TÉCNICO LISBOA

Management of Public Keys

Segurança Informática em Redes e Sistemas
2021/22

Ack: Miguel Pardal, Ricardo Chaves, Miguel P. Correia,
André Zúquete, Carlos Ribeiro

Roadmap

- Introduction
- Distribution of public keys
- Public-key certificates
- Certificate issuance
- Certificate distribution
- Certificate revocation

Roadmap

- **Introduction**
- Distribution of public keys
- Public-key certificates
- Certificate issuance
- Certificate distribution
- Certificate revocation

Management problems

- Assure correct use
  - Private keys: assure their privacy/confidentiality
  - Public keys: assure their correct distribution
- Evolution of the mapping between entity ↔ key pair:
  - Handle common management operations
    - e.g. key renewal
  - Deal with catastrophic situations
    - e.g. loss of the private key
- Assure unpredictability
  - The generation of asymmetric key pairs must use good random number generators

Management goals

- Key management
  - How and when should the asymmetric keys be generated
- Usage of private keys
  - How is their privacy/confidentiality protected
- Distribution of public keys
  - How are public keys distributed in a correct way
- Key lifetime
  - For how long should the key pairs be used
  - How to check for obsolete keys

Guidelines for key pair generation

- Use good random values generators
  - Able to generate acceptable keys for the targeted ciphering algorithm
    - Unpredictability of all the key bits
    - Equiprobability of all the key bits

- Efficiency without sacrificing security
  - Allow the computation to be accelerated in one of the ciphering directions, without compromising the security

- The key pair – especially the private key – should be generated by its owner
  - To assure the maximum privacy of the private key

Correct usage of private keys

- The private key represents its owner so:
  - The probability of it being compromised must be minimized
  - Backup copies must be physically secure
- Private keys must be protected
  - The access path to the private key must be restricted
  - Password protected, e.g., JKS, PGP
  - Security of applications using the private key must be guaranteed

Private key confinement

- Storage and use of the private key in an autonomous device, e.g., a smartcard
  - The devices generates the key pairs
  - The device ciphers/deciphers the data with the key pair controlled by on-chip access mechanisms
    - e.g. access PIN
  - Allows for qualified signatures
    - EU eIDAS Regulation

Roadmap

- Introduction
- **Distribution of public keys**
- Public-key certificates
- Certificate issuance
- Certificate distribution
- Certificate revocation

Distribution of public keys

- Techniques
  - Manual
    - Not practical
  - Using a shared secret
    - If a shared secret alread

PAST

  - ~~Public announcement~~
  - ~~Public directory~~

PRESENT

  - **Public distribution using digital certificates (next section)**

Roadmap

- Introduction
- Distribution of public keys
- **Public-key certificates**
- Certificate issuance
- Certificate distribution
- Certificate revocation

Public key certificate (digital certificate)

- **Certificates** are documents signed by a certification entity
  - **Certification Authority (CA)**, public organization or company
  - Certificates are public documents
  - Certificates have a digital signature (cryptographic protection)
- Used to distribute public keys through unsecure channels
  - Receiver can validate the certificate signature using the CA public key
  - If it trusts the CA and the signature is valid, then it can trust the public key
- Certificate structure
  - X.509 standard (RFC 3280)
  - PKCS #7 Cryptographic Message Syntax (CMS) standard (RFC 5652)
  - SPKI (Simple Public Key Infrastructure) – historical
  - KeyNote trust-management system – historical

X.509 v3 Digital Certificate (RFC3280)

- Contents
  - Version
  - Serial number (of the CA)
  - Issuer (CA)
  - Validity
    - Not Before
    - Not After
  - Subject
  - Subject Public Key info
    - Public Key Algorithm (ex. RSA)
    - Subject Public Key
  - Extensions (optional)
  - Certificate Signature Algorithm (ex.: RSA w/SHA-256)
  - Certificate Signature Value
- Extensions
  - Issuer Unique Identifier (v2)
  - Subject Unique Identifier (v2)
  - Authority Key Identifier
  - Subject Key Identifier
  - Key Usage
    - digitalSignature
    - nonRepudiation
    - keyEncipherment
    - dataEncipherment
    - keyAgreement
    - keyCertSign
    - CRLSign
    - encipherOnly
    - decipherOnly
  - Extended Key usage
  - CRL Distribution Points
  - Private Key usage period

See a certificate in the browser

Formats & Extensions for X.509

- **.PEM, .CRT, .KEY, etc.** – certificate in textual Base64 format
  - "-----BEGIN CERTIFICATE-----"
  - "-----END CERTIFICATE-----"
  - Most commonly used
- **.DER** – certificate in DER binary format
  - DER – ASN.1 Distinguished Encoding Rules (tag, length, value)
  - **.CER** – set of certificates in the DER format
  - Typically used in Java platforms
- **.P7B and .P7C** – certificate(s) in PKCS#7 textual Base64 format
  - Used in Microsoft Windows
- **.PFX and .P12** - PKCS#12 – binary format
  - Set of certificates and private keys, protected by password
  - Used in Microsoft Windows

Run *locate *.pem* then *cat* some files; same with **.der*

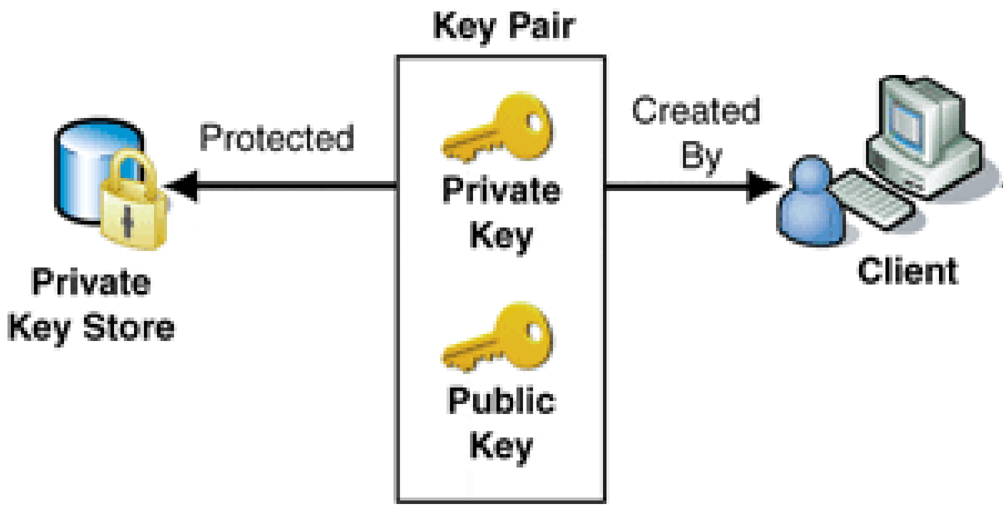
Certification Authorities (CA)

- CAs: organizations that manage certificates
  - Define policies and mechanisms for the generation and distribution of certificates
  - Manage the certificate revocation lists
- Trust in the CAs
  - Manual distribution of their public keys
    - Centralized certification (single CA)
    - Ad-hoc certification (e.g. PGP)
  - Certification hierarchy
    - Public key certificates for the CAs
    - **Manual distribution of root CA public keys, e.g., in web browsers**

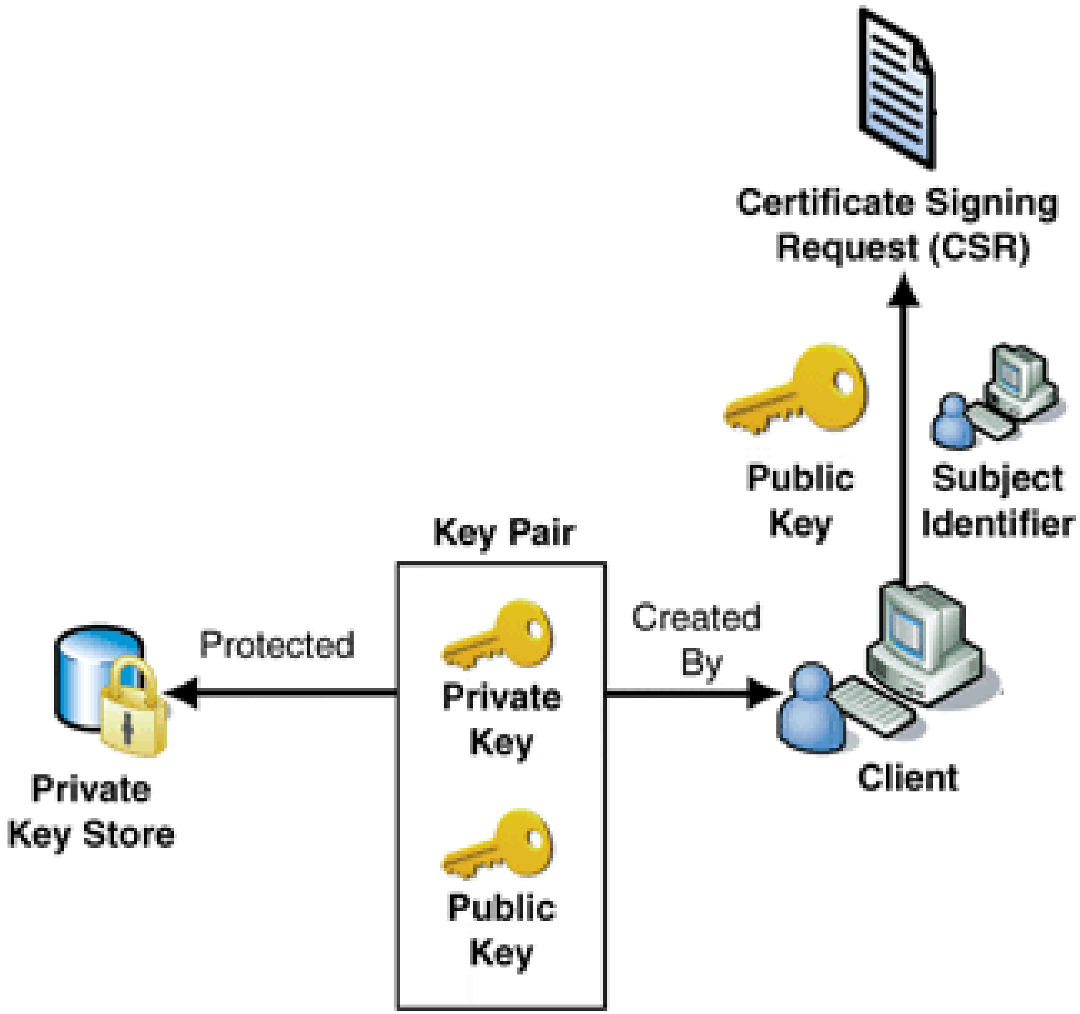
Roadmap

- Introduction
- Distribution of public keys
- Public-key certificates
- **Certificate issuance**
- Certificate distribution
- Certificate revocation

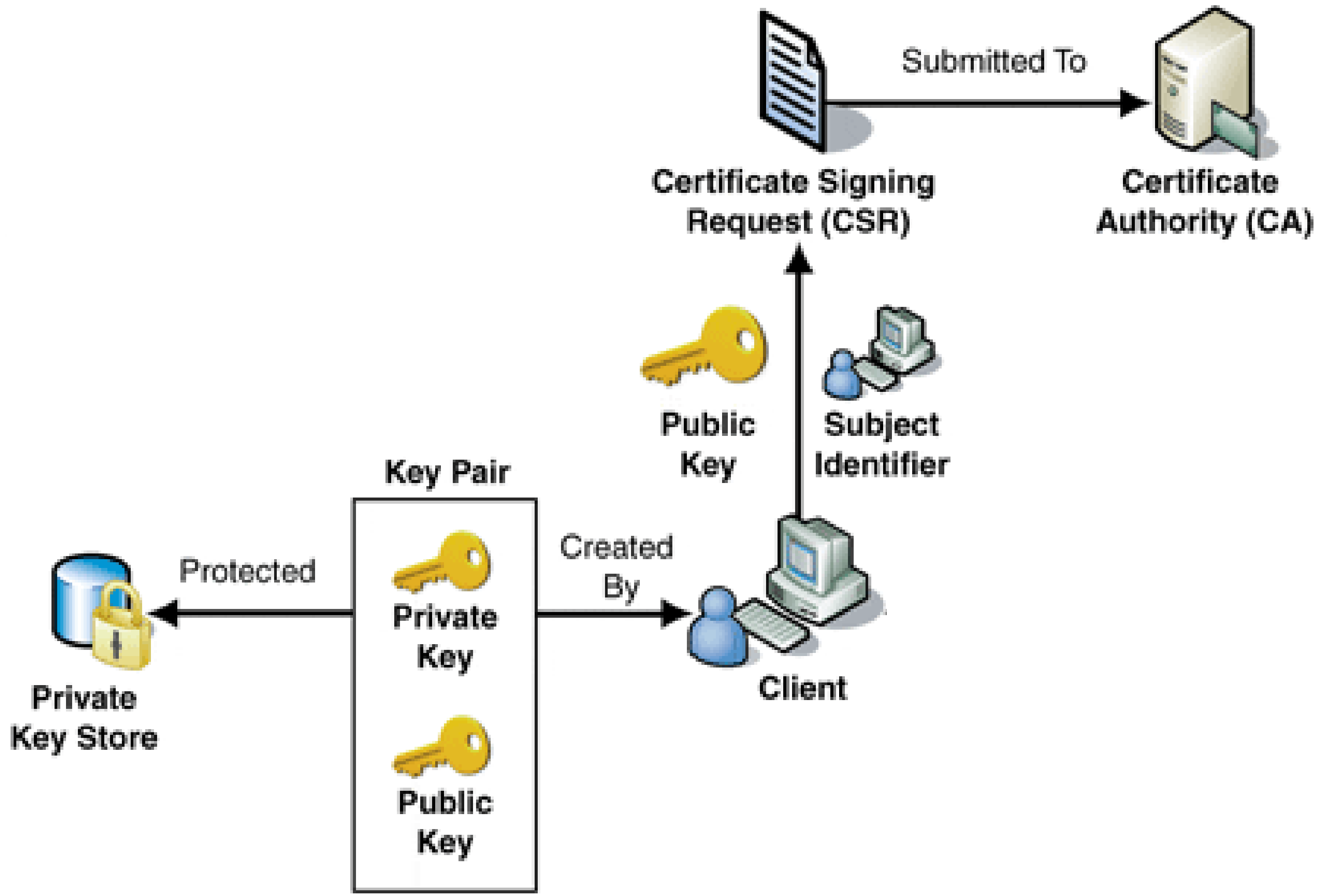
Certificate signing steps 1/4



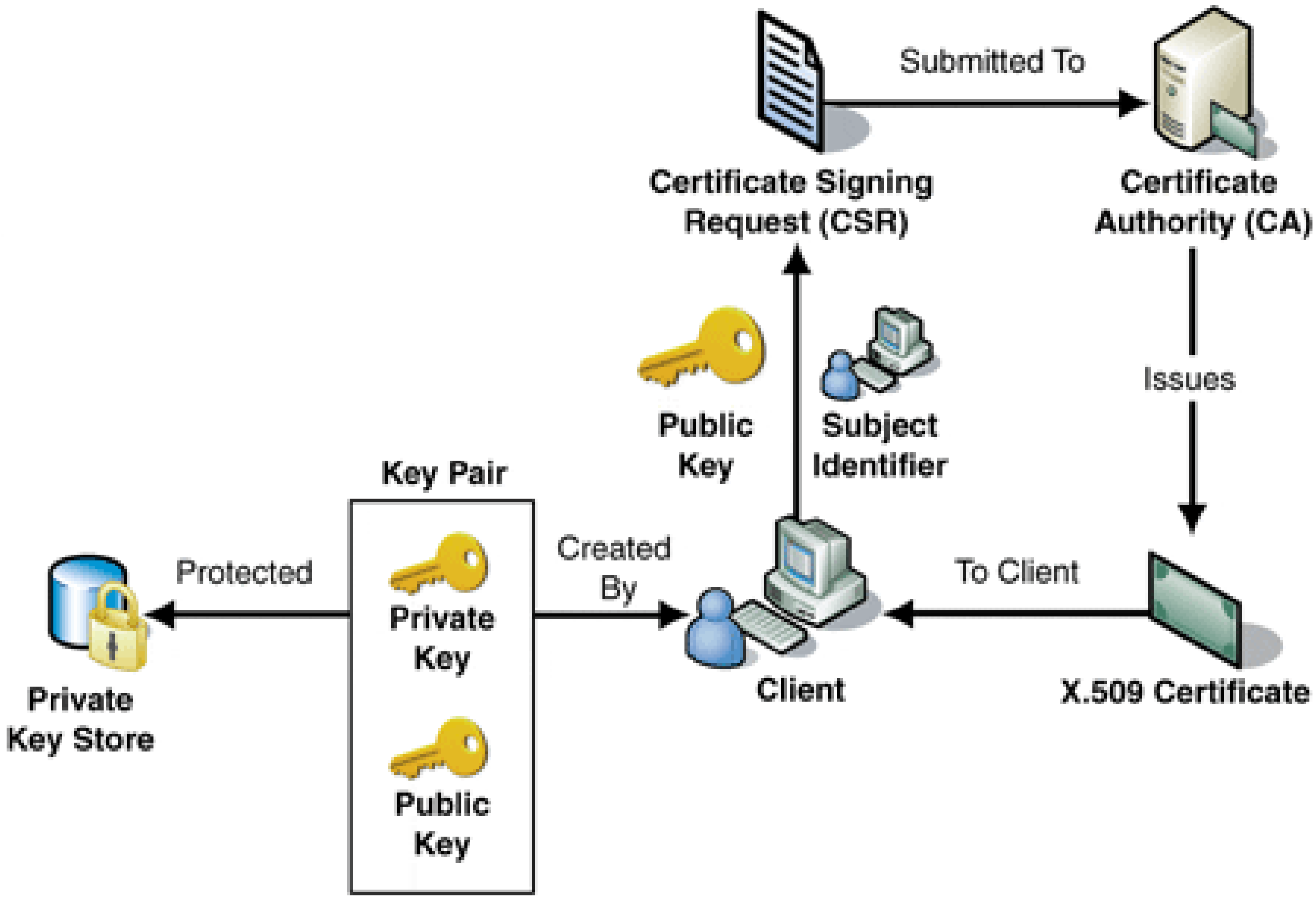
Certificate signing steps 2/4



Certificate signing steps 3/4



Certificate signing steps 4/4



Asymmetric key pairs validity

- Keys to assure **confidentiality**
  - The public key of X is used by the sender to assure confidentiality of the data sent to X
    - And the private key of X is used to decipher the received information
  - These keys can be refreshed frequently
    - In the worst scenario, the data is re-sent
- Keys to assure **authentication**
  - The private key of X is used to sign the content
    - And the corresponding public key to validate the signature
  - These keys should not be renewed frequently
    - To simplify the signature validation process

Roadmap

- Introduction
- Distribution of public keys
- Public-key certificates
- Certificate issuance
- **Certificate distribution**
- Certificate revocation

PKI (Public Key Infrastructure)

- Infrastructure to manage certificates
  in a certain context
  - Example context: the Web
- Encompasses:
  - A set of CAs and similar entities
  - Policies and mechanisms
- Operations supported
  - Secure creation of asymmetric key pairs
  - Creation and distribution of public key certificates
  - Definition and usage of certification chains
  - Update, publication and query of certificate revocation lists

PKI entities

Certification Authority (CA)

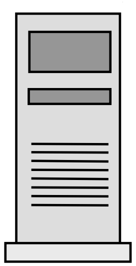
Reliable entity that creates and publishes the certificates in the repository.

Certification Revocation List Authority (CRLA)

Trusted entity that creates and publishes the revocation certificates in the repository.

Subscriber

- Generates a key pair
- Requests a certificate for its public key
- Receives the certificate
- Uses its private key



Repository

Verifier

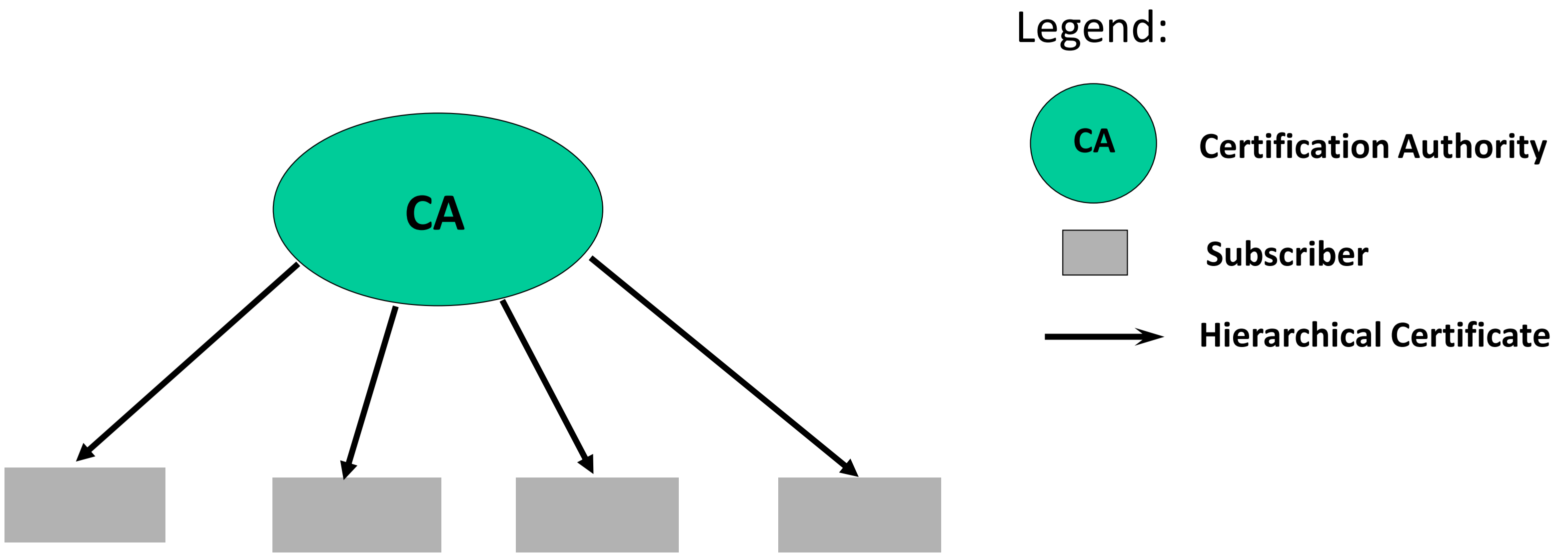
- Finds out certificates in the repository
- Validates certificates in order to validate a certification chain
- Uses the public key of the subscriber

PKI: Trust relations

- A CA establishes trust relations in two ways:
  - By issuing public key certificates of other CAs
    - Below in the hierarchy or hierarchically unrelated
  - By requiring certification of its public key to other CAs
    - Above in the hierarchy or hierarchically unrelated
- Typical trust relations
  1. Flat
  2. Hierarchical
  3. List of CAs

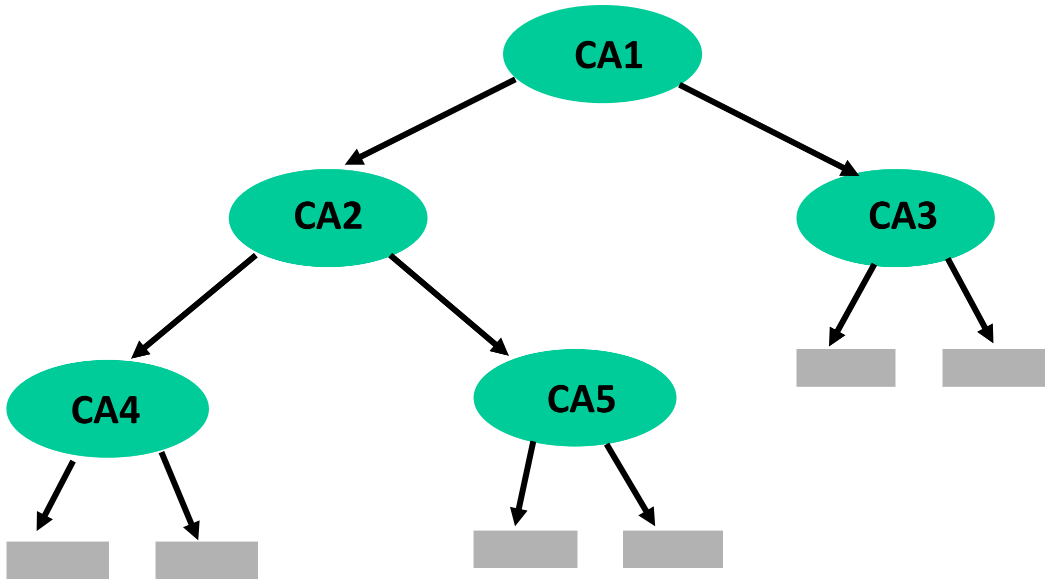
1. Flat

- Trusted single root CA
  - Verifying entities trust the public key of a single well-known CA
- Verifying entities check the certificates validity with the public key of the CA.



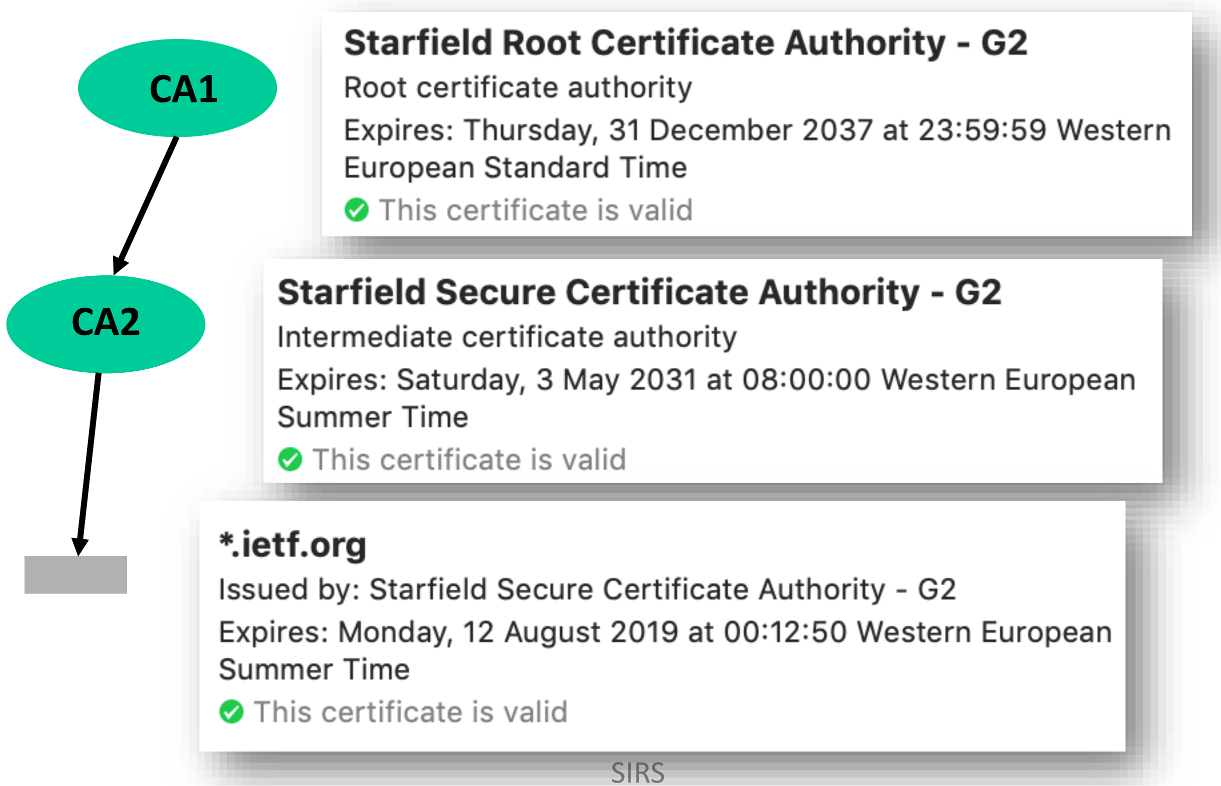
2. Hierarchical

- A tree of CAs
- The verifying entities trust the key of CA1
- CAs issue certificates to subscribers and other CAs
- *Verifying entities verify the certificates of the subscribers by sequentially checking the certificates up to the root certificate*



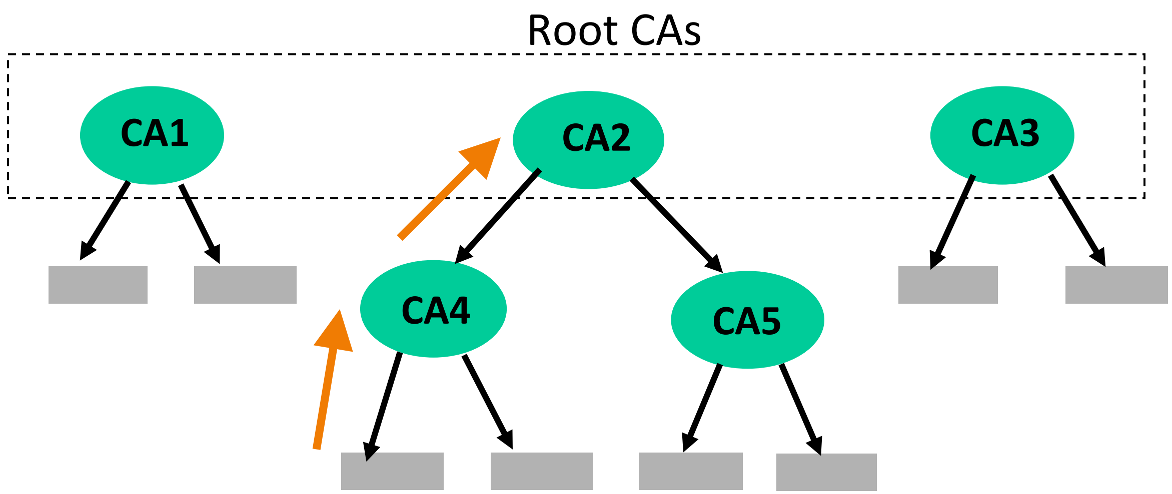
Intermediate certificates

- Two primary reasons to use intermediate certificates:
  - Protect the PKI root certificate
  - To delegate signing authority to another organization (sub-CA); needed for scalability reasons



3. List of certificates

- The verifying entities trust the keys of several root CAs
- The verifying entities validate the chain of certificates that lead to any of the CAs



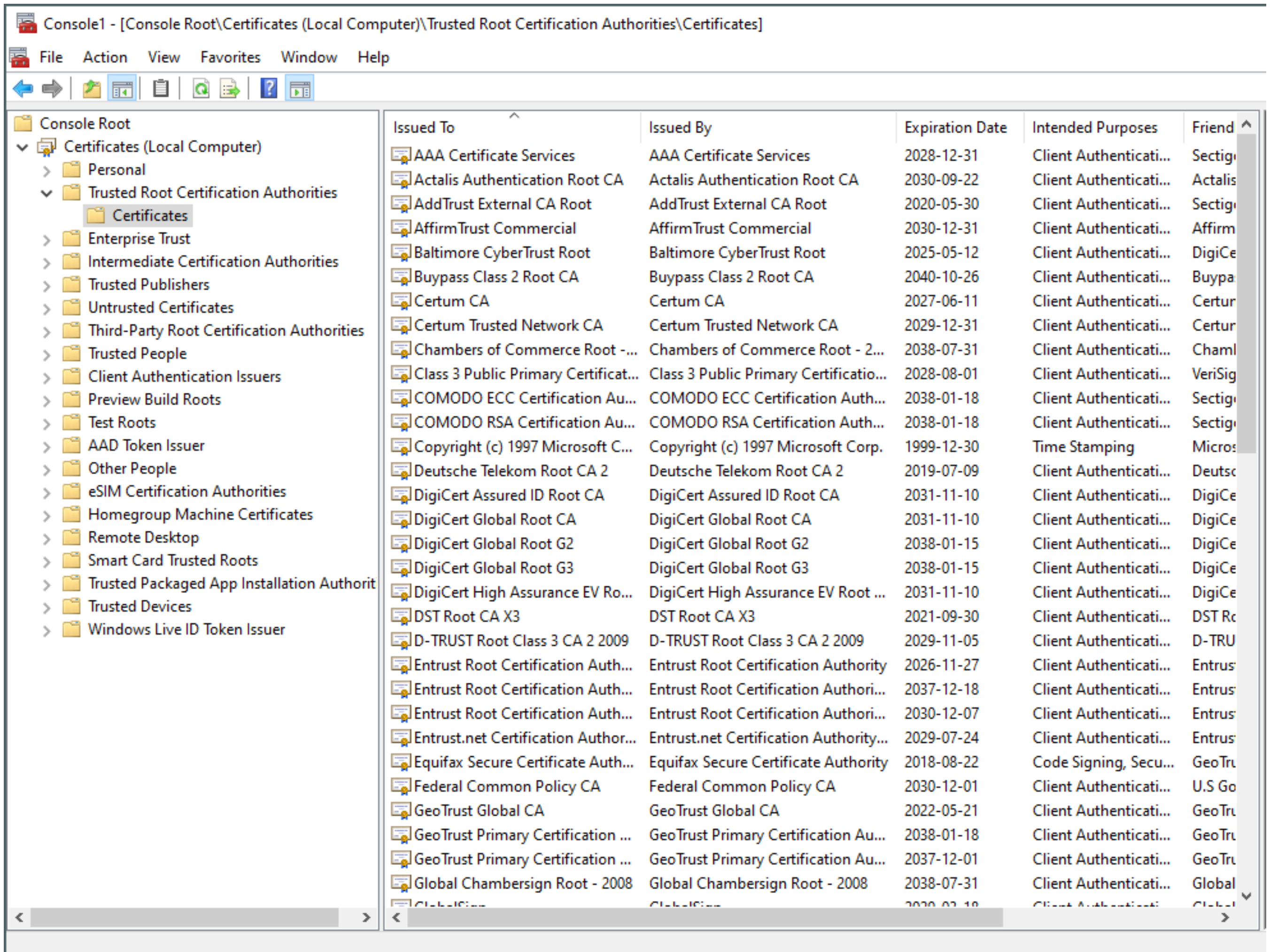
example chain with 3 certificates

OS/browser root stores

- Root store: list of certificates of trusted CAs
  - CAs trusted to issue certificates to the correct entities
  - Applications that use X.509 need to have a root store
    - Operating systems have root stores
      - Windows, OS X (Keychain), Linux
    - Browsers use root stores: Mozilla ships its own, Edge uses Windows' root store, Chrome can pick OS or its own, etc.

See OS root store

Windows Trusted Root Certificate Authorities



MMC – Microsoft Management Console

Basic Solutions: advantages & disadvantages

- Flat
  - \+ Simpler
  - – Limited to a single organization
  - – Scales poorly
- Hierarchical
  - \+ Simple to find a certification path
  - – Clients trust a single global entity
- List of Certificates
  - \+ Solves problems of the previous two
  - – Client does not know the practices of each root CA
  - – Client does not know which CA was used to verify a given certificate
  - – Revocation is difficult

Roadmap

- Introduction
- Distribution of public keys
- Public-key certificates
- Certificate issuance
- Certificate distribution
- **Certificate revocation**

Certificate withdrawal

- There are several cases when a certificate must be withdrawn:
  - Corresponding private key compromised
  - Certificate owner does not operate service any longer
  - Key ownership has changed
  - Certificates issued to entity that not the one indicated

https://www.zdnet.com/article/microsoft-warns-fraudulent-digital-certificates-issued-for-high-value-websites/

Certificate revocation

- Revocation is crucial — yet often neglected
  - No certificate should be considered valid without a revocation check
    - Because we need confirmation that a certificate is valid **at the moment** of interest, not sometime in the past
- In these cases, there are two options: CRLs and OCSP

CRL (Certificate Revocation Lists)

- CRLs are lists of revoked certificates
  - Should be regularly checked by the certificate holders
- Maintenance and dissemination of CRLs
  - Institutional certification
    - Each CAs maintains and allows reading access to the list it keeps/knows
      - Example: http://crl.multicert.com/
    - The CAs exchange CRLs themselves in order to facilitate the knowledge of all revoked certificates
  - Ad-hoc certification
    - The entity that holds the revoked key pair must create and publicize the revocation certificate the best it can

Problems with CRLs

- Intermediate certificates should be checked too
  - Induces load and network activity
- There is a time interval between two updates which is a window for attack
- CRLs can become large
  - Solution: delta CRLs that contain only latest updates
  - Requires server-side support—very rarely used
- Downloads of CRLs can be blocked by a man-in-the-middle
- For these reasons: most browsers have never activated CRLs checks by default ☹

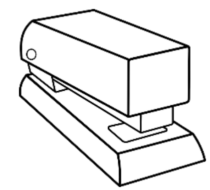
OCSP (Online Certificate Status Protocol)

- OCSP allows live revocation checks over the network
- Request-response model
  - Request: lookup of certificate in server-side CRL data structure
    - Certificates contain the URL of their issuer's OCSP server
    - Query by several hash values and certificate’s serial number
    - Protection from replay attacks with nonces
    - Query may be signed
    - Does not require encryption
  - Response:
    - Contains certificate status: **good**, **revoked**, unknown
    - Must be signed
    - Does not require confidentiality

Problems with OCSP

- Lookups go over the network
  - Induces latency
- OCSP information must be fresh
- OCSP servers must have high availability
- OCSP can be blocked by a man-in-the-middle
  - Many browsers will ‘soft-fail’ = show no error
  - Browsers ‘accept as good’ if no OCSP response received
- Privacy exposure: OCSP servers know which sites the users are accessing

OCSP stapling



- Idea: the web server obtains a fresh OCSP response and "staples" it to the certificate given to the web browser
  - The browser checks the signature of the certificate and of the recent OCSP validity assertion
- More efficient
  - One call to OCSP gets a response that can be served to multiple clients for a period
  - Browser receives certificate and OCSP from server in the same response
- More secure
  - CA will deny OCSP response for a revoked certificate
- More private
  - It is the server that calls OCSP, not the client
- Support for OCSP stapling is increasing, but still not universal
  - Servers: Windows, Apache and Nginx
  - Clients: Chrome and Firefox

New approaches to revocation

- In-browser revocation lists
  - Browsers preload a list of revoked certificates for the most common and important domains
    - Limited number, not scalable
  - Updates are distributed via the browser's update mechanism
    - E.g. Google Chrome
- Short-lived certificates
  - Give certificates a very short validity period
    - 1 hour–1 day
  - Replace certificates fast; do not attempt any other revocation
  - Works well and gives clearly-defined window of attack
  - Problem: certification becomes a frequent and 'live' operation
    - Not applied in the Web so far

Revocation conclusion

- Revocation is crucial—but not fully solved so far
  - CRLs are of limited use
  - OCSP checks are expensive (latency, load) and not enough against an attacker who can drop traffic to the CA
  - Other approaches have not gained wide adoption

Summary

- Introduction
- Distribution of public keys
- Public-key certificates
- Certificate issuance
- Certificate distribution
- Certificate revocation