



Keysigning Guide v1.0

OpenPGP

OpenPGP is a standard created to provide a greater level of digital security. The standard is cross platform and there are free programs for Windows, Linux and MacOS X. Pretty Good Privacy (PGP) is actually a commercial proprietary product that implements OpenPGP, there is also a free alternative called Gnu Privacy Guard (GnuPG)

Pretty Good Privacy is a technique that can be used to protect your privacy or to prove your identity. It is based on maths and the use of Public and Private Keys.

Each key is able to be used to encrypt a document very securely, so that only the matching key can decrypt it. A user can use the public key to encrypt something, but only the matching private key can be used to decrypt it. The public key once used to encrypt a file cannot be used the opposite way to decrypt it.

Example

Public Key A can encrypt document SECRETS. Only Private Key A can decrypt it.
Public key A cannot decrypt the document SECRETS.

The public key is handed out to all those who want to communicate securely with you, you can also publish it on your webpage or on a public keyserver. The security of the system does not depend on keeping your public key secret. On the other hand, the private key must be kept safe, secure and secret at all times.

A private key can be used to encrypt a file, but this is useless, as the public key will be known to many people, and therefore anyone can decrypt the file. If you want to encrypt a file to send to another person you must use their public key.

Most public keys are published to key servers, and you can download other peoples from the same key servers. Uploading to one keyserver is enough, as they all sync and share their keys over the course of a few days.

Example

Bob wants to send SECRETS to Joe.
Bob downloads Joe's public key from a keyserver.
Bob encrypts SECRETS with Joe's public key and sends it to Joe.
Joe decrypts SECRETS with his own private key.

Signing

Using your private key you could also "sign" a document or an email. This is where the computer performs a mathematical function on the document using your private key to produce a hash. The hash becomes unique to that document.

This is especially important for transferring documents via a unsecured medium, eg email. Any alterations to the document before it reaches the intended recipient can be detected. This is because the recipient can check the documents hash using the senders public key, if the hash sent with the document doesn't match the hash generated by the recipient they can be confident tampering has taken place. Signing emails is one of the most common uses for using OpenPGP.

Most email clients can be set up to automatically sign emails before sending them. So once you have it set up then it does not require anywork, it just happens automatically.

Example

Joan wants to send a signed message via email to Dilbert.
Joan produces a hash using her private key and her message.
Joan sends the message and the hash via email to Dilbert.
Dan intercepts Joan's email and changes the message.
Dan creates a new hash, but can't use Joan's private key.
Dan forwards on the message to Dilbert.
Dilbert receives the email and checks the hash using Joan's public key.
The hashes do not match, Dilbert now knows the email has been tampered with.

Expiry Dates and Revocation Certificates

Over time a key could become compromised, or you may decide to create a new key for other reasons. OpenPGP keys can have a expiry date built in, after which a key becomes invalid.

When you create your OpenPGP keys, you have the option to create a revocation certificate. This should be kept secure, like your private key. If you believe that your Private key has become compromised then you can use the revocation certificate to revoke the key. This would allow others to know that they can no longer trust your key or anything encrypted or signed with it past a certain date.

If your key expires or you revoke it, you have to start from scratch and get your new key signed all over again.

Key Signing

The Public/Private key system is dependant on how much you *trust* that the name on the key is the actual person you want to communicate with. Trust here has a specific and limited usage, it is not that they are a necessarily a good person, but rather that you are sure the person is who they say they are.

To create this trust, key signing parties have been invented. To confirm someone's identity you need to have seen them, checked a trusted form of ID and then exchanged keys. If you trust another persons judgement when it comes to checking keys, you can use the keys they have signed, this creates an interconnected "web of trust".

If you want your key signed you hand over a piece of paper with your public key "fingerprint" on. This will have all the emails that you have associated with your public key, and their identifying fingerprints. Someone can then check the name on your public key, check your photo ID, and then mark the paper with your keys fingerprint on to show that they have checked your ID. Though many may not actually sign your key their and then, they could. This is the first stage of a key signing. The rest is done later.

Once you get home or find some time at the conference, you can download the persons public key from a keyserver. And using your preferred program you can sign the public key you just downloaded and upload it back to a keyserver.

An extra stage is to check the email address on the public key belonging to the persons whose name is on it, doing this will confirm ownership of the key and prevent middle-man exploitations. Challenging the key is done by emailing all the email address on the fingerprint, with a block of random data that has been encrypted with the key being challenged, and signed with your private key. If the recipient is correct they should be able to decrypt the random data and return it, signed with their private key. This reply can be checked against the original random data, and so too can the signature.

Preparation Instructions

The general formula for a keysigning party is :

1. Create your key and publish it
2. Print off multiple copies of the fingerprint of your key
3. Bring the fingerprints to they key signing party to be signed
4. Go home and sign other people's keys.

There are three guides that follow:

- A. All platforms using the command line gpg client.
- B. Linux/Unix using 'seahorse', a GNOME based client.
- C. Windows using the Gnu Privacy Assistant that comes with the gpg4win suite.

It makes no difference which one of these that you use, the resulting keys are cross platform so you can use the same keys on both Windows and Linux. However, the more machines that you store your private key on, the greater the chance of it being compromised.

When you have the keys set up, you are likely to use your own email client to do the actual day to day signing of email. Default support or plugins are available for many clients including:

- Evolution – in by default, select Tools > Settings > Mail Accounts > Edit > Security.
- Mutt – in by default, add some settings to .muttrc
- Outlook – get the plugin, part of gpg4win (see below).
- Thunderbird – get the enigmail plugin <http://enigmail.mozdev.org/>
- Firefox – the FireGPG extension aims to allow you to use GPG in web mail, currently works for Gmail - <http://firegpg.tuxfamily.org/>.

To include OpenPGP support in your Python programs, you can use the *pypgpme* module, see <http://cheeseshop.python.org/pypi/pygpgme> for more details.

A. All platforms (command line)

GnuPG - <http://www.gnupg.org/>

Gnu Privacy Guard is the free software implementation of OpenPGP. It is included by default in many Linux/Unix distributions, or it has been packaged and usually easy to install. See your distribution's documentation.

Preparation for the key signing party

1. Generate a key pair

```
$ gpg --gen-key

Please select what kind of key you want:
  (1) DSA and Elgamal (default)
  (2) DSA (sign only)
  (5) RSA (sign only)
Your selection?
```

Hit enter to use the default

```
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
```

Hit enter to use the default

```
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 if you don't want an expiry date. If you do set a date, make sure you remember to create a new key after it!

```
You need a user ID to identify your key; the software constructs the
user ID
```

```
from the Real Name, Comment and E-mail Address in this form:
  "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"
```

```
Real name: Bob Jones
E-mail address: bob@jones.com
Comment: Personal
You selected this USER-ID:
  "Bob Jones (Personal) <bob@jones.com>"
```

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

Enter your full name, email address and comment (Usually "Personal" or "Work"), then

At the Key Signing Party

Bring your fingerprints and photo ID to the key signing party. Give a fingerprint to anybody you wish to sign your key, allow them to verify your identity. Verify in person, the identity of anybody you accept a fingerprint from. Ensure the name on the fingerprint matches the name on the ID.

At home or later at the Conference you will need to actually sign peoples keys, it is rude to accept key fingerprints but not sign them.

7. Download and Sign the keys

Download key

```
$ gpg --keyserver ldap://keyserver.pgp.com --recv-keys <Key_ID>
$ gpg --fingerprint <Key_ID>
```

Sign key

```
$ gpg --sign-key <Key_ID>
$ gpg --keyserver ldap://keyserver.pgp.com --send-key <Key_ID>
```

9. You are Done!

B. Linux/Unix (GNOME)

Seahorse - <http://www.gnome.org/projects/seahorse/>

Seahorse is a graphical front end to the GnuPG program. This means you will have to install GnuPG for Seahorse to function. Installing seahorse via your package manager should take care of that.

1. Key --> Create New Key
2. Choose "PGP Key"
3. Enter your full name (Seahorse will not continue until the next step unless you put in your first name and your surname), email address and a comment. The comment field is for things like "Personal Email" or "Work".

The advanced key options defaults are okay, for extra security you can have an expiry date on your key if you want. Make a note of this date, after this date you will have to create a new key.

4. Your key is created. It is made using random information gathered from your machine, a combination of keyboard strokes, mouse movement, network traffic and disk activity.
5. You now have a public and private keypair, and you need to publish the public key.

Remote --> Sync and Publish Keys

Select "Sync"

6. You need information from Seahorse to create a fingerprint of your public key. This fingerprint is what is handed out at the key signing party.

Right click on your key and select "Properties", under the "Details" tab is the information you need. Copy the "Key ID", and "Fingerprint" into a text editor and put it together with the name you put on your key and the email address.

Name : Bob Jones

Email: bob@jones.com

Key ID: 4D63FF42

Fingerprint: 363A 6B4A 3057 4B38 D76C 73BF 28C4 F148 4D63 FF42

That is it, you are ready for the key signing party.

At the Key Signing Party

Bring your fingerprints and photo ID to the key signing party. Give a fingerprint to anybody that you wish to sign your key, allow them to verify your identity. Verify in person, the identity of anybody you accept a fingerprint from. Ensure the name on the fingerprint matches the name on the ID.

At home or later at the Conference you will need to actually sign peoples keys. It is rude to accept key fingerprints but forget to sign them.

7. Find Remote Keys, enter the name of the persons key you want to download.

8. You will be presented with a list of keys that match the name. Check the Key IDs to make sure you download the correct person's key.

Click the key you want, then click "Import".

9. Return to the main Seahorse window. Under the "Other Collected Keys" tab the key you just downloaded should be there. Right click and choose "Sign..."

10. Choose how well you trust that key. Seahorse gives good descriptions of the different levels. Click "Sign". And your done.

11. You will now need to sync your keys to the key server again

Remote --> Sync and Publish Keys

Select "Sync"

12. Give yourself a pat on the back. you are done.

C. Windows

gpg4win - <http://www.gpg4win.org/>

Gpg4win is a collection of Windows programs for using OpenPGP. This guide uses the Gnu Privacy Assistant program, though it is packaged with an alternative, WinPT. It is up to you which one to use.

Creating the keys

1. Open GPA (Gnu Privacy Assistant). It should recognised that you want to create a new key pair.
 2. Enter your name.
 3. Enter your email address.
 4. Enter your passphrase, GPA will prompt if it thinks your passphrase is not secure enough.
 5. Create a backup.
 6. Click apply and GPA will generate a key for you.
 7. Back up your key, once saved keep your key on a safe medium, CD or pen drive. GNU Privacy Assistant suggests a floppy disk but they have been known to corrupt data held on them easily.
 8. You should now have GPA showing your key.
 9. Publish your key.
Server --> Send Keys

Click Yes.
- Note, GPA creates key with no expiration date, if you want to add one right click on your key and click "Edit Private Key" from here you can add an expiry date.
10. Creating a fingerprint

The main screen of GPA will show your key. If you click on your key GPA will provide you with a summary of that keys information at the bottom of the window. Copy and paste the User Name, Key ID, and Fingerprint into your favourite text editor. This will be the fingerprint you give to others to check, print of as many as you think you will need.

Name : Bob Jones
Email: bob@jones.com
Key ID: 4D63FF42
Fingerprint: 363A 6B4A 3057 4B38 D76C 73BF 28C4 F148 4D63 FF42

11. At the Key Signing Party

Bring your fingerprints and photo ID to the key signing party. Give a fingerprint to anybody you wish to sign your key, allow them to verify your identity. Verify in person, the identity of anybody you accept a fingerprint from. Ensure the name on the fingerprint matches the name on the ID.

12. At home or later at the Conference you will need to actually sign peoples keys. It is rude to accept key fingerprints but forget to sign them.

Download the key you want to sign.

Server --> Retrieve Keys

Enter the Key ID of the key you want to download. GPA will then search a keyserver and download the key

13. Sign the key

In the main GPA windows right click the key you want to sign, and select "Sign Keys..."
You will be prompted for your private key password

14. Upload the key

Right click the key you just signed and select "SendKeys to Server...", and then click "Yes".

15. We are done, go to the pub.

**Written by Ciarán Mooney
(edited by Zeth Green)**

Please send corrections/improvements to ciaran.mooney@gmail.com

Feel free to share with your friends and neighbours under the following licence:

[Creative Commons Attribution-Share Alike 2.0 UK: England and Wales License](https://creativecommons.org/licenses/by-sa/2.0/)