
TwinDB Backup Documentation

Release 2.15.3

TwinDB Development Team

Jan 14, 2018

Contents

1	TwinDB Backup	3
1.1	Features	3
1.2	Credits	4
2	Installation	5
2.1	Requirements	5
2.2	Stable release	5
2.3	From sources	5
3	Usage	7
3.1	Configuration	7
3.2	Email notification	9
4	twindb_backup	11
4.1	twindb_backup package	11
5	Contributing	33
5.1	Types of Contributions	33
5.2	Get Started!	34
5.3	Pull Request Guidelines	35
6	Credits	37
6.1	Development Lead	37
6.2	Contributors	37
7	History	39
8	Indices and tables	41
	Python Module Index	43

Contents:

TwinDB Backup is a multipurpose tool for backing up MySQL and file system. It can store backup copies on a remote SSH server or Amazon S3.

The tool can easily restore the backup copies.

Read full documentation on <https://twindb-backup.readthedocs.io>.

1.1 Features

twindb-backup key features:

- Files/directories backups
- MySQL backups
- Incremental MySQL backups
- Encrypting backup copies

twindb-backup store backups on:

- Remote SSH server
- Amazon S3
- Optionally save local copy


Other features:

- Retention policy defines how many hourly/daily/weekly/monthly/yearly copies to keep
- Separate retention policy for remote and local backup copies
- Supports non-impacting Percona XtraDB Cluster backups
- Email notifications
- cron configuration comes with a package


1.1.1 How do I get set up?

twindb-backup is distributed via package repositories. See installation instruction on <https://packagecloud.io/twindb/main/install>. Once the repository for your operating system is configured, install the `twindb-backup` package.

On CentOS and RedHat

```
# curl -s https://packagecloud.io/install/repositories/twindb/main/script.rpm.sh |   
↪ sudo bash  
# yum install twindb-backup
```

On Debian and Ubuntu

```
# curl -s https://packagecloud.io/install/repositories/twindb/main/script.deb.sh |   
↪ sudo bash  
# apt-get install twindb-backup
```

1.1.2 Configuration

Configuration is stored in `/etc/twindb/twindb-backup.cfg`. See <http://twindb-backup.readthedocs.io/en/master/usage.html> for more details.

1.2 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

2.1 Requirements

TwinDB Backup package will pull all necessary dependencies except `aws` tool. We recommend to install it from PyPi.

```
# pip install awscli
```

2.2 Stable release

To install TwinDB Backup, run this command in your terminal:

```
# yum install https://twindb.com/twindb-release-latest.noarch.rpm
# yum install twindb-backup
```

This is the preferred method to install TwinDB Backup, as it will always install the most recent stable release.

2.3 From sources

The sources for TwinDB Backup can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/twindb/backup
```

Or download the tarball:

```
$ curl -OL https://github.com/twindb/backup/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ make install
```

3.1 Configuration

Once the `twindb-backup` package is installed you need to edit configuration file `/etc/twindb/twindb-backup.cfg`.

Let's review each configuration section.

3.1.1 Source

[`source`] section defines what to backup.

`twindb-backup` supports backing up local directories and MySQL database.

In `backup_dirs` you specify which directories to backup. Each directory is separated by a white space.

`backup_mysql` tells whether the tool backups MySQL or not.

```
[source]
backup_dirs=/etc /root /home
backup_mysql=no
```

3.1.2 Destination

[`destination`] specifies where to store backup copies.

`backup_destination` can be either `ssh` (if you want to store backups on a remote SSH server) or `s3` (if you want to store backups in Amazon S3).

In the optional `keep_local_path` you can specify a local path where the tool will store a local copy of the backup. It's useful if you want to stream a MySQL backup to S3 and would like to keep a local copy as well.

```
[destination]
backup_destination=ssh
keep_local_path=/var/backup/local
```

3.1.3 Amazon S3

In [s3] section you specify Amazon credentials as well as an S3 bucket where to store backups.

```
[s3]

# S3 destination settings

AWS_ACCESS_KEY_ID=XXXXX
AWS_SECRET_ACCESS_KEY=YYYYY
AWS_DEFAULT_REGION=us-east-1
BUCKET=twindb-backups
```

3.1.4 SSH Settings

If your backup destination is an SSH server, you specify ssh parameters in [ssh] section. It is assumed you configured [SSH keys authentication](#). It will not work if you need to enter a password to login to backup_host.

```
[ssh]

backup_host=127.0.0.1
backup_dir=/tmp/backup
ssh_user=root
ssh_key=/root/.ssh/id_rsa
```

3.1.5 MySQL

XtraBackup needs to connect to MySQL. In [mysql] section you specify a defaults file with user and password.

```
[mysql]
mysql_defaults_file=/etc/twindb/my.cnf
```

3.1.6 Encryption

The tool uses [GPG](#) for encrypting/decrypting backup copies. If you want to enable encryption add [gpg] section to the configuration file. It's your responsibility to generate and manage the encryption key.

```
[gpg]
keyring = /root/.gnupg/pubring.gpg
secret_keyring = /root/.gnupg/secring.gpg
recipient = backupuser@youdomain.com
```

3.1.7 Retention Policy

In [retention] section you specify how many copies you want to keep on the remote storage (s3 or ssh).

```
[retention]
hourly_copies=24
daily_copies=7
weekly_copies=4
monthly_copies=12
yearly_copies=3
```

3.1.8 Local Retention Policy

if `keep_local_path` is defined in *Destination* the tool will apply `[retention_local]` on the local copies.

```
[retention_local]
hourly_copies=1
daily_copies=1
weekly_copies=0
monthly_copies=0
yearly_copies=0
```

3.1.9 Running Intervals

By default **twindb-backup** will run *hourly*, *daily*, *weekly*, *monthly* and *yearly*. If you would like to skip some runs `[intervals]` section is the right place to do so.

```
[intervals]
run_hourly=yes
run_daily=yes
run_weekly=yes
run_monthly=yes
run_yearly=yes
```

3.2 Email notification

The RPM package installs a cron job. If a backup job fails it will send standard error output to the specified email. The email address is specified in the cron configuration file `/etc/cron.d/twindb-backup`.

```
MAILTO=nagios@twindb.com
@hourly root twindb-backup backup hourly
@daily root twindb-backup backup daily
@weekly root twindb-backup backup weekly
@monthly root twindb-backup backup monthly
@yearly root twindb-backup backup yearly
```


4.1 twindb_backup package

4.1.1 Subpackages

twindb_backup.cache package

Submodules

twindb_backup.cache.cache module

Backup copy cache

class twindb_backup.cache.cache.Cache (*path*)

Bases: object

Class implements local cache to save full backup copies

add (*path*, *key=None*)

Add directory to cache. The directory may be a full or relative path with backup copy. The directory name must match with a file name of the backup copy. If backup copy is `/path/to/backups/master1/daily/mysql/mysql-2017-05-13_22_04_06.xbstream.gz`, then the directory can be something like `/var/tmp/mysql-2017-05-13_22_04_06.xbstream.gz/`.

Let's say we want to add `/var/tmp/mysql-2017-05-13_22_04_06.xbstream.gz/` to the cache in `/var/tmp/cache`. Then this method will create directory `/var/tmp/cache/mysql-2017-05-13_22_04_06.xbstream.gz/`.

If you want to save directory `/var/tmp/foo` in cache under a key name `mysql-2017-05-13_22_04_06.xbstream.gz` you need to specify the key e.g. `add('/var/tmp/cache', 'mysql-2017-05-13_22_04_06.xbstream.gz')`

Parameters

- **path** (*str*) – full or relative path

- **key** – if specified the directory will be added as this key name in the cache

Raise CacheException if errors

purge ()

Remove all entries from the cache

restore_in (*item*, *path*)

Restore backup copy item in path.

Parameters

- **item** (*str*) – directory in the cache
- **path** (*str*) – directory where to restore item

exception `twindb_backup.cache.cache.CacheException`

Bases: `exceptions.Exception`

Cache errors

Module contents

`twindb_backup.destination` package

Submodules

`twindb_backup.destination.base_destination` module

Module defines Base destination class and destination exception(s).

class `twindb_backup.destination.base_destination.BaseDestination`

Bases: `object`

Base destination class

basename (*filename*)

Basename of backup copy

Parameters *filename* –

Returns

delete (*obj*)

Delete object from the destination

Parameters *obj* –

Returns

find_files (*prefix*, *run_type*)

Find files

Parameters

- **prefix** –
- **run_type** –

Returns

get_full_copy_name (*file_path*)

For a given backup copy find a parent. If it's a full copy then return itself

Parameters `file_path` –

Returns

get_latest_backup ()

Get latest backup path

list_files (*prefix*, *recursive=False*)

List files

Parameters

- **prefix** –
- **recursive** –

save (*handler*, *name*)

Save the given stream.

Parameters

- **handler** (*file*) – Incoming stream.
- **name** (*str*) – Save stream as this name.

Raise DestinationError if any error

status (*status=None*)

Read or save backup status. Status is a dictionary with available backups and their properties. If status is None the function will read status from the remote storage. Otherwise it will store the status remotely.

Parameters **status** – dictionary like

```
{
  'hourly': [
    {
      'filename': '/remote/path',
      'binlog': 'mysql-bin.000001',
      'position': 43670
    }
  ]
}
```

Returns dictionary with the status

twindb_backup.destination.local module

Module defines Local destination.

class `twindb_backup.destination.local.Local` (*path=None*)

Bases: `twindb_backup.destination.base_destination.BaseDestination`

Local destination class.

delete (*obj*)

find_files (*prefix*, *run_type*)

static get_stream (*path*)

Get a PIPE handler with content of the backup copy streamed from the destination

Returns

list_files (*prefix, recursive=False*)

save (*handler, name*)

Read from handler and save it on local storage

Parameters

- **name** – store backup copy as this name
- **handler** –

Returns exit code

twindb_backup.destination.s3 module

Module for S3 destination.

class twindb_backup.destination.s3.**AWSAuthOptions** (*access_key_id, secret_access_key, default_region='us-east-1'*)

Bases: object

Class to store AWS credentials

class twindb_backup.destination.s3.**S3** (*bucket, aws_options, hostname='build-6554706-project-68257-twindb-backup'*)

Bases: *twindb_backup.destination.base_destination.BaseDestination*

S3 destination class.

Parameters

- **bucket** (*str*) – Bucket name.
- **aws_options** (*AWSAuthOptions*) – AWS credentials.
- **hostname** – Hostname of a host where a backup is taken from.

create_bucket ()

Creates the bucket in s3 that will store the backups.

Raises **S3DestinationError** – if failed to create the bucket.

delete (*obj*)

Deletes an S3 object.

Parameters **obj** (*str*) – Key of S3 object.

Raises **S3DestinationError** – if failed to delete object.

delete_all_objects ()

Delete all objects from S3 bucket.

Raises **S3DestinationError** – if failed to delete objects from the bucket.

delete_bucket (*force=False*)

Delete the bucket in s3 that was storing the backups.

Parameters **force** (*bool*) – If the bucket is non-empty then delete the objects before deleting the bucket.

Raises **S3DestinationError** – if failed to delete the bucket.

find_files (*prefix, run_type*)

Find files with common prefix and given run type.

Parameters

- **prefix** (*str*) – Common prefix.
- **run_type** (*str*) – daily, hourly, etc

Returns list of file names

Return type list(str)

Raises **S3DestinationError** – if failed to find files.

get_stream (**args, **kws*)

Get a PIPE handler with content of the backup copy streamed from the destination.

Returns Stream with backup copy

Return type generator

Raises **S3DestinationError** – if failed to stream a backup copy.

static get_transfer_config ()

Build Transfer config

Returns Transfer config

Return type boto3.s3.transfer.TransferConfig

list_files (*prefix, recursive=False*)

List files in the destination that have common prefix.

Parameters

- **prefix** (*str*) – Common prefix.
- **recursive** – Does nothing for this class.

Returns sorted list of file names

Return type list(str)

Raises **S3DestinationError** – if failed to list files.

save (*handler, name*)

Read from handler and save it to Amazon S3

Parameters

- **name** – save backup copy in a file with this name
- **handler** – stdout handler from backup source

Raises **S3DestinationError** – if failed to save the backup.

setup_s3_client ()

Creates an authenticated s3 client.

Returns S3 client instance.

Return type botocore.client.BaseClient

share (*s3_url*)

Share S3 file and return public link

Parameters **s3_url** (*str*) – S3 url

Returns Public url

Return type str

Raises **S3DestinationError** – if failed to share object.

static validate_client_response (*response*)

Validates the response returned by the client. Raises an exception if the response code is not 200 or 204

Parameters *response* (*dict*) – The response that needs to be validated.

Raises **S3DestinationError** – if response from S3 is invalid.

class `twindb_backup.destination.s3.S3FileAccess`

Bases: `object`

Access modes for S3 files

private = 'private'

public_read = 'public-read'

`twindb_backup.destination.ssh` module

Module for SSH destination.

class `twindb_backup.destination.ssh.Ssh` (*ssh_connect_info*=<`twindb_backup.destination.ssh.SshConnectInfo` object>, *remote_path*=None, *hostname*='build-6554706-project-68257-twindb-backup')

Bases: `twindb_backup.destination.base_destination.BaseDestination`

SSH destination class

Parameters

- **ssh_connect_info** (`SshConnectInfo`) – SSH connection info
- **remote_path** – Path to store backup
- **hostname** – Hostname

client

Return client

delete (*obj*)

Delete file by path

Parameters *obj* – path to a remote file.

ensure_tcp_port_listening (*port*, *wait_timeout*=10)

Check that tcp port is open and ready to accept connections. Keep checking up to *wait_timeout* seconds.

Parameters

- **port** (*int*) – TCP port that is supposed to be listening.
- **wait_timeout** (*int*) – wait this many seconds until the port is ready.

Returns True if the TCP port is listening.

Return type `bool`

execute_command (*cmd*, *quiet*=False)

Execute ssh command

Parameters

- **cmd** (*str*) – Command for execution

- **quiet** – If True don't print errors

Returns Handlers of stdin, stdout and stderr

Return type tuple

find_files (*prefix, run_type*)

Find files by prefix

Parameters

- **prefix** (*str*) – Path
- **run_type** (*str*) – Run type for search

Returns List of files

Return type list

get_stream (**args, **kwargs*)

Get a PIPE handler with content of the backup copy streamed from the destination

Parameters **path** (*str*) – Path to file

Returns Standard output.

host

IP address of the destination.

list_files (*prefix, recursive=False*)

Get list of file by prefix

Parameters

- **prefix** (*str*) – Path
- **recursive** (*bool*) – Recursive return list of files

Returns List of files

Return type list

netcat (*command, port=9990*)

Run netcat on the destination pipe it to a given command.

save (*handler, name*)

Read from handler and save it on remote ssh server

Parameters

- **name** – relative path to a file to store the backup copy.
- **handler** – stream with content of the backup.

class twindb_backup.destination.ssh.**SshConnectInfo** (*host='127.0.0.1', port=22, key='/root/.id_rsa', user='root'*)

Bases: object

Options for SSH connection

Module contents

twindb_backup.exporter package

Submodules

twindb_backup.exporter.base_exporter module

Module defines base exporter class.

```
class twindb_backup.exporter.base_exporter.BaseExporter
```

Bases: object

Base exporter class

```
export (category, measure_type, data)
```

Send data to server

```
class twindb_backup.exporter.base_exporter.ExportCategory
```

Bases: object

Category of export data: files or mysql

```
files = 0
```

```
mysql = 1
```

```
class twindb_backup.exporter.base_exporter.ExportMeasureType
```

Bases: object

Type of measure time: backup or restore

```
backup = 0
```

```
restore = 1
```

twindb_backup.exporter.datadog_exporter module

Module defines DataDog exporter class.

```
class twindb_backup.exporter.datadog_exporter.DataDogExporter (app_key,  
api_key)
```

Bases: *twindb_backup.exporter.base_exporter.BaseExporter*

DataDog exporter class

```
export (category, measure_type, data)
```

Export data to DataDog :param category: Data meant :param measure_type: Type of measure :param data:
Data to posting :raise: DataDogExporterError if data is invalid

twindb_backup.exporter.exceptions module

Module for exporters exceptions

```
exception twindb_backup.exporter.exceptions.BaseExporterError
```

Bases: exceptions.Exception

General exporters error

exception `twindb_backup.exporter.exceptions.DataDogExporterError`

Bases: `twindb_backup.exporter.exceptions.BaseExporterError`

DataDog exporters error

Module contents

twindb_backup.modifiers package

Submodules

twindb_backup.modifiers.base module

Module defines Modifier() base class and its errors.

class `twindb_backup.modifiers.base.Modifier` (*input_stream*)

Bases: `object`

Base Modifier class

callback (***kwargs*)

Method that will be called after the stream ends

get_stream (**args, **kws*)

Apply modifier and return output stream. The Base modifier does nothing, so it will return the input stream without modifications

Returns output stream handle

exception `twindb_backup.modifiers.base.ModifierException`

Bases: `exceptions.Exception`

Base Exception for Modifier error

twindb_backup.modifiers.gpg module

Module defines modifier that implements asymmetric encryption with gpg

class `twindb_backup.modifiers.gpg.Gpg` (*input_stream*, *recipient*, *keyring*, *secret_keyring=None*)

Bases: `twindb_backup.modifiers.base.Modifier`

Asymmetric encryption

get_stream (**args, **kws*)

Encrypt the input stream and return it as the output stream

Returns output stream handle

Raise `OSError` if failed to call the gpg command

revert_stream ()

Decrypt the input stream and return it as the output stream

Returns output stream handle

Raise `OSError` if failed to call the gpg command

twindb_backup.modifiers.gzip module

Module defines modifier that compresses a stream with gzip

class `twindb_backup.modifiers.gzip.Gzip` (*input_stream*)

Bases: `twindb_backup.modifiers.base.Modifier`

Modifier that compresses the input_stream with gzip.

get_stream (**args, **kws*)

Compress the input stream and return it as the output stream

Returns output stream handle

Raise OSError if failed to call the gzip command

revert_stream ()

Decompress the input stream and return it as the output stream

Returns output stream handle

Raise OSError if failed to call the gpg command

twindb_backup.modifiers.keeplocal module

Module defines modifier that save a stream on the local file system

class `twindb_backup.modifiers.keeplocal.KeepLocal` (*input_stream, local_path*)

Bases: `twindb_backup.modifiers.base.Modifier`

KeepLocal() class saves a copy of the stream on the local file system. It doesn't alter the stream.

callback (***kwargs*)

get_stream (**args, **kws*)

Save a copy of the input stream and return the output stream

Returns output stream handle

Raise OSError if failed to call the tee command

Module contents

twindb_backup.source package

Submodules

twindb_backup.source.base_source module

Module defines base source class.

class `twindb_backup.source.base_source.BaseSource` (*run_type*)

Bases: `object`

Base source for backup

get_prefix ()

Get prefix of the backup copy. It includes hostname and run type.

Returns Backup name prefix like 'db-10/daily'

get_stream()
Get backup stream in a handler

run_type = None

suffix
Backup file name suffix

twindb_backup.source.file_source module

Module defines File source class for backing up local directories.

class `twindb_backup.source.file_source.FileSource` (*path, run_type*)
Bases: `twindb_backup.source.base_source.BaseSource`

FileSource class

apply_retention_policy (*dst, config, run_type*)
Apply retention policy

get_name()
Generate relative destination file name

Returns file name

get_stream (**args, **kws*)
Get a PIPE handler with content of the source

Returns

media_type
Get media type. Media type is a general term that describes what you backup. For directories `media_type` is 'file'.

Returns 'file'

twindb_backup.source.mysql_source module

Module defines MySQL source class for backing up local MySQL.

class `twindb_backup.source.mysql_source.MySQLConnectInfo` (*defaults_file, connect_timeout=10, cursor=<class 'pymysql.cursors.DictCursor'>, hostname='127.0.0.1'*)

Bases: `object`

MySQL connection details

class `twindb_backup.source.mysql_source.MySQLSource` (*mysql_connect_info, run_type, full_backup, dst=None*)
Bases: `twindb_backup.source.base_source.BaseSource`

MySQLSource class

apply_retention_policy (*dst, config, run_type, status*)
Delete old backup copies.

Parameters

- **dst** (`BaseDestination`) – Destination where the backups are stored.

- **config** (*ConfigParser.ConfigParser*) – Tool configuration
- **run_type** (*str*) – Run type.
- **status** (*dict*) – Backups status.

Returns Updated status.

Return type dict

binlog_coordinate

Binary log coordinate up to that backup is taken

Returns file name and position

Return type tuple

datadir

Return datadir path on MySQL server

disable_wsrep_desync ()

Wait till wsrep_local_recv_queue is zero and disable wsrep_local_recv_queue then

enable_wsrep_desync ()

Try to enable wsrep_desync

Returns True if wsrep_desync was enabled. False if not supported

full

Check if the backup copy is a full copy.

Returns True if it's a full copy.

Return type bool

galera

Check if local MySQL instance is a Galera cluster

Returns True if it's a Galera.

Return type bool

static get_binlog_coordinates (*err_log_path*)

Parse innobackupex log and return binary log coordinate

Parameters **err_log_path** (*str*) – path to the innobackupex log

Returns Binlog coordinate.

Return type tuple

get_connection (**args, **kws*)

Connect to MySQL host and yield a connection.

Returns MySQL connection

static get_my_cnf ()

Get generator that spits out my.cnf files

Returns File name and content of MySQL config file.

Return type tuple

get_name ()

Generate relative destination file name

Returns file name

get_stream (*args, **kws)

Get a PIPE handler with content of the source :return:

incremental

Check if the backup copy is an incremental copy.

Returns True if it's an incremental copy.

Return type bool

is_galera ()

Check if local MySQL instance is a Galera cluster

Returns True if it's a Galera.

Return type bool

lsn

The latest LSN of the taken backup :return: LSN :rtype: int

parent

Get name of the parent copy.

Returns backup name of the parent copy or its own name if the copy is a full copy.

Return type str

parent_lsn

LSN of the parent backup copy.

Returns LSN of the parent or its own LSN if the backup copy is a full copy.

status

Backup status on a destination

Returns Backups status

Return type dict

type

Get backup copy type - full or incremental

Returns 'full' or 'incrmental'

Return type str

wsrep_provider_version

Parse Galera version from wsrep_provider_version.

Returns Galera version

Return type str

Module contents

twindb_backup.ssh package

Submodules

twindb_backup.ssh.client module

Module that implements SSH client.

class `twindb_backup.ssh.client.SshClient` (*ssh_connect_info*)

Bases: `object`

SSH client class. Allows to connect to a remote SSH server and execute commands on it.

execute (*cmd*, *quiet=False*)

Execute a command on a remote SSH server.

Parameters

- **cmd** (*str*) – Command for execution.
- **quiet** – if quiet is True don't print error messages

Returns Handlers of stdin, stdout and stderr

Return type tuple

Raises `SshDestinationError` – if any error

get_remote_handlers (**args*, ***kws*)

Get remote stdin, stdout and stderr handler

Parameters **cmd** (*str*) – Command for execution

Returns Remote stdin, stdout and stderr handler

Return type tuple(generator, generator, generator)

Raises `SshDestinationError` – if any error

session (**args*, ***kws*)

Get SSH session

Return type generator

Returns SSH session

twindb_backup.ssh.exceptions module

SSH Client Exceptions.

exception `twindb_backup.ssh.exceptions.SshClientException`

Bases: `exceptions.Exception`

Exception in `SshClient`

Module contents

4.1.2 Submodules

4.1.3 `twindb_backup.backup` module

Module that parses config file, builds a modifiers chain and fires backup jobs.

`twindb_backup.backup.backup_everything` (*run_type*, *config*)

Run backup job

Parameters

- **run_type** (*str*) – hourly, daily, etc
- **config** (`ConfigParser.ConfigParser`) – `ConfigParser` instance

`twindb_backup.backup.backup_files` (*run_type*, *config*)
Backup local directories

Parameters

- **run_type** (*str*) – Run type
- **config** (*ConfigParser.ConfigParser*) – Configuration

`twindb_backup.backup.backup_mysql` (*run_type*, *config*)
Take backup of local MySQL instance

Parameters

- **run_type** (*str*) – Run type
- **config** (*ConfigParser.ConfigParser*) – Tool configuration

Returns None

`twindb_backup.backup.prepare_status` (*dst*, *src*, *run_type*, *src_name*, *backup_start*)
Prepare status for update

`twindb_backup.backup.run_backup_job` (*cfg*, *run_type*, *lock_file*='var/run/twindb-backup.lock')
Grab a lock waiting up to allowed timeout and start backup jobs

Parameters

- **cfg** (*ConfigParser.ConfigParser*) – Tool configuration
- **run_type** (*str*) – Run type
- **lock_file** (*str*) – File used as a lock

`twindb_backup.backup.set_open_files_limit` ()
Detect maximum supported number of open file and set it

`twindb_backup.backup.timeout` (**args*, ***kws*)
Implement timeout

Parameters **seconds** (*int*) – timeout in seconds

4.1.4 twindb_backup.cli module

Entry points for twindb-backup tool

4.1.5 twindb_backup.clone module

Module defines clone feature

`twindb_backup.clone.clone_mysql` (*cfg*, *source*, *destination*, *replication_user*, *replication_password*, *netcat_port*=9990, *compress*=False)
Clone mysql backup of remote machine and stream it to slave

4.1.6 twindb_backup.configuration module

Module to process configuration file.

`twindb_backup.configuration.get_destination` (*config*, *hostname*='build-6554706-project-68257-twindb-backup')
Read config and return instance of Destination class.

Parameters

- **config** (*ConfigParser.ConfigParser*) – Tool configuration.
- **hostname** (*str*) – Local hostname.

Returns Instance of destination class.

Return type *BaseDestination*

`twindb_backup.configuration.get_export_transport (config)`
Read config and return export transport instance

Parameters **config** – Config file

Returns Instance of export transport, if it set

Raise `NotImplementedError`, if transport isn't implemented

4.1.7 `twindb_backup.export` module

Module to process export

`twindb_backup.export.export_info (cfg, data, category, measure_type)`
Export data to service

Parameters

- **cfg** – Config file
- **data** – Data
- **category** – Category of data
- **measure_type** – Type of measure
- **category** – Category

4.1.8 `twindb_backup.ls` module

Module that works with list of backup copies

`twindb_backup.ls.list_available_backups (config)`
Print known backup copies on a destination specified in the configuration.

Parameters **config** (*ConfigParser.ConfigParser*) – tool configuration

4.1.9 `twindb_backup.restore` module

Module that restores backup copies.

`twindb_backup.restore.gen_grastate (path, version, uuid, seqno)`
Generate and save grastate file.

Parameters

- **path** – Path to grastate file.
- **version** – Galera version from grastate.dat.
- **uuid** – UUID from grastate.dat.
- **seqno** – seqno from grastate.dat.

`twindb_backup.restore.get_backup_type(status, key)`
 Get backup type - full or incremental - from the backup status.

Parameters

- **status** (*dict*) – backup status
- **key** (*str*) – backup name

Returns Backup type or None if backup name is not found

Return type `str`

`twindb_backup.restore.get_galera_version(status, key)`
 Get Galera version from backup status

Parameters

- **status** (*dict*) – backup status
- **key** (*str*) – backup name

Returns Galera version or None if not listed in the status

Return type `str`

`twindb_backup.restore.get_my_cnf(status, key)`
 Get MySQL config from the status.

Parameters

- **status** (*dict*) – Backup status.
- **key** (*str*) – Backup name.

Returns Content of my.cnf or None if not found

Return type `str`

`twindb_backup.restore.restore_from_file(config, backup_copy, dst_dir)`
 Restore a directory from a backup copy in the directory

Parameters

- **config** (*ConfigParser.ConfigParser*) – Tool configuration.
- **backup_copy** (*str*) – Backup name.
- **dst_dir** (*str*) – Path to destination directory. Must exist and be empty.

`twindb_backup.restore.restore_from_mysql(config, backup_copy, dst_dir, tmp_dir=None, cache=None)`

Restore MySQL datadir in a given directory

Parameters

- **config** (*ConfigParser.ConfigParser*) – Tool configuration.
- **backup_copy** (*str*) – Backup copy name.
- **dst_dir** (*str*) – Destination directory. Must exist and be empty.
- **tmp_dir** (*str*) – Path to temp directory
- **cache** (*Cache*) – Local cache object.

`twindb_backup.restore.restore_from_mysql_full(stream, dst_dir, config, redo_only=False)`
 Restore MySQL datadir from a backup copy

Parameters

- **stream** – Generator that provides backup copy
- **dst_dir** (*str*) – Path to destination directory. Must exist and be empty.
- **config** (*ConfigParser.ConfigParser*) – Tool configuration.
- **redo_only** (*bool*) – True if the function has to do final apply of the redo log. For example, if you restore backup from a full copy it should be False. If you restore from incremental copy and you restore base full copy redo_only should be True.

Returns If success, return True

Return type bool

`twindb_backup.restore.restore_from_mysql_incremental` (*stream*, *dst_dir*, *config*,
tmp_dir=None)

Restore MySQL datadir from an incremental copy.

Parameters

- **stream** – Generator that provides backup copy
- **dst_dir** (*str*) – Path to destination directory. Must exist and be empty.
- **config** (*ConfigParser.ConfigParser*) – Tool configuration.
- **tmp_dir** (*str*) – Path to temp dir

Returns If success, return True

Return type bool

`twindb_backup.restore.update_grastate` (*dst_dir*, *status*, *key*)

If xtrabackup_galera_info exists in the destination directory then parse it and generate grastate.dat file.

Parameters

- **dst_dir** (*str*) – Path to destination directory.
- **status** (*dict*) – Backup status
- **key** (*str*) – Backup name

4.1.10 twindb_backup.share module

Module that works with sharing backups

`twindb_backup.share.share` (*config*, *s3_url*)

Function for generate make public file and get public url

Parameters

- **config** – Config file
- **s3_url** (*str*) – S3 url to file

Raise TwinDBBackupError

4.1.11 twindb_backup.util module

Module with helper functions

`twindb_backup.util.empty_dir` (*path*)

Remove all files are directories in path

Parameters `path` (*str*) – Path to directory to be emptied.

`twindb_backup.util.ensure_empty` (*path*)

Check if a given directory is empty and exit if not.

Parameters `path` (*str*) – path to directory

`twindb_backup.util.get_hostname_from_backup_copy` (*backup_copy*)

Backup copy includes hostname where the backup was taken from. The function extracts the hostname from the backup name.

Parameters `backup_copy` (*str*) – Backup copy name.

Returns Hostname where the backup was taken from.

Return type `str`

`twindb_backup.util.kill_children` ()

Kill child process

`twindb_backup.util.mkdir_p` (*path*)

Emulate `mkdir -p`

Parameters `path` (*str*) – Directory path.

`twindb_backup.util.run_command` (**args, **kws*)

Run shell command locally

Parameters

- `command` (*list*) – Command to run
- `ok_non_zero` (*bool*) – Don't consider non-zero exit code as an error.

Returns file object with stdout as generator to use with `with`

`twindb_backup.util.split_host_port` (*host_port*)

Splits a string of host and port separated by a semicolon.

Parameters `host_port` – host or host:port. Allowed values are like 10.20.31.1:3306 or just 10.20.31.1

Returns a tuple with host and port. If only address is specified it'll return (address, None). If `host_port` is None it will return (None, None)

Return type `tuple`

4.1.12 `twindb_backup.verify` module

Module that verify backup copies.

`twindb_backup.verify.edit_backup_my_cnf` (*dst_path*)

Removed options from config(besides MySQL 5.7.8)

`twindb_backup.verify.verify_mysql_backup` (*config, dst_path, backup_copy, hostname=None*)

Restore mysql backup and measure time

4.1.13 Module contents

TwinDB Backup module.

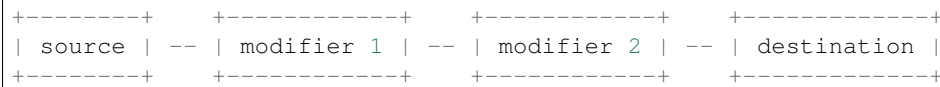
The module is a core of twindb-backup tool. It includes backup and restore functionality. The module takes a backup from something defined in a source class and saves the backup copy in something defined in a destination class.

The source class inherits from BaseSource() from twindb_backup.source.base_source.py. The source class must define get_stream() method that yields a file object that is used for next classes. Typical classes are FileSource() to backup files and directories, MySQLSource() to backup MySQL.

The destination class inherits from BaseDestination(). This is where you store backups. The destination class must define save() method that takes an input stream and saves it somewhere. Examples of the destination class are S3(), Ssh().

There are modifier classes. The modifier class sits in the middle between the source and the destination and does something with a stream before the stream is saved. The modifier class may save a local copy (KeepLocal()) or encrypt the stream or else. The modifier class inherits Modifier()

The backup process may be depicted as a chain of modifiers with the source in the head and the destination in the tail.



class twindb_backup.LessThanFilter (*exclusive_maximum, name=""*)

Bases: logging.Filter

Filters out log messages of a lower level.

filter (*record*)

exception twindb_backup.TwinDBBackupError

Bases: exceptions.Exception

Class for script errors

twindb_backup.delete_local_files (*dir_backups, keep_copies*)

Deletes local backup copies based on given retention number.

Parameters

- **dir_backups** (*str*) – directory with backup copies
- **keep_copies** (*int*) – how many to keep

Returns None

twindb_backup.get_directories_to_backup (*config*)

Get directories to backup from a config file

Parameters **config** (*ConfigParser.ConfigParser*) – instance of ConfigParser()

Returns list of strings

twindb_backup.get_files_to_delete (*all_files, keep_copies*)

If you give it a list of files and number of how many you'd like to keep the function will return files that need to be deleted

Parameters

- **all_files** (*list*) – list of strings
- **keep_copies** (*int*) – number of copied to keep

Returns list of strings (files) to delete

Return type list

`twindb_backup.get_timeout(run_type)`

Get timeout for a each run type - daily, hourly etc

Parameters `run_type` (*str*) – Run type

Returns Number of seconds the tool allowed to wait until other instances finish

Return type int

`twindb_backup.save_measures(start_time, end_time, log_path='/var/log/twindb-backup-measures.log')`

Save backup measures to log file

`twindb_backup.setup_logging(logger, debug=False)`

Configures logging for the module

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/twindb/backup/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

TwinDB Backup could always use more documentation, whether as part of the official TwinDB Backup docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/twindb/backup/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up TwinDB Backup for local development.

1. Fork the TwinDB Backup repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/backup.git twindb_backup
```

3. Install your local copy into a virtualenv:

```
$ make virtualenv
$ source env/bin/activate
$ make bootstrap
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ make test-all
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

- The pull request should include tests.
- If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
- The pull request should work for Python 2.6 and 2.7.
- Check https://travis-ci.org/twindb/backup/pull_requests and make sure that the tests pass for all supported Python versions.

6.1 Development Lead

- TwinDB Development Team <dev@twindb.com>

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

History

See [GitHub commits history](#).

CHAPTER 8

Indices and tables

- `genindex`
- `modindex`
- `search`

t

- twindb_backup, 29
- twindb_backup.backup, 24
- twindb_backup.cache, 12
- twindb_backup.cache.cache, 11
- twindb_backup.cli, 25
- twindb_backup.clone, 25
- twindb_backup.configuration, 25
- twindb_backup.destination, 18
- twindb_backup.destination.base_destination,
12
- twindb_backup.destination.local, 13
- twindb_backup.destination.s3, 14
- twindb_backup.destination.ssh, 16
- twindb_backup.export, 26
- twindb_backup.exporter, 19
- twindb_backup.exporter.base_exporter,
18
- twindb_backup.exporter.datadog_exporter,
18
- twindb_backup.exporter.exceptions, 18
- twindb_backup.ls, 26
- twindb_backup.modifiers, 20
- twindb_backup.modifiers.base, 19
- twindb_backup.modifiers.gpg, 19
- twindb_backup.modifiers.gzip, 20
- twindb_backup.modifiers.keeplocal, 20
- twindb_backup.restore, 26
- twindb_backup.share, 28
- twindb_backup.source, 23
- twindb_backup.source.base_source, 20
- twindb_backup.source.file_source, 21
- twindb_backup.source.mysql_source, 21
- twindb_backup.ssh, 24
- twindb_backup.ssh.client, 23
- twindb_backup.ssh.exceptions, 24
- twindb_backup.util, 28
- twindb_backup.verify, 29

A

add() (twindb_backup.cache.cache.Cache method), 11
 apply_retention_policy() (twindb_backup.source.file_source.FileSource method), 21
 apply_retention_policy() (twindb_backup.source.mysql_source.MySQLSource method), 21
 AWSAuthOptions (class in twindb_backup.destination.s3), 14

B

backup (twindb_backup.exporter.base_exporter.ExportMeasureType attribute), 18
 backup_everything() (in module twindb_backup.backup), 24
 backup_files() (in module twindb_backup.backup), 25
 backup_mysql() (in module twindb_backup.backup), 25
 BaseDestination (class in twindb_backup.destination), 12
 BaseExporter (class in twindb_backup.exporter), 18
 BaseExporterError, 18
 basename() (twindb_backup.destination.base_destination.BaseDestination method), 12
 BaseSource (class in twindb_backup.source.base_source), 20
 binlog_coordinate (twindb_backup.source.mysql_source.MySQLSource attribute), 22

C

Cache (class in twindb_backup.cache.cache), 11
 CacheException, 12
 callback() (twindb_backup.modifiers.base.Modifier method), 19
 callback() (twindb_backup.modifiers.keeplocal.KeepLocal method), 20
 client (twindb_backup.destination.ssh.Ssh attribute), 16
 clone_mysql() (in module twindb_backup.clone), 25

create_bucket() (twindb_backup.destination.s3.S3 method), 14
D
 DataDogExporter (class in twindb_backup.exporter.datadog_exporter), 18
 DataDogExporterError, 18
 delete() (twindb_backup.destination.base_destination.BaseDestination method), 12
 delete() (twindb_backup.destination.local.Local method), 13
 delete() (twindb_backup.destination.s3.S3 method), 14
 delete() (twindb_backup.destination.ssh.Ssh method), 16
 delete_all_objects() (twindb_backup.destination.s3.S3 method), 14
 delete_bucket() (twindb_backup.destination.s3.S3 method), 14
 delete_local_files() (in module twindb_backup), 30
 disable_wsrep_desync() (twindb_backup.source.mysql_source.MySQLSource method), 22

E

edit_backup_my_cnf() (in module twindb_backup.verify), 29
 empty_dir() (in module twindb_backup.util), 28
 ensure_wsrep_desync() (twindb_backup.source.mysql_source.MySQLSource method), 22
 ensure_empty() (in module twindb_backup.util), 29
 ensure_tcp_port_listening() (twindb_backup.destination.ssh.Ssh method), 16
 execute() (twindb_backup.ssh.client.SshClient method), 24
 execute_command() (twindb_backup.destination.ssh.Ssh method), 16
 export() (twindb_backup.exporter.base_exporter.BaseExporter method), 18

export() (twindb_backup.exporter.datadog_exporter.DataDogExporter method), 18
 export_info() (in module twindb_backup.export), 26
 ExportCategory (class in twindb_backup.exporter.base_exporter), 18
 ExportMeasureType (class in twindb_backup.exporter.base_exporter), 18
F
 files (twindb_backup.exporter.base_exporter.ExportCategory attribute), 18
 FileSource (class in twindb_backup.source.file_source), 21
 filter() (twindb_backup.LessThanFilter method), 30
 find_files() (twindb_backup.destination.base_destination.BaseDestination method), 12
 find_files() (twindb_backup.destination.local.Local method), 13
 find_files() (twindb_backup.destination.s3.S3 method), 14
 find_files() (twindb_backup.destination.ssh.Ssh method), 17
 full (twindb_backup.source.mysql_source.MySQLSource attribute), 22
G
 galera (twindb_backup.source.mysql_source.MySQLSource attribute), 22
 gen_grastate() (in module twindb_backup.restore), 26
 get_backup_type() (in module twindb_backup.restore), 26
 get_binlog_coordinates() (twindb_backup.source.mysql_source.MySQLSource static method), 22
 get_connection() (twindb_backup.source.mysql_source.MySQLSource method), 22
 get_destination() (in module twindb_backup.configuration), 25
 get_directories_to_backup() (in module twindb_backup), 30
 get_export_transport() (in module twindb_backup.configuration), 26
 get_files_to_delete() (in module twindb_backup), 30
 get_full_copy_name() (twindb_backup.destination.base_destination.BaseDestination method), 12
 get_galera_version() (in module twindb_backup.restore), 27
 get_hostname_from_backup_copy() (in module twindb_backup.util), 29
 get_latest_backup() (twindb_backup.destination.base_destination.BaseDestination method), 13
 get_my_cnf() (in module twindb_backup.restore), 27
 get_name() (twindb_backup.source.mysql_source.MySQLSource static method), 22
 get_name() (twindb_backup.source.file_source.FileSource method), 21
 get_name() (twindb_backup.source.mysql_source.MySQLSource method), 22
 get_prefix() (twindb_backup.source.base_source.BaseSource method), 20
 get_remote_handlers() (twindb_backup.ssh.client.SshClient method), 24
 get_stream() (twindb_backup.destination.local.Local static method), 13
 get_stream() (twindb_backup.destination.s3.S3 method), 15
 get_stream() (twindb_backup.destination.ssh.Ssh method), 17
 get_stream() (twindb_backup.modifiers.base.Modifier method), 19
 get_stream() (twindb_backup.modifiers.gpg.Gpg method), 19
 get_stream() (twindb_backup.modifiers.zip.Gzip method), 20
 get_stream() (twindb_backup.modifiers.keeplocal.KeepLocal method), 20
 get_stream() (twindb_backup.source.base_source.BaseSource method), 20
 get_stream() (twindb_backup.source.file_source.FileSource method), 21
 get_stream() (twindb_backup.source.mysql_source.MySQLSource method), 22
 get_timeout() (in module twindb_backup), 31
 get_transfer_config() (twindb_backup.destination.s3.S3 static method), 15
 Gpg (class in twindb_backup.modifiers.gpg), 19
 Gzip (class in twindb_backup.modifiers.zip), 20
H
 host (twindb_backup.destination.ssh.Ssh attribute), 17
I
 incremental (twindb_backup.source.mysql_source.MySQLSource attribute), 23
 is_galera() (twindb_backup.source.mysql_source.MySQLSource method), 23
K
 KeepLocal (class in twindb_backup.modifiers.keeplocal), 20
 kill_children() (in module twindb_backup.util), 29
L
 list_available_backups() (in module twindb_backup.ls), 26

- list_files() (twindb_backup.destination.base_destination.BaseDestination method), 13
 - list_files() (twindb_backup.destination.local.Local method), 13
 - list_files() (twindb_backup.destination.s3.S3 method), 15
 - list_files() (twindb_backup.destination.ssh.Ssh method), 17
 - Local (class in twindb_backup.destination.local), 13
 - lsn (twindb_backup.source.mysql_source.MySQLSource attribute), 23
- ## M
- media_type (twindb_backup.source.file_source.FileSource attribute), 21
 - mkdir_p() (in module twindb_backup.util), 29
 - Modifier (class in twindb_backup.modifiers.base), 19
 - ModifierException, 19
 - mysql (twindb_backup.exporter.base_exporter.ExportCategory attribute), 18
 - MySQLConnectInfo (class in twindb_backup.source.mysql_source), 21
 - MySQLSource (class in twindb_backup.source.mysql_source), 21
- ## N
- netcat() (twindb_backup.destination.ssh.Ssh method), 17
- ## P
- parent (twindb_backup.source.mysql_source.MySQLSource attribute), 23
 - parent_lsn (twindb_backup.source.mysql_source.MySQLSource attribute), 23
 - prepare_status() (in module twindb_backup.backup), 25
 - private (twindb_backup.destination.s3.S3FileAccess attribute), 16
 - public_read (twindb_backup.destination.s3.S3FileAccess attribute), 16
 - purge() (twindb_backup.cache.cache.Cache method), 12
- ## R
- restore (twindb_backup.exporter.base_exporter.ExportMeasure attribute), 18
 - restore_from_file() (in module twindb_backup.restore), 27
 - restore_from_mysql() (in module twindb_backup.restore), 27
 - restore_from_mysql_full() (in module twindb_backup.restore), 27
 - restore_from_mysql_incremental() (in module twindb_backup.restore), 28
 - restore_in() (twindb_backup.cache.cache.Cache method), 12
 - revert_stream() (twindb_backup.modifiers.gpg.Gpg method), 19
- revert_stream() (twindb_backup.modifiers.gzip.Gzip method), 20
 - run_backup_job() (in module twindb_backup.backup), 25
 - run_command() (in module twindb_backup.util), 29
 - run_type (twindb_backup.source.base_source.BaseSource attribute), 21
- ## S
- S3 (class in twindb_backup.destination.s3), 14
 - S3FileAccess (class in twindb_backup.destination.s3), 16
 - save() (twindb_backup.destination.base_destination.BaseDestination method), 13
 - save() (twindb_backup.destination.local.Local method), 14
 - save() (twindb_backup.destination.s3.S3 method), 15
 - save() (twindb_backup.destination.ssh.Ssh method), 17
 - save_measures() (in module twindb_backup), 31
 - session() (twindb_backup.ssh.client.SshClient method), 24
 - set_open_files_limit() (in module twindb_backup.backup), 25
 - setup_logging() (in module twindb_backup), 31
 - setup_s3_client() (twindb_backup.destination.s3.S3 method), 15
 - share() (in module twindb_backup.share), 28
 - share() (twindb_backup.destination.s3.S3 method), 15
 - split_host_port() (in module twindb_backup.util), 29
 - Ssh (class in twindb_backup.destination.ssh), 16
 - SshClient (class in twindb_backup.ssh.client), 23
 - SshClientException, 24
 - SshConnectInfo (class in twindb_backup.destination.ssh), 17
 - status (twindb_backup.source.mysql_source.MySQLSource attribute), 23
 - status() (twindb_backup.destination.base_destination.BaseDestination method), 13
 - suffix (twindb_backup.source.base_source.BaseSource attribute), 21
- ## T
- time_type() (in module twindb_backup.backup), 25
 - twindb_backup (module), 29
 - twindb_backup.backup (module), 24
 - twindb_backup.cache (module), 12
 - twindb_backup.cache.cache (module), 11
 - twindb_backup.cli (module), 25
 - twindb_backup.clone (module), 25
 - twindb_backup.configuration (module), 25
 - twindb_backup.destination (module), 18
 - twindb_backup.destination.base_destination (module), 12
 - twindb_backup.destination.local (module), 13
 - twindb_backup.destination.s3 (module), 14
 - twindb_backup.destination.ssh (module), 16

- twindb_backup.export (module), 26
- twindb_backup.exporter (module), 19
- twindb_backup.exporter.base_exporter (module), 18
- twindb_backup.exporter.datadog_exporter (module), 18
- twindb_backup.exporter.exceptions (module), 18
- twindb_backup.ls (module), 26
- twindb_backup.modifiers (module), 20
- twindb_backup.modifiers.base (module), 19
- twindb_backup.modifiers.gpg (module), 19
- twindb_backup.modifiers.gzip (module), 20
- twindb_backup.modifiers.keeplocal (module), 20
- twindb_backup.restore (module), 26
- twindb_backup.share (module), 28
- twindb_backup.source (module), 23
- twindb_backup.source.base_source (module), 20
- twindb_backup.source.file_source (module), 21
- twindb_backup.source.mysql_source (module), 21
- twindb_backup.ssh (module), 24
- twindb_backup.ssh.client (module), 23
- twindb_backup.ssh.exceptions (module), 24
- twindb_backup.util (module), 28
- twindb_backup.verify (module), 29
- TwinDBBackupError, 30
- type (twindb_backup.source.mysql_source.MySQLSource attribute), 23

U

- update_grastate() (in module twindb_backup.restore), 28

V

- validate_client_response()
(twindb_backup.destination.s3.S3 static method), 15
- verify_mysql_backup() (in module twindb_backup.verify), 29

W

- wsrep_provider_version (twindb_backup.source.mysql_source.MySQLSource attribute), 23