CSE/EE 461 – Lecture 26
Network Security II

Janet Davis
jlnd@cs.washington.edu
March 8, 2004
Reading: Peterson 8.2 – 8.3

Last time…

- "Good mathematical encryption techniques [help us send messages that] are hard to eavesdrop, alter, or forge."
- "Unbreakable encryption is the most important part of network security."
  - Actually, encryption is just a tool
  - Open problem: *Principles* for building secure systems!

Questions from last time…

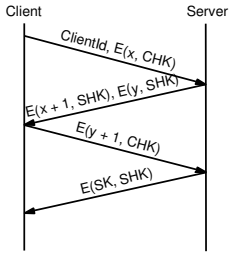
- Acronyms
  - DES = Data Encryption Standard
  - RSA = Rivest Shamir Adleman (inventors)
  - MD5 = Message Digest 5
- How long to break encryption?
  - Brute force approach is "embarrassingly parallel"
  - With enough resources, maybe only a few months!
  - Solutions: Triple-DES, bigger RSA keys
- Will this be on the test?
- Go over message digests again?

This time…

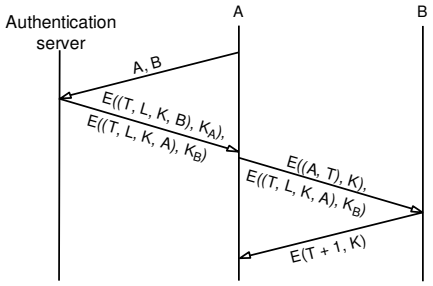
- Network security mechanisms
  - Authentication
  - Message integrity
  - Public key distribution
- Whirlwind tour of example systems
  - Pretty Good Privacy (PGP)
  - Secure Shell (SSH)
  - Secure Transport (SSL & HTTPS)
  - IP Security (IPSEC)
  - Wireless (WEP)

Authentication Protocols

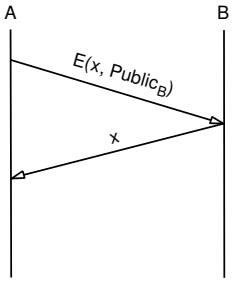
- Last time: Authentication with a secret key
  - Secret key often based on password!



Authentication via a Trusted Third Party (Kerberos)

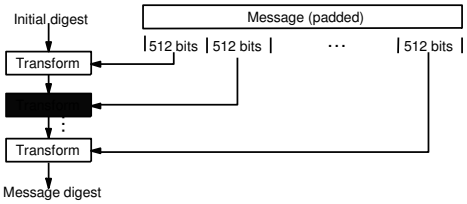


Public Key Authentication



Message Digests (MD5, SHA)

- Cryptographic checksum or hash
  - Typically small compared to message (MD5 128 bits)
  - "One-way": infeasible to find two messages with same digest



Message Integrity Protocols

- Sometimes we don't care about privacy but do care about integrity/authenticity
- Digital signature using RSA
  - Sign (encrypt) message with private key
  - Others verify (decrypt) with public key
- MD5 with RSA signature
  - Send m + E(MD5(m), $K_{private}$))
- Keyed MD5
  - Send m + MD5(m + k) + E(k, $K_{private}$)

Public Key Distribution

- When you give me your public key, how do I know you are who you say you are?

Certificates (X.509)

- Trusted *certification authority* (CA), e.g. Verisign
  - CA signs a statement: "I swear X's public key is Y"
  - X can then give this signed statement to anyone
  - Still need to bootstrap CA's public key…
- Build *chains of trust*
  - Similar to domain name hierarchy
  - "edu" (root server) certifies "washington.edu" key
  - "washington.edu" certifies "cs.washington.edu" key
  - Etc.
- Need *certificate revocation lists* to revoke keys!

Example Systems

- Secure Shell (SSH)
  - Secure login & IP tunnels
- Pretty Good Privacy (PGP)
  - Authentic and private email
- Secure transport (SSL/TLS) and Secure HTTP (HTTPS)
  - Secure web transactions
- IP Security (IPSEC)
  - Framework for encrypting/authenticating IP packets
- Wired Equivalent Privacy (WEP)
  - Privacy and access control for wireless networks

Secure Shell (SSH)

- Many protocols send password as plaintext!
  - telnet, rsh, rlogin, rcp, ftp,…
- SSH replaces these
  - Also lets you "tunnel" other IP applications for security
- Client authenticates server using public key
  - Depends on user to accept key of new/reconfigured server…
- Exchange session key to encrypt session
  - Client encrypts using server's public key
- Server authenticates client using password, public key, Kerberos, or client host key

L22.13

Pretty Good Privacy (PGP)

- Application level system
- Based on public key encryption and a "grass roots" web of trust
  - No certification authorities!
  - Instead, key-signing parties
- Sign messages for integrity/authenticity
  - e.g., RSA with MD5
- Encrypt messages for privacy
  - Encrypt message with a secret key
  - Encrypt secret key with public key of receiver

L22.14

SSL/TLS and HTTPS

- Transport layer security: SSL/TLS
  - Get HTTPS by running HTTP on top of SSL/TLS
- Extra handshake phase
  - Server provides a certificate, which client uses to authenticate server
  - Also exchange session state, e.g. session key
- Session resumption
  - Use a session id to refer to existing session state, abbreviating handshake for subsequent connections

L22.15

IP Security (IPSEC)

- Framework for encrypted and authenticated IP packets
  - Specifies packet formats, but not algorithms!
- Use new protocol headers inside IPv4 packets
  - Authentication Header
    - Signature for message integrity and origin authenticity
    - Optional "anti-replay" protection (via sequence number)
  - Encapsulation Security Payload
    - Adds encryption for privacy
- Depends on key management (ISAKMP)
  - Sets up security associations
- Example use: secure tunnels between corporate offices

L22.16

Wireless (802.11)

- Problem: Anyone nearby with a wireless card can listen to your communications
- Wired Equivalent Privacy (WEP)
  - Encrypt link-level data for privacy
  - Also access control, data integrity
- "Intercepting Wireless Communications: The Insecurity of 802.11" (Borisov, Goldberg, Wagner)
  - Despite using a good encryption algorithm, WEP has many vulnerabilities!

WEP Vulnerabilities

- WEP uses a *stream cipher* for privacy
  - (secret key + initialization vector) is expanded into a *keystream*, which is XORed with plaintext
  - Keystream reuse + known plaintext makes it possible to derive other plaintexts
- Standard does not specify key distribution
  - Typically everyone on a network shares one key
- Uses CRC-32 for message integrity
  - Not resistant to deliberate attacks
  - Attacker can trick access point into sending it decrypted packets by modifying packet destination address!
- Lessons: reuse past designs, get public review

Key Concepts

- Authentication protocols can use different kinds of encryption.
- Message integrity protocols use message digests for better performance.
- Need certification authority or a "web of trust" for public key encryption to work.
- Security at many layers: SSH, PGP, SSL/TSL, IPSEC, IP tunneled over SSH!
- To build a secure system, need not just to use encryption but to use it well!