

# Script create OpenSSL Certificate Authority

With the scripts provided, you can create an OpenSSL two-tier CA. The OpenSSL commands and the templates used are based on the following blog post:

- [Advanced PKI](#)
- [github](#) (mrwiora)

I highly recommend to read the post before creating the CA.

The scripts were developed on a Synology Diskstation with a bash version *4.4.23(1)-release* and an OpenSSL version of *1.1.1n 15 Mar 2022*.

The scripts perform the following tasks:

- Collecting information for configuring the CA
- Creating the directory structure and creating an OpenSSL two-tier CA
- Create certificate trust chain
- Requesting certificates

To implement the CA you need a basic understanding of Linux shell and certificate authorities. To run the configuration and certificate-requesting scripts, you need to logon/connect (via ssh) to your Linux box. This document does not contain any tips how to manage a Linux box. On a Windows computer you can install the Windows Terminal from Microsoft and the OpenSSH client to connect to your Linux box. The scripts were developed with these tools.

**Limitation:** Currently you can only issue certificates requested by the script. In a future version, maybe, you should also be able to issue certificates based on a *csr*, which was not created by the script.

## Disclaimer

The scripts were developed for a lab environment and experimental use. The scripts do not implement any security settings on the CA files. It's up to you to take the appropriate security measurements. Especially a root CA should not be online, except for signing a sub CA, publishing a certificate revocation list (CRL), create trust chain, etc.

## Preparing the Environment

To implement the CA, you have to create two directories on your Linux system. In one of the directories copy the files provided. The second directory is the root directory for the CA implementation.

Example:

- */volume1/install*
- */volume1/ca*

In the first step, copy the files to the directory */volume1/install*. Under the directory *install* the following structure should be created:

caConf		Fol...	13.05.2022 08:10:23
confTemplates		Fol...	12.05.2022 18:10:37
scripts		Fol...	13.05.2022 08:46:27
createCAconf.sh	1.1 KB	SH ...	13.05.2022 07:41:14
init-ca.sh	7.8 KB	SH ...	13.05.2022 10:35:57
prepCustomVars.sh	9.5 KB	SH ...	12.05.2022 18:58:45

## Create CA Configuration

Before we can start implementing the CA, we have to collect some configuration data. To collect the information please run the script

- createCAconf.sh

If you have copied the script to the folder

- /volume1/install

you can run the script with the following command:

- /volume1/install/createCAconf.sh

The script will ask you a couple of questions which are relevant for the setup of the root and signing CA. The script collects three groups of information.

- Basic data for CA implementation
- Information for root and signing CA configuration
- Organizational data for certificates

### *Basic Data for CA Implementation*

In this section the following data will be collected:

- Root directory for CA installation (full path)
- The URL for the certificate revocation list
- The URL for the authority of information access

```
Collecting some data for CA setup.
Collecting basic configuration information

Collecting directory and URL information
Please enter the root directory for the CA installation (e.g. /volume1/ca)
/volume1/ca/thePenguinCA
Please enter URL for the certificate revocation list (e.g. http://crl.your-company.com)
crl.the-penguin.lnx
Please enter URL for the AIA (e.g. http://aia.your-company.com)
aia.the-penguin.lnx

Directory and URL configuration
Root directory for CA installation:    /volume1/ca/thePenguinCA
Base URL for CRL:                     crl.the-penguin.lnx
Base URL for AIA:                     aia.the-penguin.lnx
Is the list with the directory entries correct? [y/n]: |
```

After the data was collected the script asks for confirmation that the data is correct.

### *Information for Root and Signing CA Configuration*

In this section the following data will be collected for the root and signing CA:

- Validity period for the CA in years
- Validity period for the CRL in days
- Common name for the (root/signing) CA
- Name of the CRL file (will be combined with the URL from the previous section)
- Name of the AIA file (will be combined with the URL from the previous section)

```
Collecting validity period information for root and signing ca
Collecting configuration for root CA.
Please enter the validity period in YEARS for the root CA
30
Please enter the CRL validity period in DAYS for the root CA
180
Please enter the common name (cn) for the root CA
The-Penguin-Root
Please enter the name for the root CA CRL file (e.g. root-ca.crl)
root-ca.crl
Please enter the name for the root CA AIA file (e.g. root-ca-aia.cer)
root-ca-aia.cer

Collectin configuration for signing CA.
Please enter the validity period in YEARS for the signing CA
25
Please enter the CRL validity period in DAYS for the signing CA
31
Please enter the common name (cn) for the signing CA
The-Penguin-Signing-CA
Please enter the name for the signing CA CRL file (e.g. signing-ca.crl)
signing-ca.crl
Please enter the name for the signing CA AIA file (e.g. signing-ca-aia.cer)
signign-ca-aia.cer

Root CA
Root CA valid in years:          30
Root CA CRL valid in days:      180
Root ca common name:           The-Penguin-Root
Root ca CRL file name:         root-ca.crl
Root ca AIA file name:         root-ca-aia.cer

Signing CA
Signing CA valid in years:       25
Signing CA CRL valid in days:   31
Signing ca common name:        The-Penguin-Signing-CA
Signing ca CRL file name:      signing-ca.crl
Signing ca AIA file name:      signign-ca-aia.cer
Are the root and signing CA configuration correct? [y/n]: |
```

After the data was collected the script asks for confirmation that the data is correct.

## Organizational Configuration

In this section the following data will be collected:

- Country code
- Province
- Location
- Organizational unit
- Organization name

From the data of this section, the subject name in the certificate will be created. If you don't want one of the items above, to be part of the subject name, simply enter a dot ".". If you want to omit settings, please make sure, that the respective *conf* file is configured appropriately. Per default the *conf* files require:

- Common name
- Country code
- Organization

```
57
58 [ root_ca ]
59 certificate           = $dir/$ca/certs/$ca_certFile      # cert of root ca
60 private_key          = $dir/$ca/private/$ca_keyFile      # CA private key
61 new_certs_dir        = $dir/$ca/newcerts                # Certificate archive
62 serial               = $dir/$ca/db/$ca.crt.srl           # Serial number file
63 crlnumber            = $dir/$ca/db/$ca.crl.srl           # CRL number file
64 database             = $dir/$ca/db/$ca.db               # Index file
65 unique_subject       = no                               # Require unique subject
66 default_days         = 3652                              # How long to certify for
67 default_md           = sha256                            # MD to use
68 policy               = match_pol                         # Default naming policy
69 email_in_dn          = no                                # Add email to cert DN
70 preserve             = no                                # Keep passed DN ordering
71 name_opt             = $name_opt                         # Subject DN display options
72 cert_opt             = ca_default                       # Certificate display options
73 copy_extensions      = none                              # Copy extensions from CSR
74 x509_extensions      = signing_ca_ext                   # Default cert extensions
75 default_crl_days     = 365                              # How long before next CRL
76 crl_extensions       = crl_ext                          # CRL extensions
77
78 [ match_pol ]
79 countryName          = match                             # Must match
80 stateOrProvinceName  = optional                          # Included if present
81 localityName         = optional                          # Included if present
82 organizationName     = match                             # Must match
83 organizationalUnitName = optional                        # Included if present
84 commonName           = supplied                          # Must be present
85
```

If this configuration is not adequate for your environment, please update the *conf* files before running the CA configuration script.

The *conf* files can be found in the install directory under

- caConf

For our example installation the path to the *conf* files is:

- /volume1/install/caConf

```

Collecting organizational information
Please enter the country code (ISO 3166-1 2 letter)
AT
Please enter the name of your province
Somwehre
Please enter the name of your location
Out-there
Please enter the name of your organizational unit
.
Please enter the name of your organization
The-Penguin
Please verify if the following values are correct. A dot means not present in certificate subject.
Country code:          AT
Province:              Somwehre
Location:              Out-there
Organizational unit:    .
Organization:          The-Penguin
Is the organizational information correct? [y/n]: |

```

After the data was collected the script asks for confirmation that the data is correct.

### *Saving CA Configuration*

After all data is collected, a summery of the configuration is displayed for review.

```

Summarizing collected data for OpenSSL CA installation
Basic data:
Root directory for CA installation:  /volume1/ca/thePenguinCA
Base URL for CRL:                   crt.the-penguin.lnx
Base URL for AIA:                   aia.the-penguin.lnx

Root CA installation
Root CA valid in years:              30
Root CA CRL valid in days:          180
Root ca common name:                The-Penguin-Root
Root ca CRL file name:               root-ca.crl
Root ca AIA file name:               root-ca-aia.cer
Signing CA installation
Signing CA valid in years:            25
Signing CA CRL valid in days:        31
Signing ca common name:              The-Penguin-Signing-CA
Signing ca CRL file name:             signing-ca.crl
Signing ca AIA file name:             signign-ca-aia.cer

Organizational information - a dot means not present in certificate subject
Country code:                        AT
Province:                            Somwehre
Location:                            Out-there
Organizational unit:                  .
Organization:                        The-Penguin
Do you want to save the data? [y/n]:y

Writing configuration to /volume1/Test22/scripts/helperScripts/customVars.sh

Successfully saved configuration to /volume1/Test22/scripts/helperScripts/customVars.sh

```

If the collected data is complete, press y to save the data.

With the data collected, the two-tiered CA can be configured.

## Change Configuration Data

As long as you have not configured the CA's it is possible to change the collected configuration data. To change the CA configuration data start the script

- createCAconf.sh

The script will detect that a configuration already exist and allows to

- edit a particular configuration
- edit all configurations
- cancel the operation

```
Collecting some data for CA setup.  
Edit Configurations  
Press 1 for Basic Configuration  
Press 2 for CA Installation Configuration  
Press 3 for Organizational Configuration  
Press 4 for All Configuration  
Press c Cancel  
Type the number of the configuration you want to edit, 4 for all configurations or 0 to cancel.  
Your selection: |
```

**Important:** The changes are only effective if the CA's are not yet configured.

## Two-Tiered CA Configuration

If all the configuration data is collected, the two-tiered CA will be configured, based on the collected data. The collected data is stored in the file

- customVars.sh

The file can be found in the subdirectory

- scripts/helperScripts

under the installation directory. Based on the example directory in this document, the content of the file can be viewed with the following command:

- cat /volume1/install/scripts/helperScripts/customVars.sh

```
caRootDir=/volume1/ca/thePenguinCA
crlBaseDir=crl.the-penguin.lnx
aiaBaseDir=aia.the-penguin.lnx
rootCADaysValid=10958
rootCACRLDaysValid=180
customDefault_RootCAName=The-Penguin-Root
rootCACrlName=root-ca.crl
rootCAaiaName=root-ca-aia.cer
signingCADaysValid=9131
signingCACRLDaysValid=31
signingCACrlName=signing-ca.crl
signingCAaiaName=signign-ca-aia.cer
customDefault_SigningCAName=The-Penguin-Signing-CA
customDefault_defCountryCode=AT
customDefault_defProvince=Somwehre
customDefault_defLocation=Out-there
customDefault_defOU=.
customDefault_defOrganization=The-Penguin
```

As you can see in the screenshot above, the validity period for the root and signing CA are converted from years to **days**!

**Important:** Please don't edit the file

- customVars.sh

manually. If you edit the file manually, the script could fail.

## CA Configuration Files

Before you start the CA configuration, review the *conf* files for the CA's. The *conf* files can be found in the sub directory

- caConf

under the installation directory.

Please don't change anything in the *default* section of the *conf* files. The *default* section is used to define variables for the CA configuration. In the remaining sections, please don't change any

- names of variables
- paths

Please make sure, that the root directory for the CA installation exists and is empty. The script will neither create the directory nor except a not empty directory.

Before starting the configuration, prepare two strong pass phrases for the keys of the

- root CA
- signing CA

To start the configuration of the two-tiered CA run the script

- init-ca.sh

In the example used in this document, the script will be launched with the following command:

- /volume1/Install/init-ca.sh

```
Verifying existing directory structure.
Existing directory structure is OK.
Creating directory structure for ca
Creating files for CAs
Copy conf file for root ca to /volume1/ca/thePenguinCA/root-ca/root-ca.conf
Copy conf file for issuing ca to /volume1/ca/thePenguinCA/signing-ca/signing-ca.conf
Copy templates to /volume1/ca/thePenguinCA/signing-ca/conf-templates
Updating root CA conf files
Updating signing CA conf files
Copy file createTrustChain.sh to /volume1/ca/thePenguinCA/scripts
Create key for root ca - key pass phrase for root ca requiered
Generating a RSA private key
.....++++
.....++++
writing new private key to '/volume1/ca/thePenguinCA/root-ca/private/root-ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
Signing certificate for root ca - key pass phrase for root ca requiered
Using configuration from /volume1/ca/thePenguinCA/root-ca/root-ca.conf
Enter pass phrase for /volume1/ca/thePenguinCA/root-ca/private/root-ca.key:
Check that the request matches the signature
Signature ok
```

Follow the instructions on screen. If the pass phrase for the root CA is required, the text will be displayed in red.



```

Certificate Details:
  Serial Number: 1 (0x1)
  Validity
    Not Before: May 13 08:29:39 2022 GMT
    Not After : May 13 08:29:39 2052 GMT
  Subject:
    countryName           = AT
    stateOrProvinceName   = Somwehre
    localityName          = Out-there
    organizationName      = The-Penguin
    commonName            = The-Penguin-Root
  X509v3 extensions:
    X509v3 Key Usage: critical
      Certificate Sign, CRL Sign
    X509v3 Basic Constraints: critical
      CA:TRUE

```

On screen, you get a summary of the root CA configuration.

```

Create key and signing request for signing ca - key pass phrase for signing ca requiered
Generating a RSA private key
.....
.....++++
.....
.....++++
writing new private key to '/volume1/ca/thePenguinCA/signing-ca/private/signing-ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
Signing certificate for signing ca with key pass phrase for root ca requiered
Using configuration from /volume1/ca/thePenguinCA/root-ca/root-ca.conf
Enter pass phrase for /volume1/ca/thePenguinCA/root-ca/private/root-ca.key:
Check that the request matches the signature
Signature ok

```

If the configuration of the root CA was successful, the script continues with the configuration of the signing CA. In the next step the pass phrase for the key of the signing CA is required (screenshot above), now the text will be displayed in yellow. Because the signing CA will be signed by the root CA, the pass phrase for the root CA key is required too (text in red).

```

Certificate Details:
  Serial Number: 2 (0x2)
  Validity
    Not Before: May 13 08:30:16 2022 GMT
    Not After : May 13 08:30:16 2047 GMT
  Subject:
    countryName           = AT
    stateOrProvinceName   = Somwehre
    localityName          = Out-there
    organizationName      = The-Penguin
    commonName            = The-Penguin-Signing-CA
  X509v3 extensions:
    X509v3 Key Usage: critical
      Certificate Sign, CRL Sign
    X509v3 Basic Constraints: critical
      CA:TRUE, pathlen:0

```

On screen you get a summary of the signing CA configuration.

The two-tiered CA is now configured. Under the CA root directory, in our example

- /volume1/ca

you'll find the following directory structure:

root-ca	Folder
scripts	Folder
signing-ca	Folder
thePenguinCA	Folder
trust-chain	Folder

## CA Scripts

In the course of the CA installation a couple of scripts will be copied to the scripts directory under the CA root directory. Scripts for the following tasks are provided:

- create trust chain
- configure settings
- request certificates

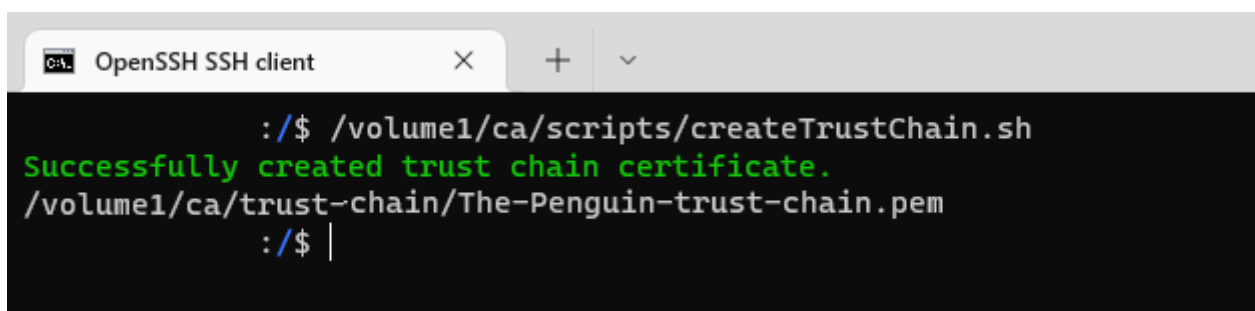
## Create Trust Chain

Computers using certificates issued by the signing-ca, should trust this certificates. To trust certificates issued by certain a CA, the CA certificates must be placed it the appropriate certificate store. On a Windows computer:

- Trusted Root Certification Authorities
- Intermediate Certification Authorities

To create the required certificates run the following command (example environment):

- /volume1/ca/scripts/createTrustChain.sh

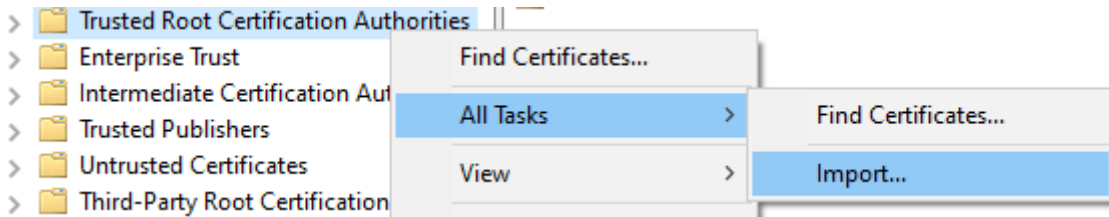


```
:/ $ /volume1/ca/scripts/createTrustChain.sh
Successfully created trust chain certificate.
/volume1/ca/trust-chain/The-Penguin-trust-chain.pem
:/ $ |
```

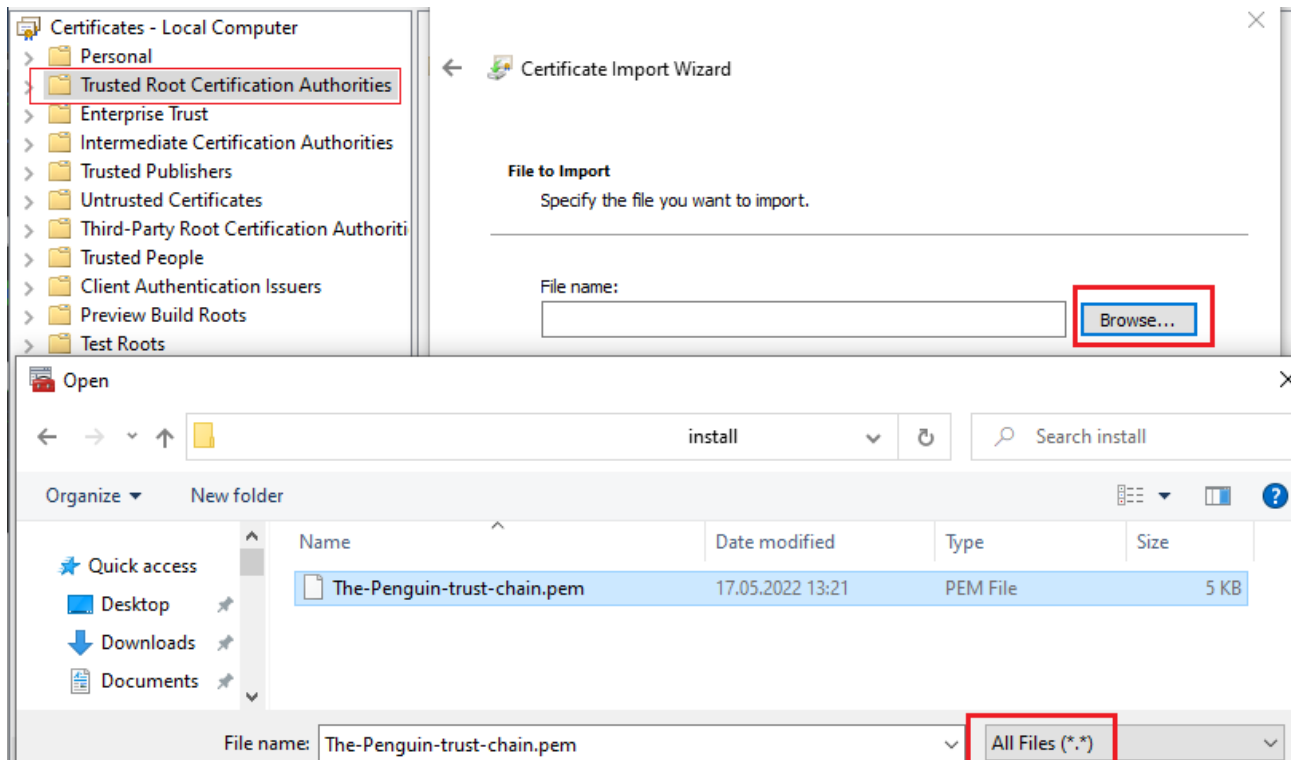
If the certificate was created successfully, you can copy the certificate to your computer and install it, or in a Windows AD environment deploy it via GPO.

On the screenshot above, you can see how to install the certificate on a Windows 10 box. Right click Trusted Root Certification Authorities and select *All Tasks/Import*

On a Windows computer open the mmc with the Certificate Snap-in, right click Trusted Root Certification Authorities and select *All Tasks/Import*



Select the file with the trust chain and import it.



## Request a Certificate

To request a certificate please run the script

- request-certificate.sh

In our example we will run the following command:

- /volume1/ca/scripts/request-certificate.sh

In a default installation, 9 certificate templates are installed. If you issuing the script *request-certificate.sh*, you get the selection from the screenshot below.

```
Please select the template for the certificate request
Press 1 for Code-signing
Press 2 for E-mail
Press 3 for Encryption
Press 4 for Identity
Press 5 for OCPS-signing
Press 6 for Time-stamping
Press 7 for TLS-client
Press 8 for TLS-server
Press 9 for TLS-server-san
Press c to cancel
Type the number of the template or c to cancel and press enter.
Input selection:
```

Type the number of the certificate template or c to cancel the request. In our example we request a TLS certificate with some SAN entries. We select the number 9 (TLS-server-san).

```
Please select the template for the certificate request
Press 1 for Code-signing
Press 2 for E-mail
Press 3 for Encryption
Press 4 for Identity
Press 5 for OCPS-signing
Press 6 for Time-stamping
Press 7 for TLS-client
Press 8 for TLS-server
Press 9 for TLS-server-san
Press c to cancel
Type the number of the template or c to cancel and press enter.
Input selection: 9
Selected template: TLS-server-san
Collecting common name for certificate
Please enter the value for the cn (max. 64 alphanumeric characters are allowed)
www.the-penguin.lnx|
```

In the first step we enter the common name for the certificate. In our case

- [www.the-penguin.lnx](http://www.the-penguin.lnx)

```

Selected tmplate: TLS-server-san
Collecting common name for certificate
Please enter the value for the cn (max. 64 alphanumeric characters are allowed)
www.the-penguin.lnx
Collecting SAN entries for certificate, leave the entry blank if you are finished.
(only entries with max. 64 alphanumeric characters are allowed)
Please enter an entry
bluebox.the-penguin.lnx
Please enter an entry
10.44.56.143
Please enter an entry
f606:8718:ad72:9581:7a0b:6ce1:d639:e688
Please enter an entry

Please review the entries (the first entry in the list represents the common name).
Common name:          www.the-penguin.lnx
SAN enties:
www.the-penguin.lnx
bluebox.the-penguin.lnx
10.44.56.143
f606:8718:ad72:9581:7a0b:6ce1:d639:e688
Is the list with cn and san entries correct? [y/n]: |

```

We are adding an additional name and two IP addresses and leave the last entry blank to finish the collection of SAN entries. Now we get an overview of the configured entries. If the entries are correct we confirm our configuration with y.

```

Verifiying how long the certificate is valid. Enter the number of days or hit ENTER to accept the default
Press ENTER to accept 365:
Do you wish to protect private key of the certificate with a pass phrase?
Protect the private key oft the certificate? [y/n]: n
Key lenght for certificate private key. Enter 2 for 2048 or 4 for 4096 bit
Press ENTER to accept 2048:

Please verify if the following values are correct.
Days valid:          365
Encrypt private key of certificate      no
Certificate Private key length          2048
Are the values correct? [y/n]: |

```

In the next step the following information are collected (screenshot above):

- How long will the certificate be valid (default 365 days)
- Should the private key of the certificate be protected with a pass phrase
- Length of the private key of the certificate (default 2048 bit)

For the key length 2048 and 4096 are allowed. If you type 2, 2048 is selected. If you type 4, 4096 is selected.

If the collected information is correct, we can confirm with y.

In the last step the information for the certificate subject will be collected. As described later in the document, this step can be skipped and default values used.

```

Country code, enter a value or hit ENTER if ok
To delete the value enter .
Press ENTER to accept AT:
Province, enter a value or hit ENTER if ok
To delete the value enter .
Press ENTER to accept Somwehre:
Location, enter a value or hit ENTER if ok
To delete the value enter .
Press ENTER to accept Out-there:
Organizational unit, enter a value or hit ENTER if ok
To delete the value enter .
Press ENTER to accept .:
Organization, enter a value or hit ENTER if ok
To delete the value enter .
Press ENTER to accept The-Penguin:

Please verify if the following values are correct. A dot means not present in certificate subject.
Country code:      AT
Province:          Somwehre
Location:           Out-there
OU:                .
Organization:      The-Penguin
Are the entries correct? [y/n]:

```

We confirm with y if the entries are correct.

The script starts the request of the certificate. To issue the certificate, the pass phrase of the signing CA is required (yellow text).

```

Writing configuration file(s).
Creating certificate request and certificate key.
Generating a RSA private key
.....+++++
.....+++++
writing new private key to '/volume1/ca/thePenguinCA/signing-ca/issuedcerts/www.the-penguin.lnx/www.the-penguin.lnx.key'
-----

Creating certificate and signing certifice with The-Penguin-Signing-CA key.
Using extension server_ext
Pass phrase for The-Penguin-Signing-CA key required.
Using configuration from /volume1/ca/thePenguinCA/signing-ca/signing-ca.conf
Enter pass phrase for /volume1/ca/thePenguinCA/signing-ca/private/signing-ca.key:

```

```

Check that the request matches the signature
Signature ok
Certificate Details:
  Serial Number: 1 (0x1)
  Validity
    Not Before: May 17 12:37:29 2022 GMT
    Not After : May 17 12:37:29 2023 GMT
  Subject:
    countryName           = AT
    stateOrProvinceName   = Somwehre
    localityName          = Out-there
    organizationName      = The-Penguin
    commonName            = www.the-penguin.lnx
  X509v3 extensions:
    X509v3 Key Usage: critical
      Digital Signature, Key Encipherment
    X509v3 Extended Key Usage:
      TLS Web Server Authentication, TLS Web Client Authentication
    X509v3 Subject Key Identifier:
      0B:DB:5D:A8:A2:20:0C:9A:D8:26:4D:3B:44:01:A8:EC:72:6C:E1:14
    Authority Information Access:
      CA Issuers - URI:aia.the-penguin.lnx/signign-ca-aia.cer

    X509v3 CRL Distribution Points:

      Full Name:
        URI:crl.the-penguin.lnx/signing-ca.crl

    X509v3 Subject Alternative Name:
      DNS:www.the-penguin.lnx, DNS:bluebox.the-penguin.lnx, IP Address:10.44.56.143, IP Address:F606:8718:AD72:9581:7A0B
:6CE1:D639:E688
Certificate is to be certified until May 17 12:37:29 2023 GMT (365 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]

```

We have to confirm that we want to sign the certificate.

```

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
Successfully created certificate www.the-penguin.lnx.
Do you want to create a PKCS#12 bundle? [y/n]: y

Creating PKCS#12 bundle
Do you want to include the trust chain? [y/n]: n

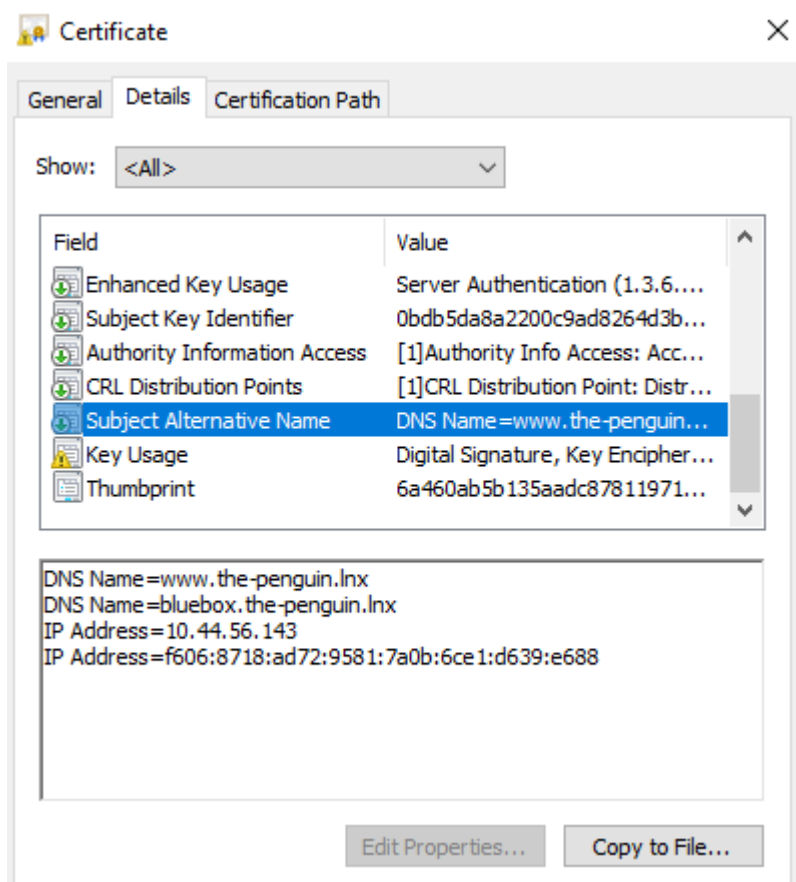
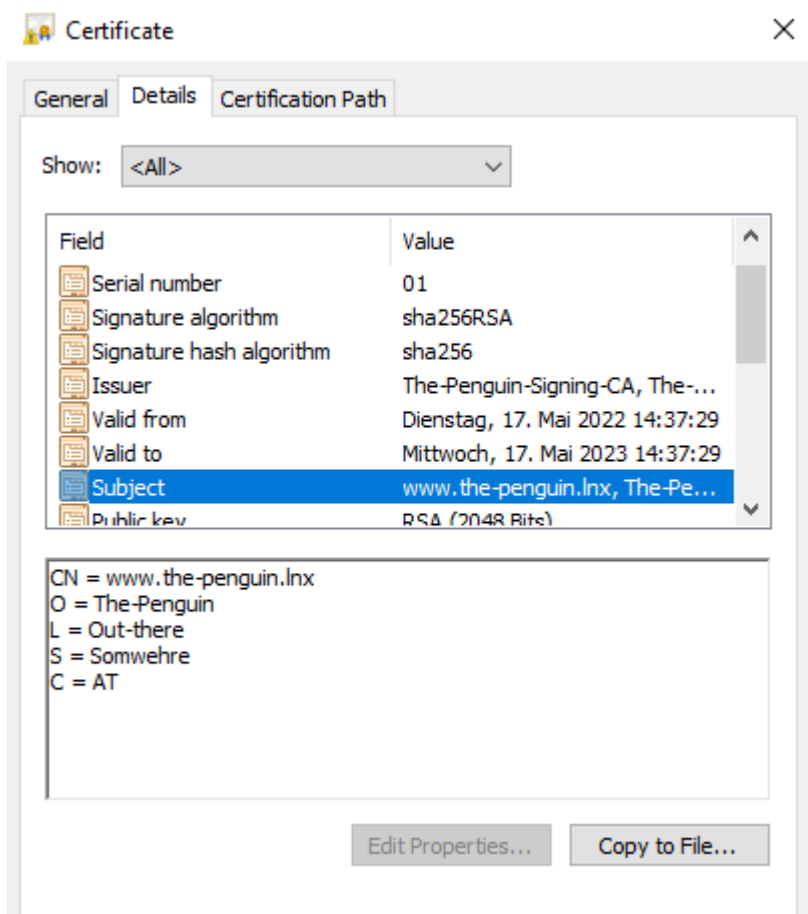
Enter Export Password:
Verifying - Enter Export Password:
Successfully created PKCS#12 bundle www.the-penguin.lnx.p12.

Key and certificates created in directory /volume1/ca/thePenguinCA/signing-ca/issuedcerts/www.the-penguin.lnx
www.the-penguin.lnx.key
www.the-penguin.lnx.crt
www.the-penguin.lnx.pem
02.pem
www.the-penguin.lnx.p12

```

In the final step, we are asked if we want to create a PKCS#12 file. Whether or not you need a PKCS#12 file depends on your requirements. If your application imports the certificate and key as separate files, you shouldn't need it. If you create a PKCS#12 file, you need to provide a password to protect the file. Be aware, that the password is needed for the import of the PKCS#12 file.

If you open the certificate on a Windows box, it should look similar to the following screenshot.





## Configure Certificate Issuing Settings

With the scripts

- `cfgOrganizationSettings.sh`
- `cfgCustomDefaults.sh`

custom settings can be configured.

The scripts are stored in *scripts* directory of the CA root directory. In our example under

- `/volume1/ca/scripts`

## Configure Organization Settings

With the script

- `cfgOrganizationSettings.sh`

you can reconfigure the organizational settings collected during the initialization of the two-tier CA.

The following settings can be configured:

- Country code
- Province
- Location
- Organizational unit
- Organization

```
Collecting organizational information
Please enter the country code (ISO 3166-1 2 letter)
To delete the value enter .
Press ENTER to accept AT:
Please enter the name of your province
To delete the value enter .
Press ENTER to accept Somwehre:
Please enter the name of your location
To delete the value enter .
Press ENTER to accept Out-there:
Please enter the name of your organizational unit
To delete the value enter .
Press ENTER to accept .:
Please enter the name of your organization
To delete the value enter .
Press ENTER to accept The-Penguin:
Please verify if the following values are correct. A dot means not present in certificate subject.
Country code:          AT
Province:              Somwehre
Location:              Out-there
Organizational unit:    .
Organization:          The-Penguin
Is the organizational information correct? [y/n]: y

Do you want to save the data? [y/n]:|
```

Configure the settings depending your requirements and configuration settings in the signing CA conf file. Every settings represents an entry of the subject name. If you don't want an entry to be present in the subject name, enter a . (dot).

If you have configured your entries review them and, if OK, confirm with y. To save the configuration confirm with y again.

## Configure Script Settings

To configure script settings launch the script

- `cfgCustomDefaults`

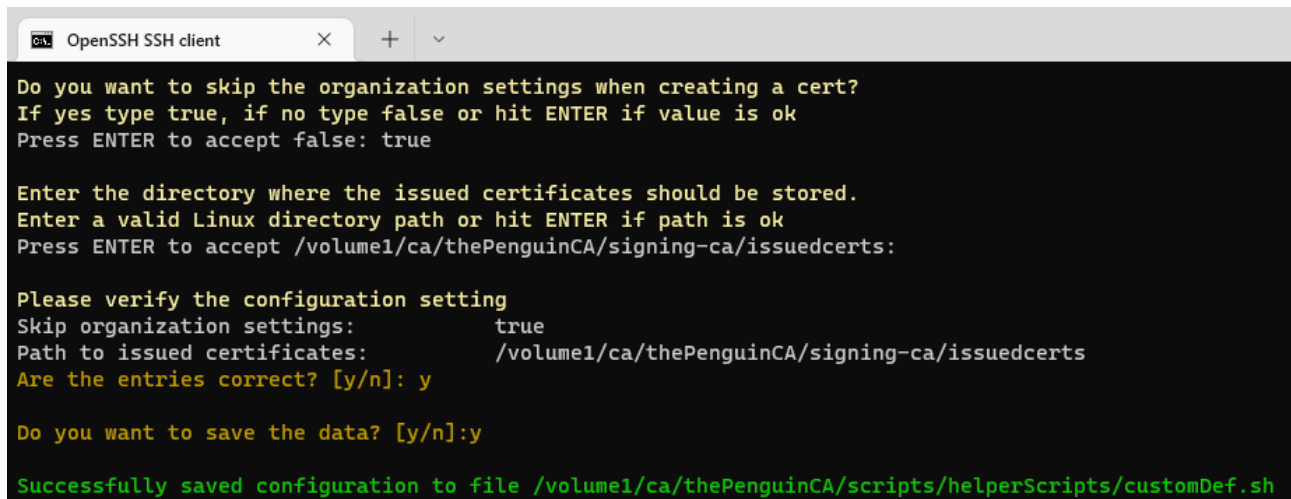
The script allows to configure the following options:

- Allow to change the organizational settings when issuing a certificate
- Custom location storing issued certificates

If you want to skip the confirmation of the organizational settings during the issuing process of a certificate, set the option to *true* (value is case sensitive!).

Per default, the certificates are stored in the *signing-ca* folder under *issuedcerts*. In our example configuration under:

- `/volume1/ca/signing-ca/issuedcerts`



```
OpenSSH SSH client
Do you want to skip the organization settings when creating a cert?
If yes type true, if no type false or hit ENTER if value is ok
Press ENTER to accept false: true

Enter the directory where the issued certificates should be stored.
Enter a valid Linux directory path or hit ENTER if path is ok
Press ENTER to accept /volume1/ca/thePenguinCA/signing-ca/issuedcerts:

Please verify the configuration setting
Skip organization settings:      true
Path to issued certificates:      /volume1/ca/thePenguinCA/signing-ca/issuedcerts
Are the entries correct? [y/n]: y

Do you want to save the data? [y/n]:y

Successfully saved configuration to file /volume1/ca/thePenguinCA/scripts/helperScripts/customDef.sh
```

If you want to store the certificates in a different location, you can configure the path to the location with this script.

Review the configuration and, if OK, confirm with *y*. Confirm with *y* again to save the configuration.

## Certificate Templates

In the subdirectory

- `conf-templates`

under the directory of the *signing-ca* are nine directories, for the certificate configuration templates. In our example installation you can find the templates under

- `/volume1/ca/signing-ca/conf-templates`

Please don't create subdirectories or store files under this path! Please review the templates to verify that the settings fit to your needs.

Please don't change anything in the red surrounded part of the *conf* file.

In the yellow surrounded part, you can change values for:

- the default key size (default value presented when certificate is requested)
- the default validity period in days (default value presented when the certificate is issued)

If there is a requirement, adapt the values in the green surrounded part of the *conf* file.

**Important:** The script was tested with the values in the original certificate templates.

Please don't remove anything from the section `*_dn`. The section is dynamically updated by the script. If you don't want the name of the organizational unit in the certificate, don't delete it from the template. The script will do that for you if you have configured the organizational settings appropriately.

```

Data > ITSData > ITInspiration > Data > _CreateCA > CA-Synology > _deployScript > confTemplates > TLS-server-san > TLS-server-san_req.conf
1  # TLS server certificate request
2  # conf file is based on files from the following post
3  # https://pki-tutorial-ng.readthedocs.io/en/latest/advanced/index.html
4  # request TSL server certificate
5  #cn;san
6  #server_ext
7  #vars for CN
8  def_CountryCode      = __countryCode_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3      # 2 letter country code in cn
9  def_Province         = __province_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3         # province name in cn
10 def_Location         = __location_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3         # location name in cn
11 def_OU               = __OU_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3             # organization unit name in cn
12 def_Organization     = __organization_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3     # organization name in cn
13 def_CommonName       = __cn_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3           # company name in cn
14 def_crlBaseDir       = __CRLbaseURL_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3       # base URL CRL
15 def_aiaBaseDir       = __AIAbaseURL_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3       # base URL AIA
16 def_ca_aiaFile       = __AIAFileName_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3     # aia file name
17 def_ca_crlFile       = __crlName_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3       # name of .crl file
18 def_privKeyEncrypt    = __privKeyEncrypt_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3   # encrypt priv key (pass phrase)
19 def_certPrivateKeyLength = __certPrivateKeyLength_replaceThis-2f56e75f-4e09-4370-bb8c-e32b747170e3 # priv key length (bit)
20 def_aia_url          = $aiaBaseDir/$ca_aiaFile                                         # aia URL
21 def_crl_url          = $crlBaseDir/$ca_crlFile                                         # crl URL
22 #default_bits         = 2048                                                           # RSA key size
23 #default_days         = 365                                                            # Default certificate validity period
24
25 [ req ]
26 default_bits         = $certPrivateKeyLength # RSA key size
27 encrypt_key          = $privKeyEncrypt       # Protect private key
28 default_md           = sha256               # MD to use
29 utf8                 = yes                  # Input is UTF-8
30 string_mask          = utf8only             # Emit UTF-8 strings
31 prompt               = no                   # Prompt for DN
32 distinguished_name   = server_dn            # DN template
33 req_extensions       = server_ext           # Desired extensions
34
35 [ server_dn ]
36 countryName          = $def_CountryCode
37 stateOrProvinceName  = $def_Province
38 localityName         = $def_Location
39 organizationName     = $def_Organization
40 organizationalUnitName = $def_OU
41 commonName           = $def_CommonName
42
43 [ server_ext ]
44 keyUsage              = critical,digitalSignature,keyEncipherment
45 extendedKeyUsage      = serverAuth,clientAuth
46 subjectKeyIdentifier  = hash
47

```

**Important:** Only values can be changed, as described above. If you change something else, the script will fail.

**Important:** Change only values not starting with a dollar sign. Don't change the values for

- default\_bits
- encrypt\_key

in the *req* section. The values starting with a dollar sign represents variables.

## Storing Issued Certificates

Per default, issued certificates are stored in the directory

- `issuedcerts`

in the directory structure of the issuing CA. In our example under

- `/volume1/ca/issuing-ca/issuedcerts`

For every certificate issued, a directory is created. The directory is named like the common name in the certificate. Under this directory you can find the following files:

- `<certificate name>.crt`, the certificate
- `<certificate.name>.pem`, the converted crt file
- `<certificate.name>.key`, the private key of the certificate
- `<certificate name>.p12`, optional file if PKCS#12 bundle was created
- `HEX#.pem`, pem file of the certificate copied from the *newcerts* directory

In the subdirectory with the name

- `conf`

you can find the *conf* and *csr* files for the certificate request.

If a certificate with the same common name, of an already issued certificate is requested, the directory of the older certificate is moved to the subdirectory

- `_archive`

and renamed. A sequential number is added to the directory name. For our example:

- A certificate with the common name [www.the-penguin.lnx](http://www.the-penguin.lnx) is requested
- An older version of a certificate with the same common name exists
- The directory of the older version is moved to the subdirectory `_archive` and renamed to [www.the-penguin.lnx\\_1](http://www.the-penguin.lnx_1)

**Note:** The *conf* files for the CA's allow to issue multiple certificates with the same common name. If this setting is not adequate for your environment, you can change it. To adapt the behavior change the entry from:

- `unique_subject = no`

to

- `unique_subject = yes`

in the file

- `signing-ca.db.attr`

in the subdirectory `db` of the signing CA directory. In our example modify the file

- `/volume1/ca/signing-ca/db/signing-ca.db.attr`

accordingly.

If a unique subject name is enforced, the request will fail, if a duplicate subject name is detected.

**Important:** Even if the request fails, the certificate with the same name is moved to the

- `_archive`

folder.