PKI

Carlos Borrego

Carlos.Borrego@uab.cat

Departament d'Enginyeria de la Informació i de les Comunicacions
Universitat Autònoma de Barcelona

Criptografia i Seguretat

Material adaptat de:
Material de classe de Criptografia i Seguretat

Dr. Guillermo Navarro

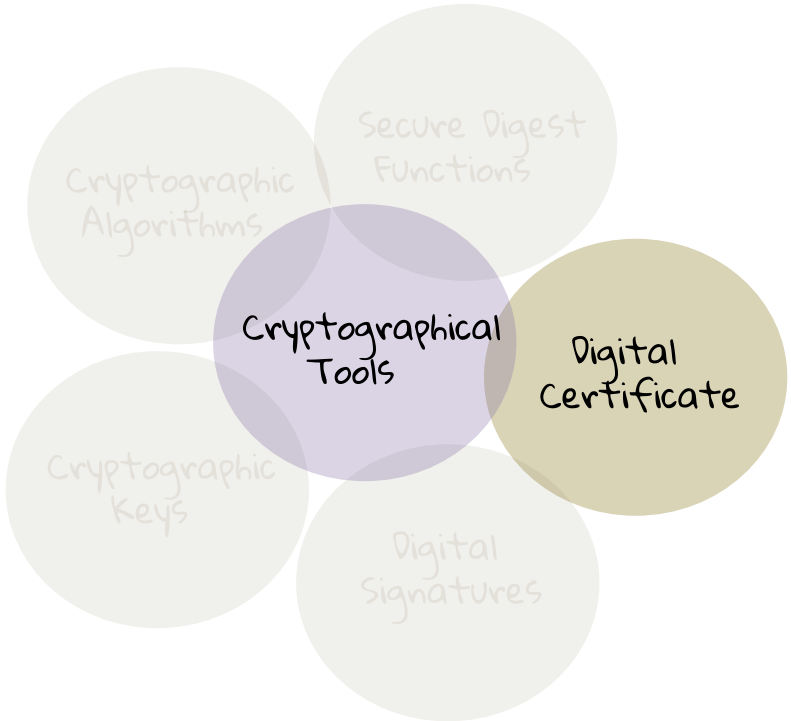
Universitat Autònoma de Barcelona

http://www.deic.uab.cat/

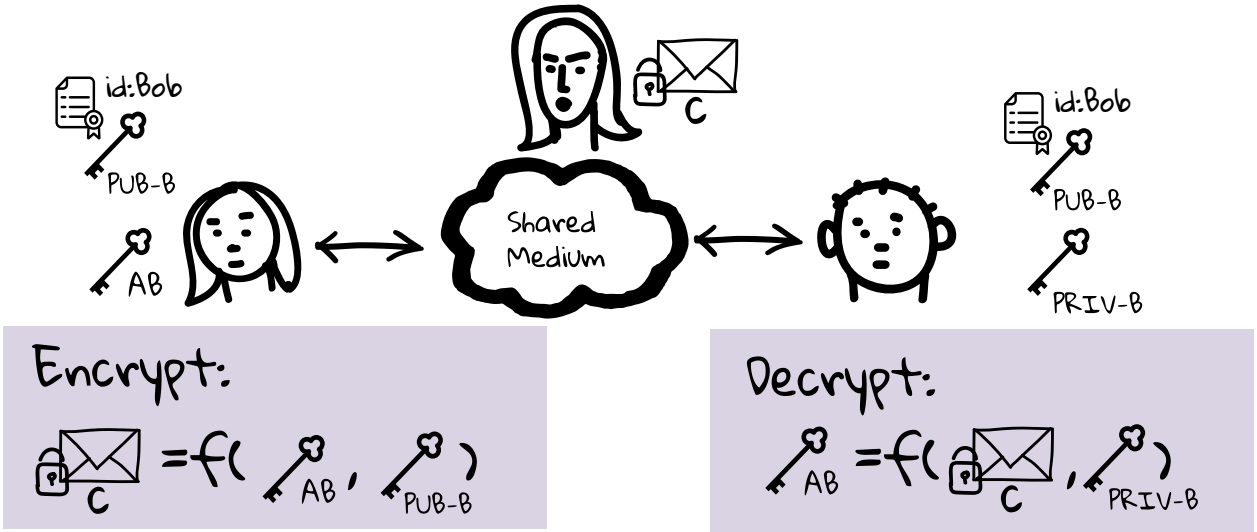
Public key cryptography in practice

- Public-key cryptography can be used both for encrypting information and ensuring the authenticity.
- However, due to the slow-working of asymmetric versus symmetric cryptosystem implementation, an asymmetric cryptosystem is never used in practice when there is a huge quantity of information:
  - **Encryption:** In order to encrypt (a huge quantity of) information, a symmetric cryptosystem is used. To send the key, it is encrypted with an asymmetric cryptosystem using the receiver public key (avoiding secret channels).
  - **Authentication and integrity:** Digital signatures involve obtaining a *hash* of the message to be signed and subsequently applying the respective private key and an asymmetric cryptosystem acting on the digest obtained.

Cryptographical Tools

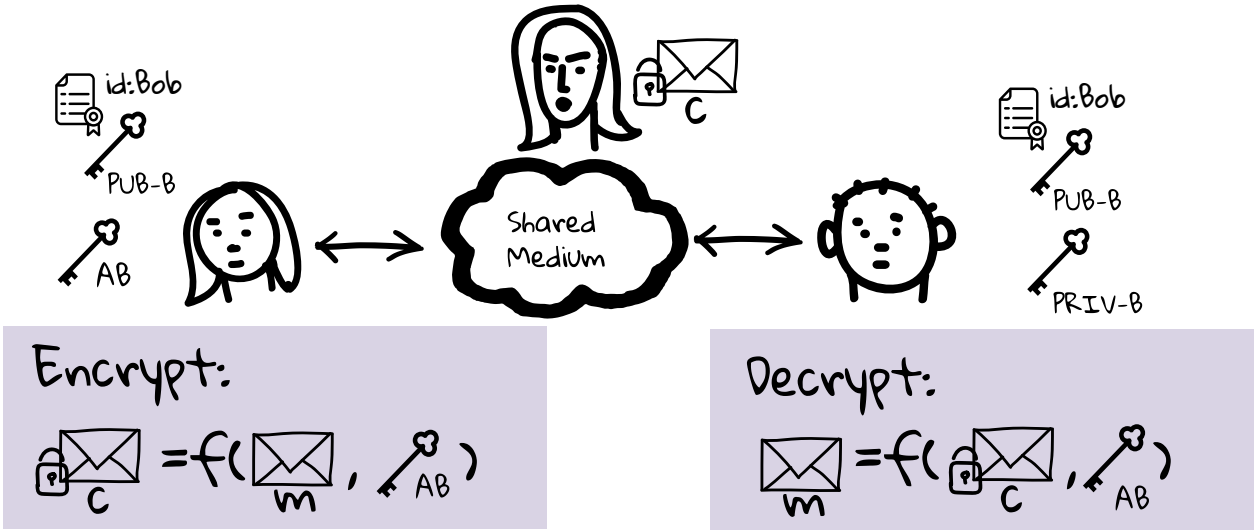


Certificates: Confidentiality



Encryption: In order to encrypt (a huge quantity of) information, a symmetric cryptosystem is used. To send the key, it is encrypted with an asymmetric cryptosystem using the receiver public key (avoiding secret channels).

Certificates: Confidentiality



Encryption: In order to encrypt (a huge quantity of) information, a symmetric cryptosystem is used. To send the key, it is encrypted with an asymmetric cryptosystem using the receiver public key (avoiding secret channels).

Who is the key owner?

Figura: The New Yorker - 1993

The problem of key distribution

We could assume that the users obtain public keys from a written trusted public directory (TPD) absolutely restricted only to a manager trusted by everyone.
This TPD-based scheme has a number of problems:

- It can become a bottleneck.
- The security of the TPD (and its administration) is critical.
- A wiretapper can intercept (and replace) the public key consulted while it is in transit from the TPD to the user that requested it.

To solve this problem, we can use **Certificates** and **Certification Authorities** (CA).

The problem of key distribution

We could assume that the users obtain public keys from a written trusted public directory (TPD) absolutely restricted only to a manager trusted by everyone.
This TPD-based scheme has a number of problems:

- It can become a bottleneck.
- The security of the TPD (and its administration) is critical.
- A wiretapper can intercept (and replace) the public key consulted while it is in transit from the TPD to the user that requested it.

To solve this problem, we can use **Certificates** and **Certification Authorities** (CA).

Digital certificates

Definition

A **digital certificate** is a data structure that contains information about the owner of cryptographic keys, the public key itself and a digital signature of both fields that validates the link.

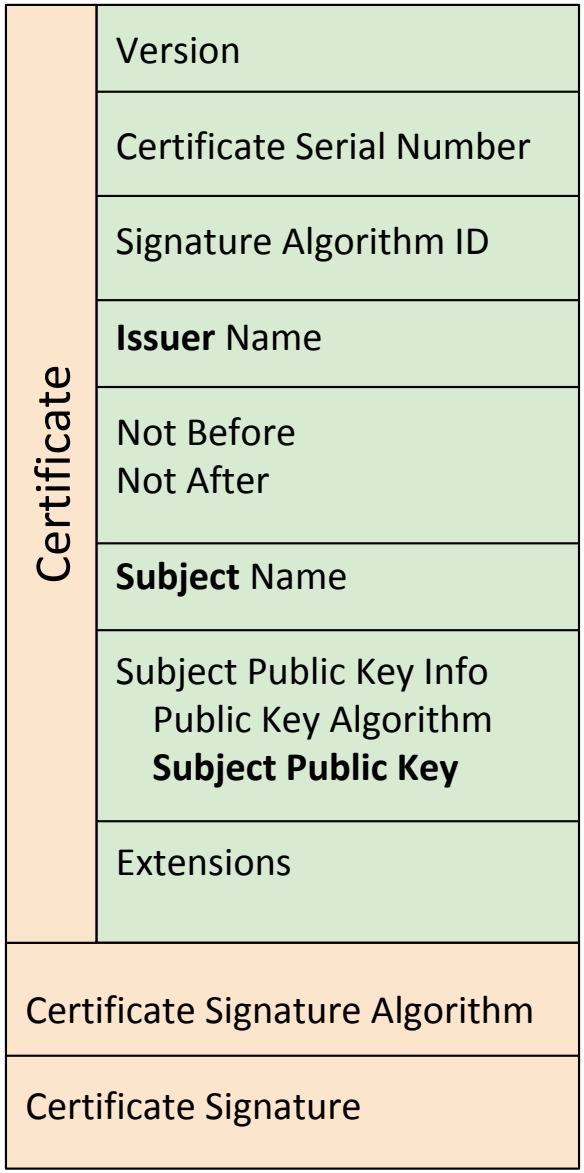
The digital signature of the digital certificate ensures the integrity of the data and the link between the public key and the owner.

- Introduced in 1978 (Kohnfelder's Bachelor's thesis)
- Digital certificates are the main tool for public key distribution.

Certificate types

- X.509 - “the standard” today
  - v.1 (1988) - not extendable
  - v.2 - not much better
  - v.3 (1997) is much better - optional extensions. Today, X.509=v.3
  - Many other standards extend X.509
- Others: PGP, SPKI, etc.

X.509 v3 (basic fields)



Certificate X.509 fields

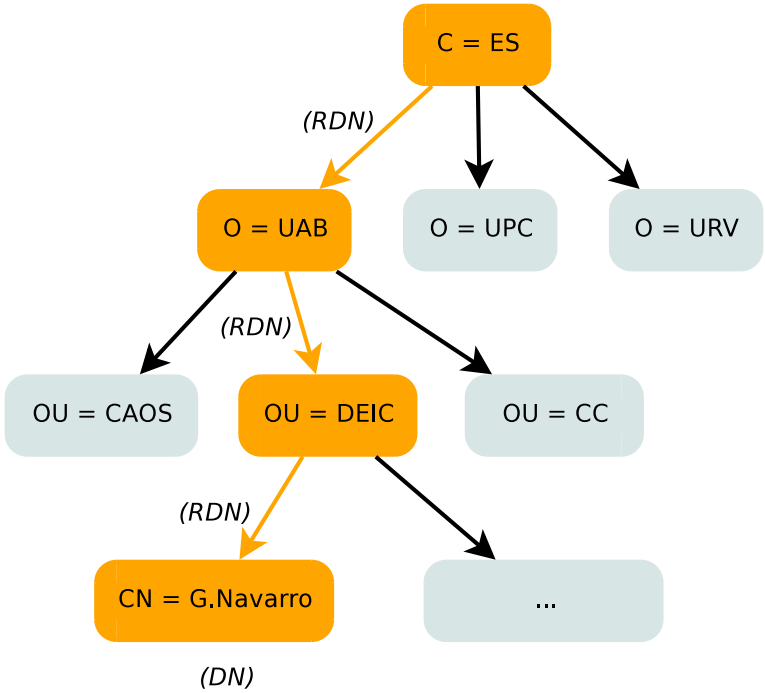
- Version: v3(2)
- Serial Number: must be unique for each CA (issuer name + serial number $\Rightarrow$ unique ID of the cert)
- **Issuer**: X.500 distinguished name (DN)
- Validity period (both included)
- **Subject**: X.500 distinguished name (DN)
- Subject **public key**
- Extensions
- Digital signature

Distinguished name

A hierarchical name composed of attributes.

- Typical X.500 DN attributes:
  - Country **C**
  - State or province **SP**
  - Locality **L**
  - Organization **O**
  - Organizational unit **OU**
  - Common name **CN**
- Typical X.500 DN: C=ES, L=Bellaterra, O=UAB, OU=DEIC, CN=Elizabeth Jennings

X.500 Directory



- Searching is performed by subtree refinement:
  - C=ES $\rightarrow$ O=UAB $\rightarrow$ OU=DEIC $\rightarrow$ CN=GNavarro
- DN are currently used as a set of attributes (NOT necessarily a hierarchical tree) with local meaning.

Example of DN: idCAT I

- CPIXSA-2 EC-idCAT: Certificat Personal d'Identificacio, Xifrat i Signatura Avançada de classe 2:
  - **Subject:**
    - Common Name (**CN**): Nom i cognoms del posseïdor de claus
    - Given Name (**G**): Nom del posseïdor de claus
    - Surname (**S**): Cognoms del posseïdor de claus
    - **Serial Number**: NIF/NIE del posseïdor de claus
    - Country (**C**): Codi de 2 lletres del país del posseïdor de claus
    - Organizational Unit (**OU**): "Serveis Publics de Certificacio CPIXSA-2"
    - Organizational Unit (**OU**): "Vegeu https://www.catcert.cat/veridCAT (c) 03"

Example of DN: idCAT II

- **Issuer**
  - Common Name (**CN**): EC-IDCat
  - Country (**C**): ES
  - Locality (**L**): Passatge de la Concepcio 11 08008 Barcelona
  - Organization (**O**): Agencia Catalana de Certificacio (NIF Q-0801176-I)
  - Organizational Unit (**OU**): Serveis Publics de Certificacio ECV-2
  - Organizational Unit (**OU**): Vegeu https://www.catcert.net/verCIC-2 (c)03
  - Organizational Unit (**OU**): Entitat publica de certificacio de ciutadans

Example of DN: CERES FNMT

Certificado de identidad de Persona Física

Subject DN:

- **CN** = NOMBRE a1 a2 n - NIF 12345678A
  - n, a1 a2: nombres, primer y segundo apellido
  - 12345678A: NIF
- **OU** = FNMT Clase 2 CA
- **O** = FNMT
- **C** = ES

Issuer DN

- C=ES, O=FNMT, OU=FNMT Clase 2 CA

Note that currently this has changed a bit in the last years

X.509v3 Certificate Extensions

- Optional sequence of extensions
- To types: critical / non-critical
  - **non-critical** extension my be ignored if not recognized
  - **critical** extensions must be recognized (otherwise the certificate must be rejected).

x.509v3 standard extensions

Mandatory:

- Basic Constraints
- Authority Key Identifier
- Subject Key Identifier
- Key Usage
- Subject Alternative Name (mandatory if subject is empty)
- Certificate Policies

Optional:

- Policy Mappings
- Issuer Alternative Name
- Subject Directory Attributes
- Name Constraints
- Policy Constraints
- Extended Key Usage
- CRL Distribution Points
- Inhibit anyPolicy
- Freshest CRL (a.k.a. Delta CRL Distribution Point)

Practical Problems

- How are Digital Certificates Issued?
- Who is issuing them?
- Why should I Trust the Certificate Issuer?
- How can I check if a Certificate is valid?
- How can I revoke a Certificate?
- Who is revoking Certificates?

PKI main objective

Definition

A **Public Key Infrastructure** (PKI) is all software and hardware components together with users, policies and procedures that allow the creation and management of digital certificates based on public key cryptography.

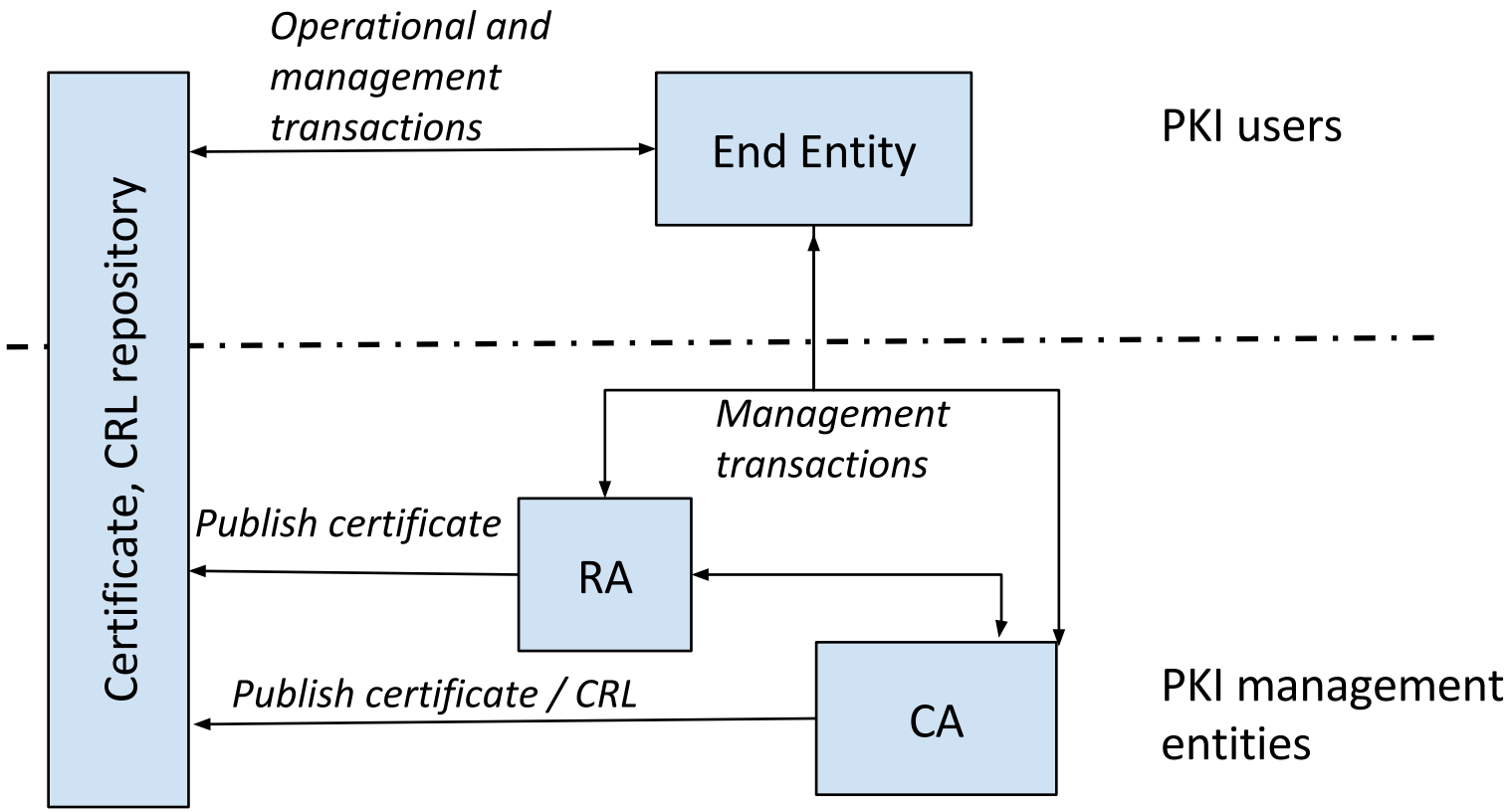
The main PKI objective is the efficient and trustworthy management of cryptographic keys and certificates that can be used for authentication, integrity, confidentiality and non-reputation purposes.

PKI standards

The interoperability through different PKI applications is obtained through the use of PKI standards.

- Internet Engineering Task Force (IETF): Request for Comments (RFC) documents (RFC 5280, etc.)
- International Telecommunication Union / International Standards Organisation (ISO).
- European Telecommunications Standards Institute (ETSI).
- RSA Laboratories. PKCS documents. *de facto* standards.

PKI Entities I



- Users (end entity)

PKI Entities II

- CA, Certification Authority
- RA, Registration Authority
- Repositories: certificates, CRLs
- Validation Authority

Users

Definition

A **subscriber**, or certificate holder, is the one that owns a private and public key certificate together with a digital certificate of the public key.

Definition

A **relaying party** is a certificate recipient, a user that can validate digital signatures produced by subscribers. Clients can also encrypt information for subscribers.

Certification authority

Definition

A **certification authority** (CA) is an entity which issues and revokes digital certificates. The CA is the trusted entity that legitimizes the link between the user and its public key.

- The core of a CA are the private and public keys that will be used to issue digital certificates and to validate them.
- The CA public key distribution is performed through a **self-signed certificate** that is signed by the CA. Such certificates should be transmitted using an authentic channel.
- A PKI may have one or several certification authorities, depending on the trust model used.

Registration authority

Definition

A **registration authority** (RA) is an entity that verifies the link between the user and its public key.

- The RA is an optional entity that can reduce the workload of the CA.
- For non-high secure applications, the RA can be automatic through a simple mail owner check.

Repositories

Definition

Repositories are data structures that contain information related to a PKI. The main repositories of a PKI are the certificate repository and the certificate revocation list (CRL).

Standards:

- The X.500 directory based on the OSI architecture is the most common used directory in PKI.
- DAP and LDAP (Lightweight Directory Access Protocol) are the protocols used to access to a X.500 directory.

Repositories do not need to be trusted

- Certificates and CRLs signed by CA

CRLs I

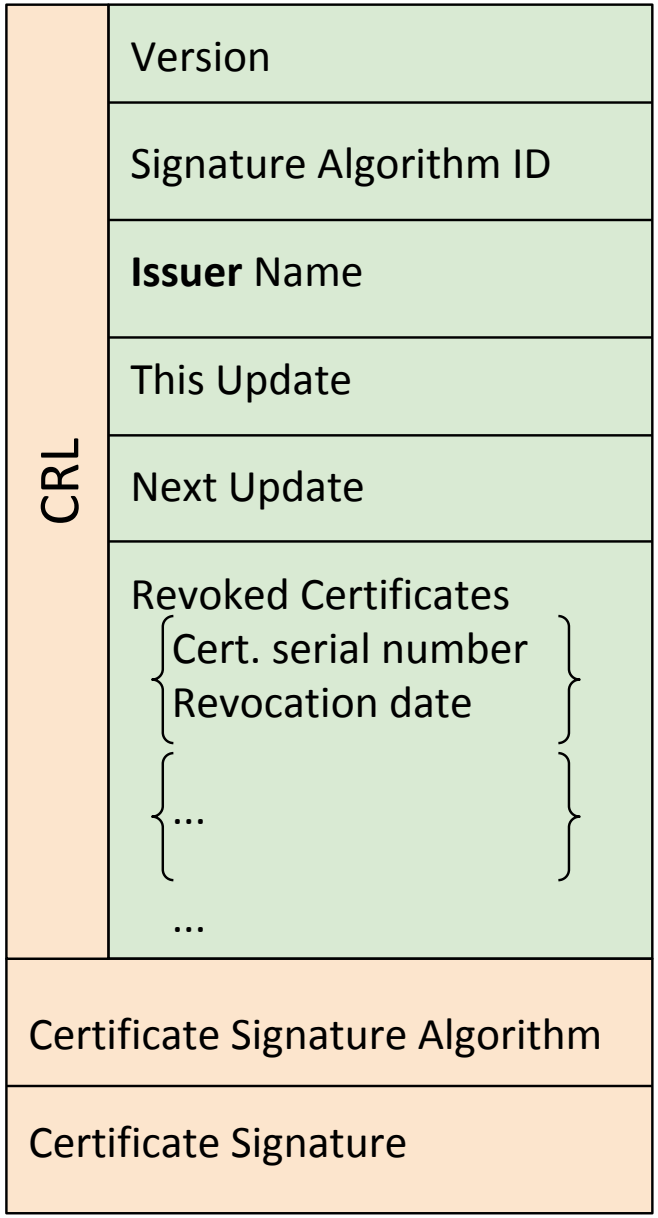
Definition

A **certificate revocation list** (CRL) includes all digital certificates that are not valid before its expiration date.

CRLs fileds:

- Issuer (CA) Distinguished Name (DN).
- Time of this update
- Time of next update
- List of revoked certificate serial numbers with dates and reasons
- CA signature

CRLs II


CRL
Version
Signature Algorithm ID
Issuer Name
This Update
Next Update
Revoked Certificates
Cert. serial number
Revocation date
...
...
Certificate Signature Algorithm
Certificate Signature

CRL types

Complete CRLs:

- Advantages: Self-contained, simple, complete.
- Problems:
  - Scalability: CRL may grow too big
  - Timeliness: Also results from CRL size
- Conclusion: appropriate for some domains

Delta CRLs:

- Incremental change
  - From Complete or Partition CRL
  - CRL_{new} = $BaseCompleteCRL_{old}$ + DeltaCRL
  - Possibly many DeltaCRLs from same BaseCRL (E.g. complete CRL issued once a week, and a new DeltaCRL (containing the previous DeltaCRLs) issued every day)

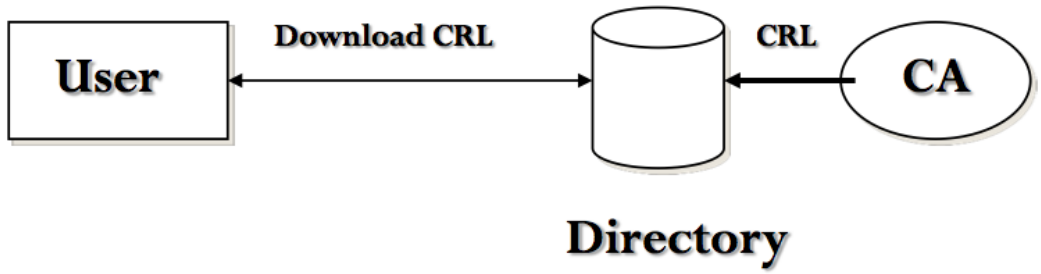
Validation authority

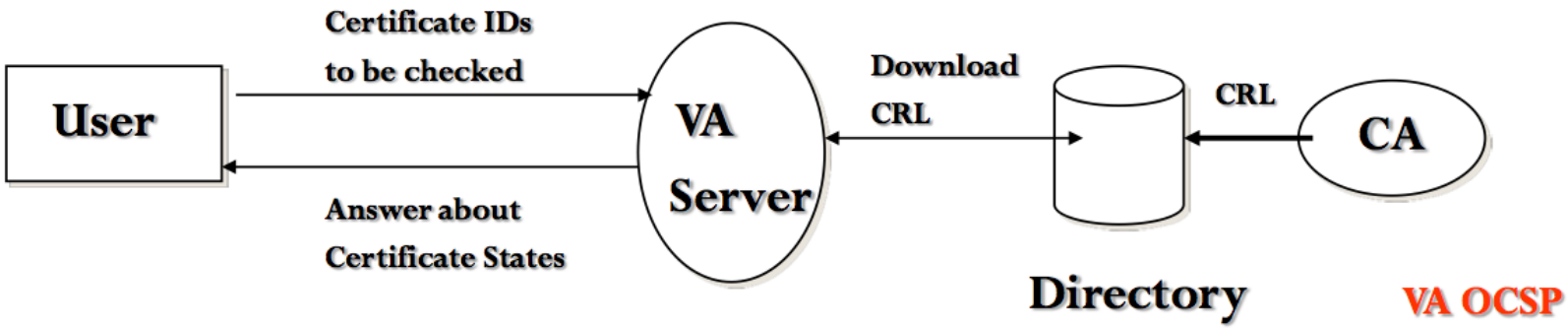
Definition

The **validation authority** (VA) is the one that checks the certificate validation.

- Like the RA, the VA is an optional entity that can reduce the workload of the CA.
- The validation protocols are mechanisms that give information about the certificate status (valid, revoked, ...) or information related to the certification chain needed to validate the certificate.
- The two main certificate validation protocols are: Online certificate status protocol (OCSP) and Simple certificate validation protocol (SCVP).

VA vs CRL


User
Download CRL
CRL
CA
Directory
CRL


User
Certificate IDs
to be checked
Answer about
Certificate States
VA
Server
Download
CRL
CRL
CA
Directory
VA OCSP

Keys and certificate life cycle

- Key generation
- Registration
- Certification
- Certificate revocation
- Certificate renewal

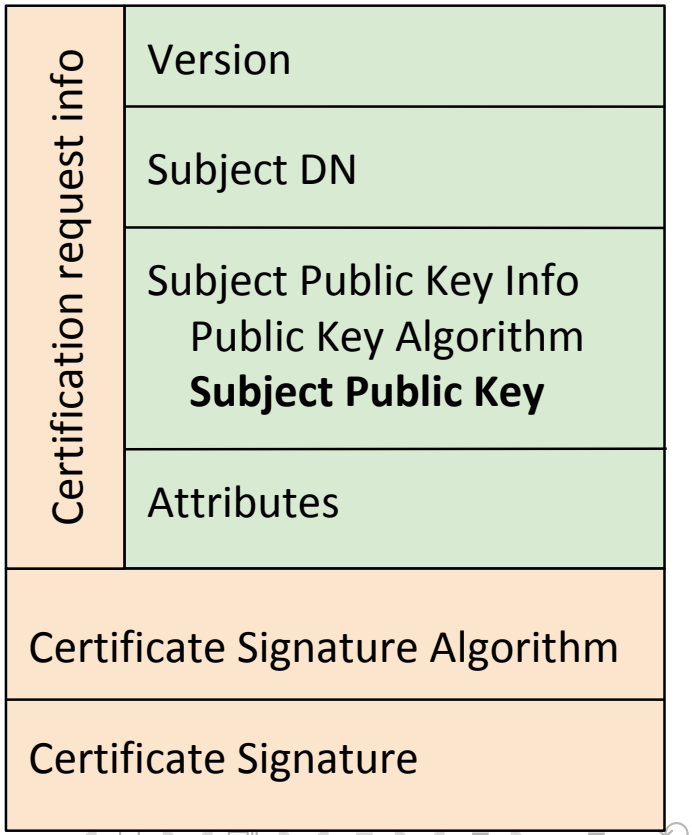
Key generation

- A pair of public and private keys is generated.
- Such generation can be performed by the user or by a trusted party (or computer) provided the private key will not be disclosed.

Registration

- The user registers his public key which means that he holds the corresponding private key.
- The registration authority is the one that verifies such registration process.

- Usually involves a Certificate Signing Request (CSR) or certification request:



Certification

- The certification authority legitimizes the link user-public key verified by the registration authority.
- A digital certificate proves such link.

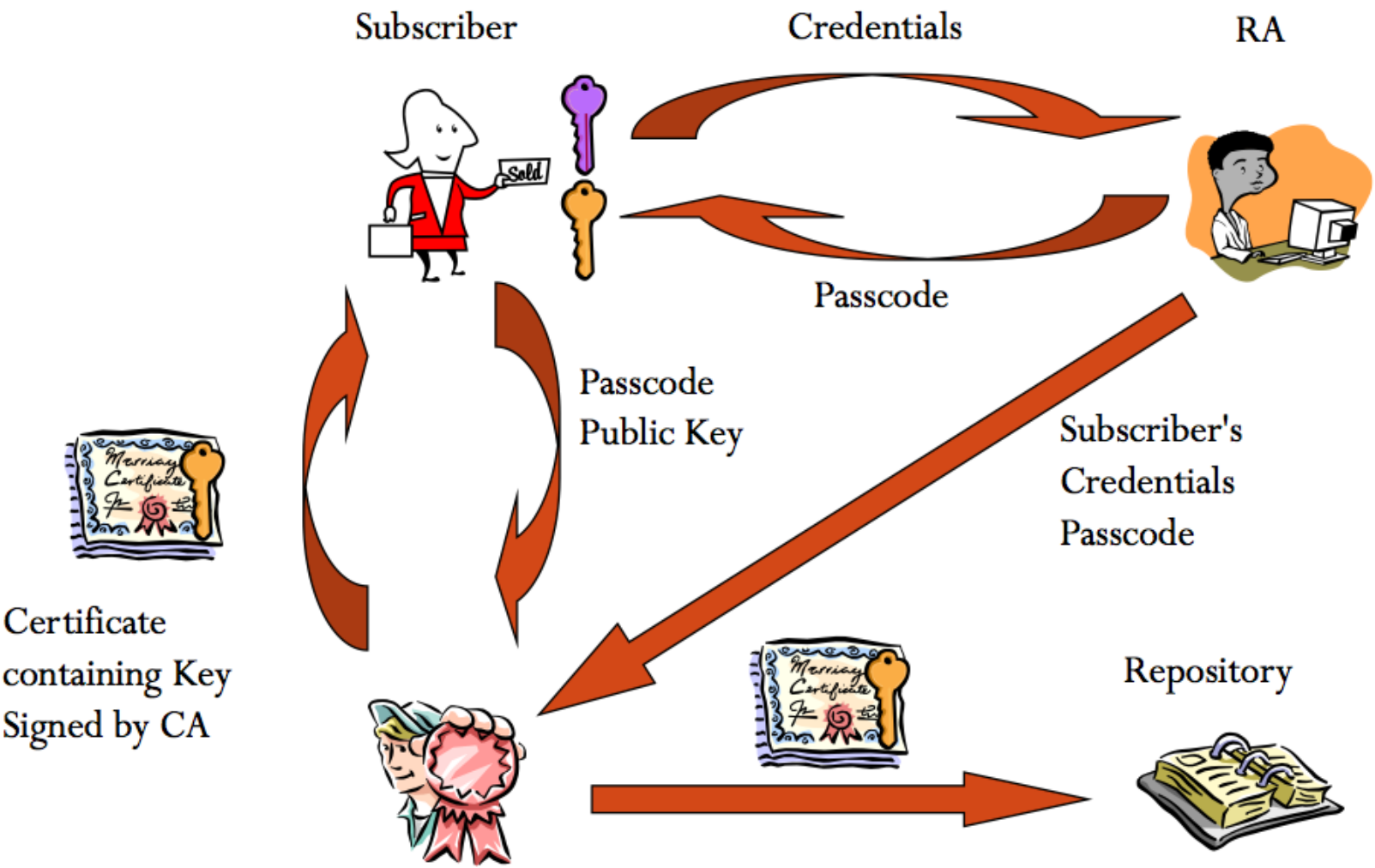
Certificate revocation

- In case of private key compromise, the user informs to the certification authority that the uniqueness link between public-private keys and user identity has been broken.
- The certification authority revokes the corresponding certificate by including it in a certification revocation list.

Certification renewal

- Digital certificates include an expiration date.
- Certification renewal must be performed in order to get a new certification with another expiration date.

How a PKI issues certificates



How certificates are used

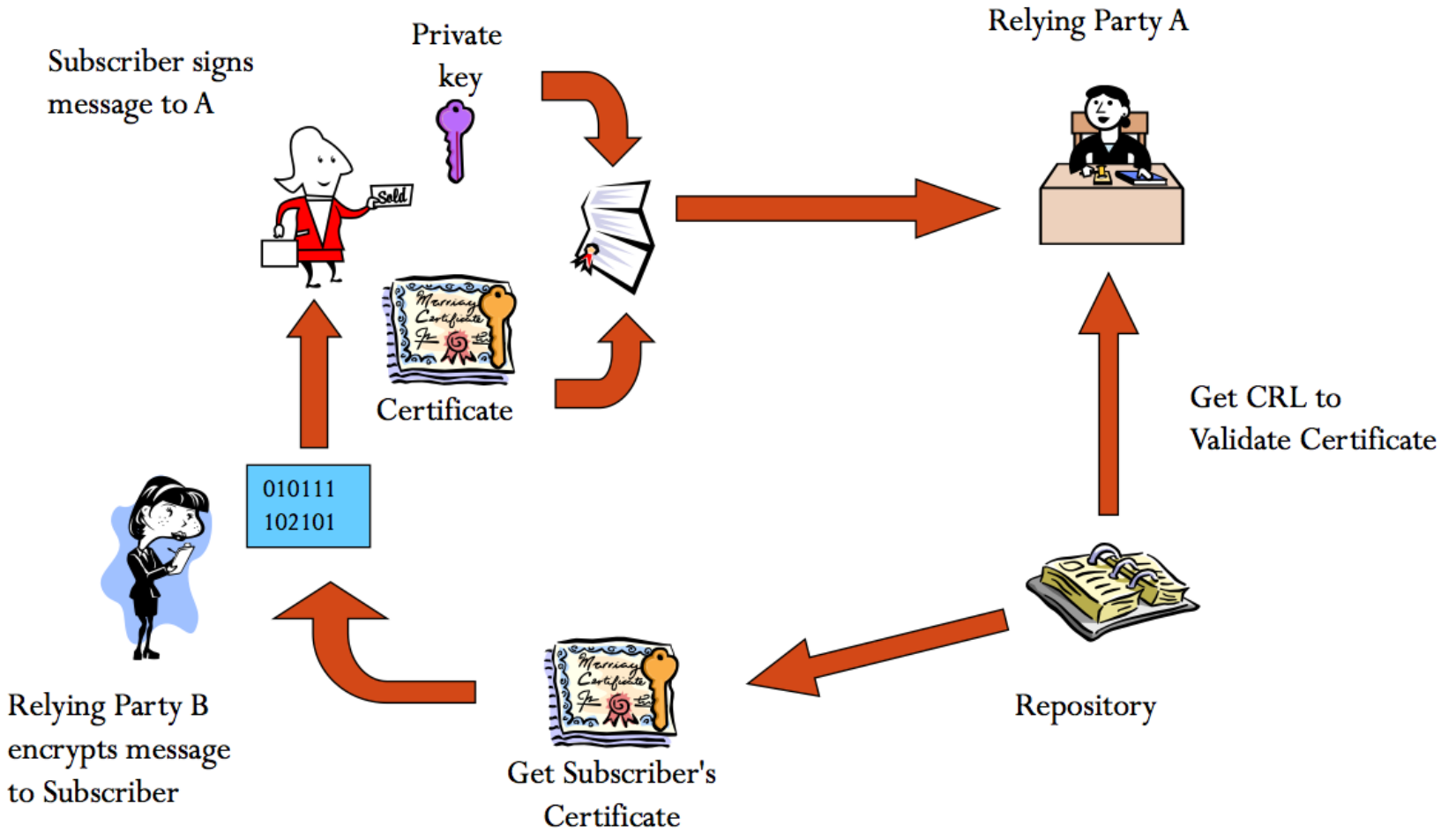


Figura: Using certificates.

Plain model

The plain model is the simplest one where **only one** CA exists.

- Any user can validate a digital certificate using the self-signed CA certificate, a digital certificate where the issuer and the CN of the subject are the same, and the public key included is the same that the one used for validate the certificate.
- Such plain model is often used in intranets environments.

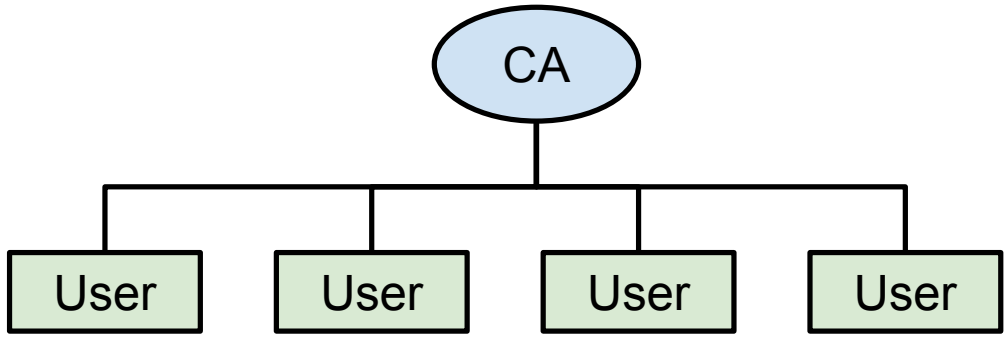


Figura: Plain model

Hierarchical model

Also known as **subordinate hierarchy**.

- In an hierarchic model, each CA is certified by another CA at the immediately upper level. These certificates are known as *hierarchical certificates*. In such certificates both the issuer and the CN of the subject are CAs but different.
- The top level CA is known as a **root CA**.
- The other CAs are intermediate or subordinate CAs.
- This kind of structure provides certificate chains or paths between individual users, hence guaranteeing a common point of trust. The only key that remains uncertified is the root public key, which must be widely broadcast (for example, in the official gazette, etc.).

Graphic of certification hierarchy

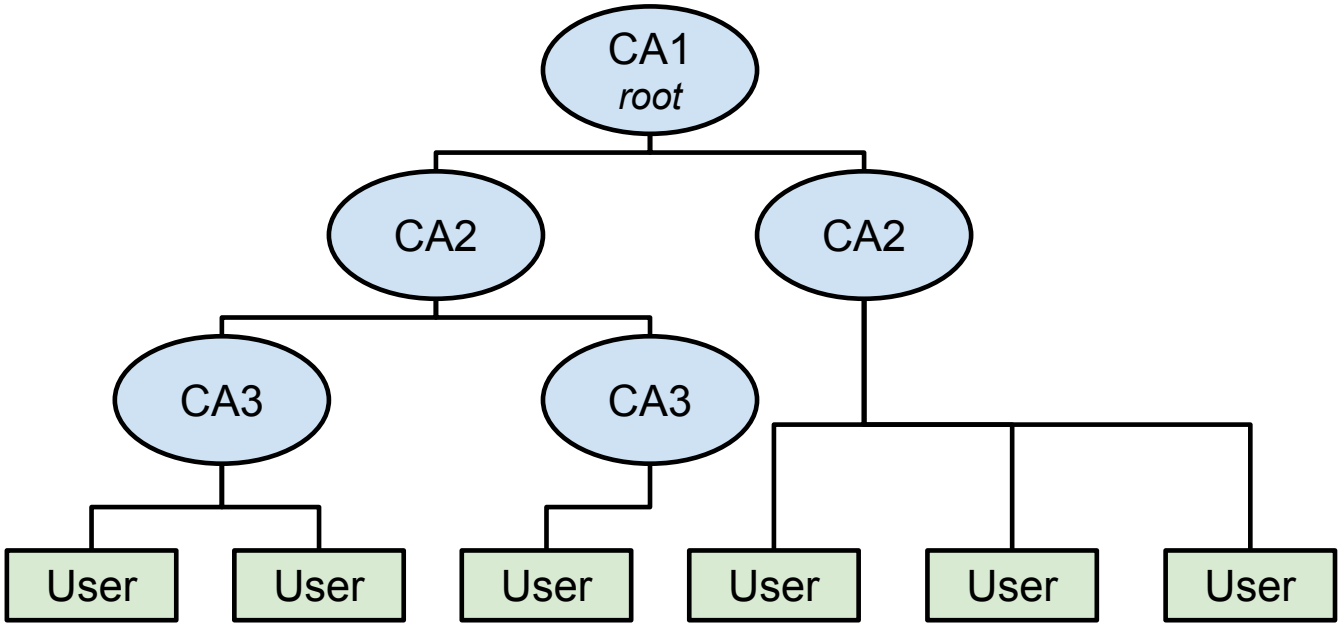


Figura: Hierarchical model.

Trust list

- Assuming a unique root CA for all PKI applications and domains is unrealistic.
- The PKI trust list model, or user centered model, every application holds a list of root CAs is which the application trust.
- Each root CA can be part of a plain model or part of a hierarchic model.
- Such root CA lists should be protected against external modifications.
- A typical examples are Web browsers.

Graphic of trust list

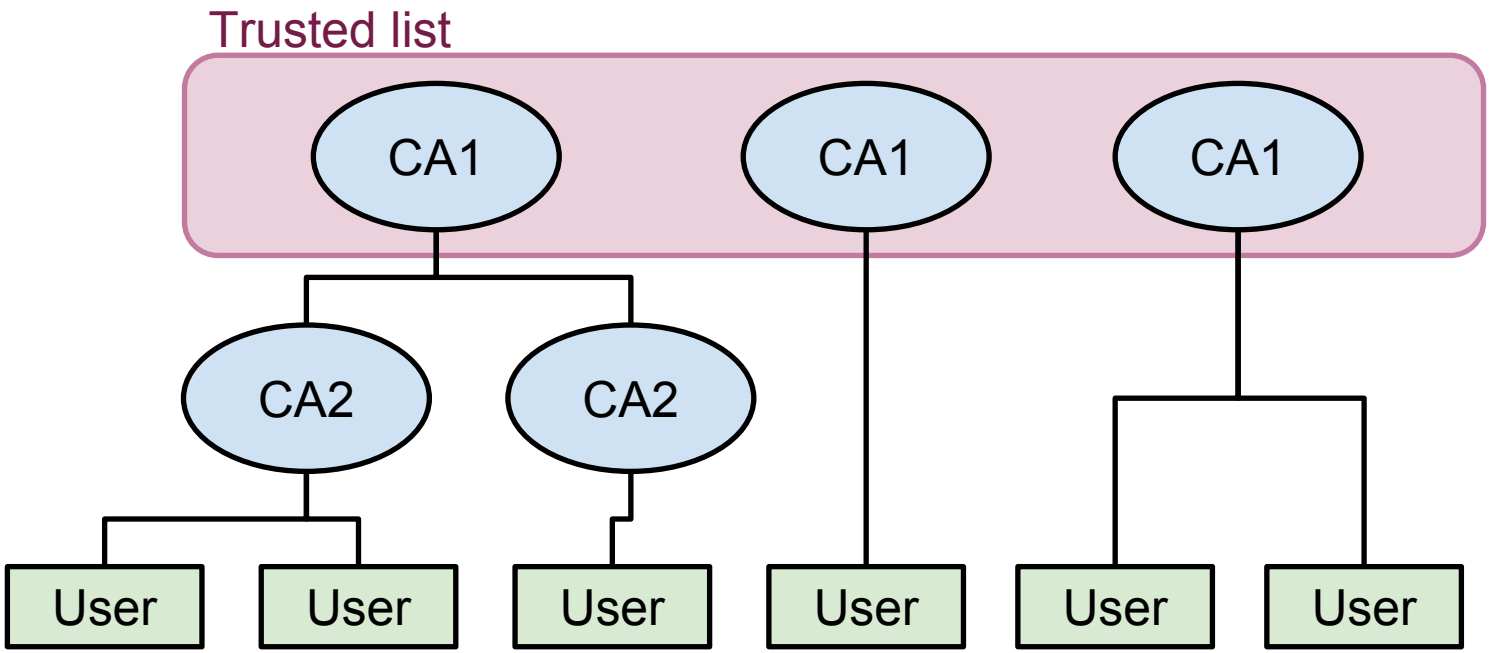


Figura: Trust list model

Cross certification

- In a cross certification model, every root CA issues a certificate for others root CAs.
- In a cross certificate, both the issuer and the CN of the subject are CAs but they are different and they belong to a different CA hierarchy.
- In applications that use a cross certification model, the user cannot add new CAs unless such new CA is certified by an already recognized one.
- This is also know as mesh structure (or mesh PKI) when only cross-certification is used.

Graphic of cross certification

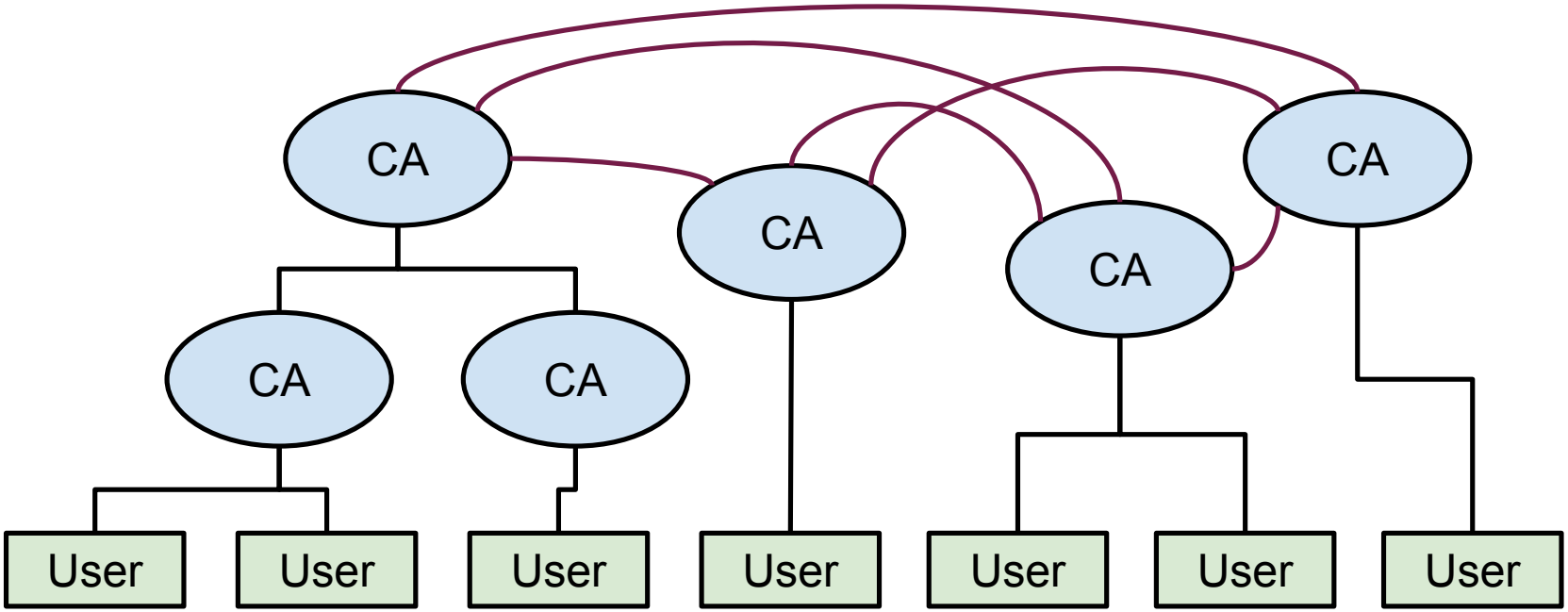


Figura: Cross certification model

Bridge CA

- The problem of a cross certification model is that the number of cross certificates grows to much when the number of CA involved is moderate (usually exponentially).
- A bridge CA includes an external CA that will act as a bridge between different CAs.
- The model is similar to a Hierarchic model, however the bridge CA is not defined as a central trust point, it is only an interconnection point.
  - The Bridge CA cross-certifies with one CA in each participating CA (usually a Root CA).

Graphic of bridge CA

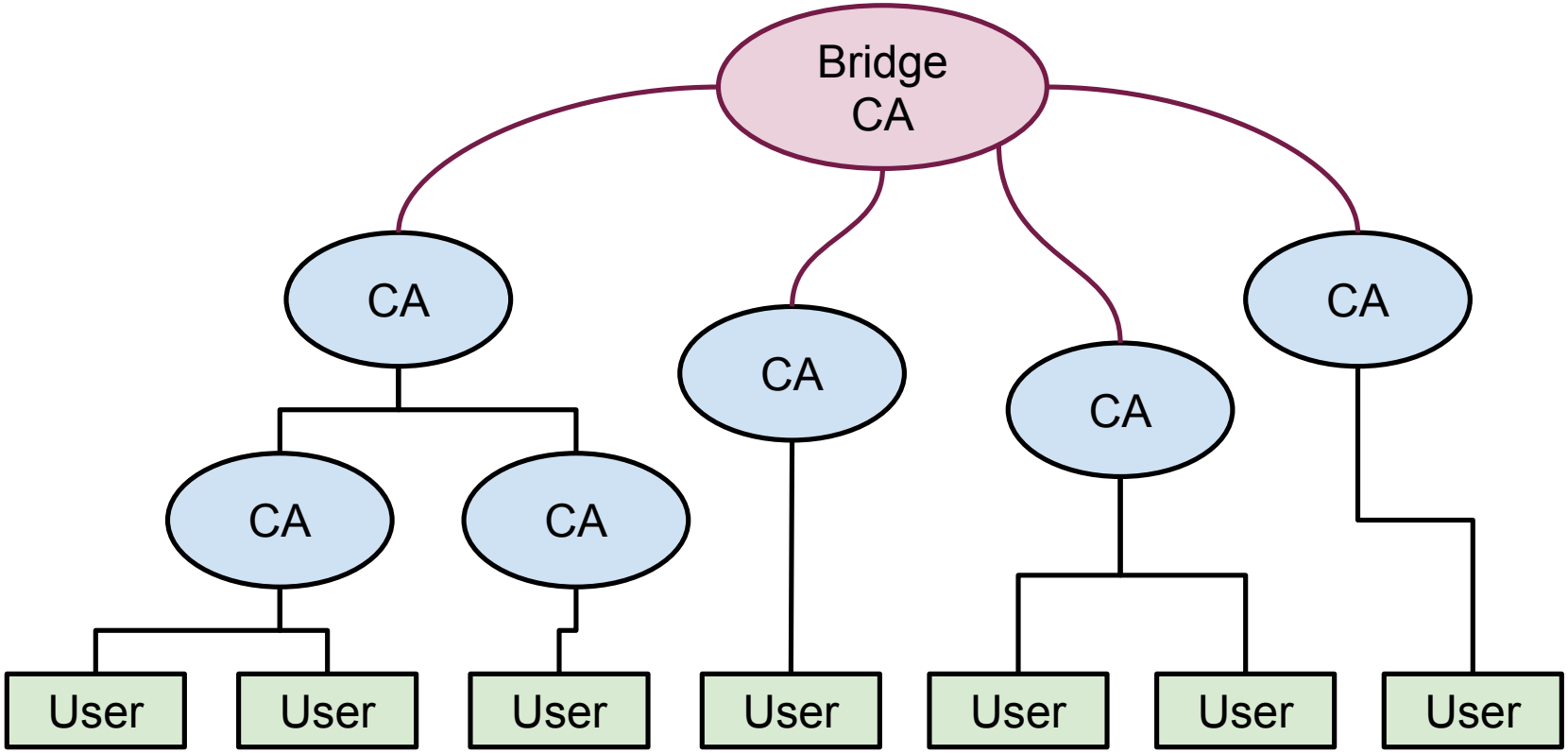
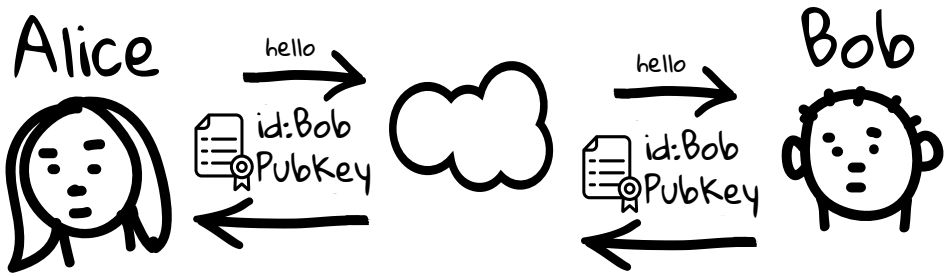


Figura: Bridge CA model

Certificates Authentication

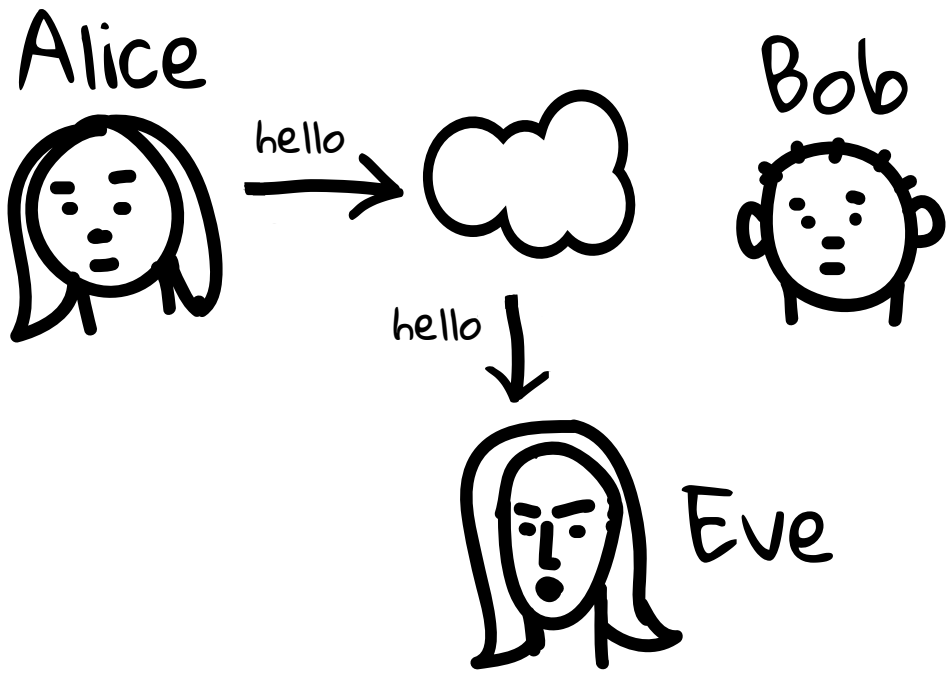
Alice asks for the other's certificate:



Is this enough to prove Bob's authentity? hmmmm....

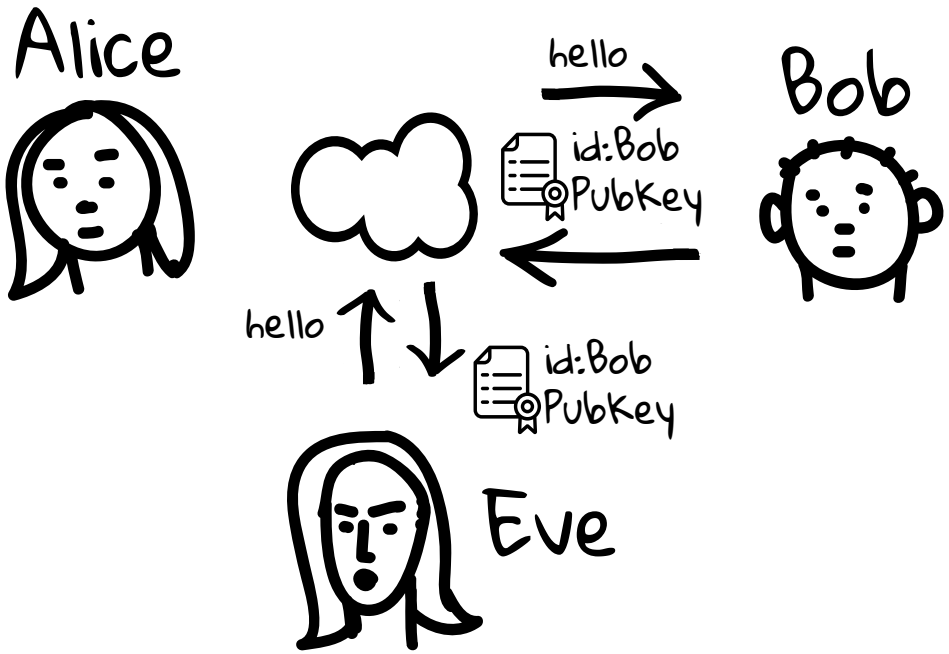
Certificates Authentication

Remember! You may not be talking to who you think:



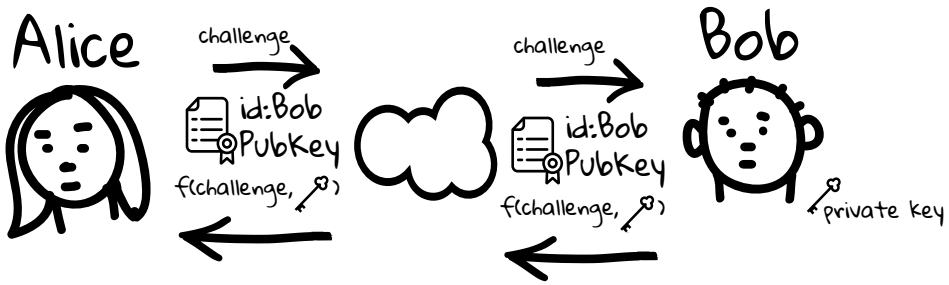
Certificates Authentication

Careful! Eve, can obtain Bob's certificate:



Certificates Authentication

First Challenge! Prove you have the private key!:



Is this enough to prove Bob's autenticity? hmmm...

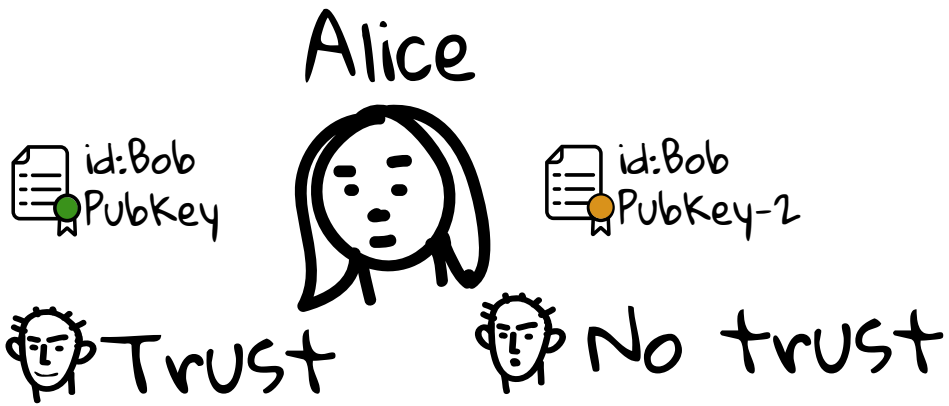
Certificates Authentication

Careful!

Eve can create a certificate with Bob's identity and its corresponding private key!:

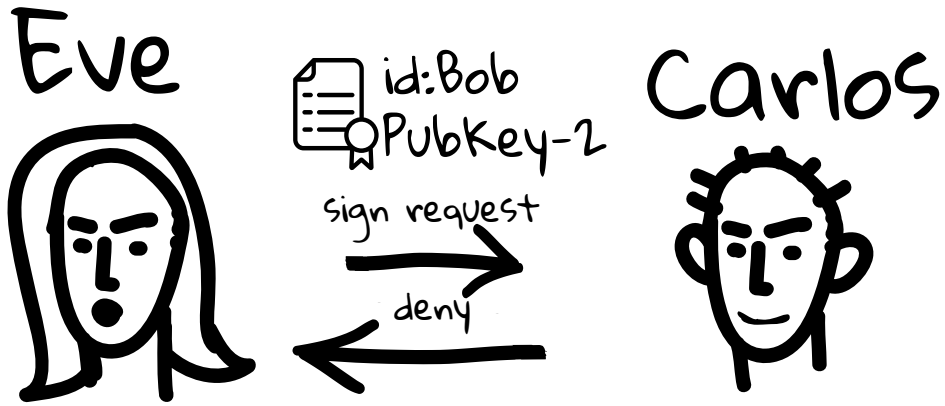
Certificates Authentication

Second challenge! Your certificate must be signed by someone with a reputation!:



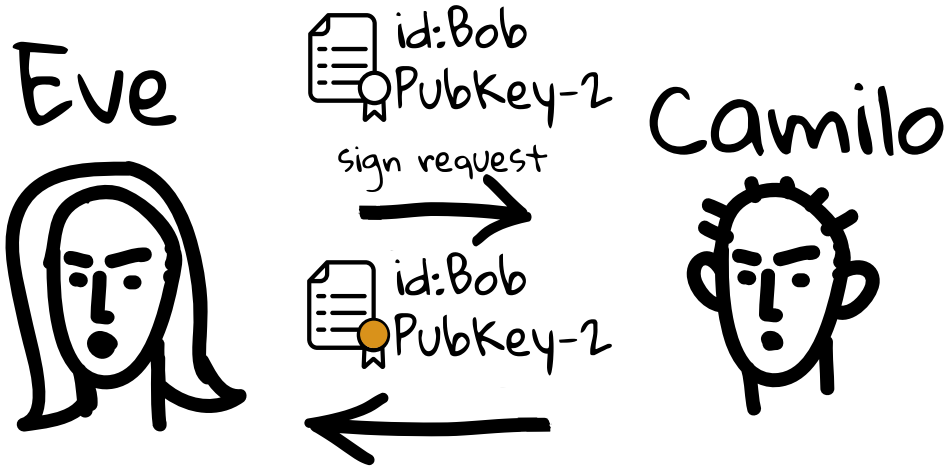
Certificates Authentication

A CA with a reputation will not sign something **thoughtlessly**:



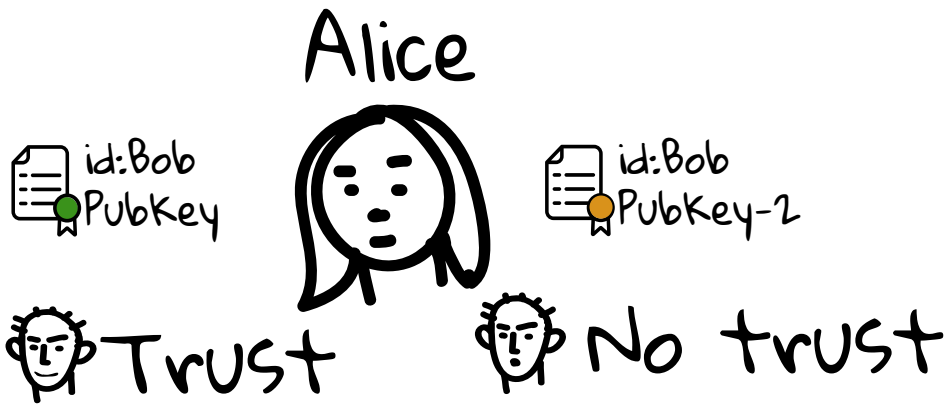
Certificates Authentication

A CA with no reputation will sign something **thoughtlessly**:



Certificates Authentication

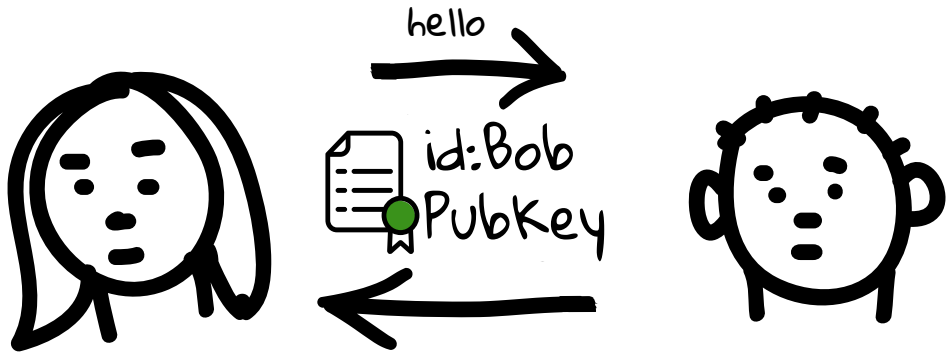
Alice will not **accept** a certificate signed by someone without reputation!



Certificates Authentication

Double security! Correspondence:

- Public-private key
- Certificate id and entity



Distributed model

- In a distributed model every user acts as a CA.
- Public keys can be signed by multiple users/CA.
- This model is known as a web of trust.
- The idea is that *the friends of my friends are my friends*.
- This model is based on the paradigms of small-world networks.

Example of distributed model: OpenPGP - RFC4880 I

- Based on the software PGP - Pretty Good Privacy released with source code in 1991.
- OpenPGP main purpose is mail encryption but can be used for general file ecryption and signature.
- OpenPGP does not provide a CA protocol but a trust management system based on the idea that *the friends of my friends are my friends*.
- Each public key has a field, **key trust value**, to indicate how much we trust the owner to certify public keys.

Example of distributed model: OpenPGP - RFC4880 II

- GnuPG is the most common used implementation of OpenPGP.
- Public key algorithms:
  - RSA, ElGamal, DSA
- Symmetric algorithms:
  - IDEA, 3DES, CAST5, Blowfish, AES, Twofish
- Hash functions:
  - MD5, SHA1, RIPE-MD160, SHA256/224,384,512
- Compression:
  - ZIP, ZLIB, BZip2

Key trust values

Each principal maintains a key ring:

trust values	meaning of this trust value
untrusted	Nothing is known about the owner's judgment in key signing. Keys on your public keyring that you do not own initially have this trust level.
none	The owner is known to improperly sign other keys.
marginal	The owner understands the implications of key signing and properly validates keys before signing them.
full	The owner has an excellent understanding of key signing, and his signature on a key would be as good as your own.

Web of Trust key validity

A key *K* is valid if:

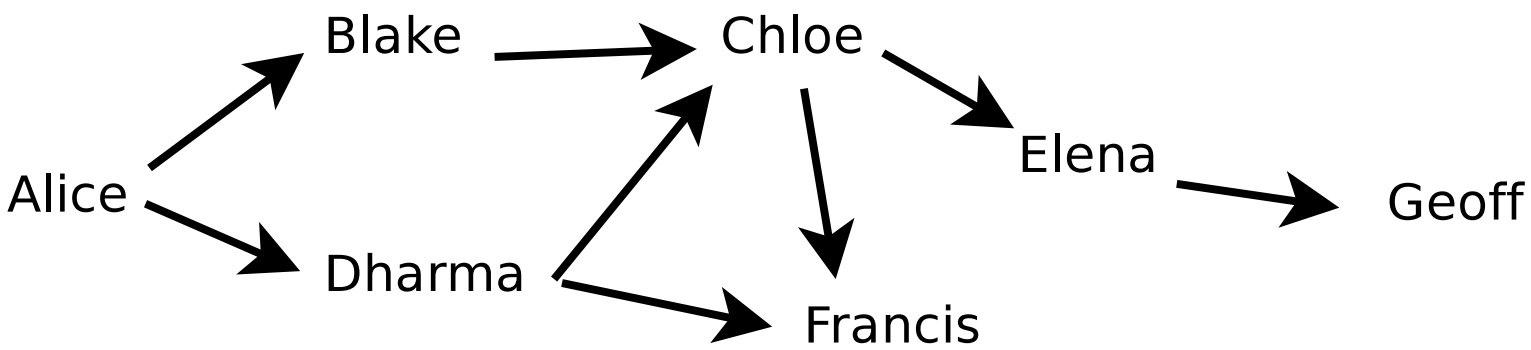
1. It is signed by enough **valid** keys, meaning
   - you have signed it personally,
   - it has been signed by 1 fully trusted key, or
   - it has been signed by 3 marginally trusted keys; and
2. the path of signed keys leading from K back to your own key is 5 steps or shorter.

Path length, number of marginally trusted keys required, and number of fully trusted keys required may be adjusted. The numbers given above are the default values used by GnuPG.

Notes on key validity computation

- Trust level is assigned by the user to keys.
- Validity level is computed by GnuGP based on trust values of **valid** keys.
- Full validity $\rightarrow$ valid key.
- Marginal validity $\rightarrow$ there is a trust path but not strong enough to be considered valid (does not fulfill all validity conditions)
- Only fully valid keys trust is taken in account for calculating validity!
- Keys signed by me are always fully valid (regardless of their trust level).

Example I



- Key valid if signed by:
  - (2 marginal keys) or (1 full key)
  - maximum path length is 3.

Example II

Alice key ring:

- Validity of the keys depends on who and at what level Alice **trusts**

	trust	
case	marginal	full
1		Dharma
2	Blake, Dharma	
3	Chloe, Dharma	
4	Blake, Chloe, Dharma	
5		Blake, Chloe, Elena

Example III

- Blake and Dharma are always valid keys (since Alice trusts herself!)

validity		
case	marginal	full
1		Blake, Chloe, Dharma, Francis
2	Francis	Blake, Chloe, Dharma
3	Chloe, Francis	Blake, Dharma
4	Elena	Blake, Chloe, Dharma, Francis
5		Blake, Chloe, Dharma, Elena, Francis