Authentication Applications

CSS 322 – Security and Cryptography

Authentication Applications

- Authentication:
  - Verify the identify of someone (or something)

- Human to computer authentication:
  - **Passwords**
- Common (but old) network computer authentication using symmetric key cryptography:
  - Kerberos
- Certificate based authentication (public key cryptography):
  - **X.509**, Public Key Infrastructure (PKI)

Authenticating People

- Computer to computer authentication:
  - Computers can remember high-quality cryptographic keys and perform cryptographic operations
- Human to computer (or human) authentication:
  - Humans cannot store large keys; Humans cannot accurately or efficiently perform cryptographic operations
- So we must have special methods for authenticating people

"Humans are also large, expensive to maintain, difficult to manage and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed. But they are sufficiently pervasive that we must design our protocols around their limitations."

- Kaufman, Perlman, Speciner "Network Security: Private Communication in a Public World", Prentice Hall 2002.

Methods for Authenticating People

- Three main techniques based on:
  - What You Know
    - Passwords
  - What You Have
    - Physical keys, ATM cards
  - What You Are
    - Voice recognition, fingerprint scanners
- We will look at passwords
  - (Biometrics may be covered at end of course of time permits)

Problems with Passwords

- We all use passwords everyday, but many problems with security of passwords:
  - Attacker may see the password when it is used
  - Attacker may read the file where computer stores password
  - Your password might be easy to guess at your computer
  - Your password might be crackable off-line (brute force)
  - Your password might be secure, but the system may become too inconvenient to use (e.g. very long password)
- Using passwords for authentication is a tradeoff between:
  - Security
  - Usability/convenience

Online Password Guessing

- Try to guess the password while computer in use
  - Attacker has only a limited time
  - Guesses can be recorded/tracked
- Security depends on:
  - Number of incorrect guesses allowed and consequence of too many incorrect guesses
    - Military example: if you guess incorrectly 1 time, you may be shot!
    - Bank example: if you guess incorrectly 5 times, lose the ATM card
- Approaches to make system more secure
  - Lock system (e.g. account) if too many guesses
    - Subject to Denial of Service attack
  - Limit the speed that guesses can be made
    - E.g. Only one attempt per second
  - Try to find the attacker
    - Record the attempts made and report to administrator or users

Online Password Guessing

- Make passwords hard to guess:
  - Force users to use certain types of passwords
    - Random characters often too hard to remember; users just write the password on paper (which is insecure)
    - Generate a pseudo-random pronounceable string
      - Easier to remember, slightly less secure than random characters
    - Let user choose own password, but give warnings about good and bad passwords:
      - Bad passwords include: dictionary words, first and last names, dates, and simple combinations
      - Good passwords include: phrases with misspellings or non-alphabetic characters ($%@+); first letters of a long phrase

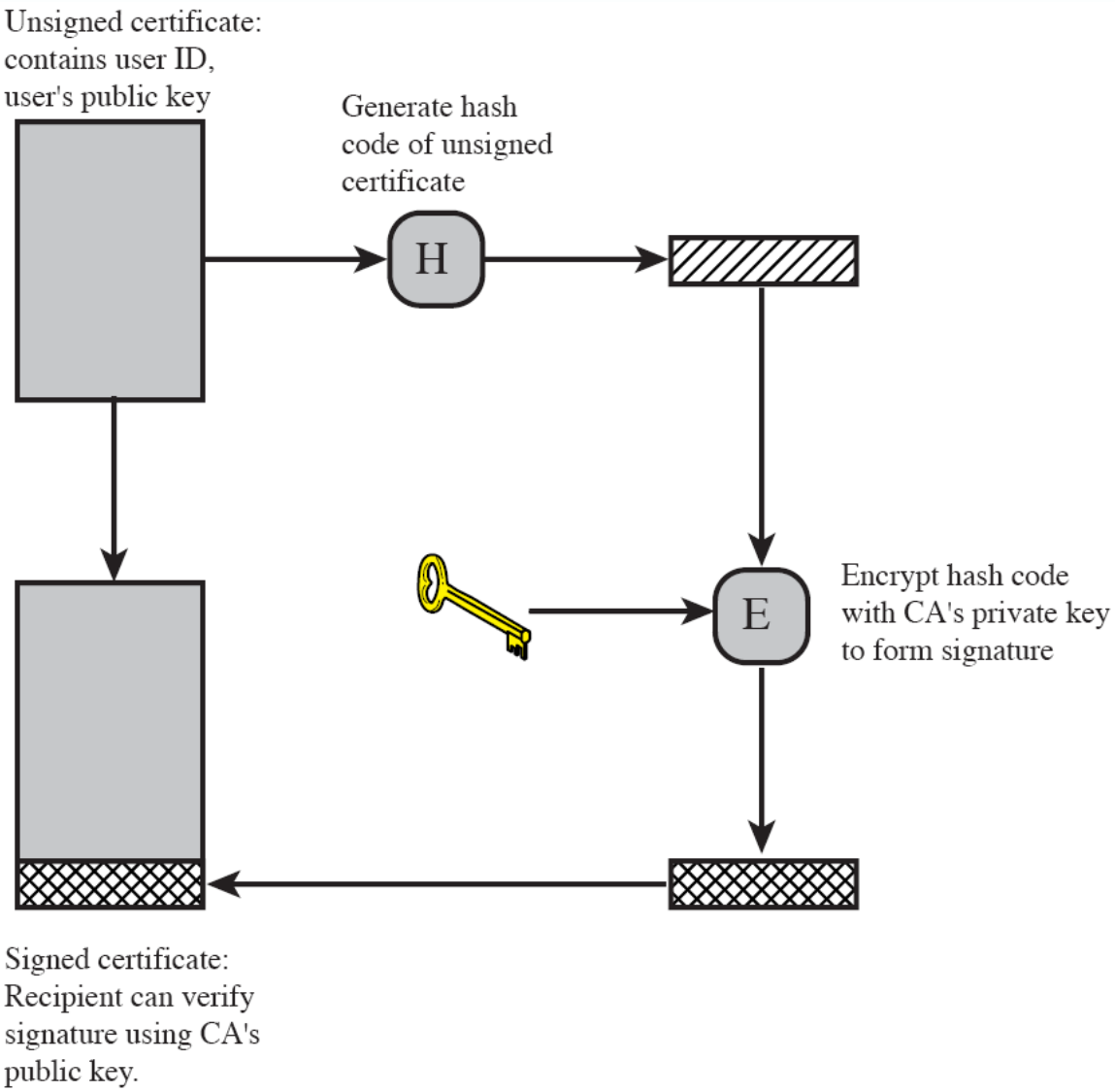
Offline Password Guessing

- Try to guess the password outside of normal operation of system
  - No practical restrictions on time
- Passwords must be known by users AND stored on computer system
  - Offline attacks often try to read the passwords stored on computer system (e.g. password file in Unix)
    - Store the cryptographic hash of password (not the actual password)
    - Hide the password file (from as many users as possible)
  - Once password file is readable, relatively easy to perform dictionary attack to find one password
- Most easy to use passwords will not be as secure as a 64-bit random number – therefore offline password guessing is always possible

X.509

- ITU-T Recommendation as part of X.500 series
  - X.500 defines a directory service
    - Server (or set of servers) that maintain database about users
    - X.509 allows directory to store public key certificates
- Uses public key certificates and digital signatures
  - Recommends RSA to be used, but other algorithms possible
- Requires a trusted Certificate Authority (or a set of CAs)
- Used by other systems:
  - S/MIME (secure email)
  - IP security (network layer security)
  - SSL/TLS (transport layer security)
  - SET (e-commerce)

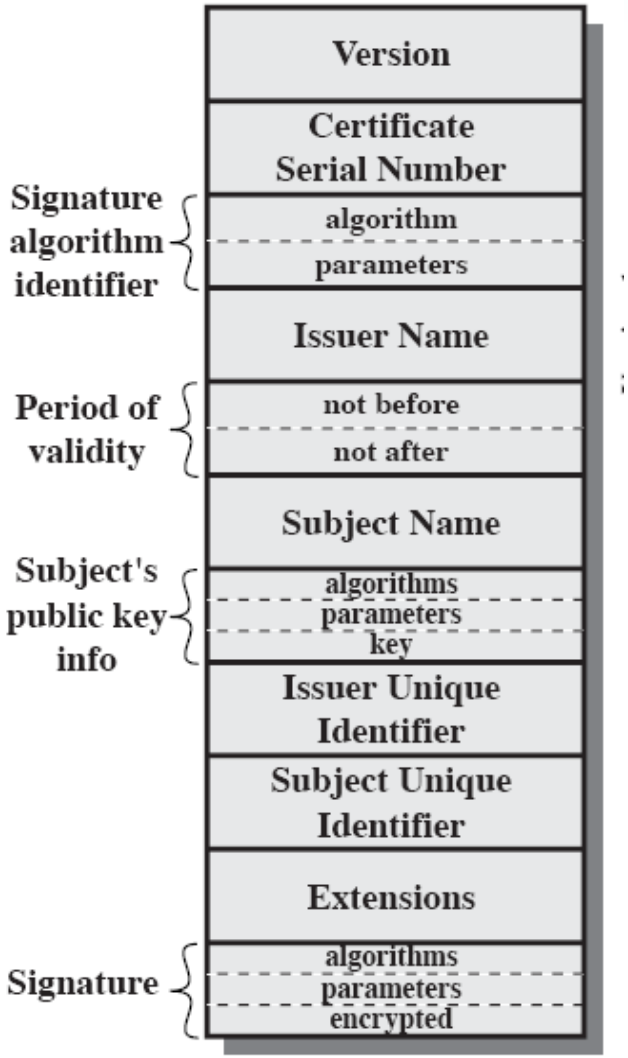
Digital Certificates Refresher



X.509 Certificates

- X.509 defines a format for the certificates
- Each user has a certificate, although it is created by the Certificate Authority (CA)
- Certificates are stored in a public directory
- Certificate format includes:
  - Version of X.509 certificate
  - Signature algorithm
  - CA's name and unique identifier
  - Period of validity
  - User's name and unique identifier
  - User's public key information
  - Signature

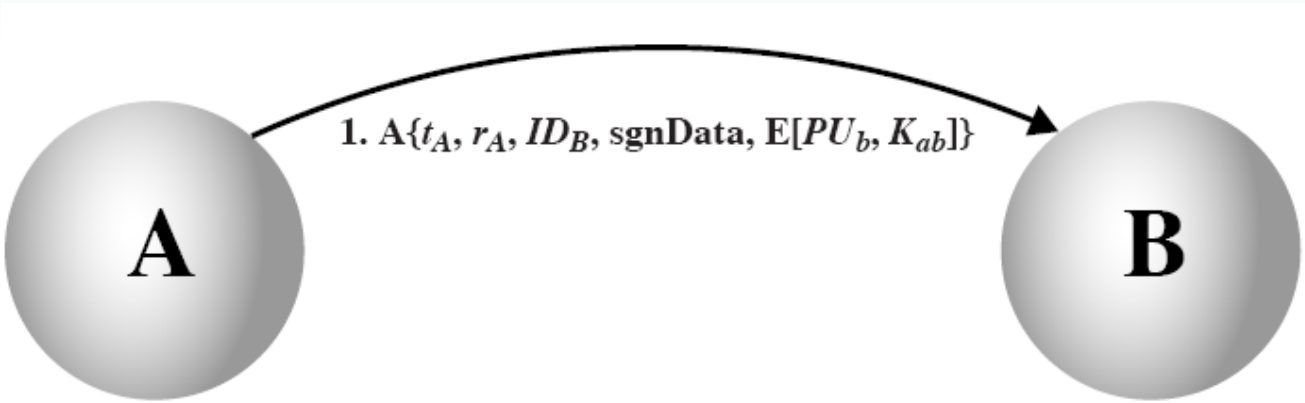
X.509 Certificates



Obtaining a User's Certificate

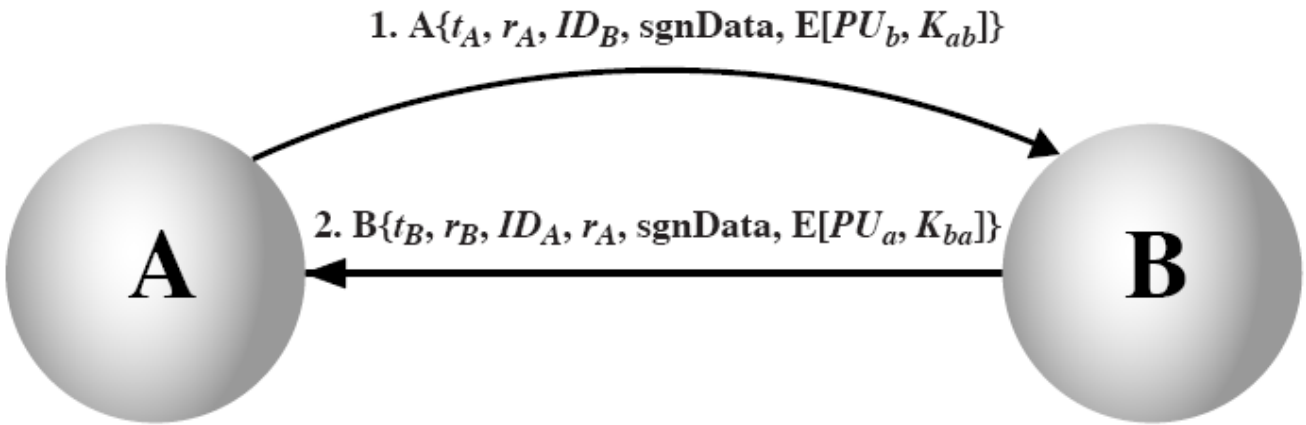
- All users use the same CA
  - Either place all certificates in a directory or send direct to other users
  - Every user must have CA's public key (distributed securely)
  - Signature of CA ensures certificates are real
- Multiple CAs
  - May not be practical for all users to use same CA (since CA's public key must be distributed securely to all users)
  - Can establish a hierarchy of CA's
  - If A's certificate is signed by X and B's certificate is signed by Y, how do A and B trust each other's certificates?
    - A gets Y's public key from X (signed by X)
    - B gets X's public key from Y (signed by Y)
    - Assumes that all CA's have securely exchanged public keys with each other
  - Can be extended to any number of CA's (e.g. a chain)

One-way Authentication with X.509



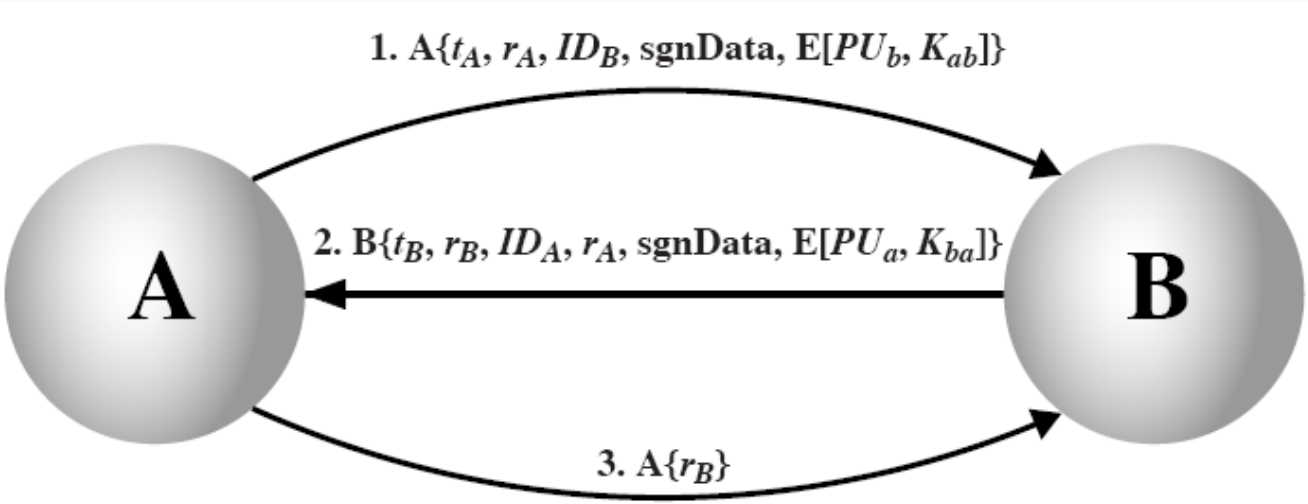
- Validates the following:
  1. Identity of A and that the message was generated by A
  2. Message as intended for B
  3. Message has not been modified and is original
- Parameters:
  - *A{ }* means signed with A's private key
  - t_A, is timestamp (prevents delayed delivery of messages)
  - r_A is nonce value (detect replay attacks)
  - *sgnData* – data to be sent to B (signed – authentic and original)
  - K_{AB} – can also securely send a session key

Two-way Authentication with X.509



- Also validates (in addition to one-way):
  - Identity of B and reply message was sent by B
  - Message was intended for A
  - Message has not been modified and is original

Three-way Authentication with X.509



- Third message includes a nonce
  - This means timestamps do not need to be checked
  - Useful when synchronised clocks are not available