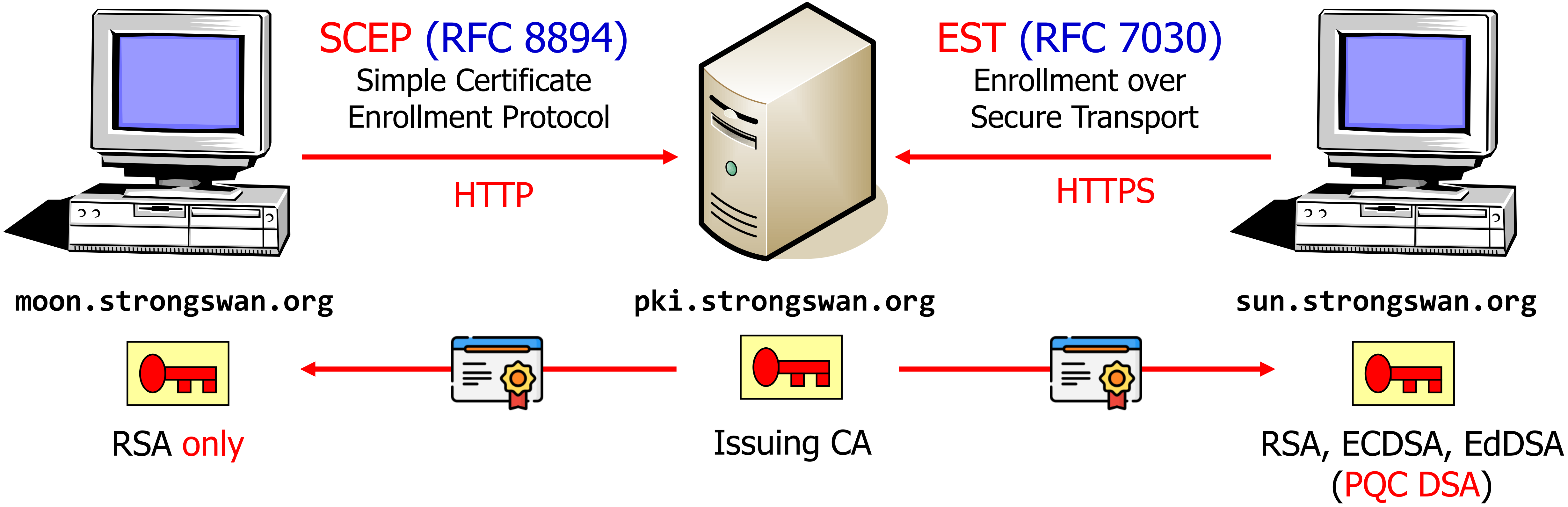
X.509 Certificate Enrollment

Andreas Steffen

andreas.steffen@strongswan.org

X.509 Certificate Enrollment Scenario



SCEP (RFC 8894)
Simple Certificate
Enrollment Protocol

HTTP

EST (RFC 7030)
Enrollment over
Secure Transport

HTTPS

moon.strongswan.org

pki.strongswan.org

sun.strongswan.org

RSA only

Issuing CA

RSA, ECDSA, EdDSA
(PQC DSA)

Private Key could also be generated and stored on a smartcard or in a TPM 2.0.

Extension of the strongSwan pki Tool

- `pki --scepca` # Get CA [and RA] certificate[s] from a SCEP server
- `pki --estca` # Get CA certificate[s] from an EST server
- `pki --scep` # Enroll or Re-enroll an X.509 certificate with a SCEP server
- `pki --est` # Enroll or Re-enroll an X.509 certificate with an EST server
- `cert-enroll` # Shell script for daily X.509 certificate validity checking and automatic certificate re-enrollment based on `pki` tool
- `pki --ocsp` # Implements an OCSP Responder (with `openxpki` plugin)

available since strongSwan 5.9.8 – complete with 5.9.12

pki --scepca Command

```
pki --scepca --url http://pki.strongswan.org/scep \
             --caout myca.crt --raout myra.crt --outform pem
```

```
Root CA cert "C=CH, O=strongSwan Project, CN=strongSwan Root CA"
  serial: 65:31:00:ca:79:da:16:6b:aa:ac:89:e2:a8:f9:49:c3:10:ab:64:54
  SHA256: 96:70:50:51:...:bf:dd:be:86
Root CA cert is untrusted, valid until Aug 12 15:51:34 2032, 'myca.crt'
Sub CA cert "C=CH, O=strongSwan Project, CN=strongSwan Issuing CA"
  serial: 74:f9:7e:72:7d:b8:fd:f2:c6:e5:1b:fa:37:f9:cb:87:bf:9c:ea:e2
  SHA256: a3:5b:4b:12:..:6a:8c:07:bf
Sub CA cert is trusted, valid until Aug 12 15:51:34 2027, 'myca-1.crt'
RA cert "C=CH, O=strongSwan Project, CN=SCEP RA"
  serial: 74:f9:7e:72:7d:b8:fd:f2:c6:e5:1b:fa:37:f9:cb:87:bf:9c:ea:e3
  SHA256: 57:22:f3:13:...:db:bb:64:df
RA cert is trusted, valid until Aug 10 15:51:34 2023, 'myra.crt'
```

see https://docs.strongswan.org/docs/5.9/pki/pkiScepCa.html

pki --estca Command

```
pki --estca --url https://pki.strongswan.org --cacert tlsca.crt \
            --caout myca.crt --outform pem
```

```
negotiated TLS 1.3 using suite TLS_AES_256_GCM_SHA384
received TLS server certificate 'C=CH, O=strongSwan Project, CN=pki.strongswan.org'
Root CA cert "C=CH, O=strongSwan Project, CN=strongSwan Root CA"
  serial: 65:31:00:ca:79:da:16:6b:aa:ac:89:e2:a8:f9:49:c3:10:ab:64:54
  SHA256: 96:70:50:51:...:bf:dd:be:86
Root CA equals trusted TLS Root CA
Root CA cert is trusted, valid until Aug 12 15:51:34 2032, 'myca.crt'
Sub CA cert "C=CH, O=strongSwan Project, CN=strongSwan Issuing CA"
  serial: 74:f9:7e:72:7d:b8:fd:f2:c6:e5:1b:fa:37:f9:cb:87:bf:9c:ea:e2
  SHA256: a3:5b:4b:12:...:6a:8c:07:bf
Sub CA cert is trusted, valid until Aug 12 15:51:34 2027, 'myca-1.crt'
```

see https://docs.strongswan.org/docs/5.9/pki/pkiEstCa.html

pki --scep Command

```
pki --scep --url http://pki.strongswan.org/scep --in moonKey.pem \
           --cacert-enc myra.crt --cacert-sig myca-1.crt --cacert myca.crt \
           --dn "C=CH, O=strongSwan Project, CN=moon.strongswan.org" \
           --san moon.strongswan.org --profile dual --outform pem > moonCert.pem
```

```
transaction ID: 4DFCF31CB18A9B5333CCEC6F99CF230E4524E334
SCEP request pending, polling indefinitely every 60 seconds
  going to sleep for 60 seconds
transaction ID: 4DFCF31CB18A9B5333CCEC6F99CF230E4524E334
  ...
  going to sleep for 60 seconds
Issued certificate "C=CH, O=strongSwan Project, CN=moon.strongswan.org"
  serial: 1e:ff:22:7b:6e:d7:4c:c1:8a:06
Issued certificate is trusted, valid from Aug 22 18:56:23 2022 until Aug 22 18:56:23 2023
```

see https://docs.strongswan.org/docs/5.9/pki/pkiScep.html

pki --est Command

```
pki --req --in sunKey.pem --type ecdsa
          --dn "C=CH, O=strongSwan Project, CN=sun.strongswan.org" \
          --san sun.strongswan.org --profile dual --outform pem > sunReq.pem

pki --est --url https://pki.strongswan.org/ --in sunReq.pem \
          --cacert tlsca.crt --cacert myca.crt --cacert myca-1.crt \
          --outform pem > sunCert.pem
```

```
negotiated TLS 1.3 using suite TLS_AES_256_GCM_SHA384
received TLS server certificate 'C=CH, O=strongSwan Project, CN=pki.strongswan.org'
EST request pending, polling indefinitely every 300 seconds
  going to sleep for 300 seconds
  ...
Issued certificate "C=CH, O=strongSwan Project, CN=sun.strongswan.org"
  serial: 1a:ff:de:66:d9:38:ea:d5:b6:da
Issued certificate is trusted, valid from Aug 22 15:19:43 2022 until Aug 22 15:19:43 2023
```

see https://docs.strongswan.org/docs/5.9/pki/pkiEst.html
https://docs.strongswan.org/docs/5.9/pki/pkiReq.html

X.509 Certificate Re-Enrollment

```
pki --scep --url http://pki.strongswan.org/scep --in moonKeyNew.pem \
           --cacert-enc myra.crt --cacert-sig myca-1.crt --cacert myca.crt \
           --san moon.strongswan.org --profile dual \
           --key moonKey.pem --cert moonCert.pem --outform pem > moonCertNew.pem

pki --req --in sunKeyNew.pem --type ecdsa --oldreq sunReq.pem \
          --outform pem > sunReqNew.pem

pki --est --url http://pki.strongswan.org/ --in sunReqNew.pem \
          --cacert tlsca.crt --cacert myca.crt --cacert myca-1.crt \
          --key sunKey.pem --cert sunCert.pem --outform pem > sunCertNew.pem
```

The fresh certificate is automatically issued by the PKI on the basis of the old certificate’s subject and the signature with the old private key.

cert-enroll Shell Script - systemd timer

```
cert-enroll.timer

[Unit]
Description=daily check of the remaining X.509 certificate lifetime
Documentation=man:cert-enroll(8)

[Timer]
# The cert-enroll script should be run once a day.
OnCalendar=*-*-* 02:00:00
RandomizedDelaySec=7200
Persistent=true

[Install]
WantedBy=timers.target
```

If systemd is not available on the host, the timer can be based on crontab instead

cert-enroll Shell Script - systemd service

```
cert-enroll.service

[Unit]
Description=X.509 certificate checking (re-enrollment if necessary)
Documentation=man:cert-enroll(8)

[Service]
Type=oneshot
User=root
ExecStart=/usr/sbin/cert-enroll
SuccessExitStatus=1
```

```
root@sun.strongswan.org:~# ls /root/certificates/
cacert-1.pem  cacert.pem  cert.pem  key.pem  new  old  req.pem
```

cert-enroll Shell Script - systemd journal

```
Sep 08 02:02:06 sun.strongswan.org cert-enroll[12729]:
    changed into the '/root/certificates' directory
    warning: validity of 'cert.pem' is only 29 days, less than the minimum of 42 days
    generated 256 bit ECDSA private key 'new/key.pem’
    negotiated TLS 1.3 using suite TLS_AES_256_GCM_SHA384
    ...
    downloaded CA certificates via EST
    negotiated TLS 1.3 using suite TLS_AES_256_GCM_SHA384
    ...
    Issued certificate is trusted, valid from Sep 08 02:02:06 2023 until
                                              Sep 08 02:02:06 2027 (currently valid)
    re-enrolled 'cert.pem' via EST
    replaced old 'key.pem' and 'cert.pem'
```

```
Sep 09 03:17:36 sun.strongswan.org cert-enroll[13560]:
    ok: validity of 'cert.pem' is 1459 days, more than the minimum of 42 days
```

pki --ocsp Command used for OCSP Responder

```
#!/bin/bash

cd /etc/ocsp
echo "Content-type: application/ocsp-response"
echo "“
cat | openssl ocsp -index index.txt -CA strongSwanIssuingCA.pem     \
                   -rkey ocspKey.pem -rsigner ocspCert.pem -nmin 10 \
                   -reqin /dev/stdin -respout /dev/stdout | cat
```

- `openssl ocsp` chokes on multiple non-revoked certificate entries in index.txt having the same subjectDistinguishedName.
- A periodic `crontab` job (every 10 minutes) has to extract the content of the OpenXPKI certificate database and convert it into the OpenSSL index.txt format.
- `pki --ocsp` will be able to verify the certificate status directly via a query into the OpenXPKI database using the new `openxpki` plugin.

Thank you for your attention!

Questions?