PIS

Policy for Internal Security

Rémi Heredero - 2026-02-14

gpg ssh x509 YubiKey security

1 Policy for Internal Security

This repo describes my P.I.S. (**P**olicy for **I**nternal **S**ecurity). You'll find my personal guidelines for SSH / GPG on YubiKey and how to configure and create a key / certificate.

I have several YubiKey, each with different purpose.

- **Master YubiKey:** A YubiKey 5C that keeps Master GPG, SSH CA and root CA for my server. These YubiKey stay in a secure place and will be used only to sign subkey, new SSH Key or new IC.
- **Keyring YubiKey:** A YubiKey 5C NFC on my keyring. This YubiKey is used to keep some passkeys and TOTP for some app. This also contains a GPG subkey and ssh key signed by SSH CA on Master YubiKey.
- **Laptop YubiKey:** A small YubiKey 5 Nano in my laptop that contains a GPG subkey and an ssh key like Keyring YubiKey. This YubiKey Nano stays mostly on my laptop. It slightly increases the security compared to having gpg and ssh directly on my laptop.
- **Backup YubiKey:** A YubiKey 5C, keep in secure place that contains the same passkey and TOTP that the Keyring YubiKey. As security depends on the weakest security measure, some of my apps have passkey enforced or TOTP on YubiKey only. This backup key prevents from losing access in case of losing the Keyring YubiKey.

1.1 Install dependencies

```bash
sudo dnf install yubikey-manager gnupg pcsc-lite pcsc-tools
sudo systemctl start pcscd
sudo systemctl enable pcscd
```

2 GPG

Different types of a GPG key exist:

- [C]ertification key (1): Used to sign other keys, this is the Master Key that we want to keep in a secure place.
- [S]igning key (10): Used to sign documents, emails, etc.
- [E]ncryption key (12): Used to encrypt documents, emails, etc.
- [A]uthentication key (11): Used for authentication, for example, for SSH.

I have the strategy below:

Type of key	Validity	Master YK	Keyring YK	Laptop YK
Master [C]	10 Years	Generate in key	-	-
Sign [S]	1Y (renew)	-	unique	unique
Encrypt [E]	10 Years	Generate	clone	clone
Auth [A]	1Y (renew)	-	unique	unique

2.1 Master YubiKey

2.1.1 Run GPG on YubiKey, change PIN/Admin/Reset and change a default key

```bash
gpg --card-edit
admin
passwd # To change PIN (default: 123456) / Admin code (default: 12345678) / Reset code
key-attr # Change type of key (select ECC 25519 for all keys)
```

2.1.2 Generate key

```bash
generate
```

Keep aside the revocation file created on your computer

2.2 Keyring YubiKey

2.2.1 Create sub-keys

We have to create the subkeys on RAM and move it on the right YubiKey after.

First, connect Master YubiKey on a laptop and edit the key

```bash
gpg --expert --edit-key [master_key_id]
```

Create a 1-year subkey for [S]igning (10) and [A]uthentication (11).

```bash
addkey
```

Save and disconnect YubiKey Master.

2.2.2 Move sub-keys

Connect YubiKey Keyring or YubiKey Laptop.

```bash
gpg --edit-key [master_key_id]
```

1. Use `key N` to select the key number *N*
2. `keytocard`
3. Use `key N` to deselect the key number *N* Repeat the operation for Signature and Authentication key

`save` when everything done

2.2.3 Encryption key

As the encryption key is cloned on several YubiKey, this key needs to be created locally, backup and then copied in all YubiKey.

Create a 10-year subkey for [E]ncryption (12)

```bash
addkey
save
```

Remember to save

Now, export the encryption key

```bash
gpg --armor --export-secret-subkeys [master_key_id]> /tmp/backup_keys.asc
```

Now move the encryption key to the Master YubiKey with **`keytocard`**. Once done and **`save`** the key is deleted of the local environnement.

Now for each other YubiKey, import the backup key and move it to the YubiKey

```bash
gpg --import /tmp/backup_keys.asc
gpg --edit-key [master_key_id]
key N # Select the encryption key
keytocard
save
```

Remember to securely delete the backup file after.

```bash
shred -u /tmp/backup_keys.asc
```

2.3 Export public key

When all subkeys are on the right YubiKey, we can export the public key to share it.

```bash
gpg --armor --export [master_key_id] > master-public.asc
```

This operation has to be done on each renewal of the signing and authentication key, as they are unique on each YubiKey.

3 SSH

3.1 Master YubiKey

I use Yubico authentificator 7.3.0 to change PIN / PUK and Management Key. I also create a certificate in slot 9c of the PIV function with ECCP384 for 10 years (like GPG).

I change PIN for PIV in Yubico authentificator GUI. It's also possible to do it with **`ykman piv access`**.

3.1.1 Generate a private key for the CA

Management Key is requested

```bash
ykman piv keys generate --algorithm ECCP384 9c public-ca.pem
```

3.1.2 Generate a self-signed certificate for the CA

PIN is requested

```bash
ykman piv certificates generate --subject "CN=SSH CA Klagarge" --valid-days 3650
9c public-ca.pem
```

3.1.3 Export and add on server

Convert to a standard public key

```bash
ssh-keygen -i -m PKCS8 -f public-ca.pem > ssh_ca_master.pub
```

`ssh_ca_master.pub` is the public key to put on the server.

For my use case, I want only 1 user with this method, so, I add a line in the **`~/.ssh/authorized_keys`** file of the user with the option **`cert-authority`** to allow this CA to sign SSH key for authentication.

```bash
cert-authority ecdsa-sha2-nistp384 ...
```

For global use, you can add the following line in **`/etc/ssh/sshd_config`** of the server after copying the public key in **`/etc/ssh/ssh_ca_master.pub`** on the server.

```bash
TrustedUserCAKeys /etc/ssh/ssh_ca_master.pub
```

Restart sshd when done with: **`sudo systemctl restart sshd`**

3.2 Child Keys

3.2.1 Create an SSH key

Disconnect YubiKey Master and connect YubiKey Keyring (or YubiKey Laptop, but commands need to be adapted). Create a key with options

```bash
ssh-keygen -t ed25519-sk -O resident -O application=ssh:Klagarge-Keyring -C
"YubiKey Keyring" -f ~/.ssh/id_ed25519_sk-keyring
```

- **`id_ed25519_sk-keyring`** is the private key that stay on the YubiKey (it's a pointer to the key on the YubiKey)
- **`id_ed25519_sk-keyring.pub`** is the standard public key that can be shared and used to sign with the CA

3.2.2 Sign it with the CA

Now disconnect YubiKey Keyring and connect YubiKey Master to sign the public key with the CA

```bash
ssh-keygen -D /usr/lib64/libykcs11.so.2 \
    -s ssh_ca_master.pub \
    -I "Klagarge-Keyring-2026" \
    -n remi,root,Klagarge,her \
    -V +365d \
    ~/.ssh/id_ed25519_sk-keyring.pub
```

This creates the file **`id_ed25519_sk-keyring-cert.pub`** that is the certificate to use for authentication.

4 LUKS

It's possible to add a Yubikey as a second option to unlock a LUKS partition.

The first step is to find the encrypted partition.

```bash
lsblk
```

`nvme1n1p3` is the encrypted partition in my case.

4.1 Enroll

Add a new way to unlock the partition with the YubiKey. This add a FIDO device, not replace the password way. You can still unlock the partition with the password if you forget the YubiKey.

This step have to be done for each Yubikey you want to use to unlock the partition.

```bash
sudo systemd-cryptenroll --fido2-device=auto /dev/nvme1n1p3
```

Actual passphrase is requested, then Yubikey Fido2 PIN, then you have to touch it 2 time to confirme presence.

4.2 Config `/etc/crypttab`

This step have to be only once.

Backup and edit crypttab

```bash
sudo cp /etc/crypttab /etc/crypttab.bak
sudo nano /etc/crypttab
```

Add **`,fido2-device=auto`** (without any space) at the end of the line that describe the encrypted partition. It should look like that at the end:

```
luks-1234... UUID=1234... none discard,fido2-device=auto
```

4.3 Re-Generate initramfs

This step have to be only once.

After enrolling the YubiKey, you need to re-generate the initramfs to be able to unlock the partition at boot time.

```bash
sudo dracut -f
```

5 Troubleshooting

5.1 GPG

Sometimes, for unknown (for me) reason, you need to kill the gpg-agent to be able to use the YubiKey again.

```bash
gpgconf --kill gpg-agent
```

You also sometimes need to restart the pcscd service if the YubiKey is not detected.

```bash
sudo systemctl restart pcscd
```

5.2 SSH

If you have an issue with your gpg-agent, you maybe have to wake up the ssh-agent to be able to use the YubiKey again. This basic commande wake up the ssh-agent.

```bash
eval $(ssh-agent) # Should response with "Agent pid [number]"
```

If your key is not found by the ssh agent, you have to manually add the key with:

```bash
ssh-add ~/.ssh/id_ed25519_sk-keyring
```