MSSQL VSS Plugin

Bacula Systems Documentation

Contents

Contents

This chapter aims at presenting the reader with information about the Bacula Enterprise MySQL Plugin. The document briefly defines the scope of its operations, describes the target technology of the plugin, and presents its main features and various techniques and strategies to backup MySQL with Bacula Enterprise.

1 Scope

The information presented here applies to **Bacula Enterprise**.

This plugin supports MySQL 4.0.x, 4.1.x, 5.0.x, 5.5.x, 5.6.x., 8.0 as well as MariaDB 10.x.

If you want to back up MySQL versions 8.1 or above, please use BE 18.0.4 or above

See also:

Go to:

- *Features*
- *Backup Strategies*
- *Installation*
- *Configuration*
- *Operations*
- *Limitations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

2 Features

The MySQL Plugin is designed to simplify backup and restore of your MySQL server. The backup administrator doesn't need to know MySQL backup techniques or how to write complex scripts. The Plugin will automatically backup essential information such as configuration or user definitions. The MySQL Plugin supports both Dump and binary backup techniques.

The MySQL Plugin is compatible with Copy/Migration jobs. Read the CopyMigrationJobsReplication for more information.

See also:

Go back to:

- *Scope*

Go to:

- *Backup Strategies*
- *Installation*
- *Configuration*
- *Operations*
- *Limitations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

3 Backup Strategies

The following article presents backup strategies for the MySQL Plugin.

3.1 Choosing Between Binary and Dump Modes

The following table article aims at helping you choose between backup techniques supported by the Bacula Enterprise MySQL Plugin. Major functionalities such as being able to restore your database at any point in time, or being able to filter objects during backup or restore should guide you. It is also possible to combine Dump and Binary techniques for the same server.

Functionality	Dump	Binary
Can restore directly a single object (table, schema...)	Yes 1	No
Backup speed	Slow	Fast
Restore speed	Very Slow	Fast
Backup size	Small	Big
Can restore at any point in time	Yes	Yes
Incremental/Differential support	Yes	Yes
Online backup	Yes	Yes
Consistent	Yes	Yes
Can restore to previous major version of MySQL	Yes 2	No
Can restore to newer major version of MySQL	Yes	No

1 To restore a single object, the dump file must be edited.

2 To restore an SQL dump to a previous version of MySQL, you might have to edit the SQL file if you use features that are not available in the previous version. Generally, restoring to a previous version of MySQL is neither supported nor guaranteed.

See also:

Go to:

- *Dump Mode*
- *Binary Mode*

Go back to the *main Backup Strategies page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

3.2 Dump Mode

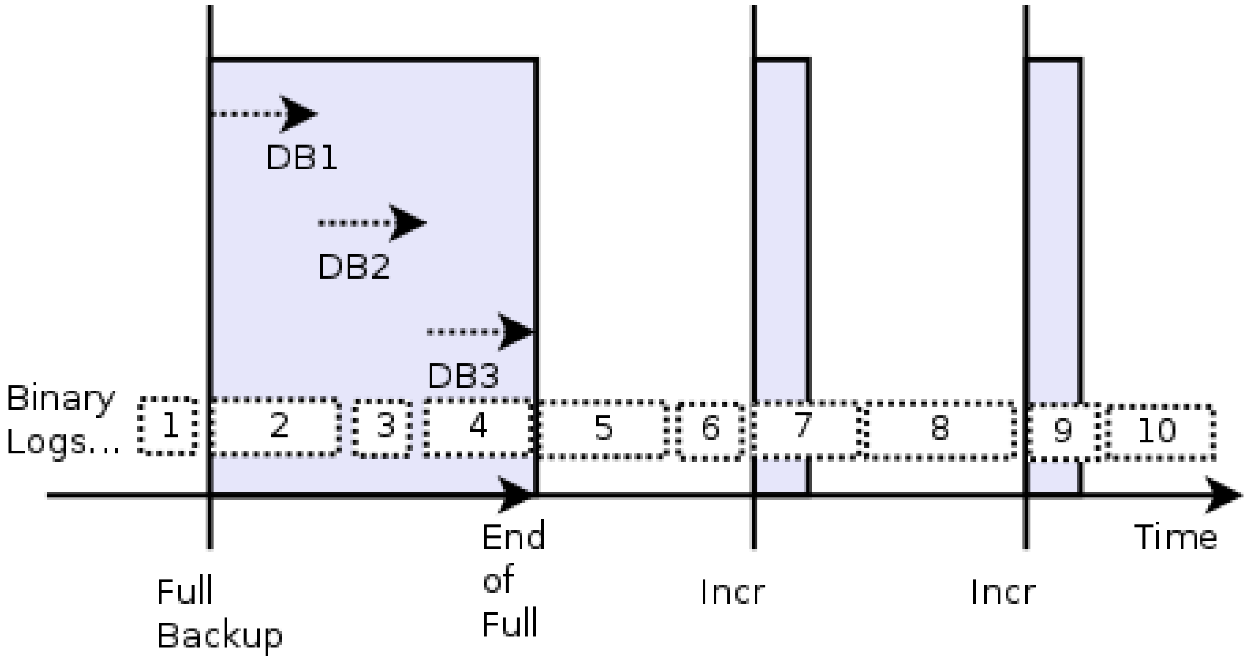


Fig. 1: Interaction between Backup and Binary Logs

During a database's life, MySQL generates logs that can be used to replicate and/or protect your database using P.I.T.R (Point In Time Recovery).

By default, the MySQL Plugin will dump each database one at a time. This means that if you restore the entire server, the databases will be consistent separately, because they were not backed up at exactly the same time, the databases will not be globally consistent. To address this issue, the Bacula Enterprise MySQL Plugin will also save log files generated during the backup. These log files may later be played back to ensure that the databases are consistent at a particular point in time.

In the example presented in the figure above, during the backup of the databases “DB1”, “DB2” and “DB3” (that can take several hours), 3 log files (logs 2, 3, and 4) were generated, and will be included in the Full backup.

The next Incremental or Differential backup will save only new binary logs generated after the Full. To ensure that only one copy of each log file is included in your backup, you should activate the Accurate option for your Job.

In the example shown above, the first Incremental job after the Full backup will include logs 5 and 6, and the second Incremental job will include logs 7 and 8. A Differential backup would include log files 5, 6, 7 and 8.

When you use the `all_databases` option on the Plugin command line, all databases will be dumped at the same time, and the log files will not be flushed at the end of the Full backup, but logs generated before the end of the job are included in the backup. In the example shown in the figure below, the Full backup will generate a single dump `all-databases.sql` and will include log files 2 and 3, but not log file 4. The first subsequent Incremental backup will include log files 4, 5 and 6.

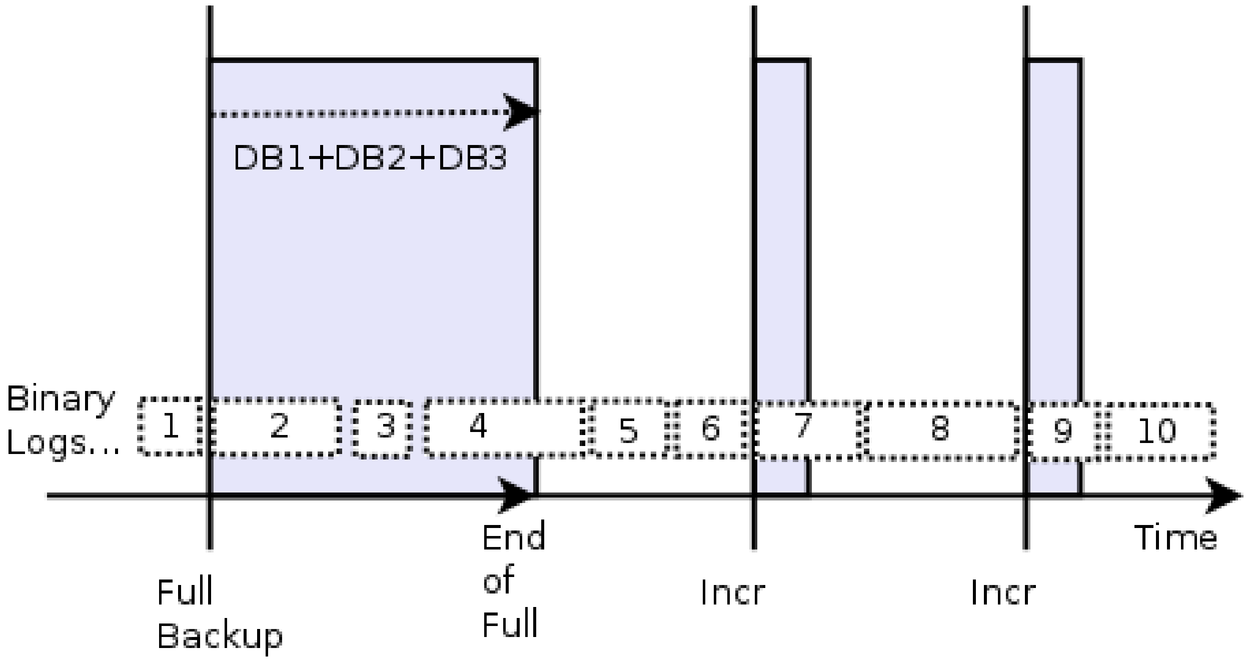


Fig. 2: Interaction between `all_databases` option and Binary Logs

See also:

Go back to:

- *Choosing Between Binary and Dump Modes*

Go to:

- *Binary Mode*

Go back to the *main Backup Strategies page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

3.3 Binary Mode - Percona XtraBackup and Mariabackup

In binary mode, the MySQL Plugin uses Percona XtraBackup, which is an open-source hot backup utility for MySQL based servers that doesn't need to lock your database during the backup. The Percona technology uses techniques that ensure consistency of the whole backup.

It can back up data from InnoDB, XtraDB, and MyISAM tables on unmodified MySQL 5.0, 5.1, 5.5, and 5.7 servers, as well as a Percona Server with XtraDB. MariaDB is supported with Percona xtrabackup, but starting with Bacula Enterprise 12.6, `Mariabackup` is the recommended backup tool for MariaDB 10.1 and above.

See also:

Click *here* to learn how to install Percona Tools.

See also:

Go back to:

- *Choosing Between Binary and Dump Modes*
- *Dump Mode*

Go back to the *main Backup Strategies page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

See also:

Go back to:

- *Scope*
- *Features*

Go to:

- *Installation*
- *Configuration*
- *Operations*
- *Limitations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

4 Installation

The following chapter aims at presenting how to install the MySQL Plugin as well as complementary software.

4.1 MySQL Plugin Installation

Prerequisites

As with all Bacula plugins, you must specify the `Plugin Directory` directive in the `FileDaemon` resource of the `bacula-fd.conf` file.

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

MySQL Plugin Installation with BIM

In order to install the MySQL Plugin with BIM, install the File Daemon with BIM and choose to install the MySQL Plugin during the FD installation.

Click here for more details on the plugin installation process with BIM.

See also:

See an alternative way of installing the MySQL Plugin - *MySQL Installation with Package Manager*.

Go back to the *main MySQL Plugin Installation page*.

Go back to the *main Installation page*.

Go back to the *main MySQL Plugin page*.

MySQL Plugin Installation with Package Manager

The MySQL Plugin is available as a Bacula Enterprise package.

You must also install the plugin on the Client where your MySQL server resides. The MySQL client package, usually "mysql-client" should also be installed, tools such as `mysqldump` and `mysql` must be available to the plugin.

After installing the MySQL Plugin, the File Daemon must be restarted.

See also:

See an alternative way of installing the MySQL Plugin - *MySQL Installation with BIM*.

Go back to the *main MySQL Plugin Installation page*.

Go back to the *main Installation page*.

Go back to the *main MySQL Plugin page*.

See also:

Go to:

- *Percona Tools Installation*
- *Mariabackup Installation*

Go back to the *main Installation page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

4.2 Percona Tools Installation

Note: Starting with Bacula Enterprise 12.6, use *Mariabackup Installation* with MariaDB 10.1 and above instead.

When using Binary Mode, you must install the Percona xtrabackup tool and ensure that the `innobackupex`, xtrabackup, and xbstream programs are properly installed and are available to the plugin.

RPMs and Debs are available on the Percona web site and must be installed prior to usage of the Bacula plugin. More information about Percona and installing the needed programs can be found at:

https://docs.percona.com/percona-xtrabackup/

See also:

Go back to:

- *MySQL Plugin Installation*

Go to:

- *Mariabackup Installation*

Go back to the *main Installation page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

4.3 Mariabackup Installation

Note: Recommended with MariaDB 10.1 and above.

`Mariabackup` is included with MariaDB 10.1.23 and later. It can also be installed with a package manager. See:

https://mariadb.com/kb/en/mariabackup-overview/#installing-with-a-package-manager

See also:

Go back to:

- *MySQL Plugin Installation*
- *Percona Tools Installation*

Go back to the *main Installation page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

See also:

Go back to:

- *Scope*
- *Features*
- *Backup Strategies*

Go to:

- *Configuration*
- *Operations*
- *Limitations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5 Configuration

The following chapter aims at presenting how to configure the MySQL Plugin.

5.1 Automatic Objects Integration

Since Bacula version 16.0.7, a new solution has been introduced, so that each object can be backed up separately with different Jobs to maximize the throughput and the resiliency. It is highly recommended to use this new solution for that purpose - Automatic Object Integration (Scan Plugin). See an example for MySQL.

See also:

Go to:

- *MySQL Specific Configuration*
- *Binary Mode Configuration*
- *Binary Mode Options*
- *Dump Mode Configuration*
- *Dump Mode Options*
- *MySQL Connection Information*
- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.2 MySQL Specific Configuration

In order to use Point-In-Time Recovery feature of MySQL, you need to enable `log_bin` in the MySQL configuration file and then restart the MySQL server. The procedure may differ between major MySQL versions, so we advise you to read the MySQL documentation corresponding to your server version.

```
log_bin = hostname-bin
```

or

```
log_bin = /U01/hostname-bin
```

If you change the `log_bin` parameter after a Full backup, you will need to schedule a new Full backup to back up binary logs in Incremental level properly.

The Bacula Enterprise MySQL Plugin usually is able to detect the `log_bin` path, however, in some cases you might need to specify the `mysqld` configuration file using `config_file` or the `logbin_dir` plugin command option.

By default, the MySQL Plugin uses the `root` account to dump and read MySQL files. On some systems, it is possible to use the `mysql` account. However, on RedHat systems, this account is locked and this is not possible.

See also:

Go back to:

- *Automatic Objects Integration*

Go to:

- *Binary Mode Configuration*
- *Binary Mode Options*
- *Dump Mode Configuration*
- *Dump Mode Options*
- *MySQL Connection Information*
- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.3 Binary Mode Configuration

With the Binary option, the MySQL Plugin uses `Percona` tools to handle Full, Incremental and Differential backups. You must install those tools before using Binary Mode. For more information, see the *Percona Tools Installation*.

```
Job {
 Name = "MySQL-BIN"
 Client = laptop1-fd
 FileSet = FS_mysql
 ...
}

FileSet {
 Name = FS_mysql
 Include {
   Options {
     Signature = MD5
   }
   Plugin = "mysql: mode=binary abort_on_error"
 }
}
```

When backing up in Binary mode, the MySQL Plugin also accepts the parameters listed in the *table* (recommended to open in a new tab).

Binary Backup General

When backup using the `mode=binary` plugin option is done, the database will be backed up in binary mode. But since there are multiple databases (even in the case of a single user database), the database will not be consistent when a restore is done. However, during the binary backup, the Percona tools will save and restore the MySQL binary logs that will permit making the databases consistent.

Making the databases consistent is, in Percona terminology, called “Prepare”. This prepare operation is commonly performed when the databases are restored. They are restored to a temporary location, then made consistent using the Prepare option on the Percona tools prior to actually modifying the live database. This Prepare operation requires having sufficient disk for twice the database size, and it consumes CPU and I/O during the process. During the restoration of a large database, the time and resources that the Prepare phase requires can be significantly high, particularly for large databases.

See also:

Go to *Binary Backup with Prepare*.

Go back to the *Binary Mode Configuration*.

Go back to the *main Configuration page*.

Go back to the *main MySQL Plugin page*.

Binary Backup with Prepare

Rather than doing the Prepare work to make the database consistent at restore time, the Prepare can be performed by the plugin automatically during the backup phase by adding the plugin option `prepare` keyword. Prepare has two options `fd` (default) and `sd`.

When the `prepare=fd` option is specified, the prepare will be done on the File Daemon machine at backup time prior to sending the prepared binary data to the Storage Daemon.

Note: Doing the prepare during the backup allows the restore to be done faster (particularly for large databases), but it requires more disk space, CPU and I/O resources. The additional resources might be undesirable if the File Daemon is running on a critical server (see below for a possible solution to this case).

As an alternative to doing the prepare on the File Daemon, it can be done on the Storage Daemon by using the plugin option `prepare=sd`. With this option there is no additional disk space, CPU or I/O required on the File Daemon. However, the additional disk, CPU, and I/O will be used on the Storage Daemon.

Note: If several File Daemons use the `prepare=sd` option at the same time, the load on the Storage Daemon can increase significantly. By having robust Storage daemons or several Storage Daemons, one can largely mitigate the extra overhead imposed on them.

It is possible to specify xtrabackup options in `/etc/my.cnf` in a **[xtrabackup]** section.

Note: The `prepare` option (either for the FD or the SD) works only with a Full backup. Incremental or Differential backups cannot use the prepare option.

See also:

Go to *Binary Backup General*.

Go back to the *Binary Mode Configuration*.

Go back to the *main Configuration page*.

Go back to the *main MySQL Plugin page*.

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*

Go to:

- *Dump Mode Configuration*
- *Dump Mode Options*
- *MySQL Connection Information*
- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.4 Binary Mode Options

Table 1: MySQL Plugin Options in Binary Mode

Option	Comment	Default	Example
mode	Enable Binary backup	dump	mode=binary
unix_user	MySQL Unix user	root	unix_user=mysql
service	MySQL server information	main	service=main
prepare	Use Percona prepare on the File daemon	fd	prepare=fd
prepare	Use Percona prepare on the Storage daemon		prepare=sd
user	MySQL super user	root	user=root
password	MySQL super password		password=xx
backup_software	MySQL backend (mariadb, xtrabackup, mysql). Used to determine the tools to use.``10``	mysql	backup_software=mariadb
bin_dir	MySQL binaries location		bin_dir=/5.1/bin
bin_format	Binary format (tar or xbstream)	xbstream	bin_format=tar
config_file	Path to my.cnf mysqld configuration file	/etc/mysql/my.cnf	
mycnf_dir	Path where the MySQL connection `.my.cnf` file is stored		mycnf_dir=/opt/bacula/etc
extra_file	Path to mysql connection file 1	/root/extra.cnf	
tmp_dir	`WorkingDirectory` Where the plugin will create files and scripts for the database backup. 2		tmp_dir=/othertmp
timeout	Timeout for SQL queries 3	60 seconds	timeout=1200
abort_on_error	Abort the job if we have MySQL connection problems 4		abort_on_error=true
xtrabackup_tmpdir	Set `tmpdir` option when doing binary backup 5		xtrabackup_tmpdir=/tmp/test
xtrabackup_args	Set additional options when doing binary backup 6		xtrabackup_args="–compress"
restore_extract	Extract xbstream archive automatically at restore 7		restore_extract
innobackupex_tmpdir	Deprecated 8		innobackupex_tmpdir=/tmp/test
innobackupex_args	Deprecated 9		innobackupex_args="–compress"

1 Available with Bacula Enterprise 8.2.4 and later.

2 Available with Bacula Enterprise 6.6.6 and later.

3 Available with Bacula Enteprise 8.6.15.

4 Available with Bacula Enterprise 8.2.9 and later.

5 Available with Bacula Enterprise 8.2.8 and later.

6 Available with Bacula Enterprise 8.2.9 and later.

7 Available with Bacula Enterprise 10.2 and later.

8 Available with Bacula Enterprise 8.2.8 and later. Deprecated since Bacula Enterprise 10.2.

9 Available with Bacula Enterprise 8.2.9 and later. Deprecated since Bacula Enterprise 10.2.

10 May be necessary only in special scenarios such as binary bundle package installations. Available with Bacula Enterprise 16.0.8 and later.

Note that the `tar` option for `bin_format` is not compatible with Incremental backups, so only the Full backup will be stored in tar format. Incremental backups will use the xbstream output format.

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*
- *Binary Mode Configuration*

Go to:

- *Dump Mode Configuration*
- *Dump Mode Options*
- *MySQL Connection Information*
- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.5 Dump Mode Configuration

With the Dump option (as opposed to the Binary option), the MySQL server should be configured with the binary log option to perform Incremental and Differential backups.

Note: If the `mode` plugin option is not specified, the backup will default to `mode=dump`.

```
Job {
 Name = "Mysql-dump"
 Client = laptop1-fd
 FileSet = FS_mysql_dump
 ...
}

FileSet {
 Name = FS_mysql_dump
 Include {
   Options {
     Signature = MD5
     Compression = GZIP
   }
   Plugin = mysql
 }
}
```

In the above example, the plugin will detect and back up all databases of your server. This simple configuration will work only if the `root` account is able to connect to the MySQL database. For more complex configurations, refer to the *options table*.

```
FileSet {
 Name = FS_mysql
 Include {
...
   Plugin = "mysql: database=bacula"
   Plugin = "mysql: database=master"
 }
}
```

In the above example, the plugin will backup databases `bacula` and `master`.

```
FileSet {
 Name = FS_mysql
 Include {
...
   Plugin = "mysql: unix_user=admin tmp_dir=/tmp"
 }
}
```

In the above example, the plugin will backup databases using the “admin” Unix user account. This account should be able to connect with all permissions to all databases that you want to dump. You need to make sure that the `tmp_dir` will be writable to your user.

In Dump mode, the MySQL plugin also accepts the parameters listed in the following *Dump Mode Options* (recommended to open in a new tab).

```
FileSet {
 Name = FS_mysql_dump
 Include {
...
   Plugin = "mysql: user=rob dump_opt=\"--ignore-table=db_name.tbl_name\""
 }
}
```

In this example, the MySQL Plugin will use MySQL account “rob” to perform a dump backup of all databases, and skip the table tbl_name in database db_name.

Note: Since 14.0.

In order to use `sudo` wrapper, you need to comment out the following option in `/etc/sudoers`.

```
Defaults requiretty
```

The MySQL Plugin permits different dump options for each MySQL version:

Table 2: MySQL Dump options

Version	Option
>= 4.1.18	`--single-transaction --opt --extended-insert --create-options --default-character-set=utf8`
>= 5.0	4.1.18 options + `--routines --master-data=2`
>= 4.1	`--single-transaction --opt --extended-insert -all --default-character-set=utf8`
3.x	`--skip-lock-tables --opt --extended-insert --create-options --default-character-set=utf8`

Backup Level in Dump Mode

When using Dump mode, depending on the Job level, the MySQL Plugin will do the following:

- For **Full** backups, the Plugin will backup all databases and logs generated during the backup.
- For **Incremental** backups, the Plugin will flush the current log and will backup all logs generated since the last backup.
- For **Differential** backups, the Plugin will flush the current log and will backup all logs generated since the last Full backup.

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*
- *Binary Mode Configuration*
- *Binary Mode Options*

Go to:

- *Dump Mode Options*
- *MySQL Connection Information*
- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.6 Dump Mode Options

Table 3: MySQL Plugin Options in Dump Mode

Option	Comment	Default	Example
dump_opt	This string will be passed to the mysqldump command		dump_opt="-X"
exclude_table	Exclude a table. Multiple parameter is allowed. If the argument points to a file on the client, each line will be excluded. 5		exclude_table=mysql
unix_user	Unix user to use for MySQL commands	root	unix_user=robert
service	MySQL server name		service=main
mycnf_dir	Path where MySQL .my.cnf file is stored		my_cnf=/tmp
use_sudo	Use sudo instead to run MySQL commands (when not root)		use_sudo
database	Will backup on databases matching this string		database=prod*
all_databases	Will generate a single dump of all databases		
bin_dir	MySQL binaries location		bin_dir=/opt/mysql/bin
user	MySQL super user	root	user=root
password	MySQL super password		password=xx
logbin_dir	mysqld log_bin directory		
config_file	Path to my.cnf mysqld configuration file	/etc/mysql/my.cnf	
extra_file	Path to mysql connection file 1	/root/my.cnf	
character_set	Character set used to dump data	utf8	character_set=utf8
tmp_dir	Where the MySQL plugin will create files and scripts for the database backup 2	/tmp	tmp_dir=/othertmp
timeout	Timeout for SQL queries 3	60 seconds	timeout=1200
skip_missing_db	Send a SKIPPED message instead of a warning when a database disapears during a job 6		skip_missing_db
abort_on_error	Abort the job if we have MySQL connection problems 4		abort_on_error=true
backup_software	MySQL backend (mariadb, mysql). Used to determine the tools to use.``7``	mysql	backup_software=mariadb

1 Available with Bacula Enterprise 8.2.4 and later.

2 Available with Bacula Enterprise 6.6.6 and later.

3 Available with Bacula Enteprise 8.6.15.

4 Available with Bacula Enterprise 8.2.0 and later.

5 Available with Bacula Enterprise 14.0 and later.

6 Available with Bacula Enterprise 14.0.4 and later.

7 May be necessary only in special scenarios such as binary bundle package installations. Available with Bacula Enterprise 16.0.8 and later.

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*
- *Binary Mode Configuration*

- *Binary Mode Options*
- *Dump Mode Configuration*

Go to:

- *MySQL Connection Information*
- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.7 MySQL Connection Information

If you are using specific connection options such as:

- TCP connection
- Non-standard port
- Password
- and others,

it is possible to create a configuration file to store these settings and use them with the Bacula Enterprise Plugin. For example, it avoids having the password exposed on the Plugin command string.

The connection file should be specified with the `extra_file` plugin command line option. In the connection file, the `[client]` section is a shortcut for all required context.

Note: The the `extra_file` plugin option was introduced in Bacula Enterprise 8.2.4 and available with MySQL 5.0.6.

```
# cat /opt/bacula/etc/database1.cnf
[client]
user=admin
password=admin1
socket=/tmp/mysql.sock

# Plugin = "mysql: extra_file=/opt/bacula/etc/database1.cnf"
```

In `bin` backup mode, the xtrabackup tool doesn't read the `.my.cnf` to get connection information. To use the binary backup mode, it is mandatory to specify the password on the Plugin command line or to use the `extra_file` plugin command option.

It is also possible to use the user specific `.my.cnf` MySQL ini file that should contain information for `client` programs such as `mysql`, `mysqldump`.

```
# comment
[client]
password=rootroot
```

MySQL programs will search the `.my.cnf` file in the HOME directory by default. With the Bacula Enterprise MySQL Plugin, the `.my.cnf` file can be stored anywhere on your system. The use of the `mycnf_dir` FileSet option permits to specify the directory where this file is stored.

```
# cat /opt/bacula/etc/.my.cnf
[client]
user=admin
password=admin1

# Plugin = "mysql: mycnf_dir=/opt/bacula/etc"
```

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*
- *Binary Mode Configuration*
- *Binary Mode Options*
- *Dump Mode Configuration*
- *Dump Mode Options*

Go to:

- *Testing Database Access Configuration*
- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.8 Testing Database Access Configuration

You can use the Bacula `estimate` command to verify that the MySQL plugin is well configured.

```
* estimate listing job=my-test
...
```

If the `estimate` or the job output display the following error,

```
Error: Can't reach MySQL server to get database config.
```

you should check that the Bacula Enterprise MySQL Plugin can retrieve information using the `mysql` command running as the `mysql` user on the Client.

You must ensure that your `unix_user` user can connect to the MySQL server without any password prompt. By default, the `unix_user` is root.

If you need to specify options such as `-h localhost` in the `mysql` command line, you will need to use a my.cnf file as described in *MySQL Connection Information*.

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*

- *Binary Mode Configuration*
- *Binary Mode Options*
- *Dump Mode Configuration*
- *Dump Mode Options*
- *MySQL Connection Information*

Go to:

- *Error Log and Debug Information*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

5.9 Error Log and Debug Information

With the MySQL Plugin, MySQL error messages generated by commands are automatically included in the job log.

For example:

```
Fatal error: Can't reach MySQL server to get database config. ERR=268435457
Error: ERROR 1045 (28000): Access denied for user 'backup_user'@'localhost' (using␣
↪password: YES)
```

If you need more details about the MySQL errors or if no error message is present in the backup job log, you can enable debug level 200 on File Daemon to not delete log files in the `/tmp` directory of the File Daemon host at the end of the job.

```
* setdebug level=200 trace=1 options=t client=mysqlserver-fd
```

For example, after enabling debug level 200 on `mysqlserver-fd`, the following files will be generated:

```
root@mysqlserver:/tmp# ls
cmd.24.sh.4OQxiQ
cmd.24.sh.OCuiKJ
cmd.24.sql.0jRa7W
cmd.24.sql.SxlbWM
grants.24.sql.yiwPHT
mysql.24.log
```

```
  Error: Pipe close error 2 on /@MYSQL/main/MyDatabase/data.sql
(sh -c 'mysqldump --opt --extended-insert --create-options --single-transaction
       --default-character-set=utf8 --routines --errorParameter "MyDatabase"
       2>>/tmp/mysql.24.log'): ERR=Child exited with code 2
```

We can see in the output log file an incorrect parameter configured in the FileSet plugin line:

```
root@mysqlserver:/tmp# cat mysql.24.log
mysqldump: unknown option '--errorParameter'
```

Note: If you often encounter errors of the lost connection to MySQL server, visit MySQL Community documentation for advice.

See also:

Go back to:

- *Automatic Objects Integration*
- *MySQL Specific Configuration*
- *Binary Mode Configuration*
- *Binary Mode Options*
- *Dump Mode Configuration*
- *Dump Mode Options*
- *MySQL Connection Information*
- *Testing Database Access Configuration*

Go back to the *Configuration page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

See also:

Go back to:

- *Scope*
- *Features*
- *Backup Strategies*
- *Installation*

Go to:

- *Operations*
- *Limitations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

6 Operations

The following chapter aims at presenting possible operations with the MySQL Plugin.

6.1 Backup

Estimate Information

The estimate command will display all information found by the MySQL plugin. For Dump mode, Bacula cannot estimate the dump size for databases, so it will display database size instead.

Backup Information in Dump Mode

The MySQL Plugin will generate the following files entries in the Bacula catalog for a server having a single database "test".

```
/etc/mysql/my.cnf
@MYSQL/main/gobal-grants.sql
@MYSQL/main/settings.txt

@MYSQL/main/test/createdb.sql
@MYSQL/main/test/schema.sql
@MYSQL/main/test/data.sql
@MYSQL/main/test/grants.sql

@MYSQL/main/logs/mysql-bin.000001
```

Table 4: Backup Content in Dump Mode

File	Context	Comment
global-grants.sql	global	List of all users, their password and specific options
settings.txt	global	Current variables for the mysql server
my.cnf	global	MySQL server configuration
createdb.sql	database	Database creation script
schema.sql	database	Schema database creation script
data.sql	database	Database data in dump format
grants.sql	database	List of all users associated to the database

See also:

Go to *Restore*.

Go back to the main *Operations page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

6.2 Restore

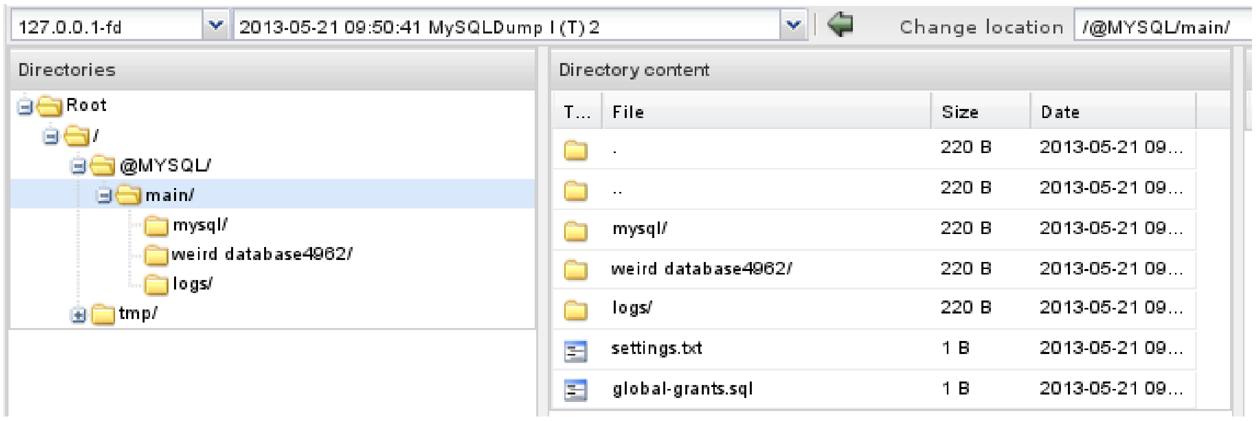
Restoring Using Dumps

Restoring Users and Roles

To restore roles and users to your MySQL server, you just select the `global-grants.sql` file located in: `/@MYSQL/<service>/global-grants.sql`.

Important: With MySQL >= 5.7, users must be created prior to the GRANT command with: `CREATE USER username`

Then, using `where=/` or `where=` the plugin will load this SQL file into your database. If some roles already exist, errors will be printed in the Job log. Note that it is possible to restore the `global-grants.sql` file to a local directory, edit the file and load it with `mysql` to restore only a particular selection.



127.0.0.1-fd | 2013-05-21 09:50:41 MySQLDump I (T) 2 | Change location | /@MYSQL/main/

Directories

```
Root
  /
    @MYSQL/
      main/
        mysql/
        weird database4962/
        logs/
    tmp/
```

Directory content

T...	File	Size	Date
	.	220 B	2013-05-21 09...
	..	220 B	2013-05-21 09...
	mysql/	220 B	2013-05-21 09...
	weird database4962/	220 B	2013-05-21 09...
	logs/	220 B	2013-05-21 09...
	settings.txt	1 B	2013-05-21 09...
	global-grants.sql	1 B	2013-05-21 09...

Fig. 3: MySQL Server content during restore

See also:

Go to:

- *Restoring Single Database*
- *Restoring Dump Files To Directory*
- *PITR Using Binary Logs*
- *Restoring Single Table*
- *Restoring Complete Server*
- *Restoring Using MySQL Command-Line Tool*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

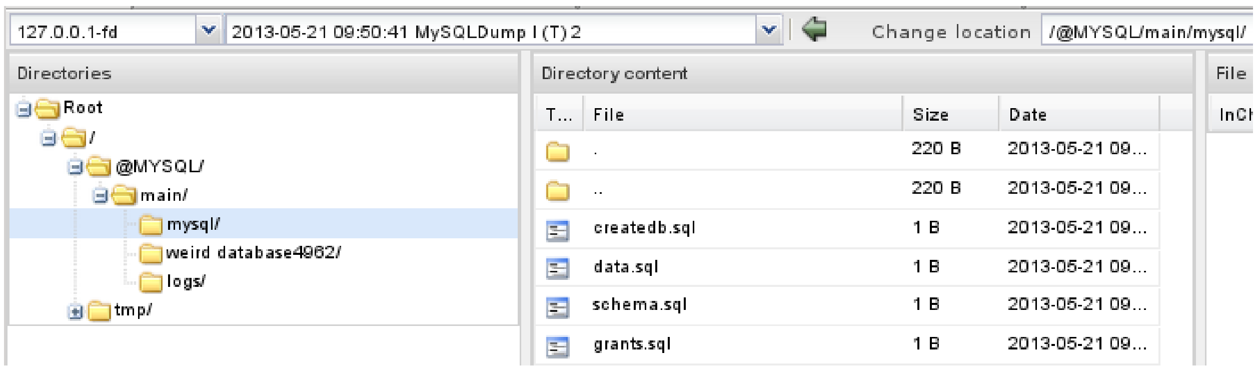
Go back to the main *Operations page*.

Restoring Single Database

To restore a single database with the Bacula Enterprise MySQL Plugin, you need only to select the database directory in the restore command, the selection should contain the data file (`data.sql`) and the database creation script (`createdb.sql`).

When the database directory is selected, you can use the `where` parameter to restore the database to a new database. If you set `where` to a single word that contains only `a..z`, `0-9`, `.` and `_`, Bacula will create the specified database and restore data into it.

```
* restore where=bacula.old
```



```
127.0.0.1-fd    2013-05-21 09:50:41 MySQLDump I (T) 2    Change location  /@MYSQL/main/mysql/
```

Directories

```
Root
  /
    @MYSQL/
      main/
        mysql/
        weird database4962/
        logs/
    tmp/
```

Directory content

T...	File	Size	Date
	.	220 B	2013-05-21 09...
	..	220 B	2013-05-21 09...
	createdb.sql	1 B	2013-05-21 09...
	data.sql	1 B	2013-05-21 09...
	schema.sql	1 B	2013-05-21 09...
	grants.sql	1 B	2013-05-21 09...

Fig. 4: Database content during restore

If you set the `replace` parameter to `never`, Bacula will check the database list, and will abort the Job if the database currently restored already exists.

Using `replace=always` is not recommended.

If the `where` parameter is a directory (containing /), Bacula will restore all files into this directory. Doing so, you will be able to use `mysql` directly and do manual restores.

See also:

Go back to:

- *Restoring Users and Roles*

Go to:

- *Restoring Dump Files To Directory*
- *PITR Using Binary Logs*
- *Restoring Single Table*
- *Restoring Complete Server*
- *Restoring Using MySQL Command-Line Tool*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Restoring Dump Files To Directory

To restore SQL dumps to a directory, you set the `where` parameter to a valid directory.

```
* restore where=/tmp
```

See also:

Go back to:

- *Restoring Users and Roles*
- *Restoring Single Database*

Go to:

- *PITR Using Binary Logs*
- *Restoring Single Table*
- *Restoring Complete Server*
- *Restoring Using MySQL Command-Line Tool*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

PITR Using Binary Logs

Point-In-Time Recovery refers to recovery of data changes made up to a given point in time. Typically, this type of recovery is performed after restoring a full backup that brings the server to its state as of the time the backup was made.

To restore data from the binary log, aside from adding the `logs/` folder from the plugin file tree, you also must know the name and location of the current binary log files when the backup was made. This information is available in the “CHANGE MASTER” line on the top of the `data.sql` file.

```
-- Position to start replication or point-in-time recovery from

-- CHANGE MASTER TO MASTER_LOG_FILE='sql-bin.000004', MASTER_LOG_POS=2083;
```

This information is also printed in the Bacula job report when restoring a dump directly into a new database using `where=newdb` parameter.

```
...
Found MASTER_LOG position sql-bin.000004:2083 for "database5276"
...
```

Once you have this information and all log files generated between the Full backup and the point in time when you want to restore, you need to use the `mysqlbinlog` program.

```
# mysqlbinlog -j 2083 sql-bin.000004 sql-bin.000005...
```

This command will generate an SQL script that you can load into your restored database to run the *recover* process. You may want to stop the *recover* process in a middle of a log file, for that, `mysqlbinlog` provides several options such as `--stop-datetime` to control this behavior. Refer to the `mysqlbinlog` documentation for all parameters:

http://dev.mysql.com/doc/refman/5.1/en/mysqlbinlog.html.

As the output of `mysqlbinlog` program is an SQL script, you can also edit the script to fit your needs. For example, if the database has a new name, you will need to edit the SQL script to change database references.

```
# mysqlbinlog -j 2083 mysql-bin.000004 ... | \
   sed 's/use `orgname`/use `newname`/'    | \
   mysql -u root newname
```

For more information on PITR with MySQL, refer to the MySQL documentation:

https://dev.mysql.com/doc/refman/8.0/en/point-in-time-recovery.html

See also:

Go back to:

- *Restoring Users and Roles*
- *Restoring Single Database*
- *Restoring Dump Files To Directory*

Go to:

- *Restoring Single Table*
- *Restoring Complete Server*
- *Restoring Using MySQL Command-Line Tool*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Restoring Single Table

To restore a single item such as a table, you currently need to restore the dump file to a directory and use the `mysql` command.

```
$ sed -n -e '/Table structure for table .mytable.$/,/Table structure for table/p' data.
↪sql
```

The above `sed` command will extract the table structure, the index and the data from the dump.

See also:

Go back to:

- *Restoring Users and Roles*
- *Restoring Single Database*
- *Restoring Dump Files To Directory*
- *PITR Using Binary Logs*

Go to:

- *Restoring Complete Server*
- *Restoring Using MySQL Command-Line Tool*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Restoring Complete Server

To restore the all-databases and the server configuration, just select all files located in `/@MYSQL/<service>` except bin-log in `logs` directory, use `replace=always` and `where=/` options.

If you are using MySQL BinLog, you will need to apply bin logs after the restore using instruction described in *PITR Using Binary Logs*.

See also:

Go back to:

- *Restoring Users and Roles*
- *Restoring Single Database*
- *Restoring Dump Files To Directory*
- *PITR Using Binary Logs*
- *Restoring Single Table*

Go to:

- *Restoring Using MySQL Command-Line Tool*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Restoring Using MySQL Command-Line Tool

It is possible to use the MySQL Command-Line Tool to restore from the dump files generated by backup jobs using the MySQL plugin in Dump mode.

The MySQL plugin generate, in a Dump mode backup, the files related in Table *Dump Mode Options*.

The MySQL plugin will generate the following files entries in the Bacula catalog for a server having a single database "test":

```
/etc/mysql/my.cnf
@MYSQL/main/gobal-grants.sql
@MYSQL/main/settings.txt

@MYSQL/main/test/createdb.sql
@MYSQL/main/test/schema.sql
@MYSQL/main/test/data.sql
@MYSQL/main/test/grants.sql
```

To restore the single database "test", please follow the below sequence of commands.

At this time, the createdb.sql file is a dummy file used to create the database if selected in the restore process. We might enhance the plugin to add the SQL command used to create a database. So please manually create the database:

```
# mysql -u root
Welcome to the MySQL monitor. Commands end with ; or \g.
...
mysql> create database test;
```

(continues on next page)

(continued from previous page)

```
Query OK, 1 row affected (0.00 sec)

mysql> quit
Bye
```

Then restore schema, data and privileges:

```
# mysql -u root --database=test < /path/to/\@MYSQL/main/test/schema.sql
# mysql -u root --database=test < /path/to/\@MYSQL/main/test/data.sql
# mysql -u root --database=test < /path/to/\@MYSQL/main/test/grants.sql
```

When restoring the MySQL server or the “mysql” database, it may be needed to restore global privileges:

```
# mysql -u root < /path/to/\@MYSQL/main/global-grants.sql
```

The “settings.txt” file contains the current variables for the MySQL server. This file is not used automatically by the restore process. Its content can be used to restore the current MySQL server settings or to re-configure a MySQL server on a new system, for example.

See also:

Go back to:

- *Restoring Users and Roles*
- *Restoring Single Database*
- *Restoring Dump Files To Directory*
- *PITR Using Binary Logs*
- *Restoring Single Table*
- *Restoring Complete Server*

Go back to the *Restoring Using Dumps page*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

See also:

Go to *Restoring Complete Server Using Binary Mode (Percona)*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Go back to the *main MySQL Plugin page*.

Restoring Complete Server Using Binary Mode (Percona)

In binary mode, the Bacula MySQL Plugin uses the Percona xtrabackup tools. You must have the Percona tools installed.

You can find useful information in the Percona manual: http://www.percona.com/doc/percona-xtrabackup/?id=percona-xtrabackup:start

The details of the restore depend on whether or not you used the `prepare` option or not. If you did use the `prepare` option, please see the next section *Restoring with Automatic Extraction*, otherwise please see the section entitled *Restoring Binary Mode Backup without Prepare*.

Restoring with Automatic Extraction

Available since Bacula Enterprise 10.2.3

Bacula can extract xbstream automatically at restore.

To do so, either specify `restore_extract` in the Plugin line

```
Plugin = "mysql: restore_extract"
```

Or overwrite the behavior in bconsole by modifying the plugin Options.

```
Automatically selected Client: localhost-fd
Using Catalog "MyCatalog"
Run Restore job
...
Plugin Options:  *None*
OK to run? (yes/mod/no): mod
Parameters to modify:
     1: Level
     2: Storage
     3: Job
     4: FileSet
     5: Restore Client
     6: When
     7: Priority
     8: Bootstrap
     9: Where
    10: File Relocation
    11: Replace
    12: JobId
    13: Plugin Options
Select parameter to modify (1-13): 13
Automatically selected : mysql: mode=bin"
Plugin Restore Options
Option                Current Value         Default Value
manual_restore:       *None*                (yes)
extract_restored_xbstream: *None*                 (no)
Use above plugin configuration? (yes/mod/no): mod
You have the following choices:
```

(continues on next page)

(continued from previous page)

```
     1: manual_restore (Do not try to start the recover process)
     2: extract_restored_xbstream (When xbstream is restored, automatically extract it)
Select parameter to modify (1-2): 2
Please enter a value for extract_restored_xbstream: yes
Plugin Restore Options
Option                   Current Value        Default Value
manual_restore:          *None*               (yes)
extract_restored_xbstream: yes                      (no)
Use above plugin configuration? (yes/mod/no): yes
```

See also:

Go to:

- *Restoring Binary Mode Backup with Prepare*
- *Restoring Binary Mode Backup without Prepare*

Go back to the *Restoring Complete Server Using Binary Mode (Percona)*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Restoring Binary Mode Backup with Prepare

If the Prepare has already been done during the backup because you used the `prepare` option on the plugin, you will not need to manually do the Prepare. This can save considerable time.

First, you should use the Bacula `restore` and select a Full backup to be restored. Once the files are restored to a suitable directory, you will find something similar to the following directory structure:

```
@MYSQL/main/all-databases.xbstream
@MYSQL/main/my.cnf
@MYSQL/main/mysql.dat
```

Once you have restored the backup content with Bacula, if you didn't choose automatic extraction with `restore_extract` or `extract_restored_xbstream`, you can then restore the data using the -x option on xbstream (or `mbstream` for MariaDB version 10 or greater) as in the following example:

```
% cd @MYSQL/main
%ls
all-databases.xbstream my.cnf   mysql.dat
# Now extract the all-databases.xbstream
% xbstream -x < all-databases.xbstream
% ls
all-databases.xbstream mysql               xtrabackup_checkpoints
backup-my.cnf          performance_schema  xtrabackup_info
ib_buffer_pool         regress             xtrabackup_logfile
ibdata1                sys
my.cnf                 testdb
```

Now the files in the local directory are ready to be used by the server. The `--copy-back` option on `innobackupex` will copy the prepared data back to its original location as defined by the datadir in your `my.cnf`. Note that you can use `--defaults-file=/path/to/my.cnf` to specify the `my.cnf` configuration file.

However, before innobackupex will allow you to overwrite the original MySQL data files, you must either move them or remove them. For example, either:

```
% rm -rf /var/lib/mysql
```

or

```
% mv /var/lib/mysql /var/lib/mysql.old
```

If you are running on a server where the MySQL data files are kept in a different directory, you will need to adjust the above paths.

Now you can actually copy the files back with:

```
% innobackupex --copy-back $PWD
...
120604 02:58:44  innobackupex: completed OK!
```

For MariaDB version 10 or greater, use the `mariabackup` command with the parameter `--innobackupex` and the same arguments instead:

```
% mariabackup --innobackupex --copy-back $PWD
```

You should check the file permissions after copying the data back. You may need to adjust them with something like:

```
% chown -R mysql:mysql /var/lib/mysql
```

Now the datadir contains the restored data. You are ready to re-start the server, typically with something like:

```
% service mysql start
```

If the MySQL server will not start due to AppArmor denials, until you solve the problems, you can temporarily disable apparmor with the following command:

```
% apparmor_parser -R /etc/apparmor.d/usr.sbin.mysqld
```

The opposite of the above is the following:

```
% apparmor_parser -a /etc/apparmor.d/usr.sbin.mysqld
```

Note: In the above two examples, `mysqld` is spelled with a d on the end in contrast to the service name where the name is simply `mysql`.

Finally, if you want to completely remove your database and create a new empty one, the following commands may help:

```
% rm -rf /var/lib/mysql          # Remove any old database setup
% mysql_install_db -u mysql      # Install new database
% systemctl unmask mysql.service # Emables the service for systemd
% service mysql start            # start the service.
```

See also:

Go back to:

- *Restoring with Automatic Extraction*

Go to:

- *Restoring Binary Mode Backup without Prepare*

Go back to the *Restoring Complete Server Using Binary Mode (Percona)*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Restoring Binary Mode Backup without Prepare

If you have done your backup with the `prepare` keyword on the plugin directive you should go back to the previous section section as the restored backup prepare has already been done.

Once you have restored the backup content with Bacula, files using the `tar` format should be extracted with `tar -i` option. With xbstream format, if you didn't choose automatic extraction with `restore_extract` or `extract_restored_xbstream`, you can extract data with the `-x` option.

```
% cd @MYSQL/main
% xbstream -x < all-databases.xbstream
% ls
all-databases.xbstream  ibdata1.delta           performance_schema
xtrabackup_logfile      ibdata1.meta            testdb
backup-my.cnf           xtrabackup_checkpoints mysql
xtrabackup_binary       xtrabackup_binlog_info
```

When the files are uncompressed you can prepare the backup with the `--apply-log` option of the `innobackupex` tool. If you plan to apply incremental backups, you need also to use the `--redo-only` option. For MariaDB the `mariabackup` command must have the `--innobackupex` parameter so that it will mimic `innobackupex` below.

```
% innobackupex --apply-log --redo-only $PWD
...
120604 02:50:02  innobackupex: completed OK!
```

Each Incremental should be extracted in a specific directory, then they should be applied to the base data.

```
% mkdir incr1
% cd incr1
% xbstream -x < ../all-databases-1220202.xbstream
% cd ..
% innobackupex --apply-log --redo-only --incremental-dir=incr1 $PWD
...
120604 02:51:02  innobackupex: completed OK!

% mkdir incr2
% cd incr2
% xbstream -x < ../all-databases-1320402.xbstream
% cd ..
% innobackupex --apply-log --redo-only --incremental-dir=incr2 $PWD
...
120604 02:52:02  innobackupex: completed OK!
```

When the files are uncompressed you can prepare the backup with the `--apply-log` option of the `innobackupex` tool:

```
% innobackupex --apply-log $PWD
...
120604 02:51:02  innobackupex: completed OK!
```

Now the files in the local directory are ready to be used by the server. The `--copy-back` option will copy the prepared data back to its original location as defined by the datadir in your `my.cnf`. Note that you can use `--defaults-file=/path/to/my.cnf` to specify the `my.cnf` configuration file.

```
% innobackupex --copy-back $PWD
...
120604 02:58:44  innobackupex: completed OK!
```

You should check the file permissions after copying the data back. You may need to adjust them with something like:

```
% chown -R mysql:mysql /var/lib/mysql
```

Now the datadir contains the restored data. You are ready to start the server.

See also:

Go back to:

- *Restoring with Automatic Extraction*
- *Restoring Binary Mode Backup without Prepare*

Go back to the *Restoring Complete Server Using Binary Mode (Percona)*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

See also:

Go to *Restoring Using Dumps*.

Go back to the *Restore page*.

Go back to the main *Operations page*.

Go back to the *main MySQL Plugin page*.

See also:

Go to *Backup*.

Go back to the main *Operations page*.

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

See also:

Go back to:

- *Scope*
- *Features*
- *Backup Strategies*
- *Installation*
- *Configuration*

Go to:

- *Limitations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.

7 Limitations

- To backup multiple MySQL instances with the `binary` method and the xtrabackup tool, you must define multiple FileSet and multiple Job resources.
- The xtrabackup tool doesn't know how to read `.my.cnf` file to get user and password information. Thus you must specify the password in the plugin command line or to use the `extra_file` option in addition to the `mycnf_dir` parameter.
- The Percona `prepare` option may only be used with Full backups. If you attempt to use the `prepare` option with an Incremental or Differental Percona backup, the backup will continue without the `prepare` option.
- The Percona `prepare` option is incompatible with Bacula Encryption, Compression, and ACL options. If you use any of those options with the `prepare` option, the resulting backup will probably be unrestorable.

  Note: All Percona backups use the xbstream program to backup the data. The xbstream program automatically uses compression.

- The Percona `prepare` option only works with backups of a single MySQL instance. There may be multiple databases within that instance that are backed up.
- With Percona `prepare=fd` the sized of the all-databases.xbstream as shown in the Bacula Catalog will alwas be reported as -1. This is because the stream is created on the fly with no intermediary file.
- Most of the MD5 signatures for a Percona `prepare` will not be valid either because the file never existed on disk, or because the file was modified without recomputing the requested checksum.
- In the current Percona `prepare=sd` implementation the Storage daemon's `Working Directory` is used for placement of the temporary files. Consequently, it should be on a very fast device (RAID or SSD) and must be sufficiently large to handle the maximum database size for as many clients that can run simultaneously.
- The backup of the Percona `prepare=fd` may include a few left over files of the Prepare process that are not really needed for a proper backup.
- The Percona tools are tailor made for each operating system and for each version of MySQL. Therefore you must be very careful about upgrading either MySQL and/or the Percona tools. Please test carefully before trying to put them into production. Apparently with Ubuntu 16.04 the Percona tools that are part of their distribution were not upgraded when they added a newer version of MySQL. Consequently just doing an upgrade of that system can lead to Backup failures. In general, the Percona site has the most current versions you need for each MySQL version and also has a matrix of which versions work together.
- The testing of the Percona `prepare` features was done with `mysql Ver 14.14 Distrib 5.7.22` and `Percona version 2.4.11`.
- As noted above doing a restore of a MySQL database on a system that uses AppArmor (Debian based, e.g. Debian, Ubuntu, ...) can run into AppArmor permissions problems. Thus we strongly recommend that you try doing a Full restore of your MySQL installation in a test environment prior to putting it into production.
- If you have MySQL binary mode backups and you plan to upgrade your MySQL server operating system, please confirm that Percona xtrabackup tools are available for the platform you plan to upgrade the system. MySQL

binary mode backups use the Percona xtrabackup tools and they must be available for the new operating system version/platform of your MySQL server.

- 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.

See also:

Go back to:

- *Scope*
- *Features*
- *Backup Strategies*
- *Installation*
- *Configuration*
- *Operations*

Go back to the *main MySQL Plugin page*.

Go back to the Dedicated Backup Solutions page.