



Hyper-V Plugin

Bacula Systems Documentation

Contents

1	Hyper-V	2
2	Hyper-V WMI plugin	12

Contents

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

Bacula Enterprise offers two ways to backup your Hyper-V virtual machines:

1 Hyper-V

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

- *Hyper-V Plugins to Cover All Backup Needs*

1.1 Hyper-V Plugins to Cover All Backup Needs

Microsoft has created various technologies for backing up Hyper-V virtual machines. Thus, multiple Bacula Plugins have been designed to maximize the benefits of each solution.

Hyper-V is equipped with a VSS writer on all compatible versions of Windows Server. This VSS writer enables developers to utilize the existing VSS infrastructure for backing up virtual machines to Bacula using the Bacula Enterprise VSS Plugins. This technology is supported by the original **Bacula Hyper-V plugin**. Although it doesn't allow incremental and differential backups and other more granular VM-based options, it can still cover any Hyper-V version. Therefore, it is highly recommended for small standalone Hyper-V servers (single node, no Failover Cluster) and older Hyper-V versions that do not offer Virtual Disk Service or Hyper-V WMI API.

The Virtual Disk Service (VDS) is a service provided by Microsoft Windows that handles query and configuration tasks upon request from end users, scripts, and applications. This service is compatible with Windows Server 2003, Windows Vista, and newer versions. The **Hyper-V Winapi Plugin** utilizes this technology to backup and restore virtual machines. It supports incremental and differential backups, making it the recommended solution for more intricate Hyper-V servers, such as Failover Clusters with multiple nodes, where local disk space is a critical resource. Depending on the relocation of virtual machines within the Cluster, it may be necessary to migrate the backed-up VMs across specific nodes.

Starting in Windows 8 and Windows Server 2012, Hyper-V supports backup via the Hyper-V WMI API. This feature enables individual Guest VMs to be backed up separately and incrementally, offering a more scalable solution compared to using VSS in the host. The Bacula Enterprise **Hyper-V WMI Plugin** uses this technology for backup and restore to/from Bacula. It backups/restores VM in the recommended Microsoft format. By utilizing the Microsoft snapshot format from/to disk, it is essential to have sufficient disk space available for the process to proceed smoothly. In

scenarios where disk space may be limited due to busy configurations, the Hyper-V Winapi Plugin can be utilized as an alternative solution.

Backups created using the **Hyper-V WMI Plugin**, **Hyper-V Winapi Plugin** and **Hyper-V Plugin** are not compatible with each other.

Overview

This white paper presents how to use the Microsoft Hyper-V Server plugin when backing up with **Bacula Enterprise** version 8.2. These solutions are not applicable to prior versions. This document is intended to be used by **Bacula Enterprise** administrators.

Bacula Windows Hyper-V Plugin

Bacula Systems provides a single plugin for Bacula Enterprise named `vss-fd.dll` that permits you to backup a number of different components on Windows machines. One of those components is Microsoft Hyper-V Server, which is the subject of this white paper.

Backing up and restoring Hyper-V virtual machine is supported with Full level backups. It is not possible to do Incremental or Differential backups because Microsoft does not support that backup level for the Hyper-V Server product. Use of the Global Endpoint Deduplication plugin and the `bothsides FileSet` option permits to minimize the data transfer and the storage.

Installation and Configuration

To activate the Hyper-V component you have to put the following into the Include section of the File Set which will be used to back up the Hyper-V Server:

```
Plugin = "vss:/@HYPERV/"
```

This will back up all Hyper-V Virtual Machine. The plugin directive must be specified exactly as shown above. A Job may have one or more of the `vss` plugin components specified.

You must ensure that the `vss-fd.dll` plugin is in the plugins directory on the FD doing the backup (done by default with the installer), and that the `Plugin Directory` directive line is present and enabled in the FD's configuration file `bacula-fd.conf`.

An example of the FD configuration file is shown in the screenshot below:

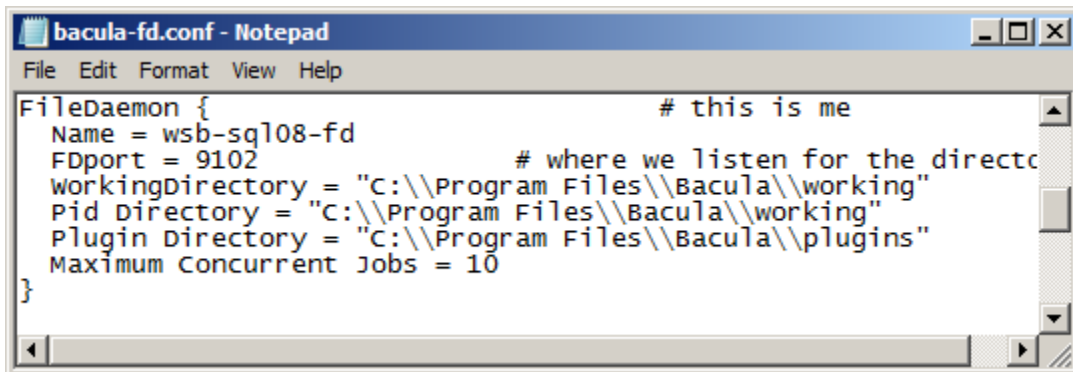


Fig. 1: File Daemon Configuration Excerpt with “Plugin Directory” Line

The status output of a client with the VSS plugin enabled:

```
*status client=wsb-sql08-fd
Connecting to Client wsb-sql08-fd at wsb-sql08:9102

wsb-sql08-fd Version: 8.2.0 (02 Feb 2015) VSS Linux Cross-compile Win64
Daemon started 20-Apr-12 13:14. Jobs: run=15 running=0.
Microsoft Windows Server 2008 R2 Standard Edition Service Pack 1 (build 7601), 64-bit
Heap: heap=0 smbytes=1,061,455 ...
Sizes: boffset_t=8 size_t=8 debug=0 ...
Plugin: vss-fd.dll
```

Backup

If everything is set up correctly as above then the backup will include the Hyper-V server data. The Hyper-V server data files backed up will appear in a **bconsole** or **bat** restore like:

```
/@HYPERV/
...
etc
```

A complete example of a FileSet and Job resource for Hyper-V Server data is shown below. As for all VSS-enabled components, it is the administrator's responsibility to make sure that the required VSS snapshots are created by explicitly mentioning at least one file or directory for each drive where data that is handled by the plugin is stored. In the example, we use the file `c:/backmeup` to ensure this.

```
FileSet {
  Name = HYPERV
  Include {
    Options {
      Signature = SHA1
      Dedup = bothsides
    }
    File = C:/backmeup
    Plugin = "vss:/@HYPERV/"
  }
}

Job {
  Name = HYPERV08
  Accurate = Yes
  File Set = HYPERV
  Client = wsb-hyp08-fd
  Job Defs = DefaultJob
  Level = Full
}
```

Note in the example above that `C:/backmeup` is explicitly included, which is required to ensure that **Bacula** creates the required VSS snapshot of that Windows drive letter. If Hyper-V Server data is also stored on other partitions, you need to create similar `File` lines for these drives, too.

Note: Starting with Bacula Enterprise version 12.6, the explicit include of a dummy file (see `File = C:/backmeup`)

in the fileset example above) is not mandatory anymore

```
File Set {
  Name = HYPERV-TestVM
  Include {
    Options {
      Signature = SHA1
    }
  }
  File = C:/backmeup
  # backup only TestVM on the server
  Plugin = "vss:/@HYPERV/ cinclude=\"Host Component\" cinclude=*/TestVM cexclude=*"
}
}
```

Hyper-V uses one of two mechanisms to back up each VM. The default backup mechanism is called the “Saved State” or “Offline” method, where the VM is put into a saved state during the processing of the PrepareForSnapshot event, snapshots are taken of the appropriate volumes, and the VM is returned to the previous state during the processing of the PostSnapshot event.

The other backup mechanism is called the “Child VM Snapshot” or “Online” method, which uses VSS inside the child VM to participate in the backup. For the “Child VM Snapshot” method to be supported, all of the following conditions must be met:

- Backup (volume snapshot) Integration Service is installed and running in the child VM. The service name is “Hyper-V Volume Shadow Copy Requestor”.
- The child VM must be in the running state.
- The Snapshot File Location for the VM is set to be the same volume in the host operating system as the VHD files for the VM.
- All volumes in the child VM are basic disks and there are no dynamic disks.
- All disks in the child VM must use a file system that supports snapshots (for example, NTFS).

To know if your VMs are “Offline” or “Online”, it is possible to use the following windows command on Windows 2012 R2:

```
C:/> echo list writers > t.txt
C:/> diskshadow /s t.txt | find "Caption: 0"
           - Caption: Offline/2012
           - Caption: Offline/windows
           - Caption: Online/centos
```

On Windows 2012 and 2008

```
C:/> echo list writers > t.txt
C:/> diskshadow /s t.txt | find /i "Caption: Backup Using"
```

- For Offline backups: Backup Using Saved State/*VMname1*
- For Online backups: Backup Using Child Partition Snapshot/*VMname2*

Restore

Restoring the VMs is done entirely by the host operating system; the VSS writers in the child VMs are not involved.

- During the processing of the PreRestore event, the Hyper-V VSS writer turns off and deletes any VMs that are about to be restored.
- After all VSS writers have processed the PreRestore event, the files are restored.
- For each VM that was restored, the Hyper-V VSS writer registers the VM with the Hyper-V management service. If the VM is restored to a nondefault location, a symbolic link is created in the default location linking to that location.
- For each VHD that was restored, the location is compared with the one specified for that VM. If the location is different, then the configuration is updated with the proper location.
- The network configuration is updated. If the virtual switches that the VM was connected to when it was backed up still exist, new ports are created and connected to the VM.

When restoring a “Offline” VM, the VM will not be re-created by Microsoft Hyper-V vss driver. It is possible to run “New-VM” powershell command to re-create the VM.

```
New-VM -VMName centos -VHDPath C:/VM/centos.vhdx -MemoryStartupBytes 512MB -SwitchName.
↪ VMNetwork
```

without_vss

With Bacula Enterprise 8.2, it is possible to restore VSS files directly on disk without using the VSS restore framework. In the restore menu, it is possible to configure Plugin Options menu and set the without_vss option to “true”.

```
Run Restore job
JobName:      RestoreFiles
Bootstrap:    /opt/bacula/working/trusty-amd64-dir.restore.9.bsr
Where:        c:/tmp
Replace:      Always
FileSet:      Full Set
Backup Client: hyperv
Restore Client: hyperv
Storage:      dedup
When:         2015-03-03 06:50:22
Catalog:      MyCatalog
Priority:      10
Plugin Options: *None*          <----- Plugin Options menu
```

Example

We assume that a correct backup of Hyper-V data exists and you start the restore with option 5 of the `bconsole restore` command, mark the complete tree of data backed up by the Hyper-V component of the VSS plugin, then finally do `lsmark @HYPERV` to show all the files selected to be restored:

```
$ mark *
31 files marked.
$ lsmark
*@HYPERV/
```

(continues on next page)

```
*Microsoft Hyper-V VSS Writer/  
  *Host Component/  
    *:component_info_5215da3c  
    *c:/  
      *programdata/  
        *microsoft/  
          *windows/  
            *hyper-v/  
              *initialstore.xml  
              *resource types/  
                *06ff76fa-2d58-4baf-9f8d-455773824f37.xml  
                *118c3be5-0d31-4804-85f0-5c6074abea8f.xml  
                *146c56a0-3546-469b-9737-fc9cf82428f4.xml  
                *dacdcf3f-6f67-4eb8-a4d0-5d93b48a2468.xml  
                *f6293891-f32f-4930-b2db-1a8961d9cb75.xml  
  
*Offline/  
  *ubuntu/  
    *:component_info_5215da3c  
    *c:/  
      *programdata/  
        *microsoft/  
          *windows/  
            *hyper-v/  
              *virtual machines/  
                *690f5094-ff23-411e-92c0-639fc7ebc598/  
                *690f5094-ff23-411e-92c0-639fc7ebc598.bin  
                *690f5094-ff23-411e-92c0-639fc7ebc598.vsv  
                *690f5094-ff23-411e-92c0-639fc7ebc598.xml  
  
    *vm/  
      *ubuntu.vhdx
```

```
$ lsmark  
*@HYPERV/  
  *Microsoft Hyper-V VSS Writer/  
    *Host Component/  
      *:component_info_5216cf46  
      *c:/  
        *programdata/  
          *microsoft/  
            *windows/  
              *hyper-v/  
                *initialstore.xml  
                *resource types/  
                  *06ff76fa-2d58-4baf-9f8d-455773824f37.xml  
                  *118c3be5-0d31-4804-85f0-5c6074abea8f.xml  
  
*Online/  
  *centos/  
    *:component_info_5216cf46  
    *c:/  
      *programdata/  
        *microsoft/  
          *windows/
```

```

*hyper-v/
  *snapshots/
    *acc145fb-9566-402d-9434-04f1e325a75f-backupsnapshot.xml
  *virtual machines/
    *acc145fb-9566-402d-9434-04f1e325a75f.xml
*vm/
  *centos-childvhd.avhdx
  *centos.vhdx

```

File Level Restore

To restore a set of files from a Hyper-V VM backup without re-importing the entire VM, it is possible to restore VHD files in a directory using the `without_vss` plugin restore option (See sec *without_vss*) and mount them in the system with the Powershell command `Mount-VHD` (or the Server Manager console (see Fig *Attach/Mount Option in Server Manager* below)). Once mounted, the VHD image is accessible like other physical disks on the system.

```
Mount-VHD -Path c:\test\testvhdx.vhdx -ReadOnly
```

More information about the `Mount-VHD` command can be found here:

<https://technet.microsoft.com/en-us/library/hh848551.aspx>

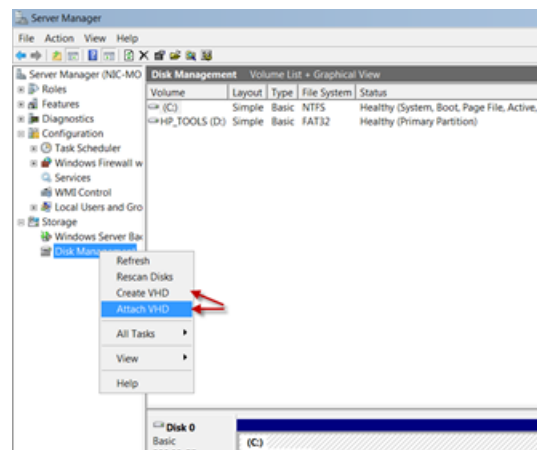


Fig. 2: Attach/Mount Option in Server Manager

We advise to restore VHD files on a different system to avoid operational problems during the restore. If the `without_vss` option is not properly set, the original VM would be deleted by Hyper-V automatically during the restore.

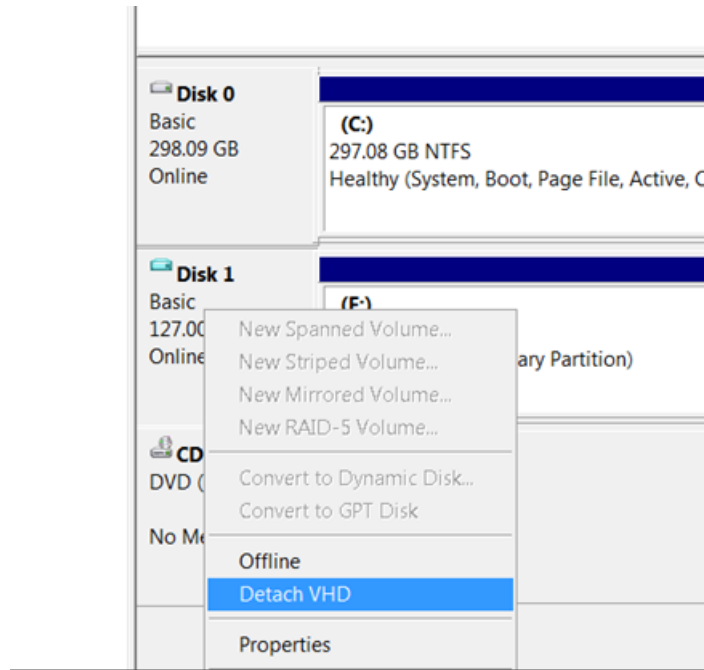


Fig. 3: Detach/Unmount Option in Server Manager

Cluster Shared Volumes

Starting with Bacula Enterprise 12.6, Cluster Shared Volumes File System (CSVFS) backup is supported, so VMs located on CSVFS volumes are backed up transparently. However, mixing CSVFS and NTFS in the same backup is not supported due to a Microsoft limitation. Noticeably, Host Component are located on C:/ which is typically a NTFS volume.

- Make sure to backup this kind of system with 2 different jobs:
- One job backups the default selection without the Host Component:

```
File Set {
  Name = HYPERV-CSVFS-1
  Include {
    Options {
      Signature = SHA1
    }
    # backup all defaults but Host Component
    Plugin = "vss:/@HYPERV/ cexclude=\"Host Component\""
  }
}

Job {
  Name = HYPERV09-CSVFS-NO-HOSTCOMPONENT
  Accurate = Yes
  File Set = HYPERV-CSVFS-1
  Client = wsb-hyp09-fd
  Job Defs = DefaultJob
  Level = Full
}
```

- Another job backups specifically the Host Component:

```
File Set {
  Name = HYPERV-CSVFS-2
  Include {
    Options {
      Signature = SHA1
    }
    # backup only Host Component
    Plugin = "vss:/@HYPERV/ cinclude=\"Host Component\" cexclude=*"
  }
}

Job {
  Name = HYPERV09-CSVFS-HOSTCOMPONENT
  Accurate = Yes
  File Set = HYPERV-CSVFS-2
  Client = wsb-hyp09-fd
  Job Defs = DefaultJob
  Level = Full
}
```

Single Item Restore

(see separate article about Hyper-V Single Item Restore that you can download on the top of the page of that article)

Plugin Notes

Windows VSS Plugin Items to Note

- One file from each drive needed by the plugins must be explicitly listed in File Set used. This is to ensure that the main **Bacula** code does a snapshot of all the required drives. At a later time, we will find a way to accomplish this automatically.
- When doing a backup that is to be used for Bare Metal Recovery, do **not** use the VSS plugin.

General Plugin Items to Note

- The `estimate` command does not handle plugins. When estimating a job that uses plugins, an error message regarding the plugin will be displayed. However, backup jobs will use the plugin.
- The File Set Include Option `CheckFileChanges = Yes` does not work with plugin-generated data. Thus, you must not use that Option in the Include section of the FileSet where you specify using the Hyper-V plugin.
- When an Offline virtual machine is currently backed up, it is not possible to start it (Hyper-V limitation).
- The Bacula `replace` flag is not respected by the Hyper-V plugin. Virtual machines will be always overwritten during restore.
- Microsoft Hyper-V vss interface doesn't support Differential and Incremental backup. Use of the Global Endpoint Deduplication plugin and the `bothsides` FileSet option permits to minimize the data transfer and the storage.

- When trying to restore Incremental or Differential jobs, Hyper-V VSS writer will print errors on PostRestore and PreRestore events. The virtual disk image (VHDX) should be restored despite these errors.

Best Practices

While it is technically possible to backup multiple VMs in one Bacula hypervisor plugin backup job (VMware, Hyper-V, RHV, Proxmox, etc), this is not necessarily the best way to perform VM backups. It is strongly recommended that one backup Job is created for each VM being backed up for the following reasons:

- By default, if one of your VMs fails to backup in a “multi-VM” backup job, the main Bacula job will terminate “Backup OK – with warnings.” The JobStatus for jobs that terminate “Backup OK” and “Backup OK – with warnings” are not differentiated in the catalog. They are both ‘T’, so this means that you will have to carefully monitor your backup job logs in case some VM backups fail and pay attention to the JobErrors field in the job summaries.
- To address this issue, there is a plugin option called “abort_on_error” in each of the Bacula hypervisor plugins, which causes Bacula to immediately fail the job as soon as an error is detected while backing up a VM. However, if you use this option, and the backup of VM number 11 in a list of 50 VMs fails, then the whole job will be failed, and VMs 12-50 will not be backed up during that job’s run.
- A 1:1 configuration (one VM backed up per job) means that the “abort_on_error” option will make more sense to enable in each job so you will immediately know when a VM fails to backup since the Bacula job will terminate with a “Backup failed” message and ‘f’ in the catalog for the job.
- With a 1:1 VM/Job configuration, re-running a specific VM backup job is simple to do after the cause of the failure is investigated and fixed.
- In the example about the 50 VMs, without a 1:1 configuration, there is no way to re-run a backup of just the one VM that failed to backup.
- Additionally, with a 1:1 VM/Job configuration, job metrics will have more meaning because each VM will be one job, and you will know to expect a specific number of jobs each night with each job representing one VM.
- With a multi-VM per job configuration, each VM will be backed up “serially”, one at a time, disk by disk, VM by VM. A 1:1 configuration will allow several VM backups to be run concurrently which will reduce the overall time to perform the VM backups. Of course, you will need to pay close attention to SD and ESXi storage and networking resources, and adjust the number of concurrent jobs accordingly.
- For some hypervisors (VMware, Proxmox, etc) Bacula provides automation scripts (eg: scan_datacenter.pl for VMware). These scripts are designed so that they will create 1:1 VM/Job configurations. If you plan to make use of these automation scripts, it is a good idea to already be thinking this way, and having your hypervisor plugin backup configurations in a 1:1 configuration from the beginning.

VSS services enable **Bacula Enterprise** to do snapshot backup of your Hyper-V virtual machines.

Single Item Restore is supported, but differential and incremental level backup cannot be done using this method.

2 Hyper-V WMI plugin

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

- *Hyper-V Plugins to Cover All Backup Needs*

2.1 Hyper-V Plugins to Cover All Backup Needs

Microsoft has created various technologies for backing up Hyper-V virtual machines. Thus, multiple Bacula Plugins have been designed to maximize the benefits of each solution.

Hyper-V is equipped with a VSS writer on all compatible versions of Windows Server. This VSS writer enables developers to utilize the existing VSS infrastructure for backing up virtual machines to Bacula using the Bacula Enterprise VSS Plugins. This technology is supported by the original **Bacula Hyper-V plugin**. Although it doesn't allow incremental and differential backups and other more granular VM-based options, it can still cover any Hyper-V version. Therefore, it is highly recommended for small standalone Hyper-V servers (single node, no Failover Cluster) and older Hyper-V versions that do not offer Virtual Disk Service or Hyper-V WMI API.

The Virtual Disk Service (VDS) is a service provided by Microsoft Windows that handles query and configuration tasks upon request from end users, scripts, and applications. This service is compatible with Windows Server 2003, Windows Vista, and newer versions. The **Hyper-V Winapi Plugin** utilizes this technology to backup and restore virtual machines. It supports incremental and differential backups, making it the recommended solution for more intricate Hyper-V servers, such as Failover Clusters with multiple nodes, where local disk space is a critical resource. Depending on the relocation of virtual machines within the Cluster, it may be necessary to migrate the backed-up VMs across specific nodes.

Starting in Windows 8 and Windows Server 2012, Hyper-V supports backup via the Hyper-V WMI API. This feature enables individual Guest VMs to be backed up separately and incrementally, offering a more scalable solution compared to using VSS in the host. The Bacula Enterprise **Hyper-V WMI Plugin** uses this technology for backup and restore to/from Bacula. It backups/restores VM in the recommended Microsoft format. By utilizing the Microsoft snapshot format from/to disk, it is essential to have sufficient disk space available for the process to proceed smoothly. In scenarios where disk space may be limited due to busy configurations, the Hyper-V Winapi Plugin can be utilized as an alternative solution.

Backups created using the **Hyper-V WMI Plugin**, **Hyper-V Winapi Plugin** and **Hyper-V Plugin** are not compatible with each other.

Features summary

- Quiescing VSS-based applications can be achieved through VSS-based guest snapshots.
- Microsoft's RCT technology enables Full, Differential, and Incremental image-level backups for virtual machines.
- Complete virtual machine images can be restored effortlessly.

Important notes

- Backups made with the Hyper-V WMI plugin cannot be used with Virtual Full jobs. It is not recommended to mix these backup methods as it may result in difficulties when restoring jobs from Virtual Full backups.
- Single Item Restore is not supported.
- Linux virtual machines cannot be backed up live at Application Consistency level.
- The **Hyper-V WMI Plugin** requires Hyper-V Virtual Machines version 6.2 or above to manage Differential and Incremental backups.

Supported platforms

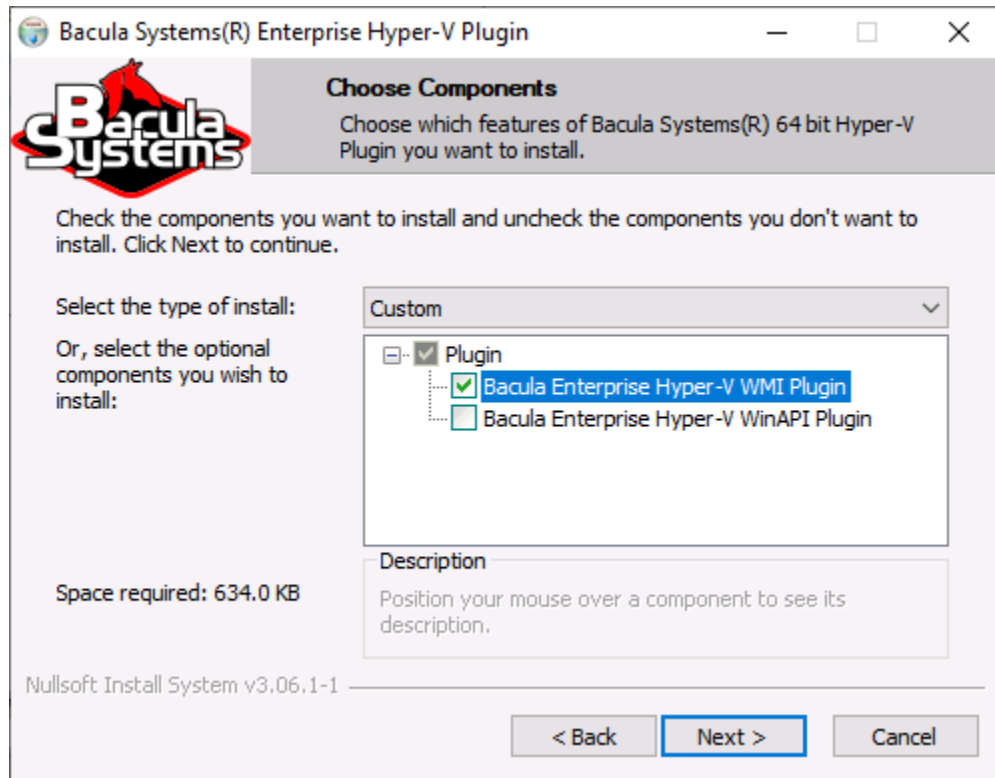
This documentation presents solutions for **Bacula Enterprise** 16.0.0 and higher, and is not applicable to prior versions of Bacula.

This plugin supports Windows 8, Windows Server 2012 and later.

Installation

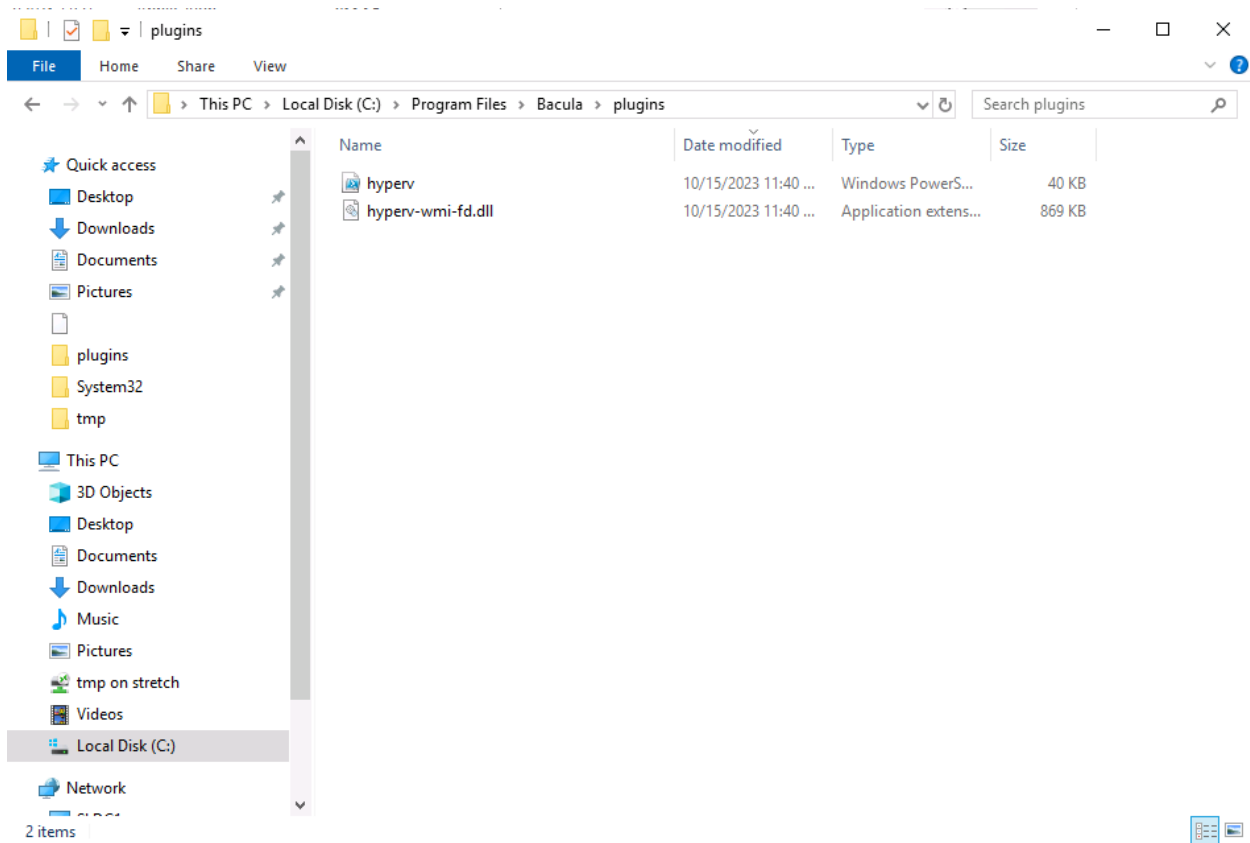
The Bacula File Daemon and the **Hyper-V WMI Plugin** need to be installed on the Hyper-V host server. The **Hyper-V WMI Plugin** Windows installer is the same as the **Hyper-V Winapi Plugin** installer.

You can choose to install one or the other from the Plugin tree.



It will deploy required components within the Bacula File Daemon plugins directory.

To configure the Bacula File Daemon, refer to the general Bacula installation documentation.

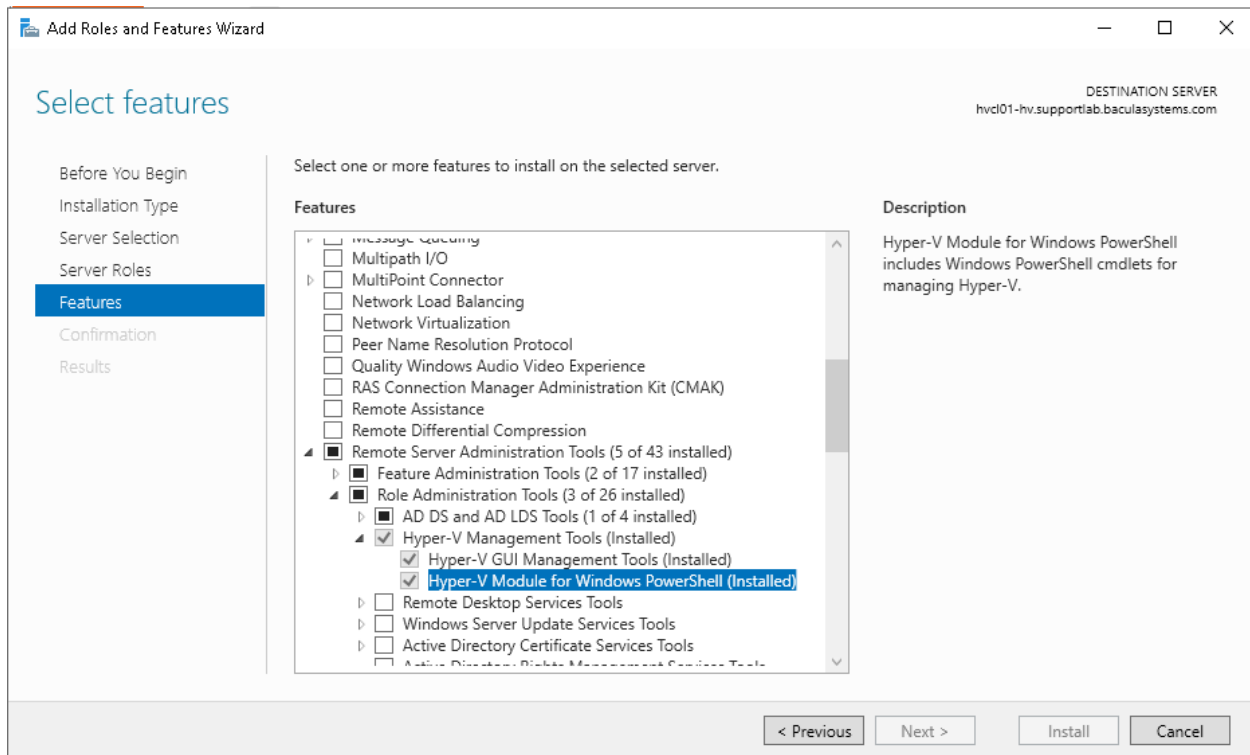


On the server side, the *Hyper-V PowerShell Module* needs to be enabled. On Windows Server or Hyper-V server 2012, 2016 and 2019, use Server Manager to install it. It should be located under Remote Server Administration Tools -> Role Administration Tools -> Hyper-V Management Tools and check Hyper-V Module for Windows PowerShell.

Verify the correct installation of the FD and the **Hyper-V WMI Plugin** by running status client from bconsole or from BWeb.

```
*status client=w2019-hv01-fd
Connecting to Client w2019-hv01-fd at 172.22.22.50:9102
w2019-hv01-fd Version: 12.8.0 (06 April 2021) VSS Linux Cross-compile Win64
Daemon started 19-Jun-21 16:31. Jobs: run=23 running=2.
Microsoft Windows 2012 Standard Edition (build 9200), 64-bit
Priv 0x73f
Memory: WorkingSetSize: 34,168,832 QuotaPagedPoolUsage: 183,768 QuotaNonPagedPoolUsage:
↪17,368 PagefileUsage:
43,687,936
APIs=OPT,ATP,LPV,CFA,CFW,
WUL,WMKD,GFAA,GFAW,GFAEA,GFAEW,SFAA,SFAW,BR,BW,SPSP,
WC2MB,MB2WC,FFFA,FFFW,FNFA,FNFW,SCDA,SCDW,
GCDA,GCDW,GVPNW,GVNFVMPW,LZO,EFS
Heap: heap=34,168,832 smbytes=39,489,074 max_bytes=69,259,353 bufs=395 max_bufs=396
Sizes: boffset_t=8 size_t=8 debug=10 trace=1 mode=0,2010 bwlimit=0kB/s
Crypto: fips=no crypto=OpenSSL
APIs: !GPFS
Plugin: alldrives-fd.dll(1.2) hyperv-wmi-fd.dll(0.1) winbmr-fd.dll(3.1.0)
```

Verify that **hyperv-wmi-fd.dll** is in the “Plugin” line (last line in the above example output).



In the case of a Failover Cluster configuration, the Bacula file daemon and the **Hyper-V WMI plugin** need to be installed on only one node.

Important considerations regarding credentials settings

Important: In order to access and backup the Hyper-V server, the delegation of the User credentials must be enabled and the Bacula file daemon must be logged as an authorized user within the Hyper-V server.

Enable delegation of user credentials on the Hyper-V server

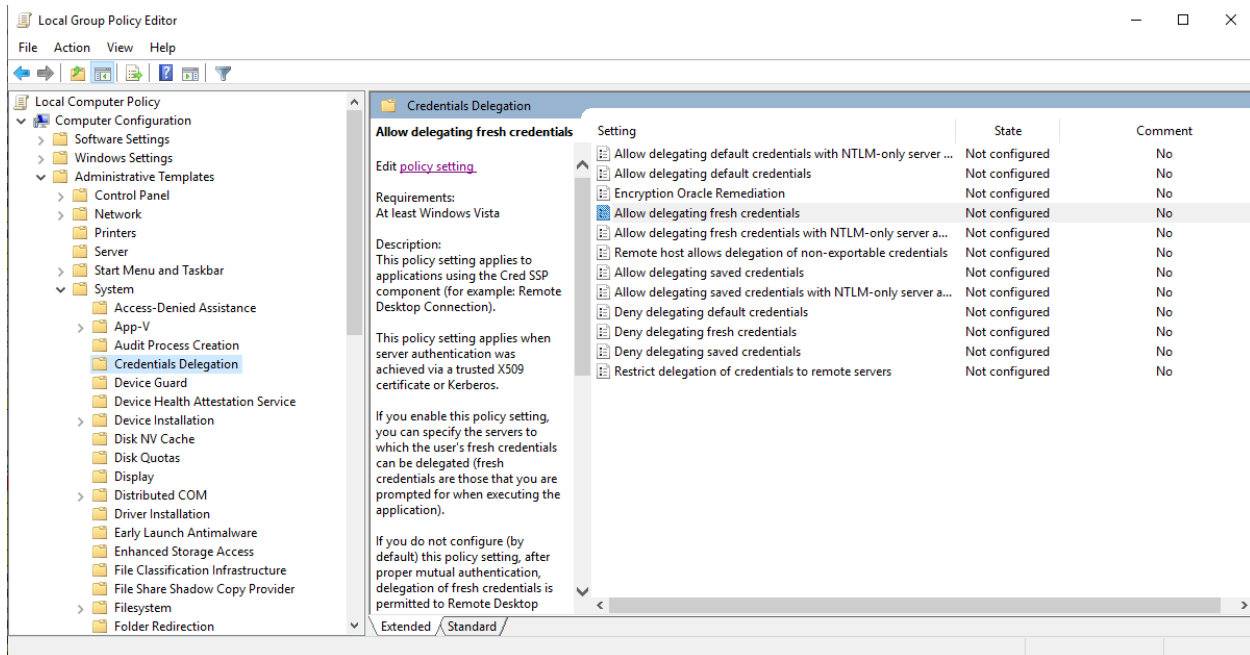
- Run `gpedit.msc` (normally in `C:\Windows\System32`) on the Hyper-V server and look at the following policy: Computer Configuration -> Administrative Templates -> System -> Credentials Delegation -> Allow Delegating Fresh Credentials.

- Verify that it is enabled and configured with the WSMAN SPN appropriate for the target computer.

For example, for a target computer name “myserver.domain.com”, the SPN can be one of the following: `WSMAN/myserver.domain.com` or `WSMAN/*:domain.com`. Introduce it in the “Add servers to the list” “Show” dialog box.

- Alternatively run a powershell console on the Hyper-V server (normally in `C:\Windows\System32\WindowsPowerShell\v1.0powershell.exe`) and enter the following commands:

```
Enable-WSManCredSSP -Role Server -Force
Enable-WSManCredSSP -Role "Client" -DelegateComputer myserver.domain.com -Force
```



Impersonation of Hyper-V WMI Plugin

The impersonation of the **Hyper-V WMI Plugin** can be achieved in different ways.

- Specify the user name and password locally on the hyper-v node. This is the **recommended method**. In a `bacula-hyperv.pwd` file, located by the `bacula-fd.conf` config file (typically `C:\Program Files\Bacula`), `bacula-hyperv.pwd` contains the user name followed by the user password, separated by a colon.

```
name@domain.com:mypassword
```

or

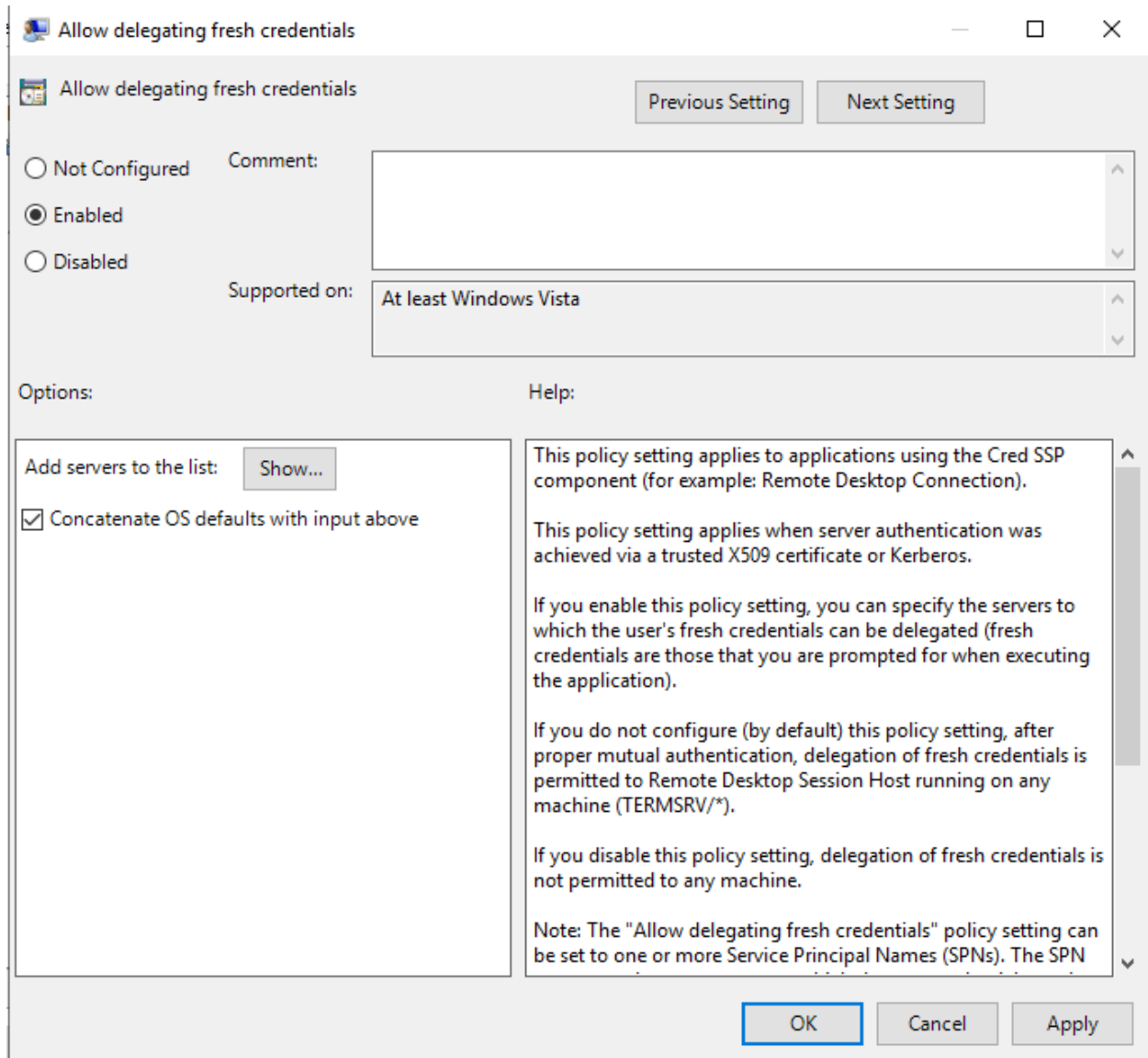
```
DOMAIN\name:mypassword
```

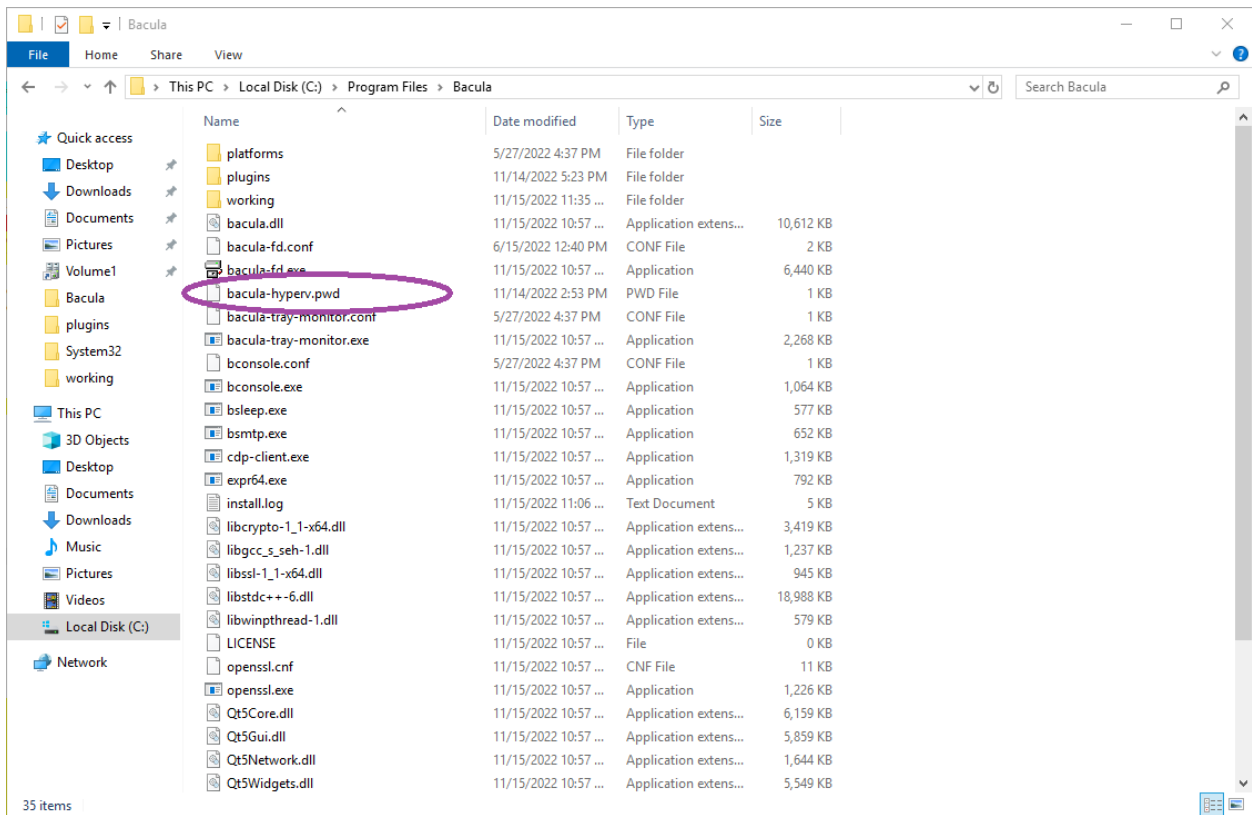
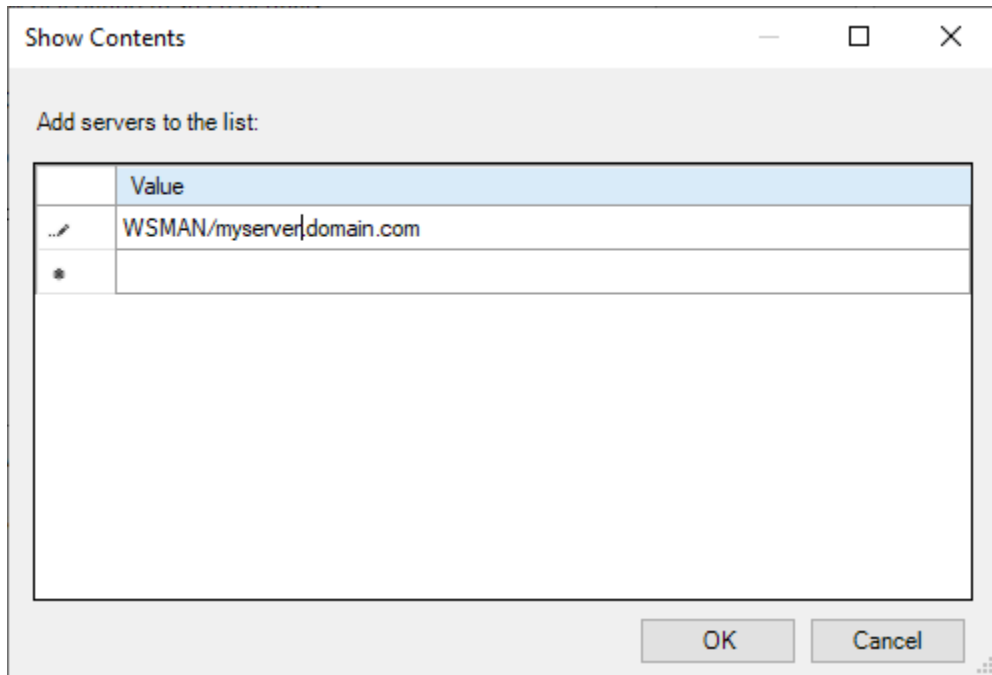
- Impersonate the **Hyper-V WMI Plugin** by passing user and password, as plugin options See Job configuration `user_name` and `user_password` options.
- Manually change the Bacula file daemon default login account:

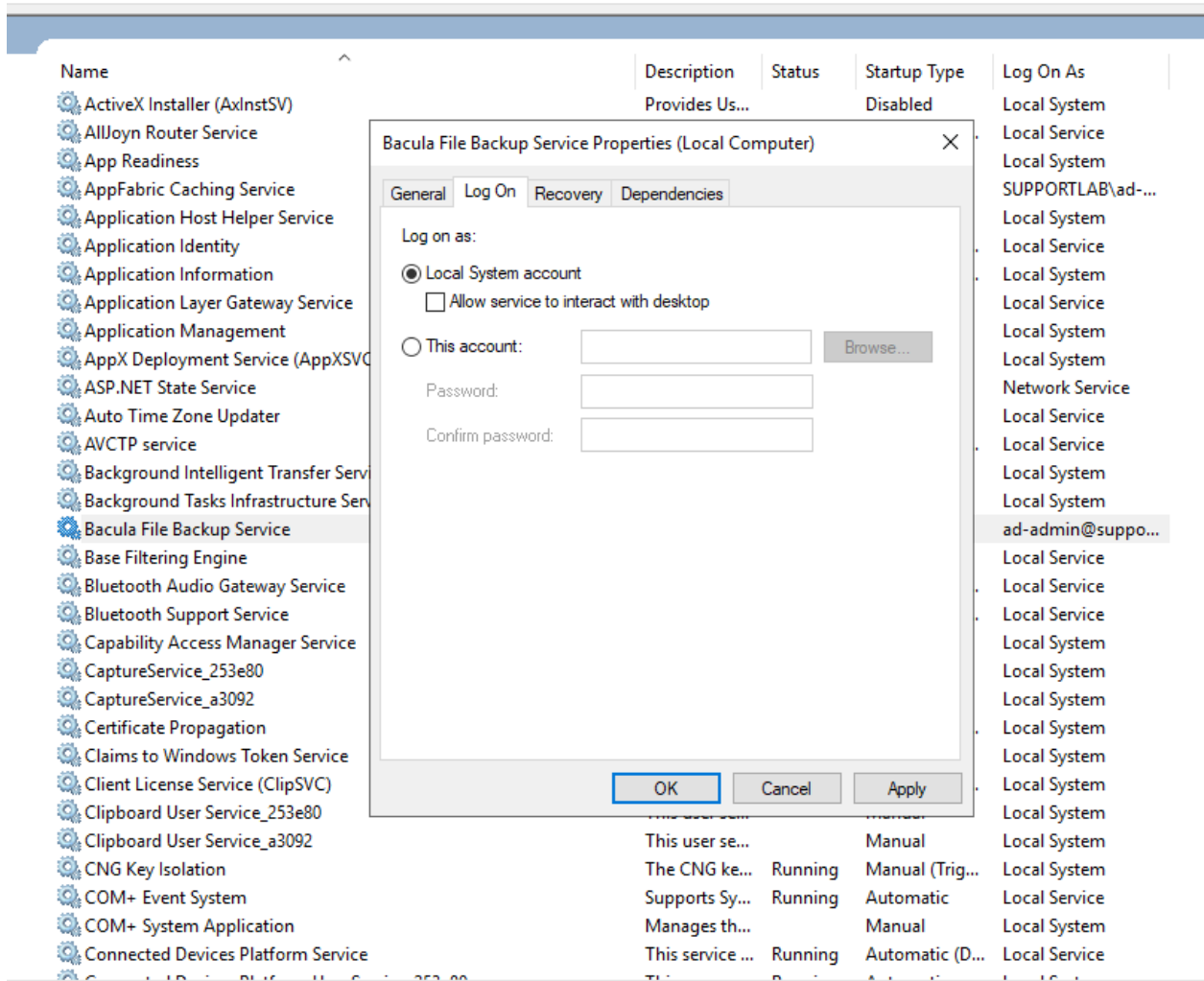
Access the Hyper-V server using administrative credentials. Go to the Windows Start menu, enter “Services”, and press Enter to display a list of all installed services. Locate the Bacula File Backup Service, right-click on it, and then select Properties. Navigate to the Log On tab, where the settings should appear as follows:

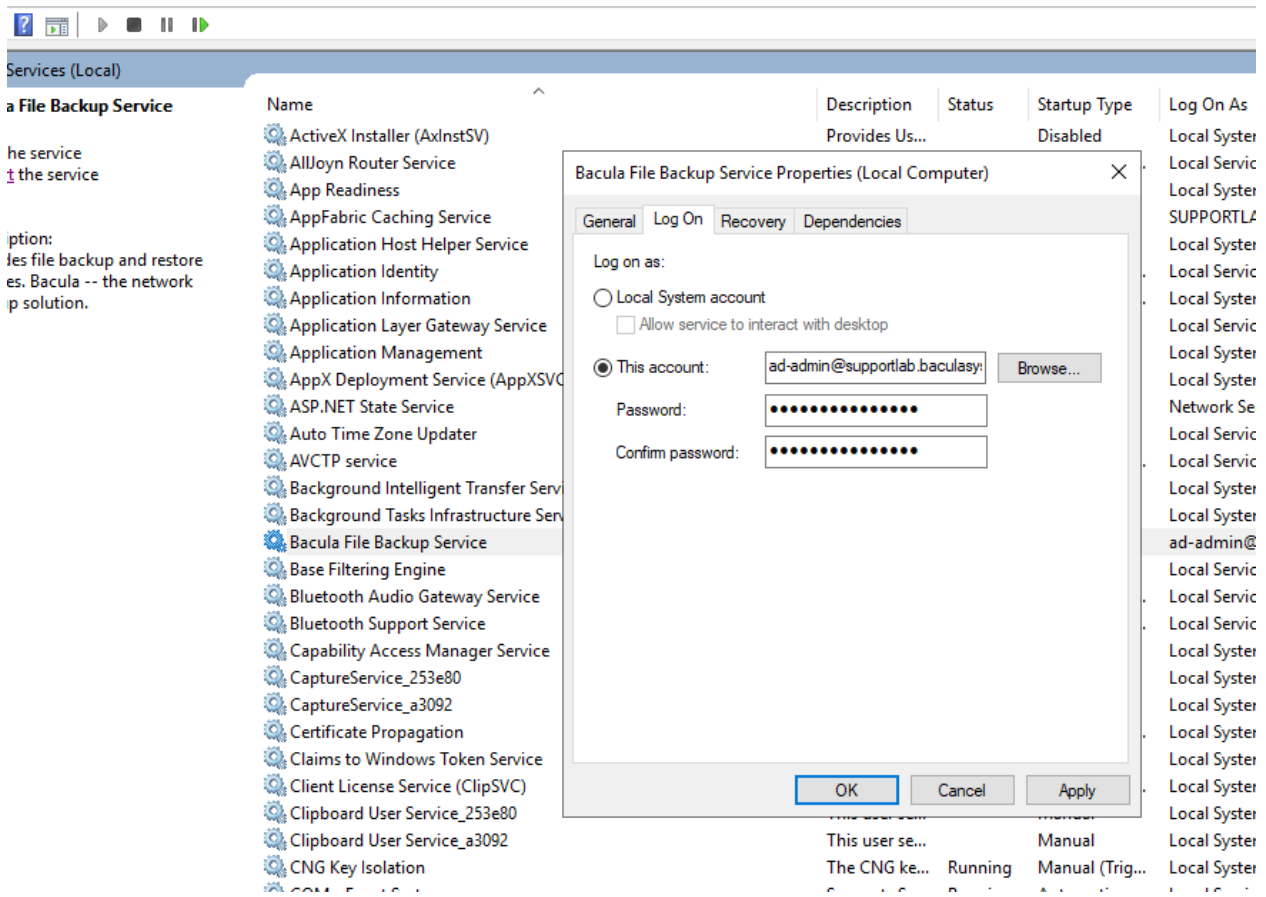
Toggle the selection from “Local System account” to “This account”. Enter the credentials of a Hyper-V administrator (either read only or read-write). Click OK.

Click on the Bacula File Backup Service entry once more with the right mouse button, then select “Restart” to ensure that the changes take effect.









Job Configuration

Once the Bacula File Daemon and the **Hyper-V WMI plugin** are correctly installed and configured, setting a backup job up is as simple as adding the job and the fileset within the Bacula Director configuration file.

Important: The `Enable VSS` parameter must be set to `no` in the FileSet (see examples below).

The following plugin options are supported:

Name	Status	Default	Description
<code>include</code>	Optional	Include all (*)	a Unix shell-style wildcards pattern for including VMs by name
<code>exclude</code>	Optional	Exclude none	a Unix shell-style wildcards pattern for excluding VMs by name
<code>tmp_dir</code>	Optional	<code>Bacula-repo</code>	specifies the Bacula working repository folder. Make sure there's enough space on this location to create VM's snapshots and exports. Default is a <code>Bacula-repo</code> folder in the VHD location.
<code>pre_backup_action</code>	Optional	None	action on the VMs before backup takes place. Can be None, Stop, Save. - None is noop. - Stop stops the VM before backup (useful when VM doesn't support VSS or kernel freeze to maintain consistent backups). - Save saves the VM before stopping it.
<code>post_backup_action</code>	Optional	None	action on the VMs after backup is completed. Can be None, Restart, ForceRestart. - None is noop. - Restart restarts the VM if it was stopped or saved pre-backup. - ForceRestart restarts the VM unconditionally.
<code>consistency_level</code>	Optional	Application	overwrites the consistency level. Can be Application Consistent or Crash Consistent. Application is the recommended value but some VMs might not support it.
<code>allow_pre_save</code>	Optional	Disabled	when enabled, allows retry with <code>pre_backup_action</code> set to Save and <code>post_backup_action</code> set to Restart, if crash consistency retry backup has failed.
<code>local-host-only</code>	Optional	Disabled	when enabled, restricts all operations to the local node (for Failover Cluster configuration).
<code>abort_on_error</code>	Optional	Disabled	abort immediately the job if a serious error is found (b.e when no VM matches the <code>include</code> patterns). By default, a Job error is raised, but the job continues.
<code>disable_vm_migration</code>	Optional	Disabled	when enabled, VMs migration is disabled during backup to avoid collision between backup and migration (for Failover Cluster configuration). Doesn't take any value. To disable, remove keyword.
<code>user_name</code>	Optional	None	the user name that will run the backup/restore operation. This is not the recommended method . The user name can be specified locally on the hyper-v node in a <code>bacula-hv.user</code> file located in the <code>fd</code> plugins folder.
<code>user_password</code>	Optional	None	the user password that will run the backup/restore operation. This is not the recommended method . The user password can be specified locally on the hyper-v node in a <code>bacula-hv.pwd</code> file located in the <code>fd</code> plugins folder.

Examples

Example 1: backup all vms using Bacula's default working directory

```
Job {
  Name = "Hyper-V-BackupAll"
  Type = Backup
  Client= w2019-hv01-fd
  FileSet="Simplest-Hyper-V-FileSet"
  Storage = File
  Messages = Standard
  Pool = Default
}

FileSet {
  Name = "Simplest-Hyper-V-FileSet"
  Enable VSS = no
  Include {
    Options {
      signature=MD5
    }
    Plugin = "hyperv-wmi:"
  }
}
```

Example 2: backup only «Linux- » prefixed vms using Bacula's default working directory

```
Job {
  Name = "Hyper-V-BackupOnlyLinux"
  Type = Backup
  Client= w2019-hv01-fd
  FileSet="Linux-Hyper-V-FileSet"
  Storage = File
  Messages = Standard
  Pool = Default
}

FileSet {
  Name = "Linux-Hyper-V-FileSet"
  Enable VSS = no
  Include {
    Options {
      signature=MD5
    }
    Plugin = "hyperv-wmi: include=\"Linux-*\""
  }
}
```

Example 3: backup any VM having «Windows» in its name, using a custom working directory

```
Job {
  Name = "Hyper-V-BackupOnlyWindowsOnF"
  Type = Backup
  Client= w2019-hv01-fd
  FileSet="WindowsOnF-Hyper-V-FileSet"
  Storage = File
  Messages = Standard
  Pool = Default
}
FileSet {
  Name = "WindowsOnF-Hyper-VFileSet"
  Enable VSS = no
  Include {
    Options {
      signature=MD5
    }
    Plugin = "hyperv-wmi: include=\"*Windows*\" tmp_dir=\"F:/backup\""
  }
}
```

Backup

The files backed up by the Hyper-V server will be visible in a bconsole or with the prefix /@HYPERV-WMI/.

Typically, a VM backup data is organized as follows:

```
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b/1ECD8F42-CCCA-462A-AB0F-B7644EA77B9B.
↪ vmcx
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b/1ECD8F42-CCCA-462A-AB0F-B7644EA77B9B.
↪ vmgs
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b/1ECD8F42-CCCA-462A-AB0F-B7644EA77B9B.
↪ VMRS
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b/backup-config.xml
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b/test1_1C6A2D1E-9CEF-4F08-A14E-
↪ E463F909C94D.avhdx
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b/test1.vhdx
/@HYPERV-WMI/1ecd8f42-ccca-462a-ab0f-b7644ea77b9b-test1
```

Where:

- 1ecd8f42-ccca-462a-ab0f-b7644ea77b9b is the VM UID of the “test1”
- the .vmcx file stores the Vm’s machine settings
- the .vmgs file stores the Vm’s guest state
- the .vmrs file stores the Vm’s running state
- the backup-config.xml contains information on the vm at the backup time
- the vhdx file stores the Vm’s Virtual Hard Drive data
- the avhdx file stores the differential data of the Vm’s Virtual Hard Drive

- 1ecd8f42-ccca-462a-ab0f-b7644ea77b9b-test1 is a convenience file reminding us that the VM name is test1 and its ID 1ecd8f42-ccca-462a-ab0f-b7644ea77b9b

Incremental-Differential backups: the **Hyper-V WMI Plugin** will automatically follow the backup level strategy as scheduled in Bacula.


Consistency Level: A backup can fail when the option “Application Consistent” is required for a VM that doesn’t support it.

- If an Application Consistent backup fails, the **Hyper-V WMI Plugin** will change automatically the Consistency Level to “Crash Consistent” and retry.
- If allow_pre_save is enabled and a “Crash Consistent” backup fails, the **Hyper-V WMI Plugin** will change the pre_backup_action to “Save” and post_backup_action to “Restart” and retry.
- If none of the above works, the backup fails with Error.

Restore

It is advisable to choose the entire fileset instead of cherry-picking backed up files, especially for one VM.

Restore parameters:

Restore Options	Advanced Options	Hyperv-wmi
Restore Options		
Restore Client:	hycl01-norbert	
Where:	C:/Volume1/restore/	
Replace:	Never	
Comment:		
Media Needed		
InChanger	Enabled	Volume
Click "Re-compute the Media" button to display the list of media that will be used during the restore.		
 Re-compute media needed to restore (the action can take some time)		

- **Where:** Can specify a path for VM restoration. If the content is not a path (does not contain slashes or backslashes), it's considered to be the new VM restore name.

Restore Options	Advanced Options	Hyperv-wmi
<code>hyperv-wmi: include="ubuntu" tmp_dir="C:/ClusterStorage/Volume1/backup"</code>		
New Virtual Machine name	<input type="text"/>	
Restore Path	<input type="text"/>	
Avoid Identical UUID collision	<input type="checkbox"/>	
Node where the VM is restored	<input type="text"/>	
Name used to process restore	<input type="text"/>	
Password used to process restore	<input type="text"/>	

- **New Virtual Machine Name:** Specify the restored VM new name
- **Restore Path:** Specify the location where Snapshot files are restored
- **Avoid Identical UUID collision:** Over restoration, the VM UID is regenerated. It avoids issue when the original VM is still existing on the Hyper-V host.
- **Node where the VM is restored:** Specify the name of the node where the VM is to be restored (clustered configuration). Note: the Restore Path need to be shared between the local host and the remote node, for this option to work (on a clustered storage b.e.)
- **Name used to process restore:** impersonation user name (see Impersonation of the **Hyper-V WMI plugin**)
- **Password used to process restore:** impersonation user password (see Impersonation of the **Hyper-V WMI plugin**)

VM Renaming:

If the 'New Virtual Machine Name' is specified, the restored VM(s) will be renamed using this value. However, if the 'Where' value is not a path, then the 'Where' value itself will be used for renaming the restored VMs. In case neither of these options are set, the restored VM(s) will retain their original name(s).

Restore Path:

If the 'Restore Path' is provided, it will be used as the restore path for all the drive-related backup files (vhdx, avhdx) and the VM-related backup files (vmcx, vmgs, vmrs). However, if the 'Restore Path' is empty and the 'Where' value is set with a path, then the 'Where' path will be used instead. To facilitate multiple restorations, the files will be restored in a specific folder named after the Bacula job name. If none of the above options are set, the default locations of the Hyper-V host will be used, and no specific folder named after the Bacula job name will be created. It is important to note that the restore files are not moved during the restoration process, so the Restore Path will be the location of the restored VM.

VMs duplication:

If the original VM still exists on the node, attempting to restore the same VM with the same unique identifier will be rejected. In such cases, the “Avoid Identical UUID collision” option can be used to assign a new unique identifier to the restored machine. If a restoration without the “Avoid Identical UUID collision” option fails, it will automatically be retried with this option enabled, and a warning will be issued.

Retries:

In the event of a VM restoration failure, the import process will be retried. If the “Avoid Identical UUID collision” option is turned off, it will be automatically enabled to prevent the most common cause of restoration errors: duplication of unique identifiers. Alternatively, an attempt will be made to rename the VM. By default, the rename will follow the pattern “<JobName>_<originalVMName>”.

Examples:

- Defaults:
 - Where: Empty
 - New Virtual Machine Name: Empty
 - Restore Path: Empty
 - Avoid Identical UUID collision: Off
 - Node where the VM is restored: Empty

The restored VM(s) are (re)created in the local Hyper-V host default VMs and VirtualHardDrives locations with original names are Unique Identifiers.

- Quick Rename:
 - Where: newVMName
 - New Virtual Machine Name: Empty
 - Restore Path: Empty
 - Avoid Identical UUID collision: Off
 - Node where the VM is restored: Empty

The restored VM(s) are (re)created in the local Hyper-V host default VMs and VirtualHardDrives locations and renamed newVMName.

- Large Restore:
 - Where: Empty
 - New Virtual Machine Name: NewVMName
 - Restore Path: C:\LargeStorage\restore
 - Avoid Identical UUID collision: Off
 - Node where the VM is restored: Empty

The restored VM(s) are (re)created and renamed NewVMName. Virtual drive(s) and VM files are located into a folder named after the JovName in C:\LargeStorage\restore. Something like : C:\LargeStorage\restore\RestoreFiles.<date>_<time>.

Best Practices

While it is technically possible to backup multiple VMs in one Bacula hypervisor plugin backup job (VMware, Hyper-V, RHV, Proxmox, etc), this is not necessarily the best way to perform VM backups. It is strongly recommended that one backup Job is created for each VM being backed up for the following reasons:

- By default, if one of your VMs fails to backup in a “multi-VM” backup job, the main Bacula job will terminate “Backup OK – with warnings.” The JobStatus for jobs that terminate “Backup OK” and “Backup OK – with warnings” are not differentiated in the catalog. They are both ‘T’, so this means that you will have to carefully monitor your backup job logs in case some VM backups fail and pay attention to the JobErrors field in the job summaries.
- To address this issue, there is a plugin option called “abort_on_error” in each of the Bacula hypervisor plugins, which causes Bacula to immediately fail the job as soon as an error is detected while backing up a VM. However, if you use this option, and the backup of VM number 11 in a list of 50 VMs fails, then the whole job will be failed, and VMs 12-50 will not be backed up during that job’s run.
- A 1:1 configuration (one VM backed up per job) means that the “abort_on_error” option will make more sense to enable in each job so you will immediately know when a VM fails to backup since the Bacula job will terminate with a “Backup failed” message and ‘f’ in the catalog for the job.
- With a 1:1 VM/Job configuration, re-running a specific VM backup job is simple to do after the cause of the failure is investigated and fixed.
- In the example about the 50 VMs, without a 1:1 configuration, there is no way to re-run a backup of just the one VM that failed to backup.
- Additionally, with a 1:1 VM/Job configuration, job metrics will have more meaning because each VM will be one job, and you will know to expect a specific number of jobs each night with each job representing one VM.
- With a multi-VM per job configuration, each VM will be backed up “serially”, one at a time, disk by disk, VM by VM. A 1:1 configuration will allow several VM backups to be run concurrently which will reduce the overall time to perform the VM backups. Of course, you will need to pay close attention to SD and ESXi storage and networking resources, and adjust the number of concurrent jobs accordingly.
- For some hypervisors (VMware, Proxmox, etc) Bacula provides automation scripts (eg: scan_datacenter.pl for VMware). These scripts are designed so that they will create 1:1 VM/Job configurations. If you plan to make use of these automation scripts, it is a good idea to already be thinking this way, and having your hypervisor plugin backup configurations in a 1:1 configuration from the beginning.

Failover Cluster

Backup operations are seamlessly executed in a Failover Cluster setup, regardless of the node responsible for hosting the VM(s) at the time of backup. As long as the user possesses the correct credentials on all nodes, the VMs within the cluster will undergo filtering via include/exclude criteria. The tmp_dir directory must be situated on the Cluster Shared Volume and be accessible to the user through an identical path on each node. By default, the tmp_dir is designated as a “Bacula-repo” folder within the VHD default directory, ensuring optimal performance as long as sufficient disk space is allocated for snapshots on the Shared Volume. It is important to note that transferring a VM from one node to another during a backup process is prohibited by Hyper-V.

The WMI provider allows you to perform differential and incremental backup for Microsoft Hyper-V. Single Item Restore is not possible using this method.