Network Security
Part I: Concepts

Raj Jain
Washington University
Saint Louis, MO 63131
Jain@cse.wustl.edu

These slides are available on-line at:
http://www.cse.wustl.edu/~jain/cse473-05/
Overview

- Security Statistics, Attacks, Requirements
- Secret Key and Public Key Encryption
- Hash Functions
- Message Authentication Code (MAC)
- Digital Signature and Digital Certificates
- RSA Public Key Encryption
Security Threat Statistics

- DoD networks were attacked 250,000 times in 1995 (well before Internet popularity)
- Of 38,000 friendly attacks, 65% succeeded
- Only 4% of successful attacks were noticed by network administrators
- Only a small fraction of those noticed were reported to authorities
- FBI reports 163 organizations lost $123M in 1999
Security Attacks

- **Passive:**
  - Release of message contents: Eavesdropping
  - Traffic analysis: monitoring frequency and length of messages, even encrypted nature of communication may be guessed
  - Difficult to detect

- **Active:**
  - Masquerade: Pretend to be some one else
  - Replay: Capture and reuse for unauthorized effect
  - Modification of message
  - Denial of Service
Security Requirements

- **Integrity**: Received = sent?

- **Availability**: Legal users should be able to use. Ping continuously ⇒ No useful work gets done.

- **Confidentiality and Privacy**: No snooping or wiretapping

- **Authentication**: You are who you say you are. A student at Dartmouth posing as a professor canceled the exam.

- **Authorization = Access Control**: Only authorized users get to the data
Link vs End-to-End Encryption

- Link $\implies$ All traffic secure. Vulnerable inside switches
Secret Key Encryption

- Also known as symmetric encryption
- Encrypted_Message = Encrypt(Key, Message)
- Message = Decrypt(Key, Encrypted_Message)
- Example: Encrypt = division
- 433 = 48 R 1 (using divisor of 9)
Public Key Encryption

- Invented in 1975 by Diffie and Hellman
- Encrypted_Message = Encrypt(Key1, Message)
- Message = Decrypt(Key2, Encrypted_Message)
Public Key Encryption

- RSA: Encrypted Message = $m^3 \mod 187$
- Message = Encrypted Message$^{107} \mod 187$
- Key1 = <3, 187>, Key2 = <107, 187>
- Message = 5
- Encrypted Message = $5^3 = 125$
- Message = $125^{107} \mod 187 = 5$
  = $125^{(64+32+8+2+1)} \mod 187$
  = $\{125^{64} \mod 187\}(125^{32} \mod 187)...$
  (125$^2$ mod 187)(125 mod 187) $\mod 187$
Modular Arithmetic

- $xy \mod m = (x \mod m) (y \mod m)$
- $x^4 \mod m = (x^2 \mod m)(x^2 \mod m)$
- $x^{ij} \mod m = (x^i \mod m)^j \mod m$
- $125 \mod 187 = 125$
- $125^2 \mod 187 = 15625 \mod 187 = 104$
- $125^4 \mod 187 = (125^2 \mod 187)^2 \mod 187$
  \[= 104^2 \mod 187 = 10816 \mod 187 = 157\]
- $128^8 \mod 187 = 157^2 \mod 187 = 152$
- $128^{16} \mod 187 = 152^2 \mod 187 = 103$
- $128^{32} \mod 187 = 103^2 \mod 187 = 137$
- $128^{64} \mod 187 = 137^2 \mod 187 = 69$
- $128^{64+32+8+2+1} \mod 187 = 69 \times 137 \times 152 \times 104 \times 125 \mod 187$
  \[= 18679128000 \mod 187 = 5\]
Public Key (Cont)

- One key is private and the other is public
- Message = Decrypt(Public_Key, Encrypt(Private_Key, Message))
- Message = Decrypt(Private_Key, Encrypt(Public_Key, Message))
Hash Functions

Example: CRC can be used as a hash
(not recommended for security applications)

Requirements:
1. Applicable to any size message
2. Fixed length output
3. Easy to compute
4. Difficult to Invert ⇒ Can’t find \( x \) given \( H(x) \) ⇒ One-way
5. Difficult to find \( y \), such that \( H(x) = H(y) \) ⇒ Can’t change msg
6. Difficult to find any pair \( (x, y) \) such that \( H(x) = H(y) \) ⇒ Strong hash
Digital Signature

- Message Digest = Hash(Message)
- Signature = Encrypt(Private_Key, Hash)
- Hash(Message) = Decrypt(Public_Key, Signature) \Rightarrow Authentic
- Also known as Message authentication code (MAC)

Private Key

Text → Hash → Digest → Signature

Public Key

Signature → Hash → Digest ← Text
Message Authentication Code (MAC)

- Authentic Message = Contents unchanged + Source Verified
- May also want to ensure that the time of the message is correct
- Encrypt({Message, CRC, Time Stamp}, Source’s secret key)
- Message + Encrypt(Hash, Source’s secret key)
- Message + Encrypt(Hash, Source’s private key)
MAC: Using One Way Hash

(a) Using conventional encryption

(b) Using public-key encryption

(c) Using secret value
Digital Certificates

- Like driver license or passport
- Digitally signed by Certificate authority (CA) - a trusted organization
- Public keys are distributed with certificates
- CA uses its public key to sign the certificate ⇒ Hierarchy of trusted authorities
- X.509 Certificate includes: Name, organization, effective date, expiration date, public key, issuer’s CA name, Issuer’s CA signature

User ID
Public Key

hash

Encrypt
CA private key

User ID
Public Key
Key Distribution

1. Application requests connection
2. Security service asks **KDC** for session Key
3. KDC distributes *session key* to both hosts
4. Buffered packet transmitted

KDC shares a secret key with each Host.
Confidentiality

- User 1 to User 2:
- Encrypted_MESSAGE = Encrypt(Public_Key2, Encrypt(Private_Key1, Message))
- Message = Decrypt(Public_Key1, Decrypt(Private_Key2, Encrypted_MESSAGE))
  ⇒ Authentic and Private

| Your Public Key | My Private Key | Message |
RSA Public Key Encryption

- Ron Rivest, Adi Shamir, and Len Adleman at MIT 1978
- Both plain text M and cipher text C are integers between 0 and n-1.
- Key 1 = \{e, n\},
  
  Key 2 = \{d, n\}
- C = M^e \mod n
  
  M = C^d \mod m
- How to construct keys:
  - Select two large primes: p, q, p ≠ q
  - N = p \times q
  - Calculate \(\Phi = (p-1)(q-1)\)
  - Select e, such that \(\text{lcm}(\Phi, e) = 1; 0 < e < s\)
  - Calculate d such that \(de \mod \Phi = 1\)
RSA Algorithm: Example

- Select two large primes: \( p, q, p \neq q \)
  \[ p = 17, \quad q = 11 \]
- \( N = p \times q = 17 \times 11 = 187 \)
- Calculate \( \Phi = (p-1)(q-1) = 16 \times 10 = 160 \)
- Select \( e, \) such that \( \text{lcm}(\Phi, e) = 1; \quad 0 < e < s \)
  say, \( e = 7 \)
- Calculate \( d \) such that \( de \mod \Phi = 1 \)
  \[ 160k + 1 = 161, 321, 481, 641 \]
  - Check which of these is divisible by 7
  - 161 is divisible by 7 giving \( d = \frac{161}{7} = 23 \)
- Key 1 = \{7, 187\}, Key 2 = \{23, 187\}
Summary

- Passive and active attacks
- Secret Key and Public Key Encryption
- Secure Hash Functions
- Message Authentication Code (MAC)
- Digital Signature and Digital Certificates
- RSA Public Key Encryption based on exponentiation
Reading Assignment

- Read Sections 21.1 through 21.4 of 7th edition of Stallings. You can skip AES, SHA-1 during this part.
Homework

- Submit answer to Exercise 21.6 in Stallings’ 7th edition