Percona XtraBackup Documentation

2.4.29 (December 18, 2023)

Percona Technical Documentation Team

Table of contents

1. Percona XtraBackup - Documentation

This documentation is for the latest release: Percona XtraBackup 2.4.29 (Release Notes).

Percona XtraBackup is an open-source hot backup utility for *MySQL* - based servers that does not lock your database during the backup. It can back up data from *InnoDB*, XtraDB, and *MyISAM* tables on *MySQL* 5.1 , 5.5, 5.6 and 5.7 servers, as well as Percona Server with XtraDB.

Note

Support for InnoDB 5.1 builtin has been removed in *Percona XtraBackup* 2.1

For a high-level overview of many of its advanced features, including a feature comparison, please see About Percona XtraBackup.

Whether it is a 24x7 highly loaded server or a low-transaction-volume environment, *Percona XtraBackup* is designed to make backups a seamless procedure without disrupting the performance of the server in a production environment. Commercial support contracts are available.

Important

Percona XtraBackup 2.4 does not support making backups of databases created in *MySQL 8.0*, *Percona Server for MySQL 8.0*, or *Percona XtraDB Cluster 8.0*. Use Percona XtraBackup 8.0 for the version 8.0 databases.

1.0.1 Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-01-25

2. Introduction

2.1 About Percona XtraBackup

Percona XtraBackup is the world's only open-source, free *MySQL* hot backup software that performs non-blocking backups for *InnoDB* and XtraDB databases. With *Percona XtraBackup*, you can achieve the following benefits:

- Backups that complete quickly and reliably
- Uninterrupted transaction processing during backups
- Savings on disk space and network bandwidth
- Automatic backup verification
- Higher uptime due to faster restore time

See the compatibility matrix in Percona Software and Platform Lifecycle to find out which versions of MySQL, MariaDB, and Percona Server for MySQL are supported by Percona XtraBackup and supports encryption with any kind of backups.

Non-blocking backups of InnoDB, Percona XtraDB Cluster, and *HailDB* storage engines are supported. In addition, Percona XtraBackup can back up the following storage engines by briefly pausing writes at the end of the backup: MyISAM, `Merge <.MRG>`, and `Archive <.ARM>`, including partitioned tables, triggers, and database options. InnoDB tables are still locked while copying non-InnoDB data. Fast incremental backups are supported for Percona Server with Percona XtraDB Cluster changed page tracking enabled.

> **Important**
>
> Percona XtraBackup 2.4 only supports Percona XtraDB Cluster 5.7. Percona XtraBackup 2.4 does not support the MyRocks storage engine or TokuDB storage engine. *Percona XtraBackup* is not compatible with MariaDB 10.3 and later.

Percona's enterprise-grade commercial MySQL Support contracts include support for *Percona XtraBackup*. We recommend support for critical production deployments.

2.1.1 What are the features of Percona XtraBackup?

Here is a short list of *Percona XtraBackup* features. See the documentation for more.

- Create hot InnoDB backups without pausing your database
- Make incremental backups of MySQL
- Stream compressed MySQL backups to another server
- Move tables between MySQL servers on-line
- Create new MySQL replication replicas easily
- Backup MySQL without adding load to the server
- Backup locks are a lightweight alternative to `FLUSH TABLES WITH READ LOCK` available in Percona Server 5.6+. Percona XtraBackup uses them automatically to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.
- Percona XtraBackup performs throttling based on the number of IO operations per second.
- Percona XtraBackup skips secondary index pages and recreates them when a compact backup is prepared.
- Percona XtraBackup can export individual tables even from a full backup, regardless of the InnoDB version.
- Tables exported with Percona XtraBackup can be imported into Percona Server 5.1, 5.5 or 5.6+, or MySQL 5.6+.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-11

2.2 How Percona XtraBackup Works

Percona XtraBackup is based on InnoDB's crash-recovery functionality. It copies your *InnoDB* data files, which results in data that is internally inconsistent; but then it performs crash recovery on the files to make them a consistent, usable database again.

This works because *InnoDB* maintains a redo log, also called the transaction log. This contains a record of every change to InnoDB data. When *InnoDB* starts, it inspects the data files and the transaction log, and performs two steps. It applies committed transaction log entries to the data files, and it performs an undo operation on any transactions that modified data but did not commit.

Percona XtraBackup works by remembering the log sequence number (LSN) when it starts, and then copying away the data files. It takes some time to do this, so if the files are changing, then they reflect the state of the database at different points in time. At the same time, *Percona XtraBackup* runs a background process that watches the transaction log files, and copies changes from it. *Percona XtraBackup* needs to do this continually because the transaction logs are written in a round-robin fashion, and can be reused. *Percona XtraBackup* needs the transaction log records for every change to the data files since it began execution.

Percona XtraBackup will use Backup locks where available as a lightweight alternative to `FLUSH TABLES WITH READ LOCK`. This feature is available in *Percona Server for MySQL* 5.6+. Percona XtraBackup uses this automatically to copy non-InnoDB data to avoid blocking DML queries that modify *InnoDB* tables. When backup locks are supported by the server, *xtrabackup* will first copy *InnoDB* data, run the `LOCK TABLES FOR BACKUP` and copy the *MyISAM* tables and `.frm` files. Once this is done, the backup of the files will begin. It will backup `.frm`, `.MRG`, `.MYD`, `.MYI`, `.TRG`, `.TRN`, `.ARM`, `.ARZ`, `.CSM`, `.CSV`, `.par`, and `.opt` files.

Note

Locking is done only for *MyISAM* and other non-InnoDB tables, and only **after** *Percona XtraBackup* is finished backing up all InnoDB/XtraDB data and logs. *Percona XtraBackup* will use Backup locks where available as a lightweight alternative to `FLUSH TABLES WITH READ LOCK`. This feature is available in *Percona Server for MySQL* 5.6+. Percona XtraBackup uses this automatically to copy non-InnoDB data to avoid blocking DML queries that modify *InnoDB* tables.

After that, *xtrabackup* will use `LOCK BINLOG FOR BACKUP` to block all operations that might change either the binary log position or `Exec_Master_Log_Pos` or `Exec_Gtid_Set` (i.e. source binary log coordinates corresponding to the current SQL thread state on a replication replica) as reported by `SHOW MASTER/SLAVE STATUS`. *xtrabackup* will then finish copying the REDO log files and fetch the binary log coordinates. After this is completed *xtrabackup* will unlock the binary log and tables.

Finally, the binary log position will be printed to `STDERR` and *xtrabackup* exits, returning 0 if all went OK.

Note that the `STDERR` of *xtrabackup* is not written in any file. You will have to redirect it to a file, e.g., `xtrabackup OPTIONS 2> backupout.log`.

It will also create the following files in the directory of the backup.

During the prepare phase, *Percona XtraBackup* performs crash recovery against the copied data files, using the copied transaction log file. After this is done, the database is ready to restore and use.

The backed-up *MyISAM* and *InnoDB* tables will be eventually consistent with each other, because after the prepare (recovery) process, *InnoDB*'s data is rolled forward to the point at which the backup completed, not rolled back to the point at which it started. This point in time matches where the `FLUSH TABLES WITH READ LOCK` was taken, so the *MyISAM* data and the prepared *InnoDB* data are in sync.

The *xtrabackup* and innobackupex tools both offer many features not mentioned in the preceding explanation. Each tool's functionality is explained in more detail further in the manual. In brief, though, the tools permit you to do operations such as streaming and incremental backups with various combinations of copying the data files, copying the log files, and applying the logs to the data.

2.2.1 Restoring a backup

To restore a backup with *xtrabackup* you can use the `xtrabackup --copy-back` or `xtrabackup --move-back` options.

xtrabackup will read from the `my.cnf` the variables `datadir`, `innodb_data_home_dir`, `innodb_data_file_path`, `innodb_log_group_home_dir` and check that the directories exist.

It will copy the *MyISAM* tables, indexes, etc. (`.frm`, `.MRG`, `.MYD`, `.MYI`, `.TRG`, `.TRN`, `.ARM`, `.ARZ`, `.CSM`, `.CSV`, `par` and .opt files) first, *InnoDB* tables and indexes next and the log files at last. It will preserve file's attributes when copying them, you may have to change the files' ownership to `mysql` before starting the database server, as they will be owned by the user who created the backup.

Alternatively, the `xtrabackup --move-back` option may be used to restore a backup. This option is similar to `xtrabackup --copy-back` with the only difference that instead of copying files it moves them to their target locations. As this option removes backup files, use it with caution. It is useful when there is not enough free disk space to hold both the data files and their backup copies.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

2.3 Understand version numbers

A version number identifies the product release. The product contains the latest Generally Available (GA) features at the time of that release.

2.4	**26**
Base version	Minor build version

Percona uses semantic version numbering, which follows the pattern of base version and minor build. Percona assigns unique, non-negative integers in increasing order for each minor build release. The version number combines the base Percona XtraBackup version number and the minor build version.

Percona does not release a new version of Percona XtraBackup 2.4 for every release of MySQL 5.7. Percona XtraBackup 2.4 is compatible with newer versions of MySQL 5.7.

The version numbers for Percona XtraBackup 2.4.26 define the following information:

- Base version - the leftmost number indicates the version of the Percona XtraBackup that is based on MySQL 5.7. An increase in base version resets the minor build version to 0.
- Minor build version - an internal number that denotes the version of the software. A build version increases by one each time the Percona XtraBackup is released.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-30

3. Installation

3.1 Installing Percona XtraBackup 2.4

Note

Install Percona XtraBackup 2.4 using the following methods. The instructions to install Percona XtraBackup 8.0 are available in the Percona XtraBackup 8.0 documentation.

We recommend that you install Percona XtraBackup 2.4 from the official Percona software repositories using the appropriate package manager for your system:

- Use an APT repo
- Use a YUM repo

3.1.1 Installation alternatives

Percona also provides the following methods:

- Download a binary tarball with the required files and binaries for a manual installation
- Compile from source
- Run in a Docker Container

Before installing Percona XtraBackup with any of these methods, we recommend that you review the release notes.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-12-06

3.2 Installing Percona XtraBackup on Debian and Ubuntu

Note

The following instructions install Percona XtraBackup 2.4 using the APT package manager. The instructions to install Percona XtraBackup 8.0 using the APT package manager are available in the Percona XtraBackup 8.0 installation documentation.

Ready-to-use packages are available from the *Percona XtraBackup* software repositories and the Percona download page.

Specific information on the supported platforms, products, and versions is described in Percona Software and Platform Lifecycle.

3.2.1 What's in each DEB package?

The `percona-xtrabackup-24` package contains the latest *Percona XtraBackup* GA binaries and associated files.

The `percona-xtrabackup-dbg-24` package contains the debug symbols for binaries in `percona-xtrabackup-24`.

The `percona-xtrabackup-test-24` package contains the test suite for *Percona XtraBackup*.

The `percona-xtrabackup` package contains the older version of the *Percona XtraBackup*.

3.2.2 Installing *Percona XtraBackup* via *percona-release*

Percona XtraBackup, like many other *Percona* products, is installed with the *percona-release* package configuration tool.

1. Download a deb package for *percona-release* the repository packages from Percona web:

   ```
   shell
   $ wget https://repo.percona.com/apt/percona-release_latest.$(lsb_release -sc)_all.deb
   ```

2. Install the downloaded package with `dpkg`. To do that, run the following commands as root or with `sudo`:

   ```
   shell
   $ sudo dpkg -i percona-release_latest.$(lsb_release -sc)_all.deb
   ```

   Once you install this package the Percona repositories should be added. You can check the repository setup in the /etc/apt/sources.list.d/percona-release.list file.

3. Enable the repository: `percona-release enable-only tools release`

   If *Percona XtraBackup* is intended to be used in combination with the upstream MySQL Server, you enable only the `tools` repository: `percona-release enable-only tools`.

4. After that you can install the `percona-xtrabackup-24` package:

   ```
   shell
   $ sudo apt install percona-xtrabackup-24
   ```

5. In order to make compressed backups, install the `qpress` package:

   ```
   shell
   $ sudo apt install qpress
   ```

Apt-Pinning the packages

In some cases you might need to “pin” the selected packages to avoid the upgrades from the distribution repositories. You’ll need to make a new file `/etc/apt/preferences.d/00percona.pref` and add the following lines in it:

```
Package: *
Pin: release o=Percona Development Team
Pin-Priority: 1001
```

For more information about the pinning you can check the official debian wiki.

3.2.3 Installing *Percona XtraBackup* using downloaded deb packages

Download the packages of the desired series for your architecture from the download page. Following example downloads the *Percona XtraBackup* 2.4.28 release package for *Debian* 10.0:

```
$ wget https://downloads.percona.com/downloads/Percona-XtraBackup-2.4/Percona-
XtraBackup-2.4.28/
\binary/debian/buster/x86_64/percona-xtrabackup-dbg-24_2.4.28-1.buster_amd64.deb
```

Now you can install *Percona XtraBackup* by running:

```
$ sudo dpkg -i percona-xtrabackup-24_2.4.28-1.buster_amd64.deb
```

Note

Installing the packages manually like this, you must resolve all dependencies and install the missing packages yourself.

3.2.4 Update the Curl utility in Debian 10

The default curl version, 7.64.0, in Debian 10 has known issues when attempting to reuse an already closed connection. This issue directly affects `xbcloud` and users may see intermittent backup failures.

For more details, see curl #3750 or curl #3763.

Follow these steps to upgrade curl to version 7.74.0:

1. Edit the `/etc/apt/sources.list` to add the following:

   ```
   text
   deb http://ftp.de.debian.org/debian buster-backports main
   ```

2. Refresh the `apt` sources:

   ```
   shell
   sudo apt update
   ```

3. Install the version from `buster-backports`:

   ```
   shell
   $ sudo apt install curl/buster-backports
   ```

4. Verify the version number:

   ```
   shell
   $ curl --version
   ```

   The result is similar to the following;

   ```
   text
   curl 7.74.0 (x86_64-pc-linux-gnu) libcurl/7.74.0
   ```

3.2.5 Uninstalling *Percona XtraBackup*

To uninstall *Percona XtraBackup* you'll need to remove all the installed packages.

1. Remove the packages

   ```
   shell
   $ sudo apt remove percona-xtrabackup-24
   ```

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-10-24

3.3 Installing Percona XtraBackup on Red Hat Enterprise Linux and CentOS

> **Note**
>
> The following instructions install Percona XtraBackup 2.4 using the YUM package manager. The instructions to install Percona XtraBackup 8.0 using the YUM package manager are available in the Percona XtraBackup 8.0 installation documentation.

Ready-to-use packages are available from the *Percona XtraBackup* software repositories and the download page. The Percona `yum` repository supports popular *RPM*-based operating systems, including the *Amazon Linux AMI*.

The easiest way to install the *Percona Yum* repository is to install an *RPM* that configures `yum` and installs the Percona GPG key.

Specific information on the supported platforms, products, and versions is described in Percona Software and Platform Lifecycle.

3.3.1 What's in each RPM package?

The `percona-xtrabackup-24` package contains the latest *Percona XtraBackup* GA binaries and associated files.

The `percona-xtrabackup-24-debuginfo` package contains the debug symbols for binaries in `percona-xtrabackup-24`.

The `percona-xtrabackup-test-24` package contains the test suite for *Percona XtraBackup*.

The `percona-xtrabackup` package contains the older version of the *Percona XtraBackup*.

3.3.2 Installing *Percona XtraBackup* from Percona `yum` repository

1. Install the `percona-release` configuration tool

   You can install the yum repository for percona-release by running the following command as a `root` user or with `sudo`:

   ```
   shell
   $ yum install https://repo.percona.com/yum/percona-release-latest.noarch.rpm
   ```

   You should see some output such as the following:

   ```
   text
   Retrieving https://repo.percona.com/yum/percona-release-latest.noarch.rpm
   Preparing...                ########################################### [100%]
      1:percona-release        ########################################### [100%]
   ```

   > **Note**
   >
   > *RHEL/Centos* 5 doesn't support installing the packages directly from the remote location so you'll need to download the package first and install it manually with rpm:

   ```
   shell
   $ wget https://repo.percona.com/yum/percona-release-latest.noarch.rpm
   $ rpm -ivH percona-release-latest.noarch.rpm
   ```

2. Testing the repository

   Make sure packages are now available from the repository, by executing the following command:

   ```
   shell
   $ yum list | grep percona
   ```

   You should see output similar to the following:

   ```
   text
   ...
   percona-xtrabackup-20.x86_64               2.0.8-587.rhel5          percona-release-x86_64
   percona-xtrabackup-20-debuginfo.x86_64     2.0.8-587.rhel5          percona-release-x86_64
   percona-xtrabackup-20-test.x86_64          2.0.8-587.rhel5          percona-release-x86_64
   percona-xtrabackup-21.x86_64               2.1.9-746.rhel5          percona-release-x86_64
   percona-xtrabackup-21-debuginfo.x86_64     2.1.9-746.rhel5          percona-release-x86_64
   percona-xtrabackup-22.x86_64               2.2.13-1.el5             percona-release-x86_64
   percona-xtrabackup-22-debuginfo.x86_64     2.2.13-1.el5             percona-release-x86_64
   percona-xtrabackup-debuginfo.x86_64        2.3.5-1.el5              percona-release-x86_64
   percona-xtrabackup-test.x86_64             2.3.5-1.el5              percona-release-x86_64
   percona-xtrabackup-test-21.x86_64          2.1.9-746.rhel5          percona-release-x86_64
   percona-xtrabackup-test-22.x86_64          2.2.13-1.el5             percona-release-x86_64
   ...
   ```

3. Enable the repository: `percona-release enable-only tools release`

   If *Percona XtraBackup* is intended to be used in combination with the upstream MySQL Server, you only need to enable the `tools` repository: `percona-release enable-only tools`.

4. Install *Percona XtraBackup* by running: `yum install percona-xtrabackup-24`

   Warning

   In order to successfully install *Percona XtraBackup* on CentOS prior to version 7, the `libev` package needs to be installed first. This package `libev` package can be installed from the EPEL repositories.

3.3.3 Percona `yum` Testing Repository

Percona offers pre-release builds from our testing repository. To subscribe to the testing repository, you'll need to enable the testing repository in /etc/yum.repos.d/percona-release.repo. To do so, set both `percona-testing-$basearch` and `percona-testing-noarch` to `enabled = 1` (Note that there are 3 sections in this file: release, testing and experimental - in this case it is the second section that requires updating).

Note

You'll need to install the Percona repository first (ref above) if this hasn't been done already.

1. To be able to make compressed backups, install the `qpress` package:

   ```
   shell
   $ yum install qpress
   ```

   See also

   Compressed Backup

3.3.4 Installing *Percona XtraBackup* using downloaded rpm packages

Download the packages of the desired series for your architecture from the download page. Following example will download *Percona XtraBackup* 2.4.28 release package for *CentOS* 9:

```
$ wget https://downloads.percona.com/downloads/Percona-XtraBackup-2.4/\
Percona-XtraBackup-2.4.28/binary/redhat/9/x86_64/percona-
xtrabackup-24-2.4.28-1.el9.x86_64.rpm
```

Now you can install *Percona XtraBackup* by running:

```
$ yum localinstall percona-xtrabackup-24-2.4.28-1.el9.x86_64.rpm
```

Note

When installing packages manually like this, you'll need to make sure to resolve all the dependencies and install missing packages yourself.

3.3.5 Uninstalling *Percona XtraBackup*

To completely uninstall *Percona XtraBackup* you'll need to remove all the installed packages.

Remove the packages

```
yum remove percona-xtrabackup
```

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-10-24

3.4 Installing Percona XtraBackup from a Binary Tarball

Note

The following instructions install Percona XtraBackup 2.4 by downloading and installing a binary tarball. The instructions to install Percona XtraBackup 8.0 by the same method are available in the Percona XtraBackup 8.0 installation documentation.

Percona provides binary tarballs of Percona XtraBackup. Binary tarballs contain pre-compiled executables, libraries, and other dependencies and are compressed `tar` archives. Extract the binary tarballs to any path.

Binary tarballs are available for download and installation. The following table lists the tarballs available in `Linux - Generic`. Select the *Percona XtraBackup* 2.4 version number and the type of tarball for your installation. Binary tarballs support all distributions.

After you have downloaded the binary tarballs, extract the tarball in the file location of your choice.

Type	Name	Description
Full	percona-xtrabackup--Linux.x86_64.glibc2.12.tar.gz	Contains binaries, libraries, test files, and debug symbols
Minimal	percona-xtrabackup--Linux.x86_64.glibc2.12-minimal.tar.gz	Contains binaries, and libraries but does not include test files, or debug symbols

Fetch and extract the correct binary tarball. For example, the following downloads the full tarball for version 2.4.24:

You may need to update the apt repositories and then install `wget`.

```
$ apt update
$ apt install wget
```

```
$ wget wget https://downloads.percona.com/downloads/Percona-XtraBackup-2.4/Percona-
XtraBackup-2.4.27/binary/tarball/percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12.tar.gz
```

Expected results

```
text
       --2022-12-06  17:01:29--      https://downloads.percona.com/downloads/Percona-XtraBackup-2.4/Percona-
XtraBackup-2.4.27/binary/tarball/percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12.tar.gz
  Resolving downloads.percona.com (downloads.percona.com)... 74.121.199.231, 162.220.4.221, 162.220.4.222
  Connecting to downloads.percona.com (downloads.percona.com)|74.121.199.231|:443... connected.
  HTTP request sent, awaiting response... 200 OK
  Length: 77216741 (74M) [application/x-gzip]
  Saving to: 'percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12.tar.gz'
```

Uncompress the file:

```
$ tar xvf percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12.tar.gz
```

Expected results

```
text
percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12/
percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12/man/
percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12/man/man1/
percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12/man/man1/xtrabackup.1
percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12/man/man1/xbstream.1
percona-xtrabackup-2.4.27-Linux-x86_64.glibc2.12/man/man1/xbcrypt.1
...
```

An installation from a binary tarball requires doing certain tasks manually, such as configuring settings, and policies, that a repository package installation performs automatically.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-12-06

3.5 Compiling and Installing from Source Code

Note

The following instructions installs Percona XtraBackup 2.4 by compiling the source code. The instructions to run Percona XtraBackup 8.0 by the same method are available in the Percona XtraBackup 8.0 installation documentation.

The source code is available from the *Percona XtraBackup Github* project. The easiest way to get the code is by using the `git clone` command. Then, switch to the release branch that you want to install, such as **2.4**.

```
$ git clone https://github.com/percona/percona-xtrabackup.git
$ cd percona-xtrabackup
$ git checkout 2.4
```

3.5.1 Step 1: Installing prerequisites

The following packages and tools must be installed to compile *Percona XtraBackup* from source. These might vary from system to system.

Important

In order to build *Percona XtraBackup* v8.0 from source, you need to use `cmake` version 3. In your distribution, it may be available either as a separate package `cmake3` or as `cmake`. To check which version is installed, run `cmake --version` and if it does report a version 3, install `cmake3` for your system.

See also

https://cmake.org/

Debian or Ubuntu using `apt`

```
$ sudo apt install build-essential flex bison automake autoconf \
libtool cmake libaio-dev mysql-client libncurses-dev zlib1g-dev \
libgcrypt11-dev libev-dev libcurl4-gnutls-dev vim-common
```

To install the man pages, install the `python3-sphinx` package:

```
$ sudo apt install python3-sphinx
```

CentOS or Red Hat using `yum`

Percona XtraBackup requires GCC version 5.3 or higher. If the version of GCC installed on your system is lower then you may need to install and enable the Developer Toolset on `RPM`-based distributions to make sure that you use the latest GCC compiler and development tools. Then, install `cmake` and other dependencies:

```
$ sudo yum install cmake openssl-devel libaio libaio-devel automake autoconf \
bison libtool ncurses-devel libgcrypt-devel libev-devel libcurl-devel zlib-devel \
vim-common
```

To install the man pages, install the `python3-sphinx` package:

```
$ sudo yum install python3-sphinx
```

3.5.2 Step 2: Generating the build pipeline

At this step, you have `cmake` run the commands in the `CMakeList.txt` file to generate the build pipeline, i.e. a native build environment that will be used to compile the source code).

1. Change to the directory where you cloned the Percona XtraBackup repository

   `shell`
   `$ cd percona-xtrabackup`

2. Create a directory to store the compiled files and then change to that directory:

   `shell`
   `$ mkdir build`
   `$ cd build`

3. Run `cmake` or `cmake3`. In either case, the options you need to use are the same.

Note

You can build *Percona XtraBackup* with man pages but this requires `python-sphinx` package which isn't available from that main repositories for every distribution. If you installed the `python-sphinx` package you need to remove the `-DWITH_MAN_PAGES=OFF` from previous command.

```
shell
$ cmake -DWITH_BOOST=PATH-TO-BOOST-LIBRARY -DDOWNLOAD_BOOST=ON \
-DBUILD_CONFIG=xtrabackup_release -DWITH_MAN_PAGES=OFF -B ..
```

More information about parameters

`-DWITH_BOOST`

For the `-DWITH_BOOST` parameter, specify the name of a directory to download the boost library to. This directory will be created automatically in your current directory.

`-B (--build)`

Percona XtraBackup is configured to forbid generating the build pipeline for `make` in the same directory where you store your sources. The `-B` parameter refers to the directory that contains the source code. In this example we use the relative path to the parent directory (..).

> **Important**
>
> CMake Error at CMakeLists.txt:367 (MESSAGE): Please do not build in-source. Out-of source builds are highly recommended: you can have multiple builds for the same source, and there is an easy way to do cleanup, simply remove the build directory (note that 'make clean' or 'make distclean' does *not* work)
>
> You *can* force in-source build by invoking cmake with -DFORCE_INSOURCE_BUILD=1

`-DWITH_MAN_PAGES`

To build *Percona XtraBackup* man pages, use `ON` or remove this parameter from the command line (it is `ON` by default).

To install the man pages, install the `python3-sphinx` package:

3.5.3 Step 2: Compiling the source code

To compile the source code in your build directory, use the `make` command.

> **Important**
>
> The computer where you intend to compile *Percona XtraBackup* 8.0 must have at least 2G of RAM available.

1. Change to the `build` directory (created at Step 2: Generating the build pipeline).
2. Run the `make` command. This command may take a long time to complete.

   ```
   shell
   $ make
   ```

3.5.4 Step 3: Installing on the target system

The following command installs all *Percona XtraBackup* binaries *xtrabackup* and tests to default location on the target system: `/usr/local/xtrabackup`.

Run `make install` to install *Percona XtraBackup* to the default location.

```
$ sudo make install
```

Installing to a non-default location

You may use the `DESTDIR` parameter with `make install` to install *Percona XtraBackup* to another location. Make sure that the effective user is able to write to the destination you choose.

```
$ sudo make DESTDIR=<DIR_NAME> install
```

In fact, the destination directory is determined by the installation layout (`-DINSTALL_LAYOUT`) that `cmake` applies (see Step 2: Generating the build pipeline). In addition to the installation directory, this parameter controls a number of other destinations that you can adjust for your system.

By default, this parameter is set to `STANDALONE`, which implies the installation directory to be /usr/local/xtrabackup.

See also

MySQL Documentation: -DINSTALL_LAYOUT

3.5.5 Step 4: Running

After *Percona XtraBackup* is installed on your system, you may run it by using the full path to the `xtrabackup` command:

```
$ /usr/local/xtrabackup/bin/xtrabackup
```

Update your PATH environment variable if you would like to use the command on the command line directly.

```
$# Setting $PATH on the command line
$ PATH=$PATH:/usr/local/xtrabackup/bin/xtrabackup

$# Run xtrabackup directly
$ xtrabackup
```

Alternatively, you may consider placing a soft link (using `ln -s`) to one of the locations listed in your `PATH` environment variable.

See also

`man ln`

To view the documentation with `man`, update the `MANPATH` variable.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-12-06

4. Run in Docker

4.1 Running Percona XtraBackup in a Docker container

Note

The following instructions runs Percona XtraBackup 2.4 in a Docker container. The instructions to run Percona XtraBackup 8.0 by the same method are available in the Percona XtraBackup 8.0 installation documentation.

Docker allows you to run applications in a lightweight unit called a container.

You can run *Percona XtraBackup* in a Docker container without installing the product. All required libraries are available in the container. Being a lightweight execution environment, Docker containers enable creating configurations where each program runs in a separate container. You may run *Percona Server for MySQL* in one container and *Percona XtraBackup* in another. Docker images offer a range of options.

Create a Docker container based on a Docker image. Docker images for Percona XtraBackup are hosted publicly on Docker Hub at `percona/percona-xtrabackup`.

```
$ sudo docker create ... percona/percona-xtrabackup --name xtrabackup ...
```

Scope of this section

This section demonstrates how to backup data on a Percona Server for MySQL running in another Docker container.

4.1.1 Installing Docker

Your operating system may already provide a package for **docker**. However, the versions of Docker provided by your operating system are likely to be outdated.

Use the installation instructions for your operating system available from the Docker site to set up the latest version of **docker**.

Note

Docker Documentation: * How to use Docker * Installing * Getting started

4.1.2 Connecting to a Percona Server for MySQL container

Percona XtraBackup works in combination with a database server. When running a Docker container for Percona XtraBackup, you can make backups for a database server either installed on the host machine or running in a separate Docker container.

To set up a database server on a host machine or in Docker container, follow the documentation of the supported product that you intend to use with Percona XtraBackup.

See also

Percona Server for MySQL Documentation: * Installing on a host machine * Running in a Docker container

```
$ sudo docker run -d --name percona-server-mysql-5.7 \
-e MYSQL_ROOT_PASSWORD=root percona/percona-server:5.7
```

As soon as Percona Server for MySQL runs, add some data to it. Now, you are ready to make backups with Percona XtraBackup.

4.1.3 Creating a Docker container from Percona XtraBackup image

You can create a Docker container based on Percona XtraBackup image with either **docker create** or **docker run** command. **docker create** creates a Docker container and makes it available for starting later.

Docker downloads the Percona XtraBackup image from the Docker Hub. If it is not the first time you use the selected image, Docker uses the image available locally.

```
$ sudo docker create --name percona-xtrabackup-2.4 --volumes-from percona-server-mysql-5.7 \
percona/percona-xtrabackup:2.4  \
xtrabackup --backup --datadir=/var/lib/mysql/ --target-dir=/backup \
--user=root --password=mysql
```

With `--name` you give a meaningful name to your new Docker container so that you could easily locate it among your other containers.

The `--volumes-from` referring to *percona-server-mysql* indicates that you indend to use the same data as the *percona-server-mysql* container.

Run the container with exactly the same parameters that were used when the container was created:

```
$ sudo docker start -ai percona-xtrabackup-2.4
```

This command starts the *percona-xtrabackup* container, attaches to its input/output streams, and opens an interactive shell.

The **docker run** is a shortcut command that creates a Docker container and then immediately runs it.

```
$ sudo docker run --name percona-xtrabackup-2.4 --volumes-from percona-server-mysql-5.7 \
percona/percona-xtrabackup:2.4
xtrabackup --backup --data-dir=/var/lib/mysql --target-dir=/backup --user=root --
password=mysql
```

See also

More in Docker documentation * Docker volumes as persistent data storage for containers * More information about containers

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-29

5. Prerequisites

5.1 Connection and Privileges Needed

Percona XtraBackup needs to be able to connect to the database server and perform operations on the server and the datadir when creating a backup, when preparing in some scenarios and when restoring it. In order to do so, there are privileges and permission requirements on its execution that must be fulfilled.

Privileges refers to the operations that a system user is permitted to do in the database server. **They are set at the database server and only apply to users in the database server**.

Permissions are those which permits a user to perform operations on the system, like reading, writing or executing on a certain directory or start/stop a system service. **They are set at a system level and only apply to system users**.

Whether *xtrabackup* or innobackupex is used, there are two actors involved: the user invoking the program - *a system user* - and the user performing action in the database server - *a database user*. Note that these are different users in different places, even though they may have the same username.

All the invocations of innobackupex and *xtrabackup* in this documentation assume that the system user has the appropriate permissions and you are providing the relevant options for connecting the database server - besides the options for the action to be performed - and the database user has adequate privileges.

5.1.1 Connecting to the server

The database user used to connect to the server and its password are specified by the `xtrabackup --user` and `xtrabackup –password` option:

```
$ xtrabackup --user=DVADER --password=14MY0URF4TH3R --backup \
  --target-dir=/data/bkps/
$ innobackupex --user=DBUSER --password=SECRET /path/to/backup/dir/
$ innobackupex --user=LUKE  --password=US3TH3F0RC3 --stream=tar ./ | bzip2 -
```

If you don't use the `xtrabackup --user` option, *Percona XtraBackup* will assume the database user whose name is the system user executing it.

Other Connection Options

According to your system, you may need to specify one or more of the following options to connect to the server:

Option	Description
--port	The port to use when connecting to the database server with TCP/IP.
--socket	The socket to use when connecting to the local database.
--host	The host to use when connecting to the database server with TCP/IP.

These options are passed to the mysql child process without alteration, see `mysql --help` for details.

Note

In case of multiple server instances the correct connection parameters (port, socket, host) must be specified in order for *xtrabackup* to talk to the correct server.

5.1.2 Permissions and Privileges Needed

Once connected to the server, in order to perform a backup you will need `READ` and `EXECUTE` permissions at a filesystem level in the server's datadir.

The database user needs the following privileges on the tables/databases to be backed up:

- `RELOAD` and `LOCK TABLES` (unless the –no-lock option is specified) in order to `FLUSH TABLES WITH READ LOCK` and `FLUSH ENGINE LOGS` prior to start copying the files, and `LOCK TABLES FOR BACKUP` and `LOCK BINLOG FOR BACKUP` require this privilege when Backup Locks are used.
- `REPLICATION CLIENT` in order to obtain the binary log position.
- `CREATE TABLESPACE` in order to import tables (see Restoring Individual Tables).
- `PROCESS` in order to run `SHOW ENGINE INNODB STATUS` (which is mandatory), and optionally to see all threads which are running on the server (see Improved FLUSH TABLES WITH READ LOCK handling).
- `SUPER` in order to start/stop the replica threads in a replication environment, use XtraDB Changed Page Tracking for Incremental Backups and for Improved FLUSH TABLES WITH READ LOCK handling.
- `CREATE` privilege in order to create the PERCONA_SCHEMA.xtrabackup_history database and table.
- `ALTER` privilege in order to upgrade the PERCONA_SCHEMA.xtrabackup_history database and table.
- `INSERT` privilege in order to add history records to the PERCONA_SCHEMA.xtrabackup_history table.
- `SELECT` privilege in order to use `innobackupex --incremental-history-name` or `innobackupex --incremental-history-uuid` in order for the feature to look up the `innodb_to_lsn` values in the PERCONA_SCHEMA.xtrabackup_history table.

The explanation of when these are used can be found in How Percona XtraBackup Works.

An SQL example of creating a database user with the minimum privileges required to full backups would be:

```
mysql> CREATE USER 'bkpuser'@'localhost' IDENTIFIED BY 's3cret';
mysql> GRANT RELOAD, LOCK TABLES, PROCESS, REPLICATION CLIENT ON *.* TO
       'bkpuser'@'localhost';
mysql> FLUSH PRIVILEGES;
```

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

5.2 Configuring xtrabackup

All of the *xtrabackup* configuration is done through options, which behave exactly like standard *MySQL* program options: they can be specified either at the command-line, or through a file such as `/etc/my.cnf`.

The *xtrabackup* binary reads the `[mysqld]` and `[xtrabackup]` sections from any configuration files, in that order. That is so that it can read its options from your existing *MySQL* installation, such as the datadir or some of the *InnoDB* options. If you want to override these, just specify them in the `[xtrabackup]` section, and because it is read later, it will take precedence.

You don't need to put any configuration in your `my.cnf` if you don't want to. You can simply specify the options on the command-line. Normally, the only thing you might find convenient to place in the `[xtrabackup]` section of your `my.cnf` file is the `target_dir` option to default the directory in which the backups will be placed, for example:

```
[xtrabackup]
target_dir = /data/backups/mysql/
```

This manual will assume that you do not have any file-based configuration for *xtrabackup*, so it will always show command-line options being used explicitly. Please see the option and variable reference for details on all of the configuration options.

The *xtrabackup* binary does not accept exactly the same syntax in the `my.cnf` file as the `mysqld` server binary does. For historical reasons, the `mysqld` server binary accepts parameters with a `--set-variable=<variable>=<value>` syntax, which *xtrabackup* does not understand. If your my.cnf file has such configuration directives, you should rewrite them in the `--variable=value` syntax.

5.2.1 System Configuration and NFS Volumes

The *xtrabackup* tool requires no special configuration on most systems. However, the storage where the `xtrabackup --target-dir` is located must behave properly when `fsync()` is called. In particular, we have noticed that NFS volumes not mounted with the `sync` option might not really sync the data. As a result, if you back up to an NFS volume mounted with the async option, and then try to prepare the backup from a different server that also mounts that volume, the data might appear to be corrupt. You can use the `sync` mount option to avoid this problem.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

6. Backup Scenarios

6.1 The Backup Cycle - Full Backups

6.1.1 Creating a backup

To create a backup, run `xtrabackup` with the `xtrabackup --backup option`. You also need to specify a `xtrabackup --target-dir` option, which is where the backup will be stored, if the *InnoDB* data or log files aren't stored in the same directory, you might need to specify the location of those, too. If the target directory does not exist, xtrabackup creates it. If the directory does exist and is empty, xtrabackup will succeed. xtrabackup will not overwrite existing files, it will fail with operating system error 17, `file exists`.

To start the backup process run:

```
$ xtrabackup --backup --target-dir=/data/backups/
```

This will store the backup at `/data/backups/`. If you specify a relative path, the target directory will be relative to the current directory.

During the backup process, you should see a lot of output showing the data files being copied, as well as the log file thread repeatedly scanning the log files and copying from it. Here is an example that shows the log thread scanning the log in the background, and a file copying thread working on the ibdata1 file:

```
160906 10:19:17 Finished backing up non-InnoDB tables and files
160906 10:19:17 Executing FLUSH NO_WRITE_TO_BINLOG ENGINE LOGS...
xtrabackup: The latest check point (for incremental): '62988944'
xtrabackup: Stopping log copying thread.
.160906 10:19:18 >> log scanned up to (137343534)
160906 10:19:18 Executing UNLOCK TABLES
160906 10:19:18 All tables unlocked
160906 10:19:18 Backup created in directory '/data/backups/'
160906 10:19:18 [00] Writing backup-my.cnf
160906 10:19:18 [00]        ...done
160906 10:19:18 [00] Writing xtrabackup_info
160906 10:19:18 [00]        ...done
xtrabackup: Transaction log of lsn (26970807) to (137343534) was copied.
160906 10:19:18 completed OK!
```

The last thing you should see is something like the following, where the value of the `<LSN>` will be a number that depends on your system:

```
xtrabackup: Transaction log of lsn (<SLN>) to (<LSN>) was copied.
```

Note

Log copying thread checks the transactional log every second to see if there were any new log records written that need to be copied, but there is a chance that the log copying thread might not be able to keep up with the amount of writes that go to the transactional logs, and will hit an error when the log records are overwritten before they could be read.

After the backup is finished, the target directory will contain files such as the following, assuming you have a single InnoDB table `test.tbl1` and you are using MySQL's `innodb_file_per_table` option:

```
$ ls -lh /data/backups/
```

The results are as follows:

```
total 182M
drwx------  7 root root 4.0K Sep  6 10:19 .
drwxrwxrwt 11 root root 4.0K Sep  6 11:05 ..
-rw-r-----  1 root root  387 Sep  6 10:19 backup-my.cnf
-rw-r-----  1 root root  76M Sep  6 10:19 ibdata1
drwx------  2 root root 4.0K Sep  6 10:19 mysql
drwx------  2 root root 4.0K Sep  6 10:19 performance_schema
drwx------  2 root root 4.0K Sep  6 10:19 sbtest
drwx------  2 root root 4.0K Sep  6 10:19 test
drwx------  2 root root 4.0K Sep  6 10:19 world2
-rw-r-----  1 root root  116 Sep  6 10:19 xtrabackup_checkpoints
-rw-r-----  1 root root  433 Sep  6 10:19 xtrabackup_info
-rw-r-----  1 root root 106M Sep  6 10:19 xtrabackup_logfile
```

The backup can take a long time, depending on how large the database is. It is safe to cancel at any time, because it does not modify the database.

The next step is getting your backup ready to be restored.

6.1.2 Preparing a backup

After you made a backup with the `xtrabackup --backup` option, you'll first need to prepare it in order to restore it. Data files are not point-in-time consistent until they've been prepared, because they were copied at different times as the program ran, and they might have been changed while this was happening. If you try to start InnoDB with these data files, it will detect corruption and crash itself to prevent you from running on damaged data. The `xtrabackup --prepare` step makes the files perfectly consistent at a single instant in time, so you can run *InnoDB* on them.

You can run the prepare operation on any machine; it does not need to be on the originating server or the server to which you intend to restore. You can copy the backup to a utility server and prepare it there.

Note

You can prepare a backup created with older *Percona XtraBackup* version with a newer one, but not vice versa. Preparing a backup on an unsupported server version should be done with the latest *Percona XtraBackup* release which supports that server version. For example, if one has a backup of MySQL 5.0 created with *Percona XtraBackup* 1.6, then preparing the backup with *Percona XtraBackup* 2.3 is not supported, because support for MySQL 5.0 was removed in *Percona XtraBackup* 2.1. Instead, the latest release in the 2.0 series should be used.

During the `prepare` operation, xtrabackup boots up a kind of modified InnoDB that's embedded inside it (the libraries it was linked against). The modifications are necessary to disable InnoDB's standard safety checks, such as complaining that the log file isn't the right size, which aren't appropriate for working with backups. These modifications are only for the xtrabackup binary; you don't need a modified *InnoDB* to use xtrabackup for your backups.

The prepare step uses this *embedded InnoDB* to perform crash recovery on the copied data files, using the copied log file. The `prepare` step is very simple to use: you simply run `xtrabackup --prepare` option and tell it which directory to prepare, for example, to prepare the previously taken backup run:

```
$ xtrabackup --prepare --target-dir=/data/backups/
```

When this finishes, you should see an `InnoDB shutdown` with a message such as the following, where again the value of `LSN` will depend on your system:

```
InnoDB: Shutdown completed; log sequence number 137345046
160906 11:21:01 completed OK!
```

All following prepares will not change the already prepared data files, you'll see that output says:

```
xtrabackup: This target seems to be already prepared.
xtrabackup: notice: xtrabackup_logfile was already used to '--prepare'.
```

It is not recommended to interrupt xtrabackup process while preparing backup because it may cause data files corruption and backup will become unusable. Backup validity is not guaranteed if prepare process was interrupted.

Note

If you intend the backup to be the basis for further incremental backups, you should use the `xtrabackup --apply-log-only` option when preparing the backup, or you will not be able to apply incremental backups to it. See the documentation on preparing [incremental backup] (incremental_backup.md#incremental-backup) for more details.

6.1.3 Restoring a Backup

Warning

Backup needs to be prepared before it can be restored.

```
$ xtrabackup --copy-back --target-dir=/data/backups/
```

If you don't want to save your backup, you can use the `xtrabackup --move-back` option which will move the backed up data to the `datadir`.

If you don't want to use any of the above options, you can additionally use `rsync` or `cp` to restore the files.

Note

The `datadir` must be empty before restoring the backup. Also it's important to note that MySQL server needs to be shut down before restore is performed. You can't restore to a `datadir` of a running mysqld instance (except when importing a partial backup).

Example of the `rsync` command that can be used to restore the backup can look like this:

```
$ rsync -avrP /data/backup/ /var/lib/mysql/
```

You should check that the restored files have the correct ownership and permissions.

As files' attributes will be preserved, in most cases you will need to change the files' ownership to `mysql` before starting the database server, as they will be owned by the user who created the backup:

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

Data is now restored and you can start the server.

Note

When `relay-log-info-repository=TABLE` is enabled, the instance recovered from the backup has errors in the error log, like the following:

```
text
2019-08-09 12:40:02 69297 [ERROR] Failed to open the relay log '/data/mysql-relay-bin.004349' (relay_log_pos 
5534092)
```

To avoid these types of issues, enable `relay_log_recovery` or execute `RESET SLAVE` prior to `CHANGE MASTER TO`.

The relay log information was backed up, but a new relay log has been created, which creates a mismatch during the restore.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

6.2 Incremental Backup

Both the xtrabackup tool and the innobackupex tool support incremental backups. An incremental backup backs up only data that has changed since the last backup.

You can take multiple incremental backups between each full backup. For example, you can take a full backup once a week and an incremental backup every day, or a full backup every day and incremental backups each hour.

Incremental backups work because each *InnoDB* page contains a log sequence number (LSN). The `LSN` is the system version number for the entire database. Each page's `LSN` shows how recently it was changed.

An incremental backup copies each page whose `LSN` is newer than the previous incremental or full backup's `LSN`. There are two algorithms in use to find the set of such pages to be copied. The first one, available with all the server types and versions, checks the page `LSN` directly by reading all the data pages. The second one, available with *Percona Server for MySQL*, enables the changed page tracking feature on the server, which will note the pages as they are being changed. This information will be then written out in a compact separate so-called bitmap file. The xtrabackup binary uses that file to read only the data pages it needs for the incremental backup. This features potentially saves many read requests. The latter algorithm is enabled by default if the xtrabackup binary finds the bitmap file. It is possible to specify `xtrabackup --incremental-force-scan` to read all the pages even if the bitmap data is available.

> **Important**
>
> Incremental backups do not compare the data files to the previous backup's data files. For this reason, running an incremental backup after a *partial backup* may lead to inconsistent data.
>
> Incremental backups read the pages and compare their `LSN` to the last backup's `LSN`. You must have a full backup to recover the incremental changes. Without a full backup to act as a base, the incremental backups are useless.
>
> You can use the `--incremental-lsn` option to perform an incremental backup without even having the previous backup, if you know its `LSN`.
>
> **See also**: Partial Backups

6.2.1 Creating an Incremental Backup

To make an incremental backup, begin with a full backup as usual. The xtrabackup binary writes a file called `xtrabackup_checkpoints` into the backup's target directory. This file contains a line showing the `to_lsn`, which is the database's `LSN` at the end of the backup. Create the full backup with a following command:

```
$ xtrabackup --backup --target-dir=/data/backups/base
```

If you look at the xtrabackup_checkpoints file, you should see similar content depending on your LSN nuber:

```
backup_type = full-backuped
from_lsn = 0
to_lsn = 1626007
last_lsn = 1626007
compact = 0
recover_binlog_info = 1
```

Now that you have a full backup, you can make an incremental backup based on it. Use the following command:

```
$ xtrabackup --backup --target-dir=/data/backups/inc1 \
--incremental-basedir=/data/backups/base
```

The /data/backups/inc1/ directory should now contain delta files, such as ibdata1.delta and test/table1.ibd.delta. These represent the changes since the `LSN 1626007`. If you examine the `xtrabackup_checkpoints` file in this directory, you should see similar content to the following:

```
backup_type = incremental
from_lsn = 1626007
to_lsn = 4124244
last_lsn = 4124244
compact = 0
recover_binlog_info = 1
```

`from_lsn` is the starting LSN of the backup and for incremental it has to be the same as `to_lsn` (if it is the last checkpoint) of the previous/base backup.

It's now possible to use this directory as the base for yet another incremental backup:

```
$ xtrabackup --backup --target-dir=/data/backups/inc2 \
--incremental-basedir=/data/backups/inc1
```

This folder also contains the `xtrabackup_checkpoints`:

```
backup_type = incremental
from_lsn = 4124244
to_lsn = 6938371
last_lsn = 7110572
compact = 0
recover_binlog_info = 1
```

Note

In this case you can see that there is a difference between the `to_lsn` (last checkpoint LSN) and `last_lsn` (last copied LSN), this means that there was some traffic on the server during the backup process.

6.2.2 Preparing the Incremental Backups

The `xtrabackup --prepare` step for incremental backups is not the same as for full backups. In full backups, two types of operations are performed to make the database consistent: committed transactions are replayed from the log file against the data files, and uncommitted transactions are rolled back. You must skip the rollback of uncommitted transactions when preparing an incremental backup, because transactions that were uncommitted at the time of your backup may be in progress, and it's likely that they will be committed in the next incremental backup. You should use the xtrabackup `--apply-log-only` option to prevent the rollback phase.

Note

If you do not use the `xtrabackup --apply-log-only` option to prevent the rollback phase, then your incremental backup is useless.After the transactions have been rolled back, further incremental backups cannot be applied.

Beginning with the full backup you created, you can prepare it, and then apply the incremental differences to it. Recall that you have the following backups:

```
/data/backups/base
/data/backups/inc1
/data/backups/inc2
```

To prepare the base backup, you need to run xtrabackup –prepare as usual, but prevent the rollback phase:

```
$ xtrabackup --prepare --apply-log-only --target-dir=/data/backups/base
```

The output should end with text similar to the following:

```
InnoDB: Shutdown completed; log sequence number 1626007
161011 12:41:04 completed OK!
```

The log sequence number should match the `to_lsn` of the base backup, which you saw previously.

Note

This backup is safe to restore, even though the operation skipped the rollback phase. If you restore it and start *MySQL*, *InnoDB* detects that the rollback phase was not performed, and it will do that in the background. This operation is the same as a crash recovery upon start. In addition, MySQL notifies you that the database was not shut down normally.

To apply the first incremental backup to the full backup, run the following command:

```
$ xtrabackup --prepare --apply-log-only --target-dir=/data/backups/base \
--incremental-dir=/data/backups/inc1
```

This applies the delta files to the files in `/data/backups/base`, which rolls them forward in time to the time of the incremental backup. It then applies the redo log as usual to the result. The final data is in `/data/backups/base`, not in the incremental directory. You should see an output similar to:

```
incremental backup from 1626007 is enabled.
xtrabackup: cd to /data/backups/base
xtrabackup: This target seems to be already prepared with --apply-log-only.
xtrabackup: xtrabackup_logfile detected: size=2097152, start_lsn=(4124244)
...
xtrabackup: page size for /tmp/backups/inc1/ibdata1.delta is 16384 bytes
Applying /tmp/backups/inc1/ibdata1.delta to ./ibdata1...
...
161011 12:45:56 completed OK!
```

Again, the LSN should match what you saw from your earlier inspection of the first incremental backup. If you restore the files from `/data/backups/base`, you should see the state of the database as of the first incremental backup.

Warning

Percona XtraBackup does not support using the same incremental backup directory to prepare two copies of backup. Do not run `xtrabackup --prepare` with the same incremental backup directory (the value of `--incremental-dir`) more than once.

Preparing the second incremental backup is a similar process: apply the deltas to the (modified) base backup, and you will roll its data forward in time to the point of the second incremental backup:

```
$ xtrabackup --prepare --target-dir=/data/backups/base \
--incremental-dir=/data/backups/inc2
```

Note

`xtrabackup --apply-log-only` should be used when merging all incrementals except the last one. That's why the previous line doesn't contain the xtrabackup `--apply-log-only` option. Even if the `xtrabackup --apply-log-only` was used on the last step, backup would still be consistent but in that case server would perform the rollback phase.

Once prepared, incremental backups are the same as the full backups and they can be restored in the same way.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

6.3 Compressed Backup

Percona XtraBackup supports compressed backups: a local or streaming backup can be compressed or decompressed with xbstream.

6.3.1 Creating Compressed Backups

In order to make a compressed backup you'll need to use the xtrabackup –compress option:

```
$ xtrabackup --backup --compress --target-dir=/data/compressed/
```

The `xtrabackup --compress` uses the `qpress` tool that you can install via the `percona-release` package configuration tool as follows:

```
$ sudo percona-release enable tools
$ sudo apt update
$ sudo apt install qpress
```

Note

Enable the repository: `percona-release enable-only tools release`. If you intend to use Percona XtraBackup in combination with the upstream MySQL Server, you only need to enable the `tools` repository: `percona-release enable-only tools`.

If you want to speed up the compression you can use the parallel compression, which can be enabled with xtrabackup –compress-threads option. Following example will use four threads for compression:

```
$ xtrabackup --backup --compress --compress-threads=4 \
--target-dir=/data/compressed/
```

Output should look like this

```
...
170223 13:00:38 [01] Compressing ./test/sbtest1.frm to /tmp/compressed/test/sbtest1.frm.qp
170223 13:00:38 [01]        ...done
170223 13:00:38 [01] Compressing ./test/sbtest2.frm to /tmp/compressed/test/sbtest2.frm.qp
170223 13:00:38 [01]        ...done
...
170223 13:00:39 [00] Compressing xtrabackup_info
170223 13:00:39 [00]        ...done
xtrabackup: Transaction log of lsn (9291934) to (9291934) was copied.
170223 13:00:39 completed OK!
```

Preparing the backup

Before you can prepare the backup you must uncompress all the files. *Percona XtraBackup* has implemented `xtrabackup --decompress` option that can be used to decompress the backup.

```
$ xtrabackup --decompress --target-dir=/data/compressed/
```

Note

`xtrabackup --parallel` can be used with `xtrabackup --decompress` option to decompress multiple files simultaneously.

Percona XtraBackup does not automatically remove the compressed files. In order to clean up the backup directory, use the `xtrabackup --remove-original` option. If the files not removed they are not copied or moved to the datadir if `xtrabackup --copy-back` or `xtrabackup --move-back` are used.

When the files are uncompressed you can prepare the backup with the `xtrabackup --prepare` option:

```
$ xtrabackup --prepare --target-dir=/data/compressed/
```

Check for a confirmation message:

```
InnoDB: Starting shutdown...
InnoDB: Shutdown completed; log sequence number 9293846
170223 13:39:31 completed OK!
```

Now the files in `/data/compressed/` are ready to be used by the server.

Restoring the backup

xtrabackup has a `xtrabackup --copy-back` option, which performs the restoration of a backup to the server's datadir:

```
$ xtrabackup --copy-back --target-dir=/data/backups/
```

The option copies all the data-related files back to the server's `datadir`, determined by the server's `my.cnf` configuration file. Check the last line of the output for a success message:

```
170223 13:49:13 completed OK!
```

Verify the file permissions after copying the data back. You may need to adjust the permissions. For example, the following command changes the owner of the file location:

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

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

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

6.4 Encrypted Backup

Percona XtraBackup has implemented support for encrypted backups. It can be used to encrypt/decrypt local or streaming backup with xbstream option (streaming tar backups are not supported) in order to add another layer of protection to the backups. Encryption is done with the `libgcrypt` library.

6.4.1 Creating Encrypted Backups

To make an encrypted backup following options need to be specified (options `xtrabackup --encrypt-key` and `xtrabackup --encrypt-key-file` are mutually exclusive, i.e., just one of them needs to be provided):

- `--encrypt=ALGORITHM` - currently supported algorithms are: `AES128`, `AES192` and `AES256`
- `--encrypt-key=ENCRYPTION_KEY` - proper length encryption key to use. It is not recommended to use this option where there is uncontrolled access to the machine as the command line and thus the key can be viewed as part of the process info.
- `--encrypt-key-file=KEYFILE` - the name of a file where the raw key of the appropriate length can be read from. The file must be a simple binary (or text) file that contains exactly the key to be used.

Both `xtrabackup --encrypt-key` option and `xtrabackup --encrypt-key-file` option can be used to specify the encryption key. Encryption key can be generated with command like:

```
$ openssl rand -base64 24
```

Example output of that command should look like this:

```
GCHFLrDFVx6UAsRb88uLVbAVWbK+Yzfs
```

This value then can be used as the encryption key

Using the `--encrypt-key` option

Example of the xtrabackup command using the `xtrabackup --encrypt-key` should look like this:

```
$ xtrabackup --backup --target-dir=/data/backups --encrypt=AES256 \
--encrypt-key="GCHFLrDFVx6UAsRb88uLVbAVWbK+Yzfs"
```

Using the `--encrypt-key-file` option

Example of the xtrabackup command using the `xtrabackup --encrypt-key-file` should look like this:

```
$ xtrabackup --backup --target-dir=/data/backups/ --encrypt=AES256 \
--encrypt-key-file=/data/backups/keyfile
```

Note

Depending on the text editor used for making the `KEYFILE`, text file in some cases can contain the CRLF and this will cause the key size to grow and thus making it invalid. Suggested way to do this would be to create the file with: `echo -n "GCHFLrDFVx6UAsRb88uLVbAVWbK+Yzfs" > /data/backups/keyfile`

6.4.2 Optimizing the encryption process

Two options have been introduced with the encrypted backups that can be used to speed up the encryption process. These are `xtrabackup --encrypt-threads` and `xtrabackup --encrypt-chunk-size`. By using the `xtrabackup --encrypt-threads` option multiple threads can be specified to be used for encryption in parallel. Option `xtrabackup --encrypt-chunk-size` can be used to specify the size (in bytes) of the working encryption buffer for each encryption thread (default is 64K).

6.4.3 Decrypting Encrypted Backups

Percona XtraBackup `xtrabackup --decrypt` option has been implemented that can be used to decrypt the backups:

```
$ xtrabackup --decrypt=AES256 --encrypt-key="GCHFLrDFVx6UAsRb88uLVbAVWbK+Yzfs"\
--target-dir=/data/backups/
```

Percona XtraBackup doesn't automatically remove the encrypted files. In order to clean up the backup directory users should remove the *.xbcrypt files. In *Percona XtraBackup* 2.4.6 `xtrabackup --remove-original` option has been implemented that you can use to remove the encrypted files once they've been decrypted. To remove the files once they're decrypted you should run:

```
$ xtrabackup --decrypt=AES256 --encrypt-key="GCHFLrDFVx6UAsRb88uLVbAVWbK+Yzfs"\
--target-dir=/data/backups/ --remove-original
```

Note

`xtrabackup --parallel` can be used with `xtrabackup --decrypt` option to decrypt multiple files simultaneously.

When the files have been decrypted backup can be prepared.

6.4.4 Preparing Encrypted Backups

After the backups have been decrypted, they can be prepared the same way as the standard full backups with the `xtrabackup --prepare` option:

```
$ xtrabackup --prepare --target-dir=/data/backups/
```

6.4.5 Restoring Encrypted Backups

xtrabackup has a `xtrabackup --copy-back` option, which performs the restoration of a backup to the server's datadir:

```
$ xtrabackup --copy-back --target-dir=/data/backups/
```

It will copy all the data-related files back to the server's `datadir`, determined by the server's `my.cnf` configuration file. You should check the last line of the output for a success message:

```
170214 12:37:01 completed OK!
```

6.4.6 Other Reading

- The Libgcrypt Reference Manual

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

7. User’s Manual

7.1 Percona XtraBackup User Manual

Percona XtraBackup is a set of following tools:

innobackupex

```
innobackupex is the symlink for *xtrabackup*. innobackupex still
supports all features and syntax as 2.2 version did, but is now
deprecated and will be removed in next major release.
```

xtrabackup

```
a compiled *C* binary that provides functionality to backup a whole *MySQL*
database instance with *MyISAM*, *InnoDB*, and XtraDB tables.
```

xbcrypt

```
utility used for encrypting and decrypting backup files.
```

xbstream

```
utility that allows streaming and extracting files to/from the
xbstream format.
```

xbcloud

```
utility used for downloading and uploading full or part of xbstream
archive from/to cloud.
```

After *Percona XtraBackup* 2.3 release the recommend way to take the backup is using the *xtrabackup* script. More information on script options can be found in how to use xtrabackup.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

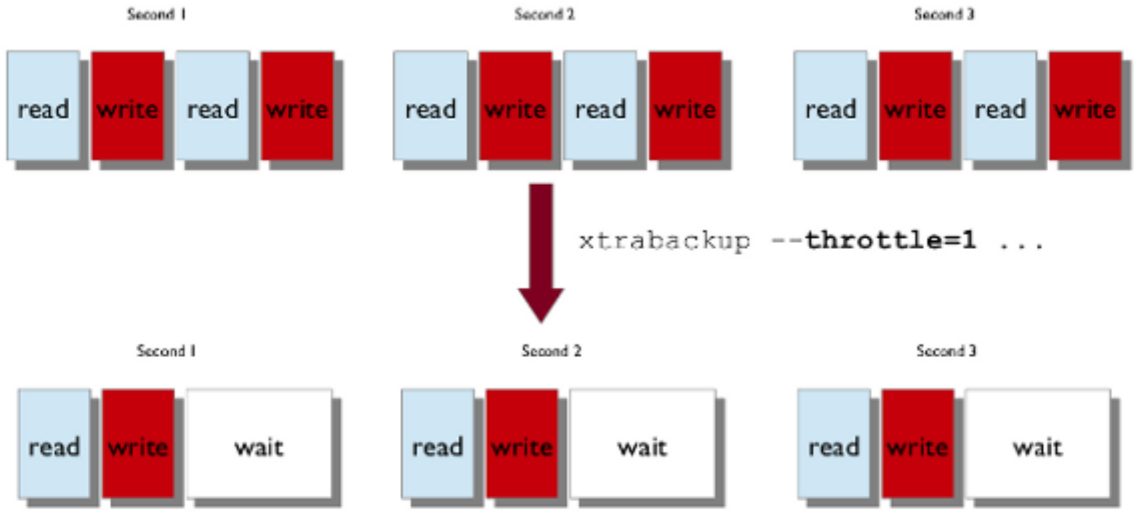
Last update: 2022-07-19

8. Advanced Features

8.1 Throttling Backups

Although xtrabackup does not block your database's operation, any backup can add load to the system being backed up. On systems that do not have much spare I/O capacity, it might be helpful to throttle the rate at which xtrabackup reads and writes data. You can do this with the `xtrabackup --throttle` option. This option limits the number of chunks copied per second. The chunk size is *10 MB*.

The image below shows how throttling works when `xtrabackup --throttle` is set to 1.



By default, there is no throttling, and xtrabackup reads and writes data as quickly as possible. If you set too strict of a limit on the IOPS, the backup may slow down so much that it will never catch up with the transaction logs that InnoDB is writing, and the backup might never be complete.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

8.2 Lockless binary log information

This feature is exclusive to *Percona Server for MySQL* starting with version 5.6.26-74.0. It is also used in *Percona XtraDB Cluster* when the node is being backed up without `xtrabackup --galera-info`.

When the Lockless binary log information feature is available on the server, *Percona XtraBackup* can trust binary log information stored in the *InnoDB* system header and avoid executing `LOCK BINLOG FOR BACKUP` (and thus, blocking commits for the duration of finalizing the `REDO` log copy) under a number of circumstances:

- when the server is not a GTID-enabled Galera cluster node
- when the replication I/O thread information should not be stored as a part of the backup (i.e. when the `xtrabackup --slave-info` option is not specified)

If all of the above conditions hold, *Percona XtraBackup* does not execute the `SHOW MASTER STATUS` as a part of the backup procedure, does not create the `xtrabackup_binlog_info` file on backup. Instead, that information is retrieved and the file is created after preparing the backup, along with creating `xtrabackup_binlog_pos_innodb`, which in this case contains exactly the same information as in `xtrabackup_binlog_info` and is thus redundant.

To make this new functionality configurable, there is now a new *Percona XtraBackup* option, `xtrabackup --binlog-info`, which can accept the following values:

- `OFF` - This means that *Percona XtraBackup* will not attempt to retrieve the binary log information at all, neither during the backup creation, nor after preparing it. This can help when a user just wants to copy data without any meta information like binary log or replication coordinates. In this case, `xtrabackup --binlog-info=OFF` can be passed to *Percona XtraBackup* and `LOCK BINLOG FOR BACKUP` will not be executed, even if the backup-safe binlog info feature is not provided by the server (but the backup locks feature is still a requirement).
- `ON` - This matches the old behavior, i.e. the one before this *Percona XtraBackup* feature had been implemented. When specified, *Percona XtraBackup* retrieves the binary log information and uses `LOCK BINLOG FOR BACKUP` (if available) to ensure its consistency.
- `LOCKLESS` - This corresponds to the functionality explained above: *Percona XtraBackup* will not retrieve binary log information during the backup process, will not execute `LOCK BINLOG FOR BACKUP`, and the `xtrabackup_binlog_info` file will not be created. The file will be created after preparing the backup using the information stored in the InnoDB system header. If the required server-side functionality is not provided by the server, specifying this `xtrabackup --binlog-info` value will result in an error. If one of the above mentioned conditions does not hold, `LOCK BINLOG FOR BACKUP` will still be executed to ensure consistency of other meta data.
- `AUTO` - This is the default value. When used, *Percona XtraBackup* will automatically switch to either `ON` or `LOCKLESS`, depending on the server-side feature availability, i.e., whether the `have_backup_safe_binlog_info` server variable is available.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

8.3 Encrypted InnoDB Tablespace Backups

As of *MySQL* 5.7.11, InnoDB supports data encryption for InnoDB tables stored in file-per-table tablespaces. This feature provides at-rest encryption for physical tablespace data files.

For authenticated user or application to access encrypted tablespace, InnoDB will use master encryption key to decrypt the tablespace key. The master encryption key is stored in a keyring. Two keyring plugins supported by xtrabackup are `keyring_file` and `keyring_vault`. These plugins are installed into the plugin directory.

8.3.1 Making a Backup Using `keyring_file` Plugin

Support for encrypted InnoDB tablespace backups with `keyring_file` has been implemented in *Percona XtraBackup* 2.4.2 by implementing `xtrabackup --keyring-file-data` option (and also `xtrabackup --server-id` option, needed for *MySQL* prior to 5.7.13). These options are only recognized by xtrabackup binary i.e., innobackupex will not be able to backup and prepare encrypted tablespaces.

Creating Backup

In order to backup and prepare database containing encrypted InnoDB tablespaces, you must specify the path to keyring file by using the `xtrabackup --keyring-file-data` option.

```
$ xtrabackup --backup --target-dir=/data/backup/ --user=root \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

With *MySQL* prior to 5.7.13, use `xtrabackup --server-id` in the backup creation command:

```
$ xtrabackup --backup --target-dir=/data/backup/ --user=root \
--keyring-file-data=/var/lib/mysql-keyring/keyring --server-id=1
```

After xtrabackup is finished taking the backup you should see the following message:

```
xtrabackup: Transaction log of lsn (5696709) to (5696718) was copied.
160401 10:25:51 completed OK!
```

> ⚠ **Warning**
>
> xtrabackup will not copy keyring file into the backup directory. In order to be prepare the backup, you must make a copy of keyring file yourself.

Preparing Backup

In order to prepare the backup you'll need to specify the keyring-file-data (server-id is stored in `backup-my.cnf` file, so it can be omitted when preparing the backup, regardless of the *MySQL* version used).

```
$ xtrabackup --prepare --target-dir=/data/backup \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

After xtrabackup is finished preparing the backup you should see the following message:

```
InnoDB: Shutdown completed; log sequence number 5697064
160401 10:34:28 completed OK!
```

Backup is now prepared and can be restored with xtrabackup –copy-back option. In case the keyring has been rotated you'll need to restore the keyring which was used to take and prepare the backup.

8.3.2 Making Backup Using `keyring_vault` Plugin

Support for encrypted InnoDB tablespace backups with `keyring_vault` has been implemented in *Percona XtraBackup* 2.4.11. Keyring vault plugin settings are described here.

Creating Backup

The following command creates a backup in the `/data/backup` directory:

```
$ xtrabackup --backup --target-dir=/data/backup --user=root
```

After xtrabackup completes taking the backup you should see the following message:

```
xtrabackup: Transaction log of lsn (5696709) to (5696718) was copied.
160401 10:25:51 completed OK!
```

Preparing the Backup

In order to prepare the backup xtrabackup will need an access to the keyring. Since xtrabackup doesn't talk to MySQL server and doesn't read default `my.cnf` configuration file during prepare, user will need to specify keyring settings via the command line:

```
$ xtrabackup --prepare --target-dir=/data/backup \
--keyring-vault-config=/etc/vault.cnf
```

See also

Data at Rest Encryption for Percona Server [keyring vault plugin settings] (https://www.percona.com/doc/percona-server/LATEST/management/data_at_rest_encryption.html#keyring-vault-plugin).

After xtrabackup completes preparing the backup you should see the following message:

```
InnoDB: Shutdown completed; log sequence number 5697064
160401 10:34:28 completed OK!
```

The backup is now prepared and can be restored with `xtrabackup --copy-back` option:

```
$ xtrabackup --copy-back --target-dir=/data/backup --datadir=/data/mysql
```

8.3.3 Incremental Encrypted InnoDB Tablespace Backups with `keyring_file`

The process of taking incremental backups with InnoDB tablespace encryption is similar to taking the Incremental Backups with unencrypted tablespace.

Creating an Incremental Backup

To make an incremental backup, begin with a full backup. The xtrabackup binary writes a file called `xtrabackup_checkpoints` into the backup's target directory. This file contains a line showing the `to_lsn`, which is the database's `LSN` at the end of the backup. First you need to create a full backup with the following command:

```
$ xtrabackup --backup --target-dir=/data/backups/base \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

⚠ **Warning**

xtrabackup will not copy keyring file into the backup directory. In order to be prepare the backup, you must make a copy of keyring file yourself. If you try to restore the backup after the keyring has been changed you'll see errors like `ERROR 3185 (HY000): Can't find master key from keyring, please check keyring plugin is loaded.` when trying to access encrypted table.

If you look at the `xtrabackup_checkpoints` file, you should see some contents similar to the following:

```
backup_type = full-backuped
from_lsn = 0
to_lsn = 7666625
last_lsn = 7666634
compact = 0
recover_binlog_info = 1
```

Now that you have a full backup, you can make an incremental backup based on it. Use a command such as the following:

```
$ xtrabackup --backup --target-dir=/data/backups/inc1 \
--incremental-basedir=/data/backups/base \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

⚠ **Warning**

xtrabackup will not copy keyring file into the backup directory. In order to be prepare the backup, you must make a copy of keyring file yourself. If the keyring hasn't been rotated you can use the same as the one you've backed-up with the base backup. If the keyring has been rotated you'll need to back it up otherwise you won't be able to prepare the backup.

The `/data/backups/inc1/` directory should now contain delta files, such as `ibdata1.delta` and `test/table1.ibd.delta`. These represent the changes since the `LSN 7666625`. If you examine the `xtrabackup_checkpoints` file in this directory, you should see something similar to the following:

```
backup_type = incremental
from_lsn = 7666625
to_lsn = 8873920
last_lsn = 8873929
```

```
compact = 0
recover_binlog_info = 1
```

The meaning should be self-evident. It's now possible to use this directory as the base for yet another incremental backup:

```
$ xtrabackup --backup --target-dir=/data/backups/inc2 \
--incremental-basedir=/data/backups/inc1 \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

Preparing the Incremental Backups

The `xtrabackup --prepare` step for incremental backups is not the same as for normal backups. In normal backups, two types of operations are performed to make the database consistent: committed transactions are replayed from the log file against the data files, and uncommitted transactions are rolled back. You must skip the rollback of uncommitted transactions when preparing a backup, because transactions that were uncommitted at the time of your backup may be in progress, and it's likely that they will be committed in the next incremental backup. You should use the `xtrabackup --apply-log-only` option to prevent the rollback phase.

> ⚠ **Warning**
>
> If you do not use the `xtrabackup --apply-log-only` option to prevent the rollback phase, then your incremental backups will be useless. After transactions have been rolled back, further incremental backups cannot be applied.

Beginning with the full backup you created, you can prepare it, and then apply the incremental differences to it. Recall that you have the following backups:

```
/data/backups/base
/data/backups/inc1
/data/backups/inc2
```

To prepare the base backup, you need to run `xtrabackup --prepare` as usual, but prevent the rollback phase:

```
$ xtrabackup --prepare --apply-log-only --target-dir=/data/backups/base \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

The output should end with some text such as the following:

```
InnoDB: Shutdown completed; log sequence number 7666643
InnoDB: Number of pools: 1
160401 12:31:11 completed OK!
```

To apply the first incremental backup to the full backup, you should use the following command:

```
$ xtrabackup --prepare --apply-log-only --target-dir=/data/backups/base \
--incremental-dir=/data/backups/inc1 \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

⚠ **Warning**

Backup should be prepared with the keyring that was used when backup was being taken. This means that if the keyring has been rotated between the base and incremental backup that you'll need to use the keyring that was in use when the first incremental backup has been taken.

Preparing the second incremental backup is a similar process: apply the deltas to the (modified) base backup, and you will roll its data forward in time to the point of the second incremental backup:

```
$ xtrabackup --prepare --target-dir=/data/backups/base \
--incremental-dir=/data/backups/inc2 \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

Note

xtrabackup `--apply-log-only` should be used when merging each incremental backup except the last one. That's why the previous line doesn't contain `xtrabackup --apply-log-only`. Even if the `xtrabackup --apply-log-only` were used on the last step, backup would still be consistent but in that case server would perform the rollback phase.

The backup is now prepared and can be restored with `xtrabackup --copy-back`. In case the keyring has been rotated you'll need to restore the keyring which was used to take and prepare the backup.

8.3.4 Restoring backup when keyring is not available

While described restore method works, it requires an access to the same keyring which server is using. It may not be possible if backup is prepared on different server or at the much later time, when keys in the keyring have been purged, or in case of malfunction when keyring vault server is not available at all.

A `xtrabackup --transition-key` should be used to make it possible for xtrabackup to process the backup without access to the keyring vault server. In this case xtrabackup will derive AES encryption key from specified passphrase and will use it to encrypt tablespace keys of tablespaces being backed up.

Creating the Backup with a Passphrase

The following example illustrates how a backup can be created in this case:

```
$ xtrabackup --backup --user=root -p --target-dir=/data/backup \
--transition-key=MySecretKey
```

If `xtrabackup --transition-key` is specified without a value, xtrabackup will ask for it.

Note

`xtrabackup --transition-key` scrapes the supplied value so that it does not appear in the `ps` command output.

Preparing the Backup with a Passphrase

The same passphrase should be specified for the `prepare` command:

```
$ xtrabackup --prepare --target-dir=/data/backup \
--transition-key=MySecretKey
```

There is no `keyring-vault` or `keyring-file` here, because xtrabackup does not talk to the keyring in this case.

Restoring the Backup with a Generated Key

When restoring a backup you will need to generate new master key. Here is the example for `keyring_file`:

```
$ xtrabackup --copy-back --target-dir=/data/backup --datadir=/data/mysql \
--transition-key=MySecetKey --generate-new-master-key \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

In case of `keyring_vault` it will look like this:

```
$ xtrabackup --copy-back --target-dir=/data/backup --datadir=/data/mysql \
--transition-key=MySecetKey --generate-new-master-key \
--keyring-vault-config=/etc/vault.cnf
```

xtrabackup will generate new master key, store it into target keyring vault server and re-encrypt tablespace keys using this key.

8.3.5 Making the Backup with a Stored Transition Key

Finally, there is an option to store transition key in the keyring. In this case xtrabackup will need access to the same keyring file or vault server during prepare and copy-back, but does not depend on whether the server keys have been purged.

The three stages of the backup process are the following:

Backup

```
$ xtrabackup --backup --user=root -p --target-dir=/data/backup \
--generate-transition-key
```

Prepare

`keyring_file` variant

```
$ xtrabackup --prepare --target-dir=/data/backup \
--keyring-file-data=/var/lib/mysql-keyring/keyring
```

`keyring_vault` variant

```
$ xtrabackup --prepare --target-dir=/data/backup \
--keyring-vault-config=/etc/vault.cnf
```

Copy-back

`keyring_file` variant

```
$ xtrabackup --copy-back --target-dir=/data/backup --datadir=/data/mysql \
--generate-new-master-key --keyring-file-data=/var/lib/mysql-keyring/keyring
```

`keyring_vault` variant

```
$ xtrabackup --copy-back --target-dir=/data/backup --datadir=/data/mysql \
--generate-new-master-key --keyring-vault-config=/etc/vault.cnf
```

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-12-06

8.4 `lock-ddl-per-table` Option Improvements

MySQL 5.7 has made improvements to bulk loading, one of these improvements is the ability to disable redo logging, which increased the speed of ADD INDEX operations.

To block DDL statements on an instance, Percona Server implemented LOCK TABLES FOR BACKUP. *Percona XtraBackup* uses this lock for the duration of the backup. This lock does not affect DML statements.

Percona XtraBackup has also implemented `--lock-ddl-per-table`, which blocks DDL statements by using metadata locks (MDL).

The following procedures describe a simplified backup operation when using `--lock-ddl-per-table`:

1. Parse and copy all redo logs after the checkpoint mark
2. Fork a dedicated thread to continue following new redo log entries
3. List the tablespaces required to copy
4. Iterate through the list. The following steps occur with each listed tablespace:
    - Query INFORMATION_SCHEMA.INNODB_SYS_TABLES to find which tables belong to the tablespace ID and take a MDL on the underlying table or tables in case there is a shared tablespace.
    - Copy the tablespace `.ibd` files.

The backup process may encounter a redo log event, generated by the bulk load operations, which notifies backup tools that data file changes have been omitted from the redo log. This event is an `MLOG_INDEX_LOAD`. If this event is found by the redo follow thread, the backup continues and assumes the backup is safe because the MDL protects tablespaces already copied and the `MLOG_INDEX_LOAD` event is for a tablespace that is not copied.

These assumptions may not be correct and may lead to inconsistent backups.

8.4.1 `--lock-ddl-per-table` redesign

Implemented in *Percona XtraBackup* version 2.4.21, the `--lock-ddl-per-table` has been redesigned to minimize inconsistent backups. The following procedure reorders the steps:

- Acquire the MDL lock at the beginning of the backup
- Scan the redo logs. An `MLOG_INDEX_LOAD` event may be recorded if a `CREATE INDEX` statement has occurred before the backup starts. At this time, the backup process is safe and can parse and accept the event.
- After the first scan has completed, the follow redo log thread is initiated This thread stops the backup process if an `MLOG_INDEX_LOAD` event is found.
- Gather the tablespace list to copy
- Copy the `.ibd` files.

8.4.2 Other Improvements

The following improvements have been added:

- If the `.ibd` file belongs to a temporary table, the `SELECT` query is skipped.
- For a FullText Index, an MDL is acquired on the base table.
- A `SELECT` query that acquires an MDL retrieves no data.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

9. Tutorials, Recipes, How-tos

9.1 How-tos and Recipes

9.1.1 Recipes for innobackupex

- Make a Local Full Backup (Create, Prepare and Restore)
- Make a Streaming Backup
- Making an Incremental Backup
- Making a Compressed Backup
- Backing Up and Restoring Individual Partitions

9.1.2 Recipes for *xtrabackup*

- Making a Full Backup
- Making an Incremental Backup
- Restoring the Backup

9.1.3 How-Tos

- How to setup a replica for replication in 6 simple steps with Percona XtraBackup
- Verifying Backups with replication and pt-checksum
- How to create a new (or repair a broken) GTID based slave

9.1.4 Auxiliary Guides

- Enabling the server to communicate via TCP/IP
- Privileges and Permissions for Users
- Installing and configuring a SSH server

9.1.5 Assumptions in this section

The context should make the recipe or tutorial understandable. To assure that this is true, a list of the assumptions, names and other objects that appears in this section. This items are specified at the beginning of each recipe or tutorial.

`HOST`

A system with a *MySQL*-based server installed, configured and running. We assume the following about this system:

- The MySQL server is able to communicate with others by the standard TCP/IP port;
- An SSH server is installed and configured - see here if it is not;
- You have an user account in the system with the appropriate permissions
- You have a MySQL's user account with appropriate Connection and Privileges Needed.

`USER`

This is a user account with shell access and the appropriate permissions for the task. A guide for checking them is here.

`DB-USER`

This is a user account in the database server with the appropriate privileges for the task. A guide for checking them is here.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

10. References

10.1 Release notes

10.1.1 Percona XtraBackup 2.4 Release notes index

- Percona XtraBackup 2.4.29 (2023-12-18)
- Percona XtraBackup 2.4.28 (2023-04-04)
- Percona XtraBackup 2.4.27 (2022-12-06)
- Percona XtraBackup 2.4.26 (2022-05-09)
- Percona XtraBackup 2.4.25 (2022-04-16)
- Percona XtraBackup 2.4.24 (2021-09-14)
- Percona XtraBackup 2.4.23 (2021-06-22)
- Percona XtraBackup 2.4.22 (2021-03-22)
- Percona XtraBackup 2.4.21 (2020-11-12)
- Percona XtraBackup 2.4.20 (2020-04-14)
- Percona XtraBackup 2.4.19 (2020-03-25)
- Percona XtraBackup 2.4.18 (2019-12-16)
- Percona XtraBackup 2.4.17 (2019-12-09)
- Percona XtraBackup 2.4.16 (2019-11-04)
- Percona XtraBackup 2.4.15 (2019-07-10)
- Percona XtraBackup 2.4.14 (2019-05-01)
- Percona XtraBackup 2.4.13 (2019-01-18)
- Percona XtraBackup 2.4.12 (2018-06-22)
- Percona XtraBackup 2.4.11 (2018-04-23)
- Percona XtraBackup 2.4.10 (2018-03-30)
- Percona XtraBackup 2.4.9 (2017-11-29)
- Percona XtraBackup 2.4.8 (2017-07-24)
- Percona XtraBackup 2.4.7-2 (2017-05-29)
- Percona XtraBackup 2.4.7 (2017-04-17)
- Percona XtraBackup 2.4.6 (2017-02-22)
- Percona XtraBackup 2.4.5 (2016-11-29)
- Percona XtraBackup 2.4.4 (2016-07-25)
- Percona XtraBackup 2.4.3 (2016-05-23)
- Percona XtraBackup 2.4.2 (2016-04-01)
- Percona XtraBackup 2.4.1 (2016-02-16)

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-12-11

10.1.2 Percona XtraBackup 2.4.29 (2023-12-18)

> **Important**
>
> Percona XtraBackup 2.4.29 is the final release of the Percona XtraBackup 2.4 series.
>
> We recommend that you either upgrade to MySQL 8.0 or stay on 5.7; we'll support you.

Percona XtraBackup for MySQL Databases enables MySQL backups without blocking user queries.

Percona XtraBackup 2.4 does not support making backups of databases created in MySQL 8.0 or higher, Percona Server for MySQL 8.0 or higher, or Percona XtraDB Cluster 8.0 or higher. Install Percona XtraBackup 8.0 to make backups for these versions.

Release highlights

This release merges the MySQL 5.7.44 code base. This release does not contain new improvements or new bug fixes from Percona.

Useful links

- Install Percona XtraBackup 2.4
- The Percona XtraBackup GitHub repository
- Contribute to the documentation
- Download product binaries, packages, and tarballs at Percona Product Downloads
- For training, contact Percona Training - Start learning now.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-12-11

10.1.3 Percona XtraBackup 2.4.28 (2023-04-04)

Release date	**April 4, 2023**
Install instructions	Install Percona XtraBackup 2.4

Percona XtraBackup for MySQL Databases enables MySQL backups without blocking user queries. Percona XtraBackup is ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, Percona XtraBackup drives down backup costs while providing unique features for MySQL backups.

Percona XtraBackup 2.4 does not support making backups of databases created in MySQL 8.0, Percona Server for MySQL 8.0, or Percona XtraDB Cluster 8.0. Install Percona XtraBackup 8.0 to make backups for these versions.

Release highlights

This release fixes the security vulnerability CVE-2022-25834 with PXB-2977.

Useful links

- Percona XtraBackup GitHub location
- Contribute to the documentation
- For training, contact Percona Training - Start learning now.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-04-04

10.1.4 Percona XtraBackup 2.4.27 (2022-12-06)

Release date	**December 06, 2022**
Install instructions	Install Percona XtraBackup 2.4
Download this version	Percona XtraBackup 2.4

Percona XtraBackup for MySQL Databases enables MySQL backups without blocking user queries. Percona XtraBackup is ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, Percona XtraBackup drives down backup costs while providing unique features for MySQL backups.

Percona XtraBackup 2.4 does not support making backups of databases created in MySQL 8.0, Percona Server for MySQL 8.0, or Percona XtraDB Cluster 8.0. Download Percona XtraBackup 8.0 to make backups for these versions.

Release highlights

This release contains fixes for backing up Percona XtraDB Cluster (PXC) and an xbcloud fix.

Bugs fixed

- PXB-2809: `wsrep_sync_wait<>0` caused a `Lock wait timeout exceeded` error when issued against Percona XtraDB Cluster (Thanks to Frank Well for providing the initial patch).
- PXB-2840: Xbcloud did not have a required set location constraint when creating a bucket on a non-default endpoint (Thanks to Jason Zareski for the patch).
- PXB-2906: Fixed failed query errors in backup for `wsrep_sync_wait` and `group-replication-consistency`.
- PXB-2729: Percona XtraBackup compilation failed if GTest packages are installed.

Platform support

- Percona XtraBackup 2.4.27 supports Oracle Linux/Red Hat Enterprise 9.
- Percona XtraBackup 2.4.27 supports Ubuntu 22.04.

Useful links

- Percona XtraBackup GitHub location
- Read the Documentation Contribution Guide to add to the documentation.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-29

10.1.5 Percona XtraBackup 2.4.26

- **Date**

  May 9, 2022

Percona XtraBackup for MySQL Databases enables MySQL backups without blocking user queries. Percona XtraBackup is ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, Percona XtraBackup drives down backup costs while providing unique features for MySQL backups.

Percona XtraBackup 2.4 does not support making backups of databases created in *MySQL 8.0*, *Percona Server for MySQL 8.0*, or *Percona XtraDB Cluster 8.0*. Use Percona XtraBackup 8.0 to make backups for these versions.

Release Highlights

Fixed a segmentation fault when creating a tmpdir.

Bugs Fixed

- PXB-2756: Fixed a segmentation fault when creating a tmpdir.

Useful Links

- The Percona XtraBackup installation instructions
- The Percona XtraBackup downloads
- The Percona XtraBackup GitHub location
- To contribute to the documentation, review the Documentation Contribution Guide

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.6 Percona XtraBackup 2.4.25

- **Date**

  April 26, 2022

Percona XtraBackup for MySQL Databases enables MySQL backups without blocking user queries. Percona XtraBackup is ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, Percona XtraBackup drives down backup costs while providing unique features for MySQL backups.

Percona XtraBackup 2.4 does not support making backups of databases created in *MySQL 8.0*, *Percona Server for MySQL 8.0*, or *Percona XtraDB Cluster 8.0*. Use Percona XtraBackup 8.0 to make backups for these versions.

Release Highlights

The xbcloud binary adds support for the Microsoft Azure Cloud Storage using the REST API.

New Features

- PXB-1883: Implements support for Microsoft Azure Cloud Storage in the xbcloud binary. (Thanks to Ivan Groenewold for reporting this issue)

Bugs Fixed

- PXB-2608: Upgraded the Vault API to V2 (Thanks to Benedito Marques Magalhaes for reporting this issue)
- PXB-2649: Fix for compilation issues on GCC-10.
- PXB-2648: CURL prior to 7.38.0 version doesn't use CURLE_HTTP2 and throws an error `'CURLE_HTTP2' is not a member of 'CURLcode'`. Added CURLE_OBSOLETE16 as a connectivity error code. In CURL versions after 7.38.0, CURLE_OBSOLETE16 is translated to CURLE_HTTP2.
- PXB-2711: Fix for libgcrypt initialization warnings in xtrabackup.
- PXB-2722: Fix for when via command line, a password, passed using the -p option, was written into the backup tool_command in xtrabackup_info.

Useful Links

- The Percona XtraBackup installation instructions
- The Percona XtraBackup downloads
- The Percona XtraBackup GitHub location
- To contribute to the documentation, review the Documentation Contribution Guide

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.7 Percona XtraBackup 2.4.24

- **Date**

  September 14, 2021

- **Installation**

  Installing Percona XtraBackup

Percona XtraBackup enables MySQL backups without blocking user queries, making it ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, it drives down backup costs while providing unique features for MySQL backups.

Improvements

- PXB-2477: The xbcloud Binary should retry on error and utilize incremental backoff (Thanks to Baptiste Mille-Mathias for reporting this issue)
- PXB-2580: With the xbcloud binary, a chunk-upload on SSL connect error to Amazon S3 was not retried. (Thanks to Tim Vaillancourt for providing the patch)

Bugs Fixed

PXB-1504: The FIND_GCRYPT macro is broken. (Thanks to Maxim Bublis for reporting this issue)

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.8 Percona XtraBackup 2.4.23

- **Date**

  June 22, 2021

- **Installation**

  Installing Percona XtraBackup

Percona XtraBackup enables MySQL backups without blocking user queries, making it ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, it drives down backup costs while providing unique features for MySQL backups.

Improvements

- PXB-2487: Problems with the typesetting in the Partial Backups document (Thanks to user qing for reporting this issue).

Bugs Fixed

- PXB-1462: Long `gtid_executed` breaks `--history` functionality.
- PXB-2486: When the `--encrypt` and `--parallel` parameters are used, XtraBackup does not handle a broken pipe correctly.
- PXB-1855: Format correction for the `xtrabackup –databasesv options.
- PXB-2427: Update the XtraBackup Help description for the parameter `--stream`.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.9 Percona XtraBackup 2.4.22

- **Date**

  March 22, 2021
- **Installation**

  Installing Percona XtraBackup

Percona XtraBackup enables MySQL backups without blocking user queries, making it ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, it drives down backup costs while providing unique features for MySQL backups.

This release fixes the security vulnerability CVE-2020-29488

Bugs Fixed

- PXB-2171: Add missing PXB help options to the xtrabackup options reference
- PXB-2395: Update versions for xbstream and xbcrypt
- PXB-2394: Correct spellings in xbcloud help

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.10 Percona XtraBackup 2.4.21

- **Date**

  November 12, 2020
- **Installation**

  Installing Percona XtraBackup

Percona XtraBackup enables MySQL backups without blocking user queries, making it ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, it drives down backup costs while providing unique features for MySQL backups.

New Features

- PXB-2112: xbcloud: support `storage_class` option with `--storage=s3` (Thanks to user rluisr for reporting this issue)

Improvements

- PXB-2254: Redesign `--lock-ddl-per-table`
- PXB-2252: Introduce debug option to print the redo log records scanned and applied

Bugs Fixed

- PXB-793: Fix syntax error when executing `--lock-ddl-per-table` queries
- PXB-2165: Modify xbcloud to store backups using s3 access key parameters if AWS access key env variables are set
- PXB-2164: Modify xbcloud to return the error when the backup doesn't exist in s3 bucket
- PXB-953: Improve stdout for the end of usage of `--lock-ddl-per-table`
- PXB-2279: Xbcloud: Upload failed: backup is incomplete (Thanks to user mrmainnet for reporting this issue)
- PXB-2127: Modify xbcloud to upload backups with empty database to min.io storage (Thanks to user hartland for reporting this issue)
- PXB-2275: Modify backup processing to add validations if an encrypted table is created
- PXB-2272: Fixed Regexp from `is_tmp_table` doesn't account for all temporary tables
- PXB-2257: fixed `--lock-ddl-per-table` to properly close database connection
- PXB-2249: Verify perl binary exists before completing version check
- PXB-2239: Partitioned table is not restored correctly when partitions are changed during backup
- PXB-2238: Provide binary tarball with shared libs and glibc suffix & minimal tarballs
- PXB-2216: Verify encryption version when opening tables to avoid changing encryption version
- PXB-2202: Modify Xbcloud to display an error when xtrabackup fails to create a backup
- PXB-2198: Modify xbcloud delete to return the error when the backup doesn't exist in s3 bucket

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.11 Percona XtraBackup 2.4.20

- **Date**

  April 14, 2020
- **Installation**

  Installing Percona XtraBackup

Percona XtraBackup enables MySQL backups without blocking user queries, making it ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, it drives down backup costs while providing unique features for MySQL backups.

This release fixes security vulnerability CVE-2020-10997

> **Seealso**
>
> Percona Database Performance Blog CVE-2020-10997

Bugs Fixed

- PXB-1783: Xtrabackup GTID is incorrect after prepare
- PXB-2154: Xbstream displayed the encrypt-key in process during backup decryption
- PXB-2152: PXB wrote a new master key to standard error output
- PXB-2145: encrypt-key could appear in the process-list
- PXB-2142: Transition key was written to backup/stream

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.12 Percona XtraBackup 2.4.19

- **Date**

  March 25, 2020
- **Installation**

  Installing Percona XtraBackup

Downloads are available from our download site and from apt and yum repositories.

Percona XtraBackup enables MySQL backups without blocking user queries, making it ideal for companies with large data sets and mission-critical applications that cannot tolerate long periods of downtime. Offered free as an open source solution, it drives down backup costs while providing unique features for *MySQL* backups.

All Percona software is open-source and free.

Bugs Fixed

- PXB-1982: The history table showed a wrong value for `lock_time`.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

10.1.13 Percona XtraBackup 2.4.18

Percona is glad to announce the release of *Percona XtraBackup* 2.4.18 on December 16, 2019. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

Bugs Fixed

Sometime between December 3rd and December 10th, a change was introduced in `AWS (Amazon Web Services)` that caused an incompatibility with our *Percona XtraBackup* `xbcloud` utility. Bug fixed PXB-1978.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.14 Percona XtraBackup 2.4.17

Percona is glad to announce the release of *Percona XtraBackup* 2.4.17 on December 9, 2019. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

Bugs Fixed

Percona XtraBackup could crash when making a backup for Percona Server 5.7.28-31 where the tablespace encryption was used. Bug fixed PXB-1968.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.15 Percona XtraBackup 2.4.16

Percona is glad to announce the release of *Percona XtraBackup* 2.4.16 on November 4, 2019. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

Improvements

Two options (`--backup-lock-timeout` and `--backup-lock-retry-count`) were added to enable the configuring of the timeout for acquiring metadata locks in `FLUSH TABLES WITH READ LOCK`, `LOCK TABLE FOR BACKUP`, and `LOCK BINLOG`
`FOR BACKUP` statements. More information in PXB-1914

Bugs Fixed

- Percona Xtrabackup was not able to connect to the database when the password was specified along with the transition-key parameter. Bug fixed PXB-1902.
- In some cases, Percona Xtrabackup stuck with redo log corruption when master key is rotated. Bug fixed PXB-1903.
- In rare cases, when both full and incremental backups were made before MySQL flushed the first page of the encrypted tablespace, Percona Xtrabackup could crash during the incremental backup prepare for the tablespace encryption. Bug fixed PXB-1894.
- An encrypted table could not be restored when ADD/DROP INDEX was run on the table. Bug fixed PXB-1905.
- In some cases `xtrabackup --prepare` could fail to decrypt a table but reported that the operation *completed ok*. Bug fixed PXB-1936.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.16 Percona XtraBackup 2.4.15

Percona is glad to announce the release of *Percona XtraBackup* 2.4.15 on July 10, 2019. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

Bugs Fixed

- When the *encrypted tablespaces* feature was enabled, encrypted and compressed tables were not usable on the joiner node (Percona XtraDB Cluster) via SST (State Snapshot Transfer) with the `xtrabackup-v2` method. Bug fixed PXB-1867.
- `xbcloud` did not update date related fields of the HTTP header when retrying a request. Bug fixed PXB-1874.
- `xbcloud` did not retry to send the request after receiving the HTTP 408 error (request timeout). Bug fixed PXB-1875.
- If the user tried to merge an already prepared incremental backup, a misleading error was produced without informing that incremental backups may not be used twice. Bug fixed PXB-1862.
- `xbcloud` could crash with the Swift storage when project options were not included. Bug fixed PXB-1844.
- `xtrabackup` did not accept decimal fractions as values of the `innodb_max_dirty_pages_pct` option. Bug fixed PXB-1807.

Other bugs fixed: PXB-1850, PXB-1879, PXB-1887, PXB-1888, PXB-1890.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.17 Percona XtraBackup 2.4.14

Percona is glad to announce the release of *Percona XtraBackup* 2.4.14 on May 1, 2019. Downloads are available from our download site and from apt and yum repositories.

Percona XtraBackup 2.4.14 enables saving backups to an Amazon S3 storage when using *xbcloud*. The following example demonstrates how to use an Amazon S3 storage to make a full backup.

```
$ xtrabackup --backup --stream=xbstream --extra-lsndir=/tmp --target-dir=/tmp | \
xbcloud put --storage=s3 \
--s3-endpoint='s3.amazonaws.com' \
--s3-access-key='YOUR-ACCESSKEYID' \
--s3-secret-key='YOUR-SECRETACCESSKEY' \
--s3-bucket='mysql_backups'
--parallel=10 \
${date -I}-full_backup
```

All Percona software is open-source and free.

New Features

Amazon S3 is now supported in *xbcloud*. More information in PXB-1813.

Bugs Fixed

- When the row format was changed during the backup, *xtrabackup* could crash during the incremental prepare stage. Bug fixed PXB-1824.
- If compressed InnoDB undo tablespaces were not removed beforehand, the incremental backup could crash at the prepare stage. Bug fixed PXB-1552.

Other bugs fixed: PXB-1771, PXB-1809, PXB-1837.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

10.1.18 Percona XtraBackup 2.4.13

Percona is glad to announce the release of *Percona XtraBackup* 2.4.13 on January 18, 2019. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

Improvements and New Features

- PXB-1548: Percona Xtrabackup enables updating the `ib_buffer_pool` file with the latest pages present in the buffer pool by setting the `xtrabackup --dump-innodb-buffer-pool` option to ON. Thanks to Marcelo Altmann for contribution.

Bus Fixed

- `xtrabackup` did not delete missing tables from the partial backup which led to error messages logged by the server on startup. Bug fixed PXB-1536.
- The `--history` option did not work when autocommit was disabled. Bug fixed PXB-1569.
- `xtrabackup` could fail to backup encrypted tablespace when it was recently created or altered. Bug fixed PXB-1648.
- When the `--throttle` option was used, the applied value was different from the one specified by the user (off by one error). Bug fixed PXB-1668.
- It was not allowed for MTS (multi-threaded slaves) without GTID to be backed up with `--safe-slave-backup`. Bug fixed PXB-1672.
- Percona Xtrabackup could crash when the `ALTER TABLE … TRUNCATE PARTITION` command was run during a backup without locking DDL. Bug fixed PXB-1679.
- `xbcrypt` could display an assertion failure and generated core if the required parameters are missing. Bug fixed PXB-1683.
- Using `--lock-ddl-per-table` caused the server to scan all records of partitioned tables which could lead to the “out of memory” error. Bugs fixed PXB-1691 and PXB-1698.
- xtrabackup –prepare could hang while performing insert buffer merge. Bug fixed PXB-1704.
- Incremental backups did not update `xtrabackup_binlog_info` with `--binlog-info=lockless`. Bug fixed PXB-1711.

Other bugs fixed: PXB-1570, PXB-1609, PXB-1632

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.19 Percona XtraBackup 2.4.12

Percona is glad to announce the release of *Percona XtraBackup* 2.4.12 on June 22, 2018. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

New features and improvements

- *Percona XtraBackup* now prints used arguments to standard output. Bug fixed PXB-1494.

Bugs fixed

- `xtrabackup --copy-back` didn't read which encryption plugin to use from `plugin-load` setting of the `my.cnf` configuration file. Bug fixed PXB-1544.
- xbstream was exiting with zero return code when it failed to create one or more target files instead of returning error code 1. Bug fixed PXB-1542.
- Meeting a zero sized keyring file, *Percona XtraBackup* was removing and immediately recreating it, which could affect external software noticing this file had undergo manipulations. Bug fixed PXB-1540.
- `xtrabackup_checkpoints` files were encrypted during a backup, which caused additional difficulties to take incremental backups. Bug fixed PXB-202.

Other bugs fixed: PXB-1526 "Test kill_long_selects.sh failing with MySQL 5.7.21".

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.20 Percona XtraBackup 2.4.11

Percona is glad to announce the release of *Percona XtraBackup* 2.4.11 on April 23, 2018. Downloads are available from our download site and from apt and yum repositories.

All Percona software is open-source and free.

New features and improvements

- The support of the *Percona Server for MySQL* encrypted general tablespaces was implemented in this version of *Percona XtraBackup*. Issue fixed PXB-1513.
- *Percona XtraBackup* is now able to backup encrypted *Percona Server for MySQL* instances which are using keyring_vault plugin. Issue fixed PXB-1514.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.21 Percona XtraBackup 2.4.10

Percona is glad to announce the release of *Percona XtraBackup* 2.4.10 on March 30, 2018. Downloads are available from our download site and from apt and yum repositories.

This release is based on MySQL 5.7.19 and is the current GA (Generally Available) stable release in the 2.4 series.

Starting from now, *Percona XtraBackup* issue tracking system was moved from launchpad to JIRA. All Percona software is open-source and free.

Bugs fixed

- xbcrypt with `--encrypt-key-file` option was failing due to regression in *Percona XtraBackup* 2.4.9. Bug fixed PXB-518.
- Simultaneous usage of both `--lock-ddl` and `--lock-ddl-per-table` options caused *Percona XtraBackup* lock with the backup process never completed. Bug fixed PXB-792.
- Compilation under Mac OS X was broken. Bug fixed PXB-796.
- A regression of the maximum number of pending reads and the unnoticed earlier possibility of a pending reads related deadlock caused *Percona XtraBackup* to stuck in prepare stage. Bug fixed: PXB-1467.
- *Percona XtraBackup* skipped tablespaces with corrupted first page instead of aborting the backup. Bug fixed PXB-1497.

Other bugs fixed: PXB-513.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.22 Percona XtraBackup 2.4.9

Percona is glad to announce the release of *Percona XtraBackup* 2.4.9 on November 29th 2017. Downloads are available from our download site and from apt and yum repositories.

This release is the current GA (Generally Available) stable release in the 2.4 series.

New Features

- *Percona XtraBackup* packages are now available for *Ubuntu 17.10 (Artful)*.
- xbcrypt now has an ability to decrypt files in parallel by specifying the number of threads with the `xtrabackup --encrypt-threads` option.
- `xtrabackup --copy-back` option can now be used with `xtrabackup --parallel` option to copy the user data files in parallel (redo logs and system tablespaces are copied in the main thread).

Bugs fixed

- *Percona XtraBackup* would fail to backup large databases on 32-bit platforms. Bug fixed PXB-481.
- *Percona XtraBackup* failed to build with *GCC 7*. Bug fixed PXB-502.
- *Percona XtraBackup* would hang during the prepare phase if there was not enough room in log buffer to accommodate checkpoint information at the end of the crash recovery process. Bug fixed PXB-506.
- When backup was streamed in tar format with with the `xtrabackup --slave-info` option output file `xtrabackup_slave_info` did not contain the slave information. Bug fixed PXB-507.
- If xtrabackup –slave-info was used while backing up 5.7 instances, the master binary log coordinates were not properly displayed in the logs. Bug fixed PXB-508.
- `innobackupex --slave-info` would report a single `m` instead of `slave info` in the standard output. Bug fixed PXB-510.
- *Percona XtraBackup* would crash while preparing the 5.5 backup with `utf8_general50_ci` collation. Bug fixed PXB-748 (*Fungo Wang*).
- *Percona XtraBackup* would crash if `xtrabackup --throttle` was used while preparing backups. Fixed by making this option available only during the backup process. Bug fixed PXB-789.
- *Percona XtraBackup* could get stuck if backups are taken with `xtrabackup --safe-slave-backup` option, while there were long running queries. Bug fixed PXB-1039.

Other bugs fixed: PXB-250, PXB-511, and PXB-512.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.23 Percona XtraBackup 2.4.8

Percona is glad to announce the release of *Percona XtraBackup* 2.4.8 on July 24th 2017. Downloads are available from our download site and from apt and yum repositories.

This release is the current GA (Generally Available) stable release in the 2.4 series.

New Features

- To avoid issues with *MySQL* 5.7 skipping redo log for DDL *Percona XtraBackup* has implemented three new options (`xtrabackup --lock-ddl`, `xtrabackup --lock-ddl-timeout`, `xtrabackup --lock-ddl-per-table`) that can be used to place MDL locks on tables while they are copied.
- New `xtrabackup --check-privileges` option has been implemented that can be used to check if *Percona XtraBackup* has all required privileges to perform the backup.

Bugs fixed

- *xtrabackup* would hang with `Waiting for master thread to be suspended` message when backup was being prepared. Bug fixed PXB-499.
- *xtrabackup* would fail to prepare the backup with `6th page is not initialized` message in case server didn't properly initialize the page. Bug fixed PXB-500.
- xbstream could run out of file descriptors while extracting the backup which contains many tables. Bug fixed PXB-503
- When a table was created with the `DATA DIRECTORY` option *xtrabackup* would back up the `.frm` and `.isl` files, but not the `.ibd` file. Due to the missing `.ibd` files backup then could not be restored. Bug fixed PXB-504.
- *Percona XtraBackup* incorrectly determined use of `master_auto_postion` on a slave, and thus generated invalid xtrabackup_slave_info file. Bug fixed PXB-505.
- *Percona XtraBackup* will now print a warning if it encounters unsupported storage engine. Bug fixed PXB-713.
- *Percona XtraBackup* would crash while backing up MariaDB 10.2.x with `--ftwrl-\*` options. Bug fixed PXB-790.
- `xtrabackup --slave-info` didn't write the correct information into `xtrabackup_slave_info` file when multi-source replication was used. Bug fixed PXB-1022.
- Along with xtrabackup_checkpints file, *xtrabackup* now copies `xtrabackup_info` file into directory specified by `xtrabackup --extra-lsndir` option. Bug fixed PXB-1026.
- GTID position was not recorded when xtrabackup –binlog-info option was set to `AUTO`. Bug fixed PXB-1030.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.24 Percona XtraBackup 2.4.7-2

Percona is glad to announce the release of *Percona XtraBackup* 2.4.7-2 on May 29th 2017. Downloads are available from our download site and from apt and yum repositories.

This release is the current GA (Generally Available) stable release in the 2.4 series.

Bugs fixed

Fixed build failure on Debian 9.0 (*Stretch*). Bug fixed PXB-501.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.25 Percona XtraBackup 2.4.7

Percona is glad to announce the release of *Percona XtraBackup* 2.4.7 on April 17th 2017. Downloads are available from our download site and from apt and yum repositories.

This release is the current GA (Generally Available) stable release in the 2.4 series.

New Features

- *Percona XtraBackup* now uses hardware accelerated implementation of `crc32` where it is supported.
- *Percona XtraBackup* has implemented new options: `xtrabackup --tables-exclude` and `xtrabackup --databases-exclude` that work similar to `xtrabackup --tables` and `xtrabackup --databases` options, but exclude given names/paths from backup.
- The xbstream binary now supports parallel extraction with the `--parallel` option.
- The xbstream binary now supports following new options: `--decrypt`, `--encrypt-threads`, `--encrypt-key`, and `--encrypt-key-file`. When `--decrypt` option is specified xbstream will automatically decrypt encrypted files when extracting input stream. Either `--encrypt-key` or `--encrypt-key-file` options must be specified to provide encryption key, but not both. Option `--encrypt-threads` specifies the number of worker threads doing the encryption, default is `1`.

Bugs fixed

- Backups were missing *.isl files for general tablespace. Bug fixed PXB-494.
- In 5.7 *MySQL* changed default checksum algorithm to `crc32`, while *xtrabackup* was using `innodb`. This caused *xtrabackup* to perform extra checksum calculations which were not needed. Bug fixed PXB-495.
- For system tablespaces consisting of multiple files *xtrabackup* updated LSN only in first file. This caused *MySQL* versions lower than 5.7 to fail on startup. Bug fixed PXB-498.
- `xtrabackup --export` can now export tables that have more than 31 index. Bug fixed PXB-58.
- `Unrecognized character \x01; marked by <-- HERE` message could be seen if backups were taken with the version check enabled. Bug fixed PXB-944.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.26 Percona XtraBackup 2.4.6

Percona is glad to announce the release of *Percona XtraBackup* 2.4.6 on February 22nd 2017. Downloads are available from our download site and from apt and yum repositories.

This release is the GA (Generally Available) stable release in the 2.4 series.

New features

Percona XtraBackup has implemented new `xtrabackup --remove-original` option that can be used to remove the encrypted and compressed files once they've been decrypted/decompressed.

Bugs Fixed

- *xtrabackup* was using username set for server in a configuration file even if a different user was defined in the users configuration file. Bug fixed PXB-463.
- Incremental backups did not include `xtrabackup_binlog_info` and xtrabackup_galera_info files. Bug fixed PXB-489.
- In case a warning was written to stout instead of stderr during the streaming backup, it could cause assertion in the xbstream. Bug fixed PXB-491.
- `xtrabackup --move-back` did not always restore out-of-datadir tablespaces to their original directories. Bug fixed PXB-492.
- innobackupex and *xtrabackup* scripts were showing the password in the `ps` output when it was passed as a command line argument. Bug fixed PXB-585
- Incremental backup would fail with path like `~/backup/inc_1` because *xtrabackup* didn't properly expand tilde. Bug fixed PXB-775.
- Fixed missing dependency check for `perl(Digest::MD5)` in rpm packages. Bug fixed PXB-777.
- *Percona XtraBackup* now supports `-H`, `-h`, `-u` and `-p` shortcuts for `--hostname`, `--datadir`, `--user` and `--password` respectively. Bugs fixed PXB-947 and PXB-1032.
- **[UPDATE 2016-02-28]:** New packages have been pushed to repositories with incremented package version to address the bug PXB-497.

Other bugs fixed: PXB-945.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.27 Percona XtraBackup 2.4.5

Percona is glad to announce the release of *Percona XtraBackup* 2.4.5 on November 29th 2016. Downloads are available from our download site and from apt and yum repositories.

This release is the GA (Generally Available) stable release in the 2.4 series.

New features

- *Percona XtraBackup* now supports `SHA256` passwords. Using the `SHA256` algorithm requires either SSL encrypted connection, or using public key encryption for password exchange which is only available when both client and server are linked with OpenSSL.
- *Percona XtraBackup* now supports Command Options for Secure Connections.
- **NOTE:** Due to xbcrypt format changes, backups encrypted with this *Percona XtraBackup* version will not be recoverable by older versions.

Bugs Fixed

- *Percona XtraBackup* would crash while preparing the backup, during the shutdown, when master thread was performing checkpoint and purge thread was expecting that all other threads completed or were idle. Bug fixed PXB-483.
- Safe slave backup algorithm performed too short delays between retries which could cause backups to fail on a busy servers. Bug fixed PXB-484.
- *Percona XtraBackup* didn't check the logblock checksums. Bug fixed PXB-485.
- Fixed new compilation warnings with GCC 6. Bug fixed PXB-487.
- xbcrypt was not setting the Initialization Vector (IV) correctly (and thus is was not using an IV). This was causing the same ciphertext to be generated across different runs (for the same message/same key). The IV provides the extra randomness to ensure that the same ciphertext is not generated across runs. Bug fixed PXB-490.
- `target-dir` was no longer relative to current directory but to `datadir` instead. Bug fixed PXB-760.
- Backup would still succeed even if *xtrabackup* would fail to write the metadata. Bug fixed PXB-763.
- *xbcloud* now supports EMC ECS Swift API Authorization requests. Bugs fixed PXB-769 and PXB-770 (*Txomin Barturen*).
- Some older versions of *MySQL* did not bother to initialize page type field for pages which are not index pages (see upstream #76262 for more information). Having this page type uninitialized could cause *xtrabackup* to crash on prepare. Bug fixed PXB-772.
- *Percona XtraBackup* would fail to backup *MariaDB* 10.2 with the `unsupported server version` error message. Bug fixed PXB-1027.
- Fixed misleading error message about missing metadata. Bug fixed PXB-752.
- Backing up with an SSL user didn't work correctly. Bug fixed PXB-750.

Other bugs fixed: PXB-486, PXB-771, PXB-773, and PXB-774.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.28 Percona XtraBackup 2.4.4

Percona is glad to announce the release of *Percona XtraBackup* 2.4.4 on July 25th 2016. Downloads are available from our download site and from apt and yum repositories.

This release is the GA (Generally Available) stable release in the 2.4 series.

New features

Percona XtraBackup has been rebased on *MySQL* 5.7.13.

Bugs Fixed

- *Percona XtraBackup* reported the difference in the actual size of the system tablespace and the size which was stored in the tablespace header. This check is now skipped for tablespaces with autoextend support. Bug fixed PXB-462.
- Because *Percona Server for MySQL* 5.5 and *MySQL* 5.6 store the LSN offset for large log files at different places inside the redo log header, Percona Xtrabackup was trying to guess which offset is better to use by trying to read from each one and compare the log block numbers and assert `lsn_chosen == 1` when both LSNs looked correct, but they were different. Fixed by improving the server detection. Bug fixed PXB-473.
- *Percona XtraBackup* didn't correctly detect when tables were both compressed and encrypted. Bug fixed PXB-477.
- *Percona XtraBackup* would crash if the keyring file was empty. Bug fixed PXB-479.
- Backup couldn't be prepared when the size in cache didn't match the physical size. Bug fixed PXB-482.
- Free Software Foundation address in copyright notices was outdated. Bug fixed PXB-663.
- Backup process would fail if the `datadir` specified on the command-line was not the same as one that is reported by the server. *Percona XtraBackup* now allows the `datadir` from `my.cnf` override the one from `SHOW VARIABLES`. *xtrabackup* will print a warning that they don't match, but continue. Bug fixed PXB-741.
- With upstream change of maximum page size from 16K to 64K, the size of incremental buffer became 1G. Which increased the requirement to 1G of RAM in order to prepare the backup. While in fact there is no need to allocate such a large buffer for smaller pages. Bug fixed PXB-753.
- Backup process would fail on *MariaDB* Galera cluster operating in GTID mode if binary logs were in non-standard directory. Bug fixed PXB-936.

Other bugs fixed: PXB-755, PXB-756, and PXB-759.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.29 Percona XtraBackup 2.4.3

Percona is glad to announce the release of *Percona XtraBackup* 2.4.3 on May 23rd 2016. Downloads are available from our download site and from apt and yum repositories.

This release is the GA (Generally Available) stable release in the 2.4 series.

New features

Percona XtraBackup has implemented new `xtrabackup --reencrypt-for-server-id` option. Using this option allows users to start the server instance with different `server_id` from the one the encrypted backup was taken from, like a replication slave or a galera node. When this option is used, *xtrabackup* will, as a prepare step, generate a new master key with ID based on the new `server_id`, store it into keyring file and re-encrypt the tablespace keys inside of tablespace headers.

Bugs Fixed

- Running DDL statements on *Percona Server for MySQL* 5.7 during the backup process could in some cases lead to failure while preparing the backup. Bug fixed PXB-247.
- *MySQL* 5.7 can sometimes skip redo logging when creating an index. If such `ALTER TABLE` is being issued during the backup, the backup would be inconsistent. *xtrabackup* will now abort with error message if such `ALTER`
  `TABLE` has been done during the backup. Bug fixed PXB-249.
- `.ibd` files for remote tablespaces were not copied back to original location pointed by the .isl files. Bug fixed PXB-466.
- When called with insufficient parameters, like specifying the empty `xtrabackup --defaults-file` option, *Percona XtraBackup* could crash. Bug fixed PXB-471.
- Documentation states that the default value for `xtrabackup --ftwrl-wait-query-type` is `all`, however it was `update`. Changed the default value to reflect the documentation. Bug fixed PXB-472.
- When `xtrabackup --keyring-file-data` option was specified, but no keyring file was found, *xtrabackup* would create an empty one instead of reporting an error. Bug fixed PXB-476.
- If `ALTER INSTANCE ROTATE INNODB MASTER KEY` was run at same time when `xtrabackup --backup` was bootstrapping it could catch a moment when the key was not written into the keyring file yet and *xtrabackup* would overwrite the keyring with the old copy of a keyring, so the new key would be lost. Bug fixed PXB-478.
- Output of `xtrabackup --slave-info` option was missing an apostrophe. Bug fixed PXB-940.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.30 Percona XtraBackup 2.4.2

Percona is glad to announce the release of *Percona XtraBackup* 2.4.2 on April 1st 2016. Downloads are available from our download site and from apt and yum repositories.

This release is the GA (Generally Available) stable release in the 2.4 series.

New features

- *Percona XtraBackup* has implemented support for InnoDB tablespace encryption.
- *Percona XtraBackup* has been rebased on *MySQL* 5.7.11.

Bugs Fixed

- When backup was taken on *MariaDB* 10 with GTID enabled, *Percona XtraBackup* didn't store `gtid_slave_pos` in `xtrabackup_slave_info` but logged it only to `STDERR`. Bug fixed PXB-715.
- Backup process would fail if `xtrabackup --throttle` option was used. Bug fixed PXB-465.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.1.31 Percona XtraBackup 2.4.1

Percona is glad to announce the release of *Percona XtraBackup* 2.4.1 on February 16th 2016. Downloads are available from our download site and from apt and yum repositories.

This release is the first GA (Generally Available) stable release in the 2.4 series.

This release contains all the features and bug fixes in Percona XtraBackup 2.3.3, plus the following:

New features

Percona XtraBackup has implemented basic support for *MySQL* 5.7 and *Percona Server for MySQL* 5.7.

Bugs Fixed

- *Percona XtraBackup* didn't respect `innodb_log_file_size` variable stored in `backup-my.cnf`. Bug fixed PXB-450.
- If server would run out of space while backups were taken with innobackupex –rsync option backup process would fail but innobackupex would still complete with `completed OK!` message. Bug fixed PXB-459.
- *Percona XtraBackup* was silently skipping extra arguments. Bug fixed PXB-747 (*Fungo Wang*).

Other bugs fixed: PXB-1368 and 1363.

CONTACT US

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-24

10.2 The xtrabackup Option Reference

This page documents all of the command-line options for the `xtrabackup` binary.

10.2.1 Options

--apply-log-only

This option causes only the redo stage to be performed when preparing a backup. It is very important for incremental backups.

--backup

Make a backup and place it in `xtrabackup --target-dir`. See Creating a backup.

--backup-lock-retry-count=

The number of attempts to acquire metadata locks.

--backup-lock-timeout=

The timeout in seconds for attempts to acquire metadata locks.

--binlog-info

This option controls how *Percona XtraBackup* should retrieve the server's binary log coordinates corresponding to the backup. Possible values are `OFF`, `ON`, `LOCKLESS` and `AUTO`. See the *Percona XtraBackup* Lockless binary log information manual page for more information.

--check-privileges

This option checks if *Percona XtraBackup* has all the required privileges. If a missing privilege is required for the current operation, it will terminate and print out an error message. If a missing privilege is not required for the current operation but may be necessary for some other XtraBackup operation, the process is not aborted, and a warning is printed.

```
xtrabackup: Error: missing required privilege LOCK TABLES on *.*
xtrabackup: Warning: missing required privilege REPLICATION CLIENT on *.*
```

--close-files

Do not keep files open. When *xtrabackup* opens a tablespace, it normally does not close its file handle to manage the DDL operations correctly. However, if the number of tablespaces is huge and can not fit into any limit, there is an option to close file handles once they are no longer accessed. *Percona XtraBackup* can produce inconsistent backups with this option enabled. Use the option at your own risk.

--compact

Create a compact backup by skipping secondary index pages.

--compress

This option tells *xtrabackup* to compress all output data, including the transaction log file and metadata files, using the specified compression algorithm. The only currently supported algorithm is `quicklz`. The resulting files have the qpress archive format.

Every `\*.qp` file produced by xtrabackup is essentially a one-file qpress archive and can be extracted and uncompressed by the qpress file archiver.

--compress-chunk-size=

Size of working buffer(s) for compression threads in bytes. The default value is 64K.

--compress-threads=

This option specifies the number of worker threads used by *xtrabackup* for parallel data compression. This option defaults to `1`. Parallel compression (`xtrabackup --compress-threads`) can be used together with parallel file copying (`xtrabackup --parallel`). For example, `--parallel=4 --compress --compress-threads=2` will create 4 I/O threads that will read the data and pipe it to 2 compression threads.

--copy-back

Copy all the files in a previously made backup from the backup directory to their original locations. This option will not copy over existing files unless `xtrabackup --force-non-empty-directories` option is specified.

--core-file

Write core on fatal signals.

--databases=

This option specifies the list of databases and tables that should be backed up. The option accepts the list of the form `"databasename1[.table_name1] databasename2[.table_name2] . . ."`.

--databases-exclude=name

Excluding databases based on name. This option operates the same way as `xtrabackup --databases`, but matched names are excluded from the backup. Note that this option has a higher priority than `xtrabackup --databases`.

--databases-file=

This option specifies the path to the file containing the list of databases and tables that should be backed up. The file can contain the list elements of the form `databasename1[.table_name1]`, one element per line.

--datadir=DIRECTORY

The source directory for the backup. This directory should be the same as the datadir for your *MySQL* server, and it should be read from my.cnf if that exists; otherwise, you must specify it on the command line.

When combined with the `xtrabackup --copy-back` or `xtrabackup --move-back` option, `xtrabackup --datadir` refers to the destination directory.

Once connected to the server, to perform a backup, you will need `READ` and `EXECUTE` permissions at a filesystem level in the server's datadir.

--debug-sleep-before-unlock=

A debug-only option that is used by the Xtrabackup test suite.

--decompress

This option decompresses all files with the `.qp` extension in a backup previously made with the `xtrabackup --compress` option. The `xtrabackup --parallel` option will allow multiple files to be decrypted simultaneously. To decompress, the qpress utility MUST be installed and accessible within the path. *Percona XtraBackup* doesn't automatically remove the compressed files. To clean up the backup directory, users should use the `xtrabackup --remove-original` option.

--decrypt=ENCRYPTION-ALGORITHM

Decrypts all files with the .xbcrypt extension in a backup previously made with `xtrabackup --encrypt` option. The `xtrabackup --parallel` option will allow multiple files to be decrypted simultaneously. *Percona XtraBackup* doesn't automatically remove the encrypted files. To clean up the backup directory, users should use `xtrabackup --remove-original` option.

--defaults-extra-file=[MY.CNF]

Read this file after the global files are read. This file must be the first option on the command-line.

--defaults-file=[MY.CNF]

Only read default options from the given file. This file must be the first option on the command-line and be a real file and cannot be a symbolic link.

--defaults-group=GROUP-NAME

This option sets up the group which should be read from the configuration file. The option is used by the `--default-group` and is required for

`mysqld_multi` deployments.

--defaults-group-suffix=

Read the usual options groups and also groups with concat(group, suffix).

--dump-innodb-buffer-pool

This option controls whether or not a new dump of buffer pool content should be done.

With `--dump-innodb-buffer-pool`, *xtrabackup* makes a request to the server to start the buffer pool dump (it takes some time to complete and is done in background) at the beginning of a backup provided the status variable `innodb_buffer_pool_dump_status` reports that the dump has been completed.

```
$ xtrabackup --backup --dump-innodb-buffer-pool --target-dir=/home/user/backup
```

By default, this option is set to OFF.

If `innodb_buffer_pool_dump_status` reports that there is running dump of the buffer pool, *xtrabackup* waits for the dump to complete using the value of –dump-innodb-buffer-pool-timeout

The file `ib_buffer_pool` stores tablespace ID and page ID data used to warm up the buffer pool sooner.

--dump-innodb-buffer-pool-timeout

This option contains the number of seconds that *xtrabackup* should monitor the value of `innodb_buffer_pool_dump_status` to determine if buffer pool dump has completed.

This option is used in combination with `--dump-innodb-buffer-pool`. By default, it is set to 10 seconds.

--dump-innodb-buffer-pool-pct

This option contains the percentage of the most recently used buffer pool pages to dump.

This option is effective if –dump-innodb-buffer-pool option is set to ON. If this option contains a value, *xtrabackup* sets the *MySQL* system variable `innodb_buffer_pool_dump_pct`. As soon as the buffer pool dump completes or it is stopped (see –dump-innodb-buffer-pool-timeout), the value of the *MySQL* system variable is restored.

--encrypt=ENCRYPTION_ALGORITHM

This option instructs xtrabackup to encrypt backup copies of InnoDB data files using the algorithm specified in the ENCRYPTION_ALGORITHM. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

--encrypt-key=ENCRYPTION_KEY

This option instructs xtrabackup to use the given `ENCRYPTION_KEY` when using the `xtrabackup --encrypt` option. It is passed directly to the xtrabackup child process. See the xtrabackup documentation for more details.

--encrypt-key-file=ENCRYPTION_KEY_FILE

This option instructs xtrabackup to use the encryption key stored in the given `ENCRYPTION_KEY_FILE` when using the `xtrabackup --encrypt` option. It is passed directly to the xtrabackup child process. See the xtrabackup documentation for more details.

--encrypt-threads=

This option specifies the number of worker threads that will be used for parallel encryption/decryption. See the xtrabackup documentation for more details.

--encrypt-chunk-size=

This option specifies the size of the internal working buffer for each encryption thread, and is measured in bytes. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

> **Note**
>
> To adjust the xbcloud/xbstream chunk size when you use encryption, you must adjust both the –encrypt-chunk-size and –read-buffer-size variables.

--export

Create files necessary for exporting tables. See Restoring Individual Tables.

--extra-lsndir=DIRECTORY

(for –backup): save an extra copy of the `xtrabackup_checkpoints` and `xtrabackup_info` files in this directory.

--force-non-empty-directories

When specified, it makes :option`xtrabackup –copy-back` and `xtrabackup --move-back` option transfer files to non-empty directories. No existing files will be overwritten. If files that need to be copied/moved from the backup directory already exist in the destination directory, it will still fail with an error.

--ftwrl-wait-timeout=SECONDS

This option specifies time in seconds that xtrabackup should wait for queries that would block `FLUSH TABLES WITH READ LOCK` before running it. If there are still such queries when the timeout expires, xtrabackup terminates with an error. The default is `0`, in which case it does not wait for queries to complete and starts `FLUSH TABLES WITH READ LOCK` immediately. Where supported (Percona Server 5.6+) xtrabackup will automatically use Backup Locks as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.

--ftwrl-wait-threshold=SECONDS

This option specifies the query run time threshold which is used by xtrabackup to detect long-running queries with a non-zero value of xtrabackup –ftwrl-wait-timeout. `FLUSH TABLES WITH READ LOCK` is not started until such long-running queries exist. This option has no effect if xtrabackup –ftwrl-wait-timeout is `0`. The default value is `60` seconds. Where supported (Percona Server 5.6+) xtrabackup will automatically use Backup Locks as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.

--ftwrl-wait-query-type=all|update

This option specifies which types of queries are allowed to complete before xtrabackup will issue the global lock. The default is `all`.

--galera-info

This options creates the `xtrabackup_galera_info` file which contains the local node state at the time of the backup. Option should be used when performing the backup of Percona XtraDB Cluster. It has no effect when backup locks are used to create the backup.

--generate-new-master-key

Generates a new master key when doing a copy-back operation.

--history=name

This option enables the tracking of the backup history in the `PERCONA_SCHEMA.xtrabackup_history` table. An optional history series name may be specified that will be placed with the history record for the backup being taken.

--incremental

This option tells xtrabackup to create an incremental backup. It is passed to the xtrabackup child process. When this option is specified, either `xtrabackup --incremental-lsn` or xtrabackup –incremental-basedir can also be given. If neither option is given, option `xtrabackup --incremental-basedir` is passed to xtrabackup by default, set to the first timestamped backup directory in the backup base directory.

--incremental-basedir=DIRECTORY

This directory contains the full backup, which is the base dataset used for the incremental backups.

--incremental-dir=DIRECTORY

When preparing an incremental backup, this is the directory where the incremental backup is combined with the full backup to make a new full backup.

--incremental-force-scan

When creating an incremental backup, force a full scan of the data pages in the instance to be used in the backup even if the complete changed page bitmap data is available.

--incremental-history-name=name

This option specifies the name of the backup series stored in the PERCONA_SCHEMA.xtrabackup_history history record to base an incremental backup on. *xtrabackup* searches the history table for the most recent (highest innodb_to_lsn), successful backup in the series and take the to_lsn value to use as the starting lsn for the incremental backup. This will be mutually exclusive with `xtrabackup --incremental-history-uuid`, `xtrabackup --incremental-basedir` and `xtrabackup --incremental-lsn`. If no valid `LSN` can be found (no series by that name, no successful backups by that name) *xtrabackup* returns an error. It is used with the `xtrabackup --incremental` option.

--innodb-checksum-algorithm=name

The algorithm InnoDB uses to calculate a page checksum. The available algorithms are CRC32, INNODB, NONE, STRICT_CRC32, STRICT_INNODB, and STRICT_NONE

--incremental-history-uuid=UUID

This option specifies the `UUID` of the specific history record stored in the PERCONA_SCHEMA.xtrabackup_history to base an incremental backup on `xtrabackup --incremental-history-name`, `xtrabackup --incremental-basedir` and `xtrabackup --incremental-lsn`. If no valid LSN is found (no success record with that UUID) *xtrabackup* returns an error. This option is used with the `xtrabackup --incremental` option.

--incremental-lsn=LSN

When creating an incremental backup, you can specify the log sequence number (`LSN`) instead of specifying `xtrabackup --incremental-basedir`. For databases created in 5.1 and later, specify the LSN as a single 64-bit integer. **ATTENTION**: If a wrong LSN value is specified (a user error that *Percona XtraBackup* cannot detect), the backup will be unusable. Be careful!

--innodb-log-arch-dir=DIRECTORY

This option is used to specify the directory containing the archived logs. It can only be used with the `xtrabackup --prepare` option.

--innodb-miscellaneous

A large group of InnoDB options are normally read from the `my.cnf` configuration file, so that *xtrabackup* boots up its embedded InnoDB in the same configuration as your current server. You normally do not need to specify these explicitly. These options have the same behavior that they have in InnoDB or XtraDB. They are as follows:

```
--innodb-adaptive-hash-index
--innodb-additional-mem-pool-size
--innodb-autoextend-increment
--innodb-buffer-pool-size
--innodb-checksums
--innodb-data-file-path
--innodb-data-home-dir
--innodb-doublewrite-file
--innodb-doublewrite
--innodb-extra-undoslots
--innodb-fast-checksum
--innodb-file-io-threads
--innodb-file-per-table
--innodb-flush-log-at-trx-commit
--innodb-flush-method
--innodb-force-recovery
--innodb-io-capacity
--innodb-lock-wait-timeout
--innodb-log-buffer-size
--innodb-log-files-in-group
--innodb-log-file-size
--innodb-log-group-home-dir
--innodb-max-dirty-pages-pct
--innodb-open-files
--innodb-page-size
--innodb-read-io-threads
--innodb-write-io-threads
```

--innodb-undo-directory=name

The directory location for the undo tablespace. The path is absolute.

--innodb-undo-tablespace=

The number of undo tablespaces to use.

--keyring-file-data=FILENAME

The path to the keyring file. Combine this option with `xtrabackup --xtrabackup-plugin-dir`.

--kill-long-queries-timeout=

This options specifies the number of seconds xtrabackup waits between starting FLUSH TABLES WITH READ LOCK and killing those queries that block it. The default is `0` (zero) seconds, which means the xtrabackup does not attempt to kill any queries.

--kill-long-query-type=select|all

This option specifies which types of queries should be killed to unblock the global lock. The default value is `select`.

--lock-ddl

Issue `LOCK TABLES FOR BACKUP` if it is supported by server at the beginning of the backup to block all DDL operations.

--lock-ddl-per-table

Lock DDL for each table before xtrabackup starts to copy it and until the backup is completed.

--lock-ddl-timeout

If `LOCK TABLES FOR BACKUP` does not return within given timeout, abort the backup.

--log-bin[=name]

The base name for the log sequence.

--log-copy-interval=

This option specifies time interval between log copying thread checks in milliseconds (default is 1 second).

--login-path=

Read this path from the login file.

--move-back

Move all the files in a previously made backup from the backup directory to their original locations. As this option removes backup files, it must be used with caution.

--no-backup-locks

This options controls if backup locks are used instead of `FLUSH TABLES WITH READ LOCK` during the backup stage. The backup locks are must be supported on the server for the option to have an affect.

This option is enabled by default. Disable the option with `--no-backup-locks`.

--no-defaults

Do not read the default options from any option file. Must be given as the first option on the command-line.

--no-lock

This option automatically uses Backup Locks, and disables table locks, as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.

Only use this option if *all* tables are InnoDB and you *do not care* about the binary log position of the backup.

Do not use this option if any DDL statements will be executed or if any non-InnoDB tables are being updated (this includes the MyISAM tables in the mysql database). Using this option in these conditions could cause an inconsistent backup.

If your backups fail to acquire a lock and you are planning to use this option, the failure may be caused by incoming replication events that prevent the lock from succeeding. Try the `--safe-slave-backup` to momentarily stop the replication slave thread.

The `xtrabackup-binlog-info` is not created when the `--no-lock` is used because `SHOW MASTER STATUS` may be inconsistent. In certain conditions, `xtrabackup_binlog_pos_innodb` can be used instead to get consistent binlog coordinates as described in Working with Binary Logs.

--no-version-check

This option disables the version check. If you do not pass this option, the automatic version check is enabled implicitly when xtrabackup runs in the `--backup` mode. To disable the version check, explicitly pass the `--no-version-check` option when invoking xtrabackup.

When the automatic version check is enabled,xtrabackup performs a version check against the server on the backup stage after creating a server connection. xtrabackup sends the following information to the server:

- MySQL flavour and version
- Operating system name
- Percona Toolkit version
- Perl version

Each piece of information has a unique identifier which is an MD5 hash value that Percona Toolkit uses to obtain statistics about how it is used. This value is a random UUID; no client information is either collected or stored.

--open-files-limit=

The maximum number of file descriptors to reserve with setrlimit().

--parallel=

This option specifies the number of threads to use to copy multiple data files concurrently when creating a backup. The default value is 1 (i.e., no concurrent transfer). In *Percona XtraBackup* 2.3.10 and newer, this option can be used with `xtrabackup --copy-back` option to copy the user data files in parallel (redo logs and system tablespaces are copied in the main thread).

--password=PASSWORD

This option specifies the password to use when connecting to the database. It accepts a string argument. See `mysql --help` for details.

--prepare

Makes `xtrabackup` perform recovery on a backup created with `xtrabackup --backup`, so that it is ready to use. See preparing a backup.

--print-defaults

Print the program argument list and exit. Must be given as the first option on the command-line.

--print-param

Makes `xtrabackup` print out parameters that to copy the data files back to their original locations to restore them. See Scripting Backups With xtrabackup.

--read-buffer-size[=#]

Set read buffer size. The given value is scaled up to page size. The default is 10MB.

Use this variable to increase the xbcloud/xbstream chunk size from the default value of 10MB.

NOTE: When you use encryption, to adjust the xbcloud/xbstream chunk size, adjust both the `--encrypt-chunk-size` and `--read-buffer-size` variables.

```
$ xtrabackup ... --read-buffer-size=1G | xbcloud put ...
```

--rebuild-indexes

Rebuild secondary indexes in InnoDB tables after applying the log. Only use with –prepare.

--rebuild-threads=

This option defines the number of threads to rebuild indexes in a compact backup. Only use with `--prepare` and `--rebuild-indexes`.

--redo-log-version=

The redo log version of the backup. Use only with `--prepare`.

--reencrypt-for-server-id=

Use this option to start the server instance with different server_id from the one the encrypted backup was taken from, like a replication replica or a Galera node. When this option is used, xtrabackup will, as a prepare step, generate a new master key with ID based on the new server_id, store it into keyring file, and re-encrypt the tablespace keys inside of tablespace headers. The option should be passed for `--prepare` (final step).

--remove-original

Implemented in *Percona XtraBackup* 2.4.6, this option when specified will remove `.qp`, `.xbcrypt` and `.qp.xbcrypt` files after decryption and decompression.

--rsync

Use the `rsync` utility to optimize local file transfers.

When this option is specified, xtrabackup uses `rsync` to copy all non-InnoDB files instead of spawning a separate copy command for each file. This option is faster for servers with a large number of databases or tables.

This option cannot be used with `--stream`.

--safe-slave-backup

When specified, xtrabackup will stop the replica SQL thread just before running `FLUSH TABLES WITH READ LOCK` and wait to start backup until `Slave_open_temp_tables` in `SHOW STATUS` is zero. If there are no open temporary tables, the backup will occur; otherwise the SQL thread will be started and stopped until there are no open temporary tables. The backup will fail if `Slave_open_temp_tables` does not become zero after xtrabackup –safe-slave-backup-timeout seconds. The replica SQL thread will be restarted when the backup finishes. This option is implemented to deal with replicating temporary tables and isn't neccessary with Row-Based-Replication.

--safe-slave-backup-timeout=SECONDS

How many seconds `xtrabackup --safe-slave-backup` should wait for `Slave_open_temp_tables` to become zero. The default is 300 seconds.

--secure-auth

Refuse client connecting to the server if it uses old (pre-4.1.1) protocol. (Enabled by default; use –skip-secure-auth to disable.)

--server-id=

The server instance being backed up.

--server-public-key-path=name

File path the server's public RSA key in PEM format.

--skip-tables-compatibility-check

This option disables the engine compatibility warning.

> **See also**
>
> `--tables-compatibility-check`

--slave-info

This option is useful when backing up a replication replica server. It prints the binary log position of the source server. It also writes the binary log coordinates to the `xtrabackup_slave_info` file as a `CHANGE MASTER` command. A new replica for this source can be set up by starting a replica server on this backup and issuing a `CHANGE MASTER` command with the binary log position saved in the xtrabackup_slave_info file.

--ssl

Enable secure connection. More information can be found in –ssl MySQL server documentation.

--ssl-ca

Path of the file, which contains a list of trusted SSL CAs. More information can be found in –ssl-ca MySQL server documentation.

--ssl-capath

The directory path that contains trusted SSL CA certificates in the PEM format. More information can be found in –ssl-capath MySQL server documentation.

--ssl-cert

Path of the file which contains X509 certificate in PEM format. More information can be found in –ssl-cert MySQL server documentation.

--ssl-cipher

List of permitted ciphers to use for connection encryption. More information can be found in –ssl-cipher MySQL server documentation.

--ssl-crl

Path of the file that contains certificate revocation lists. More information can be found in –ssl-crl MySQL server documentation.

--ssl-crlpath

Path of the directory that contains certificate revocation list files. More information can be found in –ssl-crlpath MySQL server documentation.

--ssl-key

Path of the file that contains X509 key in PEM format. More information can be found in –ssl-key MySQL server documentation.

--ssl-mode

The security state of connection to server. More information can be found in –ssl-mode MySQL server documentation.

--ssl-verify-server-cert

Verify server certificate Common Name value against host name used when connecting to server. More information can be found in –ssl-verify-server-cert MySQL server documentation.

--stats

Causes xtrabackup to scan the specified data files and print out index statistics.

--stream=name

Stream all backup files to the standard output in the specified format. Currently supported formats are `xbstream` and `tar`.

--tables=name

A regular expression against which the full tablename, in `databasename.tablename` format, is matched. If the name matches, the table is backed up. See partial backups.

--tables-compatibility-check

This option enables the engine compatibility warning.

The default value is `ON`. Use `--skip-tables-compatibility-check` to disable.

--tables-exclude=name

Filtering by regexp for table names. Operates the same way as `xtrabackup --tables`, but matched names are excluded from backup. Note that this option has a higher priority than `xtrabackup --tables`.

--tables-file=name

A file containing one table name per line, in databasename.tablename format. The backup will be limited to the specified tables. See Scripting Backups With xtrabackup.

--target-dir=DIRECTORY

This option specifies the destination directory for the backup. If the directory does not exist, `xtrabackup` creates it. If the directory does exist and is empty, `xtrabackup` will succeed. `xtrabackup` will not overwrite existing files; however it will fail with operating system error 17, `file exists`.

If this option is a relative path, it is interpreted as being relative to the current working directory from which xtrabackup is executed.

In order to perform a backup, you need `READ`, `WRITE`, and `EXECUTE` permissions at a filesystem level for the directory that you supply as the value of `--target-dir`.

--throttle=

This option limits the number of chunks copied per second. The chunk size is *10 MB*.

To limit the bandwidth to *10 MB/s*, set the option to *1*: `--throttle=1`.

--tls-version=name

The TLS version to use. The allowed values are the following:

- TLSv1
- TLSv1.1
- TLSv1.2

--tmpdir=name

This option is currently not used for anything except printing out the correct tmpdir parameter when `xtrabackup --print-param` is used.

--to-archived-lsn=LSN

This option is used to specify the LSN to which the logs should be applied when backups are being prepared. It can only be used with the `xtrabackup --prepare` option.

--transition-key

This option is used to enable processing the backup without accessing the keyring vault server. In this case, xtrabackup derives the AES encryption key from the specified passphrase and uses it to encrypt tablespace keys of tablespaces being backed up.

If `--transition-key <xtrabackup –transition-key>` does not have any value, xtrabackup will ask for it. The same passphrase should be specified for the `xtrabackup --prepare` command.

--use-memory=

This option affects how much memory is allocated and is similar to `innodb_buffer_pool_size`. This option is only relevant in the `--prepare` phase or when analyzing statistics with `--stats`. The default value is 100MB.

The recommended value is between 1GB to 2GB. Multiples are supported providing the unit (for example, 1MB, 1M, 1GB, 1G).

--user=USERNAME

This option specifies the MySQL username used when connecting to the server, if that’s not the current user. The option accepts a string argument. See mysql –help for details.

--version

This option prints *xtrabackup* version and exits.

--xtrabackup-plugin-dir=DIRNAME

The absolute path to the directory that contains the `keyring` plugin.

> **See also**
>
> Percona Server for MySQL Documentation: keyring_vault plugin with Data at Rest Encryption https://www.percona.com/doc/percona-server/5.7/security/data-at-rest-encryption.html MySQL Documentation: Using the keyring_file File-Based Plugin https://dev.mysql.com/doc/refman/5.7/en/keyring-file-plugin.html

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-12-06

10.3 The innobackupex Option Reference

This page documents all the command-line options for the `innobackupex`.

10.3.1 Options

--apply-log

Prepare a backup in `BACKUP-DIR` by applying the transaction log file named `xtrabackup_logfile` located in the same directory. Also, create new transaction logs. The InnoDB configuration is read from the file `backup-my.cnf` created by *innobackupex* when the backup was made. `innobackupex --apply-log` uses InnoDB configuration from `backup-my.cnf` by default, or from `--defaults-file`, if specified. InnoDB configuration in this context means server variables that affect data format, i.e. `innodb_page_size`, `innodb_log_block_size`, etc. Location-related variables, like `innodb_log_group_home_dir` or `innodb_data_file_path` are always ignored by `--apply-log`, so preparing a backup always works with data files from the backup directory, rather than any external ones.

--backup-locks

This option controls if backup locks should be used instead of `FLUSH TABLES WITH READ LOCK` on the backup stage. The option has no effect when backup locks are not supported by the server. This option is enabled by default, disable with `--no-backup-locks`.

--no-backup-locks

Explicity disables the option `–-backup-locks` which is enabled by default.

--close-files

Do not keep files opened. This option is passed directly to xtrabackup. When xtrabackup opens tablespace it normally doesn’t close its file handle in order to handle the DDL operations correctly. However, if the number of tablespaces is really huge and can not fit into any limit, there is an option to close file handles once they are no longer accessed. *Percona XtraBackup* can produce inconsistent backups with this option enabled. Use at your own risk.

--compress

This option instructs xtrabackup to compress backup copies of InnoDB data files. It is passed directly to the xtrabackup child process. See the xtrabackup documentation for details.

--compress-threads=

This option specifies the number of worker threads that will be used for parallel compression. It is passed directly to the xtrabackup child process. See the xtrabackup documentation for details.

--compress-chunk-size=

This option specifies the size of the internal working buffer for each compression thread, measured in bytes. It is passed directly to the xtrabackup child process. The default value is 64K. See the `xtrabackup` documentation for details.

--copy-back

Copy all the files in a previously made backup from the backup directory to their original locations. *Percona XtraBackup* `innobackupex --copy-back` option will not copy over existing files unless `innobackupex --force-non-empty-directories` option is specified.

--databases=LIST

This option specifies the list of databases that *innobackupex* should back up. The option accepts a string argument or path to file that contains the list of databases to back up. The list is of the form "databasename1[.table_name1] databasename2[.table_name2] …". If this option is not specified, all databases containing *MyISAM* and *InnoDB* tables will be backed up. Please make sure that --databases contains all of the *InnoDB* databases and tables, so that all of the innodb.frm files are also backed up. In case the list is very long, this can be specified in a file, and the full path of the file can be specified instead of the list. (See option --tables-file.)

--decompress

Decompresses all files with the .qp extension in a backup previously made with the `innobackupex --compress` option. The `innobackupex --parallel` option will allow multiple files to be decrypted and/or decompressed simultaneously. In order to decompress, the qpress utility MUST be installed and accessible within the path. *Percona XtraBackup* doesn't automatically remove the compressed files. In order to clean up the backup directory users should remove the `\*.qp` files manually.

--decrypt=ENCRYPTION-ALGORITHM

Decrypts all files with the .xbcrypt extension in a backup previously made with --encrypt option. The `innobackupex --parallel` option will allow multiple files to be decrypted and/or decompressed simultaneously.

--defaults-file=[MY.CNF]

This option accepts a string argument that specifies what file to read the default MySQL options from. Must be given as the first option on the command-line.

--defaults-extra-file=[MY.CNF]

This option specifies what extra file to read the default *MySQL* options from before the standard defaults-file. Must be given as the first option on the command-line.

--defaults-group=GROUP-NAME

This option accepts a string argument that specifies the group which should be read from the configuration file. This is needed if you use mysqld_multi. This can also be used to indicate groups other than mysqld and xtrabackup.

--encrypt=ENCRYPTION_ALGORITHM

This option instructs xtrabackup to encrypt backup copies of InnoDB data files using the algorithm specified in the ENCRYPTION_ALGORITHM. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

Currently, the following algorithms are supported: `AES128`, `AES192` and `AES256`.

--encrypt-key=ENCRYPTION_KEY

This option instructs xtrabackup to use the given proper length encryption key as the ENCRYPTION_KEY when using the --encrypt option. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

It is not recommended to use this option where there is uncontrolled access to the machine as the command line and thus the key can be viewed as part of the process info.

--encrypt-key-file=ENCRYPTION_KEY_FILE

This option instructs xtrabackup to use the encryption key stored in the given ENCRYPTION_KEY_FILE when using the –encrypt option. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

The file must be a simple binary (or text) file that contains exactly the key to be used.

--encrypt-threads=

This option specifies the number of worker threads that will be used for parallel encryption. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

--encrypt-chunk-size=

This option specifies the size of the internal working buffer for each encryption thread, measured in bytes. It is passed directly to the xtrabackup child process. See the `xtrabackup` documentation for more details.

--export

This option is passed directly to `xtrabackup --export` option. It enables exporting individual tables for import into another server. See the *xtrabackup* documentation for details.

--extra-lsndir=DIRECTORY

This option accepts a string argument that specifies the directory in which to save an extra copy of the `xtrabackup_checkpoints` file. It is passed directly to *xtrabackup*'s `innobackupex --extra-lsndir` option. See the xtrabackup documentation for details.

--force-non-empty-directories

When specified, it makes `innobackupex --copy-back` option or `innobackupex --move-back` option transfer files to non-empty directories. No existing files will be overwritten. If --copy-back or --move-back has to copy a file from the backup directory which already exists in the destination directory, it will still fail with an error.

--galera-info

This options creates the `xtrabackup_galera_info` file which contains the local node state at the time of the backup. Option should be used when performing the backup of Percona-XtraDB-Cluster. Has no effect when backup locks are used to create the backup.

--help

This option displays a help screen and exits.

--history=NAME

This option enables the tracking of backup history in the `PERCONA_SCHEMA.xtrabackup_history` table. An optional history series name may be specified that will be placed with the history record for the current backup being taken.

--host=HOST

This option accepts a string argument that specifies the host to use when connecting to the database server with TCP/IP. It is passed to the mysql child process without alteration. See `mysql --help` for details.

--ibbackup=IBBACKUP-BINARY

This option specifies which *xtrabackup* binary should be used. The option accepts a string argument. IBBACKUP-BINARY should be the command used to run *Percona XtraBackup*. The option can be useful if the *xtrabackup* binary is not in your search path or working directory. If this option is not specified, *innobackupex* attempts to determine the binary to use automatically.

--include=REGEXP

This option is a regular expression to be matched against table names in `databasename.tablename` format. It is passed directly to xtrabackup's `xtrabackup --tables` option. See the xtrabackup documentation for details.

--incremental

This option tells *xtrabackup* to create an incremental backup, rather than a full one. It is passed to the *xtrabackup* child process. When this option is specified, either `innobackupex --incremental-lsn` or `innobackupex --incremental-basedir` can also be given. If neither option is given, option `innobackupex --incremental-basedir` is passed to `xtrabackup` by default, set to the first timestamped backup directory in the backup base directory.

--incremental-basedir=DIRECTORY

This option accepts a string argument that specifies the directory containing the full backup that is the base dataset for the incremental backup. It is used with the `innobackupex --incremental` option.

--incremental-dir=DIRECTORY

This option accepts a string argument that specifies the directory where the incremental backup will be combined with the full backup to make a new full backup. It is used with the `innobackupex --incremental` option.

--incremental-history-name=NAME

This option specifies the name of the backup series stored in the PERCONA_SCHEMA.xtrabackup_history history record to base an incremental backup on. Percona Xtrabackup will search the history table looking for the most recent (highest innodb_to_lsn), successful backup in the series and take the to_lsn value to use as the starting lsn for the incremental backup. This will be mutually exclusive with `innobackupex --incremental-history-uuid`, `innobackupex --incremental-basedir` and `innobackupex --incremental-lsn`. If no valid lsn can be found (no series by that name, no successful backups by that name) xtrabackup will return with an error. It is used with the `innobackupex --incremental` option.

--incremental-history-uuid=UUID

This option specifies the UUID of the specific history record stored in the PERCONA_SCHEMA.xtrabackup_history to base an incremental backup on. `innobackupex --incremental-history-name`, `innobackupex --incremental-basedir\` and `innobackupex --incremental-lsn`. If no valid lsn can be found (no success record with that uuid) xtrabackup will return with an error. It is used with the `innobackupex --incremental` option.

--incremental-lsn=LSN

This option accepts a string argument that specifies the log sequence number (`LSN`) to use for the incremental backup. It is used with the `innobackupex --incremental` option. It is used instead of specifying `innobackupex --incremental-basedir`. For databases created by *MySQL* and *Percona Server* 5.0-series versions, specify the as two 32-bit integers in high:low format. For databases created in 5.1 and later, specify the LSN as a single 64-bit integer.

--kill-long-queries-timeout=SECONDS

This option specifies the number of seconds innobackupex waits between starting `FLUSH TABLES WITH READ LOCK` and killing those queries that block it. Default is 0 seconds, which means innobackupex will not attempt to kill any queries. In order to use this option xtrabackup user should have `PROCESS` and `SUPER` privileges. Where supported (Percona Server 5.6+) xtrabackup will automatically use Backup Locks as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.

--kill-long-query-type=all|select

This option specifies which types of queries should be killed to unblock the global lock. Default is “all”.

--ftwrl-wait-timeout=SECONDS

This option specifies time in seconds that innobackupex should wait for queries that would block `FLUSH TABLES WITH READ LOCK` before running it. If there are still such queries when the timeout expires, innobackupex terminates with an error. Default is 0, in which case innobackupex does not wait for queries to complete and starts `FLUSH TABLES WITH READ LOCK` immediately. Where supported (Percona Server 5.6+) xtrabackup will automatically use Backup Locks as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.

--ftwrl-wait-threshold=SECONDS

This option specifies the query run time threshold which is used by innobackupex to detect long-running queries with a non-zero value of `innobackupex –ftwrl-wait-timeout`. `FLUSH TABLES WITH READ LOCK` is not started until such long-running queries exist. This option has no effect if –ftwrl-wait-timeout is 0. Default value is 60 seconds. Where supported (Percona Server 5.6+) xtrabackup will automatically use Backup Locks as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables.

--ftwrl-wait-query-type=all|update

This option specifies which types of queries are allowed to complete before innobackupex will issue the global lock. Default is all.

--log-copy-interval=

This option specifies time interval between checks done by log copying thread in milliseconds.

--move-back

Move all the files in a previously made backup from the backup directory to their original locations. As this option removes backup files, it must be used with caution.

--no-lock

Use this option to disable table lock with `FLUSH TABLES WITH READ LOCK`. Use it only if ALL your tables are InnoDB and you **DO NOT CARE** about the binary log position of the backup. This option shouldn't be used if there are any `DDL` statements being executed or if any updates are happening on non-InnoDB tables (this includes the system MyISAM tables in the *mysql* database), otherwise it could lead to an inconsistent backup. Where supported (Percona Server 5.6+) xtrabackup will automatically use Backup Locks as a lightweight alternative to `FLUSH TABLES WITH READ LOCK` to copy non-InnoDB data to avoid blocking DML queries that modify InnoDB tables. If you are considering to use `innobackupex --no-lock because` your backups are failing to acquire the lock, this could be because of incoming replication events preventing the lock from succeeding. Please try using `innobackupex --safe-slave-backup` to momentarily stop the replication replica thread, this may help the backup to succeed and you then don't need to resort to using this option. `xtrabackup_binlog_info` is not created when –no-lock option is used (because `SHOW MASTER STATUS` may be inconsistent), but under certain conditions `xtrabackup_binlog_pos_innodb` can be used instead to get consistent binlog coordinates as described in Working with Binary Logs.

--no-timestamp

This option prevents creation of a time-stamped subdirectory of the `BACKUP-ROOT-DIR` given on the command line. When it is specified, the backup is done in `BACKUP-ROOT-DIR` instead.

--no-version-check

This option disables the version check. If you do not pass this option, the automatic version check is enabled implicitly when `xtrabackup` runs in the `--backup` mode. To disable the version check, explicitly pass the `--no-version-check` option when invoking `xtrabackup`. When the automatic version check is enabled,| program| performs a version check against the server on the backup stage after creating a server connection.

`xtrabackup` sends the following information to the server:

- MySQL flavour and version
- Operating system name
- Percona Toolkit version
- Perl version

Each piece of information has a unique identifier which is an MD5 hash value that Percona Toolkit uses to obtain statistics about how it is used. This value is a random UUID; no client information is either collected or stored.

--parallel=NUMBER-OF-THREADS

This option accepts an integer argument that specifies the number of threads the `xtrabackup` child process should use to back up files concurrently. Note that this option works on file level, that is, if you have several .ibd files, they will be copied in parallel. If your tables are stored together in a single tablespace file, it will have no effect. This option will allow multiple files to be decrypted and/or decompressed simultaneously. In order to decompress, the qpress utility MUST be installed and accessable within the path. This process will remove the original compressed/encrypted files and leave the results in the same location. It is passed directly to xtrabackup's `xtrabackup --parallel` option. See the `xtrabackup` documentation for details

--password=PASSWORD

This option accepts a string argument specifying the password to use when connecting to the database. It is passed to the `mysql` child process without alteration. See `mysql --help` for details.

--port=PORT

This option accepts a string argument that specifies the port to use when connecting to the database server with TCP/IP. It is passed to the `mysql` child process. It is passed to the `mysql` child process without alteration. See `mysql --help` for details.

--rebuild-indexes

This option only has effect when used together with the `--apply-log <innobackupex --apply-log>` option and is passed directly to xtrabackup. When used, makes xtrabackup rebuild all secondary indexes after applying the log. This option is normally used to prepare compact backups. See the `xtrabackup` documentation for more information.

--rebuild-threads=NUMBER-OF-THREADS

This option only has effect when used together with the `innobackupex --apply-log` and `innobackupex --rebuild-indexes` option and is passed directly to xtrabackup. When used, xtrabackup processes tablespaces in parallel with the specified number of threads when rebuilding indexes. See the `xtrabackup` documentation for more information.

--redo-only

This option should be used when preparing the base full backup and when merging all incrementals except the last one. It is passed directly to xtrabackup's `xtrabackup --apply-log-only` option. This forces `xtrabackup` to skip the "rollback" phase and do a "redo" only. This is necessary if the backup will have incremental changes applied to it later. See the *xtrabackup* documentation for details.

--rsync

Uses the `rsync` utility to optimize local file transfers. When this option is specified, `innobackupex` uses `rsync` to copy all non-InnoDB files instead of spawning a separate `cp` for each file, which can be much faster for servers with a large number of databases or tables. This option cannot be used together with `innobackupex --stream`.

--safe-slave-backup

When specified, innobackupex will stop the replica SQL thread just before running `FLUSH TABLES WITH READ LOCK` and wait to start backup until `Slave_open_temp_tables` in `SHOW STATUS` is zero. If there are no open temporary tables, the backup will take place, otherwise the SQL thread will be started and stopped until there are no open temporary tables. The backup will fail if `Slave_open_temp_tables` does not become zero after `innobackupex --safe-slave-backup-timeout` seconds. The replica SQL thread will be restarted when the backup finishes.

--safe-slave-backup-timeout=SECONDS

How many seconds `innobackupex --safe-slave-backup` should wait for `Slave_open_temp_tables` to become zero. Defaults to 300 seconds.

--slave-info

This option is useful when backing up a replication replica server. It prints the binary log position and name of the source server. It also writes this information to the `xtrabackup_slave_info` file as a `CHANGE MASTER` command. A new replica for this source can be set up by starting a replica server on this backup and issuing a `CHANGE MASTER` command with the binary log position saved in the `xtrabackup_slave_info` file.

--socket

This option accepts a string argument that specifies the socket to use when connecting to the local database server with a UNIX domain socket. It is passed to the mysql child process without alteration. See `mysql --help` for details.

--stream=STREAMNAME

This option accepts a string argument that specifies the format in which to do the streamed backup. The backup will be done to `STDOUT` in the specified format. Currently, supported formats are `tar` and `xbstream`. Uses xbstream, which is available in *Percona XtraBackup* distributions. If you specify a path after this option, it will be interpreted as the value of `tmpdir`

--tables-file=FILE

This option accepts a string argument that specifies the file in which there are a list of names of the form `database.table`, one per line. The option is passed directly to `xtrabackup` 's `innobackupex --tables-file` option.

--throttle=

This option limits the number of chunks copied per second. The chunk size is *10 MB*. To limit the bandwidth to *10 MB/s*, set the option to *1*: `--throttle=1`.

See also

More information about how to throttle a backup Throttling Backups.

--tmpdir=DIRECTORY

This option accepts a string argument that specifies the location where a temporary file will be stored. It may be used when `innobackupex --stream` is specified. For these options, the transaction log will first be stored to a temporary file, before streaming or copying to a remote host. This option specifies the location where that temporary file will be stored. If the option is not specified, the default is to use the value of `tmpdir` read from the server configuration. innobackupex is passing the tmpdir value specified in my.cnf as the `--target-dir` option to the xtrabackup binary. Both [mysqld] and [xtrabackup] groups are read from my.cnf. If there is tmpdir in both, then the value being used depends on the order of those group in my.cnf.

--use-memory=

This option accepts a string argument that specifies the amount of memory in bytes for `xtrabackup` to use for crash recovery while preparing a backup. Multiples are supported providing the unit (e.g. 1MB, 1M, 1GB, 1G).

It is used only with the option `innobackupex --apply-log`. It is passed directly to xtrabackup's `xtrabackup --use-memory` option. See the *xtrabackup* documentation for details.

--user=USER

This option accepts a string argument that specifies the user (i.e., the *MySQL* username used when connecting to the server) to login as, if that's not the current user. It is passed to the mysql child process without alteration. See `mysql --help` for details.

--version

This option displays the *innobackupex* version and copyright notice and then exits.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

10.4 The xbcloud Binary

The purpose of *xbcloud* is to download and upload full or part of xbstream archive from/to the cloud. *xbcloud* will not overwrite the backup with the same name. *xbcloud* accepts input via a pipe from xbstream so that it can be invoked as a pipeline with *xtrabackup* to stream directly to the cloud without needing a local storage.

Note

In a Bash shell, the `$?` parameter returns the exit code from the last binary. If you use pipes the `${PIPESTATUS[x]}` array parameter returns the exit codes for each binary in the pipe string.

```shell $ xtrabackup --backup --stream=xbstream --target-dir=/storage/backups/ | xbcloud put [options] full_backup

true | false echo $? 1

## 10.5 with PIPESTATUS

true | false echo ${PIPESTATUS[0]} ${PIPESTATUS[1]} 0 1 ```

The *xbcloud* binary stores each chunk as a separate object with a name `backup_name/database/table.ibd.NNNNNNNNNNNNNNNNNNNN`, where `NNN...` is a 0-padded serial number of chunk within a file. Size of chunk produced by *xtrabackup* and xbstream changed to 10MB.

**Note**

Use `--read-buffer-size` to adjust the chunk size.

If you use encryption, specify both the `--read-buffer-size` and `--encrypt-chunk-size` options to adjust the chunk size.

*xbcloud* has three essential operations: *put*, *get*, and *delete*. With these operations, backups are created, stored, retrieved, restored, and deleted. *xbcloud* operations clearly map to similar operations within the AWS S3 API.

### 10.5.1 Version specific information

- 2.4.25 - Added the support for Microsoft Azure Cloud Storage
- 2.4.21 - Added s3-storage-class and google-storage-class
- 2.4.14 - Added the support of *Amazon S3*, *MinIO* and *Google Cloud Storage* storage types.
- 2.3.1-beta1 - Implemented ability to store *xbcloud* parameters in a .cnf file
- 2.3.1-beta1 - Implemented support different authentication options for Swift
- 2.3.1-beta1 - Implemented support for partial download of the cloud backups
- 2.3.1-beta1 - `xbcloud --swift-url` option has been renamed to `xbcloud --swift-auth-url`
- 2.3.0-alpha1 - Initial implementation

### 10.5.2 Supported Cloud Storage Types

Swift was the only option for storing backups in cloud storage until *Percona XtraBackup 2.4.14*.
```

The xbcloud binary supports Amazon S3, Azure, MinIO and Google Cloud Storage. Amazon S3-compatible cloud storage types, such as Wasabi and Digital Ocean Spaces, are also supported. The xbcloud binary should work with other S3-compatible storage options.

See also

OpenStack Object Storage ("Swift")

Amazon Simple Storage Service

Azure Cloud Storage

MinIO

Google Cloud Storage

Wasabi

Digital Ocean Spaces

10.5.3 Usage

```
$ xtrabackup --backup --stream=xbstream --target-dir=/tmp | xbcloud \
put [options] <name>
```

10.5.4 Creating a full backup with Swift

The following example shows how to make a full backup and upload it to Swift.

```
$ xtrabackup --backup --stream=xbstream --extra-lsndir=/tmp --target-dir=/tmp | \
xbcloud put --storage=swift \
--swift-container=test \
--swift-user=test:tester \
--swift-auth-url=http://192.168.8.80:8080/ \
--swift-key=testing \
--parallel=10 \
full_backup
```

10.5.5 Creating a full backup with *Amazon S3*

```
$ xtrabackup --backup --stream=xbstream --extra-lsndir=/tmp --target-dir=/tmp | \
xbcloud put --storage=s3 \
--s3-endpoint='s3.amazonaws.com' \
--s3-access-key='YOUR-ACCESSKEYID' \
--s3-secret-key='YOUR-SECRETACCESSKEY' \
--s3-bucket='mysql_backups'
--parallel=10 \
$(date -I)-full_backup
```

The following options are available when using *Amazon S3*:

Option	Details
--s3-access-key	Use to supply the AWS access key ID
--s3-secret-key	Use to supply the AWS secret access key
--s3-bucket	Use supply the AWS bucket name
--s3-region	Use to specify the AWS region. The default value is **us-east-1**
--s3-api-version = <AUTO|2|4>	Select the signing algorithm. The default value is AUTO. In this case, *xbcloud* will probe.
--s3-bucket-lookup = <AUTO|PATH|DNS>	Specify whether to use **bucket.endpoint.com** or *endpoint.com/bucket** style requests. The default value is AUTO. In this case, *xbcloud* will probe.
--s3-storage-class=<name>	Specify the S3 storage class. The name options are the following: STANDARD STANDARD_IA GLACIER NOTE: If you use the GLACIER storage class, the object must be restored to S3 before restoring the backup. Also, supports using custom S3 implementations such as MinIO or CephRadosGW.

10.5.6 Creating a full backup with MinIO

```
$ xtrabackup --backup --stream=xbstream --extra-lsndir=/tmp --target-dir=/tmp | \
xbcloud put --storage=s3 \
--s3-endpoint='play.minio.io:9000' \
--s3-access-key='YOUR-ACCESSKEYID' \
--s3-secret-key='YOUR-SECRETACCESSKEY' \
--s3-bucket='mysql_backups'
--parallel=10 \
$(date -I)-full_backup
```

10.5.7 Creating a full backup with Google Cloud Storage

The support for Google Cloud Storage is implemented using the interoperability mode. This mode was especially designed to interact with cloud services compatible with *Amazon S3*.

See also

Cloud Storage Interoperability https://cloud.google.com/storage/docs/interoperability

```
$ xtrabackup --backup --stream=xbstream --extra-lsndir=/tmp --target-dir=/tmp | \
xbcloud put --storage=google \
--google-endpoint=`storage.googleapis.com` \
--google-access-key='YOUR-ACCESSKEYID' \
--google-secret-key='YOUR-SECRETACCESSKEY' \
--google-bucket='mysql_backups'
--parallel=10 \
$(date -I)-full_backup
```

The following options are available when using Google Cloud Storage:

- --google-access-key = <ACCESS KEY ID>
- --google-secret-key = <SECRET ACCESS KEY>
- --google-bucket = <BUCKET NAME>
- --google-storage-class=name

Note

The Google storage class name options are the following:

- STANDARD
- NEARLINE
- COLDLINE
- ARCHIVE

See also: Google storage classes

10.5.8 Supplying parameters

Each storage type has mandatory parameters that you can supply on the command line, in a configuration file, and via environment variables.

Configuration files

The parameters the values of which do not change frequently can be stored in `my.cnf` or in a custom configuration file. The following example is a template of configuration options under the `[xbcloud]` group:

```
[xbcloud]
storage=s3
s3-endpoint=http://localhost:9000/
s3-access-key=minio
s3-secret-key=minio123
s3-bucket=backupsx
s3-bucket-lookup=path
s3-api-version=4
```

Note

If you explicitly use a parameter on the command line and in a configuration file, *xbcloud* uses the the value provided on the command line.

Environment variables

The following environment variables are recognized. *xbcloud* maps them automatically to corresponding parameters applicable to the selected storage.

- AWS_ACCESS_KEY_ID (or ACCESS_KEY_ID)
- AWS_SECRET_ACCESS_KEY (or SECRET_ACCESS_KEY)
- AWS_DEFAULT_REGION (or DEFAULT_REGION)
- AWS_ENDPOINT (or ENDPOINT)
- AWS_CA_BUNDLE

Note

If you explicitly use a parameter on the command line, in a configuration file, and the corresponding environment variable contains a value, *xbcloud* uses the the value provided on the command line or in the configuration file.

OpenStack environment variables are also recognized and mapped automatically to corresponding **swift** parameters (`--storage=swift`).

- OS_AUTH_URL
- OS_TENANT_NAME
- OS_TENANT_ID
- OS_USERNAME
- OS_PASSWORD
- OS_USER_DOMAIN
- OS_USER_DOMAIN_ID
- OS_PROJECT_DOMAIN
- OS_PROJECT_DOMAIN_ID
- OS_REGION_NAME
- OS_STORAGE_URL
- OS_CACERT

Shortcuts

For all operations (put, get, and delete), you can use a shortcut to specify the storage type, bucket name, and backup name as one parameter instead of using three distinct parameters (--storage, --s3-bucket, and backup name per se).

Using a shortcut syntax to provide a storage type, bucket, and backup name

Use the following format: `storage-type://bucket-name/backup-name`

```
$ xbcloud get s3://operator-testing/bak22 ...
```

In this example, **s3** refers to a storage type, **operator-testing** is a bucket name, and **bak22** is the backup name. This shortcut expands as follows:

```
$ xbcloud get --storage=s3 --s3-bucket=operator-testing bak22 ...
```

You can supply the mandatory parameters not only on the command line. You may use configuration files and environment variables.

Additional parameters

xbcloud accepts additional parameters that you can use with any storage type. The `--md5` parameter computes the MD5 hash value of the backup chunks. The result is stored in files that following the `backup_name.md5` pattern.

```
$ xtrabackup --backup --stream=xbstream \
--parallel=8 2>backup.log | xbcloud put s3://operator-testing/bak22 \
--parallel=8 --md5 2>upload.log
```

You may use the `--header` parameter to pass an additional HTTP header with the server side encryption while specifying a customer key.

Example of using --header for AES256 encryption

```
$ xtrabackup --backup --stream=xbstream --parallel=4 | \
xbcloud put s3://operator-testing/bak-enc/ \
--header="X-Amz-Server-Side-Encryption-Customer-Algorithm: AES256" \
--header="X-Amz-Server-Side-Encryption-Customer-Key: CuStoMerKey=" \
--header="X-Amz-Server-Side-Encryption-Customer-Key-MD5: CuStoMerKeyMd5==" \
--parallel=8
```

The `--header` parameter is also useful to set the access control list (ACL) permissions: `--header="x-amz-acl: bucket-owner-full-control`

10.5.9 Restoring with Swift

```
xbcloud get [options] <name> [<list-of-files>] | xbstream -x
```

The following example shows how to fetch and restore the backup from Swift:

```
$ xbcloud get --storage=swift \
--swift-container=test \
--swift-user=test:tester \
--swift-auth-url=http://192.168.8.80:8080/ \
--swift-key=testing \
full_backup | xbstream -xv -C /tmp/downloaded_full

$ xtrabackup --prepare --target-dir=/tmp/downloaded_full
$ xtrabackup --copy-back --target-dir=/tmp/downloaded_full
```

10.5.10 Restoring with *Amazon S3*

```
$ xbcloud get s3://operator-testing/bak22 \
--s3-endpoint=https://storage.googleapis.com/ \
--parallel=10 2>download.log | xbstream -x -C restore --parallel=8
```

10.5.11 Incremental backups

First, make the full backup which is the base for an incremental backup:

```
$ xtrabackup --backup --stream=xbstream --extra-lsndir=/storage/backups/ \
--target-dir=/storage/backups/ | xbcloud put \
--storage=swift --swift-container=test_backup \
--swift-auth-version=2.0 --swift-user=admin \
--swift-tenant=admin --swift-password=xoxoxoxo \
--swift-auth-url=http://127.0.0.1:35357/ --parallel=10 \
full_backup
```

Then make the incremental backup:

```
$ xtrabackup --backup --incremental-basedir=/storage/backups \
--stream=xbstream --target-dir=/storage/inc_backup | xbcloud put \
--storage=swift --swift-container=test_backup \
--swift-auth-version=2.0 --swift-user=admin \
--swift-tenant=admin --swift-password=xoxoxoxo \
--swift-auth-url=http://127.0.0.1:35357/ --parallel=10 \
inc_backup
```

Preparing an incremental backup

To prepare a backup, download the full backup:

```
$ xbcloud get --swift-container=test_backup \
--swift-auth-version=2.0 --swift-user=admin \
--swift-tenant=admin --swift-password=xoxoxoxo \
--swift-auth-url=http://127.0.0.1:35357/ --parallel=10 \
full_backup | xbstream -xv -C /storage/downloaded_full
```

Prepare the downloaded full backup:

```
$ xtrabackup --prepare --apply-log-only --target-dir=/storage/downloaded_full
```

After the full backup has been prepared, download the incremental backup:

```
$ xbcloud get --swift-container=test_backup \
--swift-auth-version=2.0 --swift-user=admin \
--swift-tenant=admin --swift-password=xoxoxoxo \
--swift-auth-url=http://127.0.0.1:35357/ --parallel=10 \
inc_backup | xbstream -xv -C /storage/downloaded_inc
```

Prepare the incremental backup:

```
$ xtrabackup --prepare --apply-log-only \
--target-dir=/storage/downloaded_full \
--incremental-dir=/storage/downloaded_inc

$ xtrabackup --prepare --target-dir=/storage/downloaded_full
```

Partial download of the cloud backup

If you don't want to download the entire backup to restore a database you can restore only specific tables:

```
$ xbcloud get --swift-container=test_backup
--swift-auth-version=2.0 --swift-user=admin \
--swift-tenant=admin --swift-password=xoxoxoxo \
--swift-auth-url=http://127.0.0.1:35357/ full_backup \
ibdata1 sakila/payment.ibd \
> /storage/partial/partial.xbs

$ xbstream -xv -C /storage/partial < /storage/partial/partial.xbs
```

This command downloads the `ibdata1` table and the `sakila/payment.ibd` table from a full backup.

10.5.12 Command-line options

xbcloud has the following command line options:

--storage=[swift*Amazon S3*google]

Cloud storage option. *xbcloud* supports Swift, MinIO, and AWS S3. The default value is `swift`.

--swift-auth-url

URL of Swift cluster.

--swift-url

Renamed to xbcloud –swift-auth-url

--swift-storage-url

xbcloud attempts to get object-store URL for a specified region (if any specified) from the keystone response. One can override that URL by passing --swift-storage-url=URL argument.

--swift-user

Swift username (X-Auth-User, specific to Swift)

--swift-key

Swift key/password (X-Auth-Key, specific to Swift)

--swift-container

Container to backup into (specific to Swift)

--parallel=N

Maximum number of concurrent upload/download requests. Default is `1`.

--cacert

Path to the file with CA certificates

--insecure

Do not verify servers certificate

Swift authentication options

Swift specification describe several authentication options. *xbcloud* can authenticate against keystone with API version 2 and 3.

--swift-auth-version

Specifies the swift authentication version. Possible values are: `1.0` - TempAuth, `2.0` - Keystone v2.0, and `3` - Keystone v3. Default value is `1.0`.

For v2 additional options are:

--swift-tenant

Swift tenant name.

--swift-tenant-id

Swift tenant ID.

--swift-region

Swift endpoint region.

--swift-password

Swift password for the user.

For v3 additional options are:

--swift-user-id

Swift user ID.

--swift-project

Swift project name.

--swift-project-id

Swift project ID.

--swift-domain

Swift domain name.

--swift-domain-id

Swift domain ID.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-02-27

10.6 Exponential Backoff

This feature was implemented in Percona XtraBackup 2.4.24 in the xbcloud binary.

Exponential backoff increases the chances for the completion of a backup or a restore operation. For example, a chunk upload or download may fail if you have an unstable network connection or other network issues. This feature adds an exponential backoff, or sleep, time and then retries the upload or download.

When a chunk upload or download operation fails, xbcloud checks the reason for the failure. This failure can be a CURL error or an HTTP error, or a client-specific error. If the error is listed in the Retriable errors list, xbcloud pauses for a calculated time before retrying the operation until that time reaches the `--max-backoff` value.

The operation is retried until the `--max-retries` value is reached. If the chunk operation fails on the last retry, xbcloud aborts the process.

The default values are the following:

- –max-backoff = 300000 (5 minutes)
- –max-retries = 10

You can adjust the number of retries by adding the `--max-retries` parameter and adjust the maximum length of time between retries by adding the `--max-backoff` parameter to an xbcloud command.

Since xbcloud does multiple asynchronous requests in parallel, a calculated value, measured in milliseconds, is used for `max-backoff`. This algorithm calculates how many milliseconds to sleep before the next retry. A number generated is based on the combining the power of two (2), the number of retries already attempted and adds a random number between 1 and 1000. This number avoids network congestion if multiple chunks have the same backoff value. If the default values are used, the final retry attempt to be approximately 17 minutes after the first try. The number is no longer calculated when the milliseconds reach the `--max-backoff` setting. At that point, the retries continue by using the `--max-backoff` setting until the `max-retries` parameter is reached.

10.6.1 Retriable errors

We retry for the following CURL operations:

- CURLE_GOT_NOTHING
- CURLE_OPERATION_TIMEOUT
- CURLE_RECV_ERROR
- CURLE_SEND_ERROR
- CURLE_SEND_FAIL_REWIND
- CURLE_PARTIAL_FILE
- CURLE_SSL_CONNECT_ERROR

We retry for the following HTTP operation status codes:

- 503
- 500
- 504
- 408

Each cloud provider may return a different CURL error or an HTTP error, depending on the issue. Add new errors by setting the following variables `--curl-retriable-errors` or `--http-retriable-errors` on the command line or in `my.cnf` or in a custom configuration file under the [xbcloud] section.

The error handling is enhanced when using the `--verbose` output. This output specifies which error caused xbcloud to fail and what parameter a user must add to retry on this error.

The following is an example of a verbose output:

```
210701 14:34:23 /work/pxb/ins/2.4/bin/xbcloud: Operation failed. Error: Server returned
nothing (no headers, no data)
210701 14:34:23 /work/pxb/ins/2.4/bin/xbcloud: Curl error (52) Server returned nothing (no
headers, no data) is not configured as retriable. You can allow it by adding --curl-
retriable-errors=52 parameter
```

10.6.2 Example

The following example adjusts the maximum number of retries and the maximum time between retries.

```
xbcloud [options] --max-retries=5 --max-backoff=10000
```

The following text is an example of the exponential backoff used with the command:

```
210702 10:07:05 /work/pxb/ins/2.4/bin/xbcloud: Operation failed. Error: Server returned
nothing (no headers, no data)
210702 10:07:05 /work/pxb/ins/2.4/bin/xbcloud: Sleeping for 2384 ms before retrying backup3/
xtrabackup_logfile.00000000000000000006 [1]
. . .
210702 10:07:23 /work/pxb/ins/2.4/bin/xbcloud: Operation failed. Error: Server returned
nothing (no headers, no data)
210702 10:07:23 /work/pxb/ins/2.4/bin/xbcloud: Sleeping for 4387 ms before retrying backup3/
xtrabackup_logfile.00000000000000000006 [2]
. . .
210702 10:07:52 /work/pxb/ins/2.4/bin/xbcloud: Operation failed. Error: Failed sending data
to the peer
210702 10:07:52 /work/pxb/ins/2.4/bin/xbcloud: Sleeping for 8691 ms before retrying backup3/
xtrabackup_logfile.00000000000000000006 [3]
. . .
210702 10:08:47 /work/pxb/ins/2.4/bin/xbcloud: Operation failed. Error: Failed sending data
to the peer
210702 10:08:47 /work/pxb/ins/2.4/bin/xbcloud: Sleeping for 10000 ms before retrying backup3/
xtrabackup_logfile.00000000000000000006 [4]
. . .
210702 10:10:12 /work/pxb/ins/2.4/bin/xbcloud: successfully uploaded chunk: backup3/
xtrabackup_logfile.00000000000000000006, size: 8388660
```

The following list details the example output:

[1.] Chunk `xtrabackup_logfile.00000000000000000006` fails to upload the first time and slept for 2384 milliseconds.

[2.] The same chunk fails for the second time and the time is increased to 4387 milliseconds.

[3.] The same chunk fails for the third time and the time is increased to 8691 milliseconds.

[4.] The same chunk fails for the fourth time. The `max-backoff` =10000, which defines the maximum sleep time as 10000. Any retry sleeps the same amount of time after reaching the parameter.

[5.] The same chunk is successfully uploaded.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

10.7 Using the xbcloud binary with Microsoft Azure Cloud Storage

This feature is *technical preview* quality.

Implemented in Percona XtraBackup 2.4.25, the **xbcloud** binary adds support for the Microsoft Azure Cloud Storage using the REST API.

10.7.1 Options

The following are the options, environment variables, and descriptions for uploading a backup to Azure using the REST API. The environment variables are recognized by **xbcloud**, which maps them automatically to the corresponding parameters:

Option name	Environment variables	Description
`--azure-storage-account=name`	AZURE_STORAGE_ACCOUNT	An Azure storage account is a unique namespace to access and store your Azure data objects.
`--azure-container-name=name`	AZURE_CONTAINER_NAME	A container name is a valid DNS name that conforms to the Azure naming rules
`--azure-access-key=name`	AZURE_ACCESS_KEY	A generated key that can be used to authorize access to data in your account using the Shared Key authorization.
`--azure-endpoint=name`	AZURE_ENDPOINT	The endpoint allows clients to securely access data.
`--azure-tier-class=name`	AZURE_STORAGE_CLASS	Cloud tier can decrease the local storage required while maintaining the performance. When enabled, this feature has the following categories: * Hot - Frequently accessed or modified data * Cool - Infrequently accessed or modified data * Archive - Rarely accessed or modified data

Test your Azure applications with the Azurite open-source emulator. For testing purposes, the **xbcloud** binary adds the `--azure-development-storage` option that uses the default `access_key` and `storage account` of azurite and `testcontainer` for the container. You can overwrite these options, if needed.

10.7.2 Usage

All of the available options for **xbcloud**, such as parallel, max-retries, and others, can be used. For more information, see The xbcloud Binary.

10.7.3 Examples

An example of a **xbcloud** backup.

```
$ xtrabackup --backup --stream=xbstream --target-dir= $TARGET_DIR | xbcloud put backup_name
--azure-storage-account=pxbtesting --azure-access-key=$AZURE_KEY --azure-container-name=test
--storage=azure
```

An example of restoring a backup from **xbcloud**.

```
$ xbcloud get backup_name  --azure-storage-account=pxbtesting --azure-access-key=$AZURE_KEY
--azure-container-name=test --storage=azure --parallel=10 2>download.log | xbstream -x -C
restore
```

An example of deleting a backup from **xbcloud**.

```
$ xbcloud delete backup_name --azure-storage-account=pxbtesting --azure-access-
key=$AZURE_KEY --azure-container-name=test --storage=azure
```

An example of using a shortcut restore.

```
$ xbcloud get azure://operator-testing/bak22 ...
```

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

10.8 The xbcrypt binary

To support encryption and decryption of the backups, a new tool `xbcrypt` was introduced to *Percona XtraBackup*.

Percona XtraBackup 2.4.25 implements the XBCRYPT_ENCRYPTION_KEY environment variable. The variable is only used in place of the `--encrypt_key=name` option. You can use the environment variable or command line option. If you use both, the command line option takes precedence over the value specified in the environment variable.

This utility has been modeled after The xbstream binary to perform encryption and decryption outside of *Percona XtraBackup*. `xbcrypt` has following command line options:

-d, --decrypt

Decrypt data input to output.

-i, --input=name

Optional input file. If not specified, input will be read from standard input.

-o, --output=name

Optional output file. If not specified, output will be written to standard output.

-a, --encrypt-algo=name

Encryption algorithm.

-k, --encrypt-key=name

Encryption key.

-f, --encrypt-key-file=name

File which contains encryption key.

-s, --encrypt-chunk-size=

Size of working buffer for encryption in bytes. The default value is 64K.

--encrypt-threads=

This option specifies the number of worker threads that will be used for parallel encryption/decryption.

-v, --verbose

Display verbose status output.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

10.9 The xbstream binary

To support simultaneous compression and streaming, a new custom streaming format called xbstream was introduced to *Percona XtraBackup* in addition to the TAR format. That was required to overcome some limitations of traditional archive formats such as tar, cpio and others which did not allow streaming dynamically generated files, for example dynamically compressed files. Other advantages of xbstream over traditional streaming/archive format include ability to stream multiple files concurrently (so it is possible to use streaming in the xbstream format together with the –parallel option) and more compact data storage.

This utility has a tar-like interface:

- with the `-x` option it extracts files from the stream read from its standard input to the current directory unless specified otherwise with the `-C` option. Support for parallel extraction with the `--parallel` option has been implemented in *Percona XtraBackup* 2.4.7.
- with the `-c` option it streams files specified on the command line to its standard output.
- with the `--decrypt=ALGO` option specified xbstream will automatically decrypt encrypted files when extracting input stream. Supported values for this option are: `AES128`, `AES192`, and `AES256`. Either `--encrypt-key` or `--encrypt-key-file` options must be specified to provide encryption key, but not both. This option has been implemented in *Percona XtraBackup* 2.4.7.
- with the `--encrypt-threads` option you can specify the number of threads for parallel data encryption. The default value is `1`. This option has been implemented in *Percona XtraBackup* 2.4.7.
- the `--encrypt-key` option is used to specify the encryption key that will be used. It can't be used with `--encrypt-key-file` option because they are mutually exclusive. This option has been implemented in Percona XtraBackup 2.4.7.
- the `--encrypt-key-file` option is used to specify the file that contains the encryption key. It can't be used with `--encrypt-key` option because they are mutually exclusive. This option has been implemented in *Percona XtraBackup* 2.4.7.

The utility also tries to minimize its impact on the OS page cache by using the appropriate `posix_fadvise()` calls when available.

When compression is enabled with *xtrabackup* all data is being compressed, including the transaction log file and meta data files, using the specified compression algorithm. The only currently supported algorithm is `quicklz`.

The resulting files have the qpress archive format, i.e., every `\*.qp` file produced by *xtrabackup* is essentially a one-file qpress archive and can be extracted and uncompressed by the qpress file archiver. This means that there is no need to decompress entire backup to restore a single table as with `tar.gz`.

Files can be decompressed using the **qpress** tool that can be downloaded from here. Qpress supports multi-threaded decompression.

The default size of the chunk file is 10MB.

Note

Use `--read-buffer-size` to adjust the chunk size.

If you use encryption, specify both the `--read-buffer-size` and `--encrypt-chunk-size` options to adjust the chunk size.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

10.10 Known issues and limitations

There is a number of *Percona XtraBackup* related issues with compressed *InnoDB* tables. These issues result from either server-side bugs, or OS configuration and thus, cannot be fixed on the *Percona XtraBackup* side.

Known issues:

- For *MySQL* or *Percona Server for MySQL* versions 5.1 and 5.5 there are known and unfixed bugs with redo-logging of updates to compressed *InnoDB* tables. For example, internal Oracle bug #16267120 has been fixed only in *MySQL* 5.6.12, but not in 5.1 or 5.5. The bug is about compressed page images not being logged on page reorganization and thus, creating a possibility for recovery process to fail in case a different zlib version is being used when replaying a `MLOG_ZIP_PAGE_REORGANIZE` redo log record.
- For *MySQL* or *Percona Server for MySQL* version 5.6 it is **NOT** recommended to set `innodb_log_compressed_pages=OFF` for servers that use compressed *InnoDB* tables which are backed up with *Percona XtraBackup*. This option makes *InnoDB* recovery (and thus, backup prepare) sensible to `zlib` versions. In case the host where a backup prepare is performed uses a different `zlib` version than the one that was used by the server during runtime, backup prepare may fail due to differences in compression algorithms.
- Backed-up table data could not be recovered if backup was taken while running `OPTIMIZE TABLE` (bug https://bugs.launchpad.net/percona-xtrabackup/+bug/1541763) or `ALTER TABLE ... TABLESPACE` (bug PXB-1360) on that table.
- Compact Backups currently don't work due to bug PXB-372.
- Backup fails with `Error 24: 'Too many open files'`. This usually happens when database being backed up contains large amount of files and *Percona XtraBackup* can't open all of them to create a successful backup. In order to avoid this error the operating system should be configured appropriately so that *Percona XtraBackup* can open all its files. On Linux, this can be done with the `ulimit` command for specific backup session or by editing the /etc/security/limits.conf to change it globally (**NOTE**: the maximum possible value that can be set up is `1048576` which is a hard-coded constant in the Linux kernel).

The `xtrabackup` binary has some limitations you should be aware of to ensure that your backups go smoothly and are recoverable.

Limitations:

- The Aria storage engine is part of *MariaDB* and has been integrated in it for many years and Aria table files backup support has been added to *innobackupex* in 2011. The issue is that the engine uses recovery log files and an aria_log_control file that are not backed up by *xtrabackup*. As stated in the documentation, starting *MariaDB* without the `maria_log_control` file will mark all the Aria tables as corrupted with this error when doing a `CHECK` on the table: `Table is from another system and must be zerofilled or repaired to be usable on this system`. This means that the Aria tables from an *xtrabackup* backup must be repaired before being usable (this could ben quite long depending on the size of the table). Another option is `aria_chk --zerofil table` on all Aria tables present on the backup after the prepare phase.
- If the `xtrabackup_logfile` is larger than 4GB, the `--prepare` step will fail on 32-bit versions of `xtrabackup`.
- `xtrabackup` doesn't understand the very old `--set-variable` `my.cnf` syntax that MySQL uses. See Configuring xtrabackup.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

10.11 Frequently Asked Questions

10.11.1 Do I need an InnoDB Hot Backup license to use Percona XtraBackup?

No. Although `innobackupex` is derived from the same GPL and open-source wrapper script that InnoDB Hot Backup uses, it does not execute `ibbackup`, and the `xtrabackup` binary does not execute or link to `ibbackup`. You can use *Percona XtraBackup* without any license; it is completely separate from InnoDB Hot Backup.

10.11.2 What's the difference between innobackupex and innobackup?

The innobackupex binary is a patched version of the *Oracle* innobackup script (renamed mysqlbackup). They are similar, and familiarity with innobackup might be helpful.

Besides the available options for specific features of *innobackupex*, the main differences are:

- Prints to `STDERR` instead of `STDOUT` which enables the `innobackupex --stream` option
- Detects the configuration file - `my.cnf` - is automatically (or set with `innobackupex  --defaults-file`) instead of requiring the configuration file as the the first argument
- Defaults to *xtrabackup* as binary to use in the `innobackupex --ibbackup`

See The innobackupex Option Reference for more details.

10.11.3 Which Web-based backup tools are based on Percona XtraBackup?

Zmanda Recovery Manager is a commercial tool that uses *Percona XtraBackup* for Non-Blocking Backups:

"ZRM provides support for non-blocking backups of MySQL using Percona XtraBackup. *ZRM with* Percona XtraBackup *provides resource utilization management by providing throttling based on the number of IO operations per second.* Percona XtraBackup *based backups also allow for table-level recovery even though the backup was done at the database level. This operation requires the recovery database server to be* Percona Server for MySQL *with XtraDB."*

10.11.4 *xtrabackup* binary fails with a floating point exception

In most of the cases this is due to not having installed the required libraries (and version) by *xtrabackup*. Installing the *GCC* suite with the supporting libraries and recompiling *xtrabackup* will solve the issue. See Compiling and Installing from Source Code for instructions on the procedure.

10.11.5 How does xtrabackup handle the `ibdata/ib_log` files on restore if they are not in the MySQL datadir?

If the `ibdata` and `ib_log` files are located in different directories outside of the datadir, you move them to their proper place after the logs have been applied.

10.11.6 Backup fails with Error 24: 'Too many open files'

This error usually occurs when the database being backed up contains large amount of files and *Percona XtraBackup* can't open all of them to create a successful backup. In order to avoid this error the operating system should be configured appropriately so that *Percona XtraBackup* can open all its files. On Linux, this can be done with the `ulimit` command for specific backup session or by editing the /etc/security/limits.conf to change it globally

Note

The maximum possible value that can be set up is `1048576` which is a hard-coded constant in the Linux kernel.

10.11.7 How to deal with skipping of redo logs for DDL operations?

To prevent creating corrupted backups when running DDL operations, Percona XtraBackup aborts if it detects that redo logging is disabled. In this case, the following error is printed:

```
[FATAL] InnoDB: An optimized (without redo logging) DDL operation has been performed. All
modified pages may not have been flushed to the disk yet.
Percona XtraBackup will not be able to take a consistent backup. Retry the backup operation.
```

Note

Redo logging is disabled during a sorted index build

To avoid this error, Percona XtraBackup can use metadata locks on tables while they are copied:

- To block all DDL operations, use the `xtrabackup --lock-ddl` option that issues `LOCK TABLES FOR BACKUP`.
- If `LOCK TABLES FOR BACKUP` is not supported, you can block DDL for each table before XtraBackup starts to copy it and until the backup is completed using the `xtrabackup --lock-ddl-per-table` option.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-11-10

10.12 Glossary

.ARM file

Contains the metadata for each Archive Storage Engine table.

.ARZ file

Contains the data for each Archive Storage Engine table.

.CSM

Each table with the **CSV Storage Engine** has `.CSM` file which contains the metadata of it.

.CSV

Each table with the **CSV Storage** engine has `.CSV` file which contains the data of it (which is a standard Comma Separated Value file).

.exp

Files with the `.exp` extension are created by *Percona XtraBackup* per each *InnoDB* tablespace when the `--export` option is used in the prepare phase. See restoring individual tables.

.frm

For each table, the server will create a file with the `.frm` extension containing the table definition (for all storage engines).

.ibd

On a multiple tablespace setup (innodb_file_per_table enabled), *MySQL* will store each newly created table in a file with a `.ibd` extension.

.MRG

Each table using the **MERGE** storage engine, besides of a .frm file, will have .MRG file containing the names of the *MyISAM* tables associated with it.

.MYD

Each *MyISAM* table has `.MYD` (MYData) file which contains the data on it.

.MYI

Each *MyISAM* table has `.MYI` (MYIndex) file which contains the table’s indexes.

.opt

MySQL stores options of a database (like charset) in a file with a `.opt` extension in the database directory.

.par

Each partitioned table has .par file which contains metadata about the partitions.

.TRG

File containing the triggers associated with a table, for example `:file:mytable.TRG` With the .TRN file, they represent all the Trigger definitions.

.TRN

File containing the names of the triggers that are associated with a table, for example `:file:mytable.TRN`. With the `.TRG` file, they represent all the trigger definitions.

backup

The process of copying data or tables to be stored in a different location.

compression

The method that produces backups in a reduced size.

configuration file

The file that contains the server startup options.

crash

An unexpected shutdown that does not allow the normal server shutdown cleanup activities.

crash recovery

The actions that occur when MySQL is restarted after a crash.

data dictionary

The metadata for the tables, indexes, and table columns stored in the InnoDB system tablespace.

datadir

The directory in which the database server stores its data files. Most Linux distribution use `/var/lib/mysql` by default.

full backup

A backup that contains the complete source data from an instance.

ibdata

Default prefix for tablespace files, for example, `ibdata1` is a 10MB auto-extensible file that *MySQL* creates for the shared tablespace by default.

incremental backup

A backup stores data from a specific point in time.

InnoDB

Storage engine which provides ACID-compliant transactions and foreign key support, among others improvements over MyISAM. It is the default engine for *MySQL* as of the 8.0 series.

innodb_buffer_pool_size

The size in bytes of the memory buffer to cache data and indexes of *InnoDB*'s tables. This aims to reduce disk access to provide better performance. By default:

```
text
[mysqld]
innodb_buffer_pool_size=8MB
```

innodb_data_home_dir

The directory (relative to datadir) where the database server stores the files in a shared tablespace setup. This option does not affect the location of innodb_file_per_table. For example:

```
text
[mysqld]
innodb_data_home_dir = ./
```

innodb_data_file_path

Specifies the names, sizes and location of shared tablespace files:

```
text
[mysqld]
innodb_data_file_path=ibdata1:50M;ibdata2:50M:autoextend
```

innodb_file_per_table

By default, InnoDB creates tables and indexes in a file-per-tablespace. If the `innodb_file_per_table` variable is disabled, you can enable the variable in your configuration file:

```
text
[mysqld]
innodb_file_per_table
```

or start the server with `--innodb_file_per_table`.

innodb_log_group_home_dir

Specifies the location of the *InnoDB* log files:

```
text
[mysqld]
innodb_log_group_home=/var/lib/mysql
```

logical backup

A backup which contains a set of SQL statements. The statements can be used to recreate the databases.

LSN

Each InnoDB page (usually 16kb in size) contains a log sequence number, or LSN. The LSN is the system version number for the entire database. Each page’s LSN shows how recently it was changed.

my.cnf

The database server’s main configuration file. Most Linux distributions place it as `/etc/mysql/my.cnf` or `/etc/my.cnf`, but the location and name depends on the particular installation. Note that this is not the only way of configuring the server, some systems does not have one and rely on the command options to start the server and its default values.

MyISAM

Previous default storage engine for *MySQL* for versions prior to 5.5. It doesn’t fully support transactions but in some scenarios may be faster than InnoDB. Each table is stored on disk in 3 files: .frm, .MYD, .MYI.

physical backup

A backup that copies the data files.

point in time recovery

This method allows data to be restored to the state it was in any selected point of time.

prepared backup

A consistent set of backup data that is ready to be restored.

restore

Copies the database backups taken using the backup command to the original location or a different location. A restore returns data that has been either lost, corrupted, or stolen to the original condition at a specific point in time.

xbcrypt

To support encryption and decryption of the backups, a new tool xbcrypt was introduced to *Percona XtraBackup*. This utility has been modeled after the xbstream binary to perform encryption and decryption outside of *Percona XtraBackup*.

xbstream

To support simultaneous compression and streaming, *Percona XtraBackup* uses the xbstream format. For more information, see `--stream`

XtraDB

Percona XtraDB is an enhanced version of the InnoDB storage engine, designed to better scale on modern hardware. *Percona XtraDB* includes a variety of other features useful in high performance environments. It is fully backwards compatible, and so can be used as a drop-in replacement for standard InnoDB. More information here.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-08-01

10.13 Index of files created by Percona XtraBackup

- Information related to the backup and the server
    - `backup-my.cnf`

This file contains information to start the mini instance of InnoDB during the `xtrabackup --prepare`. This is **NOT** a backup of original `my.cnf`. The InnoDB configuration is read from the file `backup-my.cnf` created by innobackupex when the backup was made. `xtrabackup --prepare` uses InnoDB configuration from `backup-my.cnf` by default, or from `xtrabackup --defaults-file`, if specified. InnoDB configuration in this context means server variables that affect data format, i.e. `innodb_page_size` option, `innodb_log_block_size`, etc. Location-related variables, like `innodb_log_group_home_dir` or `innodb_data_file_path` are always ignored by `xtrabackup --prepare`, so preparing a backup always works with data files from the backup directory, rather than any external ones.

- `xtrabackup_checkpoints`

The type of the backup (e.g. full or incremental), its state (e.g. prepared) and the LSN range contained in it. This information is used for incremental backups. Example of the `xtrabackup_checkpoints` after taking a full backup:

```
text
    backup_type = full-backuped
    from_lsn = 0
    to_lsn = 15188961605
    last_lsn = 15188961605
```

Example of the `xtrabackup_checkpoints` after taking an incremental backup:

```
text
    backup_type = incremental
    from_lsn = 15188961605
    to_lsn = 15189350111
    last_lsn = 15189350111
```

- `xtrabackup_binlog_info`

The binary log file used by the server and its position at the moment of the backup. Result of the `SHOW MASTER STATUS`.

- `xtrabackup_binlog_pos_innodb`

The binary log file and its current position for *InnoDB* or XtraDB tables.

- `xtrabackup_binary`

The *xtrabackup* binary used in the process.

- `xtrabackup_logfile`

Contains data needed for running the: `xtrabackup --prepare`. The bigger this file is the `xtrabackup --prepare` process will take longer to finish.

- `<table_name>.delta.meta`

This file is going to be created when performing the incremental backup. It contains the per-table delta metadata: page size, size of compressed page (if the value is 0 it means the tablespace isn't compressed) and space id. Example of this file could looks like this:

```
text
  page_size = 16384
  zip_size = 0
  space_id = 0
```

- Information related to the replication environment (if using the `xtrabackup --slave-info` option):
    - `xtrabackup_slave_info`

The `CHANGE MASTER` statement needed for setting up a slave.

- Information related to the *Galera* and *Percona XtraDB Cluster* (if using the `xtrabackup --galera-info` option):
    - `xtrabackup_galera_info`

  Contains the values of `wsrep_local_state_uuid` and `wsrep_last_committed` status variables

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-07-19

10.14 Trademark policy

This Trademark Policy is to ensure that users of Percona-branded products or services know that what they receive has really been developed, approved, tested and maintained by Percona. Trademarks help to prevent confusion in the marketplace, by distinguishing one company's or person's products and services from another's.

Percona owns a number of marks, including but not limited to Percona, XtraDB, Percona XtraDB, XtraBackup, Percona XtraBackup, Percona Server, and Percona Live, plus the distinctive visual icons and logos associated with these marks. Both the unregistered and registered marks of Percona are protected.

Use of any Percona trademark in the name, URL, or other identifying characteristic of any product, service, website, or other use is not permitted without Percona's written permission with the following three limited exceptions.

First, you may use the appropriate Percona mark when making a nominative fair use reference to a bona fide Percona product.

Second, when Percona has released a product under a version of the GNU General Public License ("GPL"), you may use the appropriate Percona mark when distributing a verbatim copy of that product in accordance with the terms and conditions of the GPL.

Third, you may use the appropriate Percona mark to refer to a distribution of GPL-released Percona software that has been modified with minor changes for the sole purpose of allowing the software to operate on an operating system or hardware platform for which Percona has not yet released the software, provided that those third party changes do not affect the behavior, functionality, features, design or performance of the software. Users who acquire this Percona-branded software receive substantially exact implementations of the Percona software.

Percona reserves the right to revoke this authorization at any time in its sole discretion. For example, if Percona believes that your modification is beyond the scope of the limited license granted in this Policy or that your use of the Percona mark is detrimental to Percona, Percona will revoke this authorization. Upon revocation, you must immediately cease using the applicable Percona mark. If you do not immediately cease using the Percona mark upon revocation, Percona may take action to protect its rights and interests in the Percona mark. Percona does not grant any license to use any Percona mark for any other modified versions of Percona software; such use will require our prior written permission.

Neither trademark law nor any of the exceptions set forth in this Trademark Policy permit you to truncate, modify or otherwise use any Percona mark as part of your own brand. For example, if XYZ creates a modified version of the Percona Server, XYZ may not brand that modification as "XYZ Percona Server" or "Percona XYZ Server", even if that modification otherwise complies with the third exception noted above.

In all cases, you must comply with applicable law, the underlying license, and this Trademark Policy, as amended from time to time. For instance, any mention of Percona trademarks should include the full trademarked name, with proper spelling and capitalization, along with attribution of ownership to Percona LLC and/or its affiliates. For example, the full proper name for XtraBackup is Percona XtraBackup. However, it is acceptable to omit the word "Percona" for brevity on the second and subsequent uses, where such omission does not cause confusion.

In the event of doubt as to any of the conditions or exceptions outlined in this Trademark Policy, please contact trademarks@percona.com for assistance and we will do our very best to be helpful.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-06-24

10.15 Copyright and licensing information

10.15.1 Documentation licensing

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2023-02-15

10.16 Version Checking

Some Percona software contains “version checking” functionality which is a feature that enables Percona software users to be notified of available software updates to improve your environment security and performance. Alongside this, the version check functionality also provides Percona with information relating to which software versions you are running, coupled with the environment confirmation which the software is running within. This helps enable Percona to focus our development effort accordingly based on trends within our customer community.

The purpose of this document is to articulate the information that is collected, as well as to provide guidance on how to disable this functionality if desired.

10.16.1 Usage

Version Check was implemented in Percona Toolkit 2.1.4, and was enabled by default in version 2.2.1. Currently it is supported as a `--[no]version-check` option by a number of tools in Percona Toolkit, Percona XtraBackup, and Percona Monitoring and Management (PMM).

When launched with Version Check enabled, the tool that supports this feature connects to a Percona’s *version check service* via a secure HTTPS channel. It compares the locally installed version for possible updates, and also checks versions of the following software:

- Operating System
- Percona Monitoring and Management (PMM)
- MySQL
- Perl
- MySQL driver for Perl (DBD::mysql)
- Percona Toolkit

Then it checks for and warns about versions with known problems if they are identified as running in the environment.

Each version check request is logged by the server. Stored information consists of the checked system unique ID followed by the software name and version. The ID is generated either at installation or when the *version checking* query is submitted for the first time.

Note

Prior to version 3.0.7 of Percona Toolkit, the system ID was calculated as an MD5 hash of a hostname, and starting from Percona Toolkit 3.0.7 it is generated as an MD5 hash of a random number. Percona XtraBackup continues to use hostname-based MD5 hash.

As a result, the content of the sent query is as follows:

```
85624f3fb5d2af8816178ea1493ed41a;DBD::mysql;4.044
c2b6d625ef3409164cbf8af4985c48d3;MySQL;MySQL Community Server (GPL) 5.7.22-log
85624f3fb5d2af8816178ea1493ed41a;OS;Manjaro Linux
85624f3fb5d2af8816178ea1493ed41a;Percona::Toolkit;3.0.11-dev
85624f3fb5d2af8816178ea1493ed41a;Perl;5.26.2
```

10.16.2 Disabling Version Check

Although the *version checking* feature does not collect any personal information, you might prefer to disable this feature, either one time or permanently. To disable it one time, use `--no-version-check` option when invoking the tool from a Percona product which supports it. Here is a simple example which shows running pt-diskstats tool from the Percona Toolkit with *version checking* turned off:

```
pt-diskstats --no-version-check
```

Disabling *version checking* permanently can be done by placing `no-version-check` option into the configuration file of a Percona product (see correspondent documentation for exact file name and syntax). For example, in case of Percona Toolkit this can be done in a global configuration file `/etc/percona-toolkit/percona-toolkit.conf`:

```
# Disable Version Check for all tools:
no-version-check
```

In case of Percona XtraBackup this can be done in its configuration file in a similar way:

```
[xtrabackup]
no-version-check
```

10.16.3 Frequently Asked Questions

Why is this functionality enabled by default?

We believe having this functionality enabled improves security and performance of environments running Percona Software and it is good choice for majority of the users.

Why not rely on Operating System’s built in functionality for software updates?

In many environments the Operating Systems repositories may not carry the latest version of software and newer versions of software often installed manually, so not being covered by operating system wide check for updates.

Why do you send more information than just the version of software being run as a part of version check service?

Compatibility problems can be caused by versions of various components in the environment, for example problematic versions of Perl, DBD or MySQL could cause operational problems with Percona Toolkit.

Contact us

For free technical help, visit the Percona Community Forum.

To report bugs or submit feature requests, open a JIRA ticket.

For paid support and managed or consulting services , contact Percona Sales.

Last update: 2022-12-06