

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.

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

2. Sign the keys

```
$ gpg --sign-key <key ID 1> <key ID > ... <key ID N>
```

3. Send all the new key signatures to the keyserver

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

EXAMPLES

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

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

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

server = ckilat1.vlsm.org
keyserver = keys.gnupg.key

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

```
keynekassapa — user7@ckilat1: ~ — ssh user7@ckilat1.vlsm.org — 79x23
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
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
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
    <nw> = key expires in n weeks
    <nm> = key expires in n months
    <ny> = key expires in n years
Key is valid for? (0) 1
```

```
keynekassapa — user8@ckilat1: ~ — ssh user8@ckilat1.vlsm.org — 80x24
gpg: directory '/home/user8/.gnupg' created
gpg: new configuration file '/home/user8/.gnupg/gpg.conf' created
gpg: WARNING: options in '/home/user8/.gnupg/gpg.conf' are not yet active during
this run
gpg: keyring '/home/user8/.gnupg/secring.gpg' created
gpg: keyring '/home/user8/.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
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
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
    <nw> = key expires in n weeks
    <nm> = key expires in n months
    <ny> = key expires in n years
Key is valid for? (0) 1
Key expires at Wed Jan  4 21:05:38 2017 WIB
Is this correct? (y/N) y
```

```
keynekassapa — user9@ckilat1: ~ — ssh user9@ckilat1.vlsm.org — 80x24
There is NO WARRANTY, to the extent permitted by law.
gpg: directory '/home/user9/.gnupg' created
gpg: new configuration file '/home/user9/.gnupg/gpg.conf' created
gpg: WARNING: options in '/home/user9/.gnupg/gpg.conf' are not yet active during
this run
gpg: keyring '/home/user9/.gnupg/secring.gpg' created
gpg: keyring '/home/user9/.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
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
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
    <nw> = key expires in n weeks
    <nm> = key expires in n months
    <ny> = key expires in n years
Key is valid for? (0) 1
```

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

```
keynekassapa — user7@ckilat1: ~ — ssh user7@ckilat1.vlsm.org — 79x23
Is this correct? (y/N) y
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@duesseldorf.de>"
Real name: Keyne1
Email address: keyne.kassapa@ui.ac.id
Comment: first account Keyne
You selected this USER-ID:
    "Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>"
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.
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.
.....++++
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
```

```
keynekassapa — user8@ckilat1: ~ — ssh user8@ckilat1.vlsm.org — 80x24
Key expires at Wed Jan  4 21:05:38 2017 WIB
Is this correct? (y/N) y
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@duesseldorf.de>"
Real name: Keyne2
Email address: keyne.kassapa@ui.ac.id
Comment: second account Keyne
You selected this USER-ID:
    "Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>"
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.
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.
.....++++
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
```

```
keynekassapa — user9@ckilat1: ~ — ssh user9@ckilat1.vlsm.org — 80x24
Key expires at Wed Jan  4 21:06:57 2017 WIB
Is this correct? (y/N) y
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@duesseldorf.de>"
Real name: Keyne3
Email address: keyne.kassapa@ui.ac.id
Comment: third account Keyne
You selected this USER-ID:
    "Keyne3 (third account Keyne) <keyne.kassapa@ui.ac.id>"
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.
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.
.....++++
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
```

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

```
keynekassapa — user7@ckilat1: ~ — ssh user7@ckilat1.vlsm.org — 79x23
user7@ckilat1:~$ gpg --keyserver keys.gnupg.net --send-keys 84CA3E57
gpg: sending key 84CA3E57 to hkp server keys.gnupg.net
user7@ckilat1:~$ gpg --fingerprint Keyne1
pub 2048R/84CA3E57 2017-01-03 (expires: 2017-01-04)
Key fingerprint = FCB7 4E58 FC39 4FC5 750E F2F4 4FFC 9AFC 84CA 3E57
uid Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>
sub 2048R/72C2CE9 2017-01-03 (expires: 2017-01-04)
user7@ckilat1:~$ wget keys.gnupg.net
--2017-01-03 21:21:36-- http://keys.gnupg.net/
Resolving keys.gnupg.net (keys.gnupg.net)... 185.82.21.187, 85.93.13.183, 188.4
0.286.8, ...
Connecting to keys.gnupg.net (keys.gnupg.net)[185.82.21.187]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6807 (5.9K) [text/html]
Saving to: 'index.html.1'
index.html.1 100%[=====] 5.87K --.-KB/s in 0s
2017-01-03 21:21:36 (680 MB/s) - 'index.html.1' saved [6807/6807]
```

```
keynekassapa — user8@ckilat1: ~ — ssh user8@ckilat1.vlsm.org — 80x24
user8@ckilat1:~$ gpg --keyserver keys.gnupg.net --send-keys 5185C3E6
gpg: sending key 6C04E7D5 to hkp server keys.gnupg.net
user8@ckilat1:~$ gpg --fingerprint Keyne2
pub 2048R/6C04E7D5 2017-01-03 (expires: 2017-01-04)
Key fingerprint = 5409 04A8 3E68 5CFC 9A88 4BAF 3B4D 1635 6C04 E7D5
uid Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>
sub 2048R/5185C3E6 2017-01-03 (expires: 2017-01-04)
user8@ckilat1:~$ wget keys.gnupg.net
--2017-01-03 21:23:04-- http://keys.gnupg.net/
Resolving keys.gnupg.net (keys.gnupg.net)... 185.82.21.187, 85.93.13.183, 188.4
0.286.8, ...
Connecting to keys.gnupg.net (keys.gnupg.net)[185.82.21.187]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6807 (5.9K) [text/html]
Saving to: 'index.html.1'
index.html.1 100%[=====] 5.87K --.-KB/s in 0s
2017-01-03 21:23:04 (469 MB/s) - 'index.html.1' saved [6807/6807]
```

```
keynekassapa — user9@ckilat1: ~ — ssh user9@ckilat1.vlsm.org — 80x24
user9@ckilat1:~$ gpg --keyserver keys.gnupg.net --send-keys E4B10DDE
gpg: sending key E4B10DDE to hkp server keys.gnupg.net
user9@ckilat1:~$ gpg --fingerprint Keyne3
pub 2048R/E4B10DDE 2017-01-03 (expires: 2017-01-04)
Key fingerprint = 9086 296C EBEF 89ED 6EA6 B057 174F 3432 E4B1 0DDE
uid Keyne3 (third account Keyne) <keyne.kassapa@ui.ac.id>
sub 2048R/3EC7B794 2017-01-03 (expires: 2017-01-04)
user9@ckilat1:~$ wget keys.gnupg.net
--2017-01-03 21:22:29-- http://keys.gnupg.net/
Resolving keys.gnupg.net (keys.gnupg.net)... 185.82.21.187, 85.93.13.183, 188.4
0.286.8, ...
Connecting to keys.gnupg.net (keys.gnupg.net)[185.82.21.187]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6807 (5.9K) [text/html]
Saving to: 'index.html.1'
index.html.1 100%[=====] 5.87K --.-KB/s in 0s
2017-01-03 21:22:29 (597 MB/s) - 'index.html.1' saved [6807/6807]
```

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

```
keynekassapa — user7@ckilat1: ~ — ssh user7@ckilat1.vism.org — 79x23
2017-01-03 21:21:36 (608 MB/s) - 'index.html.1' saved [6007/6007]
user7@ckilat1:~$ gpg --recv-keys 6C04E7D5
gpg: requesting key 6C04E7D5 from hkp server keys.gnupg.net
gpg: key 6C04E7D5: public key "Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>" imported
gpg: Total number processed: 1
gpg: imported: 1 (RSA: 1)
user7@ckilat1:~$ gpg --recv-keys E4B10DDE
gpg: requesting key E4B10DDE from hkp server keys.gnupg.net
gpg: key E4B10DDE: public key "Keyne3 (Third account Keyne) <keyne.kassapa@ui.ac.id>" imported
gpg: Total number processed: 2
gpg: imported: 2 (RSA: 2)

keynekassapa — user8@ckilat1: ~ — ssh user8@ckilat1.vism.org — 80x24
index.html.1 100%[=====] 5.87K --.-KB/s in 0s
2017-01-03 21:23:04 (469 MB/s) - 'index.html.1' saved [6007/6007]
user8@ckilat1:~$ gpg --recv-keys 84CA3E57 E4B10DDE
gpg: requesting key 84CA3E57 from hkp server keys.gnupg.net
gpg: requesting key E4B10DDE from hkp server keys.gnupg.net
gpg: key 84CA3E57: public key "Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>" imported
gpg: key E4B10DDE: public key "Keyne3 (Third account Keyne) <keyne.kassapa@ui.ac.id>" imported
gpg: Total number processed: 2
gpg: imported: 2 (RSA: 2)

keynekassapa — user9@ckilat1: ~ — ssh user9@ckilat1.vism.org — 80x24
gpg: imported: 1 (RSA: 1)
user9@ckilat1:~$ gpg --recv-keys 84CA3E57
gpg: requesting key 84CA3E57 from hkp server keys.gnupg.net
gpg: key 84CA3E57: "Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>" not changed
gpg: Total number processed: 1
gpg: unchanged: 1
user9@ckilat1:~$ gpg --recv-keys 6C04E7D5
gpg: requesting key 6C04E7D5 from hkp server keys.gnupg.net
gpg: key 6C04E7D5: public key "Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>" imported
gpg: Total number processed: 1
gpg: imported: 1 (RSA: 1)
```

9. Sign the keys

```
keynekassapa — user7@ckilat1: ~ — ssh user7@ckilat1.vism.org — 79x23
usage: gpg [options] --sign-key user-id
user7@ckilat1:~$ gpg --sign-key Keyne2
pub 2048R/6C04E7D5 created: 2017-01-03 expires: 2017-01-04 usage: SC
trust: unknown validity: unknown
sub 2048R/5105C3E6 created: 2017-01-03 expires: 2017-01-04 usage: E
[ unknown ] (1). Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>

pub 2048R/6C04E7D5 created: 2017-01-03 expires: 2017-01-04 usage: SC
trust: unknown validity: unknown
Primary key fingerprint: 5489 84A8 3E68 6CFC 9A8B 4BAF 384D 1635 6C04 E7D5

Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>
This key is due to expire on 2017-01-04.
Are you sure that you want to sign this key with your
key "Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>" (84CA3E57)
Really sign? (y/N) y
You need a passphrase to unlock the secret key for
user: "Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>"

keynekassapa — user8@ckilat1: ~ — ssh user8@ckilat1.vism.org — 80x24
gpg: imported: 2 (RSA: 2)
user8@ckilat1:~$ gpg --sign-key Keyne1
pub 2048R/84CA3E57 created: 2017-01-03 expires: 2017-01-04 usage: SC
trust: unknown validity: unknown
sub 2048R/7C2EDCE9 created: 2017-01-03 expires: 2017-01-04 usage: E
[ unknown ] (1). Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>

pub 2048R/84CA3E57 created: 2017-01-03 expires: 2017-01-04 usage: SC
trust: unknown validity: unknown
Primary key fingerprint: FC07 4E5B FC39 4FC5 780E F2FA 4FFC 9AFC 84CA 3E57

Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>
This key is due to expire on 2017-01-04.
Are you sure that you want to sign this key with your
key "Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>" (6C04E7D5)
Really sign? (y/N) y
You need a passphrase to unlock the secret key for
user: "Keyne2 (second account Keyne) <keyne.kassapa@ui.ac.id>"
2048-bit RSA key, ID 6C04E7D5, created 2017-01-03

keynekassapa — user9@ckilat1: ~ — ssh user9@ckilat1.vism.org — 80x24
usage: gpg [options] --sign-key user-id
user9@ckilat1:~$ gpg --sign-key Keyne1
pub 2048R/84CA3E57 created: 2017-01-03 expires: 2017-01-04 usage: SC
trust: unknown validity: unknown
sub 2048R/7C2EDCE9 created: 2017-01-03 expires: 2017-01-04 usage: E
[ unknown ] (1). Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>

pub 2048R/84CA3E57 created: 2017-01-03 expires: 2017-01-04 usage: SC
trust: unknown validity: unknown
Primary key fingerprint: FC07 4E5B FC39 4FC5 780E F2FA 4FFC 9AFC 84CA 3E57

Keyne1 (first account Keyne) <keyne.kassapa@ui.ac.id>
This key is due to expire on 2017-01-04.
Are you sure that you want to sign this key with your
key "Keyne3 (Third account Keyne) <keyne.kassapa@ui.ac.id>" (E4B10DDE)
Really sign? (y/N) y
You need a passphrase to unlock the secret key for
user: "Keyne3 (Third account Keyne) <keyne.kassapa@ui.ac.id>"
```

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
gpg: sending key 6C04E7D5 to hkp server keys.gnupg.net
user7@ckilat1:~$

user8@ckilat1:~$ gpg --send-keys E4B10DDE 84CA3E57
gpg: sending key E4B10DDE to hkp server keys.gnupg.net
gpg: sending key 84CA3E57 to hkp server keys.gnupg.net
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
user9@ckilat1:~$
```