# CPSC 317
# COMPUTER NETWORKING

Module 8: Security – Day 4 – Authentication and TLS


Some slides based on Kurose/Ross original slides, found at https://gaia.cs.umass.edu/kurose_ross/ppt.htm

# ADMINISTRATION

- Student Experience of Instruction Survey is now available and will continue until April 15th
- No clickers today

# LEARNING GOALS

- Describe the end-point authentication problem and the pros and cons of various solutions
- Describe the necessary elements of a security protocol: handshake, secret exchange, key derivation, data transfer, termination
- Describe how TLS provides all these elements

# READING

- Reading: 8.4, 8.6

# END-POINT AUTHENTICATION

- Goal: Bob wants Alice to “prove” her identity

# END-POINT AUTHENTICATION

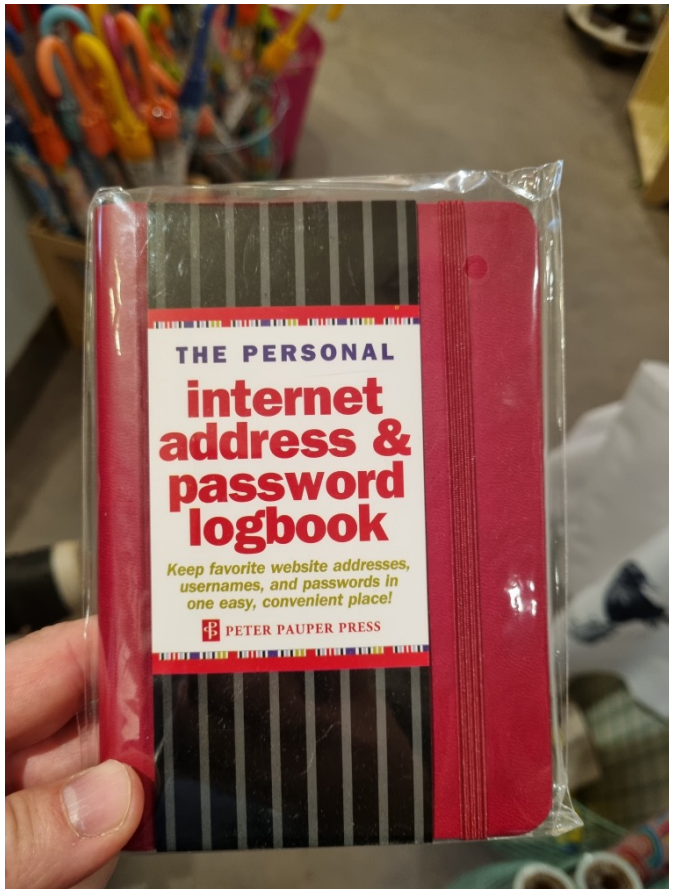
- Option 1: Alice sends a message with identification
  - Easily spoofed by Trudy

# END-POINT AUTHENTICATION

- Option 2: Alice includes own IP, known to Bob
  - Trudy can also spoof an IP and eavesdrop on connection

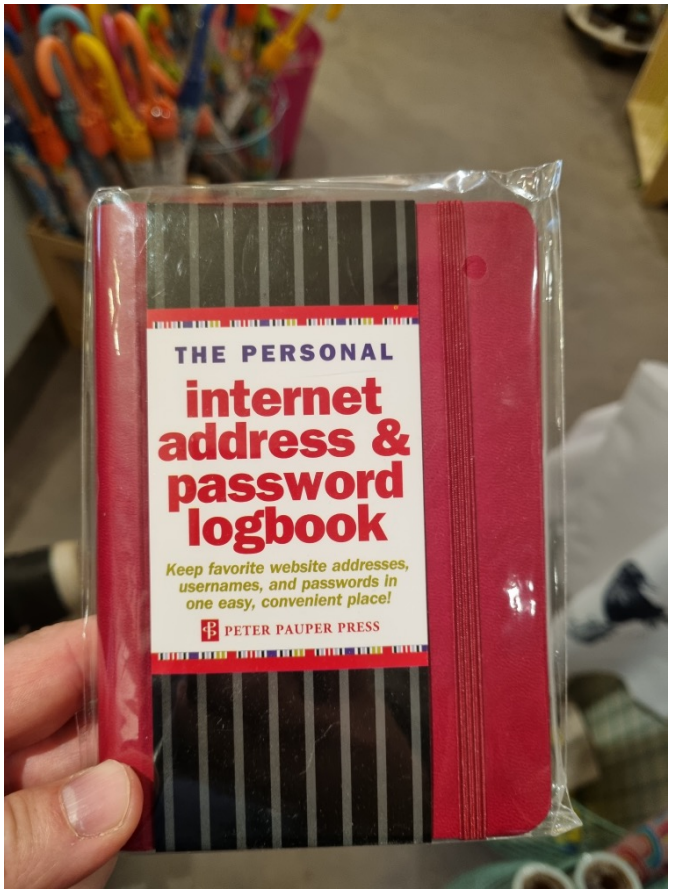
# END-POINT AUTHENTICATION: PASSWORD

- Option 3: use a password
  - If Trudy can eavesdrop, then password is leaked



# END-POINT AUTHENTICATION: PASSWORD

- Option 3.1: use an encrypted password
  - Trudy can just play it back unmodified
  - Trudy doesn't need original password if encryption is always the same
  - Called replay attack



# END-POINT AUTHENTICATION: NONCE

- Option 4: Bob sends one-time message (*R*) to Alice, Alice returns same message encrypted with shared key
  - Bob can then confirm Alice knows the shared encryption key
  - Requires a pre-established secret key
  - Password could be used, but relies on the password itself being secure enough

# END-POINT AUTHENTICATION: NONCE

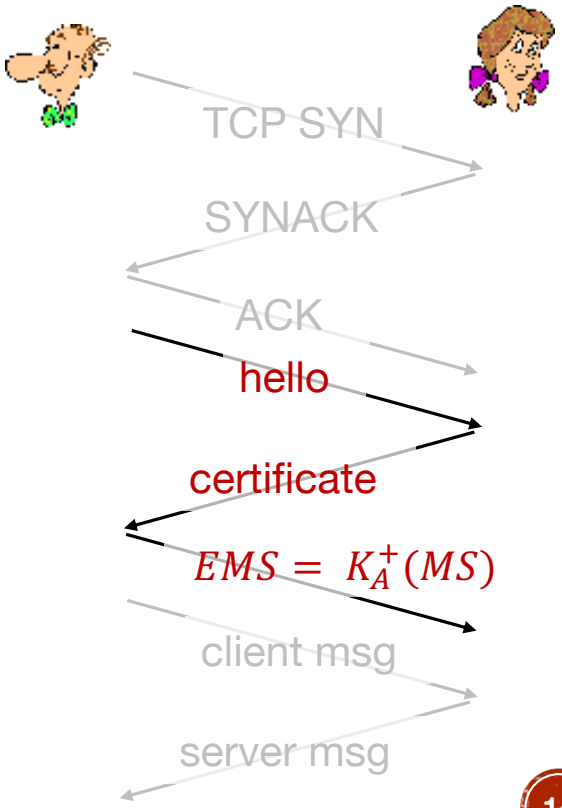
- Option 5: Bob sends one-time message (*R*) to Alice, Alice returns message encrypted with $K_A^-$
  - Only reliable if Bob has a way to confirm $K_A^+$ is actually Alice's
  - Subject to man-in-the-middle attack

# BUILDING A SECURITY PROTOCOL

- To build a complete security protocol, we need to provide:
  - Handshake: Alice, Bob use their certificates to authenticate each other, and private keys to share secret
  - Key derivation: Alice, Bob use shared secret to derive keys
  - Data transfer: series of messages (“records”)
  - Connection termination: securely close connection

# SECURITY PROTOCOL – HANDSHAKE

- Bob establishes TCP connection with Alice
- Bob verifies Alice's identity (certificate)
- Bob sends Alice master secret key MS
  - Used to generate all other keys for session
- Potential issue
  - 3 RTTs before client can start receiving data



# SECURITY PROTOCOL – KEY DERIVATION

- Alice and Bob generate the same four keys:
  - $K_B$: encryption key for data sent from Bob to Alice
  - $M_B$: MAC key for data sent from Bob to Alice
  - $K_A$: encryption key for data sent from Alice to Bob
  - $M_A$: MAC key for data sent from Alice to Bob
- Keys derived from predetermined key derivation function (KDF)
  - Process could be as simple as splitting MS into 4 parts

# SECURITY PROTOCOL – DATA TRANSFER

- Recall: TCP provides *byte stream* abstraction
- Where would MAC go?
  - Can't wait until end of transmission

# SECURITY PROTOCOL – DATA TRANSFER

- Data is broken into individual "records"
- Each record carries a MAC, created using $M_A$ or $M_B$
- Receiver can act on each record as it arrives
- To send message $m$ with length $l$ from Alice to Bob, send:

$$K_A\big(l, m, H(m + M_A)\big)$$

# SECURITY PROTOCOL – DATA TRANSFER –V2

- But we should be doing encrypt, then MAC
- To send message $m$ with length $l$ from Alice to Bob, send:

$$H(K_A(l, m) + M_A), K_A(l, m)$$

# SECURITY PROTOCOL – DATA TRANSFER

- Possible attack on data stream:
  - Reordering (attacker intercepts TCP segments and manipulates sequence numbers)
  - Replay (same TCP message sent twice, manipulating TCP)
- Solution:
  - Include sequence number $n$ in MAC computation
  - Use $H(K_A(l, m) + M_A + n)$ as hash

# SECURITY – CONNECTION TERMINATION

- Possible attack on data stream:
  - Attacker forges TCP connection close segment (truncation attack)
- Solution: special message type for closure
- To send regular message $m$ with length $l$ and sequence number $n$ from Alice to Bob, send:

$$H(K_A(l, 0, m) + M_A + n), K_A(l, 0, m)$$

- To send closing message $m$ with length $l$ and sequence number $n$ from Alice to Bob, send:

$$H(K_A(l, 1, m) + M_A + n), K_A(l, 1, m)$$

# TRANSPORT-LAYER SECURITY (TLS)

- Widely deployed security protocol above the transport layer
- Technically application layer, but could be thought of as an in-between layer
- Replaces the Secure Socket Layer (SSL) protocol
- TLS is built to support any type of application
  - Example: HTTPS is HTTP with messages sent over SSL/TLS

# TRANSPORT-LAYER SECURITY (TLS)

- Confidentiality: via symmetric encryption
- Integrity: via MAC
- Authentication: via public key cryptography and certificates

# TRANSPORT-LAYER SECURITY (TLS)

- TLS supports several algorithms for:
  - Key generation
  - Encryption
  - MAC
  - Digital signature
- Cipher suite: choice of algorithms negotiated during handshake
  - Client sends supported cipher suites
  - Server chooses one of the supported cipher suites

# QUIC

- Transport-layer protocol running on top of UDP
- Provides reliability of TCP, plus security of TLS
  - Can combine connection establishment and security handshake
- Standardized in RFC 9000

# IN-CLASS ACTIVITY

- ICA84