



Bacula Enterprise Fundamentals

Bacula Systems Documentation

Contents

1	About Bacula Enterprise	2
2	Bacula Enterprise Architecture	8
3	Bacula Enterprise Explained	13
4	Bacula Enterprise Guide to Basic Operations	23

Contents

The following chapter aims at presenting you with all the necessary information to start with Bacula Enterprise solution. To gain full understanding of Bacula, it is recommended to follow the order of browsing through the articles presented below.

1. Go through basic information in the article *About Bacula Enterprise*.
2. Familiarize yourself with Bacula Enterprise architecture in the *Bacula Enterprise Architecture*.
3. Learn about how Bacula Enterprise works in the article *Bacula Enterprise Explained*.
4. Have a look at basic tasks in Bacula in the article *Bacula Enterprise Guide to Basic Operations*.

After you got acquainted with the Fundamentals chapter, you may want to read more detailed documentation.

1 About Bacula Enterprise

The following document aims at introducing the reader to what Bacula Enterprise, or simply: Bacula -is, and what its main features are. It is meant for anybody who wants to have basic understanding of Bacula.

1.1 What is Bacula Enterprise?

Bacula is a highly secure, enterprise-class, feature-rich software that allows system administrators to easily manage backup, recovery, and verification of computer data across a network of computers of different kinds. Bacula can run entirely upon a single computer and can backup to various types of media, including tape and disk. It is independent of both hardware and infrastructure, can run on any brand of computer and fits neatly into on-premise as well as hybrid deployment or full cloud infrastructure. It is especially stable, reliable, secure and massively scalable. The broad support of plugins enormously simplifies the backup of complex sites and technologies. Bacula is used in both small deployments and in extremely large organizations, as well as HPC (High Performance Computing) environments.

1.2 Features

Bacula brings rich new features and enhancements for backup and data recovery to enterprises, data centers and managed service providers. A comprehensive list of features, organized by categories is presented below.

1.3 Bacula Users and Administrators

When considering enterprise backup and recovery environments, it is often useful to distinguish between different types or classes of people interacting with the backup and recovery tool:

- Administrators - can control all aspects of Bacula, and modify its configuration.
- Operators - interact with Bacula, following defined procedures, are responsible for certain operational aspects, but do not touch the configuration.
- Users or end-users - people who have no access to the Bacula configuration or all Bacula's features, but may access certain of its functions. A typical example is that a user can restore their own data, to their own computer, but cannot see other backup data or access computers for which they are not responsible.

Bacula itself does not have the concept of users or administrators, but has Consoles that are designed to allow some users to have limited permissions. However, the BWeb Management Suite - the Bacula Enterprise Web-based tool for management and configuration - includes the concepts of users, groups and permissions.

1.4 Core Features

The following article presents the core features of Bacula.

Bacula is:

- Especially secure – Bacula's modular architecture is designed to offer higher levels of security than other solutions. It also has a broad set of additional and specific security features to enable IT department leaders to implement a high security strategy that meets best practices and zero-trust policies.
- Network-based – all backups and restores are done via the network. This allows Bacula to run on a single server, and backup any computer in your data center.
- Centrally administered – Bacula administration is centralized in the Director. Centralizing the administration is essential when your network grows beyond a few computers.
- Run automatically – once set up, Bacula runs automatically. Normally, a properly configured Bacula installation requires little maintenance or intervention, except when adding new machines, or when hardware errors occur.
- Able to perform bookkeeping – Bacula does hard bookkeeping by maintaining a catalog of what is backed up and where. If you have hundreds or thousands of machines to be backed up, it is essential that the bookkeeping is automatically maintained by a program rather than requiring human intervention.
- Responsive to multiple platforms - Bacula is compatible with a wide range of platforms: BSD, Unix, Linux, MS Windows, Mac OS X, and others as well as a large range of hardware.
- Modularly designed – means it scales well from small shops (one machine) to very large ones (tens of thousands of machines).
- Responsive to multiple backup media – Bacula handles a variety of different media such as local disks, network disks, block storage, tapes, autochangers or cloud devices.
- Reliable – Bacula consists of advance tools for memory and lock management. It is very common to run it without problems for months.
- Characterized by high performance – Bacula's modern multi-threaded design allows running multiple simultaneous backups and can achieve high speeds writing to disk, tape or cloud.

- Customizable – Bacula can be easily customized to almost any backup/restore need.
- Able to restore data rapidly - easy and fast restores using Bacula's database and graphical user interface.
- Equipped with advanced reporting, notification, monitoring systems – Bacula has excellent reporting addons such as Bweb, which allows to get graphical and raw statistics for custom reports, billings, trends, optimizations and capacity planning. Bacula provides very good notifications and monitoring capabilities, and can also be extremely well-integrated into monitoring tools such as Nagios.
- Providing extremely good interoperability through:
 - Full featured REST API
 - Full featured Console User Interface
 - Script-friendly configuration files
 - Ability to expand any job capabilities with additional pre or post script
- Including advanced backup policies:
 - Progressive Virtual Full backup syndication
 - Differential, Full, Incremental backup types
 - Built-in Job scheduler
 - * Job multiplexing (run hundreds of jobs simultaneously)
 - Re-scheduling capabilities
 - Advanced and customizable features to control job concurrency
- Including advanced operation capabilities, such as:
 - Restart incomplete jobs features
 - Bandwidth limitation
 - Job sequencing using priorities
 - Customize restore operations to the original source or to a different target
- Enormously scalable - Bacula is scalable from small single computer systems to systems consisting of hundreds or thousands of computers located over a large network.
- Its highly modular architecture is specifically designed to manage:
 - Billions of file/object records in the catalog
 - Large amounts of data (PB per storage)
 - Thousands of Clients per Director
 - Each Daemon/Service can be separated on the network
 - Ability to extract single files from snapshots such as virtual machine

1.5 Security Features

The following article presents the security features of Bacula.

Bacula is a solution known by its especially strong security standards that allow not only to keep data safe, but also all the infrastructure around it. Because of its modern and advanced security features, Bacula is relied on by many of the largest government and defence organizations. Some more details about security related features are presented below:

- Distributed and isolated *architecture* with limited privileges among each component
- FIPS 140-2 compliance
- Automatic encryption for all network communications (can be turned off or modified with custom certificates)
- Verification of files previously catalogued:
 - File integrity purposes to detect silent data corruption
 - System break-in detection (Tripwire-like capability)
- CRAM-MD5 password authentication between each component (daemon)
- Configurable Data encryption at rest on a Client by Client basis
- Configurable Data encryption at rest globally at Storage Daemon level
- Computation of MD5, SHA1, SHA256 or SHA512 signatures of the file data
- Immutable disk volume feature for Linux based storage destinations
- Immutable NAS Support (Netapp SnapLock, DataDomain RetentionLock or HPE StoreOnce Catalyst among others)
- Immutable tape support (WORM)
- Immutable cloud support (S3 ObjectLock, Azure Blob immutable)
- Connectivity to Active Directory or LDAP services to protect access
- 2-Factor authentication through One-Time Password (OTP), allowing use of smartphones with bio-metric functions to access Bacula's web GUI
- Advanced Ransomware protection tools, such as:
 - Security module to detect vulnerabilities, bad configurations, missing updates and many other threats on Windows and Linux
 - Automatic malware protection by known hash checking for backup, restore and verify processes
 - Antivirus Plugin to detect malware on stored data
 - BGuardian Plugin to automate security analysis for backups, detecting issues and providing detailed reports and alerts
- Advanced File Daemon restriction mechanisms to limit backup, restore or scripts scope by path, user or group id
- Monitoring integrations with SNMP and Security Information and Event Management (SIEM) Systems
- Agnostic N-Tier backup support including offsite and/or cloud copies.
- Backup poisoning detection (described here)
- Automatic security configuration assessment (described here)
- Console Directory authentication (ldap/ad director connector through Bacula Pluggable Authentication Module).
- Auditory logging (through Event Messages)
- User Role Base Access capabilities through resource ACLs

1.6 Storage Destinations

The following article presents information on Bacula storage destinations.

- Global Endpoint Deduplication™ (deduplication from the client to the storage)
 - Bacula Enterprise native deduplication system, integrated into the core of the solution
 - Deduplication across any kind of datasource, globally to a Storage Daemon
 - Deduplication in the source: just send changed data through the network
 - VirtualFull or Synthetic backup integration: just copy references for extremely fast jobs
 - Ready to be used to dedup in the cloud
- Data compression
- Communication line compression, allowing users to divide by 3 the volume of data transmitted across communication lines
- Filesystems deduplication (Aligned Volume Format)
- Integration with practically any kind of tape storage
- Storage Daemon to Storage Daemon (SD2SD) for replication capabilities, using deduplication
- Storage Daemon can be deployed in any Linux systems
- Storage Daemon Reporting capabilities to retrieve a range of information such as available disk space and disk usage information
- SAN Shared Tape Library Module
- Oracle ACSLS support
- Full Hybrid Cloud capability, via S3, Azure, Glacier, Google Cloud and Oracle Cloud interfaces.

1.7 NAS and External Protection

The following article presents information on NAS and external protection.

Bacula can also use NAS and other storage systems as the source of information and protect them using specific techniques. Among them:

- Nutanix Filer
- Netapp Filer
- NDMP support
- NDMP CAB support
- Hadoop HDFS module
- Incremental Accelerator for NetApp
- S3 Objects Backup (from Amazon or any generic solution)
- OpenStack Swift Backup

1.8 Endpoint Features

The following article presents the endpoint features of Bacula.

Some relevant features about backup and restoring endpoint servers:

- Continuous Data Protection
- VSS Support in Windows systems
- Client behind NAT support
- Flexibility to configure who initiates the connection (Storage calls client)
- BareMetal Recovery for Linux hosts
- BareMetal Recovery for Windows hosts

1.9 Cloud and Virtualization

The following article presents information on Bacula cloud and virtualization.

Most of the systems in the IT world can be backed up following an agent based strategy, where the File Daemon is deployed in the target host, independently from the underlying platform.

However, Bacula also offers a very complete solution to protect Virtual Machines or Containers in agent-less mode and in a consistent way through the usage of snapshot techniques. These technologies are covered through specific Plugins. The main Virtualization platforms in the market are covered:

- Hypervisors
 - VMware
 - Hyper-V
 - Nutanix AHV
 - Xen
 - Proxmox
 - KVM
 - Red Hat Virtualization/oVirt
- Cloud Infrastructure as a Service
 - Azure VM
- Container Environments
 - Docker
 - Kubernetes Clusters
 - Red Hat OpenShift

1.10 Databases

The following article presents information on databases.

Databases have specific requirements in order to be backed up or restored. Bacula uses corresponding specific techniques to provide those backup and restore capabilities through specific plugins. The main Databases in the market are covered:

- MSSQL
- PostgreSQL
- MySQL
- Oracle
- SAP HANA
- SAP (Sybase) ASE
- MySQL Percona
- Maria DB
- IBM DB2

1.11 Supported Applications

The following article presents information on supported applications.

Bacula provides also specific plugins to handle the backup and restore of specific applications on-premise or in the cloud (such as Software as a Service). The following list of applications is supported:

- SaaS
 - Microsoft 365
 - Google Workspace
- On-premise
 - Microsoft Exchange
 - Microsoft Sharepoint
 - Microsoft Active Directory
 - LDAP

2 Bacula Enterprise Architecture

The following document aims at describing the components of Bacula Enterprise independently, and showing how they work together. It is meant for anybody who wants to have basic understanding of how Bacula is designed.

Bacula Enterprise is made up of the following three core components that work smoothly together:

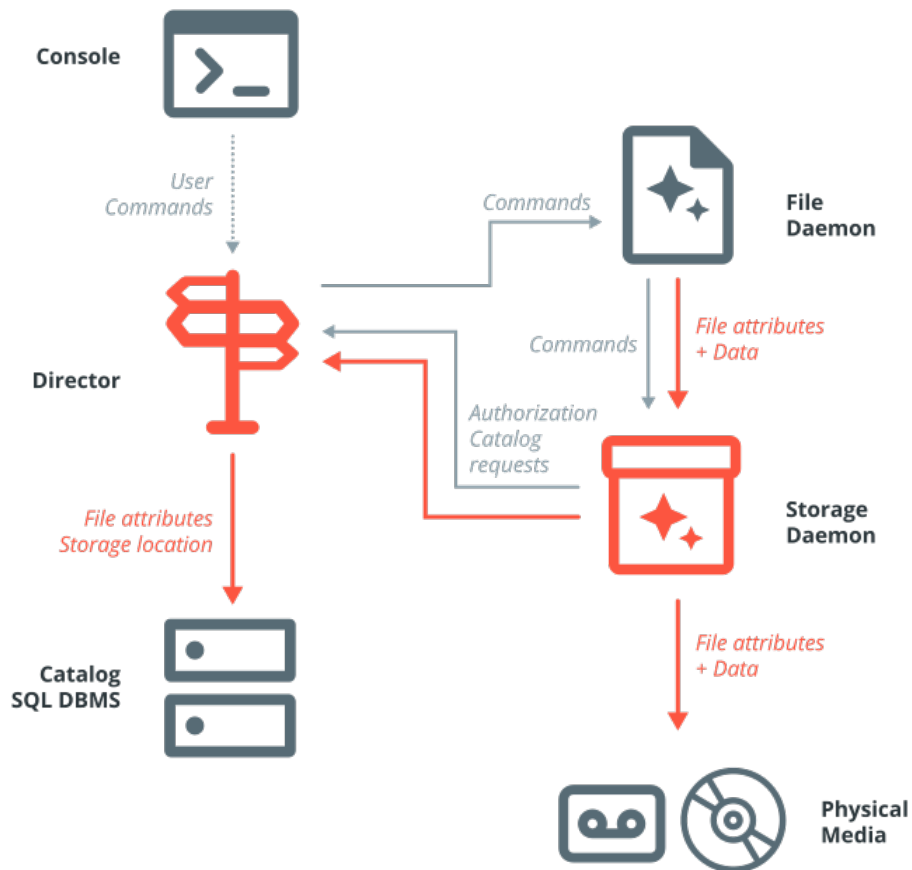


- *Director*
- *Client (File Daemon)*
- *Storage Daemon.*

What is more, vital part of the Bacula architecture are also components/services such as:

- *Catalog*
- *Console*
- *Plugins.*

The flow between particular components is quite simple.



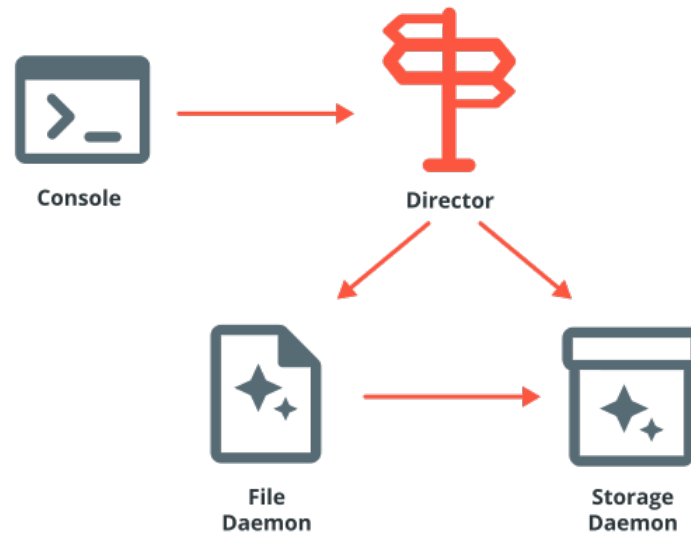
When there is a need to perform any operation in Bacula, one can access the Console and order it. The Console informs the Director about the requested operation, e.g. backup. The Director, being the main Bacula server, schedules and directs the operation. It contacts the chosen Client (File Daemon), and points to the files to be backed up and informs it where it will be saved. It also communicates with the Storage Daemon and suggests where the SD should back up the

chosen data and file attributes to Pools and Volumes. The Client sends the data to the Storage Daemon. At the same time, completely automatically, the Catalog is updated with the information about the operation.

The Bacula Architecture also contains *Plugins*. This component is strictly connected to the File Daemon, or Storage Daemon as the plugins are added to the File/Storage Daemons. They facilitate the backup operation of certain files, e.g. the ones that are managed by another application.

2.1 Fundamentals: Director

The Bacula Director is a program that supervises all the backup, restore and verify operations. System administrator use the Bacula Director to schedule backups and to recover files. The Director runs as a daemon service in the background. Bacula Enterprise users access the Director to do backups or restores of their files through a Console like **BWeb** or **bconsole**.



2.2 Fundamentals: Client (File Daemon)

The Bacula Client service (also known as the File Daemon) is a software program that is installed on each machine to be backed up. It is specific to the operating system on which it runs, and is responsible for providing file attributes and data when requested by the Director. The Client services are also responsible for the filesystem-dependent part of restoring the file attributes and data during a recovery operation. This program runs as a daemon on the machine to be backed up.



Bacula Enterprise Clients, distributed as binary packages, are available for a wide variety of Operating Systems:

- Windows

- GNU/Linux based platforms
- MacOS
- FreeBSD
- Solaris
- HP-UX
- AIX
- and many more.

Open a [support ticket](#) if you are looking to backup a platform you don't find documented.

Consult the `TechnicalReferenceFileDaemon` chapter to know more about the Client resource and all possible directives to configure it.

2.3 Fundamentals: Storage Daemon

The Bacula Storage service is a software program that performs the storage and recovery of file attributes and data to the physical backup media or volumes. In other words, the Storage Daemon is responsible for reading and writing your tapes (or other storage media, e.g., files). The Storage service runs as a daemon on the machine that has the backup device (for example a tape drive or a large disk).



The Storage Daemon can write to multiple storage types:

- its local storage
- any attached disk through Fiber Channel, iSCSI, SAN, etc.
- tapes
- clouds.

The data written can be compressed, encrypted or deduplicated. Consult the `StorageDaemonResourceTypes` chapter to know more about the Storage Daemon resource and all possible directives to configure it.

2.4 Fundamentals: Catalog

The Catalog (Bacula database) service is responsible for maintaining the bacula database for all files backed up. The Catalog services allow system administrators or users to quickly locate and restore any desired file. The Catalog services sets Bacula apart from simple archiver programs like tar and dump, because the Catalog maintains a record of all Volumes used, all Jobs run, and all Files saved, ensuring efficient restoration and Volume management.

Bacula can run with three SQL database backends: MySQL, PostgreSQL and SQLite. We strongly recommend using SQLite only for testing purposes. The two others, MySQL and PostgreSQL, provide quite a number of features, including rapid indexing, arbitrary queries, and security.

Note: PostgreSQL is strongly recommended, and the only Catalog database engine supported with new Bacula Enterprise installations.

To keep the Catalog to a manageable size, the backup information should be removed from the Catalog after the defined File Retention Period. Bacula provides the mechanisms for the Catalog to be automatically pruned according to the retention periods defined.

2.5 Fundamentals: Console

The Bacula Console is a program that allows system administrators or users to communicate with the Bacula Director. Bacula Consoles are available with different user front ends: text-based console interface, graphical user interface, and web interface. The first and simplest is to run the **bconsole** program in a shell window (i.e., TTY interface). Also, the most complete tool being **BWeb Management Suite** available in the Bacula Enterprise version.

If you want to give access to a particular user (not overall Bacula administrator), you need to configure a Console for them. In particular, it may be desirable to implement specific Access Control Lists to prevent users from accessing data they are not authorized for. This part is covered in the Console Configuration chapter.

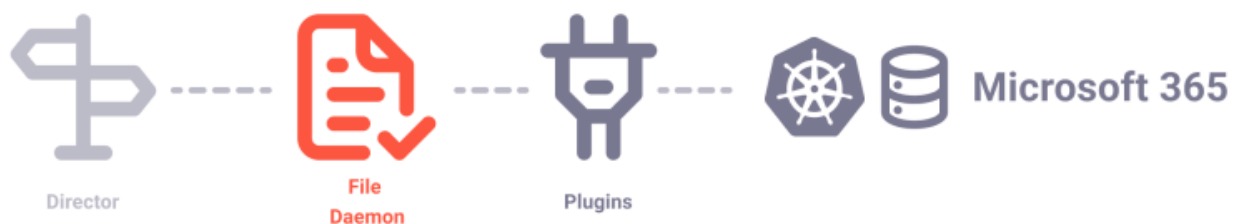
Note: If you wish to learn about BWeb authentication methods, [click here](#).

2.6 Fundamentals: Plugins

There are two kinds of plugins: one related to the Bacula Storage Daemon, and the other for the Bacula File Daemon. You can distinguish between them by their names:

- a storage daemon plugin is named `xxxx-sd.so`
- a file daemon plugin is named `yyyy-fd.so`

Be sure to install the plugin in the right place (on the File Daemon or the Storage Daemon), and read the related documentation and to set it up correctly before use.



Every plugin is dedicated to a specific set of technologies, usually to an enterprise-grade solution such as PostgreSQL, Kubernetes, or VMware. By default, when working without special plugins, Bacula backs up files and nothing else. To learn more about plugins, [look here](#).

Note: To see the list of available plugins, visit [Bacula Enterprise Dedicated Backup Solutions](#).

3 Bacula Enterprise Explained

The following articles aims at presenting the nomenclature regarding Bacula solution, familiarizing the reader with the main concepts of Bacula Enterprise, and explaining the logic behind the solution.

3.1 Fundamentals: Jobs

The Job is a basic unit in Bacula, and is run by the Director. Each Job resource definition contains the name of a Client and a Fileset, a Schedule for the Job, location for data to be stored, and specification for Pools of Volumes to be used. In effect, each Job resource must specify:



Who

In Bacula's terminology is the Client or the machine that is to be backed up.

See the [TechnicalReferenceDirClientResource](#) chapter to know more about the Client.

What

Fileset or the list of files to include and/or exclude.

A Fileset resource is required for each backup Job. It consists of a list of files or directories to be included, a list of files or directories to be excluded and various backup options such as compression, encryption, and signatures that are to be applied to each file.

To get more details, see the [TechnicalReferenceDirFilesetResource](#) chapter.

Where

It is defined by the Storage, Pool, and Catalog, where the Storage specifies on what physical device to backup the files, the Pool defines on what specific Volume, and the Catalog specifies where to keep track of it all.

Bacula Enterprise supports the following type of storage:

- Local disk on the host running the SD
- Local file system via remote storage (SAN, NFS)
- Cloud Storage (Azure, Oracle, GCP, Private clouds, Amazon among others)
- Deduplicated storage (ZFS, NetApp, DELL/EMC, HPE among others) via the Aligned plugin
- Local storage or NFS storage with the Global Endpoint Deduplication plugin
- Tape libraries supported by the Linux Kernel through mt

- FiFo

Director Storage

Each Storage resource is configured in the Director, and it defines the Storage that can be used in Jobs. Each Director Storage resource points to an Autochanger or individual device on a Storage Daemon server.

To get more details, see the [TechnicalReferenceDirStorageResource](#) chapter.

Storage Daemon Autochangers

In Bacula, SD Autochanger resources may contain one or more tape or drive devices. Autochangers are a way to group multiple devices into one resource. Jobs can be configured to use an Autochanger, and Bacula will automatically choose the tape or drive device that will be used to read or write the data. When multiple concurrent jobs are using the same Autochanger, Bacula is able to spread the jobs among the available devices in the Autochanger.

To get more details, see the [TechnicalReferenceSDAutochangerResource](#) chapter.

Storage Daemon Device

The SD Device resource allows configuring a device used for writing to and reading from backup volumes. The Device can point to a tape drive or directory on the filesystem. Multiple Devices can be grouped into an Autochanger.

To get more details, see the [TechnicalReferenceSDDeviceResource](#) chapter.

When

It is defined by what Schedule is used.

The Schedule resource provides a means of automatically starting Jobs at specific intervals.

To get more details, see the [TechnicalReferenceDirScheduleResource](#) chapter.

Types of Jobs

Those jobs are defined in the Job Resource. There are several types of jobs in Bacula:

- Backup
- Restore
- Admin
- Verify
- Copy
- Migration

Backup Job

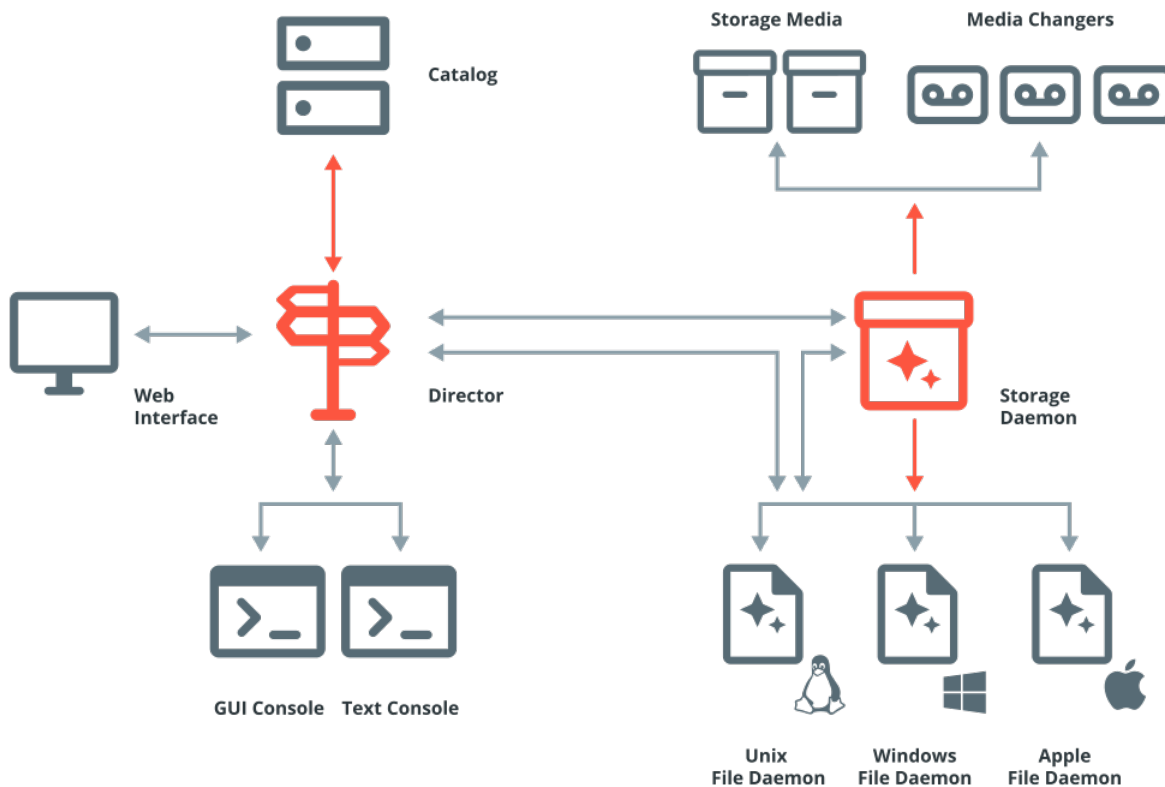
The following article aims at explaining what backup is, and introducing its types and levels.

In general, Backup refers to a Bacula Job that saves files. There are numerous types of backup. If you wish to read more about them, see the list below:

Flow

As shown in in the figure below, when running a backup job:

- the Bacula Director will connect to the File Daemon and Storage Daemon,
- the Director will then send to the File Daemon the necessary information to actually realize the backup job, like the Level, the File Set, the Storage Daemon to contact, etc.
- then the File Daemon will contact the Storage Daemon, identify the Data to be sent and send them as well as the related meta data (file attributes) to the Storage Daemon
- the Storage Daemon will put them into one or more Bacula Volumes, according to the Pool Resource chosen for the Job
- the Storage Daemon will send metadata back to the Director.



The moment Bacula will run a Backup Job depends on how this job is started:

- manually through a Console
- automatically by a defined and referenced Schedule
- automatically by an external script or command like

```
bconsole -c bconsole.conf << EOF
run job=name-of-the-job ...
EOF
```

Levels of Backup Jobs

Full

Backup that saves all files that are defined in the Fileset.

Differential

Backup that includes all files changed since the last Full backup.

Note: Other backup programs define Differential and Incremental differently from the way Bacula uses the terms.

Incremental

Backup that backs up all files changed since the last Full, Differential, or Incremental backup started. Incremental is normally specified on the Level directive within the Job resource definition, or in a Schedule resource.

Virtual Full

Virtual Full Job provides a way to consolidate several jobs into one, without requesting any data from the client, based only on existing backup data.

It relies on a **Next Pool** directive described in the Virtual Full Jobs article.

Virtual Full backups are a way to do backups in an “Incremental forever” style while continuing to provide Full backups at the same time.

When backing up data at incremental or differential levels, Bacula (by default) does not do anything regarding removed or moved files or directories. Which implies that the result of a restoration could be different than the latest state of the machine. For that reason, we advise using “Accurate” mode which is enabled by the directive of the same name. When set to “yes” in a Job, Bacula will record removed missing files or directories, and depending on additional configuration, Bacula will also consider more criteria than just time stamps to determine if a file needs to be backed up. In this case, Bacula will restore the machine to the exact same state (from a backup content point of view) that it was in during the backups.

Accurate Backup

A normal backup works by comparing the creation time and the last modification time of a file against the previous backup time. Differential and Incremental backups will backup only those files with a more recent timestamp. If files have been restored, copied or moved into the Fileset and set to their previous times, or if a directory is moved within the Fileset, those files may not be backed up because they will have old time stamps. As a consequence, with normal backups it is necessary to do periodic Full backups to ensure that you have a copy of all files. Using Accurate backup overcomes these difficulties - the files will be backed up even when they were previously skipped. It costs a bit more overhead in the server to send a list of all cataloged files to the client, and there is some overhead in the client to save and process this list. However, when it is done, whether or not a file is saved depends not just on its time stamps, but also on whether or not it is in the Catalog list (in fact, the criteria to determine if a file is to be backed up can be configured). If the file is not in the Catalog list, it will be backed up even if it has an old time stamp. This effectively resolves the problem of moved files and directories that keep old time stamps and permits fully accurate backups.

Block Level

Some backup programs monitor when a block is modified in the system. Only the modified blocks will be saved during Incremental backups. This kind of backup requires the program to have a very detailed view of the file system, and in general is not very portable. Bacula does not do this kind of backup.

Mapped Raw Backup

Some programs are able to do a raw device backup. They understand the file level structure of the raw device so that they can do a restore file by file. This kind of backup is not very portable. Bacula does not support mapped raw restore.

Mirror Backup

Mirroring is similar to replication in that the data is replicated to another site, but it is typically done immediately when the data is written rather than done in off hours as is the case with replication. Often a whole system will be mirrored, or sometimes only particular disk drives will be mirrored. Mirrored systems are very useful for maintaining a hot spare of a computer system when up time is critical.

Mirroring

Mirroring is similar to replication in that the data is replicated to another site, but it is typically done immediately when the data is written rather than done in off hours as is the case with replication. Often a whole system will be mirrored, or sometimes only particular disk drives will be mirrored. Mirrored systems are very useful for maintaining a hot spare of a computer system when up time is critical.

Offsite Backup

Offsite backup typically involves making a second copy of the backup data, which is sent off-site, most often encrypted. These off-site copies are used for disaster recovery or in case an on-site backup Volume gets irrecoverable errors.

Raw Device Backup

Raw device backup can be used for backing up very large partitions very quickly, because the whole device is backed up without looking at the file system structure. This can be used on some high performance databases that access the database using raw device interfaces. Bacula can backup and restore a raw device. For restore, Bacula must have exclusive use of the device.

Replication

Replication is reproducing the data on another machine. It in itself is a form of backup, and is typically used with databases, where a central database will collect all the changes made in a number of decentralized databases, then replicate the consolidated data in the central database to all the decentralized databases. The advantage is that the primary store of the data is in the central database, but each decentralized instance can have a full copy of all the data, and hence good response times for local users. Often, replication is done in off hours much like backups. Off-Site Off-Site backup typically involves making a second copy of the backup data, which is sent off-site, most often encrypted. These off-site copies are used for disaster recovery or in case an on-site backup Volume gets irrecoverable errors.

Server Free Backup

Server free backup (also called network free backup) is typically done on SAN (storage area network) devices. The device is directly connected to a tape drive and on command it can backup the SAN device directly to the tape drive without using the normal network. Bacula does not support server free backup.

Restore

Restore Job is a job that recovers a file from backup media. It is the inverse of a save, except that in most cases, a restore will normally have a small set of files to restore.

Admin Job

Admin Job is a job which does not backup or move any data, but which can be used to execute scripts through the Run Script resource, and Console or Command directives. The Client and the Fileset are required in an Admin Job configuration, just like other Job directives. However, they do not play a role in an Admin Job. The following definition will launch the script “/opt/bacula/scripts/my-script.pl” located on the Director machine according to the Schedule “AdminSchedule”

```
Job {
  Type = Admin
  Run Script {
    Runs When = Before
    Runs on Client = No
    Command = "/opt/bacula/scripts/my-script.pl"
  }
  Schedule = AdminSchedule
  ...
}
```

Verify Job

In general, Verify jobs permit you to compare the contents of the catalog to the file system, or to what was backed up. In addition, to verifying that a tape that was written can be read, you can also use Verify as a sort of tripwire intrusion detection.

If you wish to know more about the job, read a more advanced article: [AFUVerifyJobs](#).

Copy Job

Copy Jobs copy a Job’s data to another volume. This is usually used when the requirements include at least two places where backup data is being kept.

It relies on a **Next Pool** directive.

Refer to Replication: Copy/Migration Jobs for more details.

Migration

Migration Jobs allow the migration of job data from one volume to another one. This is usually interesting when implementing a Disk-to-Disk-to-Tape strategy, or, more general, any sort of multi tiered backup storage system.

It relies on a **Next Pool** directive.

Refer to Replication: Copy/Migration Jobs for more details.

3.2 Fundamentals: Pools

The Pool resource defines the set of storage Volumes (tapes or files) to be used by Bacula to write the data. It permits to configure the data stored on disk volumes, tapes or cloud storage. By configuring different Pools, you can determine which set of Volumes (media) receives the backup data. This permits, for example, to store all full backup data on one set of Volumes into a specific Pool, and all incremental backups on another set of Volumes belonging to another Pool. Alternatively, you could assign a different set of Volumes to each machine that you backup. Another important aspect of a Pool is that it contains the default attributes (Maximum Jobs, Retention Period, Recycle flag, etc.) that will be given to a Volume when it is created. The Pool resource is a key part of a backup strategy as it permits to group data under the same backup policy under the same Pool.

See the `TechnicalReferenceDirPoolResource` chapter to know more about the Pools.

See the Volumes chapter to know more about Volumes.

As mentioned, Pools in Bacula are a way to manage volumes and to manage collections of similar volumes.

Note: You might have either different requirements or even no requirement to separate volumes into several pools.

The examples of Pools:

- Pools: “short-term” and “long-term” to handle short and long retention periods.
- Pools: “big-customer” and “usual-customers” to keep all the backup jobs from your client “Big Customer” in a specific pool while putting all “usual” customers into another one.
- Pools: “r-and-d” and “accounting” to keep R&D backups together and the Accounting ones in another pool, separated from each other.

There can be many reasons to use or not use several Pools. Bacula configuration gives the Backup Administrator the ability to match the defined requirements.

Due to how Bacula manages Volume retention periods, the jobs put into the same pool should have the same retention periods.

See more details on how to convert an enterprise backup policy into a Bacula backup policy [here](#).

Pool Types

Bacula has different kinds of pools:

- Scratch
- Backup
- Recycle

A **Scratch** pool contains volumes that may be with any pool. If no volumes are available, Bacula will look in the Scratch pool and move/supply a volume to that pool.

A **Backup** pool contains volumes intended to keep data from backups for a defined retention period. It can also include Recycled or Purged volumes, i. e. Volumes that are eligible to be overwritten.

A **Recycle** pool contains volumes that have been used in a Backup Pool after the purge process freed them

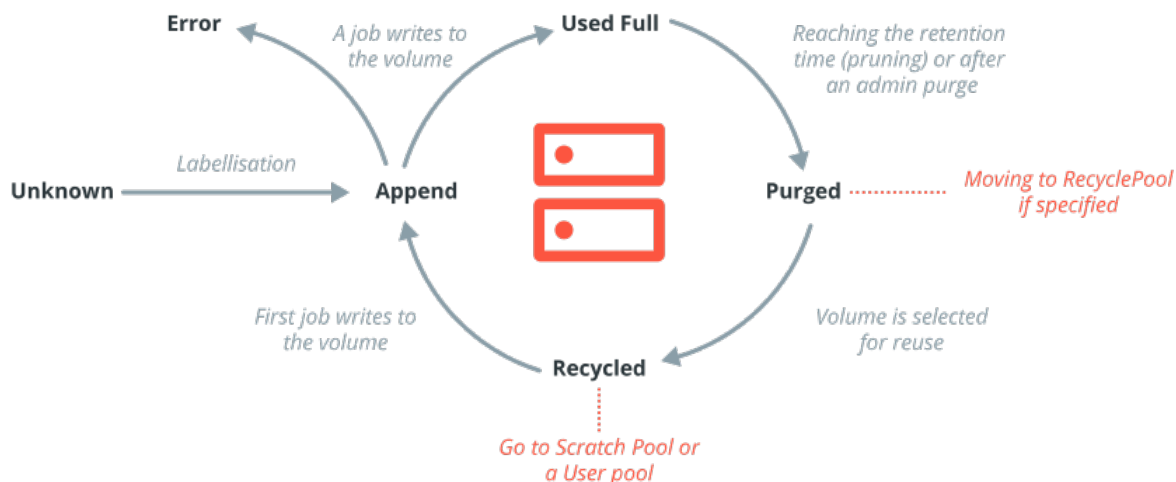
The Bacula configuration allows the Administrator to attach a Scratch Pool and a Recycle Pool to a Backup pool.

3.3 Fundamentals: Volumes

Volume is an archive unit where Bacula stores backed up data, normally a tape or a named disk file where Bacula stores the data from one or more backup jobs. When backing up to tapes, a Volume is identical to a tape and when backing up to disks, a Volume is a file.

Bacula manages all its Volumes in Pools. In particular, a Job is not configured to write to any particular Volume, but to a set of Volumes called a Pool. Volumes of Bacula are always members of only one Pool. There can however be multiple pools.

The volume cycle management process is shown in the figure below:



The Volume retention period is defined as the time Bacula will not overwrite a Volume after it was last written to. Therefore, if you write to a volume with a retention period of 20 days once, the volume will not be overwritten for at least 20 days. If you write to it seven days in a row, the volume will not be overwritten for at least 27 days. Remember that Bacula will try to keep your data safe for as long as possible.

Storage Capacity Management

If you want to overwrite a backup, you need to overwrite the corresponding volume(s). Additionally, to be able to reuse a Volume, its retention period should expire first.

As we already noted, Bacula will try to keep your data safe as long as possible. This means that you can be in a situation (disk backups only) where your storage space is too small even if you defined correct retention periods. In such cases, you can use the **ActionOnPurge** directive requesting Bacula to truncate disk volumes and thus free storage space. For this to work, the bconsole 'truncate' command needs to be scheduled in an Admin job.

Limiting the Volume Size

Tapes

On tapes, we usually don't need to explicitly limit the volume size, because the volume is identical to the tape which has a limited and defined maximum capacity.

Disks

On disks, on the other hand, as volumes are handled with files, there is a risk of filling the disk with only one volume, so we need to limit the volume size and number of volumes. We explained above how the volume retention periods are handled by Bacula. However, often we need to reuse our volumes as soon as possible to be able to reuse the storage space they are using during their retention periods. In the previous case described under the image, we should have restricted the use duration of the Volume.

There are several directives available to limit the occupied storage space in a Pool: `Maximum Volumes`, `Maximum Volumes Bytes`, etc. To browse through the directives, visit: [TechnicalReferenceDirPoolResource](#).

These limitations can be managed manually (by the Administrator with console commands) or automatically (for instance with scripts or Admin jobs).

Volumes and Pool Modifications

Because of its great flexibility, Bacula can be puzzling. Let's say you defined `Maximum Volume Bytes` to 10 GB and then realize that you want to move it to 15 GB. Then, you will modify your "Pool" resource definition. This is correct and will work for all new volumes created into the Pool. But the existing volumes are not modified by this change. To update the existing volumes already in the Pool with the new values you must update the existing volumes' meta-data.

In `bconsole`, enter `'update volume'` and answer the subsequent questions, or alternatively `'update volume allfrompool=the-pool-name'`, or use one of the graphical interfaces to update the existing volumes.

3.4 Fundamentals: Retention Periods

There are various kinds of retention periods that Bacula recognizes. The most important are `File Retention Period`, `Job Retention Period`, and the `Volume Retention Period`. Each of these retention periods applies to the time that specific records will be kept in the Catalog database. This should not be confused with the time that the data saved to a Volume is valid and available for restore – in many cases, data will be available much longer than any of the Retention Periods configured (the data remains in the Volume until the Volume is recycled or truncated).

More details about Schedules and Retentions.

File Retention Period

`File Retention Period` determines the time that File records are kept in the Catalog database. This period is important. As long as File records remain in the database, you can "browse" the database with a Console program and restore any individual file. Once the files are removed or pruned from the database, these files associated with a backup job can no longer be browsed. File records of a Volume use the most space in the database. As a consequence, you must ensure that regular pruning of the file records is done to keep your database from growing too large.

See the `Console prune` command for more details on this subject.

Job Retention Period

Job Retention Period is the length of time that Job records will be kept in the database. Note that Files records are associated to a job. File Records can also be purged disassociating themselves from a job. In this case, information will be available about the Jobs that ran, but not the details of the files that were backed up. Normally, when a job record is purged, all of its associated file records will also be purged.

Volume Retention Period

Volume Retention Period is the length of time between the last write and the volume's re-usage moment. Simply, it's the period of the files being stored. Bacula will normally never overwrite a Volume that contains the only backup copy of a file. As for the Catalog, it retains information for all files backed up for all current Volumes. Once the Volume is overwritten, data from the Catalog is removed as well. Thus, if there is a very large Pool of Volumes or a Volume is never overwritten, the Catalog database may become enormous. Bacula will normally not overwrite a Volume that contains data still inside its Retention period, but select a Volume which is out of its Retention time for recycling.

3.5 Fundamentals: Pruning

To keep the Catalog to a manageable size, the backup information should be removed (pruned) from the Catalog after the defined File and Job Retention Periods. Bacula by default automatically prunes the Catalog database entries according to the retention periods you define.

The autopruning chapter will give you more details about the pruning process and will assist you to choose between manual pruning and automatic pruning.

Pruning by default occurs at the end of a Job. When doing some tests or if you only have few jobs a day, this could be fine. But as soon as the number of jobs is growing, you would prefer to manage pruning another way to let your Catalog do its best for the backup jobs, not for the database administrative tasks. For more details, [click here](#).

3.6 Fundamentals: Purging

Once all the database records that concern a particular Volume have been pruned, respecting the retention periods, the Volume is said to be purged (i.e. has no more catalog entries).

It is, however, possible to submit commands to Bacula to purge specific information, which will not respect configured retention periods. Naturally, this is something that should only be done with the greatest care.

Bacula will try to keep your data safe as long as possible, thus purging a Volume will not automatically reclaim the used space. If you want to reuse space, you must configure Bacula accordingly.

See the Console purge command for more details on this subject.

3.7 Fundamentals: Truncation

Truncation erases the Volume and its data from disk. Therefore, there is no possibility to recover it. Unlike purging which changes the volume status in the Catalog, but does not touch your Volume or its data leaving it on disk.

4 Bacula Enterprise Guide to Basic Operations

This chapter provides a comprehensive guide to setting up and managing Bacula Enterprise, covering each crucial step from deployment to configurations. It is designed primarily for beginners but also serves as a valuable resource for anyone looking to systematically master the system.

4.1 Add New Local Data Destination

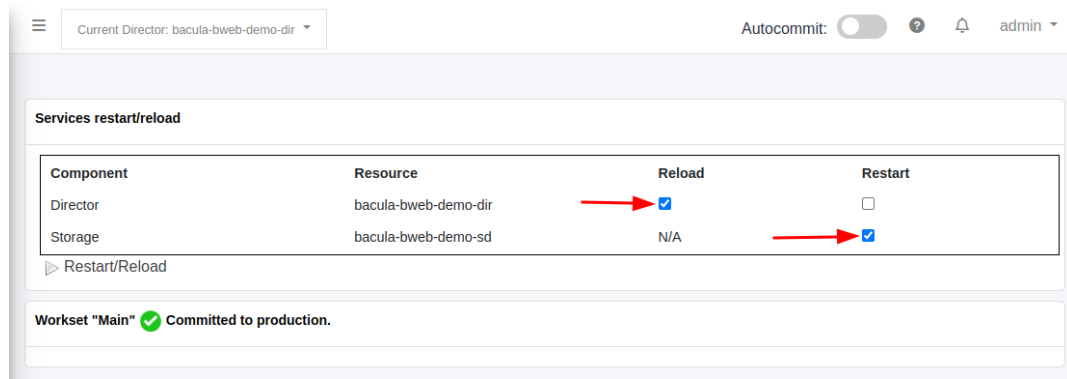
The following article aims at guiding you through the process of adding a new local data destination in Bacula Enterprise, with step-by-step instructions to configure and verify the setup, ensuring a smooth and efficient implementation.

1. Use **Add Storage Device**.
2. Choose **Select a Storage Daemon and add newly created Devices into it**, and choose **Virtual disk autochanger**.
3. Copy from **DiskAutochanger**.
4. Keep all the **default settings**, but set the media type to **localDisk** and the path where volume will be stored to **/baculaVolume/**.
5. Set a name of type to **-storage-localDisk** and set a meaningful description.
6. Make sure you **Commit** changes after reviewing them.

The screenshot shows the Bacula Enterprise web interface. At the top, there's a header with a menu icon, 'Current Director: bacula-bweb-demo-dir', 'Autocommit: [toggle]', a help icon, a user icon labeled 'admin', and a dropdown arrow. Below the header, a section titled 'Workset "Main" (Changes not yet committed to your production)' contains a message: 'The following changes are not yet committed to your production environment.' This is followed by a table with 7 columns: Date, Author, Component, Resource, and Action. The table lists 13 changes, all dated '2024-07-23 12:14' and authored by 'Storage'. The first 12 changes are for 'Device' resources named 'MyDest-storage-localDisk-01' through 'MyDest-storage-localDisk-12', all with the action 'Create'. The 13th change is for a 'Storage' resource named 'MyDest-storage-localDisk' with the action 'Create'. At the bottom right of the table, there are two buttons: 'Commit' and 'Clear Workset'. A red arrow points from the table area towards the 'Commit' button.

Date	Author	Component	Resource	Action
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-01	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-02	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-03	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-04	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-05	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-06	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-07	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-08	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-09	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-10	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-11	Create
2024-07-23 12:14	Storage	Device	MyDest-storage-localDisk-12	Create
2024-07-23 12:14	Storage	Autochanger	MyDest-storage-localDisk	Create
2024-07-23 12:14	Director	Storage	MyDest-storage-localDisk	Create

7. **Reload** the Director and **Restart** the Storage Daemon.

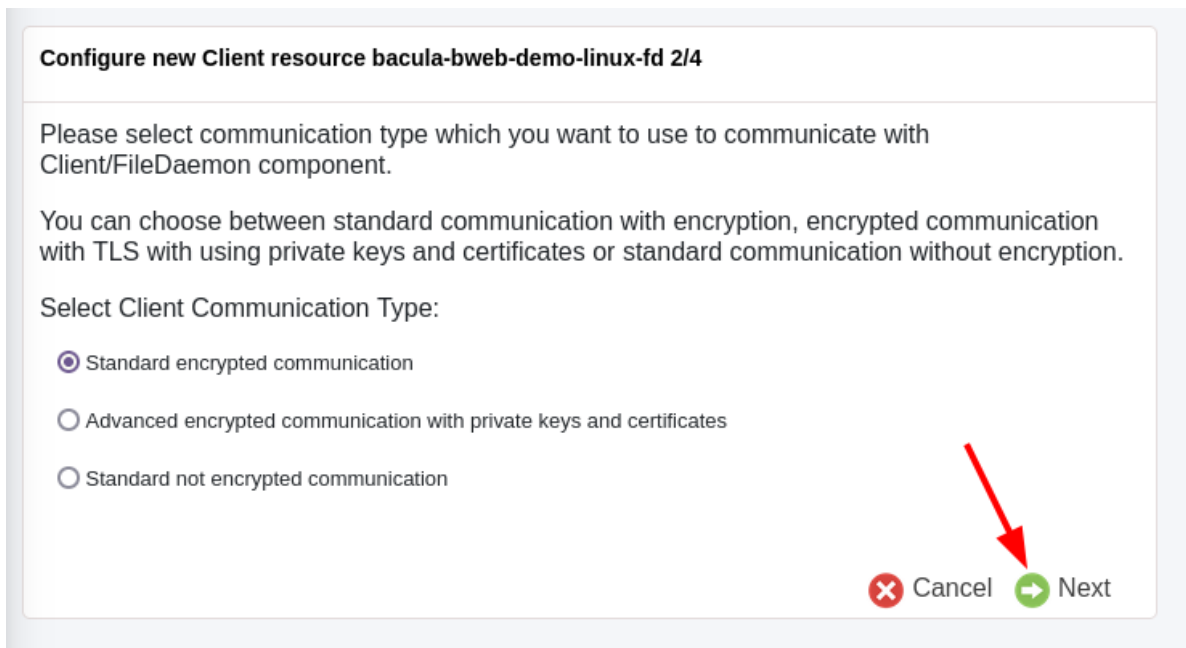


8. Connection via **ssh** to the server, create the directory needed above, and change its **owner/group** to **bacula**.

4.2 Deploy File Daemon On Linux

The following article aims at providing a step-by-step guide to deploying a File Daemon on Linux, covering the entire process from adding a client to configuring the deployment using BWeb Manager Suite.

1. Use **Add Client**.
2. Choose the **Client Name** and add a meaningful **Description**.
3. Select **Standard encrypted communication**.



4. Choose the **OS Type** from the drop-down menu, and insert the host **Address**.
5. Next click on **deploy**.

Configure new Client resource bacula-bweb-demo-linux-fd 4/4

Now you can create a backup Job for this Client. (Note that as the Client is new, the workset is not committed and the Director is not reloaded, so when editing the FileSet, it will not be possible to browse files for this Client)

Or deploy this newly created FileDaemon Resource.

Or view the bacula-fd.conf for this newly created FileDaemon Resource.

Or edit the Client Director Resource.

+ Add a Next Client Resource
✓ OK

6. Review the configuration of the client and click on **Next**.

Push Configuration to bacula-bweb-demo-linux-fd

The component that you are trying to push is not committed to your production configuration.

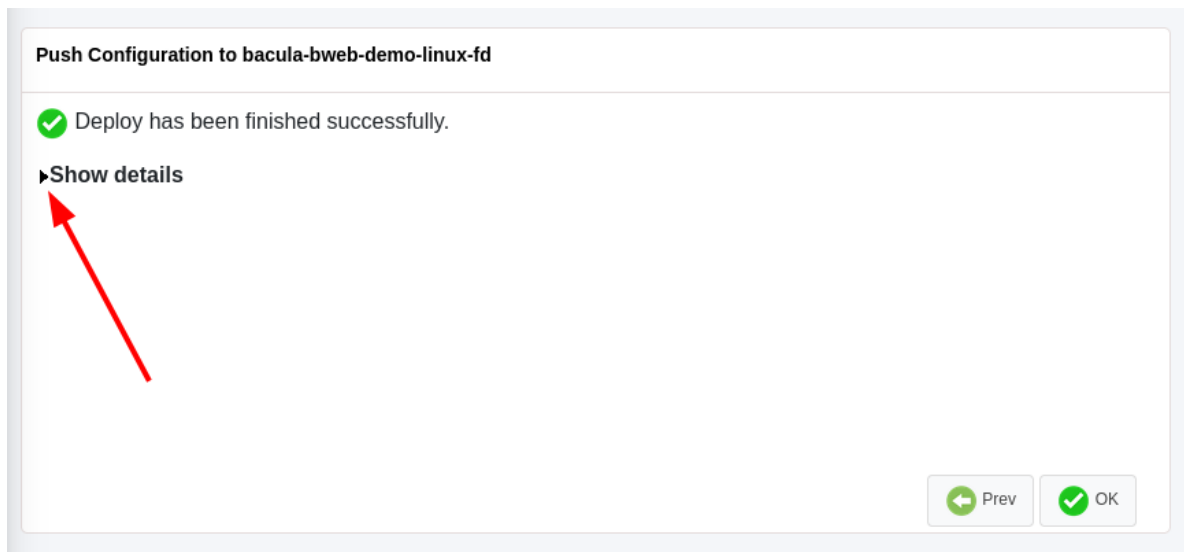
Date	Author	Component	Resource	Action
2024-07-23 12:11	FileDaemon	bacula-bweb-demo-linux-fd	Director bacula-bweb-demo-dir	Create
2024-07-23 12:11	FileDaemon	bacula-bweb-demo-linux-fd	Director bacula-bweb-demo-linux-fd-mon	Create
2024-07-23 12:11	FileDaemon	bacula-bweb-demo-linux-fd	Messages Standard	Create
2024-07-23 12:11	FileDaemon	bacula-bweb-demo-linux-fd	FileDaemon bacula-bweb-demo-linux-fd	Create

If you want to commit and push the configuration, click on Next

Please note, that the Director configuration will be reloaded automatically.

✕ Cancel
➔ Next

7. Choose the **Push Method** Linux/Unix via SSH. Make sure to flag **Install/upgrade binaries** if no previous Bacula installation is present in the target host. Select **Perform the deployment directly using BWeb Manager Suite**.
8. Choose a **Login** method. If you have a key pair, you can connect via SSH. Otherwise, insert root password.
9. You can review the deployment details by clicking on **Show Details**.



It will look something similar to this:

```
INFO: Execute user script '/opt/bweb/bin/deploy_script_linux.sh'
INFO: Checking required files on 10.0.98.35
INFO: Copy configuration files

The authenticity of host '10.0.98.35 (10.0.98.35)' can't be established.
ED25519 key fingerprint is SHA256:/2kIGZAhX8kheY01vCYHkl6CwMVRvqkn248Zi35/fLU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/etc/selinux/targeted/contexts/files/file_contexts: invalid context system_
↪u:object_r:usr_t:s0
Warning: Permanently added '10.0.98.35' (ED25519) to the list of known hosts.

root@10.0.98.35's password:

bacula-push-1930.tar          0%    0    0.0KB/s  --:--  ETA
bacula-push-1930.tar        100% 10KB  5.2MB/s   00:00
INFO: Extract configuration files
INFO: Backing up original configuration from 10.0.98.35
tar: Removing leading `/' from member names
Shared connection to 10.0.98.35 closed.

tar: Removing leading `/' from member names
/opt/bacula/etc/bacula-fd.conf
/tmp/templudtzt7m
/tmp/templudtzt7m.cfg
Shared connection to 10.0.98.35 closed.

INFO: Install binaries
=====
Welcome to Bacula Enterprise Installation Manager 2023.06.29
=====
This script will assist you during Bacula Enterprise installation
```

(continues on next page)

AlmaLinux 9.4 (Seafoam Ocelot) detected [rhel9-64].
 We do best effort on Alma Linux and we recommend to use Rocky Linux

Press ENTER to start

```
=====
Installation of File Daemon (Client) and associated plugins
=====
```

Using Download Area Code [<your-customer-code>]
 Installing command line version 18.0.3

```
[2K
[=====] 4.2% amazon-ec2 (not installed)[2K
[=====] 8.3% antivirus (not installed)[2K
[=====] 12.5% azure-vm (not installed)[2K
[=====] 16.7% big-fs (not installed)[2K
[=====] 20.8% bin ()[2K
[=====] 25.0% cdp (not installed)[2K
[=====] 29.2% delta (not installed)[2K
[=====] 33.3% google-workspace (not installed)[2K
[=====] 37.5% inventory (not installed)[2K
[=====] 41.7% ldap (not installed)[2K
[=====] 45.8% m365 (not installed)[2K
[=====] 50.0% mysql (not installed)[2K
[=====] 54.2% ndmp (not installed)[2K
[=====] 58.3% netapp-hfc (not installed)[2K
[=====] 62.5% oracle (not installed)[2K
[=====] 66.7% postgresql (not installed)[2K
[=====] 70.8% qemu (not installed)[2K
[=====] 75.0% quobyte (not installed)[2K
[=====] 79.2% s3 (not installed)[2K
[=====] 83.3% sap-hana (not installed)[2K
[=====] 87.5% security (not installed)[2K
[=====] 91.7% snapshot (not installed)[2K
[=====] 95.8% vsphere (not installed)[2K
[=====] 100.0% xenserver (not installed)
```

The Bacula Client daemon will be installed or upgraded by default.

The following plugins available for the Client can be installed at version 18.
 ↪0.3 :

```
-----
↪-----
1 : amazon-ec2          2 : antivirus          3 : azure-vm
4 : big-fs              5 : cdp                6 : delta
7 : google-workspace    8 : inventory          9 : ldap
10 : m365               11 : mysql             12 : ndmp
13 : netapp-hfc         14 : oracle            15 : postgresql
16 : qemu               17 : quobyte           18 : s3
19 : sap-hana           20 : security          21 : snapshot
22 : vsphere            23 : xenserver
-----
```

(continues on next page)

```

->-----

=====
Registration of File Daemon (Client) via BWeb
=====

=====
Managing Firewall rules
=====
A potential Director address is detected and will be used by default : bacula-
->bweb-demo
Incoming address bacula-bweb-demo will be accepted
Incoming address 10.0.98.91 will be accepted

=====
Ready to process the following operations
=====
[X] Installation of : Bacula
[ ] Registration of File Daemon (Client) via BWeb
[ ] Managing Firewall rules

=====
Executing Installation of File Daemon (Client) and associated plugins ...
=====
[2K
[=====-----] 50.0% installing keys
Updating the following modules and dependencies : bin ...
[2K
[=====] 100.0% updating bin
Installing the following modules and dependencies : gnupg, tar, bacula-
->enterprise-client ...
[2K
[=====-----] 25.0% yum clean all[2K
[=====-----] 50.0% yum install gnupg[2K
[=====-----] 75.0% yum install tar[2K
[=====] 100.0% yum install bacula-enterprise-client done
Service bacula-fd has been installed. It has been enabled and restarted.

Installation of File Daemon (Client) Successfully completed
=====
Bacula Enterprise Installation Manager. Done.
=====
Shared connection to 10.0.98.35 closed.

INFO: Clean up temporary config files
Shared connection to 10.0.98.35 closed.

INFO: Clean up temporary install files
Shared connection to 10.0.98.35 closed.

```

(continues on next page)

(continued from previous page)

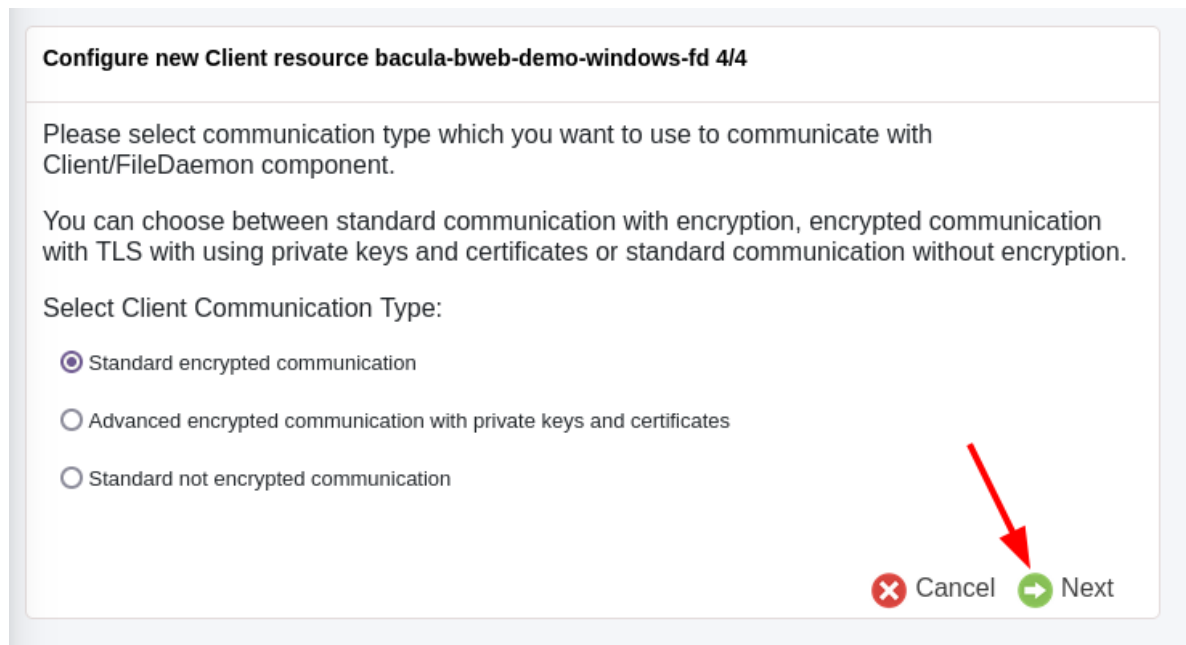
```
INFO: Restarting service on 10.0.98.35  
Shared connection to 10.0.98.35 closed.
```

```
INFO: Clean up
```

4.3 Deploy File Daemon On Windows

The following article aims at providing a step-by-step guide to deploying a File Daemon on Windows, detailing the process from adding a client to configuring and deploying the setup using BWeb Manager Suite.

1. Use **Add Client**.
2. Choose the **Client Name** and add a meaningful **Description**.
3. Select **Standard encrypted communication**.





Configure new Client resource bacula-bweb-demo-windows-fd 4/4

Please select communication type which you want to use to communicate with Client/FileDaemon component.

You can choose between standard communication with encryption, encrypted communication with TLS with using private keys and certificates or standard communication without encryption.

Select Client Communication Type:

- ☒ Standard encrypted communication
- ☐ Advanced encrypted communication with private keys and certificates
- ☐ Standard not encrypted communication

 Cancel  Next

4. Choose the **OS Type** from the drop-down menu, and insert the host **Address**.
5. Next click on **deploy**.

Configure new Client resource bacula-bweb-demo-windows-fd 4/4

Now you can create a backup Job for this Client. (Note that as the Client is new, the workset is not committed and the Director is not reloaded, so when editing the FileSet, it will not be possible to browse files for this Client)

Or deploy this newly created FileDaemon Resource.

Or view the bacula-fd.conf for this newly created FileDaemon Resource.

Or edit the Client Director Resource.

+ Add a Next Client Resource
 ✓ OK

6. Review the configuration of the client and click on **Next**.

Push Configuration to bacula-bweb-demo-windows-fd

The component that you are trying to push is not committed to your production configuration.

Date	Author	Component	Resource		Action
2024-07-29 11:01	FileDaemon	bacula-bweb-demo-windows-fd	Director	bacula-bweb-demo-dir	Create
2024-07-29 11:01	FileDaemon	bacula-bweb-demo-windows-fd	Director	bacula-bweb-demo-windows-fd-mon	Create
2024-07-29 11:01	FileDaemon	bacula-bweb-demo-windows-fd	Messages	Standard	Create
2024-07-29 11:01	FileDaemon	bacula-bweb-demo-windows-fd	FileDaemon	bacula-bweb-demo-windows-fd	Create

If you want to commit and push the configuration, click on Next

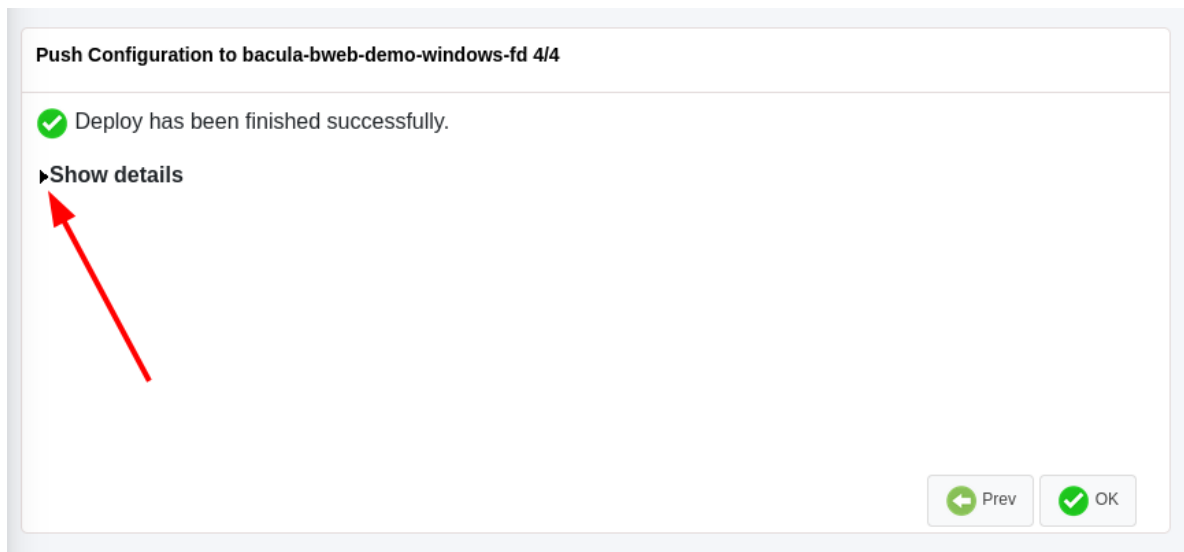
Please note, that the Director configuration will be reloaded automatically.

✗ Cancel
 ➔ Next

7. Choose the **Push Method** Windows. Make sure to flag **Install/upgrade binaries** if no previous Bacula installation is present in the target host. Select **Perform the deployment directly using BWeb Manager Suite**.
8. Insert the credentials for an **Administrator Account** and an **Administrator Password**.

Note: Either a local administrator account or a domain administrator is required to successfully log in.

1. You can review the deployment details by clicking on **Show Details**.



It will look something similar to this:

```
INFO: Execute user script '/opt/bweb/bin/deploy_script_win.sh'
Please enter the password for //10.0.99.135/C$ Administrator:
INFO: Checking required files on 10.0.99.135
INFO: Downloading binaries from the FRS into /tmp/tmp.KbHyOu01cJ
INFO: Copying binaries to 10.0.99.135
INFO: Trying with PSEXEC...
cmd=/c C:/Windows/Temp/bacula.exe /S -ComponentFile
rc=0
stdout=b''
stderr=b''
INFO: Bacula FileDaemon installation succeeded
INFO: Copying configuration to 10.0.99.135
INFO: Backing up original configuration from 10.0.99.135
tar: Removing leading `/' from member names
tar: dumped 1 files and 0 directories
Total bytes written: 1312 (0.1 MiB/s)
putting file /tmp/tmp.jGar12xsef/bacula-config-backup-10.0.99.135-
→20240729113427.tar as \Program Files\Bacula\bacula-config-backup-10.0.99.135-
→20240729113427.tar (1000.0 kb/s) (average 1000.0 kb/s)
INFO: Restarting component service on 10.0.99.135
.
bacula-fd service is stopped.
.
Successfully started service: bacula-fd
INFO: Clean up
```

4.4 Add Backup Policy

The following article aims at guiding you through the process of adding a backup policy, from setting a retention policy to configuring encrypted communication.

1. Use **Add Backup Policy**.
2. Set a **Retention Policy** - e.g. 30 days.

Configure new Backup Policy 1/3

This assistant will guide you in creating a new Retention Policy. The optimal Pool and Schedule configuration will be computed from your requirements, and a Job template (JobDefs) will be created. If you have special requirements such as keeping the first backup of the year for 5 years, we recommend to manually add it later.

Recovery Window

The Recovery Window represents the period during which your files will be available for restore, and the period during which Recovery Points will be taken.

Days

Recovery Points

The number of recovery points or Jobs (Full, Differential and Incremental) that Bacula will keep. For example, having 30 Recovery Points during a Recovery Window of 30 days means that Bacula will run a backup everyday. Having 60 Recovery Points during 30 days means that Bacula will run a backup twice a day. (value used only for estimation)

File Copies

How many copies of each file should be kept in your Storage area? For security, you may want to avoid keeping all backups in a single location.

Refresh Estimation

Cancel

Next

Resource Usage Estimation

Estimation Data

Backup Size:

GB

Change Rate:

%

Estimation Results

Full Backup Size:

Incremental Backup Size:

Average Total Size:

Peak Size:

Size / Number of Copies

Copies	Size (GIB)
1	74.5
2	55.9
3	37.3
4	18.6

3. Choose a name for the **Backup Policy** and add a **Description**.
4. **Review** the configuration and click **Save**.

Configure a new Backup Policy 3/3

We will now summarize your needs and convert them to Bacula resources.

```
JobDefs {  
  Name = "one-month-retention-30-days-recovery"  
  Description = "My Backup Policy"  
  Max Full Interval = 10 days  
  Pool = "Pool_one-month-retention-30-days-recovery"  
  Schedule = "Sched_one-month-retention-30-days-recovery"  
}  
Pool {  
  Name = "Pool_one-month-retention-30-days-recovery"  
  Volume Retention = 40 days  
  Volume Use Duration = 20 hours  
  Maximum Volume Bytes = 70 GB  
  Pool Type = Backup  
  Label Format = "Vol-"  
}  
Schedule {  
  Name = "Sched_one-month-retention-30-days-recovery"  
  Run = Level=Incremental MaxRunSchedTime="2 hours" at 3:00  
}
```

 Cancel  Prev  Save

4.5 Create New Fileset

The following article aims at providing a detailed guide on creating a new Fileset, from selecting the client and directories to configuring advanced options like signature and compression types.

1. On the left menu, navigate to **Configuration > Director > Filesets**.
2. Choose the **Fileset Name** and add a meaningful **Description**, then click on **Add Include list**.
3. From the drop-down menu **Choose Client to browse**, select the correct client. Then expand the root directory and select the folders to include.
4. Then, at the bottom right, click on **Advanced Options**.

Includes/Plugins

/home

+ Add Entry

Excludes

+ Add Entry


Advanced Options

Cancel

OK

- Choose the **Signature** and **Compression** types, and click on **Apply**.
- In the **Include** section, verify the folders previously selected, the signature, and the compression. Then click on **Add** in the top right corner.
- Make sure your new fileset is included in the **Filesets** list, and click on the notification area.

Current Director: bacula-bweb-demo-dir

Autocommit: ☐ ?  admin

Filesets

Search...

Name	Description
BaculaCatalog	
BaculaConfigs	
FullSet	
LinuxEtc	
LinuxHome	
MyFileSet	

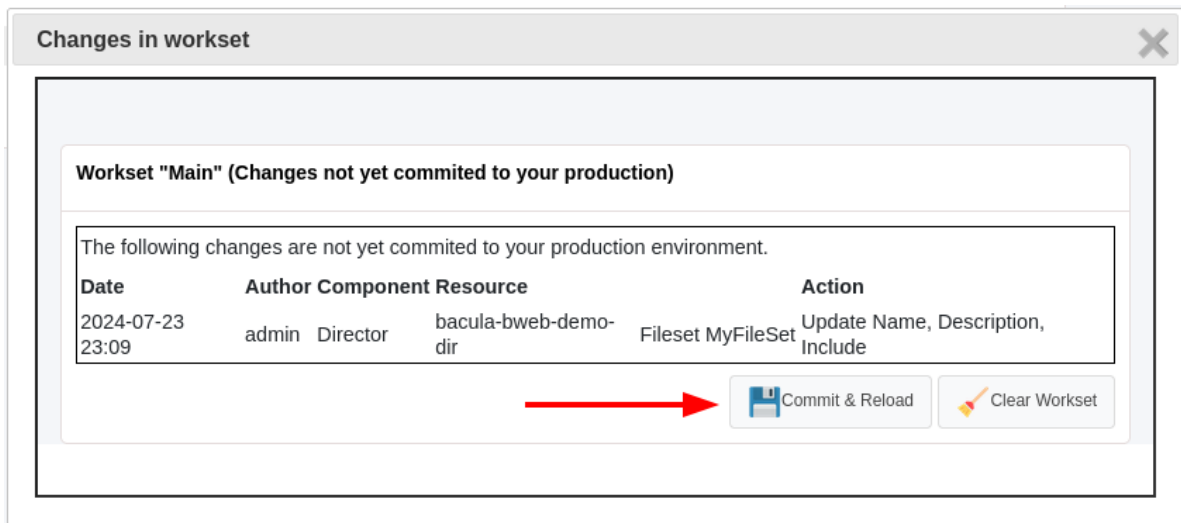
+ Add

Edit Defaults

Help

More...

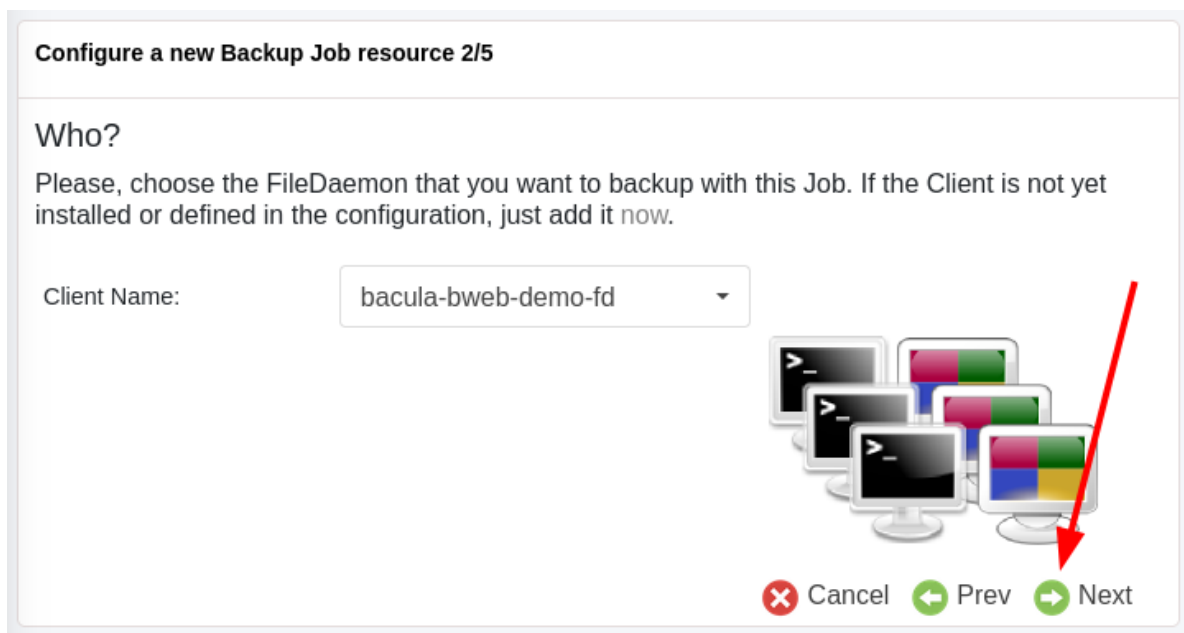
- Review all the changes and click on **Commit & Reload**.



4.6 Add Backup Job

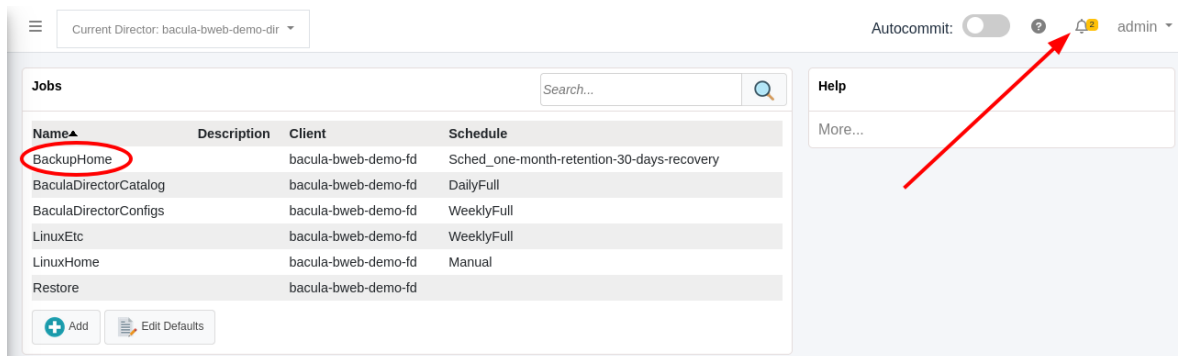
The following article aims at providing a comprehensive guide to adding a Backup Job, detailing each step from creating a job and selecting configurations to reviewing and committing the setup.

1. Use **Add Job**.
2. In the **Backup** tile, click on **Create**.
3. Choose a **Job Name** and add a meaningful **Description**. In the **Job Template** drop-down menu, select the previously created Backup Policy.
4. Set **Who** - Select the correct client in the **Client Name** menu.

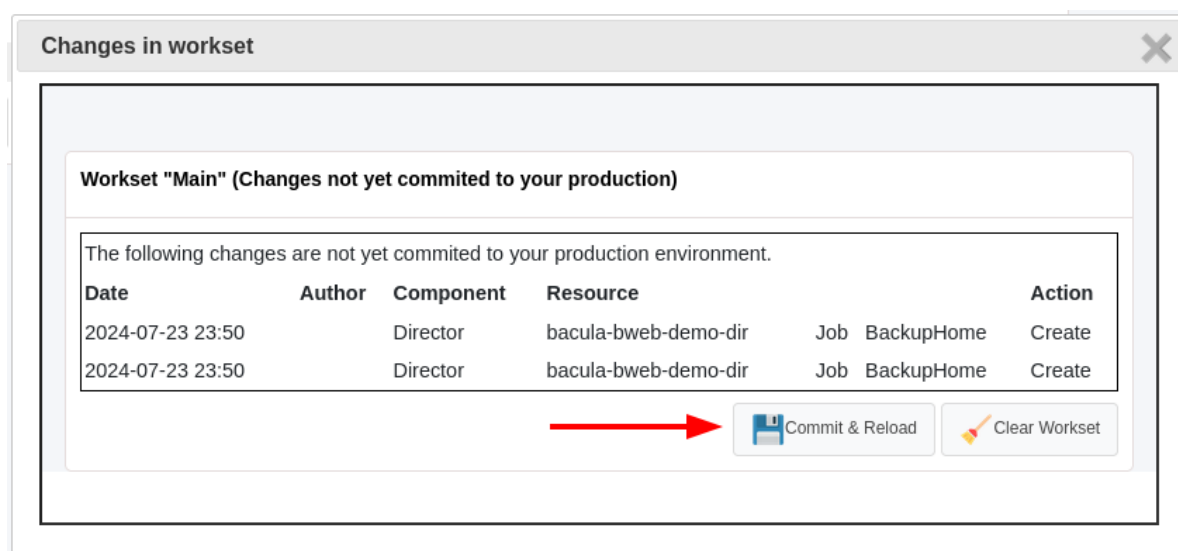


5. Set **What** - Select the previously created Fileset in the **Fileset Name** menu.
6. Set **Where** - Choose the **Pool Name** and the **Storage Name**.
7. Set **When** - Define the **Schedule Name** in the drop-down menu.

- Check out the configured **Backup Job** in the **Jobs** tab. If **Autocommit** is not toggled, be sure to click on the **Notification Area**.



- Click on **Commit & Reload** to apply the changes.



4.7 Run Restore Job

The following article aims at providing a step-by-step guide for running a Restore Job, including selecting the backup client, choosing filesets, and specifying restore locations.

- In the left menu, click on **Restore**.
- From the **Backup Client** drop-down menu, select the client where to restore from.
- Select the fileset of the backup, for instance see Create a New Fileset.
- Choose some files to restore. Navigate to the folder and drag them into the **Restore Selection Area**.
- Choose **Where** you wish to store your data, and click on **Run Restore**.
- When restore is completed, you may review the details of the operation.

```
2024-07-29 17:21:39 bacula-bweb-demo-dir JobId 4: Start Restore Job Restore.2024-07-29-17.21.37_01
2024-07-29 17:21:39 bacula-bweb-demo-dir JobId 4: Restoring files from JobId(s) 3
2024-07-29 17:21:39 bacula-bweb-demo-dir JobId 4: Connected to Storage "MyDest-
```

(continues on next page)

```

↪storage-localDisk" at bacula-bweb-demo:9103 with TLS
2024-07-29 17:21:39 bacula-bweb-demo-dir JobId 4: Using Device "MyDest-storage-
↪localDisk-02" to read.
2024-07-29 17:21:39 bacula-bweb-demo-dir JobId 4: Connected to Client "bacula-bweb-
↪demo-fd" at bacula-bweb-demo:9102 with TLS
2024-07-29 17:21:40 bacula-bweb-demo-fd JobId 4: Connected to Storage at bacula-
↪bweb-demo:9103 with TLS
2024-07-29 17:21:40 bacula-bweb-demo-sd JobId 4: Ready to read from volume "Vol-0002
↪" on File device "MyDest-storage-localDisk-02" (/baculaVolume).
2024-07-29 17:21:40 bacula-bweb-demo-sd JobId 4: Forward spacing Volume "Vol-0002"
↪to addr=295
2024-07-29 17:21:40 bacula-bweb-demo-sd JobId 4: Elapsed time=00:00:01, Transfer
↪rate=15.48 M Bytes/second
2024-07-29 17:21:40 bacula-bweb-demo-dir JobId 4: Bacula Enterprise bacula-bweb-
↪demo-dir 18.0.3 (28Mar24):
Build OS:          x86_64-redhat-linux-gnu-bacula-enterprise redhat (Blue
JobId:             4
Job:               Restore.2024-07-29_17.21.37_01
Restore Client:    "bacula-bweb-demo-fd" 18.0.3 (28Mar24) x86_64-redhat-linux-
↪gnu-bacula-enterprise,redhat,(Blue
Where:             /opt/bacula/archive/bacula-restores
Replace:           Never
Start time:        29-Jul-2024 17:21:39
End time:          29-Jul-2024 17:21:40
Elapsed time:      1 sec
Files Expected:    3
Files Restored:    3
Bytes Restored:    0 (0 B)
Rate:              0.0 KB/s
FD Errors:         0
FD termination status: OK
SD termination status: OK
Termination:       Restore OK

```

4.8 Add External Storage Daemon on Different Server

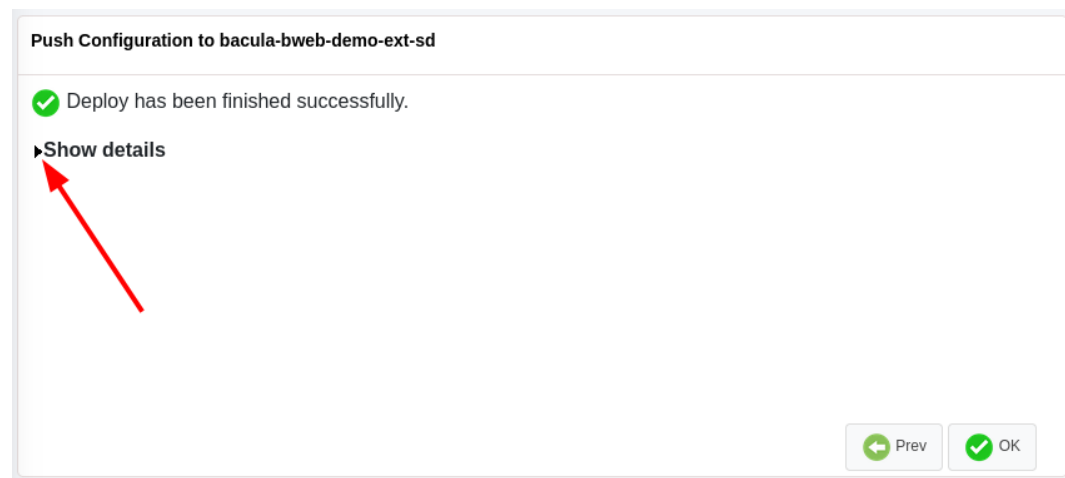
The following article aims at guiding you through the process of adding an external Storage Daemon on a different server, including setting up the daemon, configuring storage devices, and deploying the setup through BWeb Manager Suite.

1. Use **Add Storage Device**.
2. Select **Create a new Storage Daemon and add newly created Devices into it**. Choose the **Storage Daemon Name**, define the **Maximum Concurrent Jobs**, and choose the **Device Type**.
3. Select the **OS Type** from the drop-down menu. Then set the **Address** and **Password**.
4. Choose the file location and define the **Media Type**.

5. Finally, define the **Name** and add a meaningful **Description**.

Note: If **Autocommit** is not toggled, you may need to **Commit** your changes, e.g. see steps 8, 9 in here (open in a new tab).

1. In the left menu click on **SDs Overview**
2. Select the new Storage Daemon and click on **Push**.
3. Check the **Install/upgrade binaries** box and select **Perform the deployment directly form BWeb Manager Suite**.
4. Select the login method.
5. On configuration completed, you may review the details by clicking on **Show details**.



```
INFO: Execute user script '/opt/bweb/bin/deploy_script_linux.sh'
INFO: Checking required files on 10.0.98.185
INFO: Copy configuration files

The authenticity of host '10.0.98.185 (10.0.98.185)' can't be established.
ED25519 key fingerprint is SHA256:/2kIGZAhX8kheY01vCYHkl6CwMVRvqkn248Zi35/
↪fLU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/etc/selinux/targeted/contexts/files/file_contexts: invalid context system_
↪u:object_r:usr_t:s0
Warning: Permanently added '10.0.98.185' (ED25519) to the list of known_
↪hosts.
```

(continues on next page)

(continued from previous page)

```
root@10.0.98.185's password:

bacula-push-1974.tar                0%    0    0.0KB/s  --:--
↳ETA
bacula-push-1974.tar                100%  20KB  10.3MB/s  00:00
INFO: Extract configuration files
INFO: Backing up original configuration from 10.0.98.185
tar: Removing leading `/' from member names
Shared connection to 10.0.98.185 closed.

tar: Removing leading `/' from member names
/opt/bacula/etc/bacula-sd.conf
/tmp/tempQwh_aBEI.cfg
/tmp/tempQwh_aBEI
Shared connection to 10.0.98.185 closed.

INFO: Install binaries
=====
Welcome to Bacula Enterprise Installation Manager 2023.06.29
=====
This script will assist you during Bacula Enterprise installation

AlmaLinux 9.3 (Shamrock Pampas Cat) detected [rhel9-64].
We do best effort on Alma Linux and we recommend to use Rocky Linux

Press ENTER to start

=====
Installation of Storage Daemon and associated plugins
=====

Using Download Area Code [QA_SL_2-adsfJLU783jklAKjd667aJKNyX]
Installing command line version 18.0.3
[2K
[==-----] 9.1% aligned (not installed)[2K
[====-----] 18.2% bin ()[2K
[=====-----] 27.3% cloud-azure (not installed)[2K
[=====-----] 36.4% cloud-glacier (not installed)[2K
[=====-----] 45.5% cloud-google (not installed)[2K
[=====-----] 54.5% cloud-oracle (not installed)[2K
[=====-----] 63.6% cloud-s3 (not installed)[2K
[=====-----] 72.7% cloud-swift (not installed)[2K
[=====-----] 81.8% dedup (not installed)[2K
[=====-----] 90.9% key-manager (not installed)[2K
[=====] 100.0% single-item-restore (not installed)

The Bacula Storage daemon will be installed or upgraded by default.

The following plugins available for the Storage daemon can be installed at
↳version 18.0.3 :
-----
```

(continues on next page)

(continued from previous page)

```
↪ -----
1 : aligned                2 : cloud-azure
3 : cloud-glacier          4 : cloud-google
5 : cloud-oracle           6 : cloud-s3
7 : cloud-swift            8 : dedup
9 : key-manager            10 : single-item-restore
-----
↪ -----

=====
Managing Firewall rules
=====

=====
Ready to process the following operations
=====
[X] Installation of : Bacula
[ ] Managing Firewall rules

=====
Executing Installation of Storage Daemon and associated plugins ...
=====
[2K
[=====-----] 50.0% installing keys
Updating the following modules and dependencies : bin ...
[2K
[=====] 100.0% updating bin
Installing the following modules and dependencies : gnupg, tar, bzip2, ↪
↪ postgresql-server, bacula-enterprise-postgresql ...
[2K
[====-----] 16.7% yum clean all[2K
[=====-----] 33.3% yum install gnupg[2K
[=====-----] 50.0% yum install tar[2K
[=====-----] 66.7% yum install bzip2 curl[2K
[=====-----] 83.3% yum install postgresql-server[2K
[=====] 100.0% yum install bacula-enterprise-postgresql --
↪ exclude=bacula-mysql done
Service bacula-fd has been installed. It has been enabled and restarted.
Service bacula-sd has been installed. It has been enabled and restarted.

Installation of Storage Daemon Successfully completed
=====
Bacula Enterprise Installation Manager. Done.
=====
Shared connection to 10.0.98.185 closed.

INFO: Clean up temporary config files
Shared connection to 10.0.98.185 closed.

INFO: Clean up temporary install files
```

(continues on next page)

(continued from previous page)

```
Shared connection to 10.0.98.185 closed.
```

```
INFO: Restarting service on 10.0.98.185
```

```
Shared connection to 10.0.98.185 closed.
```

```
INFO: Clean up
```

4.9 Configure and Run Verify Job

The following article aims at providing a detailed guide on configuring and running a Verify Job, including selecting and scheduling the job, setting its parameters, and executing it to ensure data integrity.

1. Use **Add Job**.
2. Select the **Verify** tile.
3. Click on **Scheduled Verify Job**.
4. Click on **Verify Data**.
5. Choose a job from the **Verify Job** drop-down menu. See how to Add Backup Job.

Note: If **Autocommit** is not toggled, you may need to **Commit** your changes, e.g. see steps 8, 9 in here (open in a new tab).

1. Choose a **Job Name** and add a meaningful **Description**.
2. **See the Verify Job Summary details and click on Save.**
3. In the left menu, click on **Run Backup**.
4. From the **Job Name** drop-down menu, select the newly created **Verify Job**.
5. Click on **Run Job**.
6. Review the details after operation is completed.

```
2024-07-31 11:50:15 bacula-bweb-demo-dir JobId 4: Verifying against JobId=3
↪ Job=BackupHome.2024-07-31_11.45.14_09
2024-07-31 11:50:15 bacula-bweb-demo-dir JobId 4: Start Verify JobId=4
↪ Level=Data Job=Verify_BackupHome.2024-07-31_11.50.13_23
```

(continues on next page)

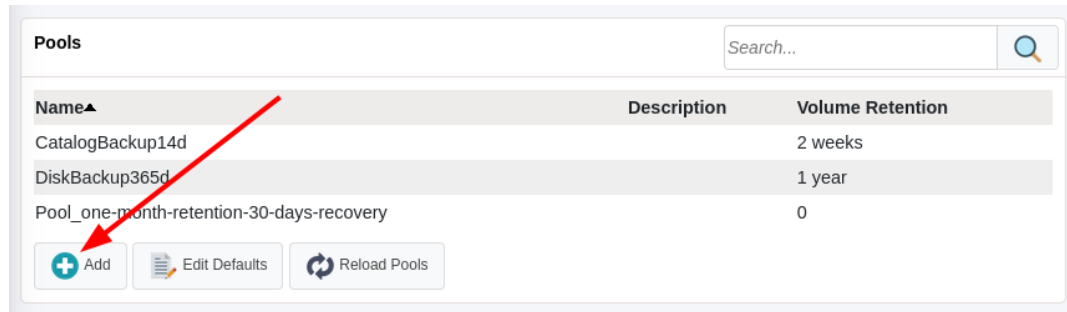
(continued from previous page)

```
2024-07-31 11:50:15 bacula-bweb-demo-dir JobId 4: Connected to Storage
↳ "DiskAutochanger" at bacula-bweb-demo:9103 with TLS
2024-07-31 11:50:15 bacula-bweb-demo-dir JobId 4: Using Device
↳ "DiskAutochanger_Dev0" to read.
2024-07-31 11:50:15 bacula-bweb-demo-dir JobId 4: Connected to Client
↳ "bacula-bweb-demo-fd" at bacula-bweb-demo:9102 with TLS
2024-07-31 11:50:15 bacula-bweb-demo-fd JobId 4: Connected to Storage at
↳ bacula-bweb-demo:9103 with TLS
2024-07-31 11:50:15 bacula-bweb-demo-sd JobId 4: acquire.c:110 Changing
↳ read device. Want Media Type="localDisk" have="DiskVolume"
File device="DiskAutochanger_Dev0" (/opt/bacula/archive)
2024-07-31 11:50:15 bacula-bweb-demo-sd JobId 4: Media Type change. New
↳ read File device "MyDest-storage-localDisk-01" (/baculaVolume) chosen.
2024-07-31 11:50:15 bacula-bweb-demo-sd JobId 4: Ready to read from volume
↳ "Vol-0002" on File device "MyDest-storage-localDisk-01" (/baculaVolume).
2024-07-31 11:50:15 bacula-bweb-demo-sd JobId 4: Forward spacing Volume
↳ "Vol-0002" to addr=295
2024-07-31 11:50:15 bacula-bweb-demo-sd JobId 4: End of Volume "Vol-0002"
↳ at addr=2244 on device "MyDest-storage-localDisk-01" (/baculaVolume).
2024-07-31 11:50:15 bacula-bweb-demo-sd JobId 4: Elapsed time=00:00:01,
↳ Transfer rate=1.289 K Bytes/second
2024-07-31 11:50:15 bacula-bweb-demo-dir JobId 4: Bacula Enterprise bacula-
↳ bweb-demo-dir 18.0.3 (28Mar24):
Build OS: x86_64-redhat-linux-gnu-bacula-enterprise redhat
↳ (Blue
JobId: 4
Job: Verify_BackupHome.2024-07-31_11.50.13_23
Fileset: MyFileset
Verify Level: Data
Client: bacula-bweb-demo-fd
Verify JobId: 3
Verify Job: BackupHome
Start time: 31-Jul-2024 11:50:15
End time: 31-Jul-2024 11:50:15
Elapsed time: 1 sec
Accurate: no
Files Expected: 6
Files Examined: 6
Non-fatal FD errors: 0
SD Errors: 0
FD termination status: OK
SD termination status: OK
Termination: Verify OK
```

4.10 Configure Copy Job

The following article aims at providing a comprehensive guide for configuring a Copy Job, including setting up pools, defining job parameters, and selecting source and destination storage options.

1. In the left menu, navigate to **Pools**.
2. Click on **Add**.



3. Choose a **Pool Name**.
4. Use **Add Job**.
5. Select the **Copy** tile.
6. Insert a **Job Name**.
7. From the **Selection Criteria** drop-down menu, select **Pool Uncopied Jobs**.
8. From the **Source Pool** drop-down menu, choose the pool previously defined.
9. As **Destination Pool**, choose the newly created pool. From the **Destination Storage** drop-down menu, choose the storage to associate to all the volumes of the pool.
10. Click on **Save**.

Configure a new Copy Job resource 5/5

Copy Job Summary

Copy Job Name: **My Copy Job**

Copy Job Description:

Data Source Parameters

Source Selection Criteria: **Pool Uncopied Jobs**

Source Pool: **Pool_one-month-retention-30-days-recovery**

Data Destination Parameters

Destination Pool: **Pool_copy-jobs**

Destination Storage: **MyDest-storage-localDisk**


Maximum Spawned Jobs: **300**


Messages:


Default


Schedule:

Nothing selected

 Cancel

 Prev

 Save



Note: If **Autocommit** is not toggled, you may need to **Commit** your changes.
