GPG Key Generation HowTo

Step By Step

Generate a key

```
gpg --full-generate-key
```

1. You want an RSA and RSA (default)
2. Key size is 4096
3. Key expires (Up to you when you want your key to expire)
4. Real Name: You real name
5. Email Address: Your@email.address
6. Comment: Add a comment about your key
7. **You dont need a passphrase**

Send the Key

```
gpg --send-keys E407B7AB
```

Walkthrough

This is a quick step-by-step guide to generating a GPG Key that can be used to sign FreePBX Modules.

```
[root@localhost devtools]# 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.
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)
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) 1y
Key expires at Tue 17 Nov 2015 03:43:38 PM AEST
Is this correct? (y/N) y
```

```
GnuPG needs to construct a user ID to identify your key.
Real name: Ed Led Ovum
Email address: edled@ovum.com
Comment: Ovum.com FreePBX Module Signing Key
You selected this USER-ID:
    "Ed Led Ovum (Ovum.com FreePBX Module Signing Key) <edled@ovum.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 `/root/.gnupg/S.gpg-agent': No such file or directory
You don't want a passphrase - this is probably a *bad* idea!
I will do it anyway.  You can change your passphrase at any time,
using this program with the option "--edit-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
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key E407B7AB 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:   4  signed:   1  trust: 0-, 0q, 0n, 0m, 0f, 4u
gpg: depth: 1  valid:   1  signed:   0  trust: 0-, 0q, 0n, 1m, 0f, 0u
gpg: next trustdb check due at 2015-11-17
pub   2048R/E407B7AB 2014-11-17 [expires: 2015-11-17]
      Key fingerprint = 7054 A588 0D84 AFF9 D449  626B 9503 530A E407 B7AB
uid                  Ed Led Ovum (Ovum.com FreePBX Module Signing Key)
<edled@ovum.com>
sub   2048R/19663421 2014-11-17 [expires: 2015-11-17]
[root@localhost devtools]#
```

If the machine freezes after saying 'We need to generate a lot of random bytes', this is often because the machine does not have enough entropy to generate a secure random number. This is extremely common on Virtual Machines. There are several workarounds for this, but the best thing to do is to generate your key on raw hardware, with a reasonably recently Intel or AMD CPU that has hardware random number generation.

Please feel free to ask in IRC for further assistance if you're stuck.

The 'pub' line is your new Key. Whilst you can use that for other things, it's probably best to keep it only for signing FreePBX Modules. Note that if this key is ever detected as compromised, it will be revoked (by us) and all modules that key has signed will be unusable. Take care of it, and back it up!

After you've generated the key, and you're happy with it, you then need to publish it to the global Web Of Trust network. This is done simply with the following command:

```
[root@localhost devtools]# gpg --send-keys E407B7AB
gpg: sending key E407B7AB to hkp server keys.gnupg.net
[root@localhost devtools]#
```

You don't need to specifically send it to any one keyserver, as they transparently replicate changes amongst themselves. However, if you're unable to reach that cluster for some reason, Ubuntu offers a keyserver on port 80, and you can explicitly send your key to that host with the following command:

```
[root@localhost devtools]# gpg --keyserver hkp://keyserver.ubuntu.com:80 --
send-keys E407B7AB
gpg: sending key E407B7AB to hkp server keyserver.ubuntu.com
[root@localhost devtools]#
```

As above, you only need to export your key to ONE keyserver, and they will share it amongst themselves.

It's a good idea to keep note of the initial server you sent the key to when sending a request for FreePBX signing otherwise you will have to wait up to 24 hours for keys to replicate.

Now that key is out in the wild, you can request it be Signed by FreePBX or use it locally!.