mkcert Local trusted dev certs

CONCEPTS

mkcert makes a private CA on your machine, trusts it, and signs certificates with it. Browsers then accept your dev certs with no warning. Never share rootCA-key.pem: whoever has it can impersonate any site to you. Dev use only, never production.

mkcert -install CA in CAROOT
↓ trusts
Trust stores system, NSS, Java
↓ signs
Leaf cert + key your hostnames
One CA, many leaf certificates

Put flags before the names, otherwise they are ignored. mkcert does not configure your server: point it at the generated .pem and -key.pem files yourself.

SETUP

mkcert -install
Create local CA and trust it

mkcert -uninstall
Remove CA from trust stores

mkcert -CAROOT
Print CA folder

sudo apt install libnss3-tools
Linux: Firefox/Chrome NSS support

brew install mkcert nss
macOS install (nss for Firefox)

choco install mkcert
Windows install

SETUP (cont.)

TRUST_STORES=system,nss mkcert -install
Limit which stores are used

ISSUE CERTIFICATES

mkcert *<name>* **...**
Cert for one or more names

mkcert example.test localhost 127.0.0.1 ::1
Hostnames plus IPs

mkcert "*.*<domain>***"**
Wildcard certificate

mkcert -cert-file *<crt>* **-key-file** *<key>* *<name>*
Set output file names

mkcert -ecdsa *<name>*
Use an ECDSA key

mkcert -pkcs12 *<name>*
Also produce a .p12 bundle

mkcert -p12-file *<f>* **-pkcs12** *<name>*
Set .p12 output path

mkcert -client *<name>*
Client-authentication cert

mkcert *<user>*@*<domain>*
S/MIME cert for an email address

mkcert -csr *<file>*
Sign an existing CSR

TRUST ELSEWHERE

CAROOT=*<dir>* **mkcert -install**
Install a copied CA on another host

TRUST ELSEWHERE (cont.)

CAROOT=*<dir>* **mkcert** *<name>*
Use a separate CA

NODE_EXTRA_CA_CERTS="$(mkcert -CAROOT)/rootCA.pem"
Make Node.js trust the CA

$(mkcert -CAROOT)/rootCA.pem
Root cert to import on phones

JAVA_HOME=*<jdk>* **mkcert -install**
Trust CA in Java keystore

RECIPES

Local HTTPS for a dev server

```
mkcert -install
mkcert -cert-file dev.pem -key-file dev-key.pem localhost 127.0.0.1 ::1
# point the server at dev.pem and dev-key.pem
```

nginx server block

```
mkcert -cert-file dev.pem -key-file dev-key.pem app.test
# ssl_certificate     /path/dev.pem;
# ssl_certificate_key /path/dev-key.pem;
curl https://app.test
```

Share the CA with a teammate or VM

```
cp "$(mkcert -CAROOT)/rootCA.pem" ./
# on the other machine, with rootCA.pem in <dir>:
CAROOT=<dir> mkcert -install
```

Inspect the result

```
openssl x509 -in dev.pem -noout -subject -issuer -dates -ext subjectAltName
```