CRYPTO 101: REAL-WORLD DEPLOYMENTS

2. PUBLIC-KEY INFRASTRUCTURES (PKI)

Alfred Menezes
cryptography101.ca

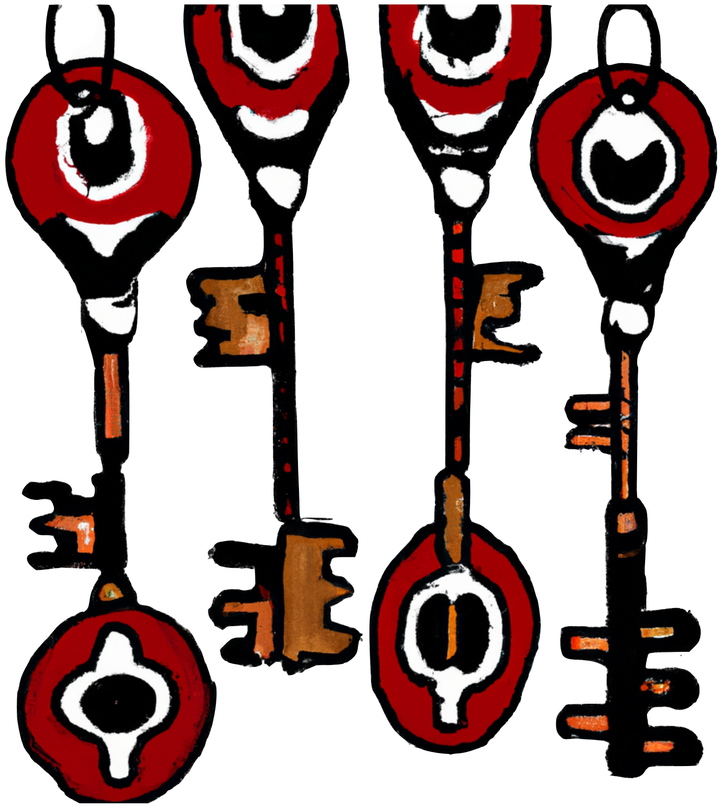
February 2025

Outline

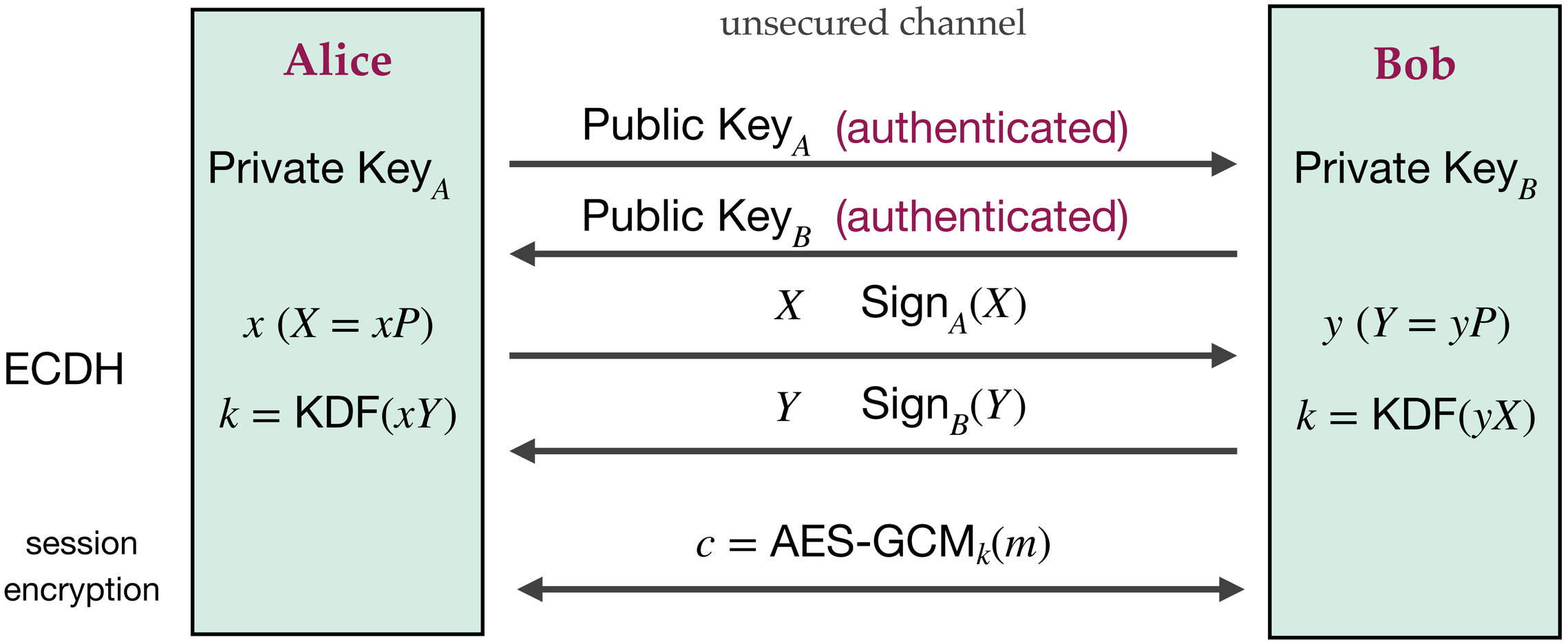
1. Key management
2. Methods for distributing public keys
3. Components of a PKI
4. Certification authorities (CAs)
5. Certificates
6. The certification process
7. Certificate revocation
8. CA trust models
9. PKI challenges

Key management

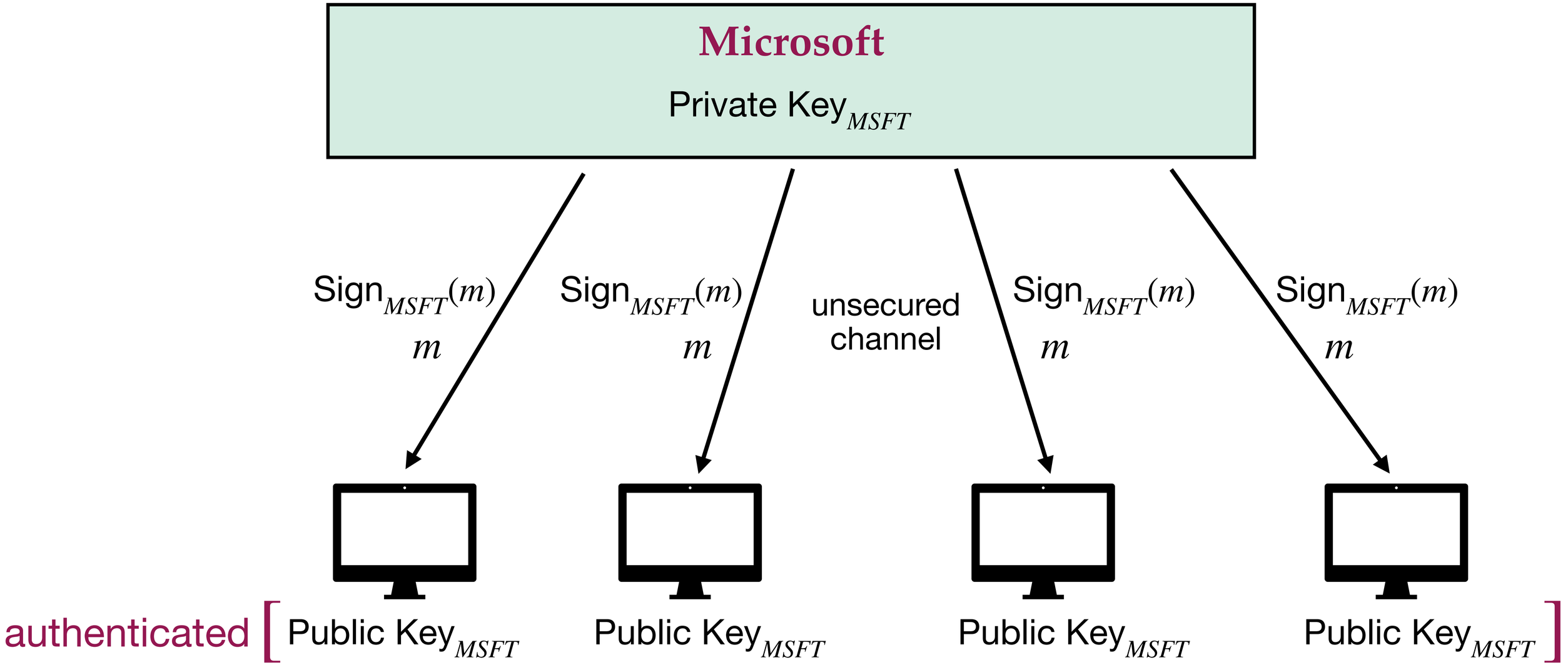
- Key management is a set of techniques and procedures supporting the creation, distribution, storage, and updating of cryptographic keys.
- *In this video, we'll consider the management of public keys that are used for public-key encryption (e.g. RSA encryption), key agreement (e.g., ECDH), and signature verification (e.g., RSA and ECDSA).*



Example: Secure channels



Example: Software updates



Three other examples

- **FIDO2: Fast IDentity Online**
  - *Reduce the world's reliance on passwords to better secure the web.*
  - Alice generates a key pair for ECDSA, and registers her public key with a website.
  - When Alice later logs into the web site, she uses her private key to sign a random challenge issued by the website.
  - The server verifies the signature using Alice's public key.

fidoalliance.org

- **ePassports** (ICAO: International Civil Aviation Organization)
  - Alice's ePassport contains a document signed by a Document Signer, that in turn is certified by a Country Signing Certification Authority (CSCA).
  - International border authorities verify the signed document.
- **Apple's mandatory code signing**: Approved app developers must sign their code.
  - See tinyurl.com/AppleAppSigning

Methods for distributing public keys

1. Point-to-point delivery over a trusted channel.
   - ✦ Trusted courier.
   - ✦ One-time user registration.
   - ✦ Visual inspection.
2. Direct access to a trusted file.
3. Use of an on-line trusted server.
4. Off-line certification authority (CA).
   - ✦ Public keys are transported in certificates, issued by a certification authority.

Public-key infrastructures (PKI)

- **Certificate**: A public-key *certificate* is a (digital) document generated by a Certification Authority. The document cryptographically binds together an entity's identity to its public key (and other attributes) in a manner that allows other entities to gain trust in the public key's authenticity.
- **PKI**: A comprehensive framework that combines cryptographic protocols, technology, policies and supporting services (legal, business, etc.), to ensure the secure and reliable use of public-key certificates.

Components of a PKI

1. Certificate format.
2. Certification Authority (CA).
3. The certification process.
4. Certificate revocation.
5. Trust models.
6. Certificate policy:
   Details of the intended use and scope of a particular certificate.
7. Certification practice statement (CPS):
   Practices and policies followed by a CA.

Certificate format

- X.509 certificate format

Version number	
Serial number	
Issuer name	← identifier of the CA
Validity period	← includes an expiration date
Subject name	← identifier of the certificate subject (e.g. “Google: google.com”)
Subject public key algorithm	← Algorithm to be used with Google’s public key (e.g. ECDSA)
Subject public key	← Google’s ECDSA public key
Certificate signature algorithm	← CA signature scheme (e.g. RSA-PKCS #1 v1.5)
Other attributes	
Certificate signature	← CA’s RSA signature on the other data fields

Certification Authorities (CAs)

- A CA issues certificates that bind an entity's identity A and its public key.
- A's certificate Cert_A consists of:
  1. Data part D_A: A's identifier, her public key, and other information such as validity period.
  2. Signature part Sign_{CA}: The CA's signature on the data part.
- B obtains an authentic copy of A's public key as follows:
  3. Obtain an authentic copy of the CA's public key Pub_{CA}.
  4. Obtain Cert_A (perhaps over an unsecured channel).
  5. Verify the CA's signature Sign_{CA} on D_A using Pub_{CA}.

Note: The CA does not have to be trusted with users' private keys.

Note: The CA has to be trusted to not create fraudulent certificates.

The certification process (1)

1. **CA key generation**: The CA's signature key pair is generated.
   a) Security of the CA's private key is paramount.
   b) Ideally, the private key is generated in a *Hardware Security Module* (HSM), and never leaves the HSM.
   c) Ideally, several copies of the private key (or shares of the private key) are stored in HSMs in different geographical locations.
2. **Certificate request**: An entity *A* requests a certificate for their public key.
   a) The request should be authenticated.
   b) It may be necessary for the CA to maintain a record of the request.

The certification process (2)

3. **Identity verification**: The CA verifies *A*'s identity.
    a) This might be delegated to a *Registration Authority* (RA) or an *Intermediate CA*.
    b) The RA generates *registration certificates* and forwards to the CA for certificate issuance.
4. **Key validation**:
    a) The CA (or RA) verifies that *A*'s public key is valid (i.e., a private key logical exists).
    b) *A* proves possession of the corresponding private key.
5. **Certificate issuance:**
    a) The CA produces *A*'s certificate.
    b) The CA should require notification that *A* has accepted.

Certificate revocation

- A CA may wish to revoke (i.e. invalidate) a certificate before its expiry date.
- Reasons for certificate revocation include:
  - Private key compromise.
  - Owner leaves an organization.
  - Owner changes roles within an organization
- An entity who relies on a certificate needs to verify that the certificate has not been revoked.
- Three methods for informing relying parties about revoked certificates are:
  - Short-lived certificates.
  - Online certificate status checking.
  - Certificate Revocation Lists (CRLs).

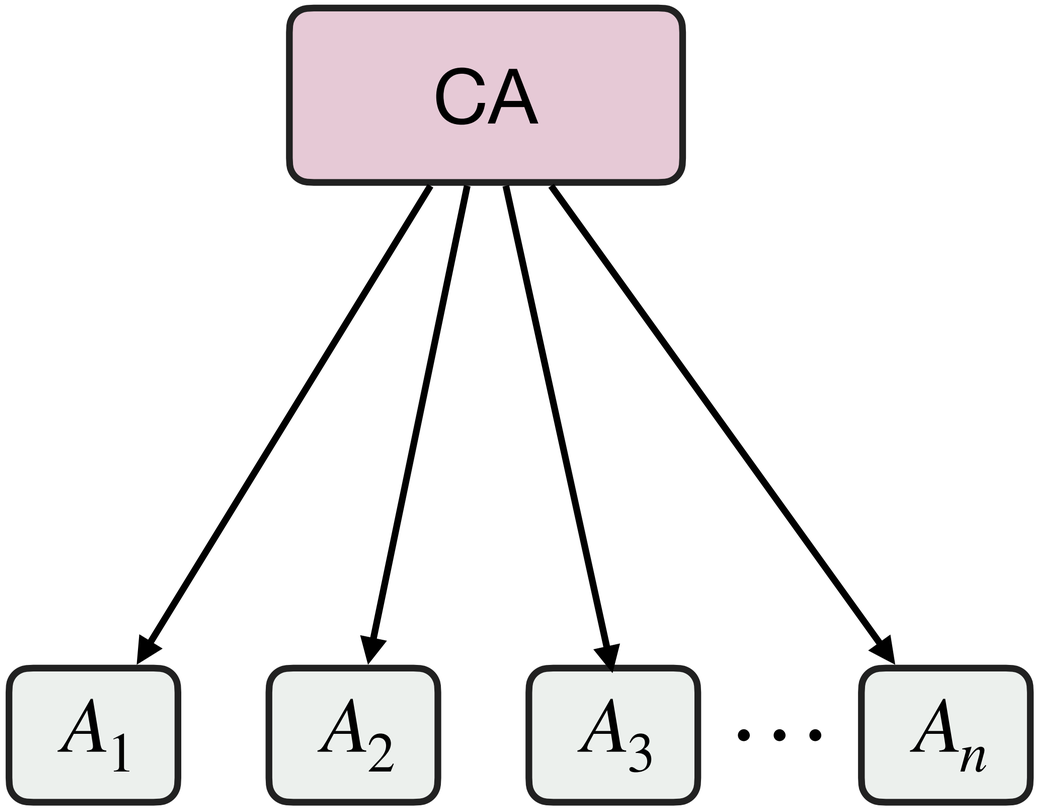
≡ WIKIPEDIA The Free Encyclopedia

The **Online Certificate Status Protocol** (**OCSP**) is an Internet protocol used for obtaining the revocation status of an X.509 digital certificate.[2] It is described in RFC 6960 and is on the Internet standards track.

Certificate revocation lists (CRLs)

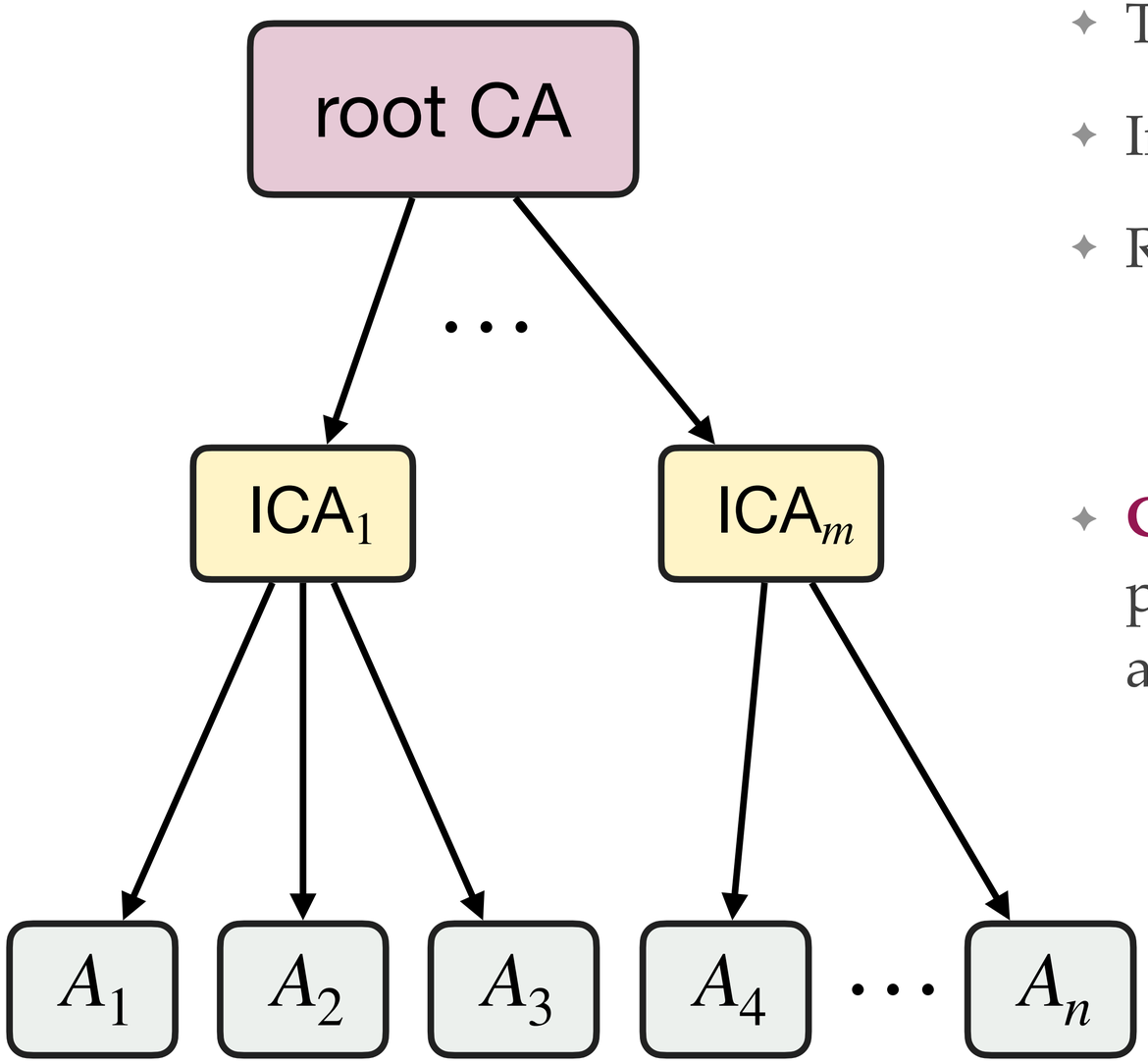
- A CRL is a list of certificates that have been revoked before their scheduled expiration dates.
- The CRL is issued and signed by a CA, and updated at regular intervals.
- An entity checks that a certificate is not included in the latest CRL before relying on that certificate.
- CRLs can be challenging to manage in large-scale PKI deployments:
  - Time granularity (time between revocation and CRL update).
  - CRL may become too large.

CA trust model: Single CA



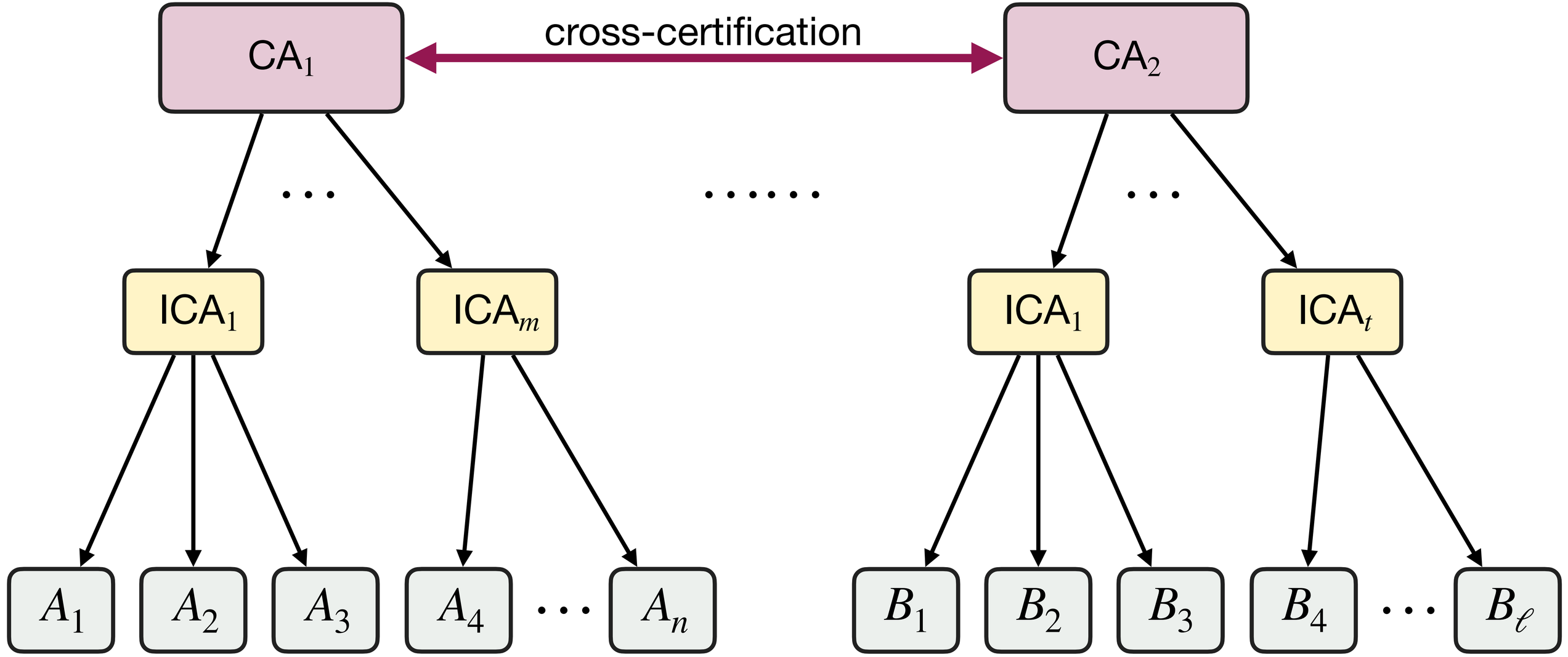
- Example: Code signing in a software company.
  - **CA**: Apple.
  - **Endusers** A_i: App developers.
  - See tinyurl.com/AppleAppSigning

CA trust model: Single-root hierarchical CA

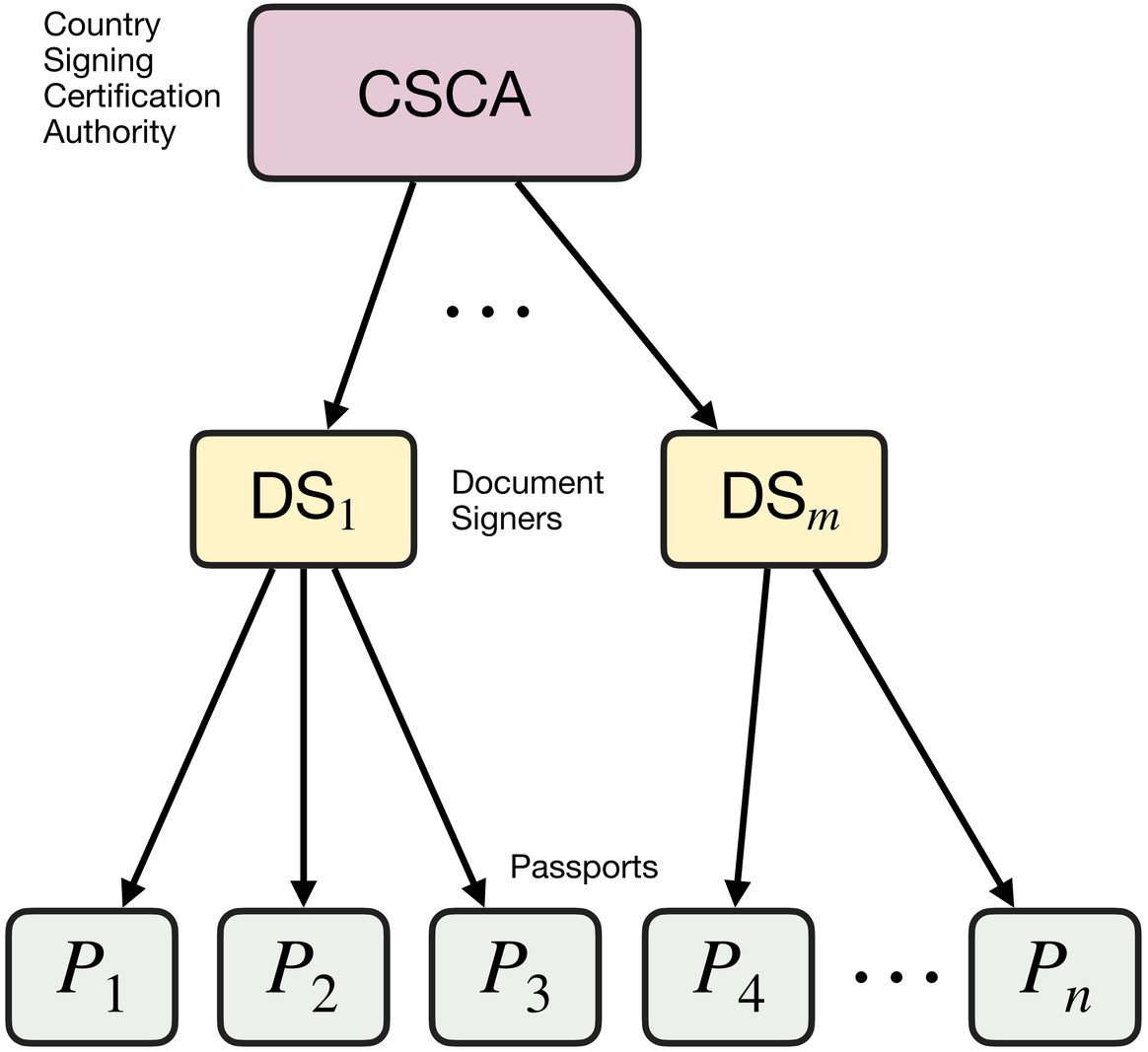


- The **root CA** issues certificates to **intermediate CAs**.
- Intermediate CAs issue certificates to **endusers**.
- Reasons for using intermediate CAs include:
  - distributing workload, reduce risk, separation by function, …
- **Certificate chains**: To obtain an authentic copy of A_2's public key, one needs A_2's certificate, ICA$_1$'s certificate, and an authentic copy of the root CA's public key.
  - Use the root CA's public key to verify ICA$_1$'s certificate, thus verifying the authenticity of ICA$_1$'s public key.
  - Use ICA$_1$'s public key to verify A_2's certificate, thus verifying the authenticity of A_2's public key.

CA trust model: Multi-root hierarchical CA



Example: ePassports PKI



icao.int/Security/FAL/PKD/Pages

- Each participating country maintains a two-level single-root hierarchical PKI.
- The CSCA issues certificates for Document Signers.
- The DS signs the biographical information in an ePassport. The ePassport stores this signed document, and also the DS's certificate.
- To verify an ePassport, one needs the CSCA public key.
- CSCA public keys are distributed via a Public Key Directory (PKD).

PKI challenges

Although conceptually very simple, there are many challenges with deploying PKI on a large scale.
Many of the challenges arise from business, legal, and usability considerations. These challenges include:

- Interoperability: alleviated by standardized and certificate formats.
- Certificate revocation (CRLs).
- Secure generation and use of private signing keys: use HSMs.
- Business and legal issues, including cross-certification and liability.
- Migration to quantum-safe cryptography.