Laboratory Sheet 2

1. Learning Outcomes

- Introduction to public key cryptography

2. Organisation / More Info

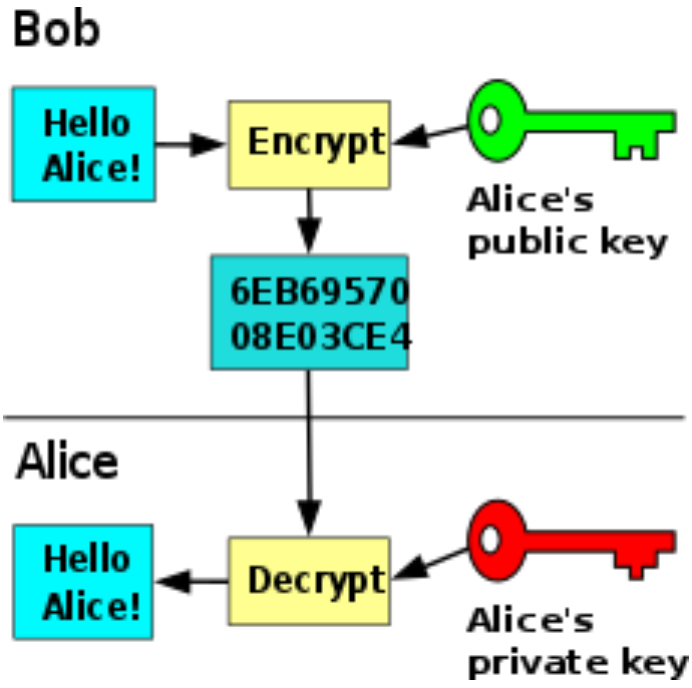
- https://www.devdungeon.com/content/gpg-tutorial

Cryptography

Last week we saw the many security vulnerabilities of online communication. One simple method of eavesdropping on communication is using a wiretap. A wiretap allows anybody with physical access to an internet cable to collect all information travelling along that cable.

Because of this, it is important to assume that no line is secure. If we want to send data securely over a wire, we need to make sure that we encrypt it. The Linux program **GNU Privacy Guard (gpg)** makes this easy for us. GPG relies on *public private key (PPK) cryptography*. In PPK, everybody has two keys, a public key, which they make available freely to anyone who wants to send them an encrypted message, and a private key, which they keep secret. Messages encrypted with the public key can only be decrypted with the private key. This allows us to send a message to someone that only they can decrypt.

For example, Alice wants to send Bob a sensitive file, but she is worried that Eve might have tapped the connection between them. Alice asks Bob to send her his public key. Alice uses Bob's public key to encrypt the file and sends it to him. Bob uses his private key to decrypt the file. Even if Eve has a copy of Bob's public key, she cannot use it to decrypt the file. Provided Bob never sends his private key over the wire they can assume their communication is secure.



Generating a Keypair

In today's lab we are going to generate a public-private keypair using gpg. We are then going to use these keys to send encrypted files. For this exercise you will work in groups of 3 or 4.

1. Start up your codespace and open a terminal. Run **gpg --full- generate-key** to interactively generate your key-pair
2. Select the default RSA and RSA encryption algorithm
3. Make your key 4096 bits long (bigger is more secure)
4. Make your key valid for 1 month
5. Enter your name and TU Dublin email address to associate with your key. Your email address is like your user ID, and will be used to identify your key
6. Choose a password for the private key, this is an extra layer of protection, anyone who manages to get their hands on your private key will need to know the password to use it.
7. Wait while the key is generated; when finished the public and private keys will be saved to your keyring

Exporting your Public Key

Now you have generated your keypair, but it is saved directly into the gpg programme. We need to export the public key to a file so we can share it.

1. Run gpg –list-keys to make sure your public key is saved
2. Export your public key **gpg --output <email address>.key –export <email address>**

When that is finished, you should find a newly created file <email address>.key in the current directory. Use the ls command to make it sure it is there.

3. Open a browser and log in to your outlook web app.
4. Send an email to the other people in your group, attach the public key file

Importing a Public Key

When you receive the public keys from your group members you can import those keys so you can begin exchanging encrypted messages.

1. Download the attached public keys to your Downloads folder

2. Go back to your terminal and change directory (cd) into the Downloads folder (remember Linux is case-sensitive)
3. Import the keys by running **gpg --import <public key filename>** for each file
4. Ensure the files have been added correctly by running **gpg --list-keys**

Encrypting a Message

Open Text Editor and create a new file in your Downloads folder. You are going to send this to your group members. The file can contain whatever you like.

1. Run **ls** to ensure the file is in the current directory
2. Encrypt the file with gpg, you need to do this once for each recipient. The following command is all on a single line **gpg --output <recipient> --encrypt --recipient <recipient email address> <unencrypted filename>**
3. Email the encrypted files to each recipient.

Decrypting a Message

When you receive your encrypted files, save them to your Downloads directory. Decrypt the file using gpg

gpg --output <filename>.txt --decrypt <encrypted filename>

You will be asked for the passphrase to unlock your secret key. Enter it and the encrypted file will be saved to your Downloads directory

Share your Public Key with the World

Public keys do not need to be kept safe. You can share it with the world. You are going to finish this section by adding your public key to your blog. By default, when you export a public key, it is exported as a file. This can be difficult to share. The *armor* format allows you to generate your public key as text which is better for sharing online.

Export your public key

gpg --armor --export <your email address>

Copy the generated text and add it to your blog.