Introduction to Cryptography CS 355

Lecture 33

Public Key Certificates

Lecture Outline

- The public key distribution problem
- Public-key certificates
- X.509
- PGP
- Certificate revocation

Public Keys and Trust

Public Key: P_A
Secret key: S_A

Public Key: P_B
Secret key: S_B

- **How are public keys stored?**
- **How to obtain the public key?**
- **How does Bob know or 'trusts' that P_A is Alice's public key?**

Distribution of Public Keys

- **Public announcement**: users distribute public keys to recipients or broadcast to community at large
- **Publicly available directory**: can obtain greater security by registering keys with a public directory
- Both approaches have problems, and are vulnerable to forgeries

Public-Key Certificates

- A certificate binds identity (or other information) to public key
- Contents signed by a trusted Public-Key or Certificate Authority (CA)
- Can be verified by anyone who knows the public-key authority's public-key
- Certificates allow key exchange without real-time access to public-key authority

X.509 Certificates

- Part of X.500 directory service standards.
- Defines framework for authentication services:
  - Defines that public keys stored as **certificates** in a public directory.
  - Certificates are **issued and signed** by an entity called **certification authority (CA).**
- Used by numerous applications: SSL, IPSec, SET
- Started 1988

X.509 Certificates

- Certificates contain:
  - version (1, 2, or 3)
  - serial number (unique within CA) identifying certificate
  - signature algorithm identifier
  - issuer X.500 name (CA)
  - period of validity (from - to dates)
  - subject X.500 name (name of owner)
  - subject public-key info (algorithm, parameters, key)
  - issuer unique identifier (v2+)
  - subject unique identifier (v2+)
  - extension fields (v3)
  - signature (of hash of all fields in certificate)

How to Obtain a Certificate?

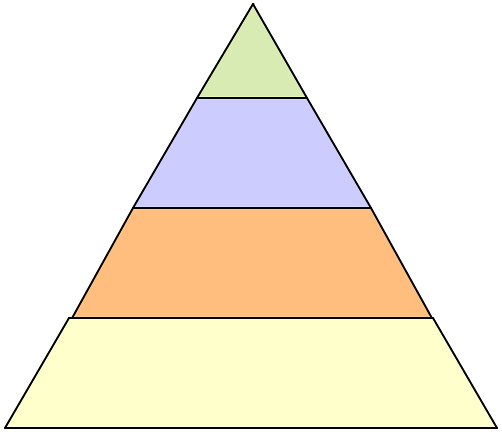
- For a particular application you can define your own CA (libraries like openssl provide the necessary tools)
  - many companies define their own CA.
- VeriSign: a company that provides certificates to many commercial companies;
- Private key remains secret and certificate must be accessible.
- Example: see certificates accepted by your browser

Validity of Certificates

- Certificates are valid if:
  - Signature of CA verifies
  - Dates of the certificate are valid
  - Certificate was not revoked
- Certificates can be revoked before expiration if
  - user's private key is compromised
  - user is no longer certified by this CA
  - CA's private key is compromised
- CA maintains a list of revoked certificates: Certificate Revocation List (CRL)
- Users should check certificates with CA’s CRL

CA Hierarchy

- If everybody has the same CA then they are assumed to know its public key, so they can verify each other’s certificate. Not scalable.
- Other approach: entities have different CAs; in this case CAs how is a certificate verified?
  - – CAs can form a hierarchy
  - – certificates linking members of hierarchy are used to validate other CAs
  - – each CA has certificates for clients (forward) and parent (backward)
  - – each client trusts parents certificates



CAs and Trust

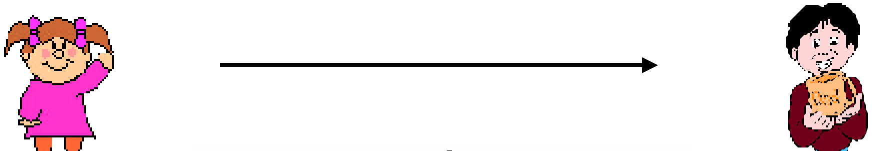
- Certificates are trusted if signature of CA verifies
- Chain of CA's can be formed, head CA is called root CA
- In order to verify the signature, the public key of the root CA should be obtain.
- TRUST is centralized (to root CA's) and hierarchical

PGP

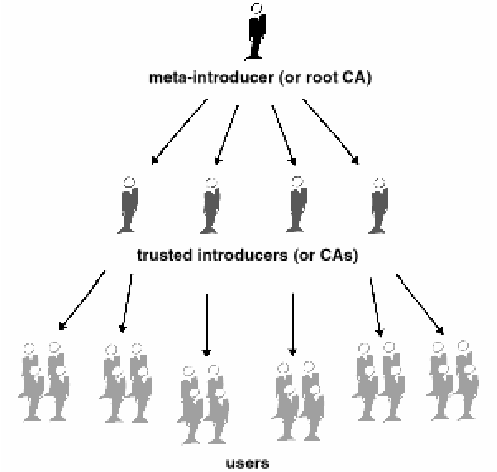
- PGP (Pretty Good Privacy) is a secure email application
- Mail is encrypted and signed using public keys
- What’s different? The way the keys are authenticated, trust about the keys is built.
- Trust is not centralized.
- **http://www.pgpi.org/**

Trust Models

- Direct Trust



- Hierarchical trust



- Web of trust: combination of both

PGP Web of Trust

- Any user can act as a CA
- Certificate is only valid if the receiving party recognize the validator as a trusted introducer
- Each user stores:
  - its own public/private keys
  - keys of entities that interacts with
  - whether or not the user considers a particular key to be valid
  - the level of trust the user places on the key that the key's owner can serve as certifier of others' keys

Certificate Revocation Approaches

- Certificate Revocation Lists (delta-CRL)
- Problems of CRL
  - Proof of validity is long (linear in the number of revoked certificates)
  - Tradeoff of security & communication cost
- OCSP (Online Certificate Status Protocol)
  - RFC2560
  - contact an online server about whether a certificate (or a list of certificates) is still valid
  - obtain a signed answer

Certificate Revocation System

- Certificate Revocation System (Micali, 1996)
  - use hash chain
  - E.g., if a certificate is valid for one year, $h^{365}(s)$ is included in the certificate
  - On day d, certificate holder obtains $h^{365-d}(s)$ from the certificate issuer, which is used to prove that the cert is still valid on day d
  - Communication cost even when a cert is not revoked.

Certificate Revocation Tree

- Certificate Revocation Tree (Kocher, 1998)
  - the serial numbers of unrevoked certificates are stored as leafs
  - a Merkle hash tree is constructed from the leaves
  - the CA signs the root of the tree
  - a proof that a given serial number is in the leaves has size O(log N)
  - the tree changes each time a new certificate is revoked

Coming Attractions …

- Key Agreement Protocols