# Getting a GPG key



## Generating a key

First of all you will need to log onto a Linux machine e.g. noric, then generate the pair,see below. (After entropy. I hit several spaces and returns to help entropy, however 30 mins later it had not appeared to finish), then it said it was generating the key again, I hit lots more characters and a few minutes later it finished (see below).

```
[cottrell@pinger ~]$ gpg --gen-key
gpg (GnuPG) 2.0.14; 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.

gpg: keyring `/u/sf/cottrell/.gnupg/secring.gpg' created
gpg: keyring `/u/sf/cottrell/.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?
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (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
      <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)
Key does not expire at all
Is this correct? (y/N) y

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

Real name: Roger Cottrell
Email address: rlacottrell@gmail.com
Comment: Les
You selected this USER-ID:
    "Roger Cottrell (Les) <rlacottrell@gmail.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

can't connect to `/u/sf/cottrell/.gnupg/S.gpg-agent': No such file or
directory
gpg-agent[6645]: directory `/u/sf/cottrell/.gnupg/private-keys-v1.d'
created
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
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[C^[[B
```

```
         gjhgjhfhfuiuiouomnmmnnbb /u/sf/cottrell/.gnupg/trustdb.gpg:
trustdb created
gpg: key 271CF0E9 marked as ultimately trusted
public and secret key created and signed.

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   2048R/271CF0E9 2013-09-11
      Key fingerprint = 0B4F EC8A D1D0 568A 654C  BD99 B058 14A9 271C F0E9
uid                  Roger Cottrell (Les) <rlacottrell@gmail.com>
sub   2048R/87C7DB76 2013-09-11

[cottrell@pinger ~]$  [cottrell@pinger ~]$ ls -la .gnupg/
total 62
drwx------   3 cottrell sf  2048 Sep 10 22:35 ./
drwxr-xr-x 108 cottrell sf 49152 Sep 10 21:50 ../
drwx------   2 cottrell sf  2048 Sep 10 22:06 private-keys-v1.d/
-rw-------   1 cottrell sf  1203 Sep 10 22:35 pubring.gpg
-rw-------   1 cottrell sf  1203 Sep 10 22:35 pubring.gpg~
-rw-------   1 cottrell sf   600 Sep 10 22:35 random_seed
-rw-------   1 cottrell sf  2581 Sep 10 22:35 secring.gpg
-rw-------   1 cottrell sf  1280 Sep 10 22:35 trustdb.gpg
```

## Set up privileges for .gnupg directory

Initially after generating the key, you will get a .gnupg directory with something like:

```
rajaasad@noric37 $ fs la .gnupg
Access list for .gnupg is
Normal rights:
  system:slac rl
  system:administrators rlidwka
  system:authuser rl
  rajaasad rlidwka
```

But you need to get rid of the first and the third entry by executing:

```
fs sa system:slac none system:authuser none
```

Now you will have the following privileges:

```
rajaasad@noric37 $ fs la .gnupg
Access list for .gnupg is
Normal rights:
  system:administrators rlidwka
  rajaasad rlidwka
```

## Exporting the key

```
208cottrell@pinger:~$gpg --export -a "Roger Cottrell" > public_key
209cottrell@pinger:~$cat public_key
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.14 (GNU/Linux)

mQENBFIv+r4BCADn5msooKvPE3S2fuOgB1wCvbqh4JySYrVWJrADn58miEn6a593
6JbnUv22CJsks1EOtPVmgEFIxZiNxYChMLMYJ3/fQDhoTKeITUrcmXVHRQw4zu21
3E11TASuBRKXT4DOj10LgTTDGEcCpwf/LI1+Z1STNm1V976mBK+e/i4L0NVre+u2
vDlpiML8NumNShL6O1JVPDY6ULlooUHgGS4v/cFvn1Z8xrfwRdJIar8vyR+qZcOm
4I2ZGOklKXUILtpvhsq99JASJR9BEz4oec+UPbGv5Ux6GrPt6VF0tEGDGAGlqvMf
gxFrJE1R/HC9AwdjLIayZQEp+Kz8l9QcsY7JABEBAAG0LFJvZ2VyIENvdHRyZWxs
IChMZXMpIDxybGFjb3R0cmVsbEBnbWFpbC5jb20+iQE4BBMBAgAiBQJSL/q+AhsD
BgsJCAcDAgYVCAIJCgsEFgIDAQIeAQIXgAAKCRCwWBSpJxzw6S1vCADf7cucsnDR
rvdtLZ+GpQlEdqvQ3rW6D6fMShzay0DzLkBpLzNr56Rkxx5ngziWevmGYuBZ7ktT
zGGicbhPtU2Zf3X1KfuJC2fOQGcWlUGApxhU39EXgvXN05LfTAFyv/wZaGSMfmVe
erHVbjm2MhHd56NSp2XW2gRRu/NPLeX3G+2qch6XfaK4K4ZashWIe07IxB9G6auc
RUkSMT/JzPSEdnNbX6/5lMQlSet5YYmUM9fxN4ToQwK1+DtuSqNXm+nqLJHE2K5c
2jt+OFE5+HVpHN7ICGI4YFOHtYp+0NogSFUkpflUWdNDlCl9r5T7F1bx9wj44ovK
Gdau4OXSXywYuQENBFIv+r4BCACaGbe4j2G1EyHCLCEmZX+s5BE8oEHFrwjcYwfG
ks3y6fZG+5hrxsWMBs1ZsQip4XlJbywzrI+XDuopxVpxKJDJTGSh48bDw0NggKCT
unuVwNB6pK22PQqu0JitvNCWaIyi3OwGd5RroeHYq59AHzAgL+N2mfuDGF33Thbk
fgR13owzau7yQPPfvyhzpnZ4Bk/k1qRDeklA5YTANn9iI5V/uqfBQMt6gOB+A7JE
yfD7GtaaOwqR9RVvtihCt5U2Mi7s++PgRBTzMdPxQG6DUei5MS+Xfz9Lu2BW69Hj
cdQANMsLY7hpY+LozHpr7NiO4IPqYFtR5XRYbCrgT9T4/4ONABEBAAGJAR8EGAEC
AAkFAlIv+r4CGwwACgkQsFgUqScc8On4IAgAkhsGXjsP+s/SJv+VM0Tljm6e6brO
DZ0l/YlhotuMTegwDW4P2EdeNi3zML9n9CB9ZkomjrSrotuJ+VwNA2hTXndnuDjf
7ppXGt/HIHh00nS5olgkqi3jWc6xGsYXJmldK+l7UDXeCvYv6yM8mObxCA72FvER
fBbNxLfnreLOCvbgjYrOM2nXY03sXDcXlBgaMmUM242lnYAklnBRn5LesibzY4N3
DXh5QYYLqENMUvzXwaEpH3Uhl+EuMnyUA3pdMoIY9lTn2WMUVmZmNXaS02XetVRm
k9EwkXpVyxHSI+9PSsqrWYwaXKk4jUTfek/RyVTp0ChVZSScweruYklzBg==
=ryhi
-----END PGP PUBLIC KEY BLOCK-----
```

## Make the key publicly available

We use ftp to do this so the key will show up at:

```
ftp://ftp.slac.stanford.edu/pgp/cottrell/cottrell.publickey
```

This is done by moving the public_key generated above to the ftp space;

```
217cottrell@pinger:$mv ~cottrell/public_key /afs/slac.stanford.edu/public/pgp/cottrell/cottrell.publickey
```

## Adding someone else's public key to your keyring

Get the other person's public key (they may email it to you or put it in a publicly accessible place). Assuming you have put it in your home directory at public.key, then

```
gpg --import public.key
```

You will then need to sign the key. Before you do this you may want a list of keys in your keyring:

```
113cottrell@pinger:~$gpg --list-keys
/u/sf/cottrell/.gnupg/pubring.gpg
---------------------------------
pub   2048R/271CF0E9 2013-09-11
uid                  Roger Cottrell (Les) <rlacottrell@gmail.com>
sub   2048R/87C7DB76 2013-09-11

pub   2048R/0C0D6DCB 2013-09-08
uid                  Martin Emmerson <emmerson@telemage.com>
sub   2048R/9993A460 2013-09-08
```

To sign the key you can proceed as follows:

```
114cottrell@pinger:~$gpg --sign-key 'Martin Emmerson'

pub  2048R/0C0D6DCB  created: 2013-09-08  expires: never       usage: SC
                     trust: unknown       validity: unknown
sub  2048R/9993A460  created: 2013-09-08  expires: never       usage: E
[ unknown] (1). Martin Emmerson <emmerson@telemage.com>


pub  2048R/0C0D6DCB  created: 2013-09-08  expires: never       usage: SC
                     trust: unknown       validity: unknown
 Primary key fingerprint: F61E 0D6F F7E8 16E1 C776  01C0 8470 9956 0C0D 6DCB

     Martin Emmerson <emmerson@telemage.com>

Are you sure that you want to sign this key with your
key "Roger Cottrell (Les) <rlacottrell@gmail.com>" (271CF0E9)

Really sign? (y/N) y

You need a passphrase to unlock the secret key for
user: "Roger Cottrell (Les) <rlacottrell@gmail.com>"
2048-bit RSA key, ID 271CF0E9, created 2013-09-11

can't connect to `/u/sf/cottrell/.gnupg/S.gpg-agent': No such file or directory
```

I don't think you need to worry about the "can't connect to `/u/sf/cottrell/.gnupg/S.gpg-agent': No such file or directory" it is probably out of sync STDERR outout from when it found out you needed to sign the key.

The --sign-key option is a short cut for --edit-key found in http://www.gnupg.org/gph/en/manual.html.

## Reading an encrypted message from somebody

Assume you got the message (e.g. as an email enclosure) and you copied it to your home directory as:

```
203cottrell@wanmon:~$gpg -d msg-from-martin.asc >! junk

You need a passphrase to unlock the secret key for
user: "Roger Cottrell (Les) <rlacottrell@gmail.com>"
2048-bit RSA key, ID 87C7DB76, created 2013-09-11 (main key ID 271CF0E9)

gpg: encrypted with 2048-bit RSA key, ID 87C7DB76, created 2013-09-11
      "Roger Cottrell (Les) <rlacottrell@gmail.com>"
gpg: Signature made Thu 12 Sep 2013 10:28:38 AM PDT using RSA key ID 0C0D6DCB
gpg: Good signature from "Martin Emmerson <emmerson@telemage.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F61E 0D6F F7E8 16E1 C776  01C0 8470 9956 0C0D 6DCB 113
```

The decrypted output is written to STDOUT, here redirected to junk.

The WARNING (if it appears) means you have not signed the key yet (see above for how to sign).

## Documentation

See http://www.gnupg.org/documentation/guides.en.html