727 Crypto Management

Lecture 13

GIOVANNI RUSSELLO

G.RUSSELLO@AUCKLAND.AC.NZ

Public-Key Certificates

The PA still represents a bottleneck in the system and shares pretty much the same security issues as the public directory

As an alternative, Kohnfelder suggested in 1978 the idea of using digital certificates

- Allows participants to share public-key securely without involving a PA
- But with the same level of reliability

Simply put, a certificate consists of a (owner ID, the public-key)-pair signed with the private key of a trusted certificate authority (e.g., Comodo, Symantec, etc)

The user presents her public key to the CA in a secure way and then gets back the certificate that can be published

Anyone obtaining the certificate can retrieve the public key of a user and use the digital signature on the certificate to verify that is valid

PK Certificate Requirements

1. Any user reading the certificates is able to determine the name and PK of the certificate owner
2. Any user can verify that the certificate originates from the CA and is not counterfeited
3. Only the CA can create and update the certificates
4. Any user can verify the currency of a certificate

Certificate Lifecycle

A user applies to a CA for a certificate providing a PK and a form of secure authentication:

- C_A = E(PR_CA, (T||ID_A||PU_A)

To verify a certificate a user computes:

- D(PK_CA, C_A) = (T||ID_A||PU_A): the certificate is readable only if it was signed by the CA private key

The timestamp tells a user if the certificate has expired in which case it should not trust its content

Main Advantage:

- Certificates are unforgeable so they can be stored in a public directory. The CA does not need to protect the directory - As long as the CA's private key is kept secure!

X.509 Certificates

It was designed as part of the ITU-T X.500 recommendations for a director services

- A set of distributed servers maintaining a database of users' information
- Mapping users' identities to network addresses

The X.509 was designed as part of an authentication service for the X.500 directory

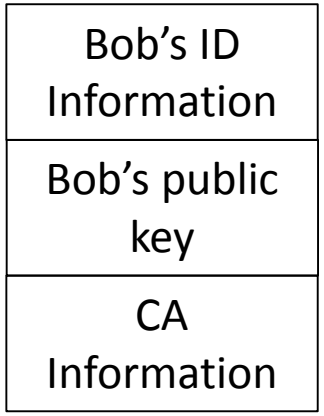
A X.509 certificate contains the public key of a user signed by a trusted authority

The standard is based on public key cryptography and hash functions

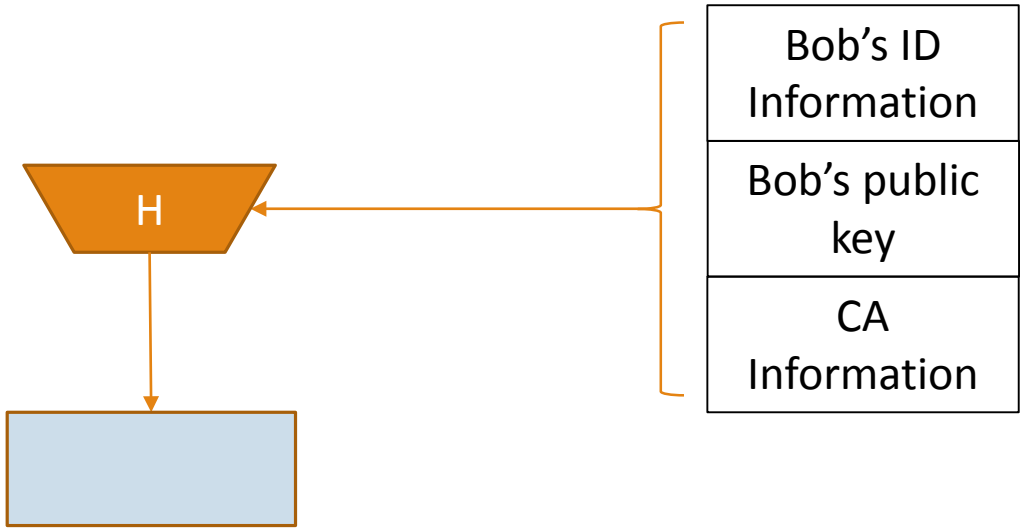
- Although the standard does not dictate which specific system to use it recommend RSA

The X.509 is important because it is used in many applications such as SSL/TSL, S/MIME, etc.

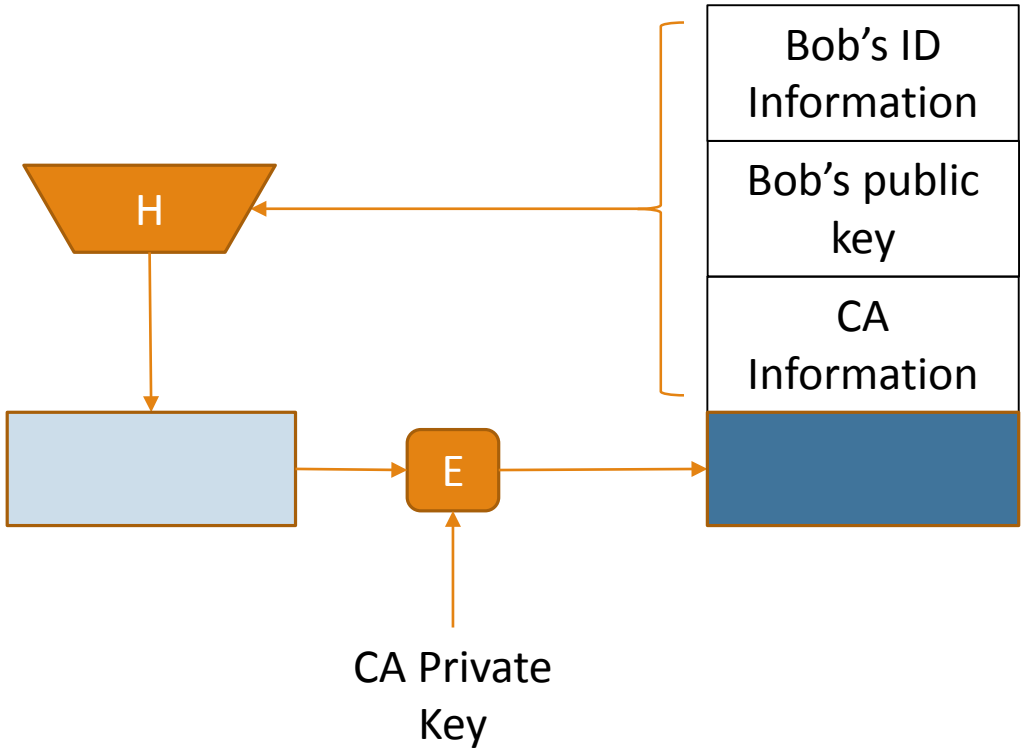
Signing a Certificate



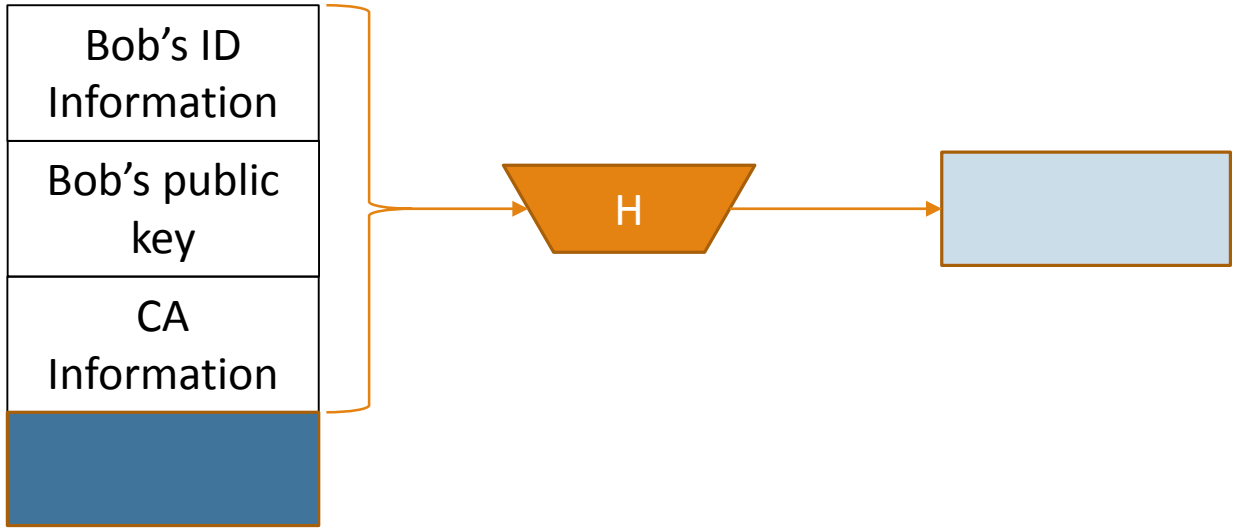
Signing a Certificate



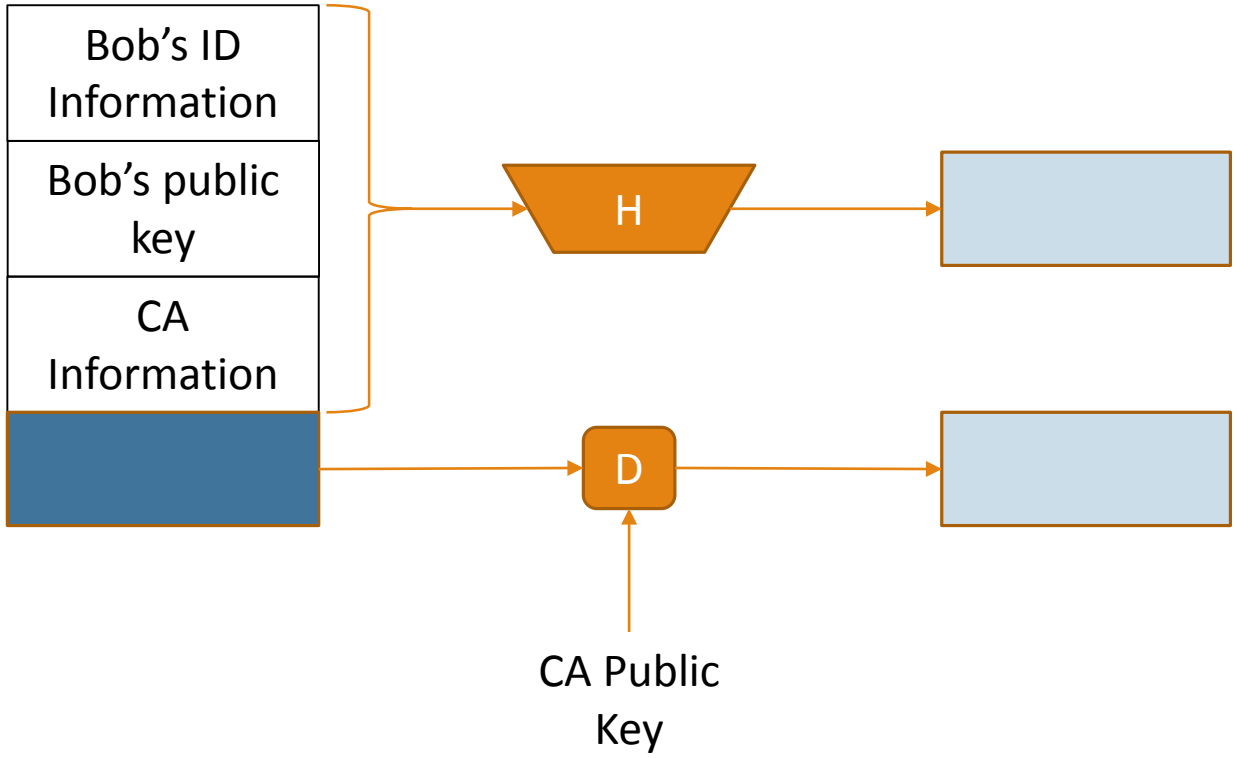
Signing a Certificate



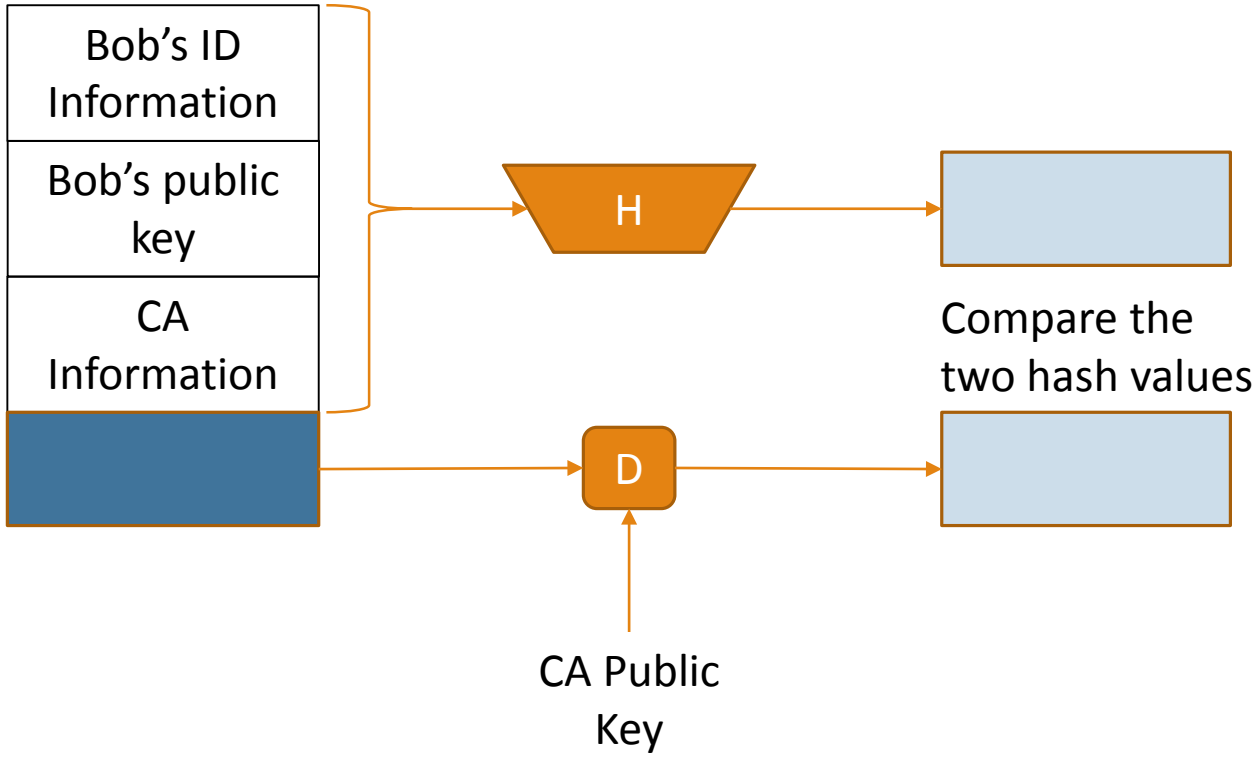
Verifying a Certificate



Verifying a Certificate



Verifying a Certificate



X.509 Format

Version: default is 1; if the Issuer or Subject Unique Identifiers are present then it should 2.1; if extensions are present then it should be 3;

Serial Number: uniquely identifies this certificate within the CA

Signature Algorithm Identifier: the algorithm used for signing this certificate

Issuer Name: X.500 name of the CA that created and signed this certificate

Period of Validity: Not before and not after date

Subject Name: the user's name of the user that holds the private key associated with the public key in this certificate

Subject's Public Key Information: Public-Key, algorithm, and parameters

X.509 Format – Cont.

Issuer Unique Identifier: Optional – this is used to identify uniquely the issuing CA in case the X.509 name has been reused for different entities

Subject Unique Identifier: Optional – this is used to identify uniquely the subject in the event the X.509 name has been reused for different entities

Extensions: a set of one or more extensions added with version 3

Signature: covers all the above fields – contains the hash of all the fields values encrypted with the private key of the CA issuing this certificate; this field include also the algorithm identifier and its parameters

Certificate Chain

Within the same organisation, user A and B can use the same CA

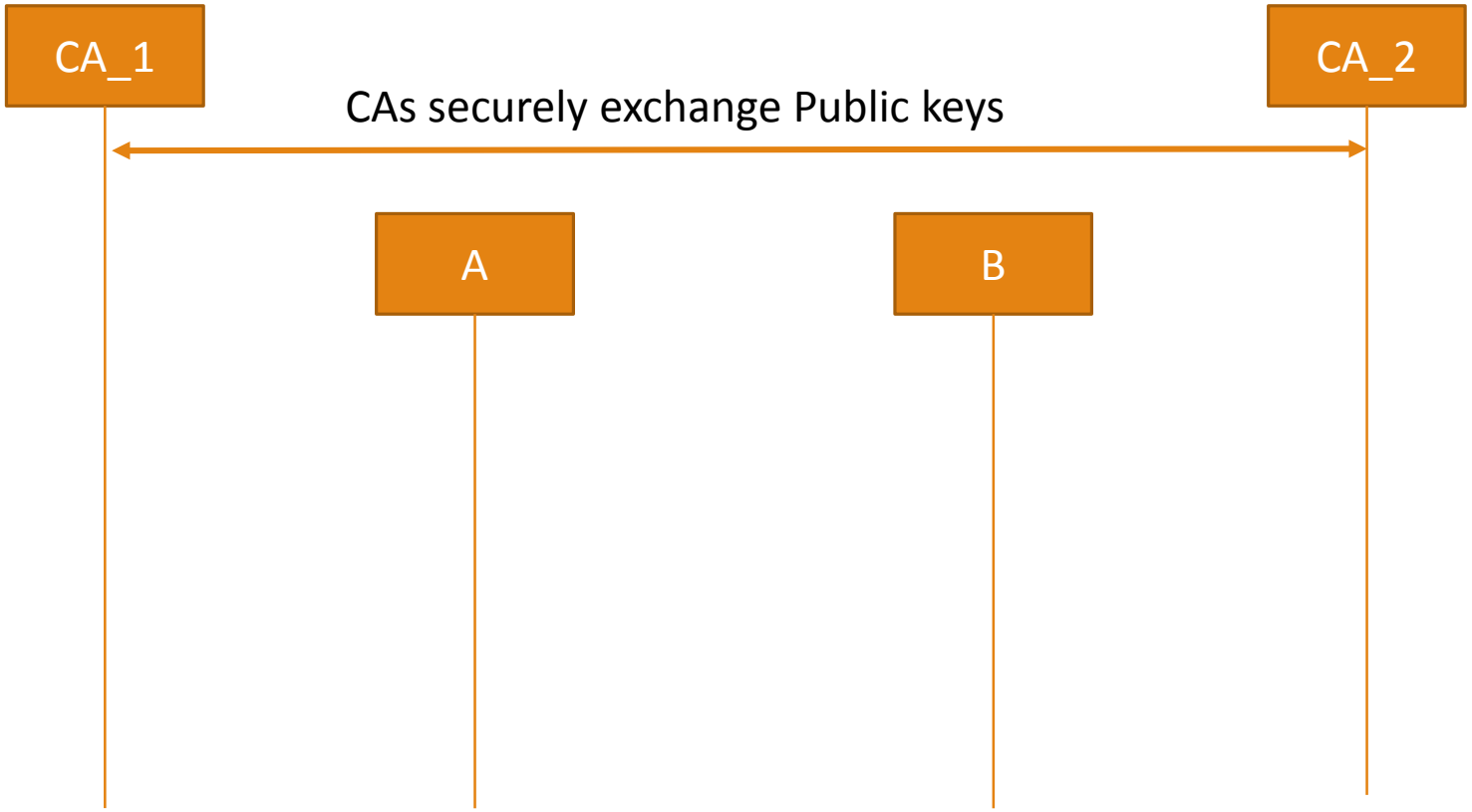
- A and B have obtained securely (WRT authenticity and integrity) the CA public key

However, when there are too many users or users from different organisations using the same CA is not advisable

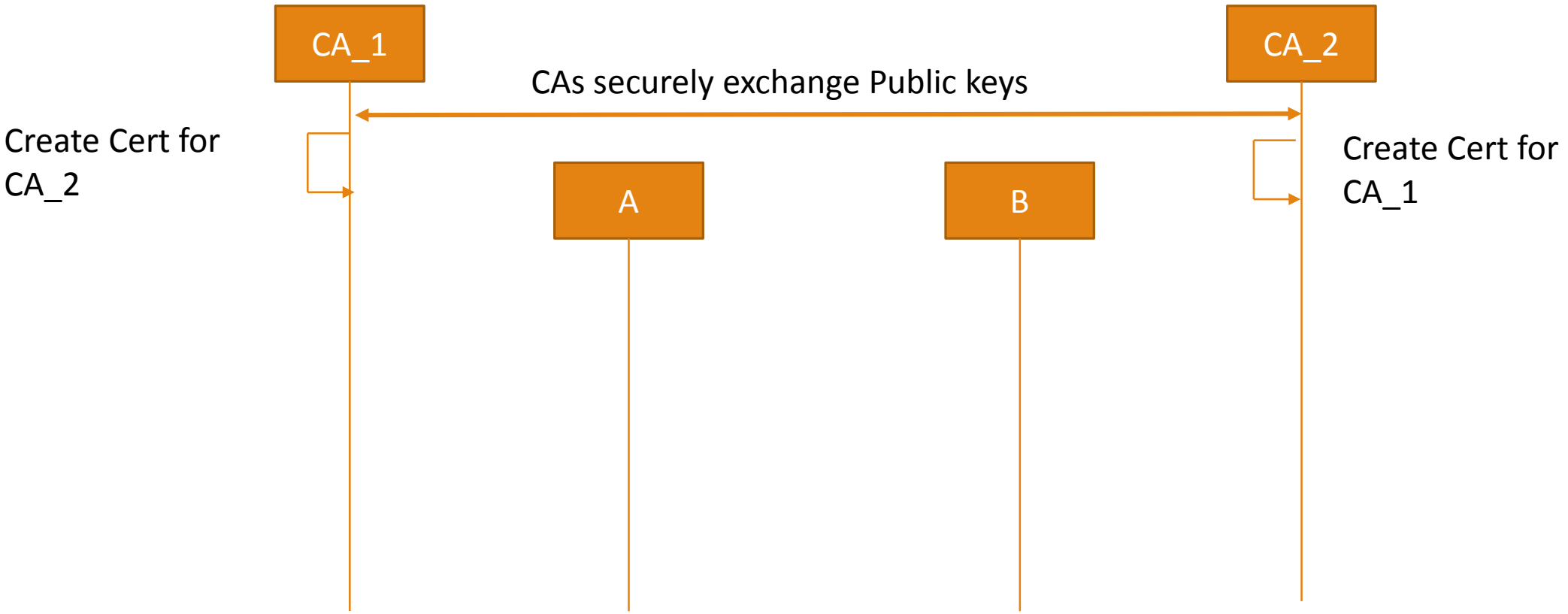
If user A gets a certificate from CA_1 and B has a certificate from CA_2:

- If A does not obtain securely CA_2 public key then B's certificate is useless
- Same for B

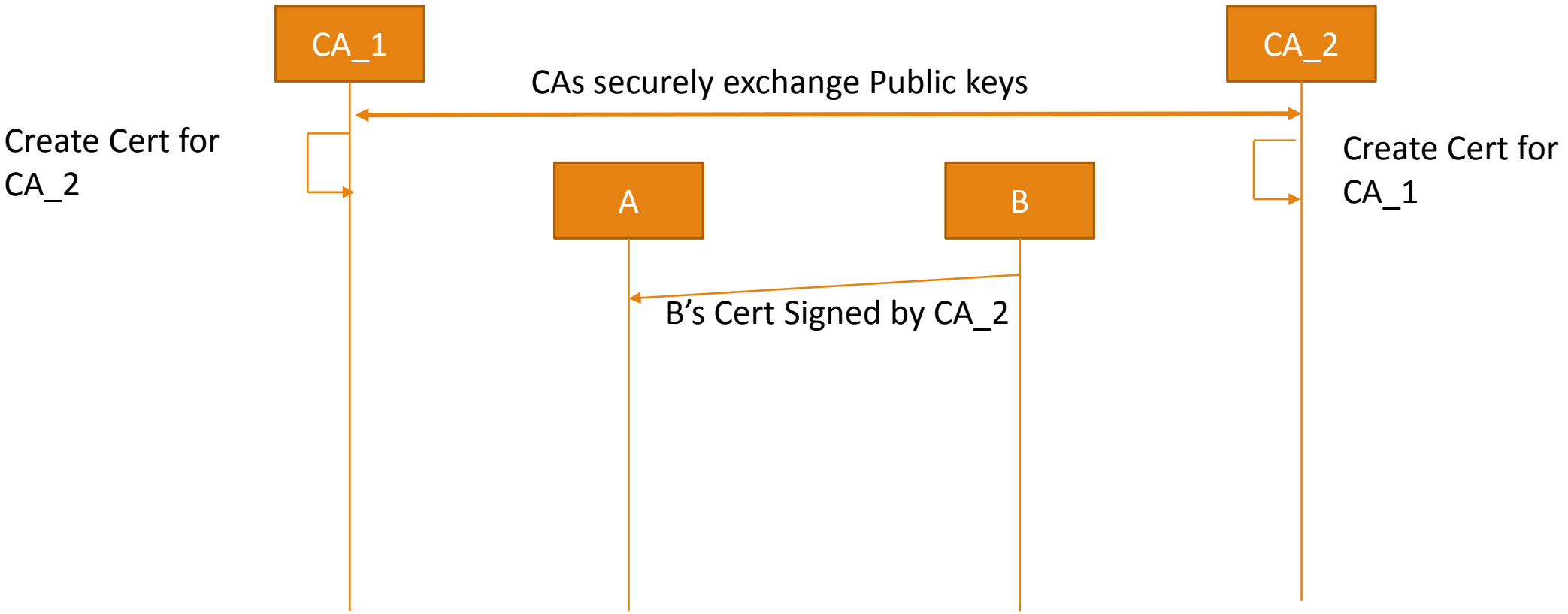
Certificate Chain Creation



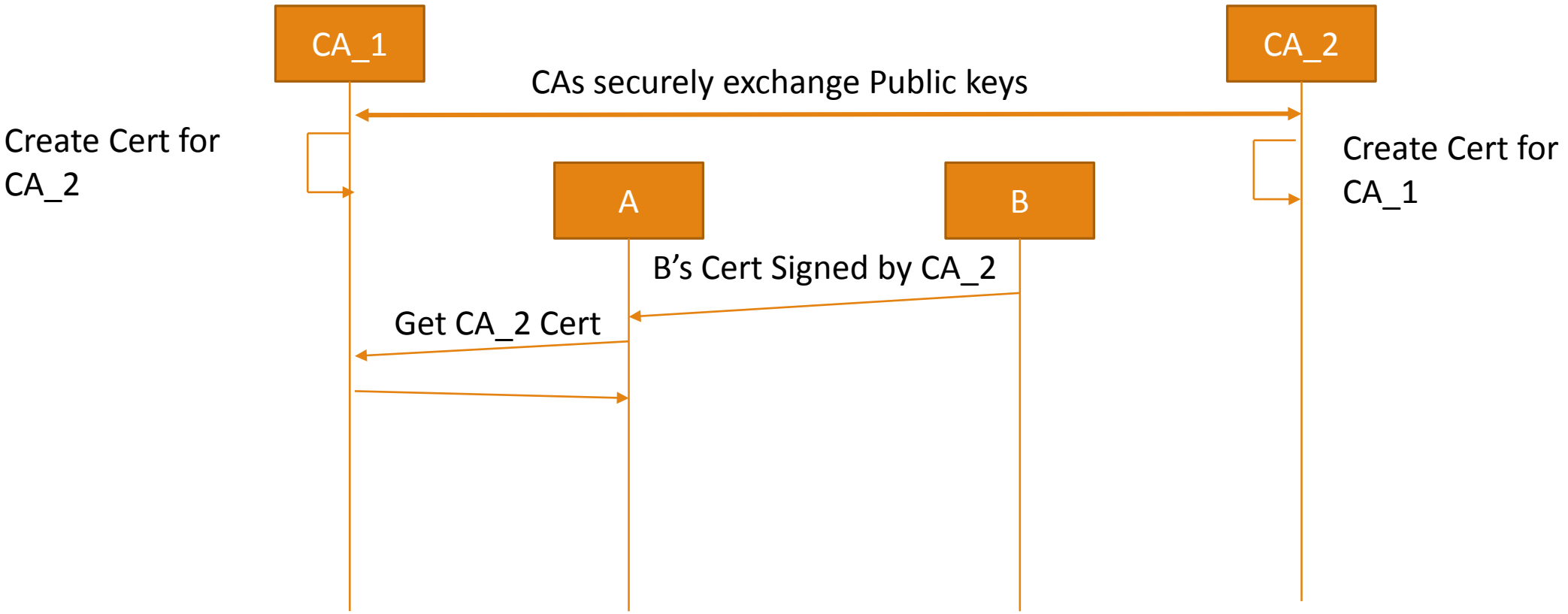
Certificate Chain Creation



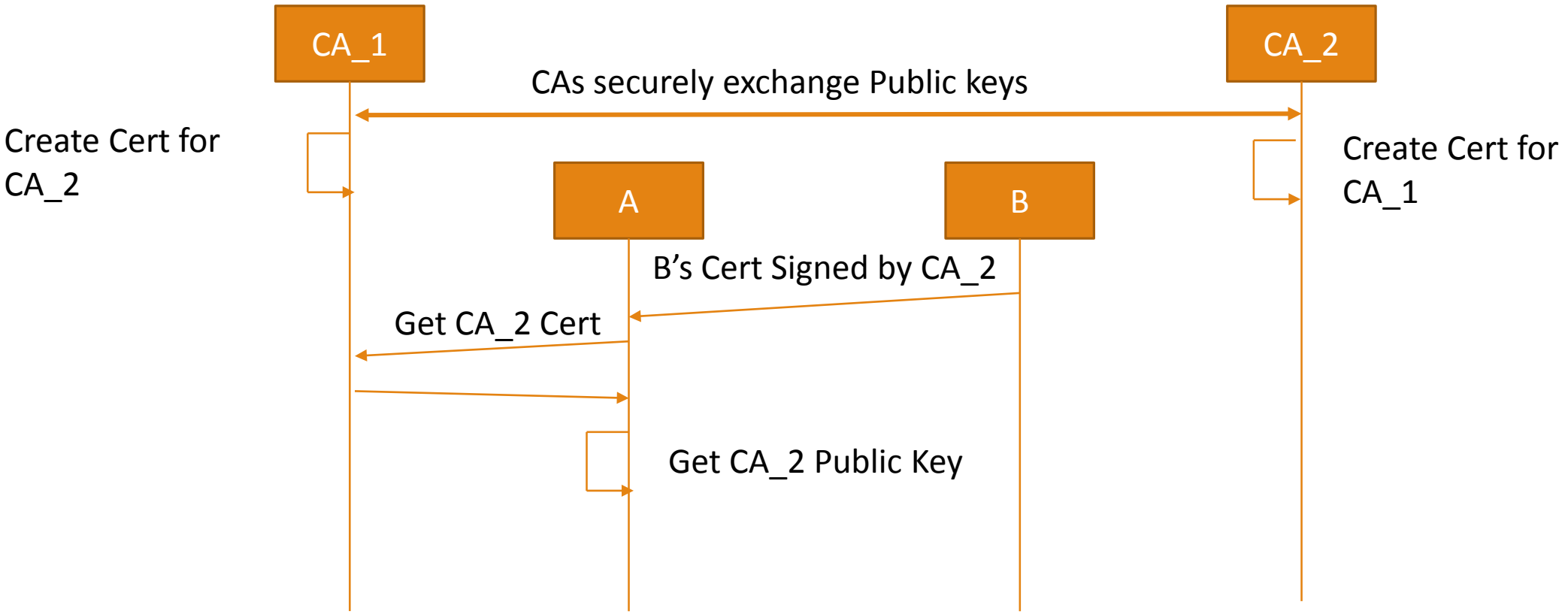
Certificate Chain Creation



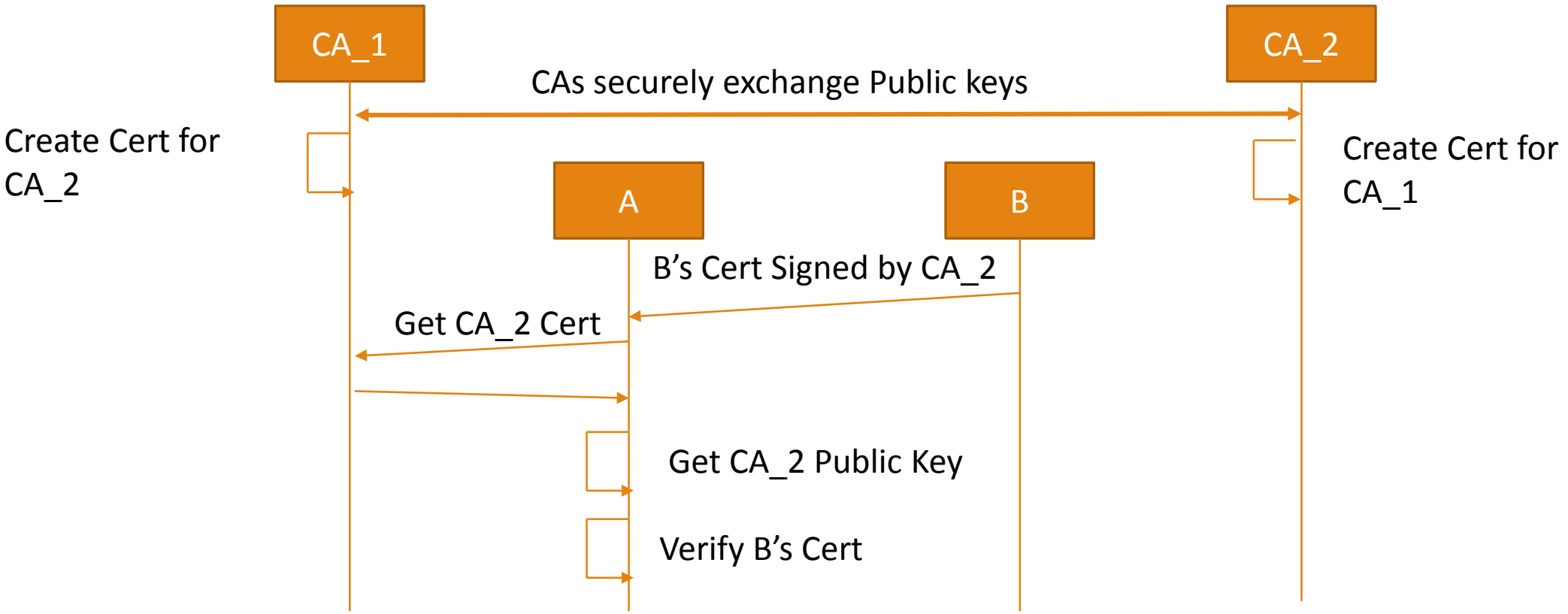
Certificate Chain Creation



Certificate Chain Creation



Certificate Chain Creation



X.509 Notation

Y <<X>> means that X's certificate was signed by Y; where Y is a certificate authority

In this notation, we can also specify a certificate chain:

- CA_1 <<CA_2>> CA_2 <<B>>

A chain can have more than two levels: X_1 <<X_2>> X_2 <<X_3>> …X_n<<B>>

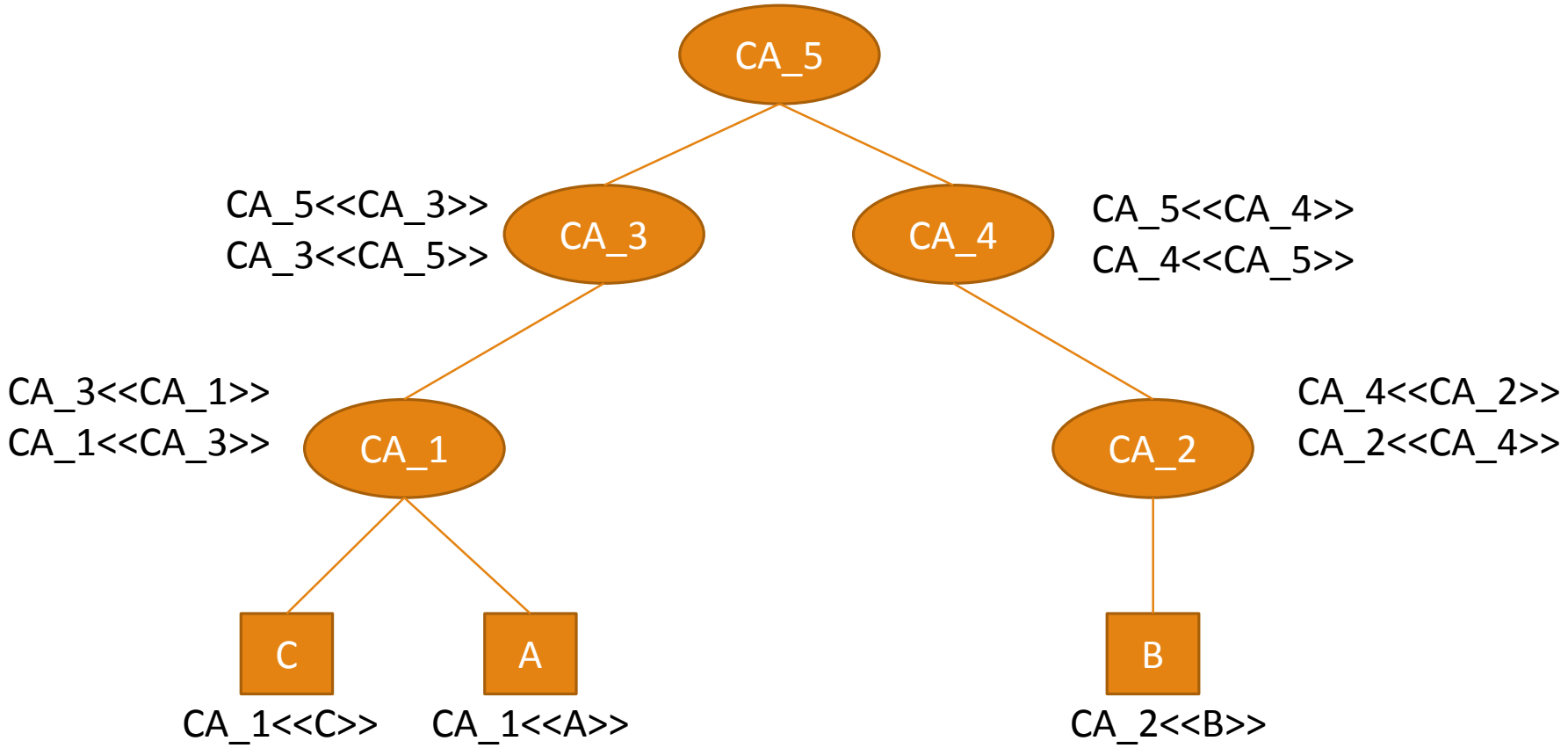
- This means that each CA involved in the chain has securely exchanged public keys and created certificates with its predecessor and successor

The certificates a CA creates for other CAs need to be available in the directory

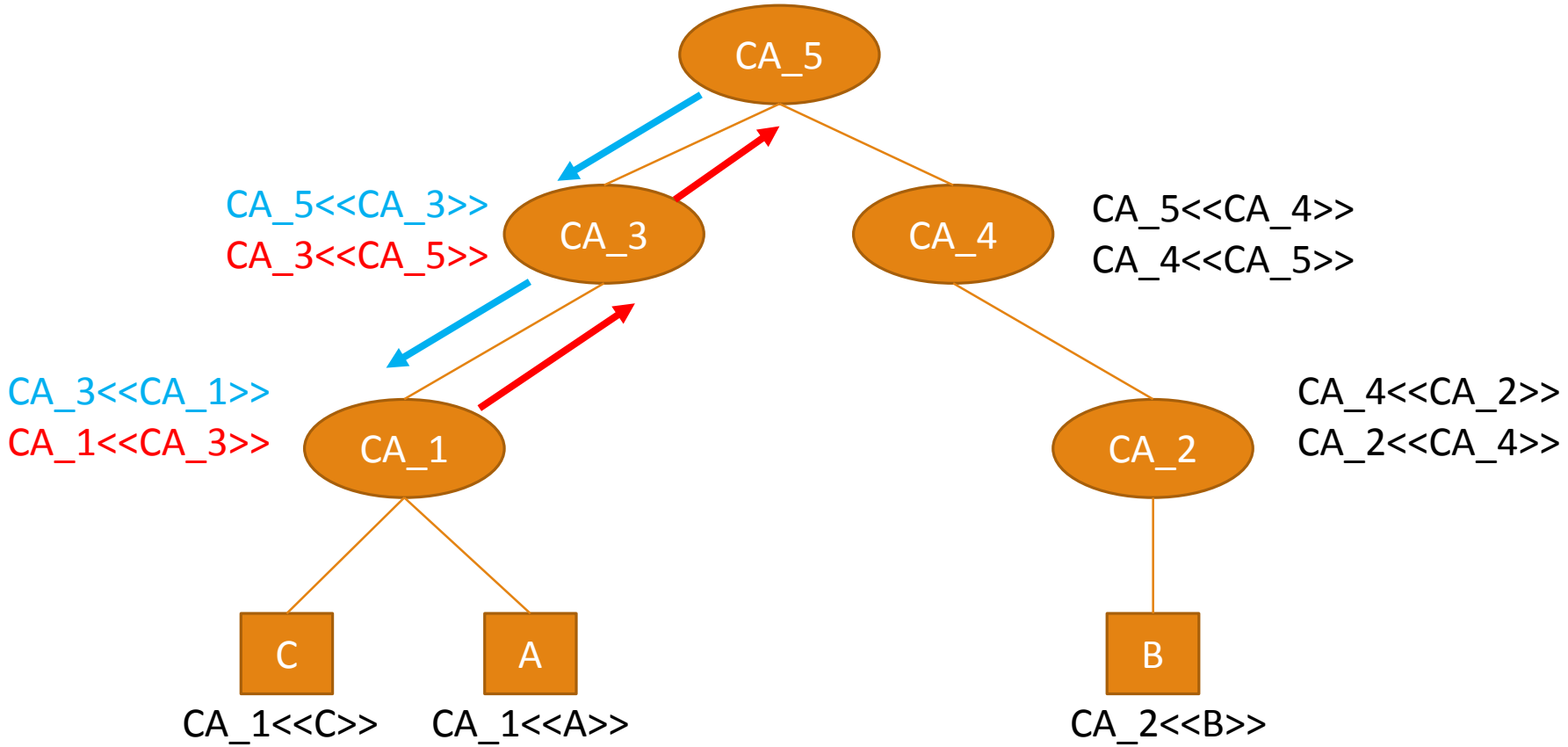
There should also be a easy way for the user to follow a path

- To facilitate this X.509 suggests to organise the CAs in a hierarchy
- Forward Certificates: Certificates of X generated by other CAs
- Reverse Certificates: Certificates generated by X for other CAs

Certificate Hierarchy

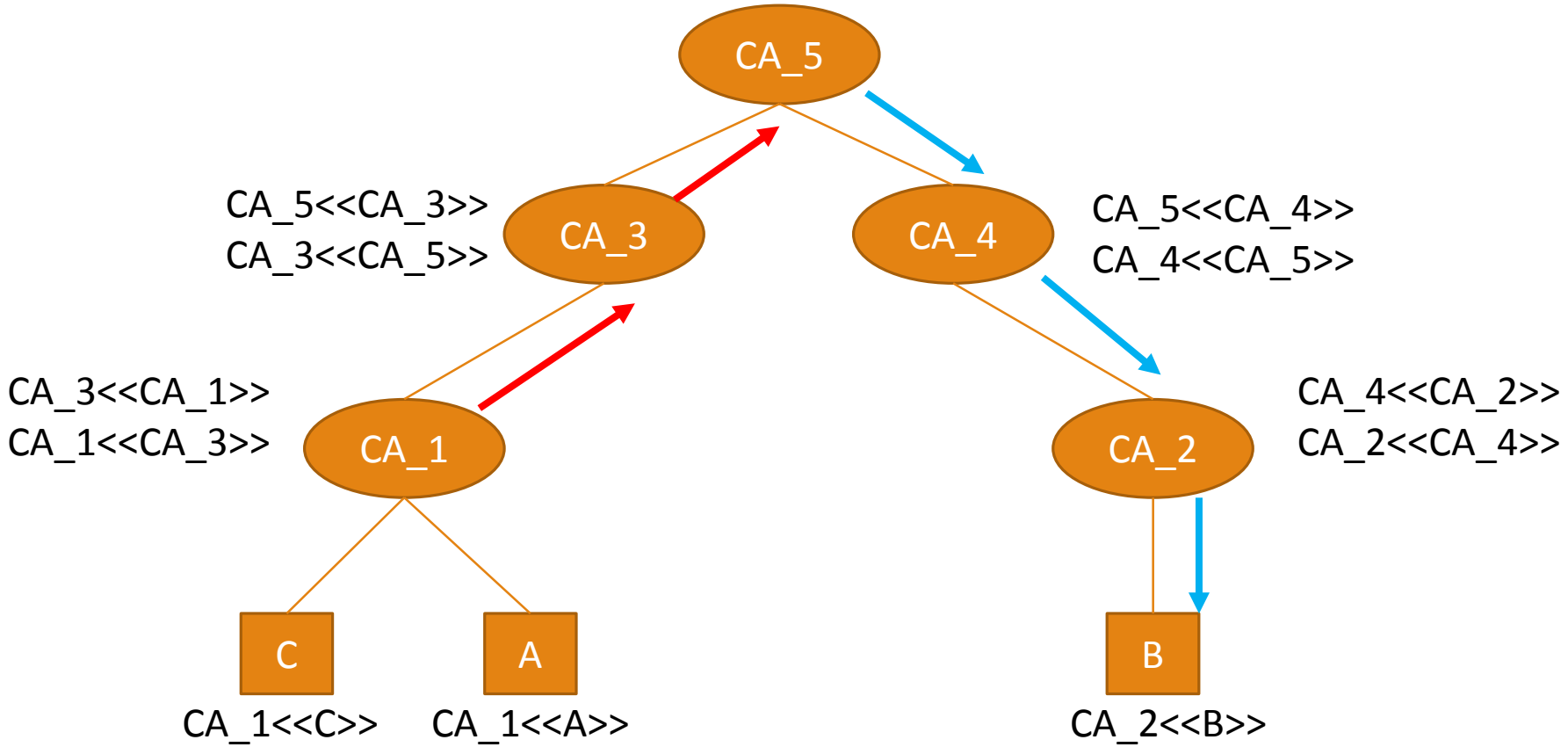


Certificate Hierarchy

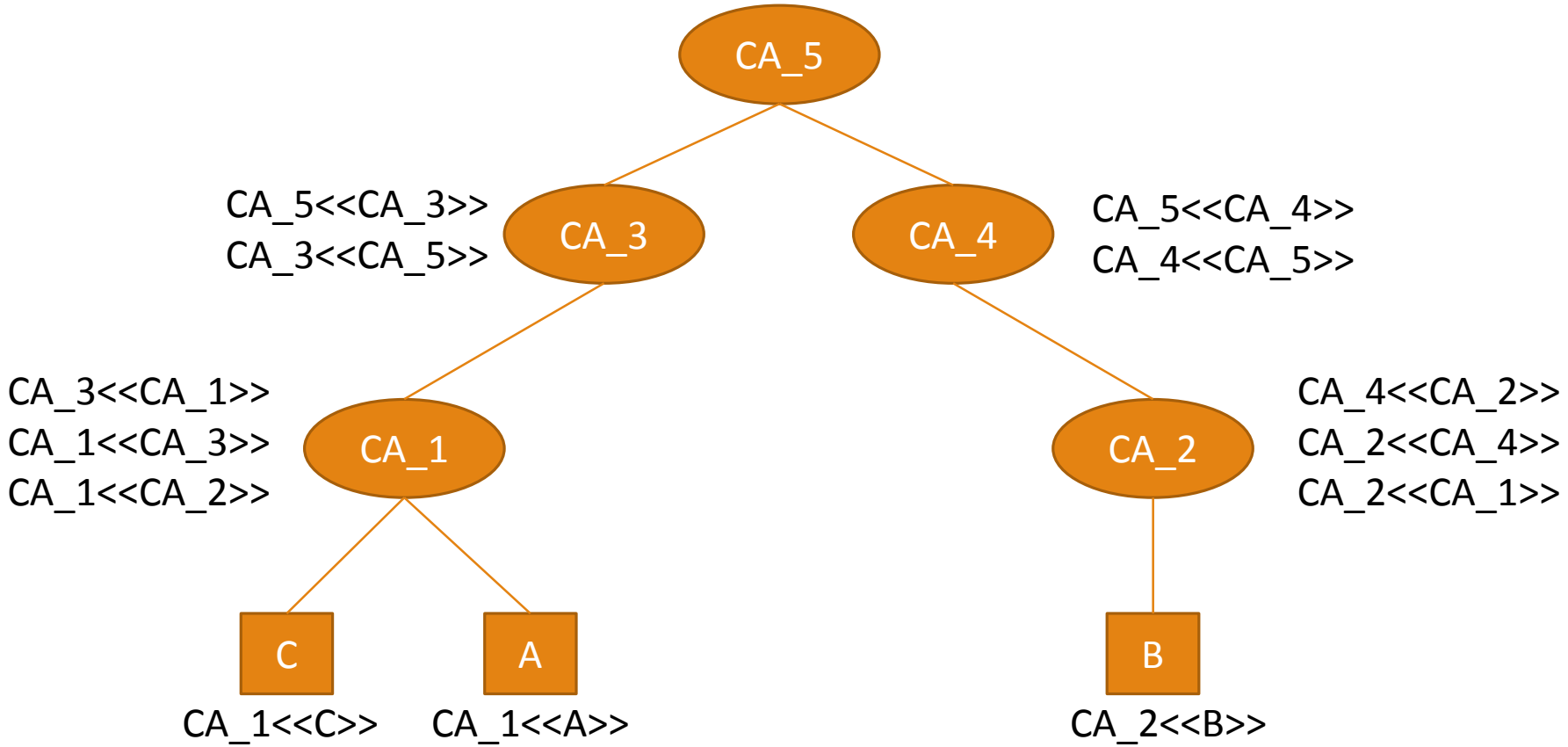


Certificate Hierarchy

CA_1<<CA_3>> CA_3<<CA_5>> CA_5<<CA_4>> CA_4 <<CA_2>> CA_2<<B>>



Certificate Hierarchy



Certificate Revocation List (CRL)

Certificates have an expiration date as credit cards

- Before they expire a new one is generated

There are cases in which a certificate needs to be revoked before its expire date

- The user's private key is assumed to be compromised
- The user's is no longer certified by the CA
- The CA's certificate is assumed to be compromised

Each CA maintains a CRL: a list of revoked but not expired certificates

The CRL is a list with the Certificate Serial Numbers of all revoked certificate signed by the CA

- It also contains the update date and the date of the next expected update

Public Key Infrastructure (PKI)

Defined by the RFC 4949, a PKI is a set of hardware, software, people, policies and procedures

- To create, store, manage, distribute and revoke digital certs based on asymmetric encryption
- To enable secure, convenient, and efficient acquisition of public keys

The Internet Engineering Task Force (IETF) maintains the PKIX, a PKI standard based on X.509 cert for providing a certificate-based architecture for the Internet

PKIX Model

End Entity: any entity (user, router, server, etc.) that can be identified in the subject field of a X.509 cert.

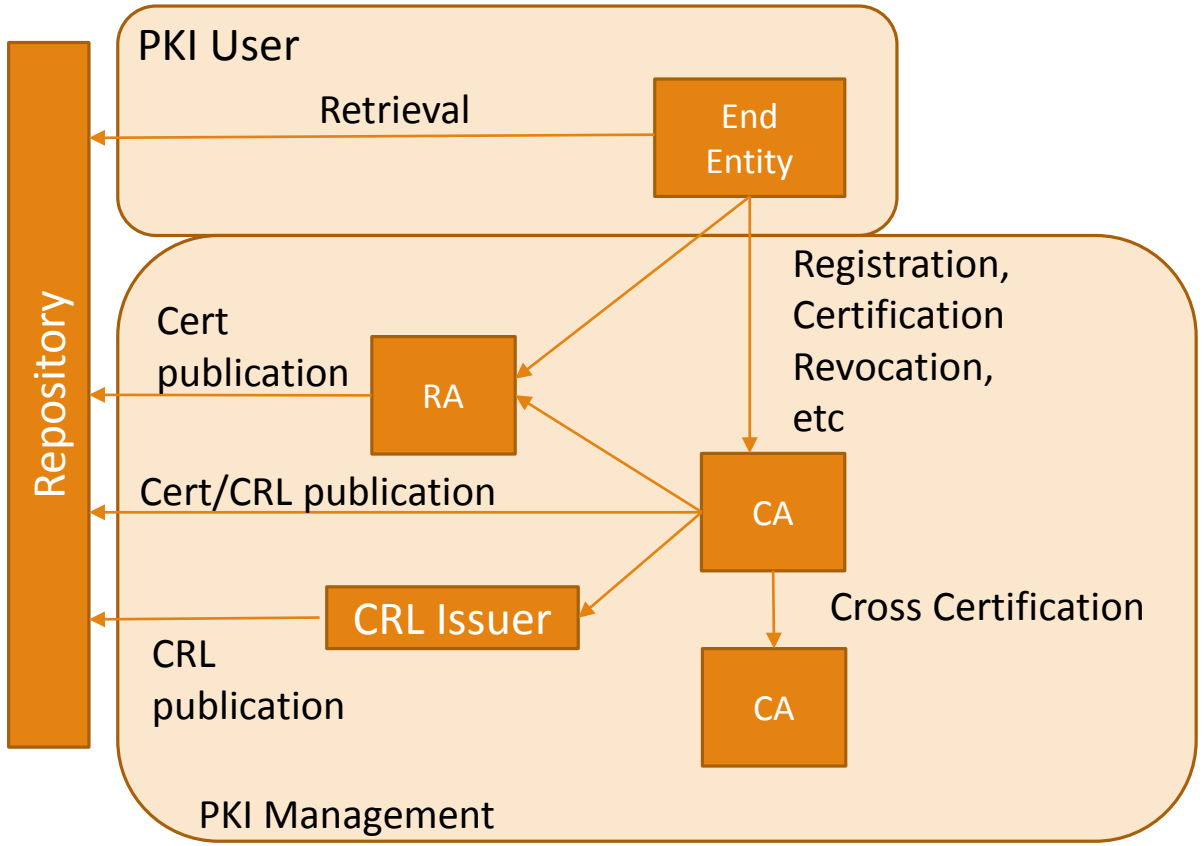
Certificate Authority (CA): issuer of certs with the responsibility of maintaining cert revocation lists

Registration Authority (RA): administrative component. Usually an RA is associated with the registration process of the end entities. CAs can also work as RAs

CRL Issuer: an optional component that CAs can use to delegate the publishing of CRLs

Repository: a generic term to denote any method for storing certs and CRLs used by end entities

PKIX Architectural Model



PKIX Management Protocols

The PKIX working group has defined two alternative management protocols between the PKI entities:

RFC 2510 defines the Cert Management Protocol (CMP) which explicitly identifies each of the management functions within the PKIX architectural model

RFC 2797 defines the Cert Management Messages over CMS (CMC). CMS (RFC 2630) is the Cryptographic Message Syntax. CMC is implemented based on existing protocols