Public Key Infrastructure

- **Certificates**
- **Standard X509v3**

Certificate and Certification Authority

Problem. Make a subject's public key available to others so that they can verifiy the key authenticity and validity

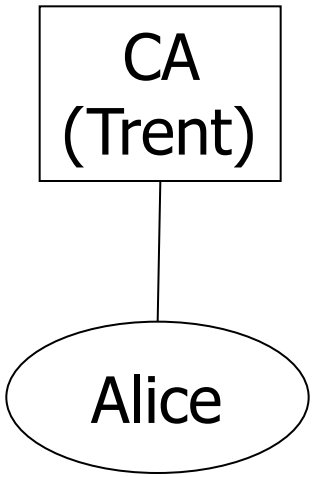
Assumption. Every subject can be uniquely identified (distinguished name)

Certification Authority is a TTP that attests the authenticity and validity of a public key

A **certificate** is a data structure that indissoluby links a subject identifier to the subject's public key

A certificate is digitally signed by the Certification Authority

Creazione di un certificato



1. CA verifies Alice's identity
2. CA verifies that the pubkey under certification is just Alices'
3. CA generates a certificate

$$C(T, A) = e_A, A, L, S_T(e_A, A, L)$$

con **L** validity period

data part — signature part

* C(T, A) is also denoted by T<<A>> or T{A}

A certificate may also specify additional information about:

- the subject, the pubkey, the signing algorithm;
- the policy for subject identification or key generation;
- other

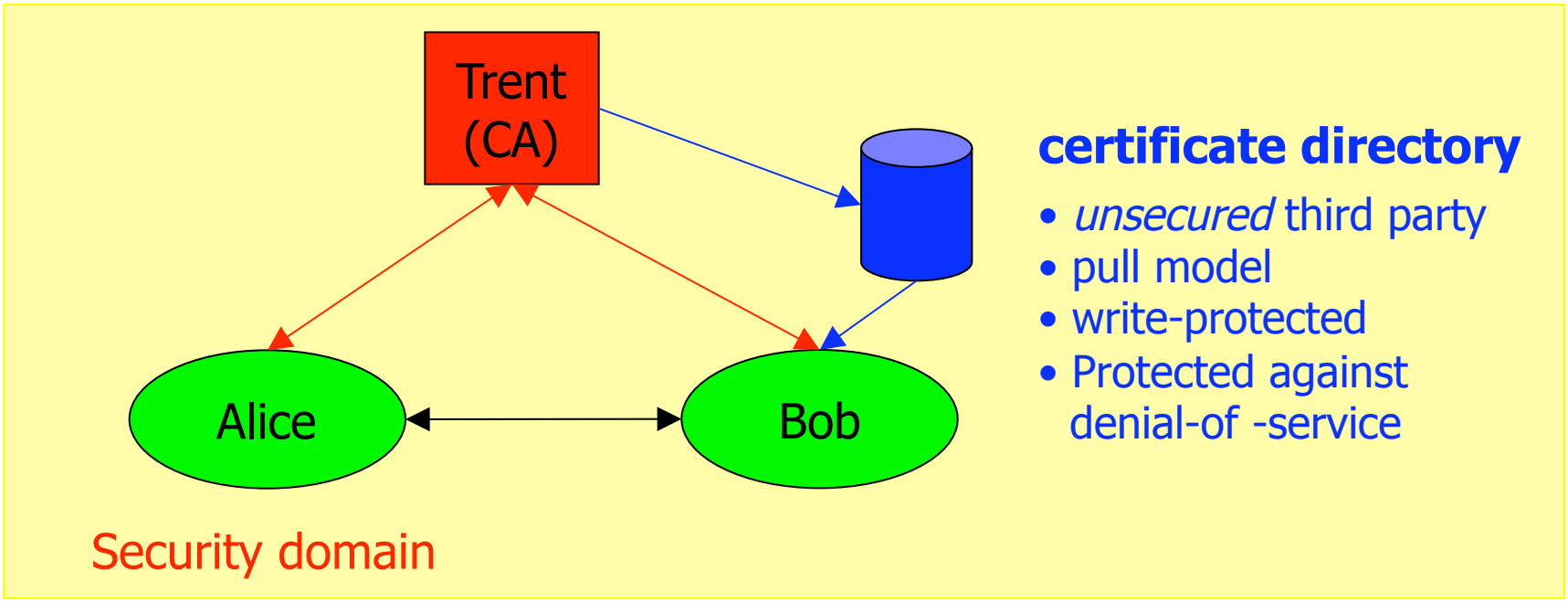
Certificate generation

In order to release a certificate, a CA has to

- ***Verify the subject's identity***
  - Typically, by off-line, non-cryptographic means
- ***Verify the pubkey authenticity***
  - ***Scenario 1***. CA itself generates the (pubkey, privkey) pair and transfers them to the subject in such a way to guarantee their authenticity
  - ***Scenario 2***. The (pubkey, privkey) pair is generated by the subject; the pubkey is transferred to the CA in such a way as to guarantee authenticiy.

    In this case, the CA requires the subject to prove that he/she holds the cognate privkey (e.g., challenge-response)

Security domain



- All entities in a security **domain** trust the same CA
- A **certicate directory** is a read-only database which store certificates and managed by an untrusted third party

Use and verification of a certificate

1. Bob obtains an authentic Trent's pubkey (e_T) [*one time*]
2. Bob obtains Alice's identifier *A*
3. Bob obtains the certificate *C*(*T*, *A*)
4. Bob verifies the certificate
    1. Bob verifies Trent's key validity
    2. Bob verifies certificate *C*(*T*, *A*) validity
    3. Bob verifies the signature on *C*(*T*, *A*)
    4. Bob verifies that certificate *C*(*T*, *A*) hasn't been revoked
5. If all verifications are successful, then Bob trusts e_A as Alice's pubkey

Use and verification of a certificate

Certification is based on trust delegation principle

- Bob trusts (and thus delegates) CA to
  - verify Alice's' identiy
  - attest the authenticity of Alice's pubkey
- Bob trusts the authenticity of CA's pubkey
- Through the certificate verification process, Bob ***transitively acquires trust*** in the pubkey contained in any certificate signed by CA

Attribute certificate

A certificate attests a link between a pubkey and an id but says nothing about the nature of this link, i.e., it doesn't say the scope of the key

- *An attribute certificate allows us to link a key to* ***attributes***
  - Authorization information;
  - Constraint to the use of the dig sig
    - Transactions of a certain maximum amount, at a certain time, and so forth

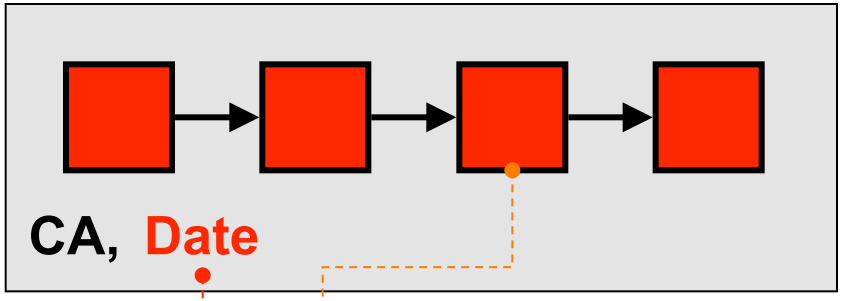
Revocation

- A certificate expires when the pubkey validity period expires
- If, for any reason, the privkey gets invalid before its expiration then the related certificate has to be revoked
  - The privkey gets compromised, or supposed so
  - The subject has changed his role; has quit; has been fired…
- **Certificate revocation must be**
  - **Correct**. Only authorized parties can issue a certificate revocation, namely the owner (subject) or the issuer (certification authority)
  - **Timely.** Revocation has to be disseminated to interested parties as soon as possible.

Revocation management

- **Expiration date.**
  - It ***limits*** exposition subsequent a compromization
- **Off-line notification.**
  - It isn't scalable
- **Public db of revoked keys**.
  - **Certificate revocation list (CRL)**
  - It should be checked ***before*** any key usage
- **Revocation certificates**.
  - Certificate where the **revocation flag** is active ;
  - In the certificate db, the recocation certificate substitutes the original certificate
  - Alternative to CRL
  - Limited timeliness revocation

Certificate Revocation List



DigSigned by CA

- serial number, revocation date, revocation reason, …
- Date gives indications on the CRL freshness

- ***A revoked certificate resides in CRL until expiration***

Certificate Revocation List

PROS

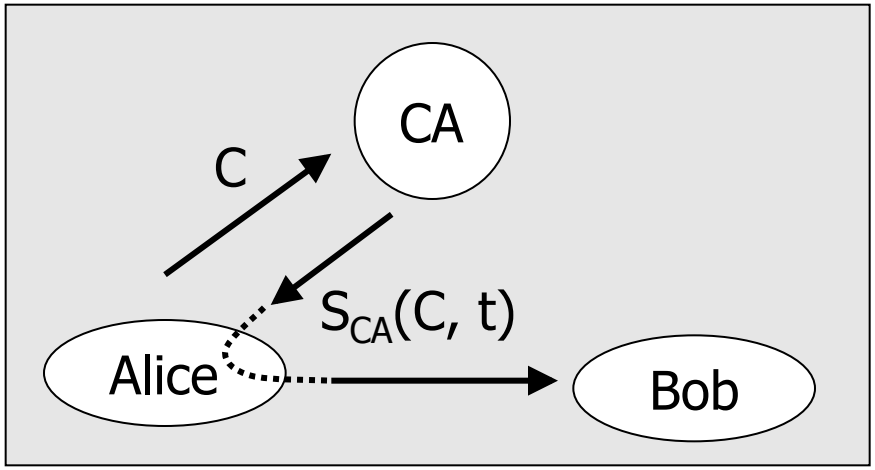
- CRL allows us to verify the validity of a certificate in the same way as credit cards

CONS

- **Timeliness.** An adversary abuses of a priv key until the corresponding certificate in not published in CRL
- Sometimes a CRL is the last component to be implemented or, even, it is not implemented at all
  - Old browsers did not access to CRL
  - The Microsoft vs Verisign case

Alternative approaches

- ON-LINE CONTROL OF CERTIFICATES
- TIMELY-CERTIFICATION (short-term certificate).
  - Bob requests Alice a recent certificate
  - Bob specifies how recent the certificate has to be



- C: valid certificate released by CA to A
- $S_{CA}(C, t)$: Up-to-date copy of C
- t time stamp in the validity period
- $S_{CA}(C, t)$ proves that at time ***t*** certificate ***C*** was valid

Non-repudiation

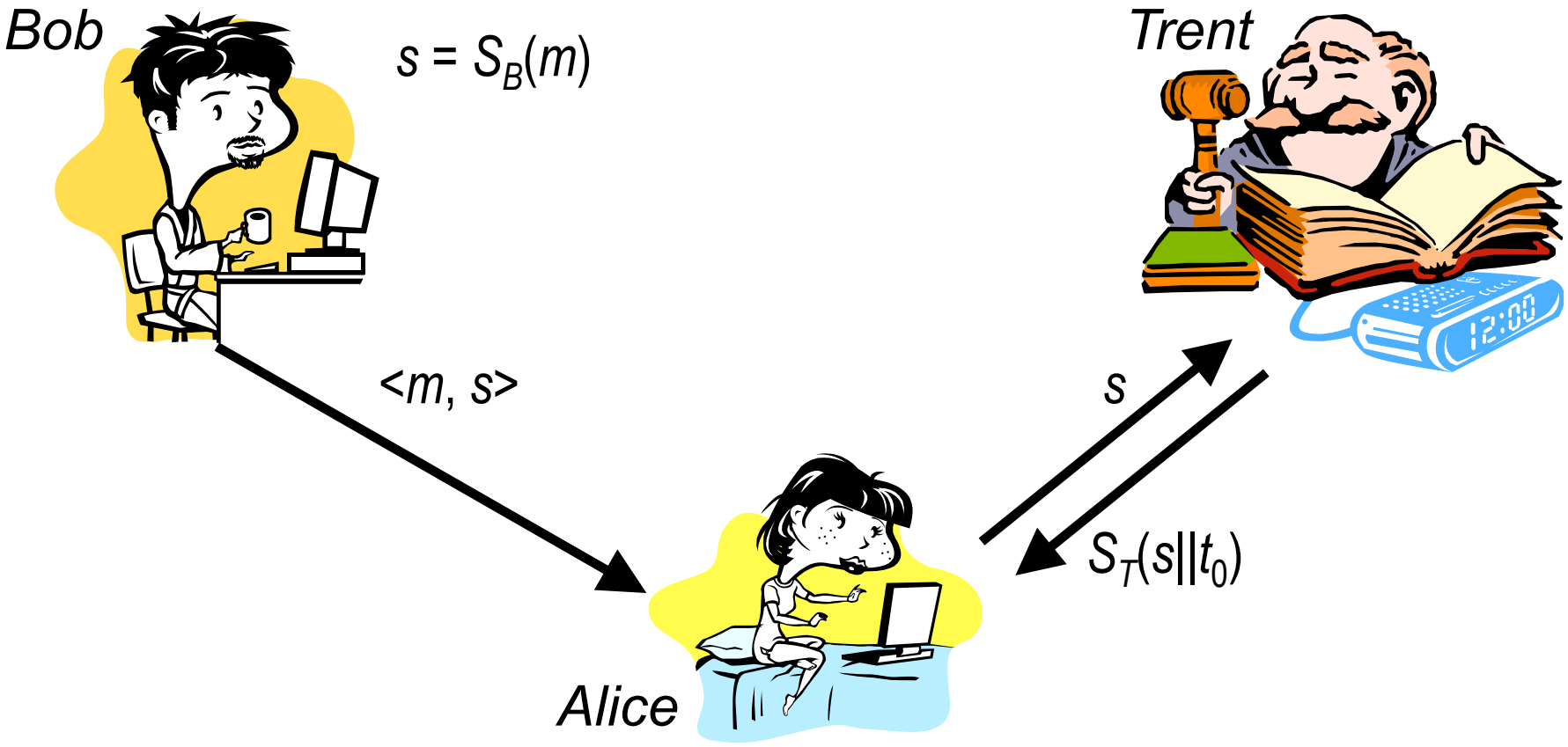
- Non-repudiation prevents a signer from signing a document and subsequently being able to successfully deny having done so.
- ***Non-repudiation vs authentication of origin***
  - Authentication (based on symmetric cryptography) allows a party to convince **itself** or a **mutually trusted party** of the integrity/authenticity of a given message at a given time t_0
  - Non-repudiation (based on public-key cryptography) allows a party to convince **others** at any time $t_1 >= t_0$ of the integrity/authenticity of a given message at time t_0

> *Alice's digital signature for a given message depends on the message and a secret* ***known to Alice only (the private key)***

Non-repudiation

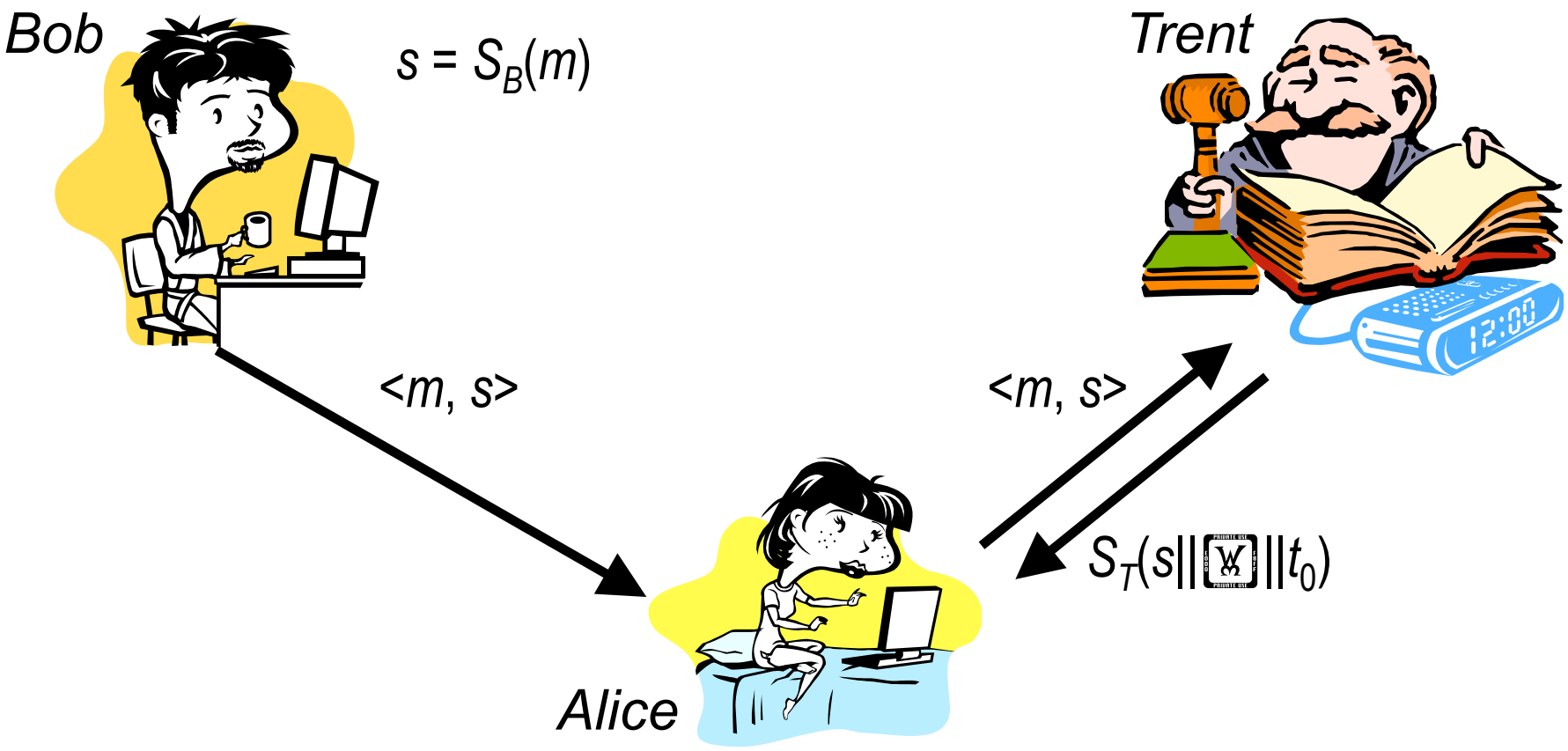
- Data origin authentication as provided by a digital signature is valid only while the *secrecy* of the signer's *private key* is maintained
- A threat that must be addressed is a signer who *intentionally* discloses his private key, and thereafter claims that a *previously* valid signature was forged
- This threat may be addressed by
  - *preventing direct access to the key*
  - *use of a trusted timestamp agent*
  - *use of a trusted notary agent*

Trusted timestamping service



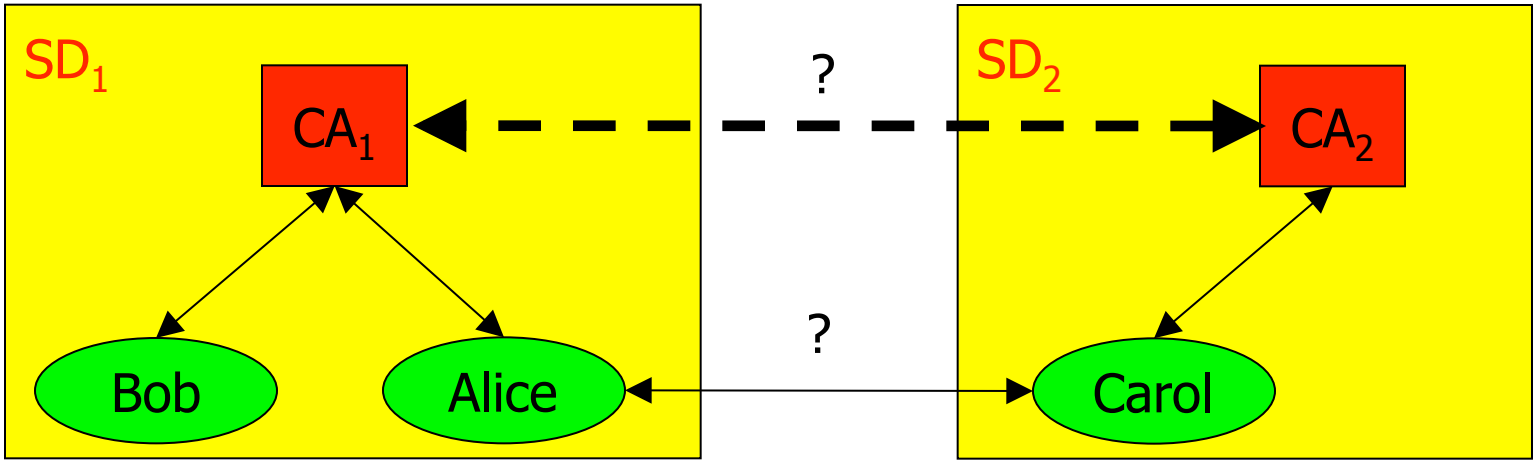
- Trent certificates that dig sig ***s exists*** at time t_0
  - Trent certificates that he has "seen" ***s*** at time t_0
- If the priv key d_B is compromised at time $t_1 > t_0$, then ***s*** is ***valid***

Trusted Notary Service



- Trent certicates that a certain *statement* σ on the dig sig *s* holds at a given instant t_0
  - *E.g,* . σ = “the dig sig is valid”

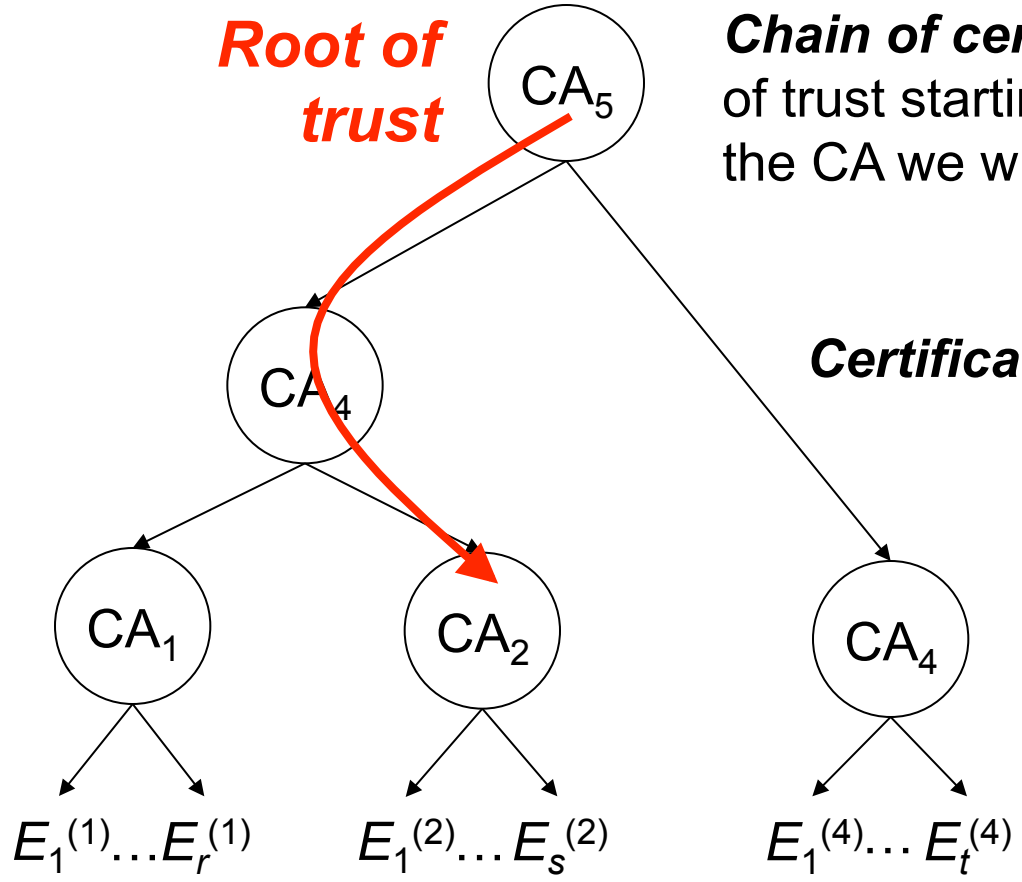
Multiple CAs and trust models



- *All entities in a security domain trust the same CA*
- *Users belonging to different trust domains can communicate if a trust relationship exists between the respective CAs*
- *Trust relationships between CAs allow us to determine how certificates released by a CA can be used and verified by another CA*

Centralized trust model

Alice needs to verify the certificate $CA_2\langle\langle E_i^{(2)}\rangle\rangle$



Chain of certificates defines a continuous chain of trust starting from a trusted CA and ending in the CA we wish to trust

Certification path represents the trust model

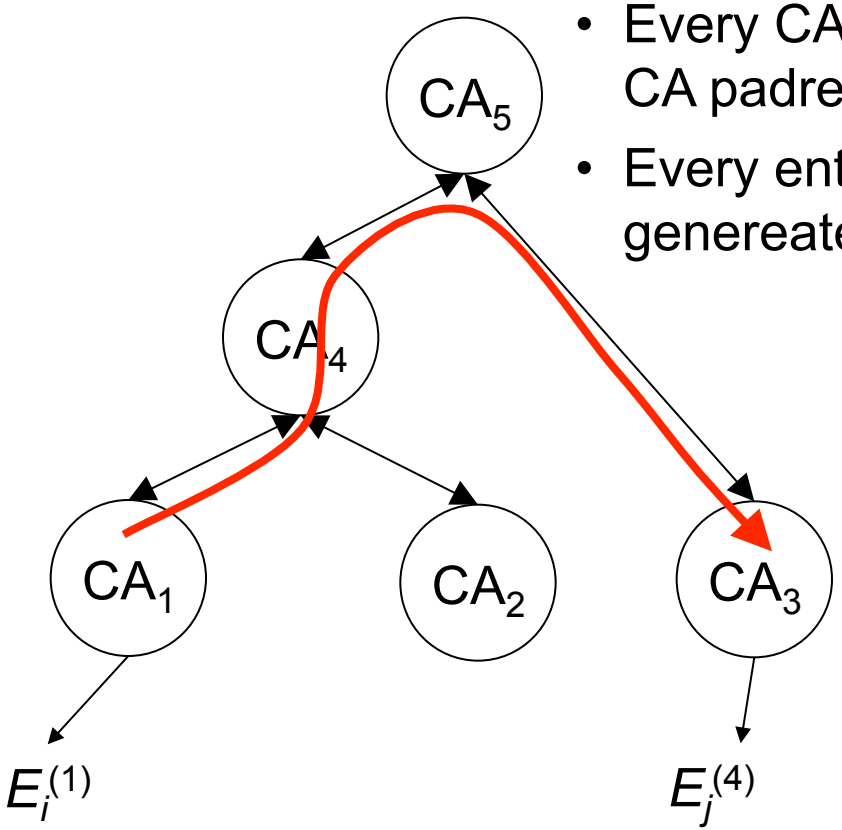
Every entity knows the pub key of the ***root of trust***

Chain of certificates for CA_2: $CA_5\{CA_4\}CA_4\{CA_2\}$

Centralized trust model

- ***The centralized trust model defines a single security domain***
- All the trust resides in the root
  - A chain of certificates is necessary even for two entities placed under the same CA
  - Chain of certificates tend to be long
- This model is not natural
  - *A more natural model*: an entity trusts a local CA (parent CA) rather than a remote one (root CA)

Inverse certificates

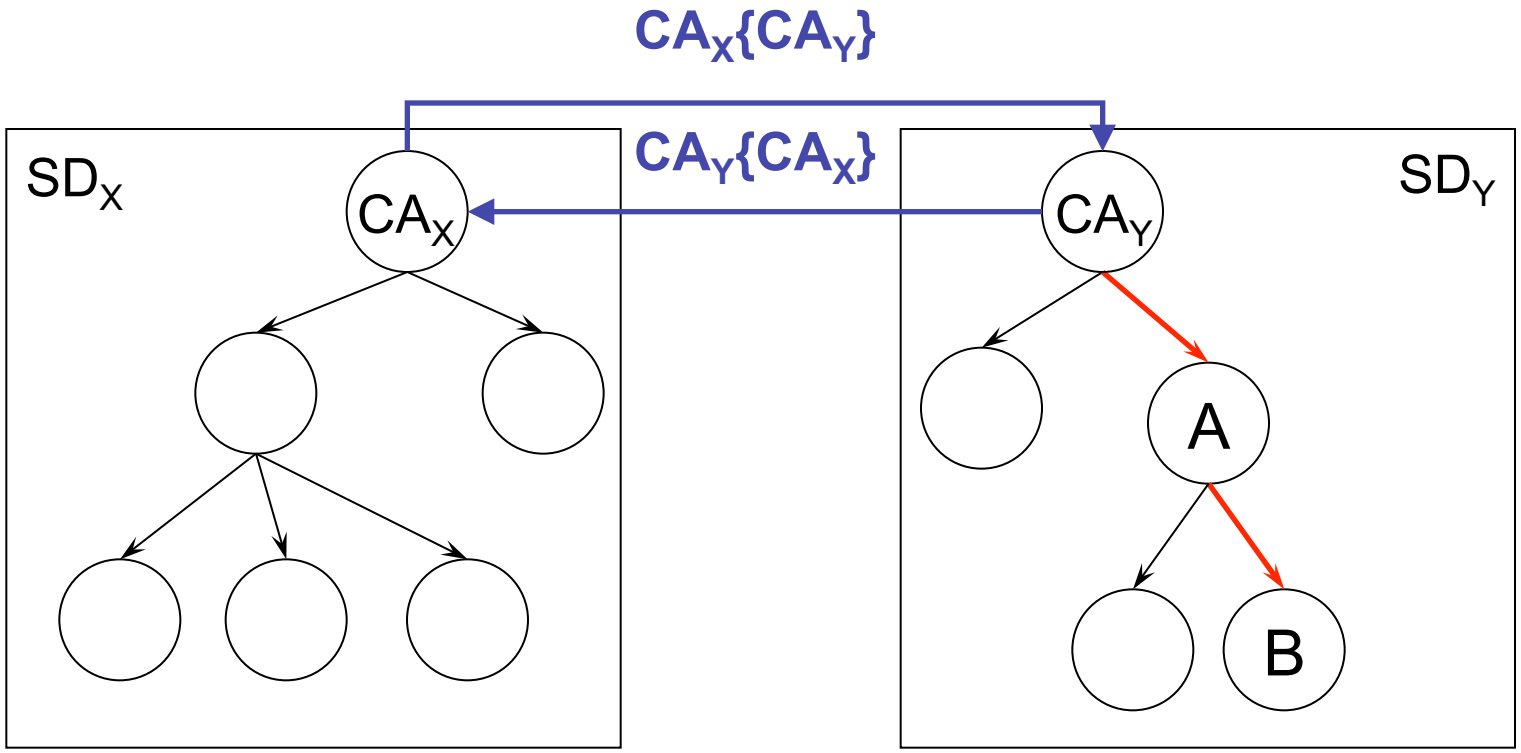


- Every CA creates a *reverse certificate* for the parent CA padre and a *forward certificate* for the child CA
- Every entity knows the pub key of the CA that genereates its certificate

- PROS
  - The trust model is more natural
- CONS
  - Long chains
  - To avoid long-chains **cross-certification** is necessary

Chain of certificates for CA_3: $CA_1\{CA_4\}CA_4\{CA_5\}CA_5\{CA_3\}$

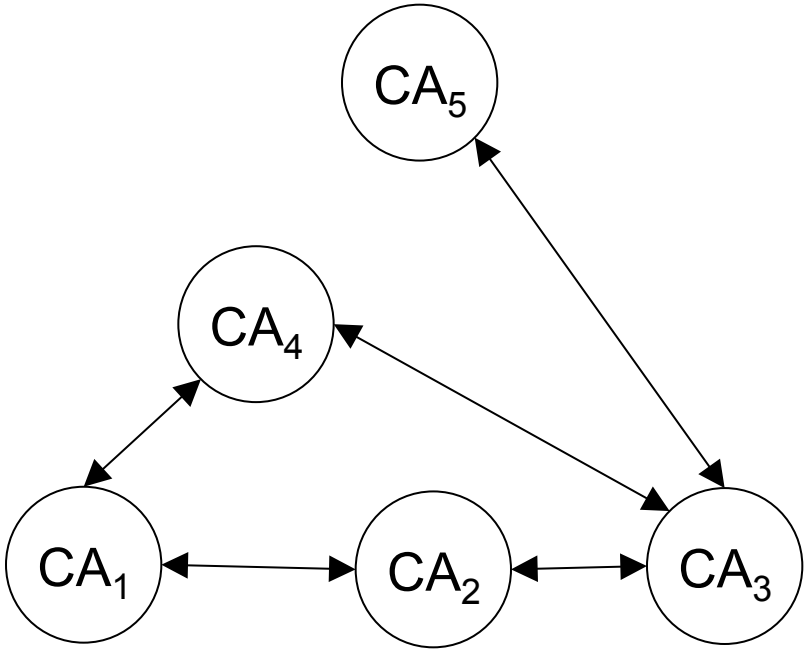
Multiple roots model



Cross-certificates allow an entity in SD_X (SD_Y) to get trust in released in SD_Y (SD_X) by CA_Y (CA_X)

Chain of certificates for B: $CA_X\{CA_Y\}CA_Y\{A\}A\{B\}$

Distributed model



- Every CA can certificate any other CA
- Every CA may certificate every user
- Every user knows the pub key of the local CA

A chain for CA_3: $CA_1\{CA_2\}CA_2\{CA_3\}$

Vincoli nei modelli di trust

- If CA_X cross-certifies CA_Y, CA_X trust in CA_Y transitively propagates to all CAs reachable from CA_Y
- CA_X may **limit** this propagation by means of **constraints** on cross-certificates
  - **Constraint on lenght**
    - *A certification chain has a limited maximum length*
  - **Constraint on domains**
    - *CA in the chain must belong to a predefined set of CAs*

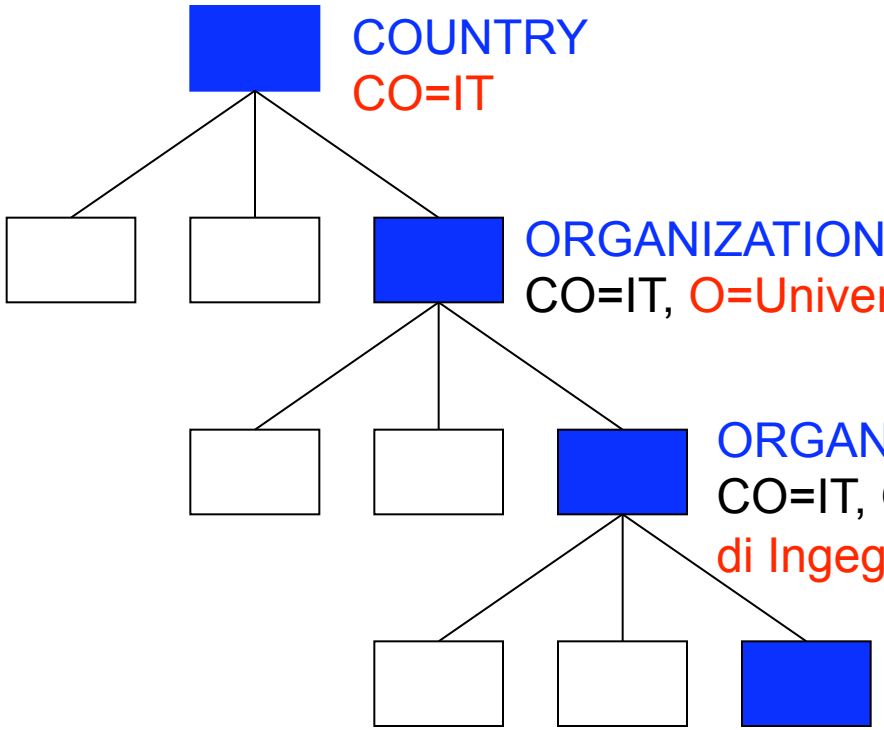
Certificato X.509 (RFC 3280)

Structure

1. Version
2. Serial number
3. Signature algorithm identifier
4. Issuer distinguished name
5. Validity interval
6. Subject distinguished name
7. Subject public key information
8. Issuer unique identifier (v=2,3)
9. Subject unique identifier (v=2,3)
10. Extensions (v=3)
11. Signature

- *Serial number* del certificato deve essere unico rispetto all'issuer
- *Distinguished name*, identificatore unico
- *Signature algorithm identifier* specifica l'algoritmo e la chiave pubblica dell'issuer
- *Subject public key information* specifica l'algoritmo, i parametri e la chiave pubblica del subject
- *Signature* di hash dei campi 1-10

Distinguished names (X.500)



L'organizzazione gerarchica "garantisce" l'unicità dei nomi ed "induce" l'organizzazione delle CA

Esempio: https://www.mps.it

Certificate name
www.mps.it
Consorzio Operativo Gruppo MPS
Terms of use at www.verisign.com/rpa (c)00
Florence
Italy, IT

Issuer
VeriSign Trust Network
www.verisign.com/CPS Incorp.by Ref. LIABILITY LTD.(c)97 VeriSign

Details
Certificate version: 3
Serial number: 0x652D0F8ADAB4C7B168A27BBD1C3E9D9D
Not valid before: Mar 2 00:00:00 2004 GMT
Not valid after: Mar 2 23:59:59 2005 GMT
Fingerprint: (MD5) CA CA 88 08 EC D0 8E 49 A6 9A 66 C4 69 31 E0 AE
Fingerprint: (SHA-1) 82 64 CB 69 F0 43 86 43 FF B4 55 D4 25 EF 51 60 65 46 D3 87

continua

Esempio: https://www.mps.it

Public key algorithm: rsaEncryption
Public-Key (1024 bit):
Modulus:
00: E1 80 74 5E E7 E5 54 8B DF 6D 00 95 B5 96 27 AC
10: 66 93 E0 49 B9 6F 5B 73 53 1C BE 1C EB 47 64 B2
20: 12 95 70 E6 CD 50 67 02 88 E3 EE 9D B1 91 49 C8
30: 8D 58 19 4B 86 8F C0 2E 65 E8 F2 D4 82 CC 55 DB
40: 43 BC 66 DA 44 2F 53 B3 48 4B 37 15 F3 AB 67 C1
50: 69 B4 53 23 19 30 1A 19 23 7F 28 E0 E3 C0 6B 18
60: FF 84 C4 AC A9 74 28 DB FF E9 48 CA 75 D5 35 D6
70: 46 FB 7D D4 A7 3F A1 4B 00 60 14 DC D5 00 CF C7
Exponent:
01 00 01
Public key algorithm: sha1WithRSAEncryption
00: 23 A6 FE 90 E3 D9 BB 30 69 CF 43 2C FD 4B CF 67
10: D7 3C 46 22 9A 08 DB 05 1D 45 DC 07 F3 1E 4D 1F
20: 4B 11 23 5B 42 91 14 95 25 88 1F BD 60 E5 6F 84
30: 44 70 7A 95 EC 30 E4 46 4F 37 87 F1 B2 FA 45 04
40: 6F 7C BE 97 25 C7 20 E7 F3 90 55 51 99 3A 72 35
50: 40 F2 E8 E3 36 3A 7D 58 61 9C 91 D6 AC 34 E7 E8
60: 09 27 64 4F 2C 4C C2 D2 A3 32 DB 2B 7E F0 B6 F3
70: 69 96 E4 2B C3 2B 42 ED CA 2C 3C C8 F5 AA E6 71

continua

Esempio: https://www.mps.it

Extensions:

X509v3 Basic Constraints: CA:FALSE
X509v3 Key Usage: Digital Signature, Key Encipherment
X509v3 CRL Distribution Points:
URI:http://crl.verisign.com/Class3InternationalServer.crl
X509v3 Certificate Policies:
Policy: 2.16.840.1.113733.1.7.23.3
CPS: https://www.verisign.com/rpa
X509v3 Extended Key Usage: Netscape Server Gated Crypto, Microsoft Server Gated Crypto, TLS Web Server Authentication, TLS Web Client Authentication
Authority Information Access:
OCSP - URI:http://ocsp.verisign.com
Unknown extension object ID 1 3 6 1 5 5 7 1 12: 0_.].[0Y0W0U..image/gif0! 0.0...+.....k...j.H.,{..0%.#http://logo.verisign.com/vslogo.gif

Esempio: https://www.mps.it

Certificate name
VeriSign Trust Network
www.verisign.com/CPS Incorp.by Ref. LIABILITY LTD.(c)97 VeriSign

Issuer
VeriSign, Inc.
Class 3 Public Primary Certification Authority
US

Details
Certificate version: 3
Serial number: 0x254B8A853842CCE358F8C5DDAE226EA4
Not valid before: Apr 17 00:00:00 1997 GMT
Not valid after: Oct 24 23:59:59 2011 GMT
Fingerprint: (MD5) BC 0A 51 FA C0 F4 7F DC 62 1C D8 E1 15 43 4E CC
Fingerprint: (SHA-1) C2 F0 08 7D 01 E6 86 05 3A 4D 63 3E 7E 70 D4 EF 65 C2 CC 4F

Esempio: https://www.mps.it

Public key algorithm: rsaEncryption
Public-Key (1024 bit):
Modulus:
00: 6F 7B B2 04 AB E7 34 4F 9C 53 A7 02 B2 90 4F 22
10: F9 3A 3C 5A 8B 51 2B FE CB 42 95 30 70 FE 8A B2
20: D3 1D C1 B8 5A 49 5C F7 39 4E 4D B7 F3 3B 09 F1
30: FA E5 28 93 3E 30 F5 63 AA 43 71 27 56 FE A3 BB
40: CA C4 6C 75 B2 32 C1 07 D9 DD 25 40 F5 5C A9 D4
50: 15 0A 34 9A ED 42 97 EA BD F1 B2 55 45 73 3C AA
60: E7 B6 5B 6C 4C F0 AA 3B 36 E6 BC D3 05 D4 BF E1
70: 2B 65 A2 25 39 18 85 1F 7D 02 19 D6 E8 80 82 D8
Exponent:
01 00 01
Public key algorithm: sha1WithRSAEncryption
00: 08 01 EC E4 68 94 03 42 F1 73 F1 23 A2 3A DE E9
10: F1 DA C6 54 C4 23 3E 86 EA CF 6A 3A 33 AB EA 9C
20: 04 14 07 36 06 0B F9 88 6F D5 13 EE 29 2B C3 E4
30: 72 8D 44 ED D1 AC 20 09 2D E1 F6 E1 19 05 38 B0
40: 3D 0F 9F 7F F8 9E 02 DC 86 02 86 61 4E 26 5F 5E
50: 9F 92 1E 0C 24 A4 F5 D0 70 13 CF 26 C3 43 3D 49
60: 1D 9E 82 2E 52 5F BC 3E C6 66 29 01 8E 4E 92 2C
70: BC 46 75 03 82 AC 73 E9 D9 7E 0B 67 EF 54 52 1A

Esempio: https://www.mps.it

Extensions:

X509v3 Basic Constraints: CA:TRUE, pathlen:0
X509v3 Certificate Policies:
Policy: 2.16.840.1.113733.1.7.1.1
CPS: https://www.verisign.com/CPS
X509v3 Extended Key Usage: TLS Web Server Authentication, TLS Web Client Authentication, Netscape Server Gated Crypto, 2.16.840.1.113733.1.8.1
X509v3 Key Usage: Certificate Sign, CRL Sign
Netscape Cert Type: SSL CA, S/MIME CA
X509v3 CRL Distribution Points:
URI:http://crl.verisign.com/pca3.crl

Assurance: il caso di Verisign

- **Verisign distributes three classes of certificates; each class defines the appropriate use and the autentication procedure**

- **Class 1 Certificates**. Class 1 Certificates offer the lowest level of assurances within the VTN. The Certificates are issued to individual Subscribers only, and authentication procedures are based on assurances that the Subscriber's distinguished name is unique and unambiguous within the domain of a particular CA and that a certain e-mail address is associated with a public key. Class 1 Certificates are appropriate for digital signatures, encryption, and access control for non-commercial or low-value transactions where proof of identity is unnecessary.
- **Class 2 Certificates**. Class 2 Certificates offer a medium level of assurances in comparison with the other two Classes. Again, they are issued to individual Subscribers only. In addition to the Class 1 authentication procedures, Class 2 authentication includes procedures based on a comparison of information submitted by the certificate applicant against information in business records or databases or the database of a VeriSign-approved identity proofing service. They can be used for digital signatures, encryption, and access control, including as proof of identity in medium-value transactions.
- **Class 3 Certificates**. Class 3 Certificates provide the highest level of assurances within the VTN. Class 3 Certificates are issued to individuals and organizations for use with both client and server software. Class 3 individual Certificates may be used for digital signatures, encryption, and access control, including as proof of identity, in high-value transactions. Class 3 individual Certificates provide assurances of the identity of the Subscriber based on the personal (physical) presence of the Subscriber before a person that confirms the identity of the Subscriber using, at a minimum, a well-recognized form of government-issued identification and one other identification credential. Class 3 organizational Certificates are issued to devices to provide authentication; message, software, and content integrity and signing; and confidentiality encryption. Class 3 organizational Certificates provide assurances of the identity of the Subscriber based on a confirmation that the Subscriber organization does in fact exist, that the organization has authorized the Certificate Application, and that the person submitting the Certificate Application on behalf of the Subscriber was authorized to do so. Class 3 organizational Certificates for servers also provide assurances that the Subscriber is entitled to use the domain name listed in the Certificate Application, if a domain name is listed in such Certificate Application.

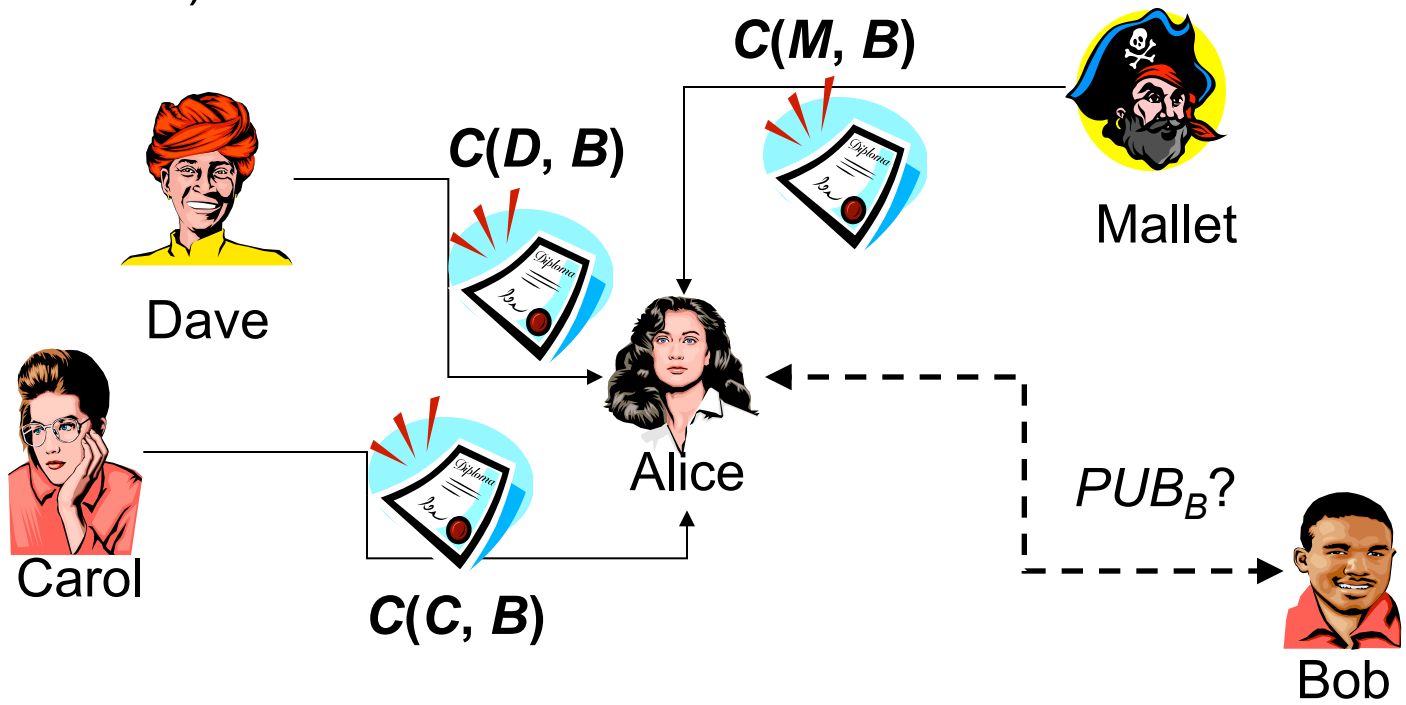
Nomi e certificati

- How much can I trust a certificate?
  - **Authentication policy**
  - **Issuance policy**
    - These policies must be public
    - If a CA certifies other CAs, their policies must be more restrictive
- The trust level cannot be quantified but it can be estimated according to the CA policies and the process to implement them

Pretty Good Privacy (PGP)

The user decides how much trust to place in a certificate

"*PGP is for people who prefer to pack their own parachutes*"
(P. Zimmerman)



Alice decides the trust in Bob according to the number of certificates and the trust level in each of them

In-house o outsourcing?

For an organization, is it more covenient to implement its own CA or resort to a commercial CA?

Cost vs quality

- High-quality certification process is expensive
- Low-quality certification process implies higher security risks

In-house solution

- PROS. Complete control on the certification process; the organization assesses risks and chooses the more appropriate solution.
- CONS
  - Costs of necessary infrastructure
  - Limited scalability (cross-certification)

Outsourcing solution

- PROS: scalability (certificates are accepted by all browser)
- CONS: trust delegation; typically Cas don't take any liability in the case on errors (*Certification Practices Statement*)