NAME

pki --est - Enroll an X.509 certificate with an EST server

SYNOPSIS

pki --est---url *url* [**--label** *label*] [**--in** *file*] **--cacert** *file* [**--cert** *file*|**--certid** *hex* **--key** *file*|**--keyid** *hex*] [**--userpass** *username:password*] [**--interval** *time*] [**--maxpolltime** *time*] [**--outform** *encoding*] [**--debug** *level*]

pki --est--options *file*

pki --est-h | **--help**

DESCRIPTION

This sub-command of **pki**(1) sends a PKCS#10 certificate request via HTTPS to a server using the Enrollment over Secure Transport (EST) Protocol (RFC 7030). After successful authorization which with manual authentication requires periodic polling by the enrollment client, the EST server returns an X.509 certificate signed by the CA.

Before the expiry of the current certificate, a new client certificate based on a fresh private key can be requested, using the old certificate and the old key for automatic TLS client authentication with the EST server.

OPTIONS

-h, --help

Print usage information with a summary of the available options.

-v, --debug *level*

Set debug level, default: 1.

-+, --options *file*

Read command line options from *file*.

-u, --url *url*

URL of the EST server.

-l, --label *label*

Label in the EST server path.

-i, --in *file*

PKCS#10 certificate request. If not given, the certificate request is read from *STDIN*.

-C, --cacert *file*
CA certificate in the trust chain used for EST TLS server signature verification or in the trust chain to verify the client certificate issued by the CA. Can be used multiple times.

-c, --cert *file*
Client certificate to be renewed.

-X, --certid *hex*
Smartcard or TPM 2.0 client certificate object handle.

-k, --key *file*
Client private key to be replaced.

-x, --keyid *hex*
Smartcard or TPM 2.0 client private key object handle.

-p, --userpass *username:password*
Optional username:password that may be used for HTTP basic authentication.

-t, --interval *time*
Poll interval in seconds, defaults to *60s*. This value might get overridden by the **retry-after** header in the HTTP 202 reply from the EST server.

-m, --maxpolltime *time*
Maximum poll time in seconds, defaults to *0* which means unlimited polling.

-f, --outform *encoding*
Encoding of the created certificate file. Either *der* (ASN.1 DER) or *pem* (Base64 PEM), defaults to *der*.

EXAMPLES

To save some typing work the following command line options are stored in a *est.opt* file:

```
--url https://pki.strongswan.org:8443
--cacert tlsca.crt
--cacert tlsca-1.crt
--cacert myca.crt
--cacert myca-1.crt
```

NOTE: For a successful HTTPS connection, trust must be established into the EST server certificate.

The TLS trust chain including the root CA certificate and optionally intermediate CA certificates must be given using multiple **--cacert** options.

The **--cacert** option must also be used to be able to verify the received client certificate issued by the CA. This second trust chain might be identical to the TLS trust chain (if the EST server is using a TLS server certificate issued by its own CA) or might be totally different, e.g. if a Let’s Encrypt EST server certificate is used.

With the following command, an X.509 certificate signed by the intermediate CA is requested from an EST server based on a PKCS#10 certificate request:

```
pki --options est.opt --in moonReq.der > moonCert.der

negotiated TLS 1.3 using suite TLS_AES_256_GCM_SHA384
received TLS server certificate ’C=CH, O=strongSwan Project, CN=pki.strongswan.org’
  using certificate "C=CH, O=strongSwan Project, CN=pki.strongswan.org"
  using trusted intermediate ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Issuing CA"
  using trusted ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Root CA"
  reached self-signed root ca with a path length of 1
  EST request pending, polling indefinitely every 300 seconds
  going to sleep for 300 seconds
  ...
Issued certificate "C=CH, O=strongSwan Project, CN=moon.strongswan.org"
  serial: 1a:ff:de:66:d9:38:ea:d5:b6:da
  using certificate "C=CH, O=strongSwan Project, CN=moon.strongswan.org"
  using trusted intermediate ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Issuing CA"
  using trusted ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Root CA"
  reached self-signed root ca with a path length of 1
Issued certificate is trusted, valid from Aug 22 15:19:43 2022 until Aug 22 15:19:43 2023 (currently valid)
```

This certificate can be renewed some time before it expires with the command:

```
pki --options est.opt --in moonReqNew.der --cert moonCert.der --key moonKey.der > moonCertNew.der

negotiated TLS 1.3 using suite TLS_AES_256_GCM_SHA384
received TLS server certificate ’C=CH, O=strongSwan Project, CN=pki.strongswan.org’
  using certificate "C=CH, O=strongSwan Project, CN=pki.strongswan.org"
  using trusted intermediate ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Issuing CA"
  using trusted ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Root CA"
  reached self-signed root ca with a path length of 1
```

```
sending TLS client certificate ’C=CH, O=strongSwan Project, CN=moon.strongswan.org’
sending TLS intermediate certificate ’C=CH, O=strongSwan Project, CN=strongSwan Issuing CA’
Issued certificate "C=CH, O=strongSwan Project, CN=moon.strongswan.org"
  serial: 1b:ff:ad:dc:2f:50:c4:cb:a1:44
  using certificate "C=CH, O=strongSwan Project, CN=moon.strongswan.org"
  using trusted intermediate ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Issuing CA"
  using trusted ca certificate "C=CH, O=strongSwan Project, CN=strongSwan Root CA"
  reached self-signed root ca with a path length of 1
Issued certificate is trusted, valid from Jul 20 12:21:00 2023 until Jul 20 12:21:00 2024 (currently valid)
```

If the private key and the certificate of the client is stored in a TPM 2.0, the renewal can be done with the following options:

```
pki --options est.opt --in moonReqNew.der --certid 0x01800004 --keyid 0x81010004 > moonCertNew.der
```

SEE ALSO

pki(1)