- Go to *Proxmox Backup and Restore Strategies*
- Go to *Proxmox Troubleshooting*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

5 Configuration

This article describes how to configure Bacula Enterprise Proxmox Plugin.

The plugin is configured using **Plugin Parameters** defined in a FileSets **Include** section of the Bacula Enterprise Director configuration.

More detailed information about Proxmox Plugin configuration:

5.1 General Parameters

The following Proxmox plugin parameter effects any type of Job (Backup, Estimation, or Restore).

abort_on_error[=<0 or 1>] specifies whether or not the plugin should abort execution (and the Bacula Job) if a fatal error occurs during a Backup, Estimation, or Restore operation. This parameter is optional. The default value is 0.

See also:

- Go to *Estimation and Backup Parameters*
- Go to *Restore Parameters*
- Go to *Fileset Examples*

Go back to *the main Proxmox Plugin configuration page*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solution page.

5.2 Estimation and Backup Parameters

These plugin parameters are relevant only for Backup and Estimation jobs:

vm=<name-label> specifies a guest VM name to backup. All guest VMs with a `<name-label>` provided will be selected for backup. Multiple `vm=...` parameters are allowed. If guest VM with `<name-label>` cannot be found, then a single job error will be generated, and the backup will proceed to the next VM unless `abort_on_error` is set, which will cause the backup job to be aborted. This parameter is optional.

vmid=<vmid> specifies a guest VM VMID to backup. Multiple `vmid=...` parameters may be provided. If a guest VM with `<vmid>` cannot be found, a job error will be generated and the backup will proceed to the next VM unless `abort_on_error` is set which will cause the backup job to be aborted. This parameter is optional.

include=<name-label-regex> specifies a list of a guest VM names to backup using regular expression syntax. All guest VMs with names matching the name regular expression provided will be selected for backup. Multiple `include=...` parameters may be provided. The match is performed case insensitive.

If no guest VMs are matching the name expression provided, the backup will proceed to the next parameters or finish successfully without backing up any VMs. The `abort_on_error` parameter will not abort the job when no guest VMs are found using name matching.

This parameter is optional.

exclude=<name-label-regex> specifies a list of guest VMs names which will be excluded from backup using regular expression matching. All guest VMs with names matching the provided regular expression, and selected for backup using the `include=...` parameter will be excluded. The match is performed case insensitive.

This parameter does not affect any guest VM selected to be backed up using `vm=...` or `vmid=...` parameters.

Multiple `exclude=...` parameters may be provided.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

This parameter is optional.

mode=<snapshot|suspend|stop> specifies the default backup mode to use. The **snapshot** mode will generate the live backup using snapshots which minimizes the downtime of a VM during the the backup process. The **suspend** mode will suspend the guest VM during a backup to achieve a consistent backup. The guest VM will remain suspended during backup and will resume running once the backup of this guest VM finishes. The **stop** mode will shut down the guest VM before backup and the VM will be restarted when its backup finishes.

This parameter is optional. If not set, then **snapshot** mode will be used by default.

vzdump_storage=<proxmox_storage> specifies the Proxmox storage resource `<proxmox_storage>` used by `vzdump` command during backup when the default *'local'* Proxmox storage resource has no backup content type available. The `<proxmox_storage>` should point to the Proxmox storage resource which has the valid backup content type added to the resource. See: `vzdumpstorage`. This parameter is optional. If not set, then a default *'local'* Proxmox storage resource will be used.

bwlimit=<N> specifies if the backup should be performed with an I/O bandwidth limitation (in KBytes per second) on the hypervisor side. This limitation is independent from Bacula's own bandwidth limitation feature, and is performed on a different level.

This parameter is optional. If not set, no bandwidth limitation on the hypervisor will be applied.

If none of the parameters `vm=...`, `vmid=...`, `include=...` and `exclude=...` are specified, all available guest VMs on the Proxmox hypervisor will be backed up.

See also:

- Go back to *General Parameters*
- Go to *Restore Parameters*
- Go to *Fileset Examples*

Go back to *the main Proxmox Plugin configuration page*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solution page.

5.3 Restore Parameters

During restore, the Proxmox Plugin will use the same parameters which were set for the backup job and saved in the Catalog. Some of them may be changed during the restore process if required.

storage: <storage> specifies a Proxmox Storage where restored guest VMs will be restored to. If not set, then a guest VM will be restored to the Proxmox Storage it was backed up from. If this parameter points to a nonexistent Storage, the original Storage of the guest VM will be used.

This parameter is optional.

pool: <resourcepool> specifies a Proxmox Resource Pool to which a restored guest VM will be attached. If not set, a restored guest VM will have no Resource Pool attached. When this parameter indicates a nonexistent pool, no pool will be attached.

This parameter is optional.

sequentialvmid: <yes or no> specifies what new VMID allocation procedure to use. If set to **yes**, the new guest VM will get the first available VMID based on the maximum VMID found on the Proxmox hypervisor. Otherwise, the restored guest VM will get a randomly generated VMID in a range of 10 above the highest used VMID found on the Proxmox hypervisor. This is the default.

Setting `sequentialvmid: yes` mitigates some conflicts which may occur during concurrent restores.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

See also:

- Go back to *General Parameters*
- Go back to *Estimation and Backup Parameters*
- Go to *Fileset Examples*

Go back to *the main Proxmox Plugin configuration page*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solution page.

5.4 Fileset Examples

Example 1

In the example below, all guest VMs will be backed up.

```
FileSet {
  Name = FS_ProxmoxAll
  Include {
    Plugin = "proxmox:"
  }
}
```

Example 2

In the example below, a single guest VM with name-label of "VM1" will be backed up.

```
FileSet {
  Name = FS_Proxmox_VM1
  Include {
    Plugin = "proxmox: vm=VM1"
  }
}
```

Example 3

The same example as above, but using **vmid** instead:

```
FileSet {
  Name = FS_Proxmox_VM1
  Include {
    Plugin = "proxmox: vmid=101"
  }
}
```

Example 4

In the following example, all guest VMs which contain "Prod" in their names will be backed up.

```
FileSet {
  Name = FS_Proxmox_ProdAll
  Include {
    Plugin = "proxmox: include=Prod"
  }
}
```

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

Example 5

In this final example, all guest VMs except VMs whose name begins with “Test” will be backed up.

```
FileSet {
  Name = FS_Proxmox_AllbutTest
  Include {
    Plugin = "proxmox: include=.* exclude=^Test"
  }
}
```

See also:

- Go back to *General Parameters*
- Go back to *Estimation and Backup Parameters*
- Go back to *Restore Parameters*

Go back to *the main Proxmox Plugin configuration page*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solution page.

See also:

- Go back to *Proxmox Scope*
- Go back to *Proxmox Features*
- Go back to *Proxmox Installation*
- Go to *Proxmox Operations*
- Go to *Proxmox Backup and Restore Strategies*
- Go to *Proxmox Troubleshooting*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

6 Operations

This article describes details regarding backup, restore, resource listing, and cluster support with Bacula Enterprise Proxmox Plugin.

6.1 Backup

The backup of a single guest VM consists of the following steps:

1. Save guest VM configuration (for LXC guest VMs).
2. Create a new backup snapshot (default), suspending or stopping the guest VM as requested.
3. Execute `vzdump` and save data.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

Backups can be performed for guest VMs in any power state (running or stopped). The Proxmox hypervisor will automatically create the required backup snapshot and remove it after the backup. Any other guest VMs snapshots will be unaffected.

The Proxmox Plugin will inform you about every guest VM backup start and finish:

```
 JobId 68: Start Backup JobId 68, Job=proxmox.2018-01-25_11.25.05_21
 JobId 68: Using Device "FileChgr1-Dev2" to write.
 JobId 68: Volume "Vol-0002" previously written, moving to end of data.
 JobId 68: proxmox: Start Backup vm: ubuntu-container (101)
 JobId 68: proxmox: Backup of vm: ubuntu-container (101) OK.
...
```

The backup will create a single (`.vma`) file for any QEMU guest VM and two files (`.conf` and `.tar`) for any LXC guest VM which is saved. Inside Bacula, those are represented as follows.

- `/@proxmox/qm/<name-label>/VM<vmid>.vma` for QEMU guest VMs
- `/@proxmox/lxc/<name-label>/VM<vmid>.conf` and
- `/@proxmox/lxc/<name-label>/VM<vmid>.tar` for LXC guest VMs

Multiple files will be created during a backup if multiple guest VMs are backed up with one job. However, note that backing up multiple VMs goes against hypervisor best practices. The distinct file names as shown above allow to locate the proper guest VM archive for restore.

Tip: You can exclude machine disks from Proxmox image backup in the Proxmox Console VM Disk Settings screen.

See also:

- Go to *Restore*
- Go to *Resource Listing*
- Go to *Cluster Support*

Go back to *the main Proxmox Plugin operations*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

6.2 Restore

The Proxmox Plugin provides two targets for restore operations:

- Restore to Proxmox hypervisor as new or original guest VM.
- Restore to a local directory as Proxmox archive files (`.vma` or the pair of `.tar` and `.conf` files).

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

Restore to Proxmox

To use this restore method, the `where=` parameter of a Bacula restore is used. The guest VM archive will be sent to the Proxmox hypervisor and restored as a new guest VM if the VMID of the restored VM is already allocated. Otherwise, the guest VM will be restored with its original Proxmox.

To restore a VM or VMs to a Proxmox hypervisor, the administrator should execute the restore command and specify the `where` parameter as in this example:

```
* restore where=/
```

and then set any other required restore plugin parameters for the restore.

In the following restore session example, the `sequentialvmid` plugin restore option is set to "yes":

```
* restore where=/
...
Run Restore job
JobName:         RestoreFiles
Bootstrap:       /opt/bacula/working/pve-dir.restore.1.bsr
Where:           /opt/bacula/archive/bacula-restores
Replace:         Always
FileSet:         Full Set
Backup Client:   pve-fd
Restore Client:  pve-fd
Storage:         File1
When:            2018-01-30 14:58:21
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 : proxmox: vmid=108 abort_on_error
Plugin Restore Options
storage:             *None*               (*Default locations*)
pool:                *None*               (*Default pool*)
sequentialvmid:      *None*               (*No*)
Use above plugin configuration? (yes/mod/no): mod
You have the following choices:
     1: storage (Storage location for restore)
     2: pool (Add the VM to the specified pool on restore)
```

(continues on next page)

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

(continued from previous page)

```
     3: sequentialvmid (Allocate new vmid in sequential order)
Select parameter to modify (1-3): 3
Please enter a value for sequentialvmid: yes
Plugin Restore Options
storage:             *None*              (*Default locations*)
pool:                *None*              (*Default pool*)
sequentialvmid:      yes                 (*No*)
Use above plugin configuration? (yes/mod/no):
```

The restore job log will indicate which guest VM is restored and which new guest VM was created:

```
JobId 76: Start Restore Job RestoreFiles.2018-01-25_13.50.31_29
JobId 76: Using Device "FileChgr1-Dev1" to read.
JobId 76: Ready to read from volume "Vol-0004" on File device "FileChgr1-Dev1" (/opt/
↪bacula/archive).
JobId 76: proxmox: VM restore: lxc/ubuntu-container/VM101 as VM222
JobId 76: End of Volume "Vol-0004" at addr=47137166325 on device "FileChgr1-Dev1" (/opt/
↪bacula/archive).
```

The new guest VM created during the restore will get a new VMID (if the original VMID is not available anymore) but the name/hostname will stay the same as it was with the original VM.

All other guest VM configuration parameters will be restored as they were backed up, including network MAC Addresses. For this reason, it is recommended to inspect and possibly update a guest VMs configuration before it is started. Otherwise, resource conflicts may arise.

New guest VMs will get a new VMID which will be allocated by the Bacula Proxmox Plugin as a random value in a range between the maximum VMID already allocated $+1$ and $+11$. This procedure is intended to mitigate possible resource allocation conflicts which could be occur when two or more guest VM restores or creations are executed at the same time, as Proxmox has no conflict resolution mechanism itself for this situation.

Using the plugin restore option `sequentialvmid` this behavior can be modified so that the newly restore guest VM gets the next available VMID, which will help limiting the range of VMID assigned over time, but will make concurrent restores unreliable.

The Storage target to be used for the restored guest VM disk(s) can be set using the **storage** plugin restore option. If this option is not set, all guest VM disks will be restored to their original Storage.

To list available Storages, a *listing mode* is available.

If an improper (e. g., non-existent) Storage is chosen for a restore, the restore process will create the guest VM in the Proxmox hypervisors default Storage.

Restore to Local Directory

It is possible to restore the guest VM data to a file and not to pass the data to the hypervisor. To do so, the``where`` restore option should point to a directory where the Proxmox plugin is installed:

```
* restore where=/tmp/bacula/restores
```

If the path does not exist, it will be created by the Bacula Proxmox Plugin.

Check the following example for the test “VM local restore”:

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

```
JobId 90: Start Restore Job RestoreFiles.2018-01-30_15.04.12_05
JobId 90: Using Device "FileChgr1-Dev1" to read.
JobId 90: Forward spacing Volume "Vol-0001" to addr=45406565308
JobId 90: proxmox: VM local restore: qm/ubuntu-server/VM108
```

The restore job log will show that the restore was done to a local directory.

See also:

- Go back to *Backup*
- Go to *Resource Listing*
- Go to *Cluster Support*

Go back to *the main Proxmox Plugin operations*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

6.3 Resource Listing

The Bacula Enterprise Proxmox Plugin supports the new Plugin Listing feature of Bacula Enterprise 8.x or newer. This mode allows a Plugin to display some useful information about available Proxmox resources such as:

- List of guest VM name-labels
- List of guest VM VMIDs
- List of Proxmox Storages
- List of Proxmox Resource Pools

The new feature uses the special `.ls` command with a new `plugin=<plugin>` parameter. The command requires the following parameters to be set:

client=<client> A Bacula Client name with the Proxmox Plugin installed.

plugin=<plugin> A Plugin name, which would be **proxmox:** in this case, with optional plugin parameters as described in section **genericparameters**.

path=<path> An object path to display.

The supported values for a `path=<path>` parameter are:

/ to display object types available to list.

vm to display a list of guest VM name-labels.

vmid to display a list of guest VM VMIDs and name-label pointers.

storage to show the list of available Storages.

pool to display the list of Resource Pools.

To display available object types, follow the following command example:

```
*.ls client=pve-fd plugin=proxmox: path=/
Connecting to Client pve-fd at pve:9102
drwxr-x---   1 root     root               0 2018-01-30 15:08:08  vm
drwxr-x---   1 root     root               0 2018-01-30 15:08:08  vmid
drwxr-x---   1 root     root               0 2018-01-30 15:08:08  storage
```

(continues on next page)

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

(continued from previous page)

```
drwxr-x---   1 root     root               0 2018-01-30 15:08:08  pool
2000 OK estimate files=4 bytes=0
```

To display the list of all available guest VMs, the following command example can be used:

```
*.ls client=pve-fd plugin=proxmox: path=vm
Connecting to Client pve-fd at pve:9102
-rw-r-----   1 root     root      2251187813 2018-01-30 15:08:49  ubuntu-container
-rw-r-----   1 root     root       594332876 2018-01-30 15:08:50  fedora-container
-rw-r-----   1 root     root      1288490188 2018-01-30 15:08:52  openhab-test
-rw-r-----   1 root     root       680735539 2018-01-30 15:08:53  vm1
-rw-r-----   1 root     root       490733568 2018-01-30 15:08:54  testvm1
-rw-r-----   1 root     root     34359738368 2018-01-30 15:08:54  ubuntu-server-template
-rw-r-----   1 root     root     34359738368 2018-01-30 15:08:55  rhel-server
-rw-r-----   1 root     root     68719476736 2018-01-30 15:08:56  testvm2
-rw-r-----   1 root     root     34359738368 2018-01-30 15:08:56  ubuntu-server
-rw-r-----   1 root     root     34359738368 2018-01-30 15:08:57  vm2
2000 OK estimate files=10 bytes=211,463,910,192
```

To display the list of guest VM VMIDs, use the following command example:

```
*.ls client=pve-fd plugin=proxmox: path=vmid
Connecting to Client pve-fd at pve:9102
 root     root      2251187813 2018-01-30 15:09:37  101 -> ubuntu-container
 root     root       594332876 2018-01-30 15:09:38  102 -> fedora-container
 root     root      1288490188 2018-01-30 15:09:40  103 -> openhab-test
 root     root       680735539 2018-01-30 15:09:41  106 -> vm1
 root     root       490733568 2018-01-30 15:09:42  107 -> testvm1
 root     root     34359738368 2018-01-30 15:09:42  100 -> ubuntu-server-template
 root     root     34359738368 2018-01-30 15:09:43  104 -> rhel-server
 root     root     68719476736 2018-01-30 15:09:43  105 -> testvm2
 root     root     34359738368 2018-01-30 15:09:44  108 -> ubuntu-server
 root     root     34359738368 2018-01-30 15:09:45  201 -> vm2
2000 OK estimate files=10 bytes=211,463,910,192
```

The VM and VMID lists display an estimated size of the guest VM.

To display available Proxmox Storages, the following command example can be used:

```
*.ls client=pve-fd plugin=proxmox: path=storage
Connecting to Client pve-fd at pve:9102
brw-r-----   1 root     root               0 2018-01-30 15:11:18  local-lvm
brw-r-----   1 root     root               0 2018-01-30 15:11:18  local
brw-r-----   1 root     root               0 2018-01-30 15:11:18  sunnfs
2000 OK estimate files=3 bytes=0
```

And, finally, use the following to show available Proxmox Resource Pools:

```
*.ls client=pve-fd plugin=proxmox: path=pool
Connecting to Client pve-fd at pve:9102
brw-r-----   1 root     root               0 2018-01-30 15:12:27  testpool
brw-r-----   1 root     root               0 2018-01-30 15:12:27  Development
brw-r-----   1 root     root               0 2018-01-30 15:12:27  Production
```

(continues on next page)

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

(continued from previous page)

```
brw-r-----   1 root     root               0 2018-01-30 15:12:27  Sales
brw-r-----   1 root     root               0 2018-01-30 15:12:27  Marketing
2000 OK estimate files=5 bytes=0
```

See also:

- Go back to *Backup*
- Go back to *Restore*
- Go to *Cluster Support*

Go back to *the main Proxmox Plugin operations*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

6.4 Cluster Support

Important: Since Bacula version 16.0.7, a new solution has been introduced to replace the scan_proxmox_cluster tool. It is highly recommended to use the new solution - Automatic Object Integration (Scan Plugin) as the scan_proxmox_tool will soon be deprecated. See an example for Proxmox.

At a regular interval, Bacula can contact a Proxmox cluster member, list all virtual machines that are hosted, and adapt the configuration dynamically.

The `scan_proxmox_cluster` tool is used to analyze a Proxmox cluster and create individual FileSets and Backup Jobs for each virtual machine or container found. The virtual machines selected by this tool may be based on the virtual machine names, or by using regular expressions.

```
cluster: cluster1
node:    proxmox1
VM:      debian1    -> Job=j_cluster1_debian1
                       FileSet=fs_cluster1_debian1
                       Client=proxmox1
```

The Job's Client directive will be set to the Proxmox cluster node member.

If a virtual machine is no longer detected, the Job resource will be disabled or removed from the configuration.

If a virtual machine is on an other member of the cluster, the Job Client directive will be adapted automatically.

To learn more detailed information about cluster support, see:

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

Cluster Support Installation

Bacula's Proxmox cluster support requires the installation of the library `Net::Proxmox::VE` which is available at: http://github.com/mrproper/proxmox-ve-api-perl

The Bacula Enterprise DAG repository includes a package for this library named `perl-Net-Proxmox-VE` for Redhat.

On Redhat, the Proxmox library needs to have the package `perl-LWP-Protocol-https` installed, otherwise the message `Login failed. Protocol scheme 'https' is not supported` will be displayed.

See also:

- Go to *Cluster Support Configuration*

Go back to *the main Proxmox Plugin operations*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

Cluster Support Configuration

Bacula's Proxmox cluster support is available with BWeb Management Console.

In this example, we have a cluster named `cluster1` with 3 nodes, `prox1`, `prox2` and `prox3`. 5 virtual machines are defined, `vm1`, `vm2`, `vm3`, `vm4`, `vm5`.

The following steps are needed to activate the support:

- Install a Bacula Enterprise FileDaemon on each node of the Proxmox cluster (prox1, prox2 and prox3)
- Install and configure BWeb Enterprise Management Configuration Module (Configuration -> Configure Bacula -> Complete the Configuration Wizard)
- Define one Client resource per cluster member in the Director configuration file `bacula-dir.conf`. For simplicity, the Bacula Client name can match the DNS Promox server name (example "prox1").
- Define a JobDefs with the parameters shared by the Proxmox jobs (Schedule, Storage, Priority, . . .)
- Configure Proxmox access in `/opt/bacula/etc/pve.conf` (click *here* for more information)

```
# chown bacula /opt/bacula/etc/pve.conf
# chmod 600 /opt/bacula/etc/pve.conf
# cat /opt/bacula/etc/pve.conf
[default]
username=root
password=password_of_root
host=prox1
skip_certificate
realm=pam
```

- Execute the `scan_proxmox_cluster` as the unix user `bacula`

```
# /opt/bweb/bin/scan_proxmox_cluster --cluster cluster1 --node prox1 --jobdefs␣
↪ProxmoxJobs
INFO Using Job Client directive to prox1
tar: Removing leading `/' from member names
INFO Doing a backup of the previous configuration tree in bacula-etc.2019-05-06_
↪10:15.tar.gz
INFO Job Modification Summary:
```

(continues on next page)

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

(continued from previous page)

```
Added:
- j_cluster1_vm1
- j_cluster1_vm2
- j_cluster1_vm3
- j_cluster1_vm4
- j_cluster1_vm5

Disabled:

Existing:

Removed:
```

The `cluster` option is used to name the resources that will be created. The `node` option is used to select the virtual machines to create. The list of the changes will be displayed in the output of the script. In the BWeb Management Configuration Module, a workset for the Proxmox cluster will be displayed with the list of the changes to apply to the configuration. The option `--commit_and_reload` can activate the changes automatically.

```
scan_proxmox_cluster --cluster cluster1 --node prox1 --jobdefs ProxmoxJobs --commit_
↪and_reload
```

- Schedule the `scan_proxmox_cluster` for each node of the Proxmox cluster at a regular interval with a Bacula Admin Job for example.

It is possible to set various options for the Job or the FileSet.

If the Proxmox node name is not equal to the Bacula Client name, it is possible to specify `--directive Client=name` in the command line.

Example:

```
scan_proxmox_cluster --cluster cluster1 --node prox1 --jobdefs DefaultJob --directive␣
↪Client=prox1-fd
```

Example of a complete configuration will be:

```
#### Configuration for the Cluster
Client {
  Name = prox1
  Password = xxx
  Address = prox1
  File Retention = 5 years
  Job Retention = 5 years
  Catalog = MyCatalog
}
Client {
  Name = prox2
  Password = xxx
  Address = prox2
  File Retention = 5 years
  Job Retention = 5 years
  Catalog = MyCatalog
}
```

(continues on next page)

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

(continued from previous page)

```
JobDefs {
  Name = PromoxJobs
  Priority = 10
  Type = Backup

  Storage = File
  Schedule = AtNight
  Pool = Default
  Messages = Standard
}
Job {
  Name = UpdateCluster1
  Type = Admin
  JobDefs = ProxmoxJobs
  Schedule = BeforeNight
  RunBeforeJob = "/opt/bweb/bin/scan_proxmox_cluster --cluster cluster1 --node prox1 --
↪commit_and_reload --jobdefs ProxmoxJobs"
  RunBeforeJob = "/opt/bweb/bin/scan_proxmox_cluster --cluster cluster1 --node prox2 --
↪commit_and_reload --jobdefs ProxmoxJobs"
  RunBeforeJob = "/opt/bweb/bin/scan_proxmox_cluster --cluster cluster1 --node prox3 --
↪commit_and_reload --jobdefs ProxmoxJobs"

}
####################################################################
#### The following example configuration snippet was generated by the scan_proxmox_
↪cluster program
FileSet {
  Name = fs_pve_vm1
  Include {
    Plugin = "proxmox: vm=vm1"
  }
}
Job {
  Name = j_pve_vm1
  Client = prox1
  FileSet = fs_pve_vm1
  JobDefs = ProxmoxJobs
}
FileSet {
  Name = fs_pve_vm2
  Include {
    Plugin = "proxmox: vm=vm2"
  }
}
Job {
  Name = j_pve_vm2
  Client = prox1
  FileSet = fs_pve_vm2
  JobDefs = ProxmoxJobs
}
...
```

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

Command Line Options

- `--pvefile file` Get credentials from a file (default: /opt/bacula/etc/pve.conf)
- `--profile=name` Profile name to use in the pvefile
- `--username string` Proxmox server username
- `--password string` Proxmox server password
- `--host string` Proxmox server host
- `--realm` pam or pve
- `--director string` Director name. If not provided, the first Director is used
- `--jobdefs string` JobDefs resource name
- `--job 'job_%v'` Jobs resource name pattern (
- `--fileset 'fs_%v'` FileSets resource name pattern (
- `--directive key=value` (storage|client|schedule . . .)=value definition
- `--fs_option key=value` FileSet options such as: signature, compression . . .
- `--plugin_option key=value` FileSet plugin options such as abort_on_error, index . . .
- `--vm value` Virtual machine to backup (don't use with vms_(include|exclude))
- `--pool value` Pool value to backup
- `--node value` Node to backup
- `--type value` Host type (lxc, qemu)
- `--vms_include pattern` Regular expression to include virtual machines
- `--vms_exclude pattern` Regular expression to exclude virtual machines
- `--commit_and_reload` Commit changes and reload just after resources are created
- `--description` New jobs' description
- `--remove_jobs` Remove jobs instead of disabling them
- `--json` Print output as JSON string

Storing Proxmox Credentials on Disk

The `--pvefile` argument can refer to a configuration file stored on disk. This file may contain the information used to connect to the Proxmox Cluster. It is possible to store multiple Proxmox server profiles and reference them with the `--profile` option.

- `username=string` Proxmox server username (default "root")
- `password=string` Proxmox server password
- `host=string` Proxmox server host
- `realm=string` pam or pve
- `cluster=string` Name of the cluster (used to generate resources)
- `skip_certificate` Do not check SSL certificates
- Other parameters used in pve_get_address program

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

- `failback=string` Address returned when the virtual machine is not found
- `base=string` Used to determine the Virtual Machine name from a pattern

```
# cat /opt/bacula/etc/pve.conf
[cluster1]
password=pass1
host=prox1
realm=pam
skip_certificates
cluster=cluster1

[cluster2]
password=pass3
host=prox3
realm=pam
skip_certificate
cluster=test
```

See also:

- Go back to *Cluster Support Installation*

Go back to *the main Proxmox Plugin operations*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

See also:

- Go back to *Backup*
- Go back to *Restore*
- Go back to *Resource Listing*

Go back to *the main Proxmox Plugin operations*.

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

See also:

- Go back to *Proxmox Scope*
- Go back to *Proxmox Features*
- Go back to *Proxmox Installation*
- Go back to *Proxmox Configuration*
- Go to *Proxmox Backup and Restore Strategies*
- Go to *Proxmox Troubleshooting*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

7 Troubleshooting

If you encounter a following error during backup job:

```
Fatal error: proxmox: Error closing backend. Err=Unknown error during program execvp
```

You should check the Proxmox Task log for the following error:

```
TASK ERROR: could not get storage information for 'local':
       can't use storage 'local' for backups - wrong content type
```

Then your 'local' Proxmox storage resource is missing the required 'backup' content type. This kind of configuration is expected by `vzdump` Proxmox command to save a backup task log and guest vm configuration. As a remedy you can add a missing 'backup' content type to the 'local' Proxmox storage resource or configure your backup job to use a different Proxmox storage resource for that. In the latter case you should add a `vzdump_storage=...` parameter to your configuration.

See also:

- Go back to *Proxmox Scope*
- Go back to *Proxmox Features*
- Go back to *Proxmox Installation*
- Go back to *Proxmox Configuration*
- Go back to to *Proxmox Operations*
- Go back to *Proxmox Backup and Restore Strategies*
- Go to *Proxmox Limitations*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

8 Limitations

This article presents limitations of the Proxmox Plugin.

- Granular restore (*Single Item Restore*) is not available yet. A file level backup of the VM with the Bacula FD inside the Guest VM is needed to enable single file restores of a Proxmox guest VM. Read more *here*.
- Concurrent backups of the same guest VM are not possible.
- Only Full level backups are possible. This is a Proxmox limitation as its API does not provide methods suitable for other backup levels. This limitation is described in details in the *Features* chapter, which also describes another module, QEMU that is free of that limitation.
- VM templates available on the Proxmox system can not be backed up. This is also a Proxmox limitation.
- In listing mode with the **vm** or **vmid** parameters, the plugin will display both guest VMs and VM templates. This limitation will be removed in the future.
- Backup of LXC created in **unprivileged** mode could fail with **Permission denied** error in Proxmox Task execution log and a following job messages error log:

  ```
  Error: proxmox: Error closing backend. Err=Unknown error during program execvp
  ```

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.

This is a known issue which will be removed in future release of the plugin.

See also:

- Go back to *Proxmox Scope*
- Go back to *Proxmox Features*
- Go back to *Proxmox Installation*
- Go back to *Proxmox Configuration*
- Go back to *Proxmox Operations*
- Go back to *Proxmox Backup and Restore Strategies*
- Go back to *Proxmox Troubleshooting*

Go back to *the main Proxmox Plugin page*.

Go back to the main Dedicated Backup Solutions page.

Copyright © 2024 Bacula Systems. All trademarks are the property of their respective owners.