Principles of Network Security

Bob and Alice want to communicate "securely".
Trudy (the adversary) has access to the channel.

Who might Bob and Alice be?
- ... well, real-life Bobs and Alices!
- Web browser/server for electronic transactions (e.g., on-line purchases)
- on-line banking client/server
- DNS servers
- routers exchanging routing table updates

What can an adversary do?
- eavesdrop: intercept messages
- actively insert messages into connection
- impersonation: can fake (spoof) source address in packet (or any field in packet)
- hijacking: "take over" ongoing connection by removing sender or receiver, inserting himself in place
- denial of service: prevent service from being used by others (e.g., by overloading resources)

What is Network Security?
Confidentiality: only sender, intended receiver should "understand" message contents
- sender encrypts message
- receiver decrypts message
Authentication: sender, receiver want to confirm identity of each other
Message Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards)
Access and Availability: services must be accessible and available to (and only to) legitimate users.

Outline
- What is network security?
- Principles of cryptography
- Authentication
- Integrity
The Language of Cryptography

- symmetric key crypto: encryption and decryption keys are identical. (both are secret)
- public key crypto: encryption key is public, decryption key is secret.

First goal of cryptography: confidentiality.

Monoalphabetic Cipher

Plaintext: bob. i love you. alice
Ciphertext: nkn. s gktc wky. mgsbc

Q1: How hard to break this simple cipher?
   - brute force?
   - other?

Q2: How to make it more difficult to break?

Symmetric Key Cryptography: DES

- DES: Data Encryption Standard
  - US encryption standard (NIST 1993)
  - 56-bit symmetric key, 64-bit plaintext input
  - encryption: initial permutation ⇒ 16 "rounds", each using different 48 bits of key ⇒ final permutation
  - decryption: reversion operation using the same key
- How secure is DES?
  - DES Challenge (1997): 56-bit-key-encrypted phrase decrypted (brute force) in 4 months
- Making DES more secure:
  - use three keys sequentially (3-DES)
  - use more bits

AES: Advanced Encryption Standard

- new (Nov. 2001) symmetric-key NIST standard, replacing DES
- processes data in 128 bit blocks
- 128, 192, or 256 bit keys
- brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

Public Key Cryptography

- symmetric key crypto
  - requires sender, receiver know shared secret key
  - Q: how to agree on key in first place? (particularly difficult if Trudy is eavesdropping on all communication)
- public key crypto
  - encryption key is different from decryption key
  - encryption key is public, known to everyone, also called public key
  - decryption key is secret, known only to receiver, also called private key

One should not be able to derive the private key from the public key.
Public Key Cryptography: RSA

- Choosing keys:
  - Choose two large prime numbers p, q (e.g., 1024 bits each)
  - Compute n = pq, z = (p-1)(q-1)
  - Choose e (with e<n) that has no common factors with z. (e, z are "relatively prime")
  - Choose d such that ed-1 is exactly divisible by z.
  - Public key is (n, e). Private key is (n, d).
- To encrypt a message, m (≤ n): do \( c = m^e \mod n \)
- To decrypt a received ciphertext, c: do \( m = c^d \mod n \)
- Magic happens: \( (m^e \mod n)^d \mod n = m \)
- Another property: \( (m^d \mod n)^e \mod n = m \)
- Note: RSA is much slower than the symmetric key cryptos.

Outline

- What is network security?
- Principles of cryptography
- Authentication
- Integrity

Authentication: version 1.0

Authentication: Bob wants Alice to "prove" her identity to him.

Protocol ap1.0: Alice says "I am Alice".

"I am Alice"  Failure scenario??

"I am Alice"  Trudy can simply declare herself to be Alice

Authentication: version 2.0

Protocol ap2.0: Alice says "I am Alice" and sends her secret password to "prove" it.

Protocol ap2.0: Alice says "I am Alice" and sends her secret password to "prove" it.

"I'm Alice"  Failure scenario??

Alice's password  playback attack: Trudy records Alice's packet and later plays it back to Bob

Authentication: version 3.0

Goal: avoid playback attack

Nonce: number (R) used only once-in-a-lifetime

ap3.0: Bob sends Alice a nonce, R. Alice must return R, encrypted with shared secret key

"I am Alice"

R  only Alice knows key to encrypt nonce, so it must be Alice!

Any failures or drawbacks?

Authentication: version 4.0

ap3.0 requires shared symmetric key. Key distribution can be a problem.

ap4.0: use nonce, public key cryptography.

"I am Alice"

Bob computes \( K_A(K_A(R)) = R \) and knows only Alice could have the private key, that encrypted R such that \( K_A(K_A(R)) = R \)
Security Hole with ap4.0

Man (woman) in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)

I am Alice

Send me your public key

Send me your public key

I am Alice

Trudy gets

\[ m = K^r_A(K^s_K(m)) \]

\[ m \in K^s_A(m) \]

\[ \text{encrypted with Alice's public key} \]

11/26/02  CSC 257/457 - Fall 2002  19

Man (Woman) in the Middle Attack

- Difficult to detect:
  - Bob receives everything that Alice sends, and vice versa. (e.g., so Bob, Alice can meet one week later and recall conversation)
  - The problem is that Trudy receives all messages as well!
- How to deal with this?

Outline

- What is network security?
- Principles of cryptography
- Authentication
- Integrity

Integrity

- Digital Signatures:
  - Cryptographic technique to ensure document integrity.
  - analogous to hand-written signatures.
  - sender (Bob) digitally signs document, establishing he is document owner/creator.
  - the recipient (Alice) receives the document and the digital signatures.
  - the recipient can be sure that the document is verifiable: Bob signed the document.
  - nonforgeable: the document hasn't been changed since Bob signed it.

Digital Signatures

- Bob signs m by encrypting with his private key, creating a digital signature \( K_B(m) \)
- \( \text{Dear Alice} \)
- Oh, how I have missed you. I think of you all the time... (blablabla)
- Bob
- Suppose Alice receives msg m and its digital signature \( K_B(m) \)
- Alice applies Bob's public key \( K_A \) to \( K_B(m) \) then checks whether \( K_A(K_B(m)) = m \).
- If so, whoever signed m must have used Bob's private key.

Message Digests

- apply a hash function \( H \) to \( m \), get a much smaller message digest \( H(m) \).
- public-key-encrypt the message digest to generate the digital signature \( K_B(H(m)) \).
- Note: it is possible for many messages sharing the same digest.
Digital signature = signed message digest

Bob sends digitally signed message digest:

H: Hash function

large message m

Bob's private key K_B

digital signature (encrypt)

H(m)

Alice verifies signature and integrity of digitally signed message:

Bob sends digitally signed message digest:

H: Hash function

large message m

Bob's private key K_B

digital signature (encrypt)

H(m)

encrypted msg digest K_B(H(m))

Alice's public key K_A

digital signature (decrypt)

H(m)

equal? H(m)

Internet Checksum: Poor Hash Function for Generating Message Digests

Given a message and its Internet checksum, it is easy to find another message with same checksum.

<table>
<thead>
<tr>
<th>message</th>
<th>ASCII format</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 . 9 30 30 2E 39 39 42 D2 42</td>
<td>IOU 1 0 0 . 9 B O B 49 4F 55 31 30 30 2E 39 39 42 D2 42</td>
</tr>
<tr>
<td>9 B O B 39 42 D2 42</td>
<td>B2 C1 D2 AC --- different messages --- B2 C1 D2 AC, but identical checksum!</td>
</tr>
</tbody>
</table>

Hash function property: given message digest x for message m, computationally infeasible to find another message m' such that x = H(m').

Good Hash Functions for Generating Message Digests

- MD5 hash function widely used
  - computes 128-bit message digest in 4-step process.
  - appears difficult to construct message m whose MD5 hash is equal to x.
- SHA-1 is also used.
  - US standard [NIST, FIPS PUB 180-1]
  - 160-bit message digest

Disclaimer

- Parts of the lecture slides contain original work of James Kurose, Larry Peterson, and Keith Ross. The slides are intended for the sole purpose of instruction of computer networks at the University of Rochester. All copyrighted materials belong to their original owner(s).