

**FSI
LEI**

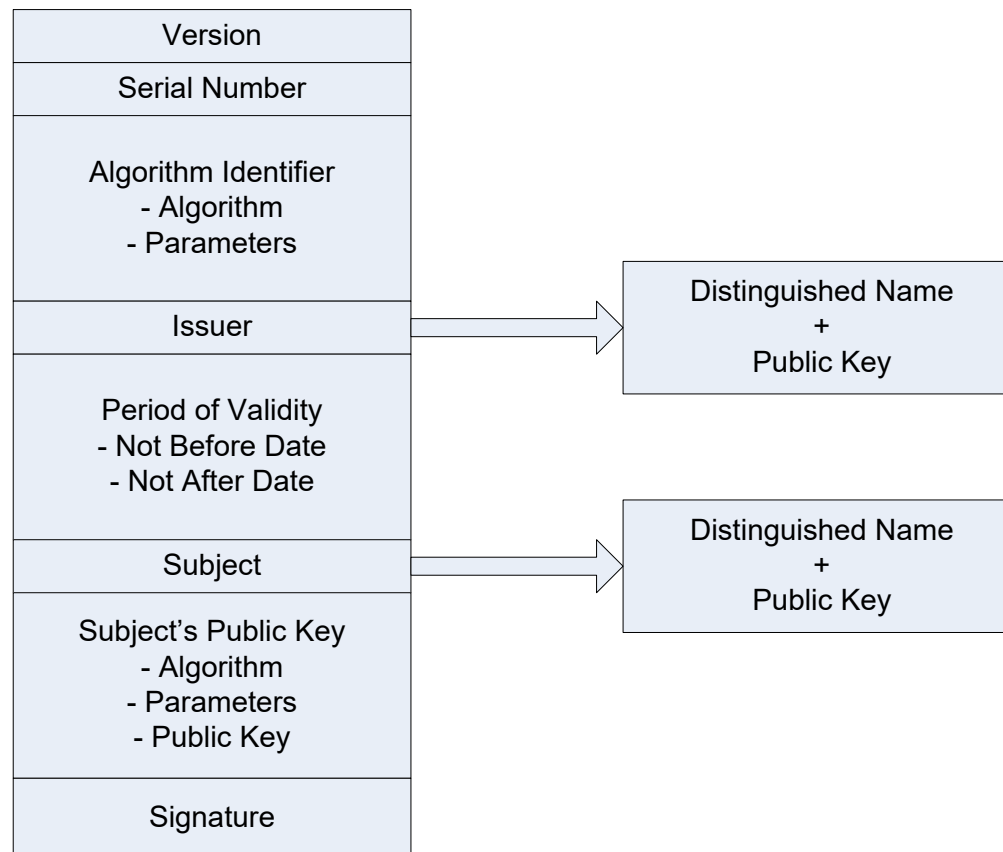
2025/2026

Practical class #5

- **Certification authorities using OpenSSL**
- **Server and client authentication with Apache**

X.509 Certificates

A X.509 certificate contains a public-key and also information about a real entity (Subject)



X.509 Certificates

Information about the entity is stored as a DN (Distinguished Name)

| | | |
|---------------------|----|----------------|
| Common Name | CN | CN = Joao Luis |
| Organization | O | O = UC |
| Organizational Unit | OU | OU = DEI |
| City / Location | L | L =Coimbra |
| State / Province | ST | ST = Coimbra |
| Country | C | C = PT |

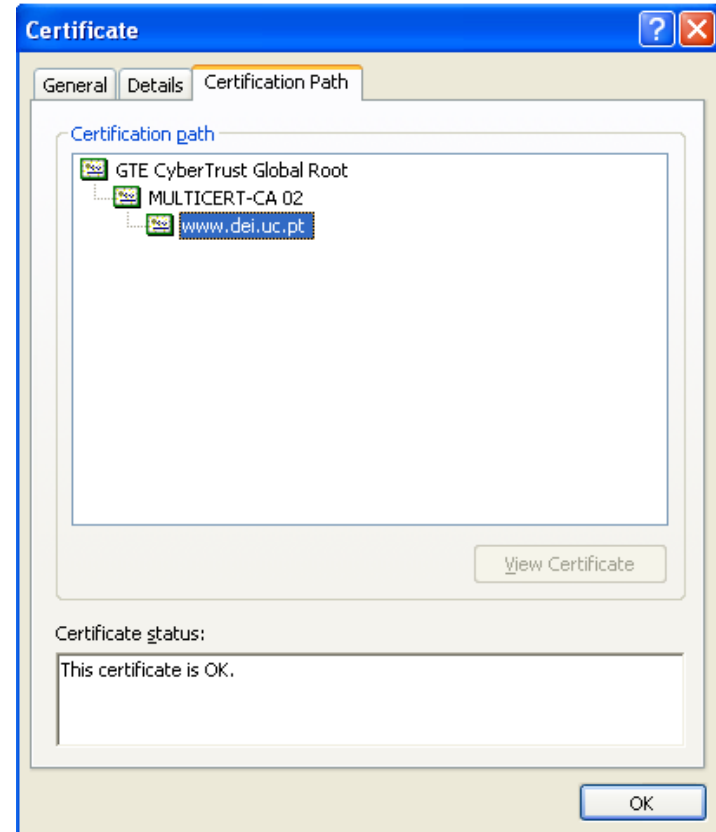
Certification Authorities

- **Verification of CSR (Certificate Signing Request)**
- **Public and Private CA (“self-signed”)**

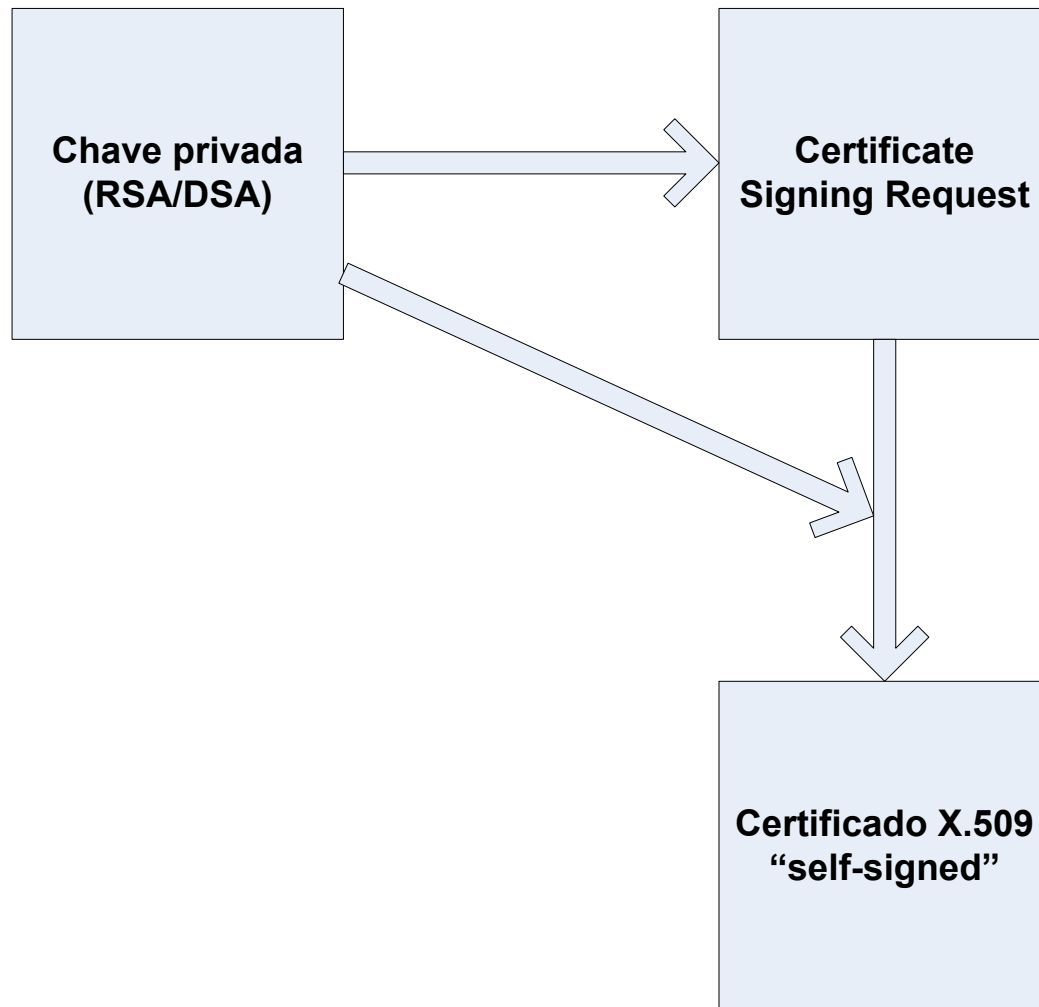
www.verisign.com

www.multicert.pt

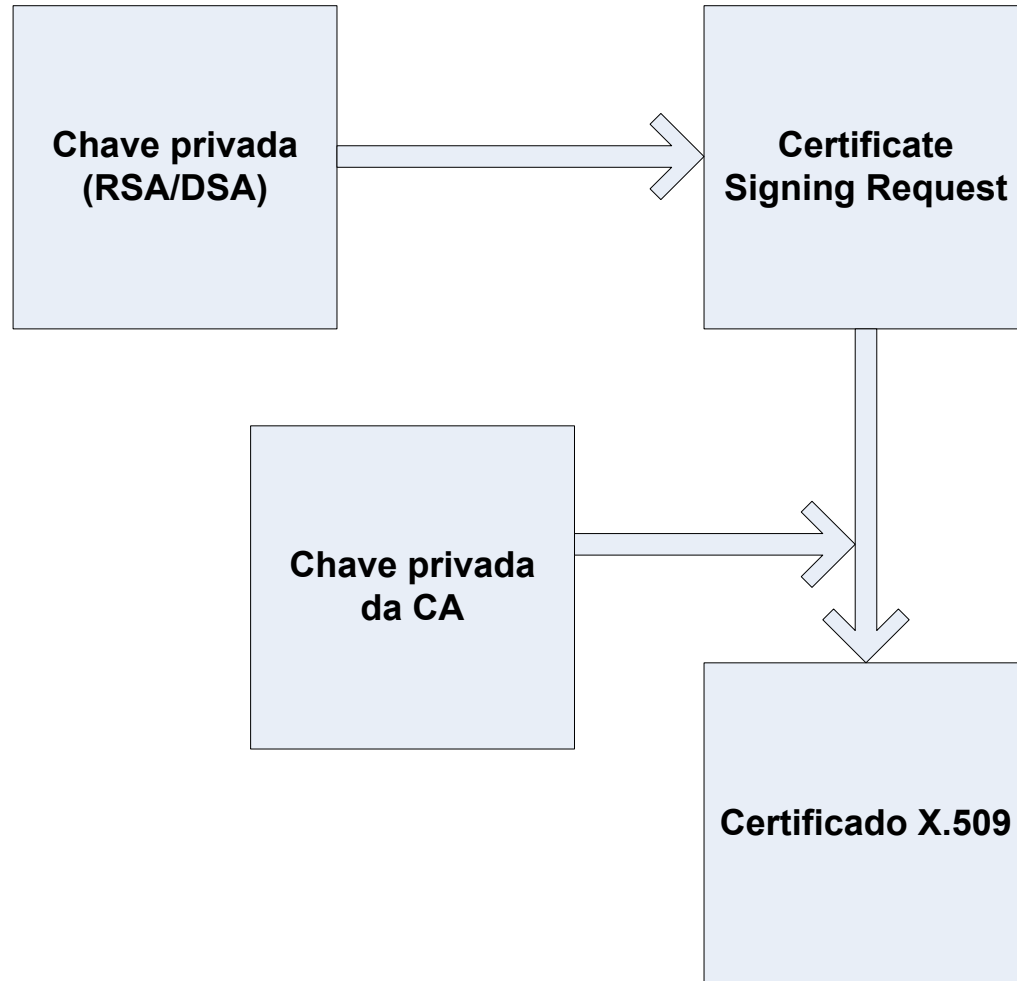
www.thawte.com



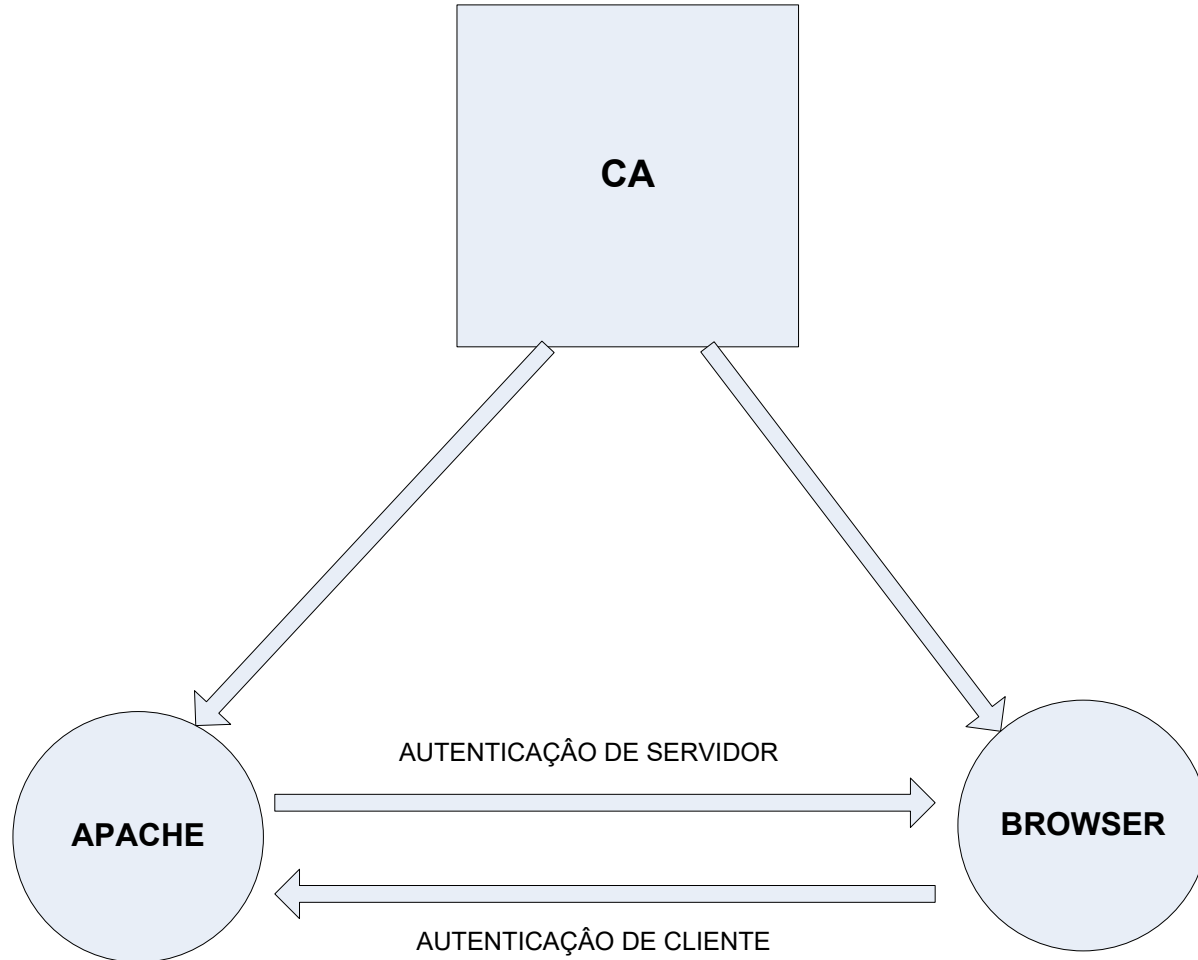
Private CA



Creation of a X.509 certificate



Authentication using X.509 certificates with Apache



Main configuration files (Apache, OpenSSL)

OpenSSL configuration:

`/etc/pki/tls/openssl.cnf`

Apache with SSL (mod_ssl):

`/etc/httpd/conf/httpd.conf`

`/etc/httpd/conf.d/ssl.conf`



APACHE
HTTP SERVER



OpenSSL usage (examples)

Creation of a 1024-bit public-key (RSA) encrypted with 3DES

```
openssl genrsa -out xpto.key -des3 1024
```

Creation of a CSR

```
openssl req -new -key xpto.key -out xpto.csr
```

Creation of a “self-signed” certificate

```
openssl x509 -req -days 365 -in xpto.csr -out xpto.crt -signkey xpto.key
```

Viewing the contents of a certificate

```
openssl x509 -in xpto.crt -text
```

Creation of a x.509 certificate using an existing CA

```
openssl ca -in cert.csr -cert ca.crt -keyfile ca.key -out cert.crt
```

Converting from PEM to PKCS#12

```
openssl pkcs12 -export -clcerts -in xpto.crt -inkey xpto.key -out xpto.p12
```

Converting from PEM to DER

```
openssl x509 -inform PEM -in xpto.crt -outform DER -out xpto.crt.der
```

Creation of the CA

Creation of a private key for the CA

```
openssl genrsa -out ca.key -des3
```

Creation of a CSR for the CA

note: CN = name of the CA

```
openssl req -new -key ca.key -out ca.csr
```

Creation of a “self-signed” certificate to represent the CA

```
openssl x509 -req -days 365 -in ca.csr -out ca.crt -signkey ca.key -extfile v3_ca.ext
```

Viewing the contents of the CA certificate

```
openssl x509 -in ca.crt -text
```

Note: recent browsers require a particular extension so that the CA certificate is recognised as such, this can be defined in the “v3_ca.ext” file (used in the creation):

```
keyUsage = cRLSign, digitalSignature, keyCertSign
```

```
basicConstraints=critical,CA:true,pathlen:0
```

Creation of a certificate for Apache

Creation of a private key for Apache

```
openssl genrsa -out apache.key -des3
```

Creation of a CSR for the Apache certificate

note: CN = domain name of URL, e.g. www.uc.pt

```
openssl req -new -key apache.key -out apache.csr
```

Creation of the certificate for Apache using our private CA

```
openssl ca -in apache.csr -cert ca.crt -keyfile ca.key -out apache.crt -extfile v3.ext
```

Note: recent browsers search for a “subjectAltName” X.509 extension in the server’s certificate. We may add this using a file (“v3.ext” in the example):

```
subjectAltName = @alt_names
```

```
[alt_names]
```

```
DNS.1 = www.uc.pt
```

Creation of a certificate for a User

Creation of a private key for a User

```
openssl genrsa -out user.key -des3
```

Creation of a CSR for the the User certificate

note: CN = name of the user, e.g. John Doe

```
openssl req -new -key user.key -out user.csr
```

Creation of the certificate for the User using our private CA

```
openssl ca -in user.csr -cert ca.crt -keyfile ca.key -out user.crt
```

To import in a browser we need to convert the user certificate and key
to the PKCS#12 format.

```
openssl pkcs12 -export -clcerts -in user.crt -inkey user.key -out user.p12
```