Public Key Infrastructure (PKI)

- Reading
  - Chapter 15

Distributing Public Keys

- Public key cryptosystems allow parties to share secrets over unprotected channels
  - Extremely useful in an open network:
    - Parties are not under a single manager
    - Symmetric keys cannot be shared beforehand
- How to distribute public keys?
  - Not a problem of secrecy (symmetric key)
  - A problem of legitimacy (identity binding)

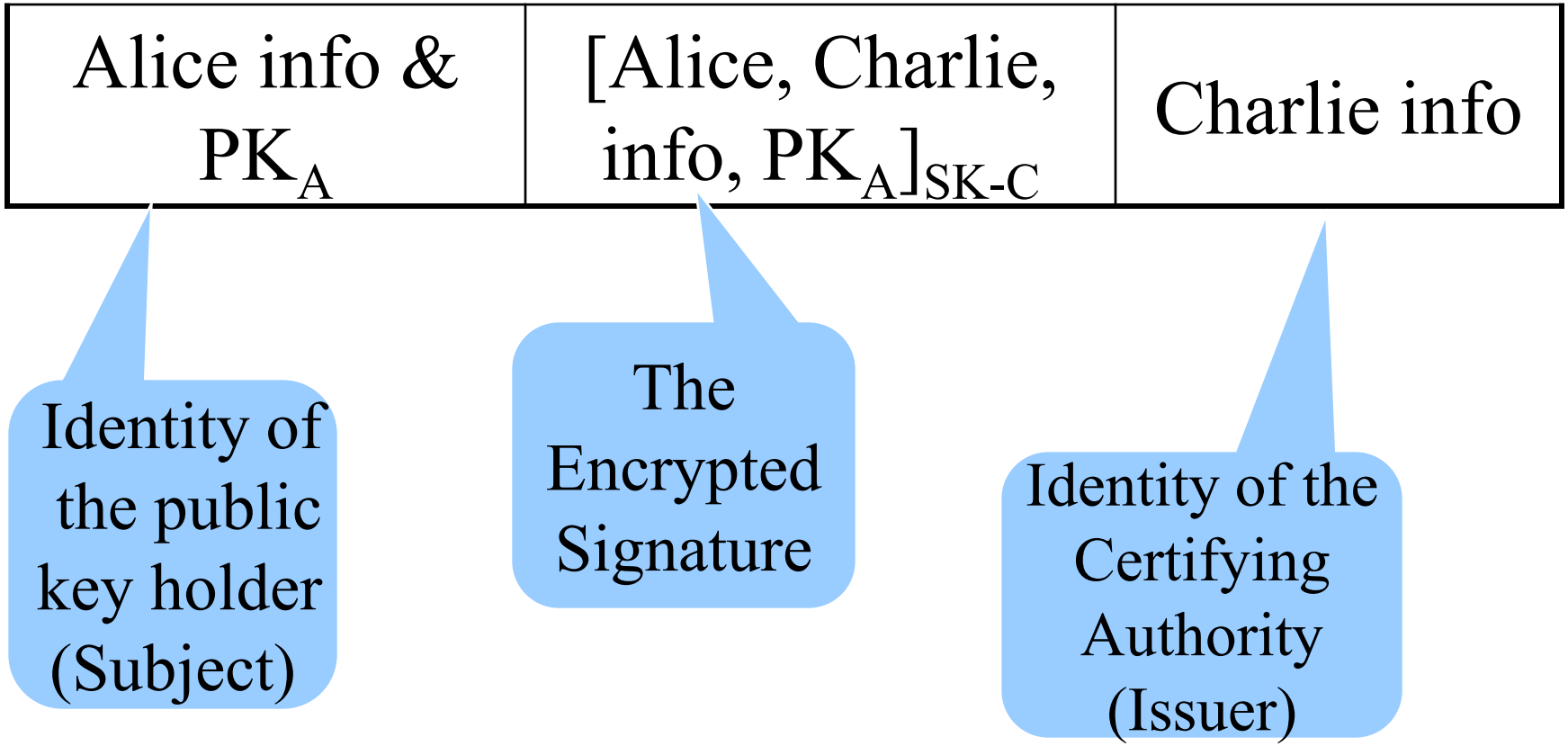
Certification

- Public keys must be certified, i.e., an authenticated statement like "public key PA belongs to user A" must be made by a trusted party.
- The Public Key Infrastructure defines:
  - The set of trusted parties ***or*** a mechanism to infer trust
  - An authentication/certification algorithm

What is a (Certificate based) PKI

- Structure and components to securely distribute public keys
  - Repository for certificates
  - Retrieving and delivering certificates to clients
  - Methodology for registering clients, and revoking certificates
  - Methodology for verifying a certificate (a trust chain)

Example Certificate



Verifying a Certificate

1. Hash the certificate body with the appropriate hashing algorithm
   - The body of the certificate is typically all the information except:
   - The signature algorithm and heading before signature
   - The signature

2. Use issuer's public key to decrypt signature
3. Verify decrypted signature in 2 with hash in 1.

Terminology

- If Alice signs a certificate for Bob,
  - Alice is the issuer, Bob is the subject
- If Alice wants to find a trusted path to Bob's key, Bob's name is the target
- A verifier evaluates a certificate or a chain of certificates
- Anyone having a public key is a principal
- A trust anchor is a public key that the verifier has decided is trusted

PKI Trust Models

- Monopoly model
- Oligarchy model
- Anarchy model

Monopoly Model

- A central Certification Authority (CA) is:
  - universally trusted
  - its public key is known to all
- The central CA signs all public key certificates, or delegates its powers:
  - to lower level CAs: Certificate chaining
  - to registration authorities (RAs): check identities, obtain and vouch for public keys
- *This is a "flat" trust model.*

Problems with Monopoly Model

- Can there be a universally trusted organization?
- Key of the monopoly entity would be difficult to change as it would be embedded in many systems
- Problems of the monopoly entity being corrupt, excessive charges, etc.
- What is the mechanism for getting a certificate?
  - How do I verify my identity
  - How do I provide my public key

Oligarchy Model

- A number of root CAs known in advance
- Certificate chaining is supported
- Web browsers support oligarchic PKIs
  - Come preconfigured with many trust anchors, trusted by the product vendor
  - For Firefox, check Certificates by following Options->Advanced->Encryption->View Certificates
  - Similarly, you can check Certificates on IE
- More security problems than the monopoly model – more points of failure
- The X.509 PKI is oligarchic

Anarchy Model

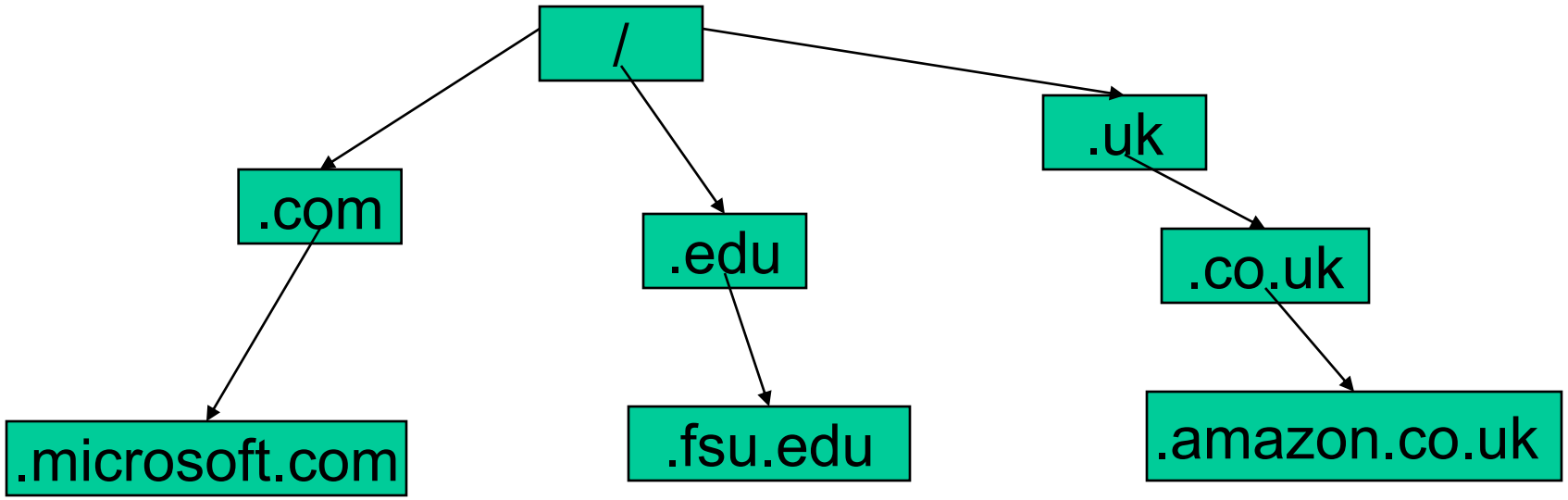
- Each user is fully responsible for deciding its trust anchors (roots).
  - For example, PGP (Pretty Good Privacy)
  - Practical for individual communication
    - Put your public key in your e-mail signature or website
    - Call user to verify PK fingerprint
  - Impractical for automated trust inference
    - How to decide that a certificate chain is trustworthy?
- *"web of trust" versus hierarchical trust model*

Constrained Naming PKIs

- Assumptions:
  - X.509 and other oligarchic PKIs cannot handle a very complex world without becoming very complex themselves
  - Many certification needs are inherently local
  - Local certification and local naming uniqueness can be maintained with minimal effort
  - Global naming conventions exist (e.g.: DNS)
  - If public keys need global certification, then rely on relationships to infer trust

Top-Down Constrained Naming

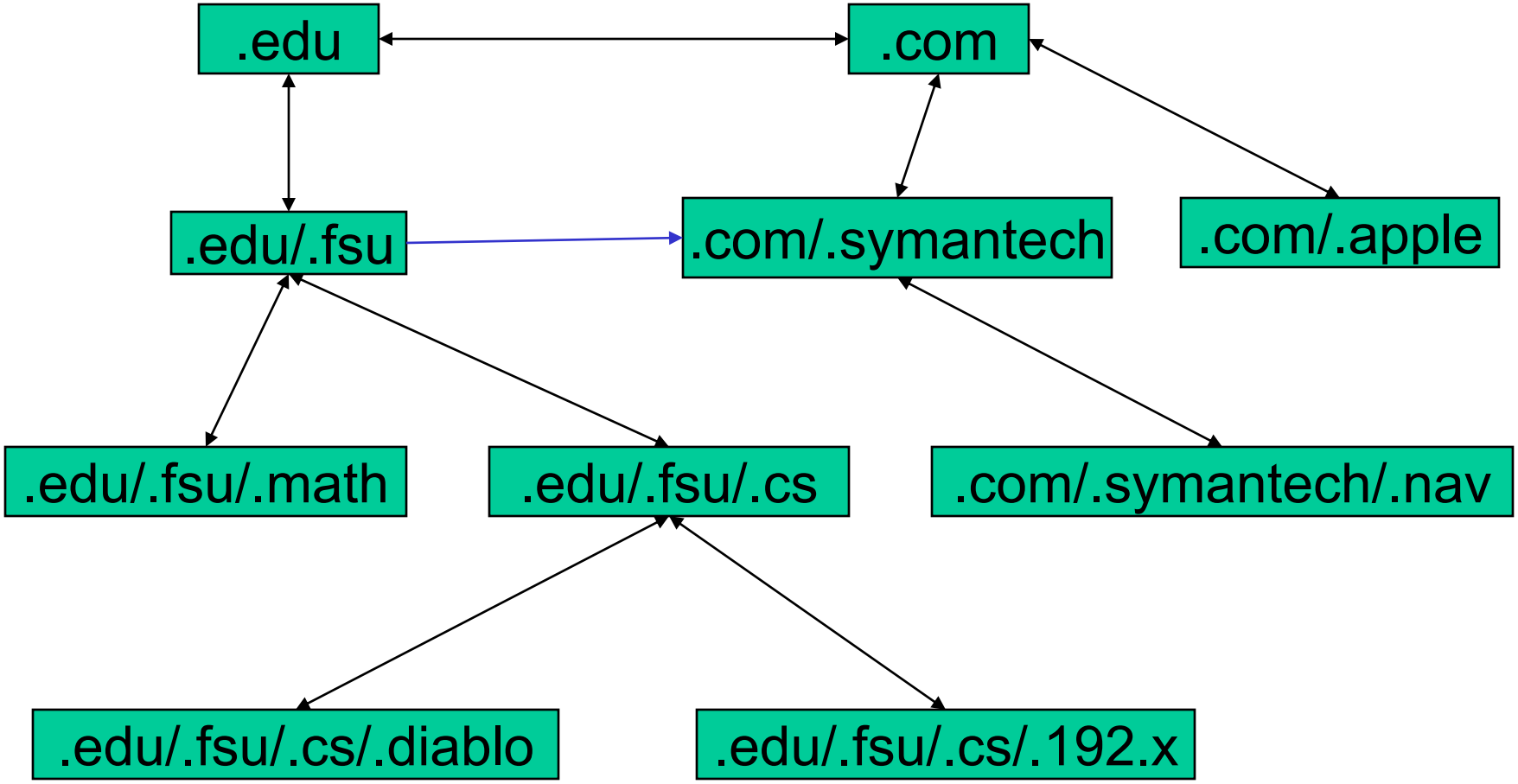
- Similar to oligarchic/ monopoly model, but delegation takes place with domain name constraints:



Bottom-Up Constrained Naming

- Each organization creates an independent PKI and then link to others:
  - Top-down links: Parent certifies child
  - Bottom-up links: Child attests parent
  - Cross-links: A node certifies another node
- To certify a node N:
  1. Start from your trust anchor: if it is also an ancestor to N, just verify the delegation chain
  2. If (1) fails, query your trust anchor for a cross-link to an ancestor of N
  3. Else repeat using the parent of your trust anchor.

Example


.edu
.com
.edu/.fsu
.com/.symantech
.com/.apple
.edu/.fsu/.math
.edu/.fsu/.cs
.com/.symantech/.nav
.edu/.fsu/.cs/.diablo
.edu/.fsu/.cs/.192.x

Advantages of Constrained Naming PKIs

- Simple and flexible
- Locally deployable
- Compartmentalized trust
- Easy to replace keys at local levels
  - Lightweight and fast revocation
- Non-monopolistic, open architecture
- PKIX/X.509 (oligarchic) has recognized the advantages of constrained naming, and support it through the NameConstraints field.

Relative Names

- Aliases, shorthand forms or non-global names that are locally understood:
  - Parent may refer to each child simply the part of the child's name that extends of its own name
  - Child refers to parent simply as "parent"
  - Think of how file systems work
  - Cross links can use global names (absolute paths) or relative names
- SPKI certificates support relative names

Certificate Revocation

- As the trusted parties multiply, so does the possibility of having to revoke trust
  - Private key of user compromised:
    - Revocation of user certificate
    - Publication of revoked certificates:
      - Certificate revocation lists, or CRLs.
  - Private key of trusted party compromised:
    - Update of CA's public key
    - Re-certification of existing certificates?
    - Timestamping?

Certificate Revocation

- CRLs:
  - Signed, time-stamped list of all revoked certificates
  - Cost to generate and verify a CRL is proportional to the number of all revoked certificates
- Δ CRLs:
  - Publish only changes from a latest full CRL
- OLRS (On-line Revocation Server)
- Affirmation of valid certificates

Other Issues

- Directories
  - A standardized mechanism for querying names is required for some PKIs (e.g. constrained names)
  - E.g.: DNS directory service
- Should a certification record be stored with the issuer or subject of the certification?
- Certificate chaining:
  - To certify Alice -- start with Alice's name and go up (forward building) or with our trust anchor and down (reverse building)?

X.509

- Certificate Management Protocol (CMP: RFC 2510)
- Online Certificate Status Protocol (OCSP: RFC 2560)
- Certificate Management Request Format (CRMF: RFC 2511)
- Time-Stamp Protocol (RFC 3161)
- Certificate Management Messages over CMS (RFC 2797)
- Internet X.509 Public Key Infrastructure Time Stamp Protocols (RFC 3161)
- Use of FTP and HTTP for transport of PKI operations (RFC 2585)

X.509

- PKIX Working Group (established 1995)
- Goal: develop Internet standards needed to support an X.509-based PKI:
  - RFC 2459, profiled X.509 version 3 certificates and version 2 CRLs for use in the Internet.
  - Profiles for the use of Attribute Certificates (RFC XXXX [pending])
  - LDAP v2 for certificate and CRL storage (RFC 2587)
  - X.509 Public Key Infrastructure Qualified Certificates Profile (RFC 3039)
  - Internet X.509 Public Key Infrastructure Certificate Policy and certification Practices Framework (RFC 2527 - Informational)

X.509

- The IETF chose to use X.500 naming standards for certificates
  - C=US, O=Sun, OU=Java, CN=java.sun.com
- Browsers know websites by DNS names, not X.500 names
  - Initial browser implementations did not check CN.
  - Today, DNS names are included either in CN or in SubjectAltName field
- Rationale: DNS does not support certificate lookup

X509 + PKIX Certificates

- Version
- SerialNumber
- Signature
- Issuer
- Validity
- Subject
- SubjectPublicKeyInfo
- IssuerUniqueIdentifier
- SubjectUniqueIdentifier

- AlgorithmIdentifier
- Encrypted
- Extensions
  - AuthorityKeyIdentifier
  - SubjectAltName
  - SubjectKeyIdentifier
  - KeyUsage
  - CertificatePolicies
  - PolicyMappings
  - NameConstraints
  - ...

X.509 Certificate

Version	Certificate Version (e.g. X.509_v3)
Serial Number	Unique Identifier for the Certificate
Signature	ID of the Algorithm Used to Sign the Certificate
Issuer	Unique Name of the Certificate Issuer
Validity	Time Period of Certificate Validity
Subject	Unique Name of the Certificate Owner
Subject Public Key Info	Public Key and Algorithm ID of the Owner
Issuer Unique ID	Optional Unique ID of the Certificate Issuer
Subject Unique ID	Optional Unique ID of the Certificate Owner
Extensions	Optional Extensions

Reading Assignment

- Chapter 17