Duplicity Encrypted incremental backups

CONCEPTS

Argument order decides the action: local dir then URL backs up, URL then local dir restores. Delete actions only list until you add --force.

Full whole tree, GPG-encrypted
↓ incremental
Incremental changes since last set
↓ more runs
Chain full + its increments
↓ remove-older-than
Pruned old chains deleted

An incremental needs every earlier set of its chain back to the full. Run a fresh full regularly (--full-if-older-than) so restores stay short and safe.

Encryption defaults to a symmetric GnuPG passphrase read from PASSPHRASE. Lose the passphrase or private key and the backup is unreadable.

BACKUP

duplicity *<dir> <url>*
Backup: incremental, or full if none yet

duplicity full *<dir> <url>*
Always start a new full chain

duplicity --full-if-older-than 1M *<dir> <url>*
Auto-full when last full is old

duplicity --dry-run *<dir> <url>*
Show what would happen

duplicity --exclude *<pattern> <dir> <url>*
Skip matching files

duplicity --include *<pattern>* **--exclude '**'** *<dir> <url>*
Back up only the included paths

duplicity --exclude-filelist *<file> <dir> <url>*
Read exclude rules from a file

ENCRYPTION

export PASSPHRASE=*<phrase>*
Symmetric GPG passphrase

ENCRYPTION (cont.)

duplicity --encrypt-key *<key-id> <dir> <url>*
Encrypt to a public key

duplicity --sign-key *<key-id> <dir> <url>*
Sign the backup

Back up your GPG private key and passphrase separately from the data, for example with gpg --export-secret-keys.

TARGETS

file:///*<abs-path>*
Local or mounted directory

sftp://*<user>***@***<host>***//***<abs-path>*
SFTP; a double slash means absolute

s3:///*<bucket>***/***<prefix>*
Amazon S3 (credentials from AWS_* env)

export FTP_PASSWORD=*<pw>*
Backend password via env, not URL

RESTORE AND VERIFY

duplicity *<url> <dir>*
Restore everything (dir must not hold files)

duplicity restore *<url> <dir>*
Same, with the explicit action

duplicity --path-to-restore *<relpath> <url> <dir>*
Restore one file or folder

duplicity -t *<time> <url> <dir>*
Restore as of a point in time

duplicity --force *<url> <dir>*
Overwrite existing target files

duplicity list-current-files *<url>*
List files in the latest backup

duplicity list-current-files -t *<time> <url>*
List files as of a time

duplicity verify *<url> <dir>*
Verify against stored signatures

RESTORE AND VERIFY (cont.)

duplicity verify --compare-data *<url> <dir>*
Also compare file contents with local

STATUS AND CLEANUP

duplicity collection-status *<url>*
List chains, sets and volumes

duplicity remove-older-than *<time>* **--force** *<url>*
Delete sets older than a time

duplicity remove-all-but-n-full *<n>* **--force** *<url>*
Keep only the last n full chains

duplicity remove-all-inc-of-but-n-full *<n>* **--force** *<url>*
Drop increments of older chains

duplicity cleanup --force *<url>*
Remove leftovers of failed runs

duplicity --archive-dir *<path> <dir> <url>*
Local signature cache location

COMMON OPTIONS

-v*<level>* **/ --verbosity** *<level>*
0 error, 4 notice (default), 8 info, 9 debug

-t, --time *<time>*
Point in time for restore, list, verify

duplicity --version
Print the installed version

RECIPES

Nightly encrypted backup to SFTP

```
export PASSPHRASE=<phrase>
duplicity --full-if-older-than 1M
\
  --encrypt-key <key-id> --exclude
'**/.cache' \
  /home/<user> sftp://
<user>@<host>//backups/home
duplicity remove-older-than 6M --
force sftp://<user>@<host>//
backups/home
```