

Introduction

Here at the Linux Supporters Group Adelaide we use GPG (Gnu Privacy Guard) as our encryption method of choice, because it is open source and has no arbitrary restrictions. This talk simply aims at introducing you to the basics, so you are familiar with them when we have a couple of talks next year on the detailed mathematics of encryption, using encryption in email, managing keys, and other advanced topics. There are two basic but very useful `bash` scripts named `store` and `restore` on our site at <http://linuxlsga.net/tutorials>.

INSTALLING THE GNU VERSION OF PGP

You get this from your distribution repository and for Debian the package is called `gnupg`.

```
$ sudo apt-get install gnupg
```

GENERATING A PUBLIC AND PRIVATE KEY

This is how you generate a public and private key once-off at the start; move the mouse when asked in order to generate randomness used to select your prime integers for the key.

```
$ gpg --gen-key
This asks for method (DSA & ElGamal); [the default is OK]
key size (2048): [the default is OK]
validity (0): [the default is OK]
Real Name: [eg] John Doe
email address: [eg] jdoe@email
Comment: [eg] jdoe (your username on this computer, so the script works!)
passphrase: [eg] ! good long passphrase with strange %$#@(*&~%
and it stores the keys in the ~/.gnupg directory.
```

LISTING AND SHARING YOUR PUBLIC KEY

You may list your public key and send it to others in ASCII form by:

```
$ gpg --list-keys
$ gpg --export -armor
```

ENCRYPTING AND DECRYPTING A DIRECTORY

To get familiar using `gpg`, use the two scripts overleaf.

- First make sure your path contains the directory `/usr/local/bin` by executing `export PATH=$PATH:/usr/local/bin`.
- Then get into the directory you want to encrypt by typing `cd DIRECTORYNAME` and then type `store`. You will be left with an encrypted archive called `store.tar.gpg`.
- To recover the directory, get into the directory you want to decrypt, which should have only the file `store.tar.gpg` in it, then type `restore`. You will be left with the original contents of the directory.

BASH SCRIPT FOR ENCRYPTING A DIRECTORY

Enter this script into a file called `store` and make it executable by `chmod +x store` and move it to a directory in your path like this: `sudo mv store /usr/local/bin/store`.

```
#!/bin/bash
# /usr/local/bin/store

# if store.tar.gpg already exists in this directory, do nothing, since another
# process is already using it or it is left over when you aborted this store run
if [ -e store.tar.gpg ]
then
    echo -e "\007store.tar.gpg already exists in this directory $(pwd)"
    echo "nothing done"
    exit 1
fi

# make an archive, encrypt it, and then remove all files
tar --exclude=store.tar.gpg -c -v -f - .|gpg --encrypt -r $USERNAME -o store.tar.gpg
/bin/ls -A|grep -v store.tar.gpg|tr '\n' '\0'|xargs -0 /bin/rm -rf
echo "completed encrypted store -> store.tar.gpg"
```

BASH SCRIPT FOR DECRYPTING A DIRECTORY

Enter this script into a file called `restore` and make it executable by `chmod +x restore` and move it to a directory in your path like this: `sudo mv store /usr/local/bin/restore`.

```
#!/bin/bash
# /usr/local/bin/restore

# are there any encrypted contents?
if [ ! -e store.tar.gpg ]
then
    echo -e "\a store.tar.gpg does not exist in this directory $(pwd)"
    echo "nothing done"
    exit 1
fi

# if files have been added since store, it might not be safe when extracted
if [ $(/bin/ls|grep -v store.tar.gpg|wc -l) -gt 0 ]
then
    echo -e "\a store.tar.gpg is not alone in this directory"
    echo "nothing done"
    exit 1
fi

gpg --output - store.tar.gpg|tar -x -v -f -
rm store.tar.gpg
echo "completed restore of encrypted directory $(pwd)"
exit 0
```