**Lab title:** Symmetric (secret key) and asymmetric (public key) encryption.
**Learning outcomes:** Learn how to perform symmetric and asymmetric encryption.

In this lab we will use OpenPGP which is a non-proprietary protocol for encryption using public key cryptography. It is based on the original PGP (Pretty Good Privacy) software. Specifically we will use GNU Privacy Guard (GnuPG or gpg) which is an open-source implementation of OpenPGP. We will use this tool via the Linux terminal. Descriptions of how to use this tool can be found at:
**https://www.gnupg.org/gph/en/manual.html**
**https://www.dewinter.com/gnupg_howto/english/GPGMiniHowto.html**

**Symmetric encryption**
We will now encrypt and decrypt a file using symmetric key encryption. Create a file which you wish to encrypt using symmetric encryption (for example secret.txt). With the -c option, gpg encrypts the file using a symmetric key. The file secret.txt can be encrypted using the following command:

gpg -c secret.txt

This generates an encrypted file with extension gpg (secret.txt.gpg). Compare the sizes of the original and encrypted files. With the -d option, gpg decrypts the file using a symmetric key. The file secret.txt.gpg can be decrypted using the following command:

gpg -d secret.txt.gpg

**Asymmetric encryption**
We will now encrypt and decrypt a file using asymmetric key encryption.

**1. Generating public and private keypair**
Enter the following command into the terminal
gpg --full-gen-key

A list of options corresponding to different public-key cryptosystems will be provided. Hit enter to select the default RSA algorithm. RSA is the most popular asymmetric cryptographic algorithm, and is built on the difficulty of factoring extremely large composites (a composite number is a positive integer greater than one which is not prime).

GnuPG next asks you to choose the length of your keys. Longer keys are more secure but take longer to compute. For the purpose of this lab, press enter to choose the default length of 2014 bits.

Next you will be asked how long you want the key to be valid for. Make an appropriate choice.

The generator will next ask you to answer a series of questions.

Eventually a dialog box will appear asking for a passphrase. This is necessary to ensure that even if someone does gain access to your secret key, they will not be able to use it.

Finally, gpg will begin to generate a random key. This requires many random bits and therefore you will be asked to perform random actions such as moving the mouse randomly. If you chose a large key size earlier this might take a while. However this only occurs once per keypair that you generate.

Following key generation a block of text will be displayed. The public key ID is a long random string of characters on the line following pub.

To return a list of keypairs stored on your computer (your keyring) enter the following command:

gpg --list-keys

The keypair just created should be displayed.

**2. Exporting and Importing Keys**
To get somebody else's public key into your keyring, they need to export it and you need to import it.

You can export a public key to a file using the command:
gpg --export PUB_KEY_ID > PUB_KEY_ID.pub

Where PUB_KEY_ID is the key ID and looks like a long random string of characters. This will create a file with extension pub. This file can then be shared and imported by somebody into their local keyring using the command (you will need to share this file using an email attachment or USB memory stick):

gpg --import PUB_KEY_ID.pub

**3. Publish public key to keyserver**
An alternative way to share keys is to use a keyserver. A keyserver is a repository of public keys. It allows other people to import your public key into their local keyring given your ID. To publish your public key to a keyserver enter the following command:

gpg --send-keys PUB_KEY_ID

GnuPG will default to using the keys.gnupg.net public keyserver; most public keyservers synchronize with one another, so you can safely leave this as the default. You can confirm that your key has been sent by going to the site https://pgp.mit.edu/ and searching for your name.

You may import another individual’s public key into your local keyring from the keyserver. This can be achieved using the following command (replace PUB_KEY_ID with the public ID of the person in question):

gpg --recv-keys PUB_KEY_ID

To return a list of keypairs stored on your computer (your keyring) enter the following command:

gpg --list-keys

The keypair just imported should be displayed.

**4. Encrypting a document**
Create a file which you wish to encrypt using asymmetric encryption (for example secret.txt). You can encrypt this file using the following command:

gpg --encrypt secret.txt

You will be asked to enter the ID for the recipient of this document. The ID in question must be into your local keyring. For now use the ID you just created.

**5. Decrypting a document**
To decrypt a document use the following command:

gpg --decrypt secret.txt.gpg

You will be asked for the private key corresponding to the public key used to perform encryption. If the private key is already stored on your keyring you will be asked for the password to access it.

Communicate with one of your fellow students using asymmetric encryption.