Certificates and Public Key Infrastructure (PKI)

Abstract

Basic concepts of certificates and Public Key Infrastructure (PKI) are presented. The chain of trust of certificates and Certification Authorities is explained. The model of X.509 certificates is presented.

Outline

- Public key cryptography and Public Key Infrastructure (PKI)
- Certificates and chain of trust
- X.509 certificates

Public Key Cryptography

- The encryption key is public, but the decryption key is known only by the intended recipient
  - everyone knows what has to be done to find the decryption key, based on public data
  - security is based on the fact that finding the decryption key from public data is theoretically possible, but computationally unfeasible
- How can I trust that a public key is really posted by the legitimate entity?

- Public Key Infrastructure (PKI)

Public Key Infrastructure (PKI)

- Set of policies and procedures
  - to generate and publish keys
  - to generate, publish, validate and revoke digital certificates
  - to manage public-key encryption
- PKIs define certification and validation operations
  - a *certificate* binds a public key to the respective entity (user or piece of information)
  - *validation* guarantees that a certificate is valid
  - a certificate is signed by its publisher (*Certification Authority, CA*)
  - two types of certificates
    - *identity certificates* provide and guarantee some entity's identity information (e.g., address, public keys)
    - *credential certificates* provide and guarantee information describing access rights to a resource

Trusting a Certificate and Public Keys

- A Certificate of Alice published by a CA
  - provides and guarantees the public key K_A of Alice
  - is signed by the Certification Authority (or *Trusted Authority*, TA) using its private key K^{-1}_{TA}

    $$C_A = \{ A, K_A, \{h(A, K_A)\}_{K^{-1}TA} \}$$
  - can be validated by anyone using the public key K_{TA} of TA
- Who trusts what?
  - if I trust CA ➔ I trust the information in the certificate (K_A)
    ➔ I do not necessarily trust A
  - I have to trust K_{CA} to trust the certificate
  - a CA can certify a lower-level CA (also *Registration Authority*, RA) building a *chain of trust* (multiple-level hierarchy)
  - a first-level CA signs its own certificate with its private key, while its public key is supposed trustable (provided with Internet browsers)

X.509 Certificates

- The ITU-T X.500 series of Recommendations defines a directory service
  - directory: database of user information, incl. public keys
  - v1: 1988; v7: 2012
- ITU-T Rec. X.509 define authentication services by the X-500 directory to its users
- X.509 is based on public-key cryptography and digital signatures
  - does not dictate specific digital signature and hash algorithms
- The X.509 certificate format is used in S/MIME, IPsec, SSL/TLS

Basic Information in an X.509 Certificate

- *Version*
- *Serial number* of the certificate (unique for each CA)
- *Issuer name* (the CA who created and signed the certificate)
- *Period of validity* (from/to)
- *Subject name* (the certificate certifies the public key of the subject who holds the corresponding private key)
- *Public Key of the Subjec*t
- *Signature* (a field specifies the signature algorithm)

Revocation of Certificates

- Each certificate includes a period of validity
- In addition, a certificate could be *revoked* before it expires, because
  - the user's private key has been compromised
  - the user is no longer certified by that CA
  - the CA's certificate has been compromised
- Each CA must maintain a list of all *revoked but not expired* certificates, which it previously issued
  - the list (signed by the CA) is posted in the directory
  - certificates are identified by their SNs
  - users should maintain a local cache to avoid checking the revocation list each time a certificate is received

Example of X.509 Certificate

- Subject's Public Key (RSA):
  - control bits
  - n ($p \cdot q$, 1024 bits)
  - $e = 2^{16}+1$