James Oguya (https://oguya.ch/)

Linux nuggets

GPG Subkeys

Posted on Apr 1, 2016

#gpg (https://oguya.ch//tags/gpg) #openpgp (https://oguya.ch//tags/openpgp) #linux (https://oguya.ch//tags/linux)

OpenPGP supports subkeys which are like the normal keys, except they're bound to a master key pair. A subkey can be used for signing or for encryption. The really useful part of subkeys is that they can be revoked independently of the master keys, and also stored separately from them.

When using subkeys, you'll only use the master keypair under the following circumstances:

- creating a new subkey
- changing the preferences on a UID
- revoking an existing UID or subkey
- signing a key or revoking an existing signature
- creating a new UID or marking an existing UID as primary
- changing the expiration date on a master key or any of its subkeys

The procedure for creating GPG subkey is as simple as follows:

1. Create a regular GPG keypair. By default GPG creates one signing subkey(your identity) and one encryption subkey.
2. Use `gpg` to add an additional signing subkey to your keypair. This new subkey is linked to the first signing key. So we have three subkeys.
3. Store your master keypair in a safe place, for its loss will be catastrophic.
4. Use `gpg` to remove the original signing subkey, leaving on the new signing subkey & the encryption subkey.

Create a regular GPG Keypair

- Use `gpg2 --gen-key` command to create a new GPG keypair. It's always a good idea to set your key to expire within a year or less and use 4096 key length instead of the default 2048:

```
$ gpg2 --gen-key
gpg (GnuPG) 2.0.29; Copyright (C) 2015 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) 4096
Requested keysize is 4096 bits
...
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
gpg: next trustdb check due at 2016-12-10
pub   4096R/48CCEEDF 2015-12-11 [expires: 2016-12-10]
      Key fingerprint = EE73 1886 CA3D 89BD 34C1  F7E4 66B3 AC6C 48CC EEDF
uid       [ultimate] John Doe <jdoe@doeworks.ch>
sub   4096R/8E93D277 2015-12-11 [expires: 2016-12-10]
```

- We've successfully created a GPG keypair with the ID `48CCEEDF`:

```
$ gpg2 --list-keys 48CCEEDF
pub   4096R/48CCEEDF 2015-12-11 [expires: 2016-12-10]
uid       [ultimate] John Doe <jdoe@doeworks.ch>
sub   4096R/8E93D277 2015-12-11 [expires: 2016-12-10]
```

Create a Signing Subkey

- Using the `gpg2--edit-key` command, at the `gpg>` prompt, use the `addkey` command to create a subkey:

```
$ gpg2 --edit-key 48CCEEDF
...
pub  4096R/48CCEEDF  created: 2015-12-11  expires: 2016-12-10  usage: SC
                     trust: ultimate      validity: ultimate
sub  4096R/8E93D277  created: 2015-12-11  expires: 2016-12-10  usage: E
[ultimate] (1). John Doe <jdoe@doeworks.ch>

gpg> addkey
Key is protected.
...
Please select what kind of key you want:
   (3) DSA (sign only)
   (4) RSA (sign only)
   (5) Elgamal (encrypt only)
   (6) RSA (encrypt only)
Your selection? 4
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 4096
Requested keysize is 4096 bits
...
pub  4096R/48CCEEDF  created: 2015-12-11  expires: 2016-12-10  usage: SC
                     trust: ultimate      validity: ultimate
sub  4096R/8E93D277  created: 2015-12-11  expires: 2016-12-10  usage: E
sub  4096R/A85EA103  created: 2015-12-11  expires: 2016-12-10  usage: S
[ultimate] (1). John Doe <jdoe@doeworks.ch>

gpg> save
```

- This creates for us a new subkey with the key ID `A85EA103`.

Create a Revocation Certificate

- If your master keypair is lost or stolen, a revocation certificate will come in handy in telling people to ignore the stolen/lost key.

```
$ gpg2 --output 48CCEEDF-revocation-cert.asc --gen-revoke 48CCEEDF

sec  4096R/48CCEEDF 2015-12-11 John Doe <jdoe@doeworks.ch>

Create a revocation certificate for this key? (y/N) y
Please select the reason for the revocation:
  0 = No reason specified
  1 = Key has been compromised
  2 = Key is superseded
  3 = Key is no longer used
  Q = Cancel
(Probably you want to select 1 here)
Your decision? 1
Enter an optional description; end it with an empty line:
>
Reason for revocation: Key has been compromised
(No description given)
Is this okay? (y/N) y

You need a passphrase to unlock the secret key for
user: "John Doe <jdoe@doeworks.ch>"
4096-bit RSA key, ID 48CCEEDF, created 2015-12-11
```

Export the GPG keypair

- Now that we've created the master keypair—public, private keys & revocation certificate—and used it to create a subkey, we should export it & back it up somewhere safe:

```
$ gpg2 --export-secret-keys --armor 48CCEEDF > 48CCEEDF-private.gpg
$ gpg2 --armor --export 48CCEEDF > 48CCEEDF-public.gpg
```

- We can also export the subkey for use on other laptops/workstations:

```
$ gpg2 --armor --export-secret-subkeys A85EA103 > A85EA103-private-subkey.gp
```

- Now that we've exported the master keypair & kept it somewhere safe, we can delete this local copy:

```
$ gpg2 --delete-secret-keys 48CCEEDF
```

Using GPG subkey

- You can import your subkey in your other laptop/workstation & use for daily tasks. Don't forget to shred the secret file once you're done with importing it!

```
$ gpg2 --import A85EA103-private-subkey.gpg
$ shred --remove A85EA103-private-subkey.gpg
```

- You can verify that the subkey was successfully imported by listing all private keys:

```
$ gpg2 --list-secret-keys
/home/jdoe/.gnupg/secring.gpg
------------------------------
sec#  4096R/48CCEEDF 2015-12-11 [expires: 2016-12-10]
uid                  John Doe <jdoe@doeworks.ch>
ssb   4096R/8E93D277 2015-12-11
ssb   4096R/A85EA103 2015-12-11
```

- You can notice that there's a line that begins with `sec#` instead of `sec`, the pound(#) sign means the signing master key is not this keyring.

Lost/Compromised Subkey

- In case of emergency whereby your subkey is lost & or compromised, get your master keypair, import it & revoke your subkey:

```
$ gpg2 --import 48CCEEDF-public.gpg 48CCEEDF-private.gpg
gpg: key 48CCEEDF: public key "John Doe <jdoe@doeworks.ch>" imported
gpg: key 48CCEEDF: secret key imported
gpg: key 48CCEEDF: "John Doe <jdoe@doeworks.ch>" not changed
gpg: Total number processed: 2
gpg:               imported: 1  (RSA: 1)
gpg:              unchanged: 1
gpg:       secret keys read: 1
gpg:   secret keys imported: 1
```

- Using `gpg2 --edit-key`, select the compromised key—in this case, key 2—& revoke it:

```
$ gpg2 --edit-key 48CCEEDF
...
pub  4096R/48CCEEDF  created: 2015-12-11  expires: 2016-12-10  usage: SC
                     trust: ultimate      validity: ultimate
sub  4096R/8E93D277  created: 2015-12-11  expires: 2016-12-10  usage: E
sub  4096R/A85EA103  created: 2015-12-11  expires: 2016-12-10  usage: S
[ultimate] (1). John Doe <jdoe@doeworks.ch>

gpg> key 2

pub  4096R/48CCEEDF  created: 2015-12-11  expires: 2016-12-10  usage: SC
                     trust: ultimate      validity: ultimate
sub  4096R/8E93D277  created: 2015-12-11  expires: 2016-12-10  usage: E
sub* 4096R/A85EA103  created: 2015-12-11  expires: 2016-12-10  usage: S
[ultimate] (1). John Doe <jdoe@doeworks.ch>

gpg> revkey
Do you really want to revoke this subkey? (y/N) y
Please select the reason for the revocation:
  0 = No reason specified
  1 = Key has been compromised
  2 = Key is superseded
  3 = Key is no longer used
  Q = Cancel
Your decision? 1
...
gpg> save
```

- Finally, you have to distribute your key to a keyserver:

```
$ gpg --keyserver hkp://keys.gnupg.net --send-key 48CCEEDF
```

Further Reading

1. GNU Privacy Guard(GPG) Handbook (https://www.gnupg.org/gph/en/manual.html)
2. GPG Subkeys (https://wiki.debian.org/Subkeys)
3. GPG How-to (https://help.ubuntu.com/community/GnuPrivacyGuardHowto)

Contact me (mailto:oguyajames@gmail.com)

← Older (https://oguya.ch/posts/2016-02-13-mysql-replication/)

Newer → (https://oguya.ch/posts/2016-04-13-safely-rotating-mysql-slow-logs/)