# How to Implement Key Signing Party

**Keyne Kassapa**
University of Indonesia

January 2017

**Abstract**

The purpose of this event is to verify and sign keys at any time and help to extend the WOT. A key signing parties significantly is a gathering of PGP users with a purpose to meet other PGP user and sign each other key. There are lots of benefits we can get from Key Signing Party. First, we can build tightly linked web of trust which will make it difficult to defeat. For developers and users, this will be a special significance to the Free Software Community. The community rely on PGP to cryptograph in purposed to protect their software and security. Second, key signing parties will make peoples get integrated to the security culture. It also encourage them to gain more understanding of PGP and other cryptography worldLast, it will build communities which purposed to get to know each other, network, and discuss important issues like the regulation, etc.

## BEFORE THE PARTY

1. Generate your PGP keypair / gnupg keys.

   $ gpg --gen-key

2. Select your key type.
   for this example we use owner id: user7; user7@something.com

```
$gpg --gen-key

gpg: directory `/home/user7/.gnupg' created
gpg: new configuration file `/home/user7/.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/user7/.gnupg/gpg.conf' are not yet
active during this run
gpg: keyring `/home/user7/.gnupg/secring.gpg' created
gpg: keyring `/home/user7/.gnupg/pubring.gpg' created
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? 1
```

3. Select your key size. (at least 2048)

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

4. Set the lifetime of the key.
   ex: 1 year (1y)

```
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) 1y
```

5. Fill your name and email address. Put your comment too. Then, don't forget to choose a password/pass phrase.

```
You need a user ID to identify your key; the software constructs
the user ID from the Real Name, Comment and Email Address in
this form:
         "Heinrich Heine (Der Dichter) <heinrich@something.de>"

Real name: user7
Email address: user7@something.com
Comment: first account
You selected this USER-ID:
         "user7 (first account) <user7@something.com>"

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

**You can see your key id after the process done. it will be "pub XXXXX/<key id>"**
**You can modify your key too. ex: add multiple email to your key.**

$ gpg —list-secret-keys

6. Send your public key to the PGP key server. (key server ex: keys.gnupg.net, pgp.mit.edu, etc.)

$ gpg —keyserver <keyserver> —send-keys <your key ID>

```
$gpg --keyserver keys.gnupg.net --send-keys ABC123

gpg: sending key ABCD1234 to hkp server keys.gnupg.net
```

7. Go to an RSVP page
   for example : https://linux.ucla.edu/keysigning/
   Check your key fingerprint

```
$ gpg —fingerprint <your name>
```

```
$gpg ——fingerprint user7

pub       XXXXX/ABCD1234 2017-01-03 [expires: 2018-01-03]
          Key fingerprint = 9086 29C6 EBEF 89ED B057 6EA6 174F 3433 E482 0DDE
uid                 user7 (first account) <user7@something.com>
sub       XXXXX/ABCD1234 2017-01-03 [expires: 2018-01-03]
```

8. The preparation is to finalize the keylist and checksum. Receive the keylist (you may printout your own keylist)

   This instruction is may be done or not depends on the host.

```
$ wget <keyserver>
```

## PREPARATION

**Prepare the following information (key ID, key type, key size, key fingerprint)**
**You must bring your identification! (ex: your photo, card id, etc)**

User7 < name: user7,
key ID: ABCD1234,
key type: RSA and RSA,
key fingerprint: 9086 29C6 EBEF 89ED 6EA6 B057 174F 3433 E482 0DDE >

**Don't forget to bring a pen and a downloaded participants keylist printed paper.**

## AT THE PARTY

**The host of the party will read each of the keyID and the owner id, you must to make sure that your key ID and your fingerprint is correct!**

Meet other participants and checkmark the keylist if you believe they are the true owner of a website
Other participants will also meet you, all you need to do is to show them your identification.

An example :

| Key ID | Owner | Fingerprint | Key info | Owner ID |
|---|---|---|---|---|
| 84CA3E00 | user8 <user8@something.com) | AAA1 2B22 CC33 444D 555E F6F6 7G7G 8888 22CC 1S11 | ✓ | ✓ |
| 6C04E700 | user9 <user9@something.com) | HJH1 2D22 BB77 444D 424E F6F6 7X7X 4545 22BB 1S77 | ✓ | ✓ |
| E4B10D00 | user10 <user10@something.com) | ZZZ1 2L22 PP77 444B 999E F6F6 9G7K 3333 22CC 1K11 | ✓ | ✓ |
| … | … | … | … | … |

## AFTER THE PARTY

**Don't forget the most important thing!!**
**Retrieved your noted keylist and make sure to sign the participants key.**

1. Check your noted keylist.
   Import the key of person you believe the true owner of a website.

```
$ gpg —recv-keys <key ID 1> <key ID > … <key ID N>
```

```
$gpg –recv-keys 6C04E700 E4B10D00

gpg: requesting key 6C04E700 from hkp server keys.gnupg.net
gpg: requesting key E4B10D00 from hkp server keys.gnupg.net
gpg: key 6C04E700: public key "user9 (third account) <user9@something.com> imported
gpg: key E4B10D00: public key "user10 (fourth account) <user9@something.com> imported
gpg: Total number processed: 2
gpg:               imported: 2   (RSA: 2)
```

2. Sign the keys

```
$ gpg —sign-key <key owner>
```

```
$gpg –sign-key user9

pub       XXXXX/6C04E700      created: 2017-01-03           expires: 2018-01-03    usage: SC
                              trust: unknown                validity: unknown
sub       XXXXX/WXYZ8888      created:2017-01-03            expires: 2018-01-03    usage: E
[unknown] (1). user9 (third account) <user9@something.com>

pub       XXXXX/6C04E700      created: 2017-01-03           expires: 2018-01-03    usage: SC
                              trust: unknown                validity: unknown
  Primary key fingerprint: AAA1 2B22 CC33 444D 555E  F6F6 7G7G 8888 22CC 1S11


          user9 (third account) <user9@something.com>

This key is due to expire on 2018-01-03
Are you sure that you want to sign this key with your key "user7 (first account)
<user7@something.com>? (ABCD1234)


Really sign? (y/N) y

…
```

3. Send all the new key signatures to the keyserver

```
$ gpg —send-keys <key ID 1> <key ID > … <key ID N>
```

```
$gpg –send-keys 6C04E700 E4B10D00

gpg: sending key 6C04E700 to hkp server keys.gnupg.net
gpg: sending key E4B10D00 to hkp server keys.gnupg.net
```

== EOF ==