Authentication and Key Distribution

Dawn Song
dawnsong@cs.berkeley.edu

Review
- Hash functions
  - Different cryptographic properties
- MAC functions
- Digital signatures

Obtaining Public Key
- Public-key encryption and digital signature both require knowing the mapping: (name, pub_key)
  - Why?
- How do we obtain this mapping securely?
Public-key Infrastructure

- One approach – the big directory (white pages)
  - Need to make secure big directory
  - Need to keep it updated

- Better approach: allow one party to attest to another
  - Public key infrastructure (PKI)
  - Public key certificate (PKC)
  - Certificate authority (CA)

- Check the root CAs and certificates in browser

PKI Terminology

- PKI: Public Key Infrastructure
- CA: Certificate Authority (similar to TTP (Trusted Third Party) in symmetric-key protocols)
- A public-key certificate (or simply "certificate") binds a name to a public key
- Certificate repository: stores certificates
- Trust anchor: certificates of public keys that are trusted to sign other certificates

Sample Certificate

Certificate

Version: v3 (0x2)
Serial Number: 3 (0x3)
Signature Algorithm: PKCS #1 MD5 With RSA Encryption
Issuer: OU=Ace Certificate Authority, O=Ace Industry, C=US
Validity:
  Not After: Sun Oct 17 18:36:25 1999
Subject: CN=Jane Doe, OU=Finance, O=Ace Industry, C=US
Subject Public Key Info:
  Algorithm: PKCS #1 RSA Encryption
  Public Key:
    Modulus:
      43:7d:45:6d:71:4e:17:3d:f0:36:4b:5b:7f:a8:51:a3:a1:00:
      91:f4:15
    Public Exponent: 65537 (0x10001)
Extensions:
  Identifier: Certificate Type
    Critical: no
    Certified Usage:
      SSL Client
  Identifier: Authority Key Identifier
    Critical: no
    Key Identifier:
      26:c9
Signature:
  Algorithm: PKCS #1 MD5 With RSA Encryption
  Signature:
Today's PKI "Hierarchy"

- **Verisign**: KV
- **CNN**: \{CNN, K_{CNN}\}^{KV}\textsuperscript{-1}
- **Yahoo**: \{Yahoo, K\}^{KV}\textsuperscript{-1}
- **EBay**: \{EBay, K_{EB}\}^{KV}\textsuperscript{-1}
- **USPS**: K_{U}\textsuperscript{-1}
- **Carol**: \{C, K_{C}\}^{K_{U}}\textsuperscript{-1}
- **Dave**: \{D, K_{D}\}^{K_{U}}\textsuperscript{-1}

PKI Models (continued)

- **Anarchy model**
  - PGP’s web of trust
  - Proposed by Phil Zimmermann in 1992

Authentication and Key Establishment Protocols

- **Client C and Server S want to securely communicate with each other**
  - Each knows the other’s public key
  - How?
- **Public-key encryption is much more expensive than symmetric-key encryption**
  - Establish session key: shared secret for the session
  - How?
Example: Needham-Schroeder Protocol

\[ \{N_C, C\}_{KS} \]
\[ \{N_C, N_S\}_{KC} \]
\[ \{N_S\}_{KS} \]

- \( KS, KC \) are public keys of \( S \) and \( C \) respectively
- Goal:
  - Mutual authentication: \( C \to S, S \to C \)
  - Shared secret: \( N_C, N_S \)

What May Go Wrong?

- Desired security property
  - Confidentiality
  - Integrity
  - Authenticity

Protocol Analysis

- Analyze high level security properties
  - Secrecy
  - Authentication
  - Atomicity
  - Non-repudiation
- Assume cryptographic primitives secure
  - Signature: secure against existential forgery
  - Public key/Private key encryption: secure against adaptive chosen-ciphertext attack
- Security protocols are notoriously hard to get right
**Active Attacker**

- An active attacker may
  - Eavesdrop on previous protocol runs, even on protocol runs by other principals, replay messages at a later time
  - Inject messages into the network, e.g., fabricated from pieces of previous messages
  - Alter or delete a principal’s messages
  - Initiate multiple parallel protocol sessions
  - Run dictionary attack on passwords
  - Run exhaustive attack on low-entropy nonce

**Intruder Model**

Intruder can
- Intercept, drop, generate messages, full control of network
- Collude with malicious parties

**Flaw in Needham-Schroeder**

Flaw (discovered 18 years after publication):
- Authentication: C→E, S→C
- Secrecy: E knows N_c, N_s
- How to fix it?
  - The second message should be \{S, N_c, N_s\}_KC
SSL / TLS

• Goal: Perform secure e-commerce across Internet
  – Secure bank transactions
  – Secure online purchases
  – Secure web login (e.g., Blackboard)

• Security requirements
  – Secrecy to prevent eavesdroppers to learn sensitive information
  – Entity and message authentication to prevent message alteration / injection

Position of Security in Protocol Stack

<table>
<thead>
<tr>
<th>Physical Layer</th>
<th>Data Link Layer</th>
<th>Network Layer</th>
<th>Transport Layer</th>
<th>Application Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet</td>
<td>802.3 MAC</td>
<td>IP</td>
<td>TCP, UDP</td>
<td>DNS, HTTP, SMTP</td>
</tr>
</tbody>
</table>

SSL History

• SSL: Secure Sockets Layer protocol
• SSL v1: Designed by Netscape, never deployed
• SSL v2: Deployed in Netscape Navigator 1.1 in 1995
• SSL v3: Substantial overhaul, fixing security flaws, publicly reviewed
• TLS: Transport Layer Security protocol
• TLS v1: IETF standard improving on v3
5-min Break

- Wait list
- In-class final, Dec 10

Discrete Logarithm Problem

- Public values: large prime p, generator g
- \( g^a \mod p = x \)
- Discrete logarithm problem: given x, g, and p, find a
- Table g=2, p=11

<table>
<thead>
<tr>
<th>a</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>( g^a )</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>3</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

- Number field sieve is fastest algorithm known today to solve discrete logarithm problem
  - Running time: \( O((1.923+o(1))(\ln(p))^{1/3}(\ln(\ln(p)))^{2/3}) \)

CDH and DDH

- Computational Diffie Hellman (CDH) Assumption
  - Given large prime p, generator g, \( x=g^a \mod p \), \( y=g^b \mod p \)
  - it is difficult to compute \( g^{ab} \mod p \).
- Decisional Diffie Hellman (DDH) Assumption
  - Given large prime p, generator g, \( x=g^a \mod p \), \( y=g^b \mod p \), \( z=g^t \mod p \)
  - it is difficult to determine whether \( z = g^{ab} \mod p \).
Diffie-Hellman Key Agreement

- Public values: large prime $p$, generator $g$
- Alice picks secret random value $a$
- Bob picks secret random value $b$
- Protocol: generate shared key $g^{ab}$

Diagram:
- Alice
  - $g^a$
- Bob
  - $g^b$