HPE StoreOnce Catalyst Plugin

Bacula Systems Documentation

Contents

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.

The HPE StoreOnce Catalyst Plugin, also known as Bacula Catalyst File System (BCFS), is a filesystem client for HPE StoreOnce Catalyst Stores.

The HPE StoreOnce Catalyst Plugin enables the mounting of a StoreOnce Catalyst Store as a filesystem specifically for the storage of **Bacula's** backups. It offers the advantage of supporting Immutability, which is not available with StoreOnce NAS Share. It is important to note that the HPE StoreOnce Catalyst

Plugin does not fully comply with POSIX standards; further details can be found in the section entitled *Limitations* below.

By default, access to the filesystem is restricted to the user who mounted it, and it is advised against using the `chown` command to alter this access.

To optimize functionality, it is advisable to utilize the `setuid=bacula` option, allowing BCFS to operate under the *bacula* user (not as root), while also ensuring that the *mountpoint* is owned by the *bacula* user.

1 Plugin Usage

The HPE StoreOnce Catalyst Store can be mounted as a Bacula Catalyst File System (BCFS) filesystem by using the following command:

```
bcfs <CONFIG-FILE> <mountpoint> [options]
```

The <CONFIG-FILE> file holds the *Catalyst's* configuration parameters, see below for the format and the parameters.

To unmount it, the following commands can be used depending on the operating system:

```
fusermount3 -u <mountpoint>   # Linux
umount <mountpoint>           # OS X, FreeBSD or recent Linux
```

1.1 bcfs Options

-o opt,[opt...]
: mount options, see below for details. A variety of FUSE options can be given here as well. The `allow_root` is forced by the program. The use of the `direct_io` option is **discouraged** as Bacula does not meet any criteria to leverage this option. See *Mounting from fstab* for more options.

-h, --help	print help and exit.
-V, --version	print version information and exit.
-d, --debug	print debugging information.
-v, --verbose	print debugging information.
-f	do not daemonize, stay in foreground.
-s	Single threaded operation.

-o logfile=<PATH>
: to specify the location of the BCFS log file. (default is *stderr*)

-o loglevel=<LEVEL>
: to specify the level of BCFS log file. Valid values are: `debug`, `info`, `warning` and `error` (default is `error`)

2 BCFS Plugin Configuration

2.1 Configuration File

The configuration file, typically referred to as `bcfs.conf`, is structured as follows:


```
#
# Connection section
#
address = 10.0.99.242
store_name = test2
cmd_port = 9387
data_port = 9388
user = Admin
password = MySecretPassword
#
# Logging section
#
catalyst_log_file = /tmp/bcatalystfs.log
catalyst_log_size_mb = 50
# catalyst_log_level = debug|trace|info|quiet|error
catalyst_log_level = error
#
# Rule Section
#
rule_20 = fixed=4096:*.add
rule_50 = variable:*
```

Exercise caution, as this file contains a **password** and must be accessible by the BCFS process. If you use the `setuid=UID` option when initializing the filesystem, ensure that the user with the specified UID has the necessary permissions to access this file.

Note

This file is formatted as a `.ini` file. You may initiate a comment by placing a # (number sign) at the start of a line. *Quote* characters are treated like any other character; therefore, **do not enclose your strings in quotes**.

The = (Equal) sign serves as a delimiter between the *key* and the *value*. Any *spaces* at the beginning or end of the *key* or *value* will be disregarded, while spaces within the values are considered significant.

The configuration file options are:

address
: This is the *ip* address or FQDN of your *Catalyst Server* (*MANDATORY*).

store_name
: This is the name of your *Catalyst Store* (*MANDATORY*).

cmd_port
: This is the command session port number, default is 9387.

data_port
: This is the data session port number, default is 9388.

user
: This is the user name used to connect to the Store of the *Catalyst Server* (*MANDATORY*).

password
: This is the password of the user above (*MANDATORY*).

catalyst_log_file
: This is the name of your local *Catalyst's* log file. This log file is handled by the *Catalyst's* API (*MANDATORY*). This is different of the *fuse* log file, see *Logging* below for more.

catalyst_log_size_mb
: This is the maximum size of the *Catalyst's* log file in MB. Do not specify the *MB* unit! The beginning of the file is overwritten when the file reaches the limit, this is how the Catalyst log file works. Default is 50MB.

catalyst_log_level
: This is the log level of the *Catalyst's* log file. From the most to the less verbose you can use: *debug*, *trace*, *info*, *quiet*, *error*. Default is *error*.

rule_XXX
: Keys starting with `*rule_\*` are for rules that define the deduplication mode regarding the pattern of the file, see below.

> **Note**
>
> The BCFS daemon has its own log file that is controlled by the command line parameters.

Deduplication Rules

The *XXX* part in the *rule_XXX* has no other purpose than uniquely identifying the rule. It is not used to define any priority order. Each rule consists of two components, which are divided by a `:` (colon). The first part is the deduplication mode, and the second part is a well known shell *globbing* file name. See the unix glob(7) manpage for more.

The first part can be of the 2 forms:

variable
: meaning that the *Catalyst* will use variable block deduplication for the files matching the pattern.

fixed=SIZE
: meaning that the *Catalyst* will use fixed block deduplication to the files matching the pattern. `SIZE` is the size in bytes of the blocks and must be between `OSCMN_MINIMUM_FIXED_CHUNK_SIZE` and `MaximumFixedChunkSize`. These two values are displayed in the log file during the initialization of the BCFS client and can also be found in the `sys/version` special file. The minimum value is hardcoded in the API at 2048 (2KiB), and the maximum value is set at 8192 (8KiB) on our *Catalyst* test server. If the size exceeds these limits, the error message displays them. The recommended value is **4096**. **Use the fixed mode only for files that will be accessed on an aligned block boundary, otherwise file input/output errors will occur.**

Filenames corresponding to the second part will use the deduplication mode defined in the first part. Rules are applied in the order they are defined- - the first one that matches, applies. If no rules apply, the default is *variable*.

Deduplication Rules Examples

It is possible to use the Bacula Aligned Plugin, and store the Aligned volumes on the BCFS file system.

When using the Aligned Plugin and storing your Data and Metadata volumes on the BCFS mount point, it is advisable to deduplicate the Data part of the volume that has the *.add* extension and not deduplicate the metadata volume using these rules:

```
rule_20 = fixed=4096:*.add
rule_50 = variable:*
```


4096 is the value recommended by the HPE support.

When using a Bacula Storage without the Aligned Plugin, it is advisable to use the *Variable Block Deduplication* for all the volumes that need deduplication. By prefixing these volumes with the identifier "DedupVolume", the following rules can be applied:

```
rule_20 = variable:DedupVolume*
rule_50 = variable:*
```

Important

Remember that this is *globbing*, not *regex* patterns.

Logging

BCFS generates two distinct log files:

- The first log file is managed by the *Catalyst* library and is configured through parameters such as `catalyst_log_file`, `catalyst_log_level`, and `catalyst_log_size_mb` within the configuration file. This log operates as a circular buffer, resembling a ring structure, which renders the use of the `tail` command ineffective. Additionally, it contains a binary header of a few bytes.
- The second log file is of greater significance as it captures the BCFS logs. It is configured using the `-o logfile=PATH` and should be maintained at the `error` level with the option to increase the level for more information. The level can be set using `-o loglevel=<LEVEL>`, where the permissible levels include: `debug`, `info`, `warning`, and `error`

It is crucial to ensure that the BCFS process is able to write to these log files, particularly when using the `setuid=UID` option.

2.2 Mounting from fstab

The BCFS can be mounted automatically from `/etc/fstab`, using the following line:

```
/etc/bcfs.conf /backup fuse.bcfs setuid=bacula,nodev,noexec,noatime,
↪loglevel=error,logfile=/tmp/bcfs.log 0 0
```

The first parameter is the configuration file. Notice that the file system type is `bcfs` (for backwards compatibility, you may also use `fuse.bcfs`). The `bcfs` binary must be in the *system path*, usually it is installed into `/usr/bin`. The `setuid=bacula` changes the user ID of the process to `bacula`. If this adjustment is made, it is essential to ensure that the *bacula* user possesses the necessary read and write permissions for the mount point. Additionally, it is crucial to confirm that the `/sbin/nologin` shell is not set for user `bacula`:

```
# grep bacula /etc/passwd /etc/shadow
/etc/passwd:bacula:x:990:986:Bacula:/opt/bacula/working:/bin/sh
/etc/shadow:bacula:!!:19550::::::
```

It is possible to use the commands `chsh` to change the *shell* and `passwd` to lock the account:

```
# chsh -s /bin/sh bacula
# passwd -l bacula
```

The `nodev` and `noexec``options are recommended for enhancing security, while the ``noatime` option prevents unnecessary updates. For the `loglevel` and `logfile` options, see *Logging* above.

3 Bacula Storage Configuration

Using the *Aligned* driver with the *Fixed* mode and a 4KiB block size will yield optimal deduplication performance on the *Catalyst* system. If there is a discrepancy in alignment between your data source, such as when dealing with virtual machines that have misaligned partitions, or if your data is subject to changes over time due to the management of numerous uncompressed text documents, the *Variable* mode may deliver superior results. Regardless, the *Aligned* driver consistently outperforms the *File* driver in terms of deduplication efficiency.

Here is a typical device configuration for a Bacula *Aligned* device:

```
Device {
  Name = storeonce1
  DeviceType = Aligned
  MediaType = sobcfs
  ArchiveDevice = /opt/bacula/archive/catalyst001
  AlwaysOpen = no
  LabelMedia = yes
  AutomaticMount = yes
  RemovableMedia = no
  RandomAccess = yes

  MaximumConcurrentJobs = 1
  SetVolumeReadOnly = yes
  MinimumVolumeProtectionTime = 30 days
      #FileAlignment = 64k
}
```

The following directives are noteworthy:

Device Type = Aligned
: Chose the *Aligned* driver for the best deduplication performance.

Media Type = sobcfs
: Do not forget to define a specific *MediaType* for the volumes in the same store.

Set Volume Read Only = yes
: This allows the device to activate the *immutability* on the volumes.

Minimum Volume Protection Time = 30 days
: This value must match the value configured on the *Catalyst* system.

Maximum Concurrent Jobs = 1
: If you are using *Aligned*, it is good to set the *MCJ* to 1, as the *Aligned* driver forces the *MCJ* to 1 anyway.

Do not modify the directives controlling the size of the *chunks* related to the *Aligned* driver; the default settings, particularly the *64K* for *FileAlignment*, are optimal.

Certainly, multiple devices or even an autochanger can use the same *Catalyst Store*.

Note

The `SetVolumeImmutable` and the `SetVolumeAppendOnly` directives are not supported with the Bacula Catalyst File System (BCFS). Instead, `SetVolumeReadOnly` must be used.

Note

It is useless to compress the data in the *Fileset* as the *Catalyst* already compress the chunks.

4 Immutability

The BCFS offers support for immutability, provided that this option is activated on the *Catalyst Store*.

First, verify in the *StoreOnce Management Console* that the option is enabled for your *Catalyst Store*. Check the *details* page of the *Store* to ensure that the value for *Server Controlled Data Immutability Retention* in the *Security* section is not set to *No limits*.

In the Bacula Device configuration, immutability is configured by setting `SetVolumeReadOnly = yes`, and the *Server Controlled Data Immutability Retention* value configured by using the Minimum Volume Protection Time directive.

When you enable the *Server Controlled Data Immutability* feature, you need to specify the duration of the Immutable Period, which defaults to 30 days. The Immutable Period begins once Bacula marks the item as Complete. See *Querying the Bacula Catalyst File System (BCFS) Tags*.

Note

The Immutable Period includes a one-hour grace period during which you can delete or append data to the end of a Catalyst item. After this grace period, the Immutable Period officially starts.

The *volumes* that are designated as read-only on the filesystem will have the *Complete* tag applied and will either be eligible for immutability or already be immutable.

When immutability is configured in the Bacula Device resource using the Bacula Catalyst File System (BCFS) to store the Bacula volumes, the volume will be marked as Read-Only if:

- during a backup job, the volume reaches its full capacity (for example, when `MaximumVolumeBytes` is reached), then it is marked as Read-Only:

```
03-Jun 10:53 bacula-catalyst-sd JobId 7419: Marking Volume "Vol-0001" as read-
↪only. Retention set to 04-Jun-2025 10:53 (1 day).
```

- when the bconsole `update volumeprotect` command is run:

```
*update volumeprotect
Found 1 volumes with status Used/Full that must be protected
Connected to Storage "bacula-catalyst" at bacula-catalyst:9103 with TLS
3000 Marking volume "Vol-0001" as read-only.
```

Note

When Bacula considers the volume as Used (for example, when `MaximumVolumeJobs` is reached), the volume is not automatically set as read-only. It requires the `update volumeprotect` command to be run. See: *Best Practices*.

Note

It is recommended to disable `Autoprune` when using the immutability feature. Instead, use `prune volume expired yes` to prune volumes based on the `VolumeRetention` value.

5 About Deduplication

Two data segments of 4KiB will deduplicate due to their identical nature. Their similarity arises from sharing the same origin. It is highly unlikely for two data segments of a size significant enough to warrant consideration for deduplication to lack a common origin. Achieving a favorable deduplication ratio is not necessarily advantageous. The choice between duplicated and centralized data presents both advantages and disadvantages, and it is essential to determine which option best suits your needs.

The major sources of duplication are:

- Always running *Full* backups instead of *Incremental*.
- Running backups of Virtual Machines that share the same OS.
- Having multiple copies of identical data.

If you are aware of duplicated data and are experiencing a low deduplication ratio, it is advisable to take this matter seriously.

Additionally, the implementation of *compression* or *encryption* may disrupt the deduplication algorithm.

5.1 Querying the Bacula Catalyst File System (BCFS) Tags

The Bacula Catalyst File System (BCFS) adds extra attributes to the Bacula Volumes files.

This extra information can be retrieved from the volume file itself by using the `xattr` linux command.

The Bacula Catalyst File System (BCFS) tags will show a *true* value when present:

```
$ xattr -l /catalyst/Vol-0001
dedup: var
chunksize: 0
Complete: true
BaculaEnterprise: true
```

dedup
: can be *var* or *fixed* depending on the rules you have set up in the *Configuration File*.

chunksize
: when *dedup* is set to *fixed*, this is the size you have set up in the configuration file, else it is *0*.

Complete or Incomplete
: are *tags* handled by the *Catalyst*, *Complete* is synonymous to *read-only* while *Incomplete* is synonymous to *read/write*.

BaculaEnterprise
: is a *tag* that is added to every object that are created by the BCFS. A file without this *tag* has been created by another application.

If `xattr` is not available, you can try `getfattr`:


```
$ getfattr -d -m ".*" /catalyst/Vol-0001
getfattr: Removing leading '/' from absolute path names
# file: catalyst/Vol-0001
BaculaEnterprise="true"
Complete="true"
chunksize="0"
dedup="var"
```

5.2 Querying the sys/ Filesystem

The *sys/* sub-directory comprises a set of virtual files that serve as an interface to the BCFS internal.

command_arguments
holds the arguments on the command line at the time BCFS was started:

```
bcfs -oallow_root -osubtype=bcfs,fsname=/etc/bcfs.conf -o rw,nodev,noexec,
↪noatime,nosuid /catalyst
```

configuration
holds the configuration (the password line has been removed):

```
address = 10.0.99.242
user = Admin
cmd_port = 9387
data_port = 9388
store_name = test2
catalyst_log_file = /tmp/bcatalystfs.log
catalyst_log_size_mb = 50
# catalyst_log_level = debug|trace|info|quiet|error
catalyst_log_level = error
rule_10 = variable:*.add
#rule_20 = fixed=4096:*.quatre
rule_90 = variable:*
```

version
holds the BCFS version and information from the *Catalyst's* library and server:

```
version: 1.0.1
fuse_version: 29
fuse_package_version: NA
read_bandwidth: high
write_bandwidth: low
OSCMN_MINIMUM_FIXED_CHUNK_SIZE: 2048
MaximumFixedChunkSize: 8192
ClientSoftwareVersion: RHEL_x64_2022-04-28T12.28.723_4.3.2-2217.20
ServerSerialNumber: 17TLGM2FH78D9WAM
ServerSoftwareVersion: 4.3.6-2323.20
DiskCapacity: 882079956992
SupportIsvPerObjectImmutability: 1
MaximumDataSessions: 256
FreeDataSessions: 61
```

cmd_connections
holds information about recycled command connections:


```
0x55895bf8afb0 in_use=0 age=0s
0x55895bf8afc8 in_use=0 age=0s
0x55895bf8afe0 in_use=0 age=0s
0x55895bf8aff8 in_use=0 age=0s
0x55895bf8b010 in_use=0 age=14s
```

data_connections
one line per data connection to the *Catalyst*. Any open file has one line:

```
0x7ff0dc0225d0 off=6 age=14s name=foobar
```

store
holds the information about the store and the server that can change over time:

```
DiskCapacity: 882079956992
DiskSpaceFree: 870689128448
DedupeRatio: 3.9
UserDataSize: 25092777408
UserDataStored: 25092777408
DedupedDataSizeOnDisk: 6336564418
DedupeMetaSizeOnDisk: 501318090
DatabaseSizeOnDisk: 13312879
ScratchPadSizeOnDisk: 0
NumObjects: 51
SupportUserDataSizeQuotas: true
UserDataSizeLimit: 0 (unlimited)
SupportDedupedDataSizeOnDiskQuotas: true
DedupedDataSizeOnDiskLimit: 0 (unlimited)
last_update: 2024-11-20T13:36:20Z
last_update_epoch: 1732109780
```

The available `DiskSpaceFree` is shared among all the *Catalyst stores*. The `DedupeRatio` represents the ratio of `UserDataSize` and `DedupedDataSizeOnDisk`. The `UserDataStored` excludes the sparse regions from the total `UserDataSize`. The quota values, namely``UserDataSizeLimit`` and `DedupedDataSizeOnDiskLimit`, relate to the two previous values. The `last_update` indicates the timestamp when these value were retrieved.

6 Best Practices

In some cases, for example when the status of the Volume is changed by the Director via the update volume command, the Storage Daemon will not be able to change the permission on the Volume. Some Volumes may have the Full/Used status without the proper protection.

The command `update volumeprotect` is designed to determine the list of the volumes that are not protected and connect the Storage Daemon to update the permissions.

```
*update
Update choice:
     1: Volume parameters
     2: Pool from resource
     3: Slots from autochanger
     4: Long term statistics
     5: Snapshot parameters
```

(continues on next page)

(continued from previous page)

```
     6: Volume protection attributes on Storage Daemon
Choose catalog item to update (1-6): 6
Found 1 volumes with status Used/Full that must be protected
Connected to Storage "bacula-catalyst" at bacula-catalyst:9103 with TLS
3000 Marking volume "Vol-0001" as read-only.
```

or

```
*update volumeprotect
Found 1 volumes with status Used/Full that must be protected
Connected to Storage "bacula-catalyst" at bacula-catalyst:9103 with TLS
3000 Marking volume "Vol-0002" as read-only.
```

This command can be scheduled in an Admin Job to run every day. For example:

```
Job {
  Name = update-protected-admin-job
  JobDefs = DefaultJob
  Type = Admin
  Runscript {
     Console = "update volumeprotect"
     RunsOnClient = no
     RunsWhen = Before
  }
}
```

7 Limitations

The HPE StoreOnce Catalyst Plugin has been implemented on top of the Catalyst API to address the storage requirements of the Bacula volumes and the Aligned Plugin.

In Bacula, the usage of the *file* based volumes mimics the use of the *tape* volumes, and the access patterns do not require the full range of features typically offered by contemporary filesystems.

Additionally, the Catalyst API has its limitations and may not readily fulfill all the demands of a traditional filesystem.

Nevertheless, it is anticipated that the Catalyst API will meet the storage needs for Bacula's volumes and its Aligned Plugin.

The following outlines the primary distinctions between *the HPE StoreOnce Catalyst Plugin* and a *POSIX* compliant filesystems.

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.

7.1 No Sub-directory Tree

The Catalyst API exclusively supports *Stores* and *Items* and does not accommodate structures resembling a directory tree. However, this does not pose a limitation for Bacula, as each volume is required to have a unique name and is frequently stored within the same directory. Bacula handles all management tasks, eliminating the need for users to directly access the archive directory for their daily operations.


```
$ mkdir ~/mnt/dir
mkdir: cannot create directory ‘/home/bac/mnt/dir’: Function not implemented
```

7.2 File Cannot Be Renamed

The name of a Catalyst item is fixed and cannot be altered, which means the corresponding file is also unchangeable. However, the file can be duplicated under a new name, allowing for the original file to be removed.

```
$ mv ~/mnt/a4 ~/mnt/a99
mv: cannot move '/home/bac/mnt/a4' to '/home/bac/mnt/a99': Function not␣
↪implemented
```

7.3 Deletion of Open Files

When a file in use is deleted, FUSE attempts to rename it to a *hidden* file to delete it later. However, if the file cannot be renamed, as indicated in the section titled *File cannot be renamed* above, the operation will not succeed.

```
$ rm  ~/mnt/file
rm: cannot remove '/home/bac/mnt/file': Function not implemented
```

7.4 Content of File Cannot Be Modified

Modifying data within a file is possible, but **it will truncate the file at the point of modification, resulting in the loss of any subsequent data**. This is *CRUCIAL* to note, as attempting to run applications other than Bacula may lead to unforeseen consequences.

7.5 Characters Allowed in File Names

The name may consist of the following characters: `0-9 a-z A-Z _ . + [ ] ^ | ? * ( ) # : ; = @ { }`. However, spaces are not permitted in the name.

7.6 Inconsistency between atime, mtime and ctime

The *Calatyst* API does not provide support for the `atime` value, `atime` is always a copy of the `mtime` value. When the file is not open (nobody reading or writing to it), these values come from the Catalyst server, using the Catalyst clock. However, when a file is open, these three values are managed at the fuse level in a consistent manner using the time of the FUSE host. They are initialized using the Catalyst API. If there is a discrepancy between the clocks of the client and the server, inconsistencies may arise.

