SSL/TLS as Real-Time Protocols

- A real-time protocol is one where parties negotiate interactively to authentication each other and establish a session key
- Examples: IPsec, SSL/TLS, SSH
  - Public key based
- SSL: Secure Socket Layer
- TLS: Transport Layer Security

SSL/TLS is on top of TCP

- SSL/TLS is in a user-level process
  - No requirement on OS changes
- Relies on TCP to ensure reliable delivery
  - Timing out issues or lost data will be retransmitted
- But relying on TCP introduces the rogue packet problem...

The Rogue Packet Problem

- A rogue packet with malicious data can be inserted into TCP stream
- TCP won’t notice and forwards that to SSL
  - And will expect next packet in sequence
- SSL discard it
- Now the genuine packet comes
- TCP now discards the packet because the packet appears to be a duplicate :)
A Compromised Decision

• SSL/TLS is on top of TCP, not UDP
  – No worry about reliability issues
• But has to live with the rogue packet problem

Quick History

• SSLv1: never deployed
• SSLv2: deployed in Netscape Navigator 1.1 in 1995
• Microsoft introduced PCT (Private Communication Tech) by improving SSLv2
• Netscape overhauled the protocol as SSLv3
• IETF introduced TLS to unify all of them
  – Seems just another incompatible protocol

SSL/TLS Processing Unit

• TCP stream is partitioned into records
• Each record has a header and crypto protection
• Four types of records:
  – User data
  – Handshake messages (we focus on this one)
  – Alerts
  – Change cipher spec
    • should be regarded as handshake

SSLv3/TLS Basic Protocol

Choose secret \( S \), compute

\[ K = f(S, R_{Alice}, R_{Bob}) \]

\( \{\text{keyed hash of handshake msgs}\}^{Bob} \)

\( \{\text{keyed hash of handshake msgs}\}^{Alice} \)

Data protected w/ keys derived from \( K \)

– E.g. Alice, password, credit card info

If a Keyed Hash Result in Plaintext

Choose secret \( S \), compute

\[ K = f(S, R_{Alice}, R_{Bob}) \]

\( \{\text{keyed hash of handshake msgs}\}^{Bob} \)

\( \{\text{keyed hash of handshake msgs}\}^{Alice} \)

Data protected w/ keys derived from \( K \)

– E.g. Alice, password, credit card info

Several Important Terms

• \( R_{Alice} \): a random number from Alice
• \( S \): pre-master secret
• \( K \): master secret
• \( \{\}_\text{Bob} \) stands for message encrypted with Bob’s public key
• \( \{\}_\text{Alice} \) stands for protected message using encryption and/or integrity protection through secret key algorithm
How Bob Verifies the Key Hash

- Decrypt $\{S\}_B$ using his private key
- Compute $K=\{S, R_{Alice}, R_{Bob}\}$
- Calculate $hash(K, (m1, m2, "CLNT"))$
  - HMAC algorithm
- Compares the result with the received one
- Verified if equal

Q: must the keyed hash be protected?

How Alice Verifies the Key Hash

- Calculate $hash(K, (m1, m2, "SRVR"))$
  - HMAC algorithm
  - Recall Alice knows $K$ already
  - The constant string make the hash different from what Bob receives
- Compares the result with the received one
- Verified if equal

Q: must the keyed hash be protected?

Questions

- Can Eve eavesdrop?
- Can Mallury manipulate the data stream?

When Eve is Eavesdropping

When Mallury is Manipulating

Questions

- When hashing, why add "CLNT" or "SRVR"?
- What if not?
If Verified, What does Bob Prove?

- The following can be regarded as the same entity:
  - The one sending, or forwarding, message 1
  - The one computing the pre-master secret that Bob received
  - The one sending message 3
- But not necessarily Alice, even claimed so!
  - Could be Mallory!
  - But Alice won’t be deceived

If Verified, What does Alice Prove?

- The following are the same entity:
  - The one sending message 2
  - The one computing S and K on the other end, and
  - The one sending message 4
- And this entity is Bob!
  - Based on the certificate
- Also, this entity knows S and K
  - S and K are decided by Alice
- All handshake messages so far have NOT been tampered
  - Otherwise?

More Issues on SSL/TLS

- Six secrets to protect Alice-Bob communication
- Handling a long session with many connections
- What if Alice also has a certificate

Six Secrets

- In fact, it’s not a single key K for a session
- Definition: write keys and read keys
  - Write keys: keys for transmission
  - Read keys: keys for reception
- Each direction needs three write keys
  - Integrity protection key
  - Encryption key
  - IV, if required by encryption algorithms
- And also three read keys
- Computed using $g_i(K, R_{Alice}, R_{Bob})$

One Session w/ Multiple Connections

- From a long SSL session, after one connection is set up, many other connections can further be derived
  - Alice (a browser) and Bob (a web site) can have many connections, for instance
- Simplify the SSL for later connections between Alice and Bob
  - They have gone through the pain anyway . . .
Session Initiation

I want to talk, ciphers I support, \( R_{Alice} \)

\[
\text{session_id, certificate, cipher, } R_{Alice}
\]

\[
(\text{keyed hash of handshake msgs})
\]

Data protected with keys derived from \( K \)

\( \text{e.g., Alice, password, credit card info} \)

Session Resumption

\( \text{session_id, ciphers I support, } R_{Bob} \)

\[
(\text{keyed hash of handshake msgs})
\]

Data protected with keys derived from \( K \)

\( \text{e.g., Alice, password, credit card info} \)

SSL/TLS is Asymmetrical

- Alice authenticated Bob
- But Bob does not authenticate Alice
  - Until Alice login to Bob
    - Kind of late
  - Could be Mallory handshaking with Bob
- SSL/TLS can be enhanced for mutual authentication
  - If the client has a certificate

Quiz 3

- How would the SSL/TLS protocol work if Alice also has a certificate?

Quiz 1: 410avg=4.5  510avg=9.1
Quiz 2: 410avg=3.5  510avg=8.1

Encoding SSL/TLS Protocol

- Read Textbook Page 490 - 497.