Contents

- 1 Duplicity - Secure (gpg) Incremental (diff) Compressed Backup (tar)
  - 1.1 About Duplicity Encrypted Backup
  - 1.2 Install Duplicity Encrypted Backup
  - 1.3 Simple unEncrypted Backup over SCP
  - 1.4 Encrypted Backup over FTP
  - 1.5 Restore Encrypted Backup from FTP
  - 1.6 Restore a single old or deleted File from the Encrypted Backup
  - 1.7 Restore a folder from the Encrypted Backup
  - 1.8 List the Current Files in an Encrypted Backup
  - 1.9 Backup and Restore Debian Packages
  - 1.10 More Information on Duplicity
  - 1.11 Offsite Backup Restore

Duplicity - Secure (gpg) Incremental (diff) Compressed Backup (tar)

About Duplicity Encrypted Backup

Duplicity is a stable Debian package used for Encrypted backup using rsync algorithm. The benefits of duplicity are as follows:

- Encryption using GPG
- Differential Backups - Takes an inital full-backup, and from then on - only the files which have changed, saving bandwith and high disk usage.
- Can be done over ftp or scp - Some backup servers only allow ftp on a shared server. As a result files need to be encrypted.

Install Duplicity Encrypted Backup

```
apt-get install duplicity
```

Duplicity does not run as a service, and has no default global configuration file. It is more a program/application rather than a service. Duplicity can be run completely from the command line to backup or restore files.

Simple unEncrypted Backup over SCP

Setup ssh keys on the backup server allowing root to seamlessly login to the backup server. Setting up ssh keys can be found here Sshkeys.

```
//Backup of a specific Folder:
```

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

//Restore specific Folder:
duplicity scp://uid@other.host/some_dir /home/me
//or if you dont want to overwrite all files:
duplicity scp://uid@other.host/some_dir /var/tmp/me
```

Encrypted Backup over FTP

Generate a GPG key for encryption purposes. As the GPG key passphrase must be placed in a backup script - I did not use my personal gpg key - but generated a new one for backup purposes. Details on generating gpg keys are found here Gnupg.

To see available gpg keys:

```
gpg --list-keys
```

When running Duplicity on the command line (with no config file) - both the FTP password, and the GPG passphrase need to be exported to the environment variables.

```
export FTP_PASSWORD=ftppass
export PASSPHRASE=gpgpassphrase

duplicity --encrypt-key "69111111" --sign-key "69111111" --include /etc --include /home --include /r
//Syntax = duplicity | gpg_encrypt_and_sign | --include files | --exclude files | MAIN_FOLDER_to_BAC
//Thus the above line includes what I want, excludes '**' and backups / (root directory).
```

If there are errors etc. you need to check whether the environment variables (ftp and gpp passwords) are set. Whether your gpg key exists and matches the id, e.g. 69111111.

If you recieve the error: **Temporary error '450 No files found'. Trying to reconnect in 10 seconds.** This is specific to the version of python been used. If the folder on the ftp server (e.g. /backupfoldername as above) is empty - it can cause a problem. The quick solution is to touch a file, or put a temp file into the folder.

I made the following script and called it via a nightly Crontab.

```
#!/bin/bash
export FTP_PASSWORD=ftppass
export PASSPHRASE=gpgpassphrase

duplicity --encrypt-key "69111111" --sign-key "69111111" --include /etc --include /home --include /r
```

Also - you will have to manually backup the gpg key used for backup. Otherwise if you loose the key, or if it dies with the server - your backups are encrypted and you loose. I simply tar'd up the .gnupg folder in /root/ on the server been backup'd.

Restore Encrypted Backup from FTP

On the server you want to restore the backup to:

```
gpg --list-keys
```

```
//and make sure you have the key id 69111111. Otherwise it wont work!

mkdir /var/tmp/backupfoldername
export FTP_PASSWORD=ftppass
export PASSPHRASE=gpgpassphrase
duplicity --encrypt-key "69111111" --sign-key "69111111" ftp://username@backupserver/backupfoldernam
```

Thats all folks.

Restore a single old or deleted File from the Encrypted Backup

If you want to restore or recover a specific file from a Backup, and dont want to bother restoring the whole backup only to get 1 file, you can do the following:

```
duplicity --encrypt-key "" --sign-key "" --file-to-restore home/sburke/file.txt ftp://user@ftpserver
```

If the above file was deleted 1 day ago, and there was a backup since, you need to use the "-t 1D" option. E.g.:

```
duplicity --encrypt-key "" --sign-key "" -t 1D --file-to-restore home/sburke/file.txt ftp://user@ftp
#the number of days can be set, e.g. -t #D
```

Restore a folder from the Encrypted Backup

Actually this is the same as above using the --file-to-restore switch!

```
duplicity --encrypt-key "" --sign-key "" --file-to-restore home/sburke/Maildir/cur ftp://user@ftpser
```

List the Current Files in an Encrypted Backup

```
duplicity --encrypt-key "" --sign-key "" --list-current-files ftp://user@ftpserver/folderbackupname
```

Note the "-t 1D" did not seem to work with the above. This is quite annoying, as a search for the filename of the file that was deleted cannot be carried out. The user must know the exact file and exact path they are looking for. Otherwise the entire backup needs to be Decrypted, and the file picked out.

Backup and Restore Debian Packages

```
dpkg --get-selections > selections-$(date -I)
dpkg --set-selections < selections-$(date -I)
```

More Information on Duplicity

```
man duplicity (must have it installed via apt-get!)
```

http://www.debian-administration.org/articles/209

http://duplicity.nongnu.org/

http://savannah.nongnu.org/bugs/?2441 -> details the 405 error message

Offsite Backup Restore

So I downloaded a full offsite backup and wanted to test recovering files with duplicity. I got a Debian LiveCD, apt-get installed duplicty, and copied over the .gnupg folder, I had stored locally.

So I used:

```
duplicity restore --encrypt-key "xxx" --sign-key "xxx" --time-separator='_' file:///media//servernam
```

At first duplicity said "No backup signatures found". This didn't sound good, but uncle google led me to: https://bugs.launchpad.net/deja-dup/+bug/601243 which said to use "--time-separator='_'" Luckily this worked and all was sorted. The reason I used the LiveCD was that it had the same version of Duplicity it used.

As I also used a livecd (http://live.debian.net/) /tmp space for duplicity was limited. As a result, I added:

```
--tempdir /mnt/tmp
#see http://duplicity.nongnu.org/duplicity.1.html
```

A couple of things on the way: I wanted to connect my external USB drive so I had somewhere to extract from with the LiveCD. I found: http://xjqian.wordpress.com/2007/11/11/readwrite-ntfs-in-debian/ which provided:

```
apt-get install ntfs-3g
mount -t ntfs-3g /dev/hda1 /media/win -o umask=0,nls=utf8
```

I could now happily mount and write to my ntfs external drive. Unforunately I got some errors when extracting with duplicty. They were:

```
Invalid or incomplete multibyte or wide character
```

I think this was down to the ntfs drive I had. It was a file with a very long file name that caused the issue. Luckly I just worked around this folder with the following:

```
duplicity restore --encrypt-key "xxx" --sign-key "xxx" --time-separator='_' --file-to-restore home/u
```