Secure Sockets Layer (SSL) and Transport Layer Security (TLS): Part 2

Gaurav S. Kasbekar
Dept. of Electrical Engineering
IIT Bombay

References

- J. Kurose, K. Ross, "*Computer Networking: A Top Down Approach*", Sixth Edition, Pearson Education, 2013
- A. Tanenbaum, D. Wetherall, "*Computer Networks*", Fifth Edition, Pearson Education, 2012.
- L. Peterson, B. Davie, "*Computer Networks: A Systems Approach*", Fifth Edition, Morgan Kaufmann, 2012.
- W. Stallings, "*Cryptography and Network Security: Principles and Practice*", Pearson Education, 7th edition, 2016
- E. Rescorla, "*SSL and TLS: Designing and Building Secure Systems*", Addison-Wesley, 2001

Actual SSL

- SSL does not mandate that a particular symmetric key encryption algorithm, public key encryption algorithm or MAC computation algorithm be used
- Instead, these can be agreed upon by Alice and Bob during handshake
- Advantage of this flexibility:
  - if a specific algorithm broken or weakness found in it, then another one can be used
- Several security systems offer such flexibility for similar reasons
- Also, during SSL handshake, Alice and Bob send nonces to each other; these are used in computation of session keys E_B, E_A, M_B and M_A

Actual SSL Handshake

1) Client sends a list of cryptographic algorithms it supports, along with a client nonce, to server
2) Server selects a symmetric key algorithm, a public key algorithm, a MAC computation algorithm, etc., from above list; sends its choices, a certificate and a server nonce to client
3) Client verifies certificate, extracts server's public key, generates a random Pre-Master Secret (PMS), encrypts it with server's public key and sends encrypted PMS to server
4) Client and server independently compute the Master Secret (MS) from the PMS and nonces
   - E_B, E_A, M_B and M_A are then generated from the MS
   - Also, when the chosen symmetric key algorithm uses Cipher Block Chaining, the two Initialization Vectors (IVs) (one for each direction) are obtained from the MS
   - Henceforth, all messages sent between client and server are encrypted and a MAC is added to them
5) The client sends a MAC of all the handshake messages it sent and received
6) The server sends a MAC of all the handshake messages it sent and received

Actual SSL Handshake (contd.)

- Reason for sending MACs of all handshake messages in steps 5 and 6:
  - ❑ to detect any modification by an intruder
  - ❑ *e.g.*, if Trudy controls a compromised intermediate router, in step 1, may delete strong cryptographic algorithms from list, forcing server to select a weak one
  - ❑ if server (respectively, client) detects inconsistency in MAC received in step 5 (respectively, 6), then terminates connection
- Reason for using nonces in steps 1 and 2:
  - ❑ client nonce protects client against "connection replay" attack; server nonce protects server against "connection replay" attack
  - ❑ *e.g.* of connection replay attack: an intruder Trudy may sniff all the messages of an SSL session; next day may try to connect to Alice masquerading as Bob or vice versa
- Is server authentication done in above handshake?
  - ❑ Yes, since server sends certificate, client verifies it, client encrypts PMS using public key in the certificate and client verifies MAC in step 6
- Is client authentication done?
  - ❑ No

Client Authentication

- Client authentication is optionally performed as part of SSL handshake (details on next slide)
- Often, it is not performed in SSL handshake since:
  - clients are often users (e.g., Gmail account users) who do not have public-private key pairs or certificates
- Instead, after the handshake is completed:
  - the server-side application using SSL prompts the client (user) for a password and verifies the password
- Is this method of client authentication vulnerable to replay attack?
  - No; since password is encrypted using a session key (E_B), which is different for each SSL session

Client Authentication (contd.)

- If client authentication is done as part of SSL handshake, following steps are performed:
  - ❑After server sends its own certificate to client, server sends a "Certificate Request" message
  - ❑Client sends its certificate to server; server extracts public key of client from it
  - ❑Then client sends a "Certificate Verify" message to server, which contains hash value of handshake messages exchanged so far, signed using client's private key
  - ❑Server applies client's public key to signed hash value and verifies the result to authenticate client
- Can an intruder later falsely authenticate itself as client by replaying "Certificate Verify" message?
  - ❑No; since hash value in "Certificate Verify" message is a function of server and client nonces

Connection Closure

- Suppose Bob wants to close the connection
- One approach: Bob sends a “TCP FIN” packet to close underlying TCP connection
- Is this a secure approach?
  - ❑No, an intruder Trudy may close connection before Bob has finished sending his data
- Procedure to close connection in SSL:
  - ❑Bob sends an SSL record with the type field in it indicating that he wants to close the connection
- Recall: although type field sent in plaintext form, it is included in MAC computation; hence, modification can be detected

Cryptographic Algorithm Options

- Recall: in steps 1 and 2 of SSL handshake:
  - client sends a list of cryptographic algorithms it supports to server
  - server selects a symmetric key algorithm, a public key algorithm, a MAC computation algorithm, etc., from above list and sends its choices
- Examples of symmetric key algorithms:
  - DES, 3DES, DES40, AES, IDEA, RC4, RC2
- Examples of MAC computation algorithms:
  - MD5, SHA-1
- The key exchange method used by client and server for agreeing upon the Pre-Master Secret is also selected in steps 1 and 2 of SSL handshake

Key Exchange Methods

- Recall: key exchange method used by client and server for agreeing upon the Pre-Master Secret is selected in steps 1 and 2 of SSL handshake
- In step 3 of above SSL handshake, client generates a random Pre-Master Secret (PMS), encrypts it with server's RSA public key and sends encrypted PMS to server
- Alternative key exchange method: *Ephemeral Diffie-Hellman* (DH):
  - ❑ Client and server randomly choose one-time DH private and public keys
  - ❑ DH public key signed using RSA private key of sender sent to receiver
  - ❑ Using received values, client and server compute shared secret as in DH, which acts as PMS
- Why is DH public key signed using RSA private key of sender?
  - ❑ To prevent man-in-the-middle attack
- Advantage of Ephemeral DH over method in which client generates random PMS and sends PMS encrypted using server's RSA public key:
  - ❑ Consider eavesdropping intruder who records all messages exchanged between client and server; suppose later, intruder somehow obtains RSA private key of server
  - ❑ Then under latter method, intruder can decrypt all communication that took place between client and server
  - ❑ Under Ephemeral DH method, intruder cannot decrypt communication since PMS cannot be computed using messages exchanged between client and server
  - ❑ That is, Ephemeral DH method provides *Forward Secrecy*, whereas latter method does not