Table of Contents

Name

duplicity - Encrypted backup using rsync algorithm

Synopsis

duplicity [*options***]** *source_directory* target_url

duplicity [*options***]** *source_url* target_directory

duplicity full [*options***]** *source_directory* target_url

duplicity incremental [*options***]** *source_directory* target_url

duplicity restore [*options***]** *source_url*

target_directory

duplicity verify [*options***]** *source_url* target_directory

duplicity collection-status [*options***]** *target_url*

duplicity list-current-files [*options***]** *target_url*

duplicity cleanup [*options***] [***--force***]** *target_url*

duplicity remove-older-than *time* **[***options***] [***--force***]** *target_url*

duplicity remove-all-but-n-full *count* **[***options***] [***--force***]** *target_url*

duplicity remove-all-inc-of-but-n-full *count* **[***options***] [***--force***]** *target_url*

Description

Duplicity incrementally backs up files and directory by encrypting tar-format volumes with GnuPG and uploading them to a remote (or local) file server. Currently local, ftp, ssh/scp, rsync, WebDAV, WebDAVs, HSi and Amazon S3 backends are available. Because duplicity uses librsync, the incremental archives are space efficient and only record the parts of files that have changed since the last backup. Currently duplicity supports deleted files, full Unix permissions, directories, symbolic links, fifos, etc., but not hard links.

Duplicity will read the PASSPHRASE environment variable to find the passphrase to give to GnuPG. If this is not set, the user will be prompted for the passphrase.

If you are backing up the root directory /, remember to --exclude /proc, or else duplicity will probably crash on the weird stuff in there.

Examples

Here is an example of a backup, using scp to back up /home/me to some_dir on the other.host machine:

```
duplicity /home/me scp://uid@other.host/some_dir
```

If the above is run repeatedly, the first will be a full backup, and subsequent ones will be incremental. To force a full backup, use the *full* action:

```
duplicity full /home/me scp://uid@other.host/some_dir
```

Now suppose we accidentally delete /home/me and want to restore it the way it was at the time of last backup:

```
duplicity scp://uid@other.host/some_dir /home/me
```

Duplicity enters restore mode because the URL comes before the local directory. If we wanted to restore just the file "Mail/article" in /home/me as it was three days ago into /home/me/restored_file:

```
duplicity -t 3D --file-to-restore Mail/article scp://uid@other.host
/some_dir /home/me/restored_file
```

The following command compares the files we backed up, so see what has changed since then:

```
duplicity verify scp://uid@other.host/some_dir /home/me
```

Finally, duplicity recognizes several include/exclude options. For instance, the following will backup the root directory, but exclude /mnt, /tmp, and /proc:

```
duplicity --exclude /mnt --exclude /tmp --exclude /proc / file:///usr
/local/backup
```

Note that in this case the destination is the local directory /usr/local/backup. The following will backup only the /home and /etc directories under root:

```
duplicity --include /home --include /etc --exclude ’**’ / file:///usr/local
```

```
/backup
```

Duplicity can also access a repository via ftp. If a user name is given, the environment variable FTP_PASSWORD is read to determine the password:

```
FTP_PASSWORD=mypassword

duplicity /local/dir ftp://user@other.host/some_dir
```

Actions

cleanup
: Delete the extraneous duplicity files on the given backend. Non-duplicity files, or files in complete data sets will not be deleted. This should only be necessary after a duplicity session fails or is aborted prematurely. Note that *--force* will be needed to delete the files rather than just list them.

collection-status
: Summarize the status of the backup repository by printing the chains and sets found, and the number of volumes in each.

full
: Indicate full backup. If this is set, perform full backup even if signatures are available.

incr
: If this is requested an incremental backup will be performed. Duplicity will abort if old signatures cannot be found. The default is to switch to full backup under these conditions.

list-current-files
: Lists the files currently backed up in the archive. The information will be extracted from the signature files, not the archive data itself. Thus the whole archive does not have to be downloaded, but on the other hand if the archive has been deleted or corrupted, this command may not detect it.

remove-older-than *time*
: Delete all backup sets older than the given time. Old backup sets will not be deleted if backup sets newer than *time* depend on them. See the **TIME FORMATS** section for more information. Note, this action cannot be combined with backup or other actions, such as cleanup. Note also that *--force* will be needed to delete the files rather than just list them.

remove-all-but-n-full *count*

Delete all backups sets that are older than the count:th last full backup (in other words, keep the last *count* full backups and associated incremental sets). *count* must be larger than zero. A value of 1 means that only the single most recent backup chain will be kept. Note that *--force* will be needed to delete the files rather than just list them.

remove-all-inc-of-but-n-full *count*
Delete incremental sets of all backups sets that are older than the count:th last full backup (in other words, keep only old full backups and not their increments). *count* must be larger than zero. A value of 1 means that only the single most recent backup chain will be kept intact. Note that *--force* will be needed to delete the files rather than just list them.

verify
Enter verify mode instead of restore. If the --file-to-restore option is given, restrict verify to that file or directory. duplicity will exit with a non-zero error level if any files are different. On verbosity level 4 or higher, log a message for each file that has changed.

Options

--allow-source-mismatch
Do not abort on attempts to use the same archive dir or remote backend to back up different directories. duplicity will tell you if you need this switch.

--archive-dir *path*
The archive directory. **NOTE:** This option changed in 0.6.0. The archive directory is now necessary in order to manage persistence for current and future enhancements. As such, this option is now used only to change the location of the archive directory. The archive directory should **not** be deleted, or duplicity will have to recreate it from the remote repository (which may require decrypting the backup contents).

When backing up or restoring, this option specifies that the local archive directory is to be created in *path*. If the archive directory is not specified, the default will be to create the archive directory in *~/.cache/duplicity/*.

The archive directory can be shared between backups to multiple targets, because a subdirectory of the archive dir is used for individual backups (see **--name**).

The combination of archive directory and backup name must be unique in order to separate the data of different backups.

The interaction between the **--archive-dir** and the **--name** options allows for four possible combinations for the location of the archive dir:

1.
neither specified (default) ~/.cache/duplicity/
hash-of-url
2.
--archive-dir=/arch, no --name /arch/
hash-of-url
3.
no --archive-dir, --name=foo ~/.cache/duplicity/foo
4.
--archive-dir=/arch, --name=foo /arch/foo

--asynchronous-upload
(EXPERIMENTAL) Perform file uploads asynchronously in the background, with respect to volume creation. This means that duplicity can upload a volume while, at the same time, preparing the next volume for upload. The intended end-result is a faster backup, because the local CPU and your bandwidth can be more consistently utilized. Use of this option implies additional need for disk space in the temporary storage location; rather than needing to store only one volume at a time, enough storage space is required to store two volumes.

--dry-run
Calculate what would be done, but do not perform any backend actions

--encrypt-key *key*
When backing up, encrypt to the given public key, instead of using symmetric (traditional) encryption. Can be specified multiple times.

--exclude *shell_pattern*
Exclude the file or files matched by *shell_pattern*. If a directory is matched, then files under that directory will also be matched. See the **FILE SELECTION** section for more information.

--exclude-device-files
Exclude all device files. This can be useful for security/permissions reasons or if rdiff-backup is not handling device files correctly.

--exclude-filelist *filename*
Excludes the files listed in *filename*. See the **FILE SELECTION** section for more information.

--exclude-filelist-stdin

Like **--exclude-filelist,** but the list of files will be read from standard input. See the **FILE SELECTION** section for more information.

--exclude-globbing-filelist filename
Like **--exclude-filelist** but each line of the filelist will be interpreted according to the same rules as **--include** and **--exclude.**

--exclude-if-present filename
Exclude directories if filename is present. This option needs to come before any other include or exclude options.

--exclude-other-filesystems
Exclude files on file systems (identified by device number) other than the file system the root of the source directory is on.

--exclude-regexp *regexp*
Exclude files matching the given regexp. Unlike the **--exclude** option, this option does not match files in a directory it matches. See the **FILE SELECTION** section for more information.

--extra-clean
When cleaning up, be more aggressive about saving space. For example, this may delete signature files for old backup chains. See the **cleanup** argument for more information.

--file-to-restore *path*
This option may be given in restore mode, causing only *path* to be restored instead of the entire contents of the backup archive. *path* should be given relative to the root of the directory backed up.

--full-if-older-than *time*
Perform a full backup if an incremental backup is requested, but the latest full backup in the collection is older than the given *time*. See the **TIME FORMATS** section for more information.

--force
Proceed even if data loss might result. Duplicity will let the user know when this option is required.

--ftp-passive
Use passive (PASV) data connections. The default is to use passive, but to fallback to regular if the passive connection fails or times out.

--ftp-regular
Use regular (PORT) data connections.

--gio
: Use the GIO backend and interpret any URLs as GIO would.

--ignore-errors
: Try to ignore certain errors if they happen. This option is only intended to allow the restoration of a backup in the face of certain problems that would otherwise cause the backup to fail. It is not ever recommended to use this option unless you have a situation where you are trying to restore from backup and it is failing because of an issue which you want duplicity to ignore. Even then, depending on the issue, this option may not have an effect.

  Please note that while ignored errors will be logged, there will be no summary at the end of the operation to tell you what was ignored, if anything. If this is used for emergency restoration of data, it is recommended that you run the backup in such a way that you can revisit the backup log (look for lines containing the string IGNORED_ERROR).

  If you ever have to use this option for reasons that are not understood or understood but not your own responsibility, please contact duplicity maintainers. The need to use this option under production circumstances would normally be considered a bug.

--imap-mailbox *option*
: Allows you to specify a different mailbox. The default is "INBOX". Other languages may require a different mailbox than the default.

--gpg-options *options*
: Allows you to pass options to gpg encryption. The *options* list should be of the form "opt1=parm1 opt2=parm2" where the string is quoted and the only spaces allowed are between options.

--include *shell_pattern*
: Similar to **--exclude** but include matched files instead. Unlike **--exclude**, this option will also match parent directories of matched files (although not necessarily their contents). See the **FILE SELECTION** section for more information.

--include-filelist *filename*
: Like **--exclude-filelist**, but include the listed files instead. See the **FILE SELECTION** section for more information.

--include-filelist-stdin
: Like **--include-filelist**, but read the list of included files from standard input.

--include-globbing-filelist *filename*
Like **--include-filelist** but each line of the filelist will be interpreted according to the same rules as **--include** and **--exclude.**

--include-regexp *regexp*
Include files matching the regular expression *regexp*. Only files explicitly matched by *regexp* will be included by this option. See the **FILE SELECTION** section for more information.

--log-fd *number*
Write specially-formatted versions of output messages to the specified file descriptor. The format used is designed to be easily consumable by other programs.

--log-file *filename*
Write specially-formatted versions of output messages to the specified file. The format used is designed to be easily consumable by other programs.

--name *symbolicname*
Set the symbolic name of the backup being operated on. The intent is to use a separate name for each logically distinct backup. For example, someone may use "home_daily_s3" for the daily backup of a home directory to Amazon S3. The structure of the name is up to the user, it is only important that the names be distinct. The symbolic name is currently only used to affect the expansion of **--archive-dir** , but may be used for additional features in the future. Users running more than one distinct backup are encouraged to use this option.

If not specified, the default value is a hash of the backend URL.

--no-encryption
Do not use GnuPG to encrypt files on remote system. Instead just write gzipped volumes.

--no-print-statistics
By default duplicity will print statistics about the current session after a successful backup. This switch disables that behavior.

--null-separator
Use nulls (\0) instead of newlines (\n) as line separators, which may help when dealing with filenames containing newlines. This affects the expected format of the files specified by the --{include|exclude}-filelist[-stdin] switches as well as the format of the directory statistics file.

--num-retries *number*
Number of retries to make on errors before giving up.

--old-filenames
Use the old filename format (incompatible with Windows/Samba) rather than the new filename format.

--rename *orig new*
Treats the path *orig* in the backup as if it were the path *new.* Can be passed multiple times. An example:

```
duplicity restore --rename Documents/metal Music/metal
scp://uid@other.host/some_dir /home/me
```

--s3-european-buckets
When using the Amazon S3 backend, create buckets in Europe instead of the default (requires **--s3-use-new-style**). Also see the **EUROPEAN S3 BUCKETS** section.

--s3-unencrypted-connection
Don’t use SSL for connections to S3.

This may be much faster, at some cost to confidentiality.

With this option, anyone who can observe traffic between your computer and S3 will be able to tell: that you are using Duplicity, the name of the bucket, your AWS Access Key ID, the increment dates and the amount of data in each increment.

This option affects only the connection, not the GPG encryption of the backup increment files. Unless that is disabled, an observer will not be able to see the file names or contents.

--s3-use-new-style
When operating on Amazon S3 buckets, use new-style subdomain bucket addressing. This is now the preferred method to access Amazon S3, but is not backwards compatible if your bucket name contains upper-case characters or other characters that are not valid in a hostname.

--scp-command *command*
This option only matters when using the ssh/scp backend. The *command* will be used instead of scp to send or receive files. The default command is "scp". To list and delete existing files, the sftp command is used. See **--ssh-options** and **--sftp-command**.

--sftp-command *command*
This option only matters when using the ssh/scp backend. The *command*

will be used instead of sftp for listing and deleting files. The default is "sftp". File transfers are done using the sftp command. See **--ssh-options**, **--use-scp**, and **--scp-command**.

--sign-key *key*
: This option can be used when backing up or restoring. When backing up, all backup files will be signed with keyid *key*. When restoring, duplicity will signal an error if any remote file is not signed with the given keyid. *key* should be an 8 character hex string, like AA0E73D2.

--ssh-askpass
: Tells the ssh/scp backend to use FTP_PASSWORD from the environment, or, if that is not present, to prompt the user for the remote system password.

--ssh-options *options*
: Allows you to pass options to the ssh/scp/sftp backend. The *options* list should be of the form "opt1=parm1 opt2=parm2" where the option string is quoted and the only spaces allowed are between options. The option string will be passed verbatim to both scp and sftp, whose command line syntax differs slightly: options passed with **--ssh-options** should therefore be given in the long option format described in **ssh_config(5)** , like in this example:

```
duplicity --ssh-options="-oProtocol=2 -oIdentityFile=/my/backup/id"
/home/me scp://uid@other.host/some_dir
```

--short-filenames
: If this option is specified, the names of the files duplicity writes will be shorter (about 30 chars) but less understandable. This may be useful when backing up to MacOS or another OS or FS that doesn’t support long filenames.

--tempdir *directory*
: Use this existing directory for duplicity temporary files instead of the system default, which is usually the /tmp directory. This option supersedes any environment variable.

-t*time***, --time** *time***, --restore-time** *time*
: Specify the time from which to restore or list files.

--time-separator *char*
: Use *char* as the time separator in filenames instead of colon (":").

--use-agent

If this option is specified, then *--use-agent* is passed to the GnuPG encryption process and it will turn off any passphrase interaction with the user with respect to *--encrypt-key* or *--sign-key.*

--use-scp
: If this option is specified, then the ssh backend will use *scp* rather than *sftp* for the get and put backend operations. The default is to use *sftp* for all operations. With this option, duplicity will use *sftp* for list and delete operations,

and *scp* for put and get operations

-v*verb***, --verbosity** *verb*
: Specify verbosity level (0 is total silent, 4 is the default, and 9 is noisiest). Verbosity may also be one of: character *ewnid,* or word *error, warning, notice, info, debug.* The default is 4 (Notice). The options *-v4, -vn,* and *-vnotice* are functionally equivalent, as are the mixed/upper-case versions, *-vN, -vNotice,* and *-vNOTICE.*

--version
: Print duplicity's version and quit.

--volsize *number*
: Change the volume size to *number* Mb. Default is 25Mb.

Environment Variables

TMPDIR, TEMP, TMP
: In decreasing order of importance, specifies the directory to use for temporary files (inherited from Python's tempfile module).

FTP_PASSWORD
: Supported by most backends which are password capable. More secure than setting it in the backend url.

URL Format

Duplicity tries to maintain a standard URL format as much as possible. The generic format for a URL is:

scheme://user[:password]@host[:port]/[/]path

It is not recommended to expose the password on the command line since it could be revealed to anyone with permissions to do process listings, however, it is permitted. Consider setting the environment variable FTP_PASSWORD instead, which is supported by most, but not all backends. Regardless of its name it can be used with other backends.

In protocols that support it, the path may be preceded by a single slash, ’/path’, to represent a relative path to the target home directory, or preceded by a double slash, ’//path’, to represent an absolute filesystem path.

Formats of each of the URL schemes follow:

cf+http://container_name

file:///some_dir

ftp://user[:password]@other.host[:port]/some_dir

hsi://user[:password]@other.host/some_dir

imap://user[:password]@host.com[/from_address_prefix]

imaps://user[:password]@host.com[/from_address_prefix]

using rsync daemon
rsync://user[:password]@host.com[:port]::/module/some_dir

using rsync over ssh (only key auth)
rsync://user@host.com[:port]/relative_path
rsync://user@host.com[:port]//absolute_path

s3://host/bucket_name[/prefix]

s3+http://bucket_name[/prefix]

scp://user[:password]@other.host[:port]/some_dir

ssh://user[:password]@other.host[:port]/some_dir

tahoe://alias/directory

webdav://user[:password]@other.host/some_dir

webdavs://user[:password]@other.host/some_dir

Time Formats

duplicity uses time strings in two places. Firstly, many of the files duplicity creates will have the time in their filenames in the w3 datetime format as described in a w3 note at http://www.w3.org/TR/NOTE-datetime. Basically they look like "2001-07-15T04:09:38-07:00", which means what it looks like. The "-07:00" section means the time zone is 7 hours behind UTC.

Secondly, the **-t**, **--time**, and **--restore-time** options take a time string, which can be given in any of several formats:

1. the string "now" (refers to the current time)
2. a sequences of digits, like "123456890" (indicating the time in seconds after the epoch)
3. A string like "2002-01-25T07:00:00+02:00" in datetime format
4. An interval, which is a number followed by one of the characters s, m, h, D, W, M, or Y (indicating seconds, minutes, hours, days, weeks, months, or years respectively), or a series of such pairs. In this case the string refers to the time that preceded the current time by the length of the interval. For instance, "1h78m" indicates the time that was one hour and 78 minutes ago. The calendar here is unsophisticated: a month is always 30 days, a year is always 365 days, and a day is always 86400 seconds.
5. A date format of the form YYYY/MM/DD, YYYY-MM-DD, MM/DD/YYYY, or MM-DD-YYYY, which indicates midnight on the day in question, relative to the current time zone settings. For instance, "2002/3/5", "03-05-2002", and "2002-3-05" all mean March 5th, 2002.

File Selection

duplicity accepts the same file selection options **rdiff-backup** does, including --exclude, --exclude-filelist-stdin, etc.

When duplicity is run, it searches through the given source directory and backs up all the files specified by the file selection system. The file selection system comprises a number of file selection conditions, which are set using one of the following command line options: **--exclude**, **--exclude-device-files**, **--exclude-filelist**, **--exclude-filelist-stdin**, **--exclude-globbing-filelist**, **--exclude-regexp**, **--include**, **--include-filelist**, **--include-filelist-stdin**, **--include-globbing-filelist**, and **--include-regexp**. Each file selection condition either matches or doesn’t match a given file. A given file is excluded by the file selection system exactly when the first matching file selection condition specifies that the file be excluded; otherwise the file is included.

For instance,

```
duplicity --include /usr --exclude /usr /usr scp://user@host/backup
```

is exactly the same as

```
duplicity /usr scp://user@host/backup
```

because the include and exclude directives match exactly the same files, and the **--include** comes first, giving it precedence. Similarly,

```
duplicity --include /usr/local/bin --exclude /usr/local /usr
scp://user@host/backup
```

would backup the /usr/local/bin directory (and its contents), but not /usr/local/doc.

The **include**, **exclude**, **include-globbing-filelist**, and **exclude-globbing-filelist** options accept *extended shell globbing patterns*. These patterns can contain the special patterns *, **, **?**, and **[...]**. As in a normal shell, * can be expanded to any string of characters not containing "/", **?** expands to any character except "/", and **[...]** expands to a single character of those characters specified (ranges are acceptable). The new special pattern, **, expands to any string of characters whether or not it contains "/". Furthermore, if the pattern starts with "ignorecase:" (case insensitive), then this prefix will be removed and any character in the string can be replaced with an upper- or lowercase version of itself.

Remember that you may need to quote these characters when typing them into a shell, so the shell does not interpret the globbing patterns before duplicity sees them.

The **--exclude** *pattern* option matches a file iff:

1. *pattern* can be expanded into the file’s filename, or
2. the file is inside a directory matched by the option.

Conversely, **--include** *pattern* matches a file iff:

1. *pattern* can be expanded into the file’s filename,
2. the file is inside a directory matched by the option, or

•

3. the file is a directory which contains a file matched by the option.

For example,

> **--exclude** /usr/local

matches /usr/local, /usr/local/lib, and /usr/local/lib/netscape. It is the same as --exclude /usr/local --exclude '/usr/local/**'.

> **--include** /usr/local

specifies that /usr, /usr/local, /usr/local/lib, and /usr/local/lib/netscape (but not /usr/doc) all be backed up. Thus you don't have to worry about including parent directories to make sure that included subdirectories have somewhere to go. Finally,

> **--include** ignorecase:'/usr/[a-z0-9]foo/*/**.py'

would match a file like /usR/5fOO/hello/there/world.py. If it did match anything, it would also match /usr. If there is no existing file that the given pattern can be expanded into, the option will not match /usr.

The **--include-filelist**, **--exclude-filelist**, **--include-filelist-stdin**, and **--exclude-filelist-stdin** options also introduce file selection conditions. They direct duplicity to read in a file, each line of which is a file specification, and to include or exclude the matching files. Lines are separated by newlines or nulls, depending on whether the --null-separator switch was given. Each line in a filelist is interpreted similarly to the way *extended* shell patterns are, with a few exceptions:

•

1. Globbing patterns like *, **, ?, and **[...]** are not expanded.

•

2. Include patterns do not match files in a directory that is included. So /usr/local in an include file will not match /usr/local/doc.

•

3. Lines starting with "+ " are interpreted as include directives, even if found in a filelist referenced by **--exclude-filelist**. Similarly, lines starting with "- " exclude files even if they are found within an include filelist.

For example, if file "list.txt" contains the lines:

```
/usr/local

- /usr/local/doc

/usr/local/bin

+ /var

- /var
```

then "--include-filelist list.txt" would include /usr, /usr/local, and /usr/local/bin. It would exclude /usr/local/doc, /usr/local/doc/python, etc. It neither excludes nor includes /usr/local/man, leaving the fate of this directory to the next specification condition. Finally, it is undefined what happens with /var. A single file list should not contain conflicting file specifications.

The **--include-globbing-filelist** and **--exclude-globbing-filelist** options also specify filelists, but each line in the filelist will be interpreted as a globbing pattern the way **--include** and **--exclude** options are interpreted (although "+ " and "- " prefixing is still allowed). For instance, if the file "globbing-list.txt" contains the lines:

```
dir/foo

+ dir/bar

- **
```

Then "--include-globbing-filelist globbing-list.txt" would be exactly the same as specifying "--include dir/foo --include dir/bar --exclude **" on the command line.

Finally, the **--include-regexp** and **--exclude-regexp** allow files to be included and excluded if their filenames match a python regular expression. Regular expression syntax is too complicated to explain here, but is covered in Python’s library reference. Unlike the **--include** and **--exclude** options, the regular expression options don’t match files containing or contained in matched files. So for instance

```
--include ’[0-9]{7}(?!foo)’
```

matches any files whose full pathnames contain 7 consecutive digits which aren’t followed by ’foo’. However, it wouldn’t match /home even if /home/ben /1234567 existed.

Operation and Data Formats

This section describes duplicity's basic operation and the format of its data files. It should not necessary to read this section to use duplicity.

The files used by duplicity to store backup data are tarfiles in GNU tar format. They can be produced independently by **rdiffdir**(1) . For incremental backups, new files are saved normally in the tarfile. But when a file changes, instead of storing a complete copy of the file, only a diff is stored, as generated by **rdiff**(1) . If a file is deleted, a 0 length file is stored in the tar. It is possible to restore a duplicity archive "manually" by using **tar** and then **cp**, **rdiff**, and **rm** as necessary. These duplicity archives have the extension **difftar**.

Both full and incremental backup sets have the same format. In effect, a full backup set is an incremental one generated from an empty signature (see below). The files in full backup sets will start with **duplicity-full** while the incremental sets start with **duplicity-inc**. When restoring, duplicity applies patches in order, so deleting, for instance, a full backup set may make related incremental backup sets unusable.

In order to determine which files have been deleted, and to calculate diffs for changed files, duplicity needs to process information about previous sessions. It stores this information in the form of tarfiles where each entry's data contains the signature (as produced by **rdiff**) of the file instead of the file's contents. These signature sets have the extension **sigtar**.

Signature files are not required to restore a backup set, but without an up-to-date signature, duplicity cannot append an incremental backup to an existing archive.

To save bandwidth, duplicity generates full signature sets and incremental signature sets. A full signature set is generated for each full backup, and an incremental one for each incremental backup. These start with **duplicity-full-signatures** and **duplicity-new-signatures** respectively. These signatures will be stored both locally and remotely. The remote signatures will be encrypted if encryption is enabled. The local signatures will not be encrypted and stored in the archive dir (see **--archive-dir**).

European S3 Buckets

Amazon S3 provides the ability to choose the location of a bucket upon its creation. The purpose is to enable the user to choose a location which is better located network topologically relative to the user, because it may allow for faster data transfers.

duplicity will create a new bucket the first time a bucket access is attempted. At this point, the bucket will be created in Europe if **--s3-european-buckets** was given. For reasons having to do with how the Amazon S3 service works, this also requires the use of the **--s3-use-new-style** option. This option turns on subdomain based bucket addressing in S3. The details are beyond the scope of this man page, but it is important to know that your bucket must not contain upper case letters or any other characters that are not valid parts of a hostname. Consequently, for reasons of backwards compatibility, use of subdomain based bucket addressing is not enabled by default.

Note that you will need to use **--s3-use-new-style** for all operations on European buckets; not just upon initial creation.

You only need to use **--s3-european-buckets** upon initial creation, but you may may use it at all times for consistency.

Further note that when creating a new European bucket, it can take a while before the bucket is fully accessible. At the time of this writing it is unclear to what extent this is an expected feature of Amazon S3, but in practice you may experience timeouts, socket errors or HTTP errors when trying to upload files to your newly created bucket. Give it a few minutes and the bucket should function normally.

Imap

An IMAP account can be used as a target for the upload. The userid may be specified and the password will be requested.

The **from_address_prefix** may be specified (and probably should be). The text will be used as the "From" address in the IMAP server. Then on a restore (or list) command the **from_address_prefix** will distinguish between different backups.

a Note on Ssh/Scp Protocols

Duplicity specifies two protocol names for the same protocol. This is a known and user-confusing issue. Both use the same protocol suite, namely *ssh* through its’ utility routines *scp* and *sftp*. Older versions of duplicity used *scp* for get and put operations and *sftp* for list and delete operations. The current version uses *sftp* for all four supported operations, unless the *--use-scp* option is used to revert to old behavior. The change was made to all-sftp in order to allow the remote system to chroot the backup, thus providing better security.

Bugs

Hard links currently unsupported (they will be treated as non-linked regular files).

Bad signatures will be treated as empty instead of logging appropriate error message.

Author

Original Author - Ben Escoto <bescoto@stanford.edu>

Current Maintainer - Kenneth Loafman <kenneth@loafman.com>

See Also

rdiffdir(1) , **python**(1) , **rdiff**(1) , **rdiff-backup**(1) .

Table of Contents