

PV181 Laboratory of security and applied cryptography



Digital certificates

Marek Sýs, Zdeněk Říha



Digital certificate

- used to prove ownership of the public key
- binds public key to identity (identity, email,...)
- **Public key is signed** by trusted third party
 - Certification Authority (CA)
- two models: centralized and decentralized

Certificates history

- Introduced in 1978 – Kohnfelder's bachelor thesis
- X.509 – standard
 - v.1 (1988) – not extendable
 - v.2 – similar to v.1
 - **v.3** (1997) optional extension
 - Other extend X.509

Others

- PGP, SPKI, etc.

Trust models

- Public key infrastructure (PKI)
 - centralized – hierarchy of CA's
 - cert signed by party
 - used in web browsers
 - standard X.509
- Web of trust
 - decentralized model
 - signed by many parties
 - used in PGP, GPG
 - standard OpenPGP

PGP

- 1991 - Paul Zimmerman
- tool to en/decryption, sign data (files, emails,...)
- Algs: hash, asymmetric, symmetric, compression
- Confidentiality, authentication, integrity
- standardized as OpenPGP - RFC 4880
- PGP – commercial
 - free alternative GPG

OpenPGP encryption

- Authentication & integrity & non-repudiation
 - hash of plaintext **signed** by Alice's private key
- Confidentiality
 1. Alice generates randomly symmetric key
 2. Symmetric key is encrypted with Bob's public key
- Alice sends:
 - encrypted signed data
 - encrypted session key

PGP certificates

- Two formats:
 - PGP cert
 - X.509
- PGP cert:
 - PGP version number
 - Public key (+algorithm)
 - Identity information
 - Digital signature -
 - Validity period
 - Preferred symmetric encryption

PGP signing and trust

- Key distribution: server or sent with data
- signing: PGP party – verification of binding
 - KEY-ID, KEYNAME, FINGERPRINT + identity
- Trust:
 - user – set of certs signed by different entities
 - 4 levels – unknown, non, marginal, full
 - signed by myself or fully or 3 marginally keys

Public key Infrastructure (PKI)

- set of roles and procedures:
 - issue, maintain, administer, revoke, suspend, reinstate, and renew digital certificates
 - create and manage a public key repository

PKI:

- CA – stores, issues, signs certs
- RA – verifies identity
- Central directory– cert requests issued and revoked,
- Management system
- Cert policy

X.509 PKI certificate

- Certification Authority – trusted third party
- Certificate revocation lists (CRL) – certificates no longer be trusted (compromised key, CA,...)
- RFC5280 – defines format and semantics of certs and CRLs
- X.509 versions 1,2,3

X.509 PKI certificate content

Serial Number: unique ID of cert

Subject: ID of entity

Signature algorithm:

Signature:

Issuer: verifier of info and issued cert

Valid-From: date cert is first valid from

Valid-To: expiry date

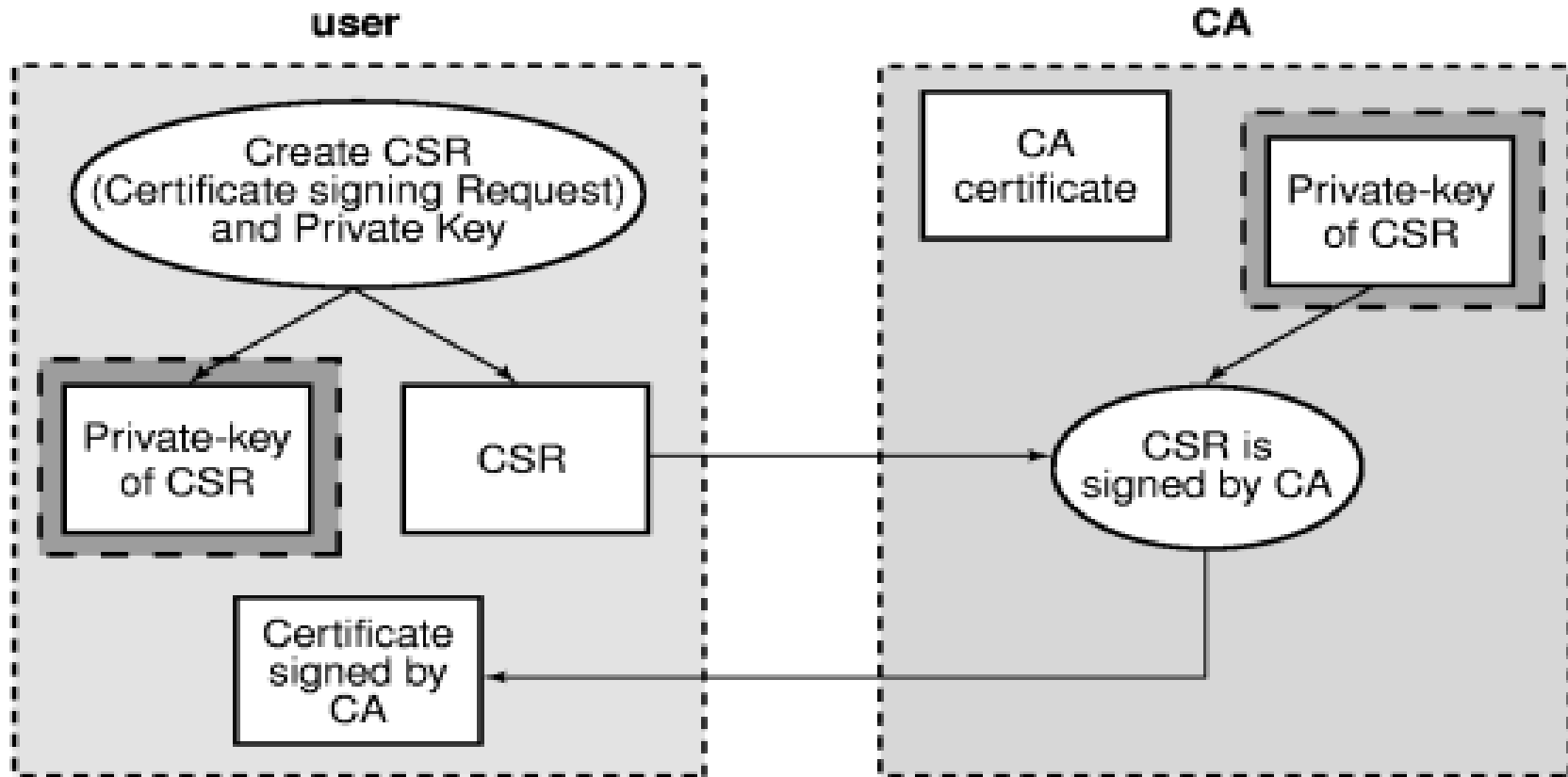
Key-Usage: purpose of PK (signature, cert signing, ...)

Public Key:

Thumbprint algorithm: to compute hash of PK cert

Thumbprint (fingerprint): hash of abbreviated PK cert

Certificate issuing



VM-0908A-AI

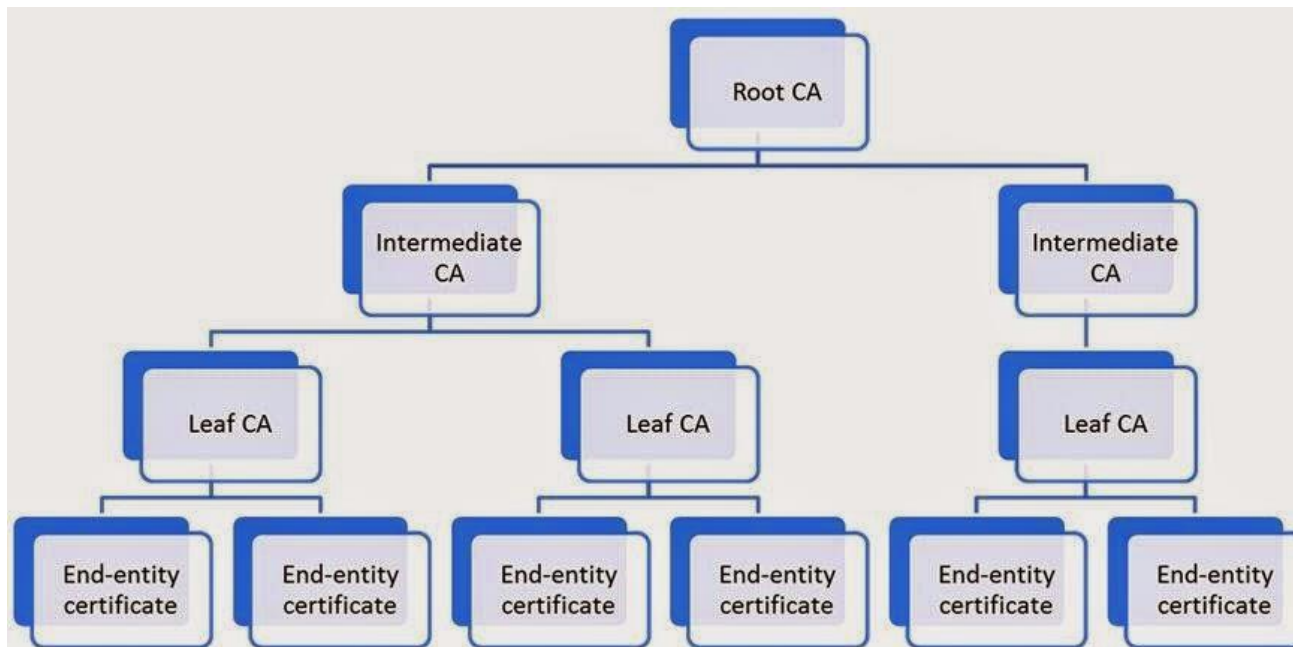
Certificate verification

Checking single cert:

- current **date** against validity period
- current validity of CA public key
- signature of CA on cert
- check whether certificate is revoked
 - CRL or OCSP
- policies

Certificates hierarchy

- **root CA (trust anchor)** - self-signed certificate
- Intermediate CA's
- **End entity** – user certificate



Chain of trust

- Trust transfer – to lower CA's
- Root cert, intermediate certs, end-user cert.
- Chain:
 - end-user cert – signed by CA1
 - CA1 cert – signed by CA2...
 - root CA cert – signed by itself

Server – sends all certs up to root cert to browser

Chain of trust

End-entity Certificate

Owner's name
Owner's public key
Issuer's (CA's) name
Issuer's signature

reference

Intermediate Certificate

Owner's (CA's) name
Owner's public key
Issuer's (root CA's) name
Issuer's signature

reference

sign

sign

self-sign

Root CA's name
Root CA's public key
Root CA's signature

Root Certificate

Certificate path validation

Input: cert path, trust anchor

Path validation:

1. Check all certs if still valid
2. Check revocation status of certs
3. Check issuer = of previous cert subject
4. Check policy constraints
5. ...

Revocation

- Reasons for revocation
 - **key compromise** (most common), CA compromise, affiliation change,...
- Two states:
 - revoked – irreversibly for compromised private key
 - hold – unsure user about key compromising, can be reinstalled
- Checked using:
 - CRL – list of revoked certs
 - Online Certificate Status Protocol – on demand

CRL

Issued by CA:

- Certificate Revocation List (CRL):
 - list of revoked certificates of end-users
 - Authority Revocation list
 - List of revoked cert of CA's
1. Issuer name
 2. Date list created
 3. Date next CRL scheduled
 4. Entries = serial number + revocation date of cert

CRL distribution problem

- **Fixed** validity of CRLs
 - massive load when CRL expires
(e.g. 10M clients download a 1MB CRL issued once a minute
= ~150GB/s traffic)

Solutions:

delta CRL – updates only – newly revoked certs

Online Certificate Status Protocol (OCSP)

- status of cert -OK / revoked / unknown

Time aspect

What if private key is compromised?

How to prove signature was created using secure private key?

- Time - **critical** parameter!!!!
- Time stamping authority – sign time stamp
- Time stamp – $\text{Sig}(\text{H}(\text{H}(\text{data}) \mid \text{time}))$
 - is stored by requester with data
- Standards - RFC 3161, X9.95, ISO 18014