Contents

1. Creating a new key pair (Gpg command)

GNU Privacy Assistant is a graphical user interface for the GnuPG (GNU Privacy Guard).

```
sudo apt-get install gpa
```

Generate a Keypair

This will involve using the command line. Launch Terminal.app (or your preferred terminal emulator) and do this:

```
gpg2 --gen-key
```

You'll see:

```
gpg (GnuPG/MacGPG2) 2.0.12; Copyright (C) 2009 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)

Your selection?
```

Hit enter, since the defaults tend to work fine. Then:

```
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
```

You may want to go for the default. After deciding, you'll be presented with this screen:

```
Requested keysize is 2048 bits
Please specify how long the key should be valid.
        0 = key does not expire
     <n>  = key expires in n days
     <n>w = key expires in n weeks
     <n>m = key expires in n months
     <n>y = key expires in n years
Key is valid for? (0)
```

Hit enter again. It'll also ask you for confirmation; just hit y then enter again. The generator will then ask you to answer a series of questions:

```
GnuPG needs to construct a user ID to identify your key.

Real name: <Type your full name, hit enter>
Email address: <Type your email address, hit enter>
Comment: <Optional: type a comment or your homepage URL, hit enter>
You selected this USER-ID:
   "[ultimate] Mahmoud <mahmoud.aboualy@arcada.fi>"
```

After that it will ask you to confirm:

```
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit?
```

Just type O (that's the capital letter, not the numeral) and then enter. At this point a dialog box will appear asking for a passphrase.

At this point, you'll see something like this:

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
```

You should see something along the lines of:

```
gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
pub   rsa2048 2017-04-09 [SC]
      E866D0A59499155CE3DD967CC9756B4B1A0F5EAD
uid           [ultimate] Mahmoud <mahmoud.aboualy@arcada.fi>
sub   rsa2048 2017-04-09 [E]
```

2. Creating a text file that i first encrypt with my PGP public key and then decrypt with the my PGP private key, just to test

- sudo nano doc

To encrypt a document the option --encrypt is used. You must have the public keys of the intended recipients.

- gpg --output doc.gpg --encrypt --recipient mahmoud.aboualy@arcada.fi doc

To decrypt a message the option --decrypt is used. You need the private key to which the message was encrypted.

- gpg --allow-secret-key-import --import <keyring>

- gpg --decrypt doc.gpg > secret.txt

3. Download the group member's PGP public key to Ubuntus "keyserver", keyserver.ubuntu.com,

- gpg --send-keys --keyserver keyserver.ubuntu.com C9756B4B1A0F5EAD
- gpg --send-keys --keyserver keyserver.ubuntu.com $GPGKEY

4. Download the course teacher's public PGP key from http://keyserver.ubuntu.com

- gpg --search-keys --keyserver keyserver.ubuntu.com 'Jonny Karlsson'

```
ubuntu@euca-172-16-0-230:~$ gpg --search-keys --keyserver keyserver.ubuntu.com '
Jonny Karlsson'
gpg: data source: http://cassava.canonical.com:11371
(1)     Jonny Karlsson (Jonnys Publika PGP-nyckel) <jonny.karlsson@arcada.fi>
          2048 bit RSA key 5BB156412FBEA057, created: 2017-04-05
(2)     Jonny Karlsson (Kurslärarens publika nyckel) <karlssoj@arcada.fi>
          2048 bit RSA key 06DDBAF6FFDE3D74, created: 2017-04-05
(3)     Jonny Karlsson (Jonnys kursnyckel) <karlssoj@arcada.fi>
          2048 bit RSA key 28CD8B46C1933FFA, created: 2015-01-20
(4)     Jonny Karlsson <jonnyk@hbl.nu>
          1024 bit DSA key F066929C4C584647, created: 2002-08-12
(5)     Jonny Karlsson <jonnyk@hbl.nu>
          1024 bit DSA key 796C0AE244226F91, created: 2002-07-03
(6)     Jonny Karlsson <jonny.karlsson@rivermen.se>
          1024 bit DSA key A460E33CA6EC9C96, created: 2000-12-11
Enter number(s), N)ext, or Q)uit >
```

Choose number 1

- gpg --list-keys

```
pub   rsa2048 2017-04-09 [SC]
      E866D0A59499155CE3DD967CC9756B4B1A0F5EAD
uid           [ultimate] Mahmoud <mahmoud.aboualy@arcada.fi>
sub   rsa2048 2017-04-09 [E]

pub   rsa2048 2017-04-05 [SC]
      DCBFF758E3AE739E99FD54955BB156412FBEA057
uid           [ unknown] Jonny Karlsson (Jonnys Publika PGP-nyckel) <jonny.karlsson@arcada.fi>
sub   rsa2048 2017-04-05 [E]
```

5. Encrypt a text file with the course teacher's public PGP key and send the encrypted.

To encrypt a document the option --encrypt is used. You must have the public keys of the intended recipients.

- gpg --output doc1.gpg --encrypt --recipient jonny.karlsson@arcada.fi doc

6. Decrypt the teacher´s file:

To decrypt a message the option --decrypt is used. You need the private key to which the message was encrypted.

- gpg --decrypt mahmoudRESPONSE.gpg > secret1.txt

The content of given decrypt file is :

Labben godkänd 19.3 kl 22:26!