# Inhaltsverzeichnis

***Neues CA Zertifikat erzeugen*** ..... 3
***neues Server Zertifikat erstellen*** ..... 3
***sub-CA Zertifikat erstellen*** ..... 4
***.p12 Datei erzeugen*** ..... 5
***CodeSigning*** ..... 5

# Zertifiats Authority

## Neues CA Zertifikat erzeugen

- https://help.univention.com/t/how-to-extend-the-end-date-of-the-ucs-ca-root-certificate/9740
- https://help.univention.com/t/renewing-the-ssl-certificates/37
- https://help.univention.com/t/add-subject-alternative-names-to-existing-certificate/7433
- https://help.univention.com/t/cool-solution-creation-and-management-of-user-and-windows-certificates/11782

## neues Server Zertifikat erstellen

Ausrechnen wie lange das Zertifikat laufen darf, am besten das Ablaufdatum der CA verwenden.

```
echo \(`date -d 18-Oct-2027 +'%s'` - `date +'%s'`\) /86400 |bc

declare -x ServerName=FQHN
declare -x days=$(echo \(`date -d 18-Oct-2027 +'%s'` - `date +'%s'`\) /86400
|bc)
```

Somit sind die Anzahl Tage und der Servername in Variablen gespeichert und können im folgenden benutzt werden.

```
. /usr/share/univention-ssl/make-certificates.sh
univention-certificate new -name "${ServerName}" -days $days
cd /etc/univention/ssl/${ServerName}
nano openssl.cnf
```

Ändern oder Anpassen der SAN auf ca Zeile 103. Die Zeile enthält bereits `subjectAltName = DNS:${ServerName}, …` den FQDN und den einfachen Hostnamen. Es können kommagetrennt weitere SAN angegeben werden, wie z.B. `IP:1.2.3.4` oder `DNS:ein.anderer.hostname.domain`.

Es kann auch in der Sektion `[ v3_req ]` eine Linie mit `extendedKeyUsage = serverAuth`

angehängt werden, damit das Zertifikat vom Host auch für RDS Verbindungen benutzt werden kann. Dann die Datei `openssl.cnf` speichern.

```
openssl req -new -key private.key -config openssl.cnf -out req.pem
openssl req -in req.pem -noout -text | grep -E "(Subject Alternative
Name|DNS)"
univention-certificate renew -name ${ServerName} -days $days
```

das ganze kann dann in eine pkcs12 Datei verpackt werden, damit sie z.B. unter Windows importiert werden kann:

```
openssl pkcs12 -export -out /root/${ServerName}.p12 \
-in /etc/univention/ssl/${ServerName}/cert.pem \
-inkey /etc/univention/ssl/${ServerName}/private.key \
-passout pass:EinSehrSicheresPasswort
```

Zertifikatsfingerabdruck auslesen und auf dem Terminalserver via `wmic /namespace:rootcimv2TerminalServices PATH Win32_TSGeneralSetting Set SSLCertificateSHA1Hash="Fingerabdruck"` das Zertifikat für RDP verwenden.

# sub-CA Zertifikat erstellen

```
. /usr/share/univention-ssl/make-certificates.sh
declare -x ServerName=FQHN
univention-certificate new -name "${ServerName}" -days 1825
cd /etc/univention/ssl/${ServerName}
nano openssl.cnf
```

Ändern der `[ v3_req ]` Sektion, um Zeile 100:

```
[ v3_req ]
basicConstraints        = critical, CA:TRUE
keyUsage                = cRLSign, keyCertSign
nsCertType              = sslCA, emailCA, objCA
subjectAltName          = email:copy
#issuerAltName           = issuer:copy
nsComment               = This is a Fake Root CA Certificate
```

Bevor der Request unterschrieben werden kann muss das `/etc/univention/ssl/openssl.cnf` File angepasst werden. Diese Änderung unbedingt wieder rückgängig machen nach dem signieren des Requests! Ungefährt Zeile 100 ebenfalls in der `[ v3_req ]` Sektion

```
[ v3_req ]

basicContraints = critica, CA:TRUE
keyUsage                = cRLSign, keyCertSign
#basicConstraints = critical, CA:FALSE
#keyUsage = nonRepudiation, digitalSignature, keyEncipherment
```

Danach den Zertifikatsrequest mit `openssl req -new -config openssl.cnf -key private.key -out req.pem` erstellen und dann von der CA unterschreiben lassen: `univention-certificate renew -name ${ServerName} -days 1825`

## .p12 Datei erzeugen

```
export ServerName=my.domain.name
openssl pkcs12 -export -out /root/${ServerName}.p12 -in
/etc/univention/ssl/${ServerName}/cert.pem -inkey
/etc/univention/ssl/${ServerName}/private.key  -passout
pass:xZtGhgBBt635mTapXzK
```

Quelle

## CodeSigning

To be recognized as a CodeSigning Certificate, it is needed to set also the following extended Key Usage 'EKU' Property: Extended Use key flag -eku `1.3.6.1.5.5.7.3.3` so the cert can be used for code signing by Powershell.

To generate a CodeSigning Cert with the Univention CA, follow these steps:

1. create a new cert by using univention-certificate new, use a name you recognize as CS Cert
2. create a special extension file
3. generate the cert again manually by the CA with the extension
4. use this cert for signing

```
declare -x CertName=CodeSign-YourName
declare -x ExportPassword=SuperSecurePasswordForP12File
declare -x days=$(echo \(`date -d 18-Oct-2027 +'%s'` - `date +'%s'`\) /86400
|bc)

. /usr/share/univention-ssl/make-certificates.sh
univention-certificate new -name "${CertName}" -days ${days}
cd /etc/univention/ssl/${CertName}

echo "
authorityKeyIdentifier = keyid,issuer
basicConstraints       = CA:FALSE
subjectAltName         = @alt_names
extendedKeyUsage       = codeSigning
[alt_names]
DNS.1 = ${CertName}
" > code_sign_cert.conf

grep output_password /etc/univention/ssl/openssl.cnf
openssl x509 -req -CA ../ucsCA/CAcert.pem -CAkey ../ucsCA/private/CAkey.pem
-in req.pem -out cert.pem -days ${days} -CAcreateserial -extfile
```

```
code_sign_cert.conf

openssl pkcs12 -export -out /root/${CertName}.p12 -in cert.pem -inkey
private.key -passout pass:${ExportPassword}
```

created with thanks to the infos found here:
https://stackoverflow.com/questions/72207572/how-to-create-a-self-signed-code-signing-certificate-from-a-csr

From:
https://aha-it.ch/wiki/ - **AHa-IT**

Permanent link:
**https://aha-it.ch/wiki/lx/ucs/ca**

Last update: **25.07.2026 22:13**