Security

- Examples of Attack
- Principles
- Threats
- Cryptography
- Security Systems
Attacks

- Buffer Overflow
- Denial of Service Attack
- Email virus
- ARP attack
Buffer Overflow

- Basic Mechanism:
  Attacker overwrites program stack to force execution of her code

- Examples:
  - Virus
  - Corrupt files
Buffer Overflow - Illustration

Request overflows program counter → Forces execution of “code”
Buffer Overflow - Remedies

- Protect memory by preventing overwrite of stack [either through OS or through language]

- Check validity of request
Denial of Service – SYN Flood

- **Basic Mechanism**
  - Flood a host with a rapid sequence of SYNs

- **Effect**
  - Host sets aside some space to store state of new TCP connection
  - If rapid sequence, then host runs out of space and crashes

- **Remedies**
  - Check for “valid” SYNs, i.e., SYNs followed by requests; discard invalid SYNs to clear memory
  - Use smart firewall that forwards only valid SYNs to hosts
  - Store “state” in cookie that comes back with request
Denial of Service – DDOS

Distributed Denial of Service Attack

- **Basic Mechanism**
  - Saturate a link to a host by sending requests from many nodes across the Internet

- **Effect**
  - Host is incapacitated

- **Remedies**
  - Verify that source IP exists (i.e., is not spoofed)
  - Block packets that DDOS tools use (some ICMPs)
  - Limit rate of ICMP flows
  - Limit rate of SYNs
  - Trace back from last router upstream to block packets toward that link
Email

- Basic Mechanism
  - Attachment that contains virus

- Effect
  - Some email programs execute code in virus without authorization

- Remedies
  - Firewall to check attachments and remove specific ones
  - Avoid automatic execution of attachments
ARP

- Basic Mechanism
  - Intruder replies to ARP request and performs denial of service on host
    A → [ARP: Who is IP B]
    C → [ARP: I am IP B]; DoS B

- Effect
  - C impersonates B for A

- Remedies
  - Check source of ARP
  - Avoid DoS
Principles

- You would somehow like to have your data (or that of others) be secure. This often means you want to:
  - know who really sent it
  - know nobody else read it

- More specifically, protect from:
  - eavesdropping, masquerading, replay, traffic analysis, exploit-based attacks, denial-of-service
Principles

These attacks are often classified as

- Active:
  - somebody actually generates or modifies network traffic
  - easier to detect, harder to prevent

- Passive:
  - somebody just collects and analyses network traffic
  - harder to detect, easier to prevent
# Threats

<table>
<thead>
<tr>
<th>Against</th>
<th>Type</th>
<th>Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computers</td>
<td>Physical Infection</td>
<td>Physical security (lock)</td>
</tr>
<tr>
<td></td>
<td>Intrusion</td>
<td>Virus detection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Firewall, passport control</td>
</tr>
<tr>
<td>Users</td>
<td>Identity Privacy</td>
<td>Signature, password, watermark</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Encryption, relay</td>
</tr>
<tr>
<td>Documents</td>
<td>Integrity</td>
<td>Message authentication code</td>
</tr>
<tr>
<td></td>
<td>Confidentiality</td>
<td>Encryption</td>
</tr>
</tbody>
</table>
Cryptography

- Basic Mechanism
- Main Issues
- Secret Key
- Public Key
- Hashing
Basic Mechanism

\[ P \rightarrow [E(.)] \rightarrow C \rightarrow [D(.)] \rightarrow P \]

- **Plaintext**: Original message
- **Encryption function**: Converts plaintext to ciphertext
- **Cyphertext**: Sent (Encrypted Text)
- **Decryption function**: Converts ciphertext back to plaintext

**Flowchart:**
- **P**: Plaintext
- **[E(.)]**: Encryption function
- **C**: Cyphertext
- **[D(.)]**: Decryption function
- **P**: Plaintext (output of decryption)

**Diagram Title:** Basic Mechanism

**Diagram Elements:**
- Encryption function
- Decryption function
- Plaintext
- Cyphertext
- Flowchart representation of the encryption and decryption process.
Basic Mechanism

- Two flavors:
  - **Secret Key**: $E(\cdot)$ and $D(\cdot)$ are known only to Bob and Alice
  - **Public Key**: Alice advertises $E(\cdot)$ that should be used to encrypt messages to her

\[
P \rightarrow [E(\cdot)] \rightarrow C \rightarrow [D(\cdot)] \rightarrow P
\]

Bob \quad Channel \quad Alice
Basic Mechanism

P -> [ E(.) ] -> C -> [D(.)] -> P

D(.) is the inverse of E(.)

E(.) should be a “one-way function”
- Easy to compute
- Hard to invert:
  - Looking at E(P) it should be hard to get P
  - Looking at many E(P), it should be hard to figure out the function D(.), or
  - Knowing many pairs (P, E(P)), it should be hard to figure out the function D(.)
Main Issues

- For the cryptographer, the main issues:
  - choice of the transformation (D and E)
    - is the underlying mathematical basis efficient for decoding and encoding with keys and hard without them?
  - do you publish the algorithm or not?
  - generation and distribution of keys
    - might like to use random numbers, but computers aren’t exactly random devices
    - how do you get a secret from one person to another if you don’t already have keys!?
Main Issues

- For the cryptanalyst, the main issues:
  - what is already known?
    - algorithm, plaintext-ciphertext pairs, any information about generation of the keys
  - types of attacks
    - ciphertext only (freq analysis, brute force)
    - known plaintext
    - chosen plaintext
Secret Key

\[ E(.) = E(.; K), \ D(.) = D(.; K) \]

where K is a shared secret: Key distribution
EXAMPLE 1: One-Time PAD:

\[ C = P + K \]

(addition bit-by-bit modulo 2, no carry)

\[ K = \text{random string of bits (50\% = 0, 50\% = 1)} \]

If used only once, this is a perfect code! (C is perfectly random and contains no information about P.)

Application: Top Secret transmissions. K is stored in a CD-ROM that is delivered securely ahead of time.
EXAMPLE 2: Data Encryption Standard - DES

Algorithm is known, but the key is secret

K = 40 bits => Weak
   56 bits => Marginal
   128 bits => Safe
Secret Key

- Note: DES Modes of Encryption
Secret Key

Note: DES Modes of Encryption

\[ E \]

\[ D \]

64 bits

C_{n-1}

64 bits

C_n

P_n

Z_{n-1}

C_n-1

P_n

Z_{n-1}
Secret Key

Note: DES Modes of Encryption
Public Key

- Bob: use E(.; Bob) to talk to me.
- Only Bob knows D(.; Bob).
- Trapdoor one-way function.

Example: Rivest-Shamir-Adleman:

$p, q$ prime; $n = pq$; $z = (p-1)(q-1)$ and $e$ coprime;
$d$ s.t. $ed = 1 \mod z$.

If $P$ in $\{0, 1, \ldots, n-1\}$ and $C = P^e \mod n$,
then $C^d \mod n = P$.
(e, $n$) public. (d, n) private.
Finding $d$ from (e, n) is believed to be hard.
Hashing

- \( H(P) \) short (e.g., 160 bits).
- Hard to find \( P \) and \( P' \) s.t. \( H(P) = H(P') \).
Security Systems

- Integrity
- Key Management
- Identification
Alice sends $P^*H(P)$ where $H(P)$ is protected by

- a. Authentic channel.
- b. Message Authentication Code:
  - Note that $E(H(P); K)$ with $K$ secret may not be secure. For instance, $Z = H(P) + R$ is not secure since then Eve can compute $H(P)$ from $P$ and $R$ from $Z + H(P)$ and then Eve can send $P'$ and $H(P') + R$.
  - Secure: $H(K_2^*H(K_1^*P))$ where $K_1$ and $K_2$ are secret to Alice and Bob, since Eve cannot compute $H(K_1^*P')$. 

Integrity

Alice sends $P^{*}H(P)$ where $H(P)$ is protected by
  
  c. Digital Signature.

Alice sends $C = D(P; Alice)$ and Bob recovers $P = E(C; Alice)$.

However, if Eve constructs $C'$ and computes $P' = E(C'; Alice)$, Bob will think that Alice sent $D(P'; Alice)$.

Instead, Alice should send $D(P^{*}H(P); Alice)$ because it is unlikely that Eve can find some $C'$ so that $E(C'; Alice)$ has the form $P'^{*}H(P')$ for some $P'$. 
Key Management

To share a secret $K$:

- **a. Hand-delivery**

- **b. Encrypt and distribute $K$** using some other secret key (e.g., Kerberos)
  - Shared Key with Kerberos
  - Get Login Key
  - Get Session Key.

- **c. Use a public key to distribute secret key $K$** (e.g., PGP)
d. Public key agreement: Diffie-Hellman:

Alice and Bob agree on public \((z, p)\).
Alice chooses \(a\) and Bob chooses \(b\).
Alice computes \(A = z^a \mod p\) and sends it to Bob.
Bob computes \(A^b \mod p\).
Bob computes \(B = z^b \mod p\) and sends it to Alice.
Alice computes \(B^a \mod p\).
One can show that

\[
A^b \mod p = B^a \mod p = z^{(ab)} \mod p =: K.
\]

Indeed: \(A = z^a + mp\) so that \(A^b = (z^a + mp)^b = \ldots = z^{(ab)} \mod p.\)
However, D-H is not robust to a "person-in-the-middle" attack.
   ❑ Indeed: Imagine Eve gets in the middle and plays the role of Alice.

Solution: Signing the exchange:
   ❑ Alice sends A to Bob and Bob sends B to Alice
   ❑ Alice signs (A, B) and sends it to Bob
   ❑ Bob signs (A, B) and sends it to Alice
   ❑ Eve cannot fake these signatures.
Identification

Bob wants to ascertain the identity of Alice.

- **a. Passwords:** Alice has a secret password K and sends (Alice, K). Bob maintains H(K) to verify Alice. However: can be intercepted.

- **b. Challenge/Response:** Bob sends string X to Alice who computes f(X, K) where K is a secret that Alice and Bob share. However, Bob must know K.

- **c. Public Key:** Bob chooses X and sends E(X; Alice) to Alice who computes X and sends it back to Bob.

- **d. Digital signature:** Bob sends X to Alice who signs it and returns it to Bob.
Secure Shell (SSH)

- Provides remote login and file transfer service
- Consists of 3 protocols:
  - SSH-TRANS creates TCP based secure channel using RSA to establish a session key
  - SSH-AUTH is used for authenticating client
  - SSH-CONN is used to run other insecure applications over a SSH tunnel using port forwarding
Secure Socket Layer (SSL)

- Provides Transport Layer Security (TLS)
- Creates a layer between Application and TCP layers
- When HTTP makes use of this, it’s called HTTPS
- Includes 2 parts
  - Handshake protocol to negotiate various communication parameters (cryptographic parameters, certificates, session key, integrity and compression schemes, etc.)
  - Record protocol for actual data transfer
IPsec

- Provides security at network layer
- It can be used in 2 modes
  - Transport mode
    - IPsec header is inserted after the IP header
    - Authentication Header (AH) provides integrity but no encryption
    - Authentication data includes some fields of IP header that don’t change
  - Tunnel mode
    - Entire IP packet is encapsulated in a new IP packet
    - Especially useful when tunnel ends at a point other than final destination
    - Encapsulating Security Payload (ESP) header provides almost all the features of AH, and also provided encryption