Digital signature and Public Key Infrastructures

Valentina Casola

Common Uses of Public-Key Cryptography

- Secure E-mail and other communications
  - Secure electronic communications between individuals
  - S/MIME standard
  - PEC
- Secure WWW transactions
  - Consumer-merchant purchases
  - On-line banking
  - SSL, TLS, HTTPs
- Business-to-business transactions
  - Electronic Data Interchange
  - Electronic Trading
- Other e-commerce solutions

Requirements for commercial applications

- Confidentiality
- Integrity
- Authenticity
- Non-repudiation

Traditional paper-based solutions

Confidentiality	Envelopes
Integrity	Signatures, Watermarks, Barcodes
Authenticity	Notaries, strong ID, physical presence
Non-repudiation	Signatures, receipts, confirmations
Availability	Alternate routes, sites, etc.

Electronic Threats

- Confidentiality
  - Eavesdropping (listen for secrets)
- Integrity
  - Modification of data, viruses
- Authenticity
  - Spoofing, …
- Availability
  - SYN flooding, …..

Electronic Solutions

Confidentiality	Data Encryption
Integrity	Digital Signatures, Certificates, Digital Ids
Authenticity	Hash Algorithms, Message Digests, Digital Signatures
Non-repudiation	Digital Signatures, Audit Logs
Availability	Redundant Systems, Automatic Failover

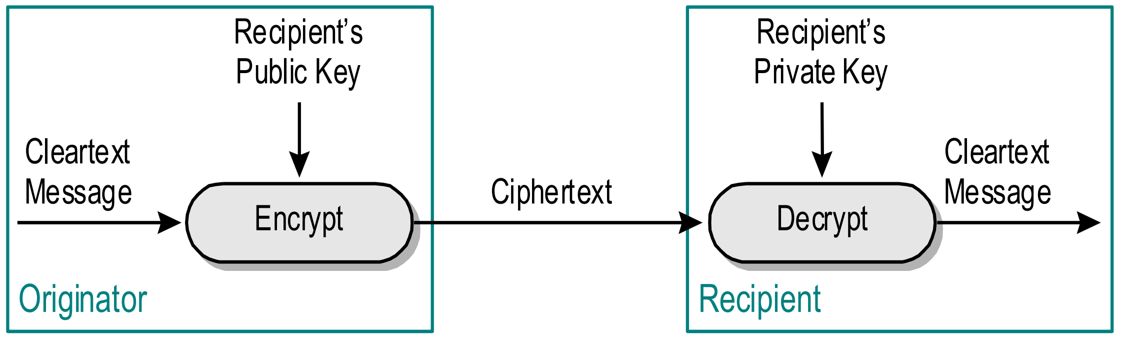
Adding Confidentiality 1

- **Symmetric** Cryptography
  - Single key, shared secret
- Problems:
  - key exchange in large environments
  - lifetime vs. length of key
  - “brute force attacks”

Secret Key
Secure Channel

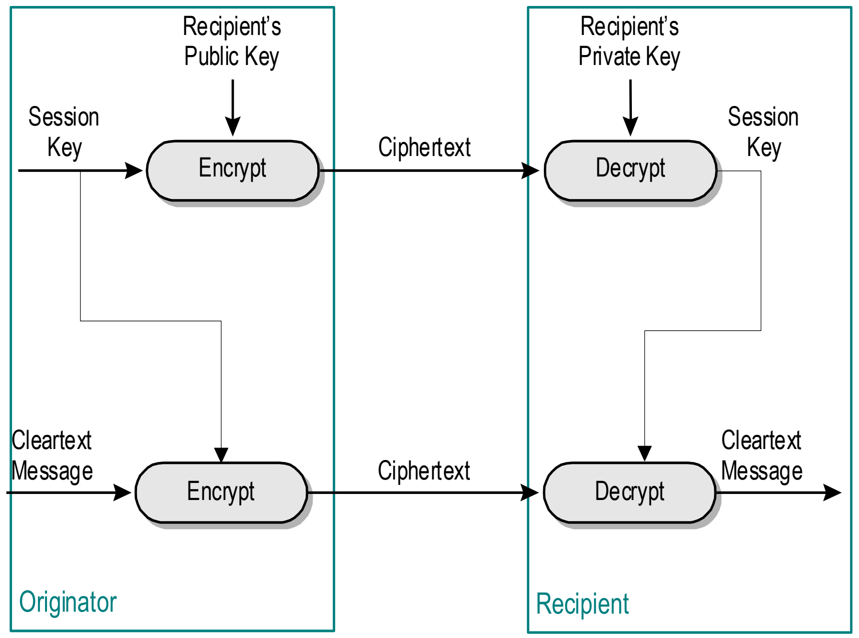
Adding Confidentiality 2

- **Asymmetric** (public-key) cryptography
  - Two keys used: public key and private key
  - Either can be used for encryption/decryption
- Problems:
  - Computationally intensive



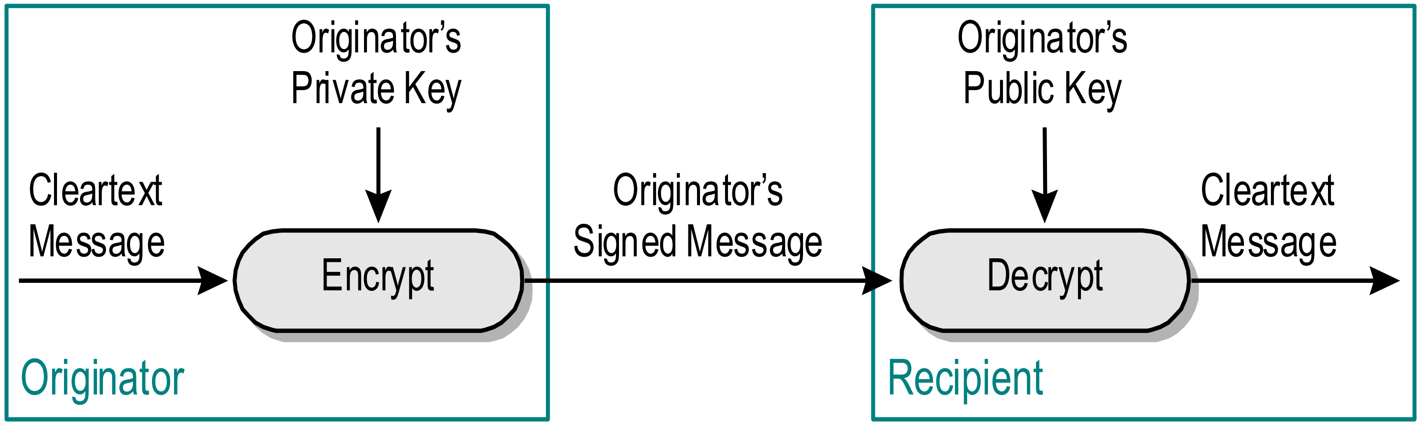
Adding Confidentiality 3

- **Key exchange** using asymmetric cryptography
  - Uses asymmetric keys to distribute encryption keys
  - Allows rapid distribution of short-term keys (session keys)



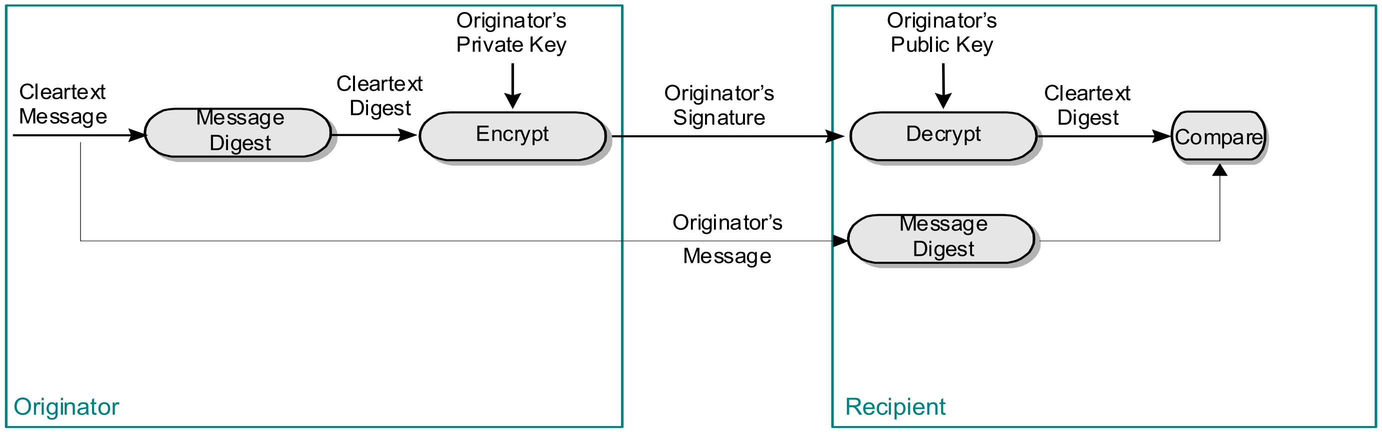
Adding Authenticity

- Asymmetric crypto
  - used to verify authenticity of origin



Adding Integrity and Non-Repudiation

- Digital Signatures (detailed later)
  - Also add integrity of data
  - Non-repudiation of message origin and content



Note on integrity and secrecy

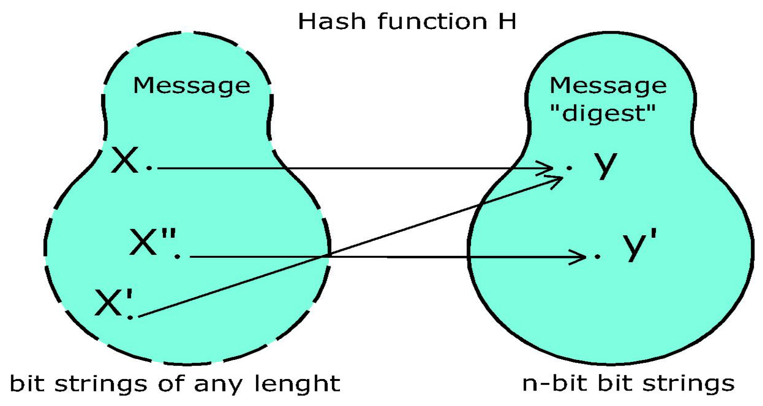
- **Integrity**: attacker cannot tamper with message
- Encryption may not guarantee integrity!
  - Intuition: attacker may able to modify message under encryption without learning what it is
  - Given one-time key K, encrypt M as MK… Perfect secrecy, but can easily change M under encryption to MM' for any M'
- "RSA encryption is intended primarily to provide confidentiality… It is not intended to provide integrity"
- Some encryption schemes provide secrecy AND integrity

Integrity

- Software manufacturer wants to ensure that the executable file is received by users without modification…
- Sends out the file to users and publishes its hash in NY Times
- The goal is integrity, not secrecy

- Idea: given goodFile and hash(goodFile), very hard to find badFile such that hash(goodFile)=hash(badFile)

Hash functions



- H is a lossy compression function
  - Collisions: h(x)=h(x') for some inputs x, x'
  - Result of hashing should "look random"
- Cryptographic hash function needs a few properties…

One way functions

Intuition: hash should be hard to invert

- Let h(x'))=y   for a random x'

Given y, it should be hard to find any x such that h(x)=y

How much hard?

- Brute-force: try every possible x, see if h(x)=y
- SHA-1 (common hash function) has 160-bit output

Some hash functions

- MD5
  - 128-bit output
  - Designed by Ron Rivest, used very widely

- RIPEMD-160
  - 160-bit variant of MD-5

- SHA-1 (Secure Hash Algorithm)
  - 160-bit output
  - US government (NIST) standard as of 1993-95
  - Also the hash algorithm for Digital Signature Standard (DSS)

Electronic and Digital Signatures

"Electronic Signature" and "Digital Signature" do **not** mean the same thing especially in the law context (US and Italy)

The term ''electronic signature'' means an electronic sound, symbol, or process, attached to or logically associated with a contract or other record and executed or adopted by a person with the intent to sign the record.

(*Electronic Signatures in Global and National Commerce Act, E-Sign*)

Both are "electronic", but Electronic Signature, as it is defined in US and Italian law, does not involve any cryptographic technique ensuring identity, integrity, etc…

Digital Signature

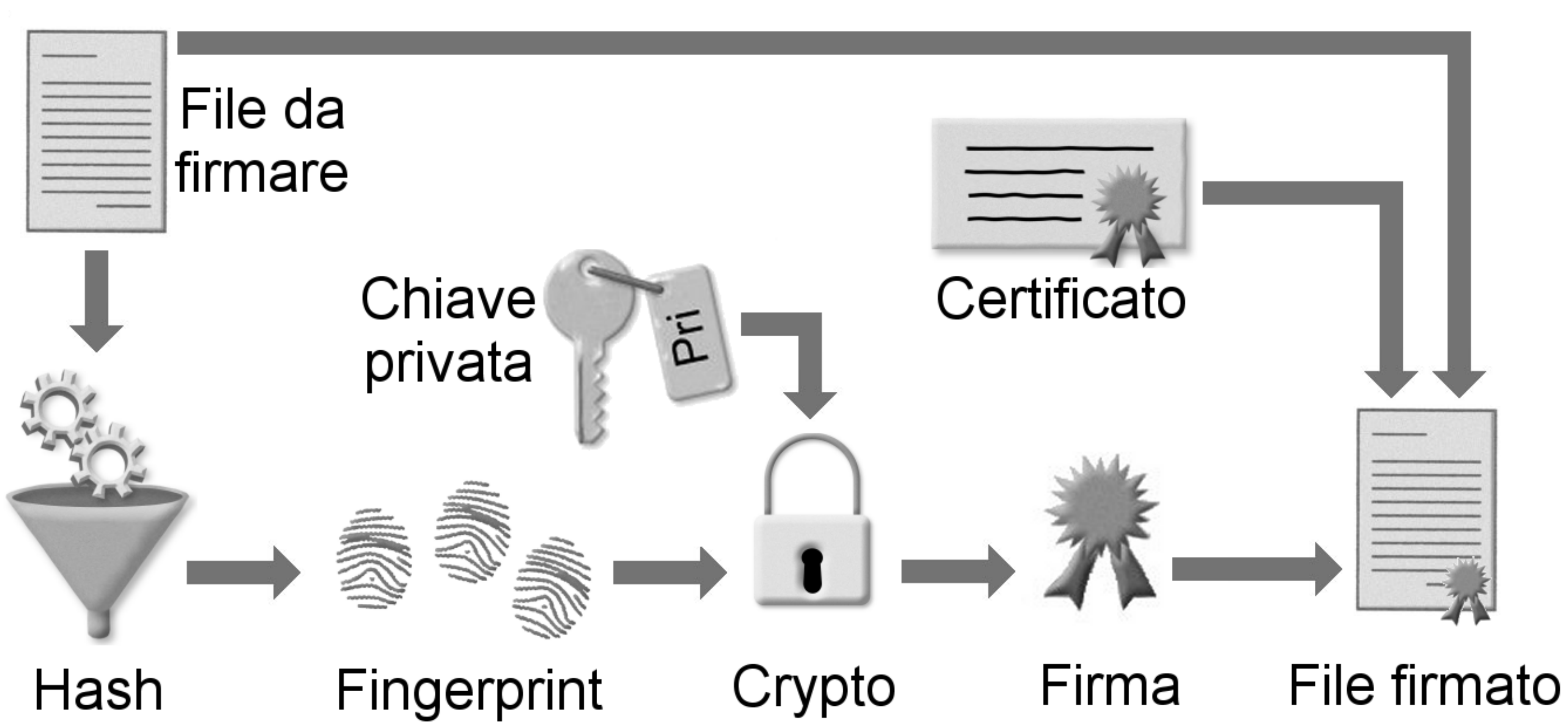
It is a <u>type of Electronic Signature, it combines one-way secure hash functions with public key cryptography:</u>

- Hash function generates fixed length value
- No two documents produce the same hash value
- Secure Hash Algorithm 1 (SHA-1)

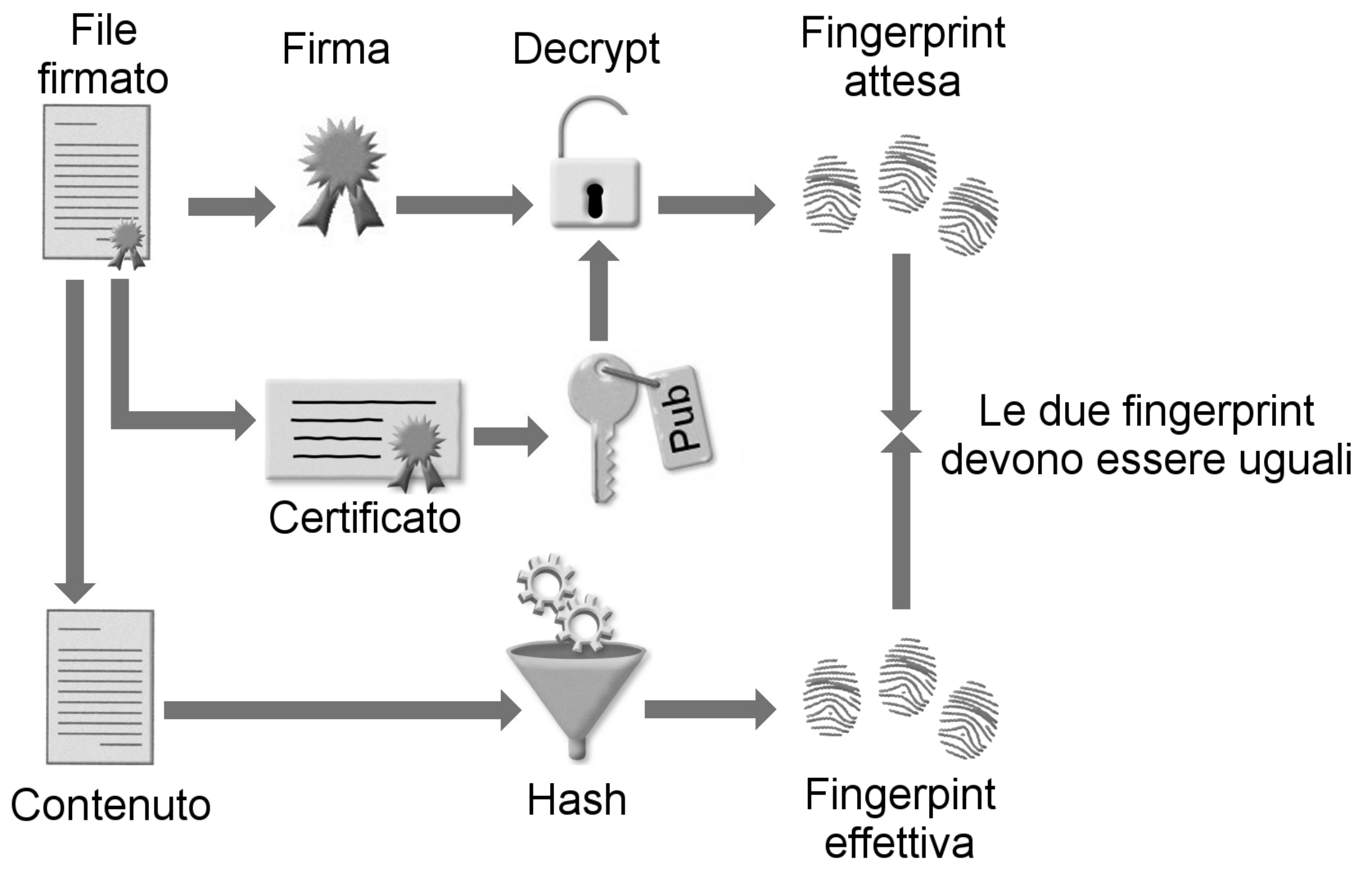
Characteristics:

- **Data Integrity** - hash value
- **Non-repudiation** – encrypted with private key
- Does NOT provide confidentiality

Signature production



Signature validation



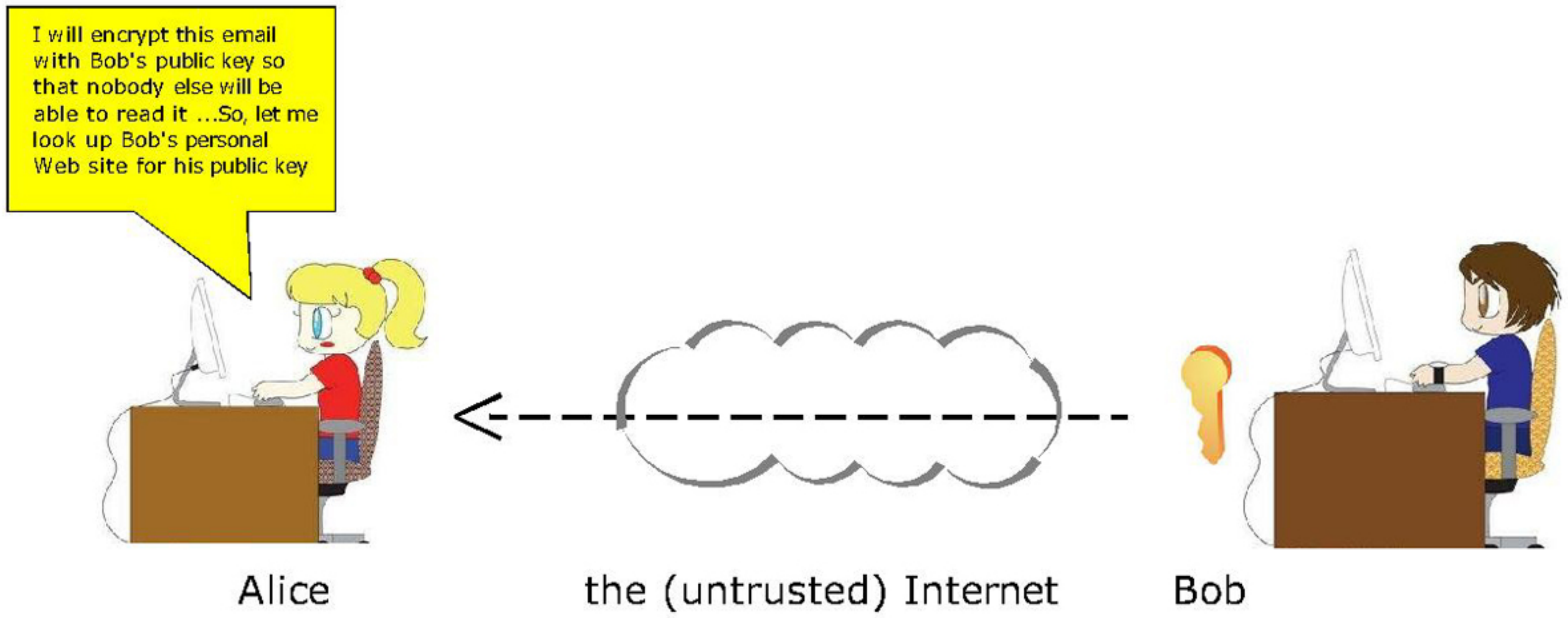
Are public keys “secure”?

Public key algorithms support several security attributes:

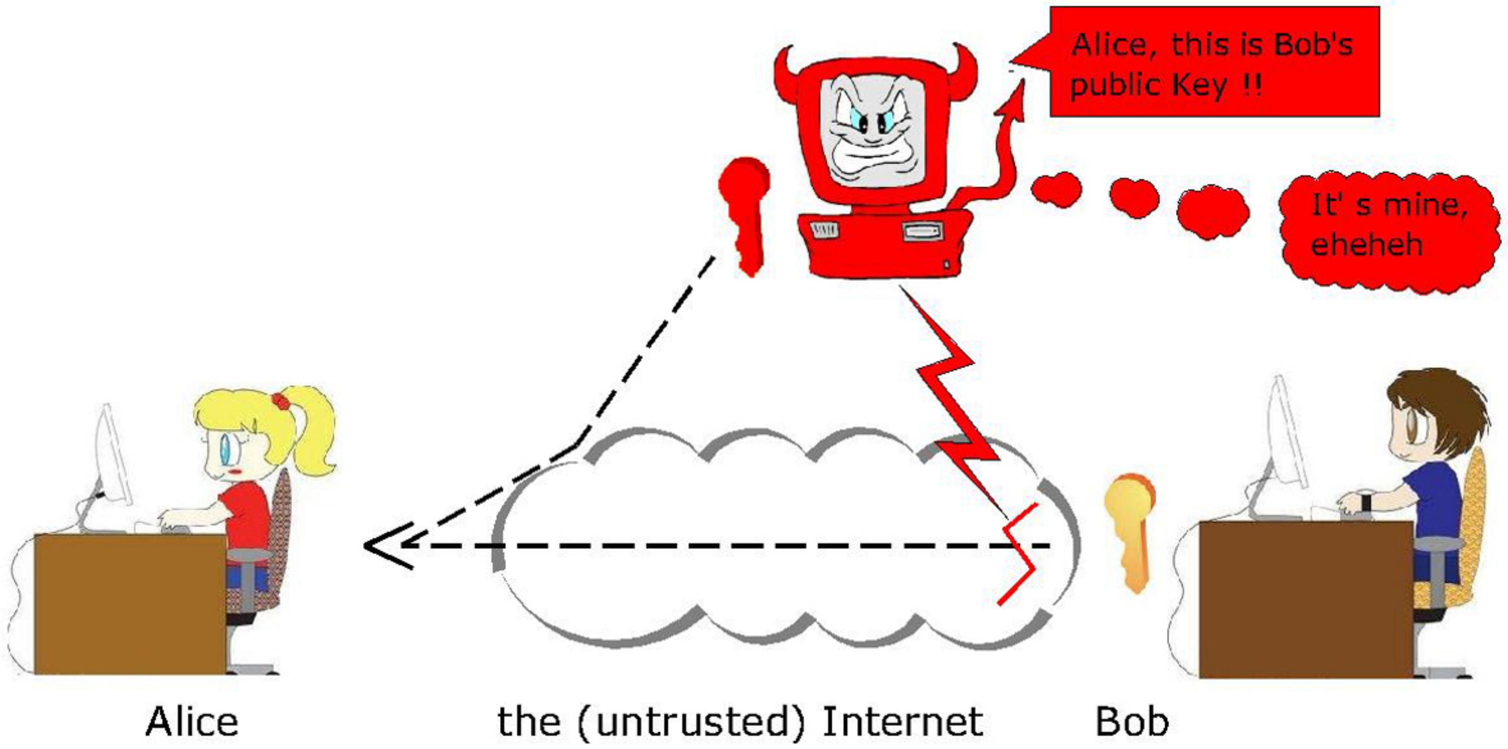
- authentication
- integrity
- non-repudiation
- confidentiality (if used with secret key algorithms)

…but application security depends on how you use them!!

Authenticity of Public Key

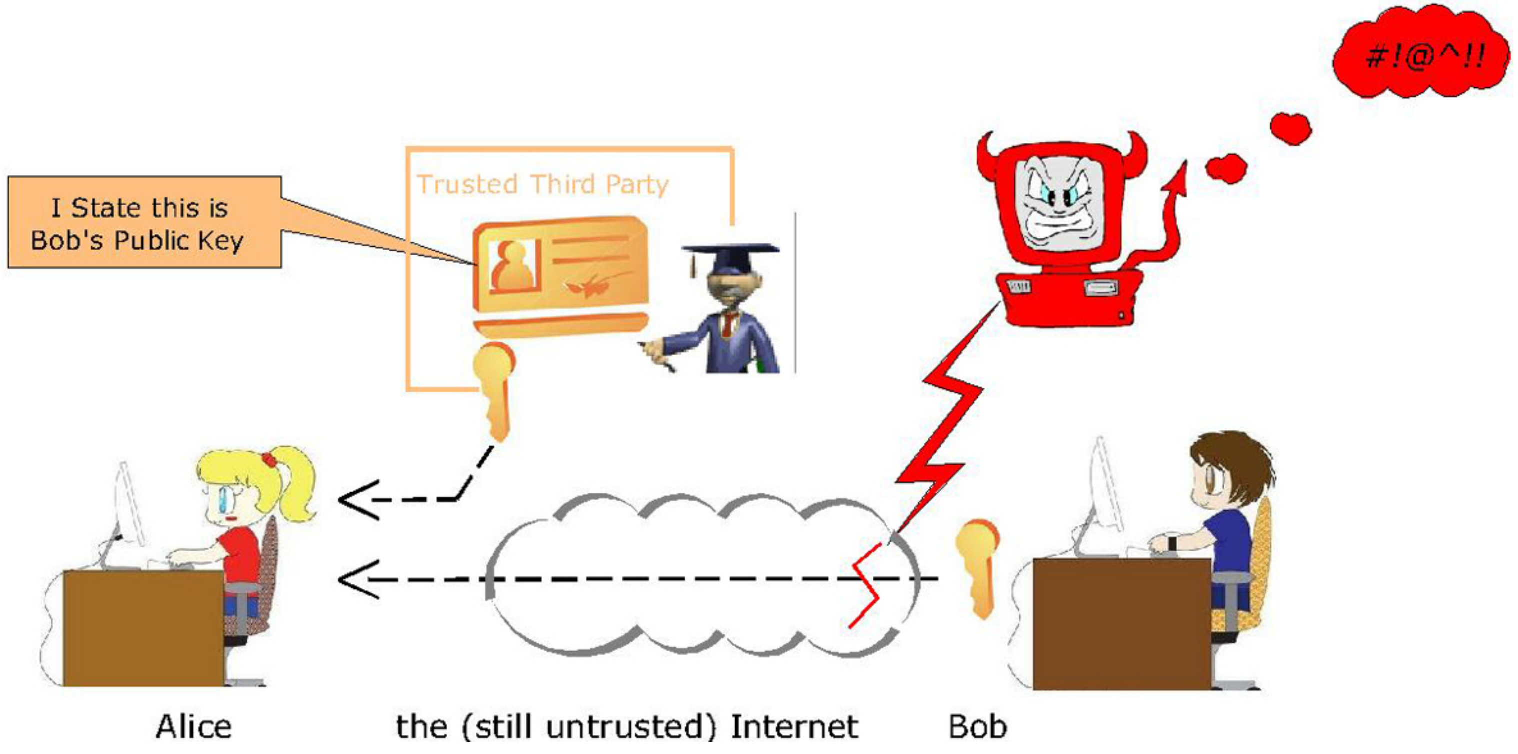

I will encrypt this email with Bob's public key so that nobody else will be able to read it …So, let me look up Bob's personal Web site for his public key
Alice
the (untrusted) Internet
Bob

Authenticity of Public Key (2)



Problem: How does Alice know that the public key she received is really Bob’s public key?

The solution


I State this is Bob's Public Key
Trusted Third Party
#!@^!!
Alice
the (still untrusted) Internet
Bob

Rely on some trusted third-party attesting user’s identity

Requirements of Public-Key systems

- SECRECY of the private key
  - Must be known only to owner
  - **Key ownership = Identity**
- AVAILABILITY of the public key
  - Must be available to anyone
  - Requires a public directory

Public Key Infrastructures

Public Key Infrastructure (PKI) provides the
means to bind public keys to their owners
and helps in the distribution of reliable public
keys in large heterogeneous networks.
NIST

The set of hardware, software, people, policies
and procedures needed to create, manage, store,
distribute, and revoke Public Key Certificates based
on public-key cryptography.
IETF PKIX working group

The Key distribution problem

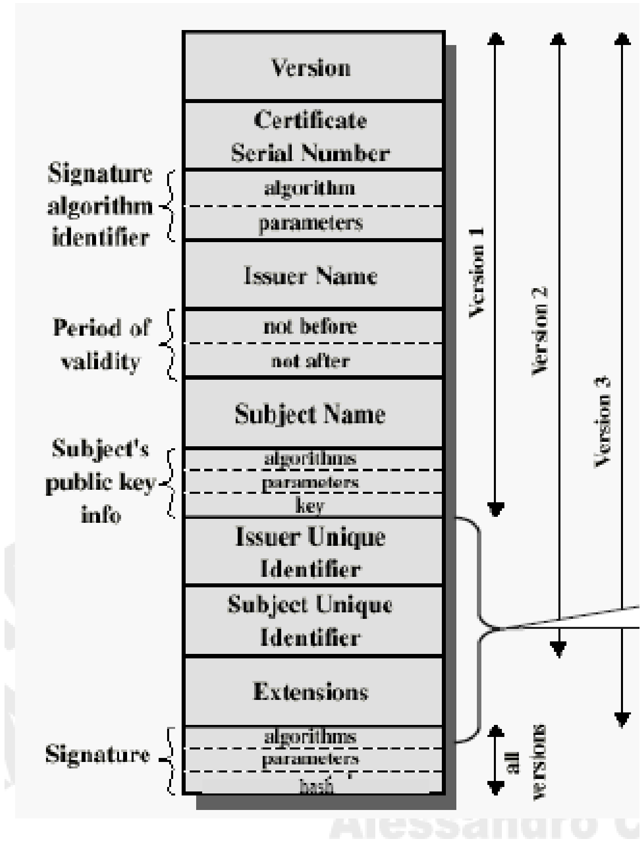
- Public-key certificate
  - Signed statement specifying the key and identity
- Common approach: certificate authority (CA)
  - **Single agency** responsible for certifying public keys
  - After generating a private/public key pair, user proves his identity and knowledge of the private key to obtain CA's certificate for the public key (offline)
  - Every computer is pre-configured with some CA's public key

Digital Certificates

Digital Certificates

- Binds a public key to its owner
- Issued and digitally signed by a *trusted* third party (the CA)
- Like an electronic photo-id
- X.509 V3 standard – RFC 2459, RFC 3280

Digital Certificates x.509 v3



x.509 fields

- **Version**: v1, v2, or v3.
- **Serial #**: a unique number.
- **Signature method**: The method used to sign the digital certificate (e.g., RSA).
- **Issuer name**: The entity whose private key signed the certificate.
- **Valid time period**: begin time and end time.
- **Subject name**: The entity whose public key is included in the certificate.
- **Subject's public key**: public key and public key method.

“Software” Key Store

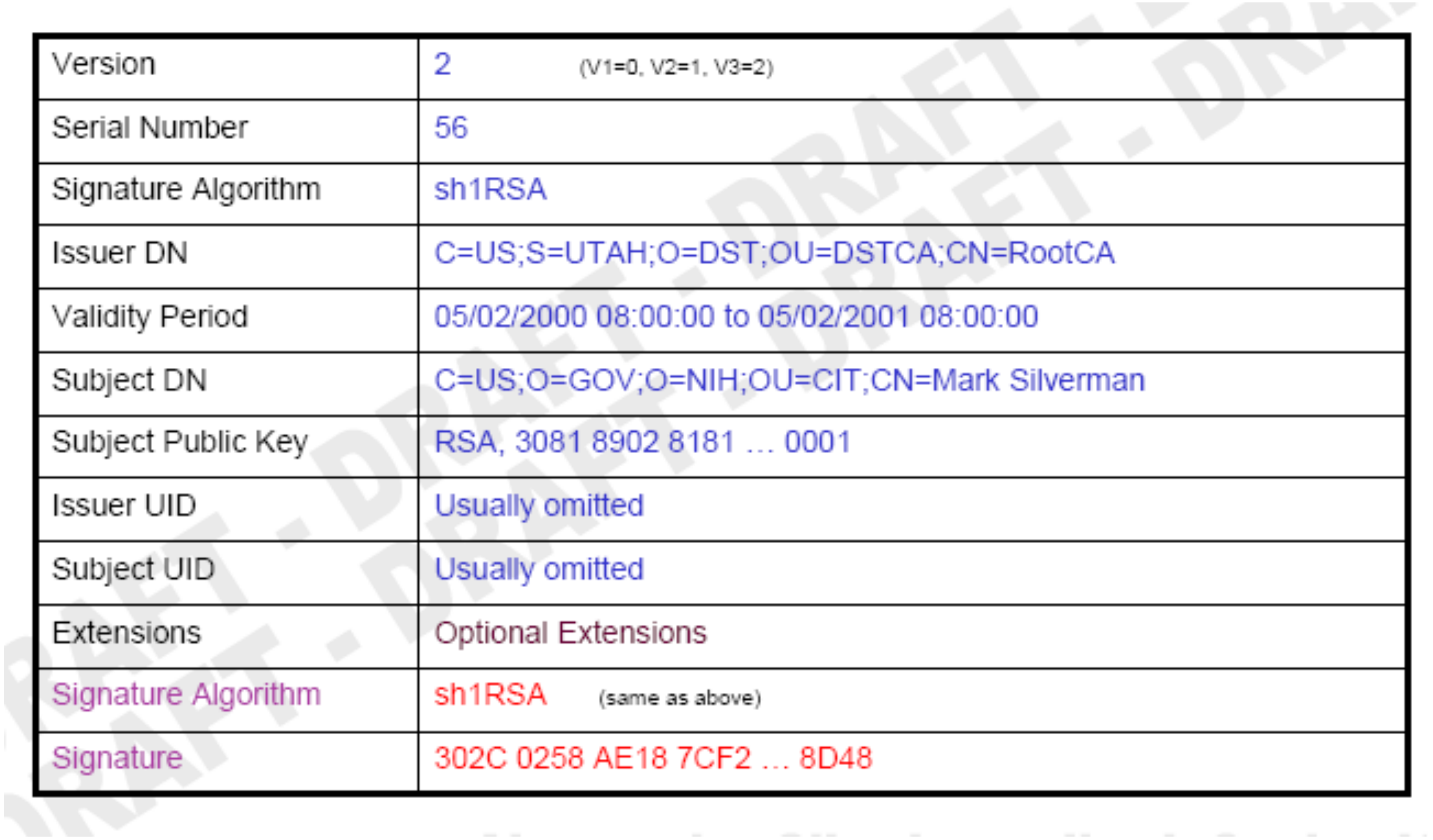
- Stores certificates and encrypted keys on the local computer’s file system
- Encryption is password protected
- Relatively vulnerable to key theft (depending on implementation)
- Requires exporting and importing to use the key on another computer or in a different key store on the same computer

All PKI applications support this type of key store – for some it is the only type supported.

“Hardware” Key Store

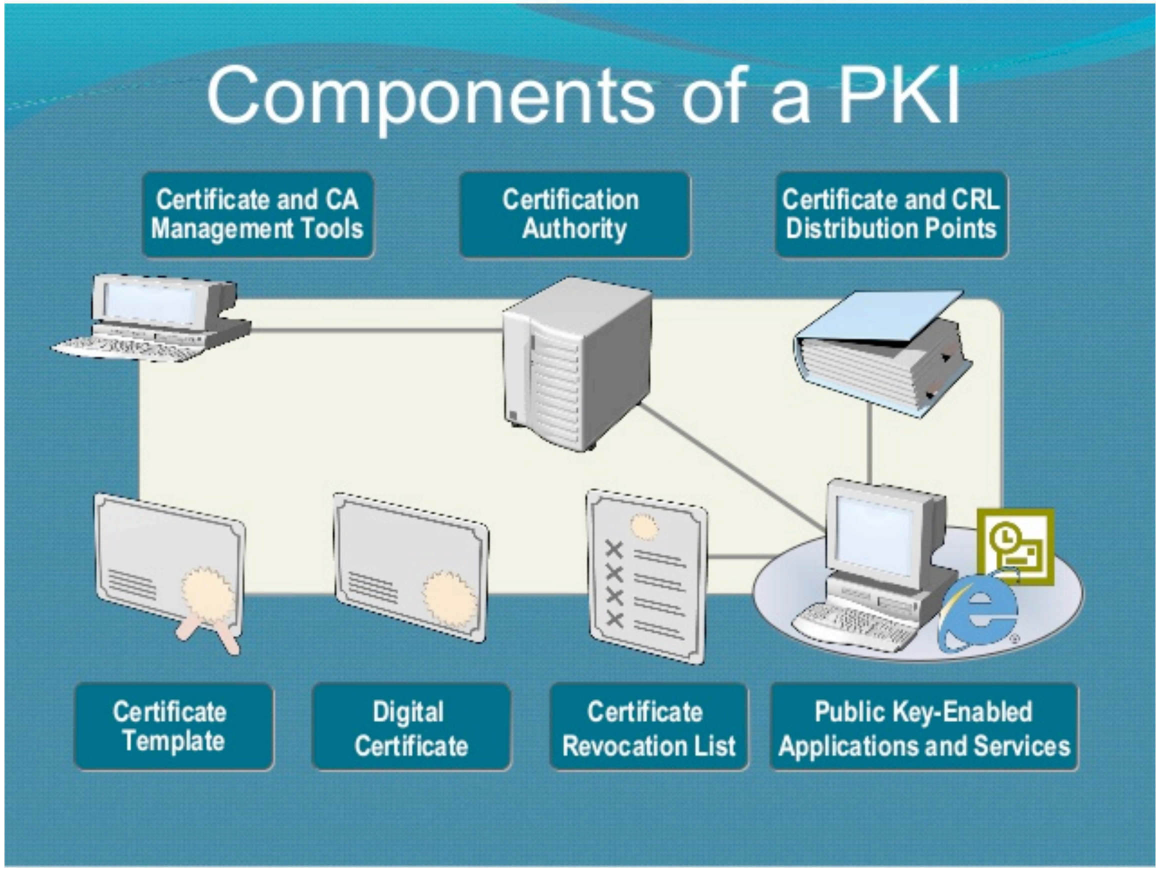
- Stores certificates and keys in special purpose hardware (typically USB token or smart card and reader)
- Much higher assurance - the key cannot be used without the user’ s password, but still not unbreakable
- Allows easy private key mobility between computers and applications
- Two-factor security (need token plus password to do anything) makes hardware key stores much more secure than software key stores

x.509 fields



Version	2 (V1=0, V2=1, V3=2)
Serial Number	56
Signature Algorithm	sh1RSA
Issuer DN	C=US;S=UTAH;O=DST;OU=DSTCA;CN=RootCA
Validity Period	05/02/2000 08:00:00 to 05/02/2001 08:00:00
Subject DN	C=US;O=GOV;O=NIH;OU=CIT;CN=Mark Silverman
Subject Public Key	RSA, 3081 8902 8181 … 0001
Issuer UID	Usually omitted
Subject UID	Usually omitted
Extensions	Optional Extensions
Signature Algorithm	sh1RSA (same as above)
Signature	302C 0258 AE18 7CF2 … 8D48

Components of the PKI


Components of a PKI
Certificate and CA Management Tools
Certification Authority
Certificate and CRL Distribution Points
Certificate Template
Digital Certificate
Certificate Revocation List
Public Key-Enabled Applications and Services

Components of the PKI

- End Users
- Certification Authorities
- Registration Authorities
- Certificate Directories
- Root CA(s)
- Certification Practice Statements (CPS)
- Certificate Management Protocols & APIs

Major Issues with CAs and RAs

- End Entity Registration
- Trust models
- Certification Practice Statement (CPS)
- Key management
- Certificate Revocation
- Publishing Issues
- Ownership and Maintenance
- Liability

Public-Key Cryptography Standards

- PKCS #1: RSA Cryptography Standard
- PKCS #3: Diffie-Hellman Key Agreement Standard
- PKCS #5: Password-Based Cryptography Standard
- PKCS #6: Extended-Certificate Syntax Standard
- PKCS #7: Cryptographic Message Syntax Standard
- PKCS #8: Private-Key Information Syntax Standard
- PKCS #9: Selected Attribute Types
- PKCS #10: Certification Request Syntax Standard
- PKCS #11: Cryptographic Token Interface Standard
- PKCS #12: Personal Information Exchange Syntax Standard
- PKCS #13: Elliptic Curve Cryptography Standard
- PKCS #15: Cryptographic Token Information Format Standard

Major Questions about PKI Deployment

- What mechanisms do users have to trust each other?
- How can users protect the uniqueness of their private key?
- What components of the PKI can be outsourced?
- Who is liable when problems occur?
- How can multiple applications work with each other?

Registration

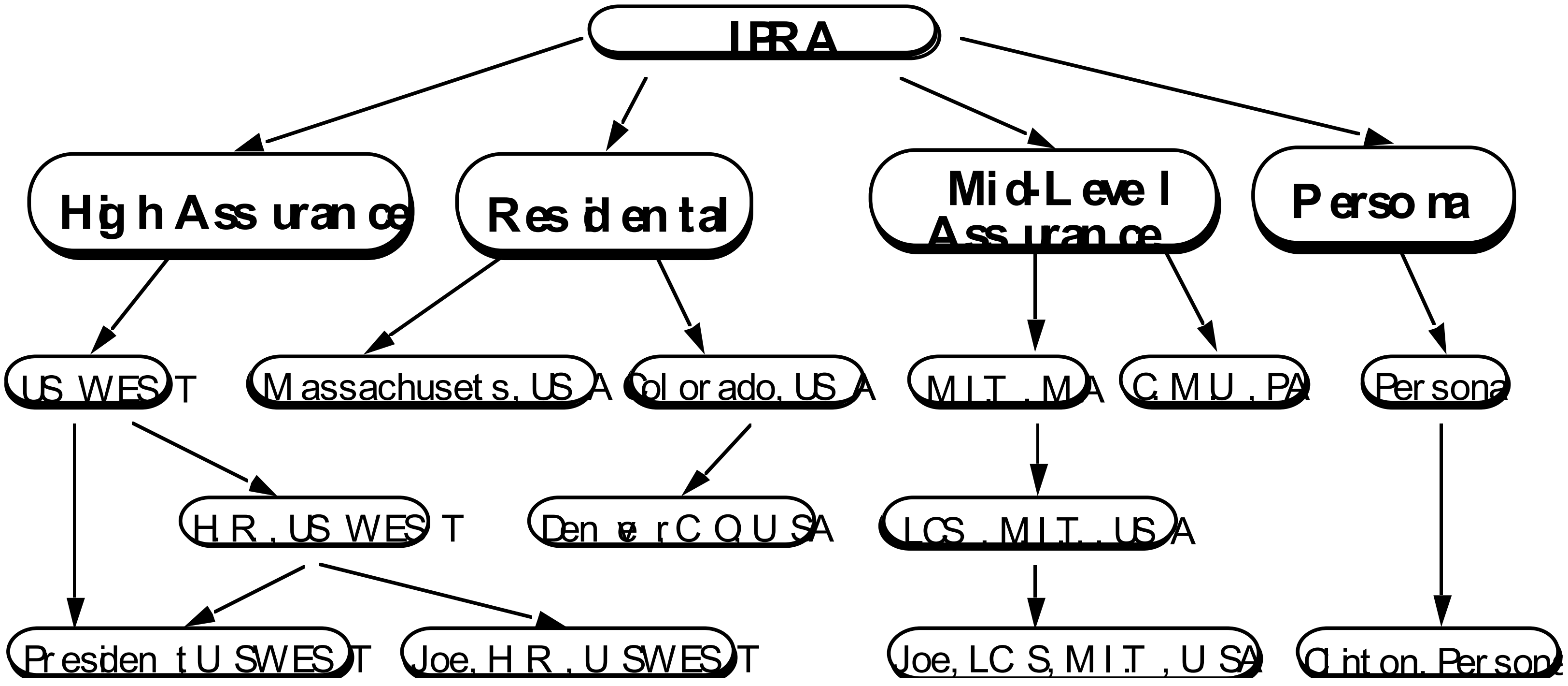
- Registration Authority (RA)
  - verification of user info
  - policy enforcement
  - no liability
  - only handles registration, not re-issuance, revocation, etc.
  - works with CA (generate key in the smart card, send a certificate request, download the digital certificate)
- Registration can be local, or outsourced

Trust Models

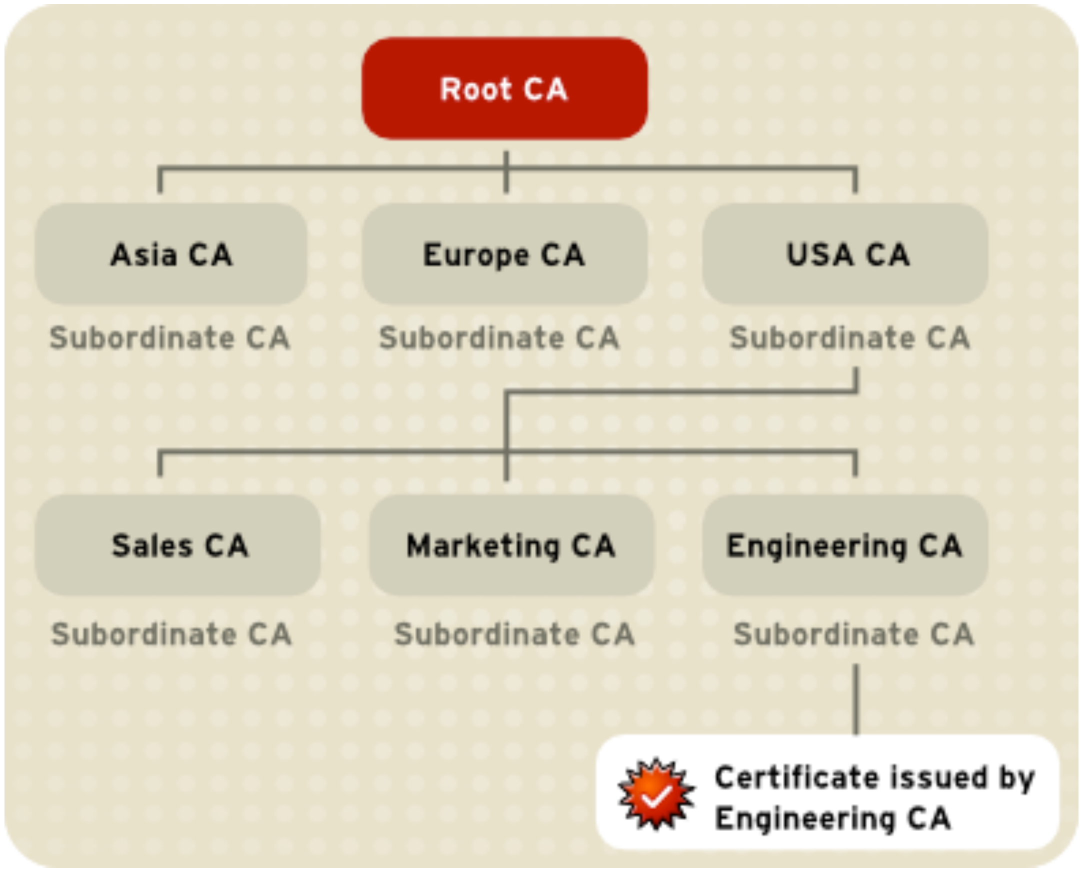
- Hierarchical model
  - Name subordination
- Cross-certification

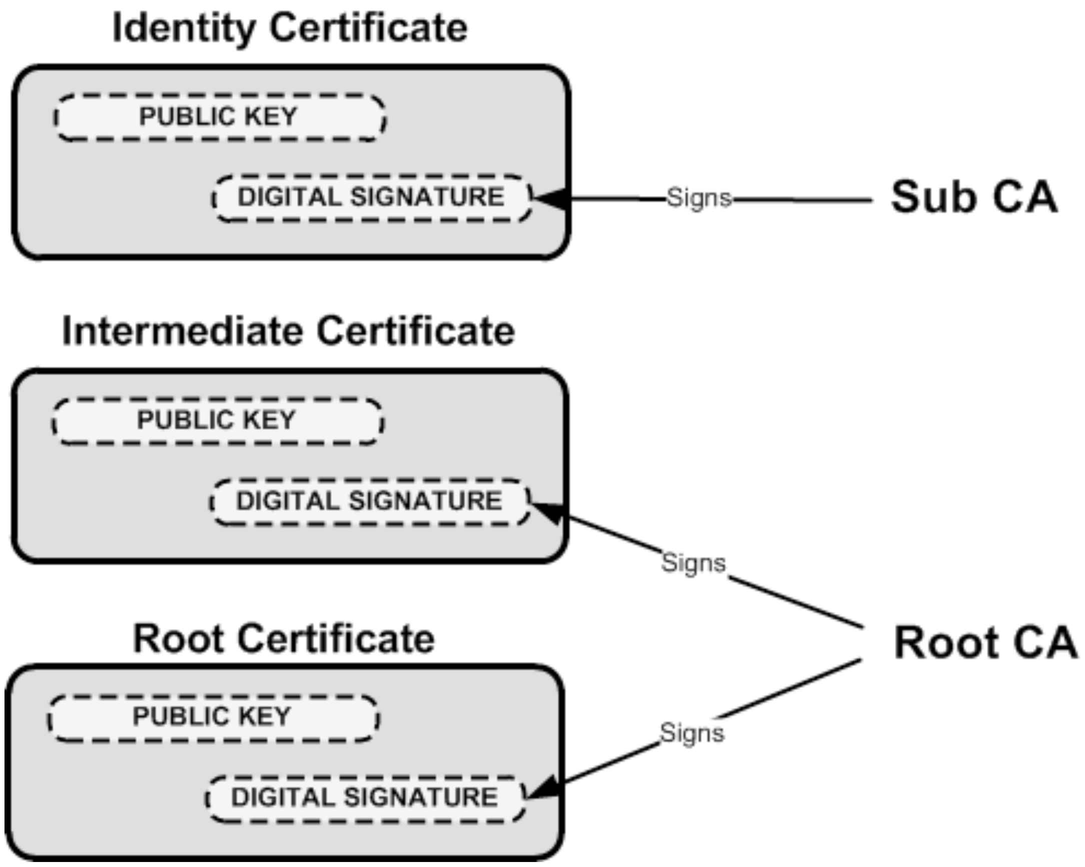
Hierarchical Trust

- Difficult to bootstrap



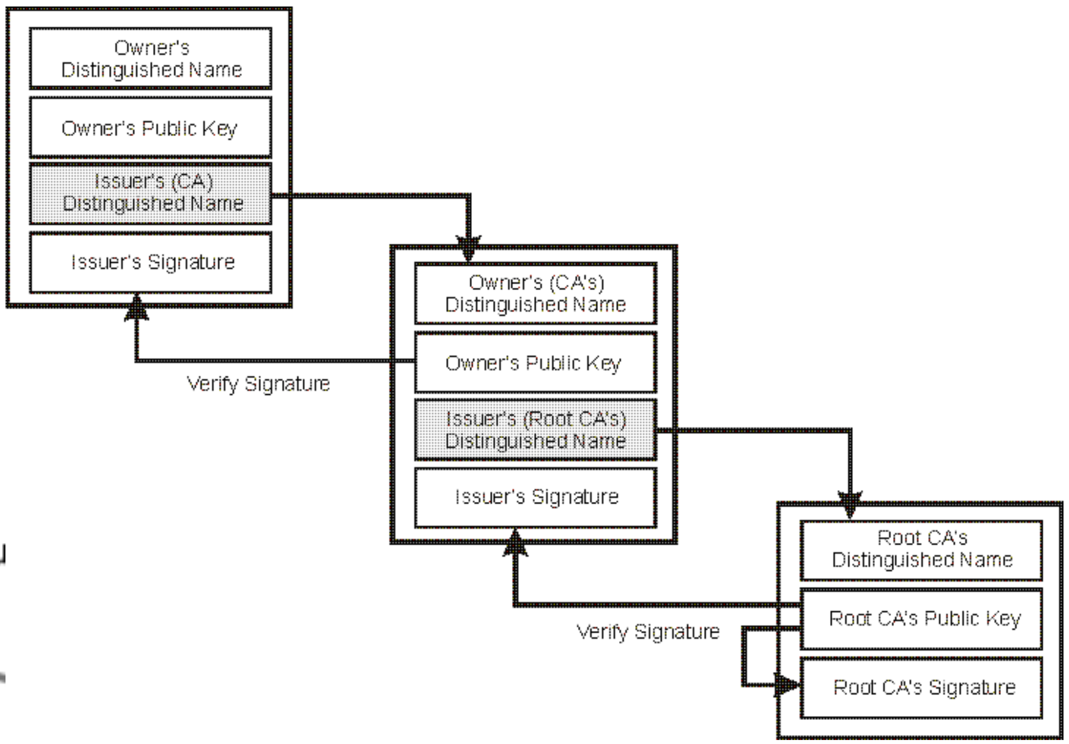
Hierarchical Trust





Identity Certificate
PUBLIC KEY
DIGITAL SIGNATURE
Signs
Sub CA
Intermediate Certificate
PUBLIC KEY
DIGITAL SIGNATURE
Signs
Root CA
Root Certificate
PUBLIC KEY
DIGITAL SIGNATURE
Signs

Certificate Chain


Owner's Distinguished Name
Owner's Public Key
Issuer's (CA) Distinguished Name
Issuer's Signature
Owner's (CA's) Distinguished Name
Owner's Public Key
Issuer's (Root CA's) Distinguished Name
Issuer's Signature
Verify Signature
Root CA's Distinguished Name
Root CA's Public Key
Root CA's Signature
Verify Signature

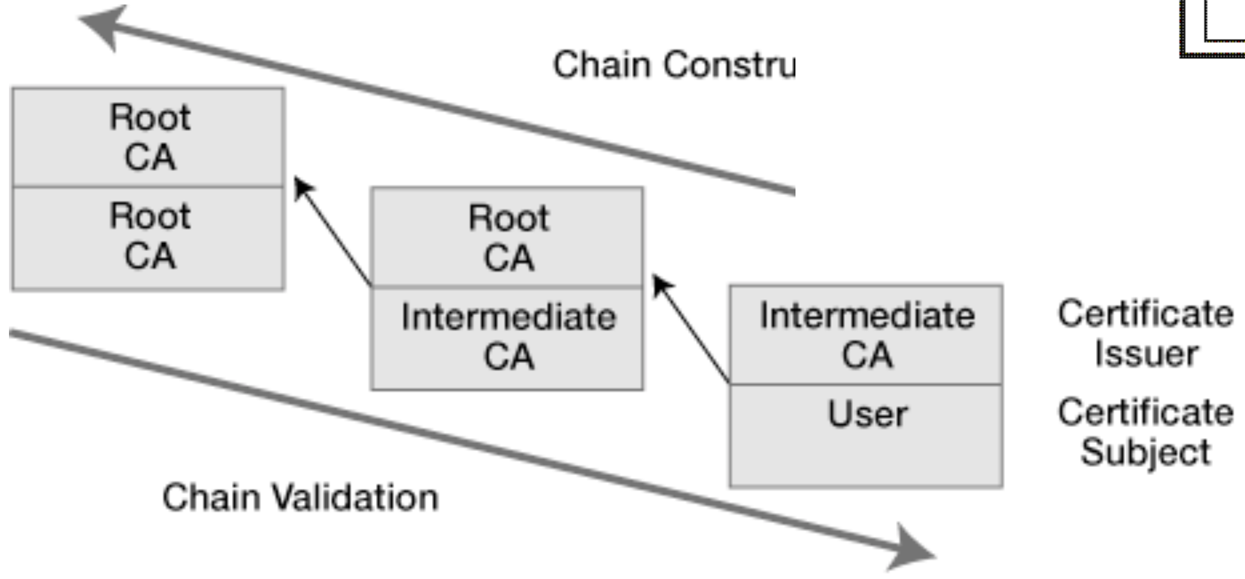
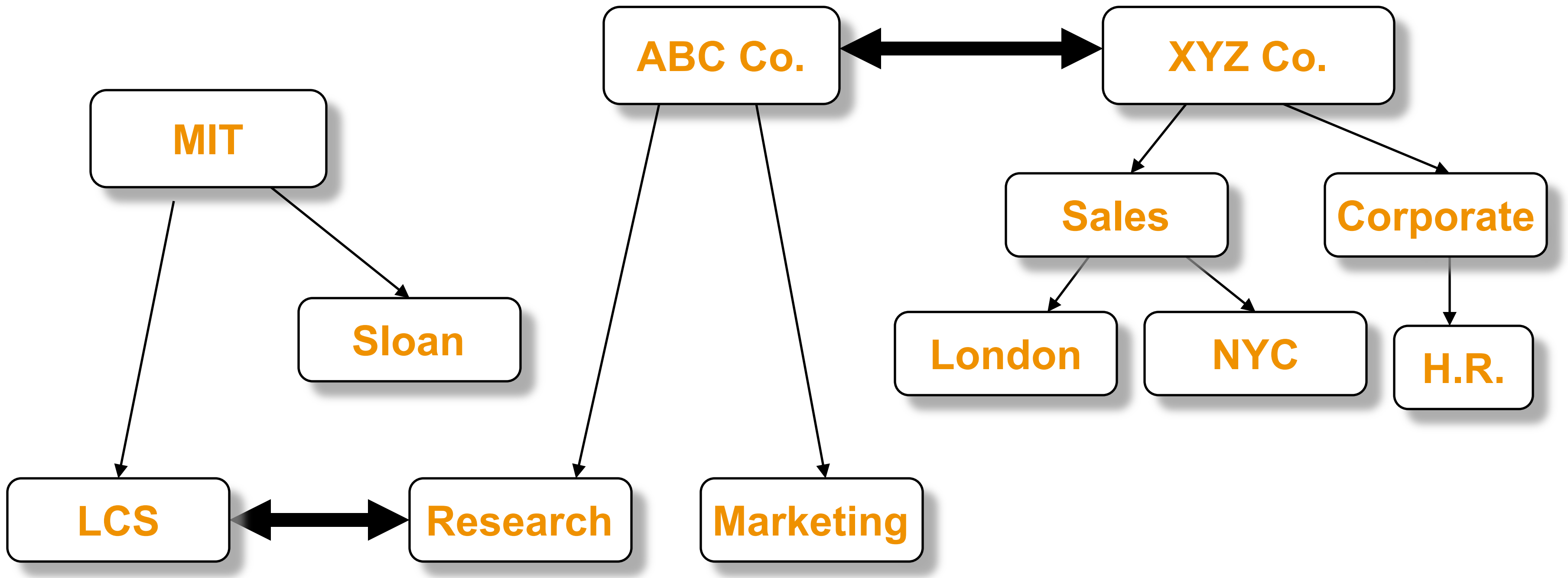


FIGURE 1: Certificate-chain-processing overview

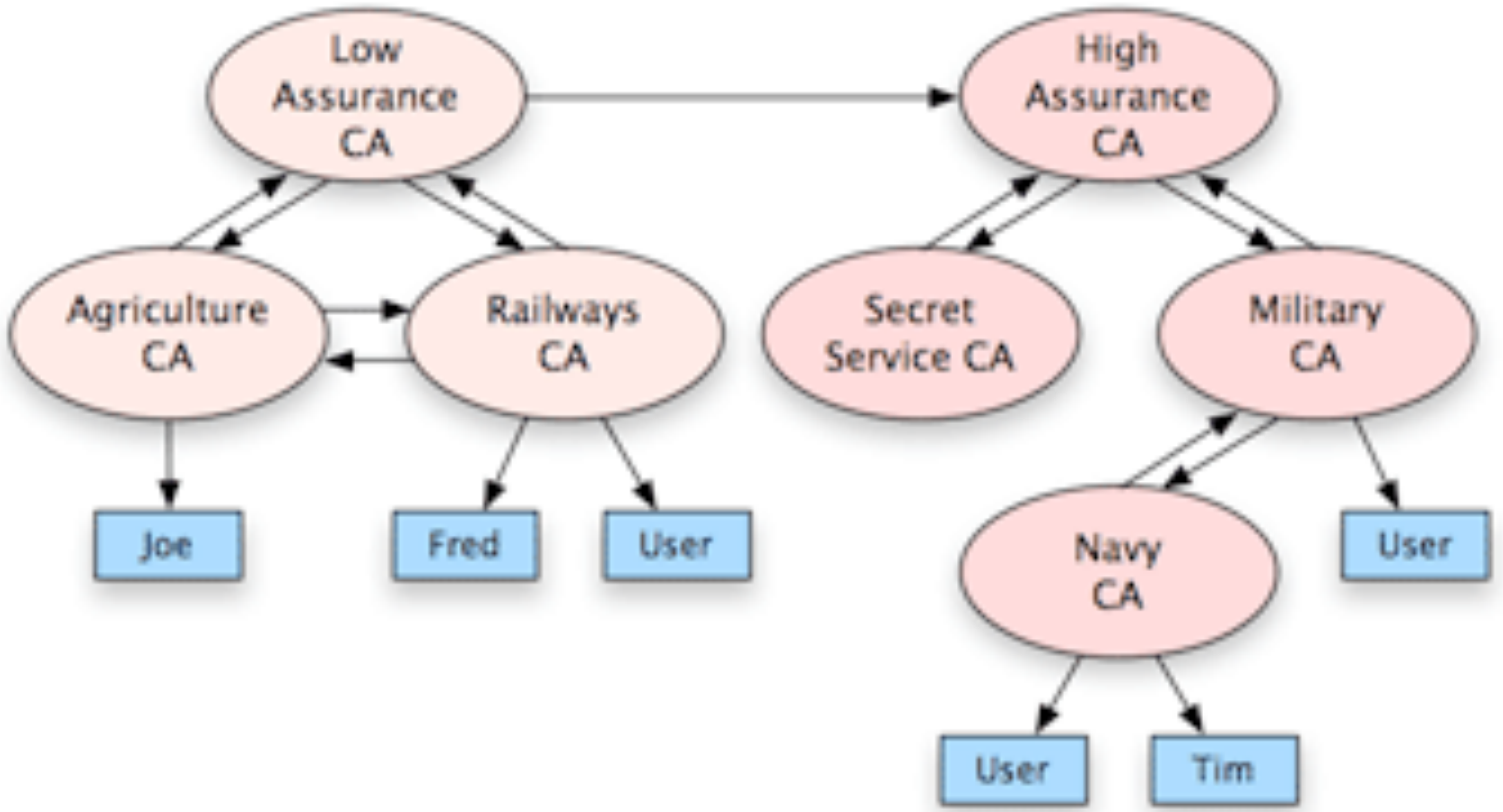
Cross-Certification

- Allows transference of trust between hierarchies



Cross-Certification

Allows transference of trust between hierarchies



Policy Issues

- Verification of Identity
- What is being certified?
- Validity Periods of Certificates
- CRL issuance / Certificate Revocation
- Publishing
- Re-issuance
- Scope of clients
- *All are presented in the Certification Practice Statement (CPS)*

Certification Practice Statement

- Outlines the CA's practices with regard to:
  - certificate issuance and user registration
  - certificate lifetimes and revocation
  - trust model
  - certificate publishing practices
- Designed for other purposes:
  - Awareness of customers
  - Limiting liability
  - Outlining procedures for personnel

Key management

- Generation of key-pairs
  - CA, RA, end entity?
- Storage of private keys at CA
  - smartcards, or on embedded devices
  - Software keystore
- Archival of keys

Certificate Revocation

- What constitutes revocation?
- Push/Pull model of CRLs
- Publishing Issues
- Real-time verification?
- Are CRLs the right model?

Revocation Models

- Certificate Revocation Lists (CRLs)
  - Traditional model
  - Supported by Entrust, Verisign, most CAs
- On-line Certificate Status Protocol (OCSP)
- CRL Distribution Points (CDPs)

New Revocation Models

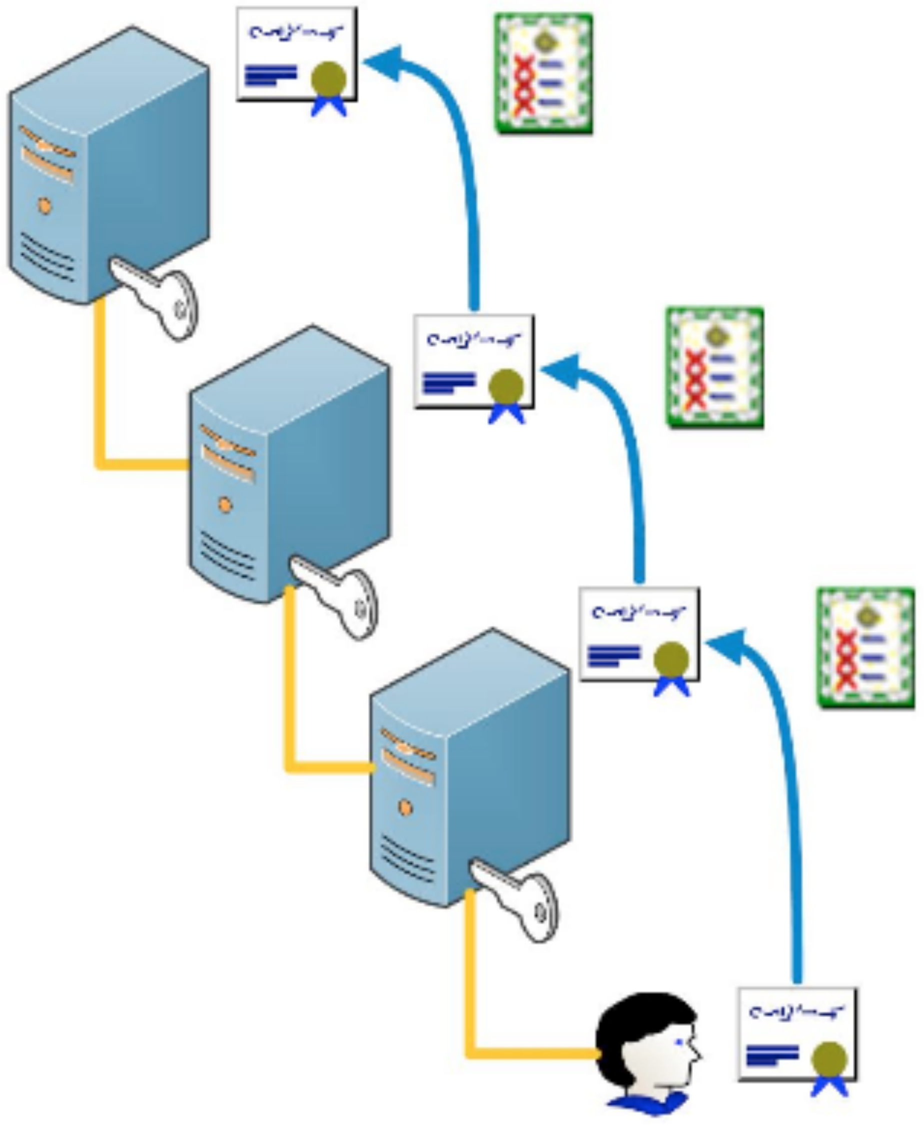
- On-line Certificate Status Protocol (OCSP)
  - IETF proposed protocol - introduced by VeriSign
  - real-time verification of certificates
  - OCSP responders - provide info to clients
  - acceptance suspended pending response
- Certificate Revocation Trees (Valicert)
  - Offers service and product for real-time verification
  - CRL "trees" - contained within product or at server

Certificate Directories

- Lightweight Directory Access Protocol (LDAP)
  - runs on TCP/IP, new life into X.500
- Gaining heavy industry support
  - Microsoft AD, Netscape Directory Servers
- Also included in client products
  - msIE, Netscape Communicator
  - etc.

Certificate validation: basic and extended

- Cryptographic verifications over the certificate path (i.e. verifying the digital signature of each certificate).
- Verifying each certificate validity period.
- Verify that the first certificate in the chain is a Trust Anchor.
- Verify the certificate's status to ensure that it has not been revoked or suspended (via CRL and OCSP).
- Evaluate and/or Compare Policies (manually or via automatic tools)



Interoperability Issues

- Technical Issues
  - Certificate content, extensions, etc.
  - Import/export of data between products
  - Support for revocation
  - Use of directories / publishing issues
  - Frameworks (CDSA, CAPI, etc.)
- Policy Conflicts
  - Registration process
  - Identification
  - Revocation
- Liability Issues
  - Due diligence for CAs, RAs
  - Largely unclear

References

- **RFC3280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile**
- http://tools.ietf.org/html/rfc3280

The Italian state of the art on PKIs

- Concepts:
  - Electronic Signature
  - Advanced Electronic Signature
  - Digital Signature
  - Qualified Certification Authority
  - -> Policy -> manuale operativo
- References: DIGITPA website (AgiD)

Further readings

- Public-Key Cryptography Standards

  http://www.rsa.com/rsalabs/node.asp?id=2124
- MANDATORY: 3 simple exercises with Java Cryptography Architecture:

http://docs.oracle.com/javase/7/docs/technotes/guides/security/crypto/CryptoSpec.html

OpenSSL: www.openssl.org

- MANDATORY: Installing OpenCA, issue certificates,
- NEW:
  - Develop an application for basic certificate validation (extract information from a digital certificate by parsing it)

Java Cryptography Architecture

- Key Pair Generator
- Message Digest
- Signature
- Encryption DES
- Password Based Encryption
- Key store
- Certificate parsing
- …..

OpenSSL functions

Implemented functions:

- Symmetric Cipher: DES, Blowfish, cast, RC2, RC4, RC5,IDEA, AES(to appear)
- Authentication codes and Hash functions: MD2, MD4, MD5, SHA1, RIPEMD 160, MDC2, HMAC
- Public Key Cryptography and Key management: RSA, DSA, D-H,ECC(to appear)
- Certificates: X509, X509v3
- Input/Output, Data Encoding: asn1, bio, evp, pem, pkcs7, pkcs12
- Internal Functions: bn, buffer, lhash, object, stack

OpenSSL Supported Standards

- PKCS#1(full), PKCS#7 (almost complete),
- PKCS#8(full), PKCS10(full), PKCS#12
- X509v3
- ASN.1 with DER encoding
- SSLv3 and TLSv1
- OCSP