Public Key Infrastructures

Gianluca Dini
Dept. of Ingegneria dell’Informazione
University of Pisa
Email: gianluca.dini@.unipi.it
Version: 14/04/25

1

Public Key Infrastructures

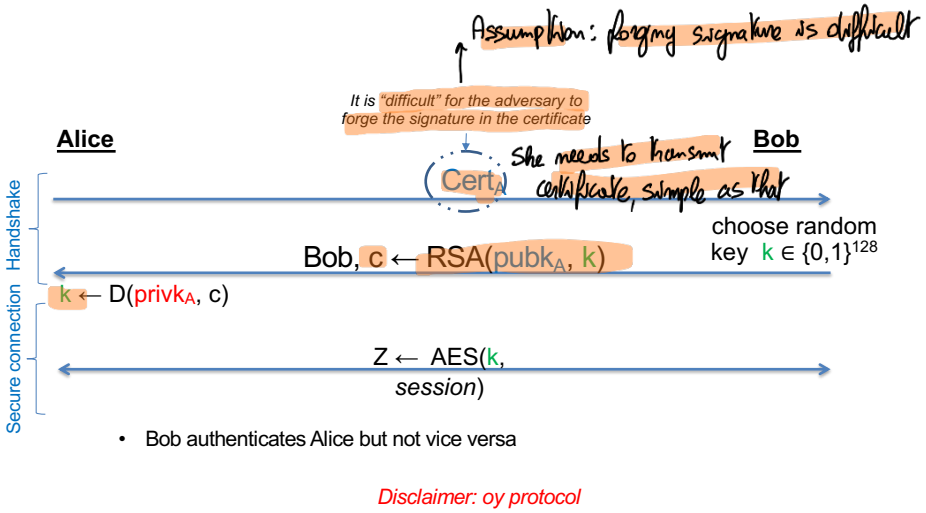
CERTIFICATES: HOW TO DEFEAT THE MAN-IN-THE-MIDDLE

2

Certificates and Certification Authority

- Certificate format (basic):
  - $Cert_A$ = Alice, $pubK_A$, L,…, S($priv_{CA}$, Alice || $pubK_A$ || L, …)
    with L = validity period
- A certificate indissolubly binds the identity of a subject (Alice) to his/her public key ($pubK_A$); the binding is the digital signature of a Trusted Third Party called **Certification Autority** (CA)
  - In order to verify a certificate you need the CA's public key $pubK_{CA}$

Key Transport with Certificates

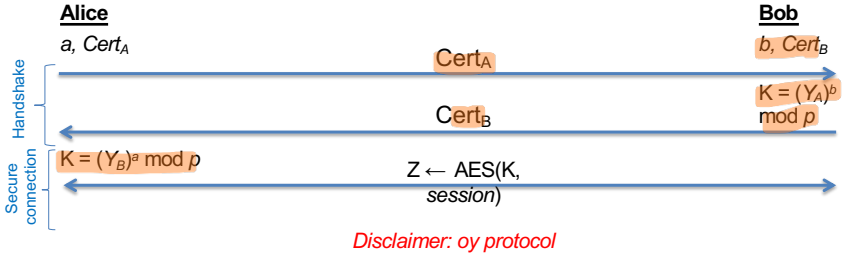


- Bob authenticates Alice but not vice versa

Disclaimer: oy protocol

Diffie-Hellman with certificates

- $Cert_A$ = *Alice*, Y_A, L_A, S_{CA}(*Alice* || Y_A || L_A) with
- $Y_A = g^a \bmod p$ Alice's public key, and a Alice's private key
- S_{CA} digital signature by certification authority CA



Disclaimer: oy protocol

Another execution mode of TLS. I can choose key establishment scheme

8

PKIs

CERTIFICATION AUTHORITIES

9

CA's obligations [→]

- CA must be reliable
  I. CA must verify that owner of ($privK_A$, $pubK_A$) pair is really entitled to use that name I could go to CA and claim I am Donald Trump
     - CA establishes rules/policies to verify that a person has rights to the name
     - Identifying a subject is not easy; depends on country
  – CA must verify that the name (e.g., Alice) goes along with the key ($privK_A$) → I could go to CA after finding Trump's public key and get a certification: certifying that I own the corresponding private key.
    Those are the only two cases in which CA is liable for damages.

10

CA's obligations [↓]

- CA's certificate must be (immediately) available
  – CA's certificate is released at user registration time
  – CA's certificate is published in newspapers
  – CA's certificate is embedded in a browser installation package (is this secure?)
  – …

You need pubkey of CA to verify a certificate. Typically the pubkey is in a self signed certificate of the authority.

11

Trust delegation

- Certification is based on trust delegation/transfer

1. Bob trusts and delegates CA to verify Alice’s identity and attest the authenticity of $pubK_A$,
2. Bob trusts the authenticity of CA’s $pubK_{CA}$

consequently

- Through certificate $Cert_A$ signed by CA, Bob acquires trust (believes) in the authenticity of $pubK_A$

Important to remember

- What a certificate does
  - A certificate defines an indissoluble link between a subject’s identifier and public key
- A certificate does not
  - specify the meaning of that link
  - the possible uses of that key
  - make any statement on the trustworthiness of the subject

Assurance [→]

- How much can I trust that the identifier actually corresponds to the legitimate owner of the key?
- CA Policies
  - Authentication policy Policy to identify subject
  - Issuance policy Policy by means of which CA verifies subject has corresponding private key.
  - These policies are public
    - A child-CA cannot have less restrictive policies
    - Usually in a document linked inside the certificate
- Assurance is not quantifiable
  - Estimate according to the policy and the application rigor
    - How much I can trust a cert is not something I can quantify.

Assurance [→]

- Specification, design and implementation contribute to the assurance
- Example: medecine
  - The process
    - A medicine is produced by a known and honorable pharmaceutical manufacturer
    - The medicine is delivered to chemists in a in sealed container
    - When the medicine is sold, the seal is still intact

Assurance [↓]

- Trust foundations
  - Ministry allows sale if the medicine passes certain tests and complies with certain clinical standards
  - Auditing committees verify that the production process satisfies industrial standards
  - Presence of the safety seal

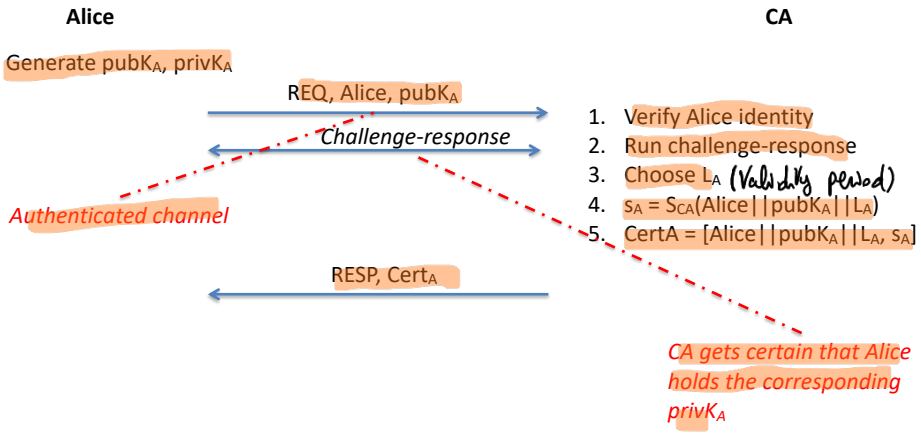
16

In-house or external CA?

- Implement your own CA or exploit a commercial one?
  - Cost-convenience ratio
    - High quality certification ➔ high costs
    - Low quality certification ➔ high risks
  - In-house
    - Pros – Complete control of the certification process
    - Cons – Cost of the infrastructure; limited scale
  - Commercial
    - Pros – Large scale; recognised by anyone
    - Cons – Trust delegation; no liability

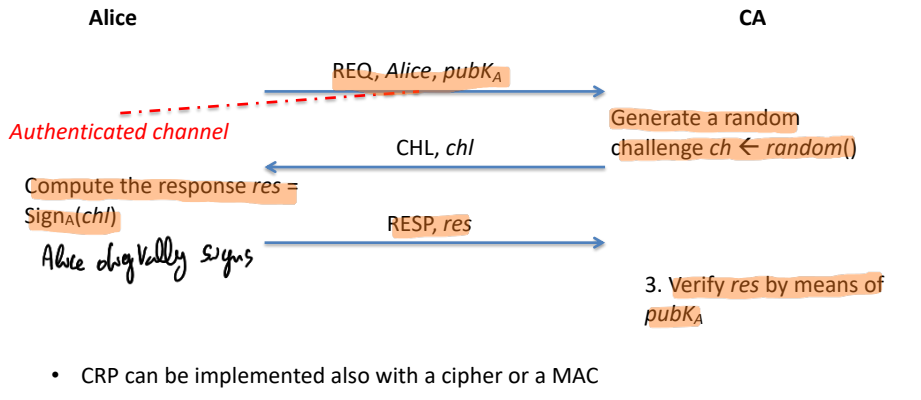
17

Certificate generation: user-provided keys [→]



18

Challenge-response protocol

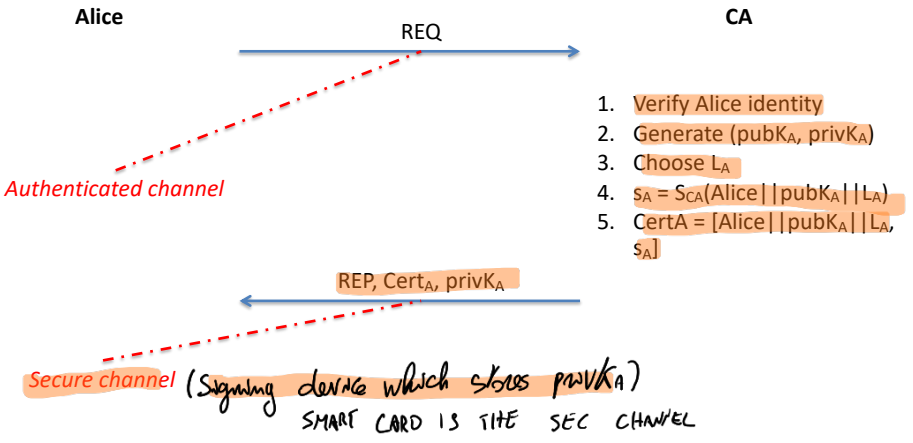


- CRP can be implemented also with a cipher or a MAC

19

Certificate generation: CA-generated keys

Most frequent case



20

On key generation at CA-side

- Fatal crypto flaw in some Taiwan government-certified smartcards makes forgery a snap (www.arstechnica.com)
- Fatal flaw in the hw RNG

PrivK was predictable

- Smartcards passed two international certifications (FIPS 140-2, Common Criteria)
- Research paper at AsiaCrypt 2013

21

Backup of private key [→]

Number of problems that might arise:

- Public key encryption ➔ backup of decryption key privK
  - Otherwise, encrypted data may become inaccessible
  - Be able to decrypt even after key lifetime expiration
  - Who makes backup matters
    - Government backs up of citizen’s privK ➔ privacy issues
    - Company backs up of employee’s privK ➔ Encrypted data belong to the company

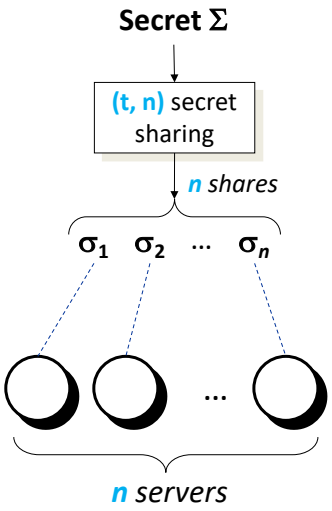
22

Backup of private key [→]

- Digital signature ➔ backup of signing key
  - Private key backup has adverse impact on non-repudiation
  - Delete the key after key expiration
  - Recovery from signing key loss is expensive in large scale apps as you must redistribute the pubK (e.g., Microsoft, Adobe, Visa)
  - Technological solution: Threshold crypto (t out of n)
- You want different key pairs for encryption and signing

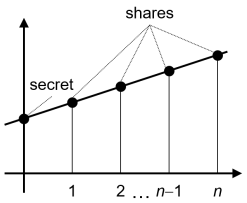
23

Threshold crypto (intuition)



SECRET SHARING

- The secret Σ (e.g., private key) is split into **n** shares
- At least **t shares** are necessary to reconstruct the secret
- The system tolerates the compromise of **t–1** nodes



Example: Polynomial **(2, n)** secret sharing

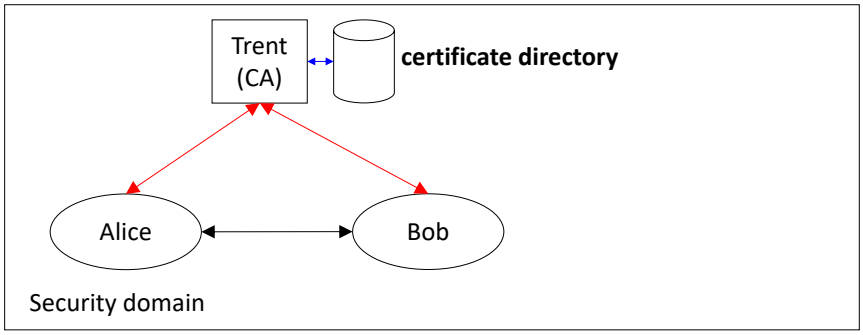
24

PKIs

EXPIRATION AND REVOCATION OF A CERTIFICATE

25

Single CA Model



- **Security domain** under control of the CA
- **Certificate directory** is a read-only database that stores certificates

26

Expired & revoked certificates

- A certificate is expired if the validity period is expired
- If the private key gets compromised before expiration, then the certificate must be revoked
  - Examples: the private key has been revealed; the subject has changed role or left the organization
- Certificate revocation must be
  - Correct: revocation can be granted only to authorized parties, i.e., the owner or the issuer
  - Timely: revocation must be disseminated to all interested parties as soon as possible

27

How to verify a certificate

- Bob’s verification of Alice’s $Cert_A$
  1. Bob obtains CA’s public key $pubK_{CA}$ [once at set-up]
  2. Bob verifies validity of CA’s public key [once at set-up]
  3. Bob verifies the digital signature in $Cert_A$ by using $pubK_{CA}$
  4. Bob verifies that $Cert_A$ is valid
  5. Bob verifies that $Cert_A$ is not revoked
- If all these checks are successful, then Bob accepts $pubK_A$ as authentic Alice’s key

28

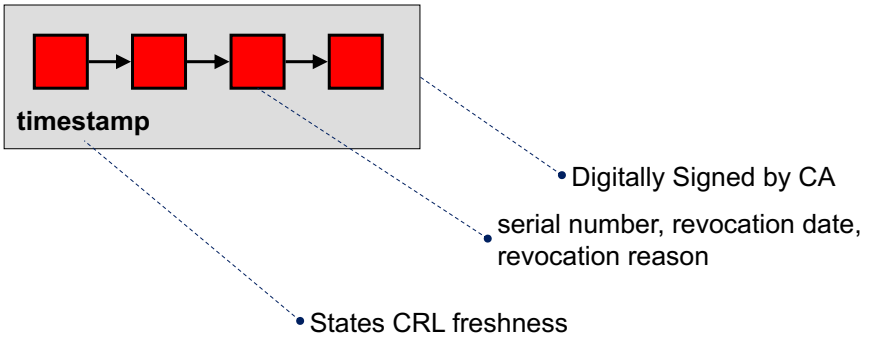
Revocation methods

- Offline → Certificate Revocation List (CRL)
- Online → Online Certificate Status Protocol (OCSP)

29

CRL

- A CRL is published periodically
- A revoked certificate lies in CRL until expiration
- Δ-CRL for efficiency



30

OCSP

- Protocol sketch
  - Alice $\rightarrow$ OCSP: <OCSP RQST, Bob’s cert serial nr.>
  - OCSP $\rightarrow$ Alice: <OCSP RESP, OK|KO>$_{OCSP}$
  - Protocol Pros
    - Lighter and simpler that CRL protocol
    - Effective if the adversary is not a MIM
  - Protocol Cons
    - In the clear ➔ confidentiality issues
    - Exposed to replay attack (nonces are an extension ☹)
    - Browsers silently ignore OCSP if the query times out (➔MIM)

31

PKIs

THE X.509 STANDARD

33

X.509 certificate format

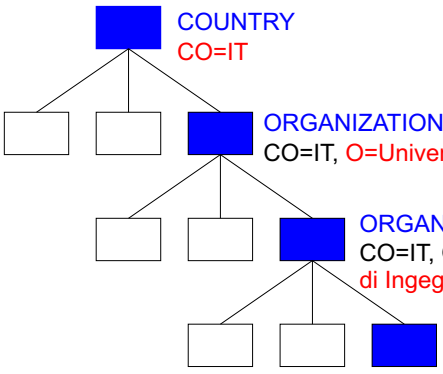
A data structure with several fields

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

X.509 uses the Abstract Syntax Notation, ASN.1, (RFC 1422)
X.509 has been conceived for X.400 mail standard
X.509 uses Distinguished Names

34

Distinguished names (X.500)



COUNTRY
CO=IT

ORGANIZATION
CO=IT, O=University of Pisa

ORGANIZATIONAL UNIT
CO=IT, O=University of Pisa, OU=Dipartimento di Ingegneria della Informazione

COMMON NAME
CO=IT, O=University of Pisa, OU=Dipartimento di Ingegneria della Informazione, CN=Gianluca Dini

35

Example: 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

contd

36

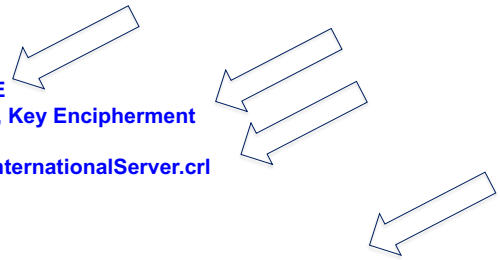
Example: https://www.mps.it

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

contd

37

Example: 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

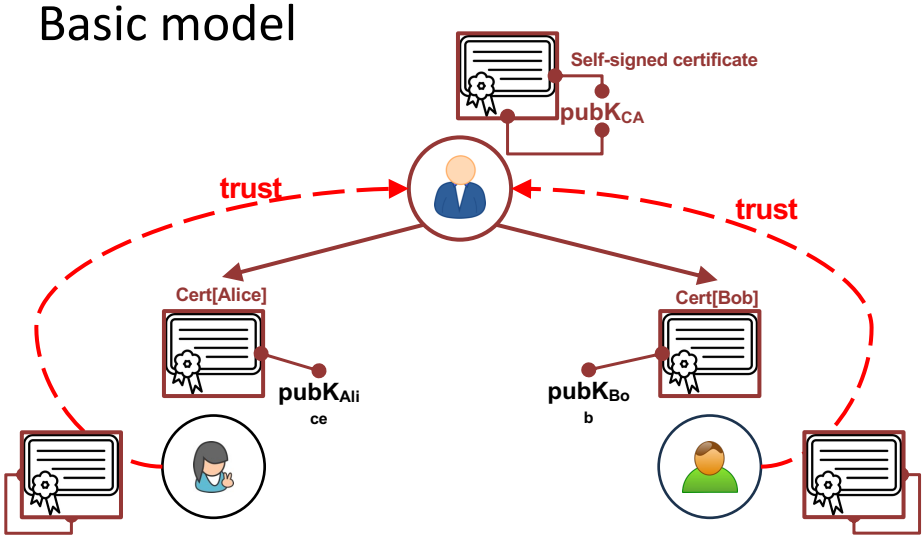
38

PKIs

TRUST MODELS

39

Basic model



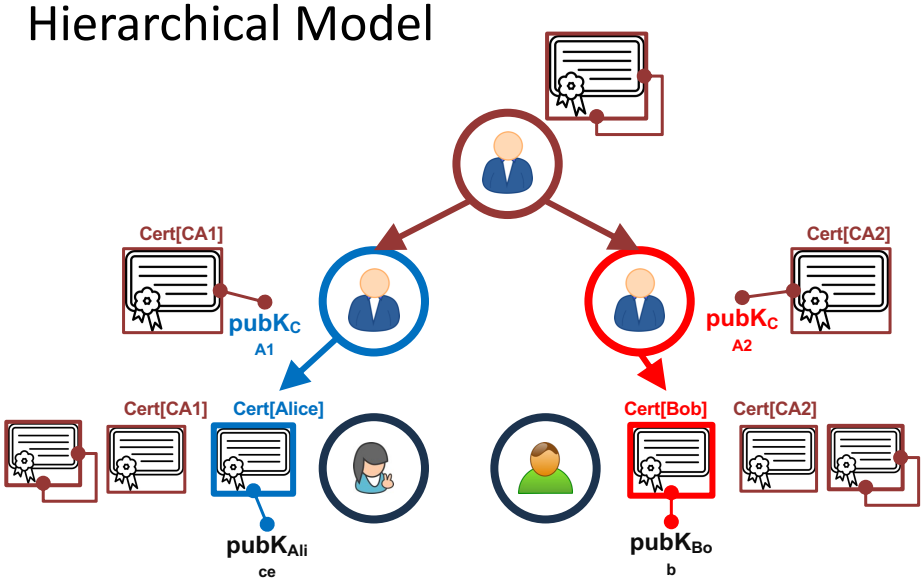
41

Basic Model

- The Model
  - Every user trusts the root
  - The root releases certificates
- Inconvenient
  - Users have to go to the root in order to get the root self-signed certificate

42

Hierarchical Model



43

Constrains on the certification path

- If CA_X certificates CA_Y (subordinate), the trust that CA_X has in CA_Y transitively propagates to all CAs reachable from CA_Y
- CA_X may limit this propagation by posing constraints

1. **on the chain length:** The chain after CA_Y has a limited length
2. **on the set of domains:** CAs in the chain after CA_Y must belong to a prefefined set of CAs

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

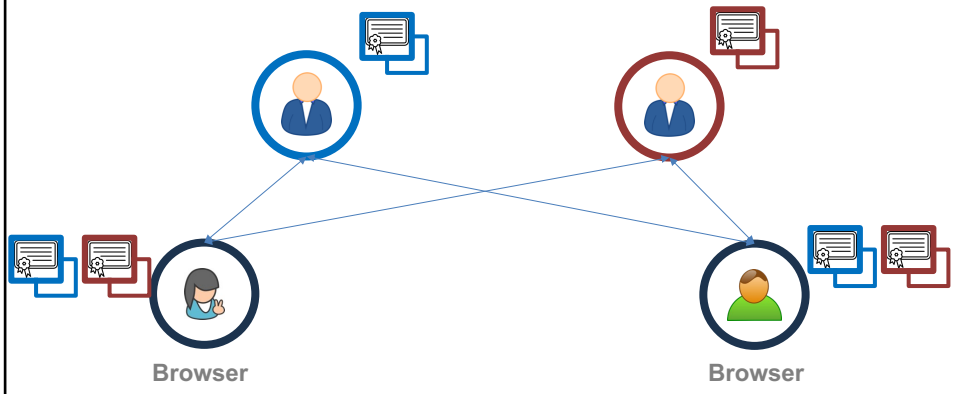
46

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

47

Web Model



apr-25 Public Key Cryptography 48

48

Examples

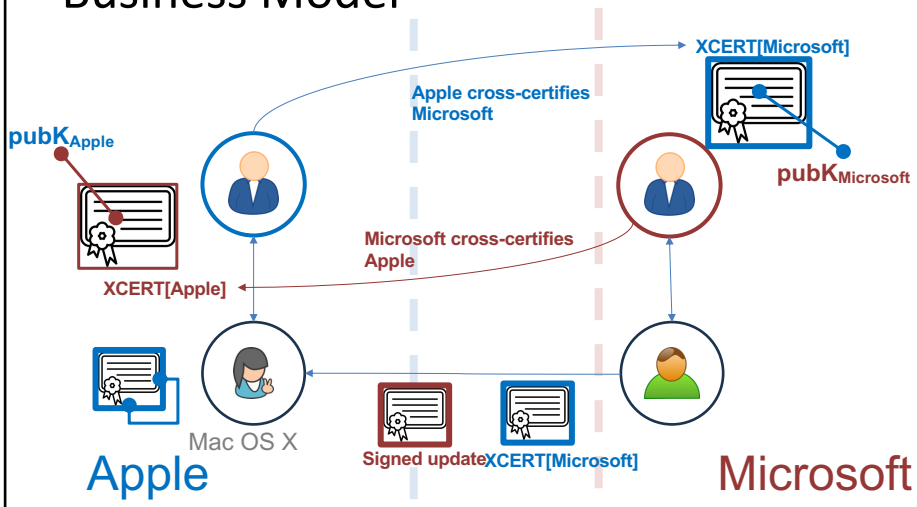
- Firefox
- Chrome
- Edge

- www.unipi.it ← GEANT Vereniging ← USERTrust (root)

apr-25 Public Key Cryptography 49

49

Business Model



50

Incidents

- March 2011 – Comodo
  - 9 fraudulent certs
- Summer 2011 – DigiNotar
  - 500+ fraudulent certs
  - FOX-IT final report (long)
  - ENISA’s resume (short)
- January 2013 – Turktrust
  - 100+ fraudulent certs
  - The TURKTRUST SSL certificate fiasco – what really happened, and what happens next?

62

Countermeasures (→)

- Public key pinning
  - List of presumed-good CAs and list of known-good certs
  - Chrome
- Certificate transparency
  - To make public that a CA issued a cert
  - Resistance from business
- Convergence
  - Download a cert directly and from a set of trusted CAs and compare them

63

Countermeasures

- DANE (DNS-based Authentication of Name Entities)
  - Store a pubK in a DNS record; require DNSSEC
- Extended Validation certificates
  - «Prove the legal entity controlling the website or sw package… promise what we were promised a decade ago and we never got» [The inevitable collapse of the certification model]

64

(K_1, K_2)

A B

$y = E_{K_1}(x)$

$t = MAC_{K_2}(x)$

$\xrightarrow{y \| t}$

$x = D_{K_1}(y)$

$\cdot V_{K_2}(t, x)$

$y = E_{K_1}(x)$

$t = MAC_{K_2}(y)$

$\xrightarrow{y \| t}$

$(V_{K_2}(y, t))$

↳ $x = D_{K_1}(y)$

$b / gcd(b, m)$: $X \rightarrow X' \equiv X \cdot b^e \bmod m$

$\downarrow$

$S' \equiv (X')^d \bmod m$

$S \equiv S' \cdot b^{-1} \bmod m \equiv (X')^d b^{-1} \bmod m \equiv (X \cdot b^e)^d b^{-1} \bmod m$

$comm$ ($comm)^d$

- p, g
- $b \rightarrow c = g^b \bmod p$

$Comm = [comm,\ \text{identity string},\ H(comm, nS)^d]$

$(C_{1L}, C_{1R}), \dots (C_{100L}, C_{100R})$

$H(comm_1, nS_1) \times b_1^e \quad 1 \le 1 \le 100$

A → B

$((I_{JL}, I_{JR})_1 \ 1 \le J \le 100)_{1 \le N \le 99})$

$H(comm_N, nS_N)^d \times b_N$

$[comm, nS, h(comm_N, nS_N)^d]$

A ← M

$b_1, b_2, \dots b_{100}$

$I_{1b_1}, \dots, I_{100 b_{100}}$

$[comm, nS, h(comm, nS)]$,
$I_{1b_1}, \dots I_{100 b_{100}}$

$CertA = Alice, PubKA, L, \dots, S(privKCA, Alice \| PubKA, L \dots)$

A → CA

$RQS, Alice$

- $S_{CA}(Alice \| PubKA \| L_A)$
- $CertA = [Alice \| PubKA \| L_A \| S_{CA}]$

$RSP, CertA, PrivKA$