Certificate Authority

1 - Repository

https://github.com/jlangenegger/ssh_certificate/

2 - Prepare Yubikey

This needs to be done on a offline machine!

Step 1 - Activate CCID

Activate the USB interface CCID on the Yubikey. To install the Yubikey manager check https://developers.yubico.com/yubikey-manager/.

Activate the mode using:

```
ykman mode CCID
```

Step 2 - Install libraries that are later used

To setup the yubikey the `yubico-piv-tool` is used. It must be installed from source to work correctly. For the installation the following packages are needed:

```
apt-get install git cmake build-essential libtool libssl-dev pkg-config check libpcsclite-dev gengetopt help2man
```

Step 3 - Install the tool

```
git clone https://github.com/Yubico/yubico-piv-tool.git

cd yubico-piv-tool

mkdir build; cd build
cmake ..
make
sudo make install
```

Step 4 - Change default pins and management key of yubikey

To prepare the PIV applet in the YubiKey the management key, the pin and the punk needs to be set.

```
yubico-piv-tool -a set-mgm-key -n 010203040506070801020304050607080102030405060708
yubico-piv-tool -k $key -a change-pin -P 123456 -N 123456
yubico-piv-tool -k $key -a change-puk -P 12345678 -N 12345678
```

Step 5 - Generate RSA private keys for SSH Host CA

Then generate a RSA private key for the SSH Host CA, and generate a dummy X.509 certificate for that key. The only use for the X.509 certificate is to make PIV/PKCS#11 happy. They want to be able to extract the public-key from the smart-card, and do that through the X.509 certificate.

```
YUBIKEYNUM=0
PATH_TO_CERTIFICATE="/etc/ssh-ca"

mkdir -p $PATH_TO_CERTIFICATE

# generate key directly on yubikey and self-sign the certificate
yubico-piv-tool -k 123456 -s 9c -a generate -o yubikey$YUBIKEYNUM.pem
yubico-piv-tool -k 123456 -a verify-pin -a selfsign-certificate --valid-days 10000 -s 9c -S "
/CN=yubikey`$YUBIKEYNUM`/" -i yubikey$YUBIKEYNUM.pem -o yubikey$YUBIKEYNUM-cert.pem
# import self-signed certificate
yubico-piv-tool -k 123456 -a import-certificate -s 9c -i yubikey$YUBIKEYNUM-cert.pem

# convert public key to RSA
ssh-keygen -f yubikey$YUBIKEYNUM.pem -i -mPKCS8 > yubikey$YUBIKEYNUM.pub

# move public key to correct place and remove leftovers
mv yubikey$YUBIKEYNUM.pub $PATH_TO_CERTIFICATE
rm yubikey$YUBIKEYNUM-cert.pem yubikey$YUBIKEYNUM.pem
```

3 - Sign client's public keys

To sign client's public keys there is the script `'generate_client_certificate.sh'` to simplify the procedure.
The scripts does have the following options:

- -g
  - This takes a github user name as an argument and generates a certificate for each key stored in github.
- -f
  - Instead of the github user name, one can provide a file that contains all the keys.
  - Nevertheless the flag `'-g'` is needed as the certificate holder's name.
- -V
  - Add the validity interval of a certificate
  - Per default a certificate is valid for seven days.
  - More information can be found here: validity_interval
- -n
  - This flag restricts the certificate to a list of principals that the client is allowed to log in.

The output of `'generate_client_certificate.sh'` is a .tar archive that contains the certificate, the public key that is used to authenticate servers as well as an instruction to install the certificate on the client's machine. It is stored in the home directory `'$HOME/signed_keys'`.

4 - Troubleshooting

Export public key

```
PATH_TO_YKCS11="/usr/local/lib/libykcs11.so"
ssh-keygen -D PATH_TO_YKCS11 -e
```

Reset PIV on Yubikey

```
yubico-piv-tool -averify-pin -P471112
yubico-piv-tool -averify-pin -P471112
yubico-piv-tool -averify-pin -P471112
yubico-piv-tool -averify-pin -P471112
yubico-piv-tool -achange-puk -P471112 -N6756789
yubico-piv-tool -achange-puk -P471112 -N6756789
yubico-piv-tool -achange-puk -P471112 -N6756789
yubico-piv-tool -achange-puk -P471112 -N6756789
yubico-piv-tool -areset
yubico-piv-tool -aset-chuid
yubico-piv-tool -aset-ccc
```

Viewing an SSH certificate

```
ssh-keygen -L -f hello_world-cert.pub
hello_world-cert.pub:
        Type: ssh-rsa-cert-v01@openssh.com host certificate
        Public key: RSA-CERT SHA256:diEzE7FgTzHHu87G3ssTLkJcGIikFWe832M3q7OMpS/0
        Signing CA: RSA SHA256:dGhZ6Zs5q9+6Ze3dt4zfbcmz+soOudwe56TfGvY+U
        Key ID: "hello_world"
        Serial: 0
        Valid: from 2020-05-29T06:09:00 to 2021-05-28T06:10:37
        Principals:
                hello_world.netdef.org
        Critical Options: (none)
        Extensions: (none)
```

Sign server's RSA key manually

```
YUBIKEYNUM=0
PATH_TO_CERTIFICATE="/etc/ssh-ca"
PATH_TO_YKCS11="/usr/local/lib/libykcs11.so"

ssh-keygen  -D $PATH_TO_YKCS11
            -s $PATH_TO_CERTIFICATE/yubikey$YUBIKEYNUM.pub
            -I server_name \
            -h \
            -n server.netdef.org \
            -V +52w \
            /etc/ssh-ca/ssh_host_rsa_key.pub
```

Options explanation:

- -D
  - is used to access the yubikey
- -s
  - provides the public certificate to access the yubikey
- -I server_name
  - The key identifier to include in the certificate.
- -h
  - Generate a host certificate (instead of a user certificate)
- -n server.netdef.org
  - The principal names to include in the certificate.
  - For host certificates this is a list of all names that the system is known by.
  - Note: Use the unqualified names carefully here in organizations where hostnames are not unique (ca.netdef.org vs. ca.dev.netdef.org)
- -V +52w
  - The validity period.
  - For host certificates, you'll probably want them pretty long lived.
  - This setting sets the validity period from now until 52 weeks hence.
- /etc/ssh-ca/ssh_host_rsa_key.pub
  - The path to the host RSA public key to sign.
  - Our signed host key certificate will be /etc/ssh-ca/ssh_host_rsa_key-cert.pub.

Sign client's RSA key manually

```
YUBIKEYNUM=0
PATH_TO_CERTIFICATE="/etc/ssh-ca"
PATH_TO_YKCS11="/usr/local/lib/libykcs11.so"

ssh-keygen  -D $PATH_TO_YKCS11
            -s $PATH_TO_CERTIFICATE/yubikey$YUBIKEYNUM.pub
            -I client_name \
            -n root \
            -V +24h \
            /etc/ssh_ca/id_rsa.pub
```

Options explanation:

- -D
  - is used to access the yubikey
- -s
  - provides the public certificate to access the yubikey
- -I client_name
  - The key identifier to include in the certificate.
- -n root
  - The principal names to include in the certificate.
  - For client certificates this is a list of all users that the system is allowed to log in.
- -V +24h
  - The validity period.
  - For client certificates, you'll probably want them short lived.
  - This setting sets the validity period from now until 24 hours.
  - One an SSH session is authenticated the certificate can safely expire without impacting the established session.
- /etc/ssh_ca/id_rsa.pub
  - The name of the host RSA public key to sign.
  - Our signed host key (certificate) will be /etc/ssh_ca/ssh_host_rsa_key-cert.pub.