

Introduction.

Welcome a to short presetation on GnuPrivacy Guard to the West Yorkshire Linux Users Group by Simon Wood.

I have not had anything to do with the development of this project, I am just a 'happy user'.

Most of the information in this presentation has been taken from my personal experiences and the Gnu Privacy Handbook, available from the GnuPG website.

I hope to inspire you all to download and use GnuPG for your secure (and normal) email communictions.

On with the presentation....

What is Gnu Privacy Guard?

- GnuPG is an Encryption program.
- Uses Public Key Encryption.
- Main use is for email (but there are others).
- Licensed under the GPL.
- Developed in Germany, to avoid US Export Restrictions.

You may ask 'why should I use it?', mainly to protect your privacy. Email is very insecure, who has the right to know what you write in your emails, where your going on holiday, who you fancy, how much is on your bank account, etc...

What is Public Key Encryption?

Public Key Encryption is a scheme that uses a pair of keys, rather than a single one. When one key is used to encrypt a document, the other must be used to decrypt the result.

These keys are commonly referred to as Public and Private keys, you distribute your Public key and keep the Private one secret. People can then encrypt emails to you with your public key, knowing that only you can decrypt it (with your Private key).

Main disadvantage of Symmetrical Encryption (that uses a single key) is that it is difficult to distribute the key in a secure manner, in order for people to send you secure email.

Where do I get GnuPG?

GnuPG is freely available from their website:

<http://www.gnupg.org>

It is (currently) legal in the UK, although there may be restrictions in it's use in other countries.

There are also Handbooks, usage guides and other information on how to use it. This is where I got most of my information (as well as the Gnu picture below).

How do I build and install GnuPG?

GnuPG is built using the usual Unix commands.

```
tar -xzf gnupg-1.0.1.tar.gz
```

```
cd gnupg
```

```
./configure
```

```
make
```

and then as root

```
make install
```

A quick check that everything is working can be done

```
gpg --help
```

Generating a Key Pair.

Before GnuPG can be used you will have to generate a Public and Private Key.

```
gpg --gen-key
```

Answer questions and follow instructions. It is common to get an error message about the lack of randomness – just cause the computer to do some work and the generation will then complete.

The keys are then placed on your ‘Keyring’, a storage space for your keys and other public keys of your friends and colleagues. This is normally kept in the directory `~/.gnupg/`

You can list your keys with

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

Revoke Certificate.

In some cases you might want to cancel your key pair, and notify people that they should no longer use this key. This might be that the key is no longer secure, or that you have forgotten your pass phrase.

This is done with a 'revoke certificate', which can be created with

```
gpg --armor --gen-revoke simon@wylug
```

When needed the certificate can be sent to a key server, and will then mark your public key as 'Do not use'.

Note: you should generate a revoke certificate when you create a key pair and keep it in a safe place (ideally printed out).

Encrypting and Decrypting.

To encrypt a message to a friend you would use the command.

```
gpg --armor --encrypt --recipient  
joe@bloggs message.txt
```

This will output the encrypted file to the screen, but can be redirected to a file with ‘`--output`’.

The armor option ensures that the output is 7 bit friendly, important for email.

To decrypt a message you use the command.

```
gpg --decrypt secret_message.gpg
```

You will need to enter you passphrase, and then the message will appear on the screen.

Exporting and Importing Keys.

In order to enable people to send you encrypted messages you will have to 'export' your Public Key. This is done with

```
gpg --armor --export simon@wylug
```

This can then be emailed, posted on your web site or placed on a key server (more later). You can export any of the public keys on your keyring, so you can pass on keys to other people.

When you receive a Public Key from a friend you need to add it to your keyring. Again this is simple.

```
gpg --import me@home
```

Signing.

One major advantage of the Public Key system is that you can 'Sign' documents. This is similar to a checksum, but is made with your private key.

Other people can then verify that the document is intact and has not been altered.

You can sign in three ways.

- '--sign' – compressed and signed.

- '--clearsign' – document viewable with a 'signature' appended to the end.

- '--detach-sign' – signature is a separate file.

A signature can be checked with the verify command.

```
gpg --verify simon@wylug secret.gpg
```

Key Management.

Once you have a few keys on you keyring you might have to remove or change the properties of the keys. This is handled with.

```
gpg --edit-key simon@wylug
```

This starts a command session, with it's own set of commands. Type help for more information.

You can change details such as:

- trust
- expiry date
- passphrase
- user id

Key Fingerprints.

This encryption scheme is great, but how do you know that the key that you think is Peter's is really his? Perhaps someone has tricked you and will intercept all of your messages to him.

All keys have a fingerprint, a short ID string, that can be checked with the key holder. You could always phone Peter and ask for his key's fingerprint.

```
gpg --fingerprint simon@wylug
```

Web of Trust.

In order to prevent you having to personally contact every one who you wish to send a message to confirm their key fingerprint GnuPG uses a trust system.

You can sign the keys of your direct friends and place a level of trust in them. This way you are confirming that you believe this to be the real key – you should only sign keys if you are 100% certain.

Say I want to send Sue a message, her key has been signed Peter and I trust Peter. This means that I can be sure that Sue's key is really hers and can use it without having to confirm the key's fingerprint with her.

Key Servers.

Keyservers hold lists of people and their keys. These keys are often signed by 'validated' signers.

Keys can be sent to a server with

```
gpg --keyserver certserver.pgp.com  
--send-key simon@wylug
```

and retrieved with (this requires that you know the key ID)

```
gpg --keyserver certserver.pgp.com  
--recv-key 0xBB7576AC
```

The major keyservers will synchronise with each other.

Key Signing Parties.

In order to propagate your public key there is the idea of a key signing party.

You can organise a meeting where people can bring their public keys (usually on a floppy), with a passport, driving license, etc.

A 'validated' user will then sign their keys – confirming that they are who they say they are. These keys can then be sent to a key server.

If enough people are interested we will organise a WYLUG signing party.

Beyond the command line.

As in the case of many great Unix applications, GnuPG is command line based. There are some email applications which are planning to integrate support for GnuPG.

- Arrow
- Mahogany
- Mutt

Summary.

GnuPG can be used to secure and authenticate communications.

Freely available from

<http://www.gnupg.org>

I'm Simon Wood and my public key can be found at

<http://www.mungewell.ndirect.co.uk>