# How to Implement Key Signing Party

**Keyne Kassapa** University of Indonesia

January 2016

# **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

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

- 2. Select your key size. (at least 2048)

DSA keypair will have 2048 bits. About to generate a new ELG-E keypair. minimum keysize is 768 bits default keysize is 1024 bits highest suggested keysize is 2048 bits What keysize do you want? (1024) 2048<return> Do you really need such a large keysize? yes<return>

3. Set the lifetime of the key. ex: 1 year

Key is valid for? (0) 1y<return>
Key expires at Sun Jan 1 00:00:15 2015 EDT
Is this correct (y/n)? y<return>

4. Fill your name and email address. Put your comment too. Then choose a password/pass phrase.

Real name: Keyne Kassapa Email address: keyne.kassapa@ui.ac.id Comment: You selected this USER-ID: "Keyne Kassapa <keyne.kassapa@ui.ac.id>"

5. You can modify your key. You can add multiple email to your key.

\$ gpg —list-secret-keys

6. Send your public key with the PGP key server

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

7. Go to an RSVP page

for example : https://linux.ucla.edu/keysigning/ Check your key fingerprint

\$ gpg —fingerprint <your name>

8. The preparation is to finalize the keylist and checksum.

\$ wget <keyserver>

9. Verify the checksum

\$ sha1sum —check keylist.txt.sha1

## The Party

- 1. Bring the following information (key ID, key type, key size, key fingerprint) and your identification
- 2. Receive the keylist (or you may printout your own keylist)
- 3. Make statement that your fingerprint is correct
- 4. Other participant will meet you and check your identification (picture and names usually). It also works vice versa with you checking other participant.

# After the party

1. Check your noted keylist.

Import the key of person you believe the true owner of a website.

2. Sign the keys

3. Send all the new key signatures to the keyserver

#### **EXAMPLES**

[3 dummy users; User7, User8, User9]

## **BEFORE THE PARTY**

#### User7, User8, User9 generate its own keypair. In this example we will use server ckilat1.vlsm.org

- 1. Generate your PGP keypair / gnupg keys
- 2. Select your key size. (at least 2048
- 3. Set the lifetime of the key. ex: 1 year







4. Fill your name and email address. Put your comment too. Then choose a password/pass phrase.

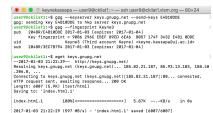




# Select the keyserver and sync your public key with the keyserver In this example we will use <u>keys.gnupg.org</u>

- 5. You can modify your key. You can add multiple email to your key.
- 6. Sync your public key with the PGP key server
- 7. Check your fingerprint





#### PREPARATION:

Prepare the following information (key ID, key type, key size, key fingerprint) and your identification (your photo)

User7 < name: Keyne1, key ID: 84CA3E57, key type: RSA and RSA, key fingerprint: FCB7 4E58 FC39 4FC5 750E F2F4 4FFC 9AFC 84CA 3E57>

User8 < name: Keyne2, key ID: 6C04E7D5 , key type: RSA and RSA, key fingerprint: 5409 04A8 3E68 5CFC 9A88 4BAF 3B4D 1635 6C04 E7D5>

User9 < name: Keyne3, key ID: E4B10DDE, key type: RSA and RSA, key fingerprint: 9086 296C EBEF 89ED 6EA6 B057 174F 3432 E4B1 0DDE>

Bring a pen and a keylist of other participant.

#### **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 is correct!!

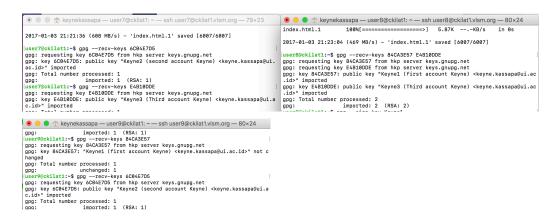
Meet other participants and checkmark the keylist if you believe they are the true owner of a website

Key ID	Owner	Fingerprint	Key info	Owner ID /
84CA3E57	Keyne1 < keyne.kassapa@ui.ac.id)	FCB7 4E58 FC39 4FC5 750E F2F4 4FFC 9AFC 84CA 3E57		
6C04E7D5	Keyne2 < <u>keyne.kassapa@ui.ac.id</u> )	5409 04A8 3E68 5CFC 9A88 4BAF 3B4D 1635 6C04 E7D5		
E4B10DDE	Keyne3 <keyne.kassapa@ui.ac.id)< td=""><td>9086 296C EBEF 89ED 6EA6 B057 174F 3432 E4B1 0DDE</td><td></td><td></td></keyne.kassapa@ui.ac.id)<>	9086 296C EBEF 89ED 6EA6 B057 174F 3432 E4B1 0DDE		

# AFTER THE PARTY

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

8. Import the key of person you believe the true owner of a website.



#### 9. Sign the keys



#### 10. Send all the new keys to the keyserver

```
| user7@ckilat1:~$ gpg --send-keys E4B10DDE | gpg: sending key E4B10DDE to hkp server keys.gnupg.net | user7@ckilat1:~$ gpg --send-keys 6C04E7D5 | user7@ckilat1:~$ gpg --send-keys 6C04E7D5 | gpg: sending key 6C04E7D5 to hkp server keys.gnupg.net user7@ckilat1:~$ | user7@ckilat1:~$ | user8@ckilat1:~$ | user8@ckilat1:
```

[user9@ckilat1:~\$ gpg --send-keys 84CA3E57
gpg: sending key 84CA3E57 to hkp server keys.gnupg.net
[user9@ckilat1:~\$ gpg --send-keys 6C04E7D5
gpg: sending key 6C04E7D5 to hkp server keys.gnupg.net