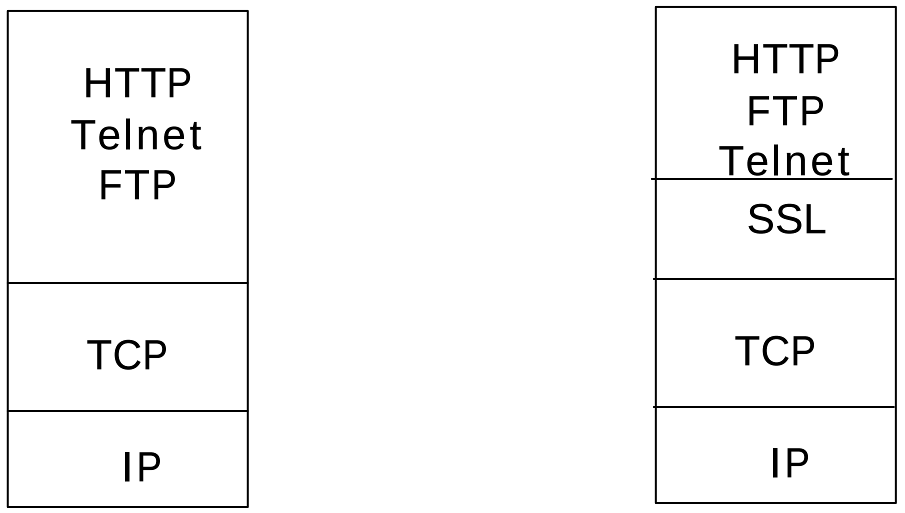
# SSL: Secure Socket Layer
# TLS: Tansport Layer Security

# SSL: Secure Sockets Layer



Suite of (*stateful*) protocols for:

- Entity authentication
- Message integrity
- Message confidentiality

# SSL: Secure Sockets Layer

It provides also

- secure key exchange between a browser (client) and server.
- security parameters negotiation.
- does not offer non-repudiation

Client to server authentication, not user authentication

Authentication at session level not at appliation level or network level

## SSL

- Advandages
  - Easy to deploy because embedded in the web software (browser and servers) ..and it comes for free
  - User friendly

- Disadvantages
  - Application that do not use web interface need to be SSLzed
  - No assurance at user level

## An SSL Session

1. Negotiation of cryptographic parameters

   Two computers probably don’t know each other’s capabilities.

2. Key Agreement.

   Client and Server generate shared secret key.

3. Authentication

   Client authenticates server. Optionally mutual authentication.

4. Confidentiality and integrity.

   Private messages exchanged between C&S.

# SSL: Handshake

- Negotiate Cipher-Suite Algorithms
  - Symmetric cipher to use
  - Key exchange method
  - Message digest function
- Establish and share pre-master secret
- Optionally authenticate server and/or client
- Generate shared secret from pre-master and random value exchanged

## Handshake Phases

- Hello messages
- Certificate and Key Exchange messages
- Change CipherSpec

## Hello and Negotiate Parameters

- Client sends server a plaintext message to suggest some parameters for conversation:

Version:

SSL 3.1 if you can, else SSL 3.0

Key Exchange:

RSA if you can, else Diffie-Hellman

Secret Key Cipher Method:

TripleDES if you can, else DES

Message Digest:

MD5 if you can, else SHA-1

Random #: 777,666,555

## Hello and Negotiate Parameters

- Server responds by its choice of parameters in a plaintext message:

Version:

SSL 3.1

Key Exchange:

RSA

Secret Key Cipher Method:

TripleDES

Message Digest:

SHA-1

Random #: 444,333,222

## SSL: Hello and Negotiation

- Client "*Hello*" - initiates session
  - Propose protocol version
  - Propose cipher suite
  - Server chooses protocol and suite
- Client may request use of cached session
  - Server chooses whether to honor request

## Key Agreement and Exchange

- Server sends certificate containing public key (RSA) or Diffie-Hellman parameters
- Client sends encrypted “pre-master” secret to server using Client Key Exchange message
- Master secret calculated
  - Use random values passed in Client and Server Hello messages

# Key Agreement and Exchange

- If RSA is agreed. The client generates a 48-byte random value (called pre-master secret), encrypts it with server's public RSA key, and sends it to server.
- The server decrypts this message and generates six keys.

| client | server |
|---|---|
| DES secret key | DES secret key |
| Secret key for message integrity (MAC) | Secret key for message integrity (MAC) |
| IV for block cipher | IV for block cipher |

## Key Agreement and Exchange

- Generation of six shared secret keys:
  - Random values exchanged.
  - Pre-master secret.
  - Pseudo-random function generator.

Example with TLS 1.0:

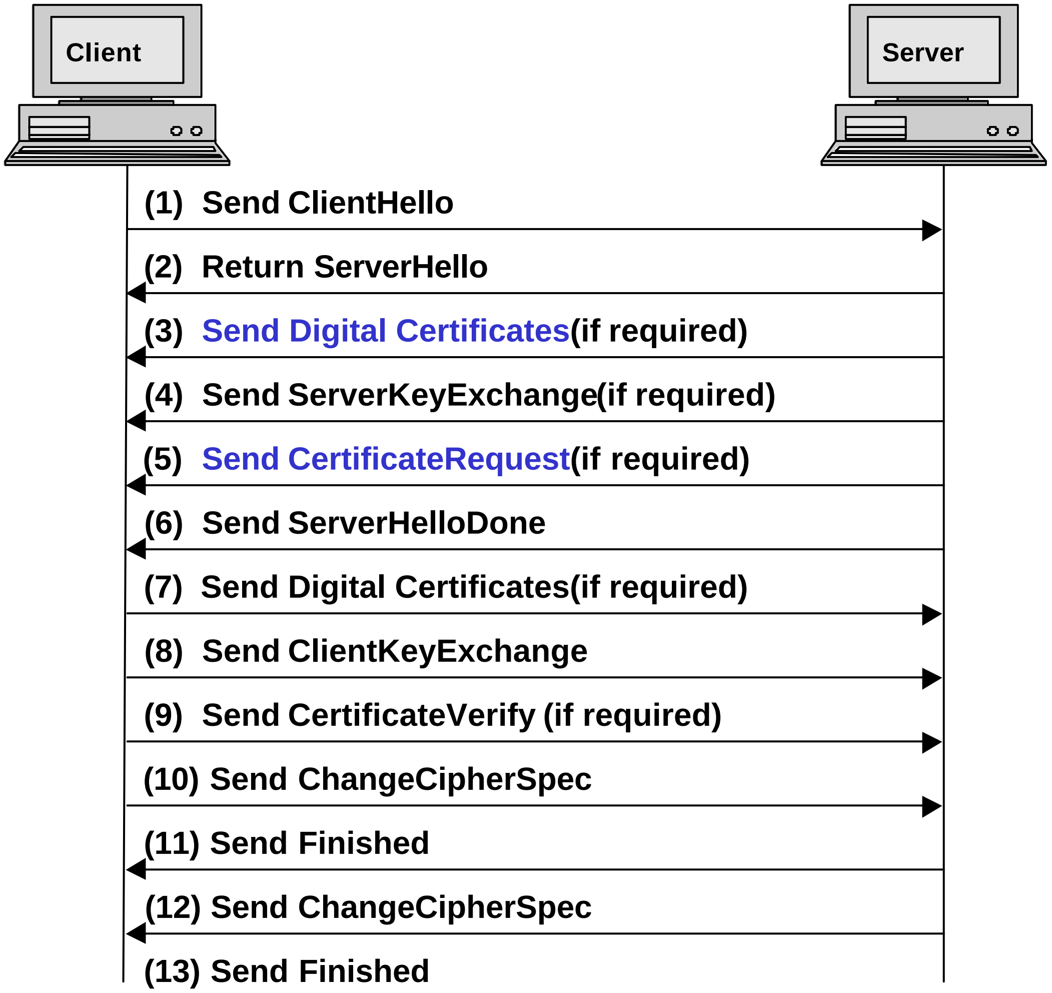
master_secret=

PRF(pre-master secret, master_secret, Client.random+Server.random)

48 bits long

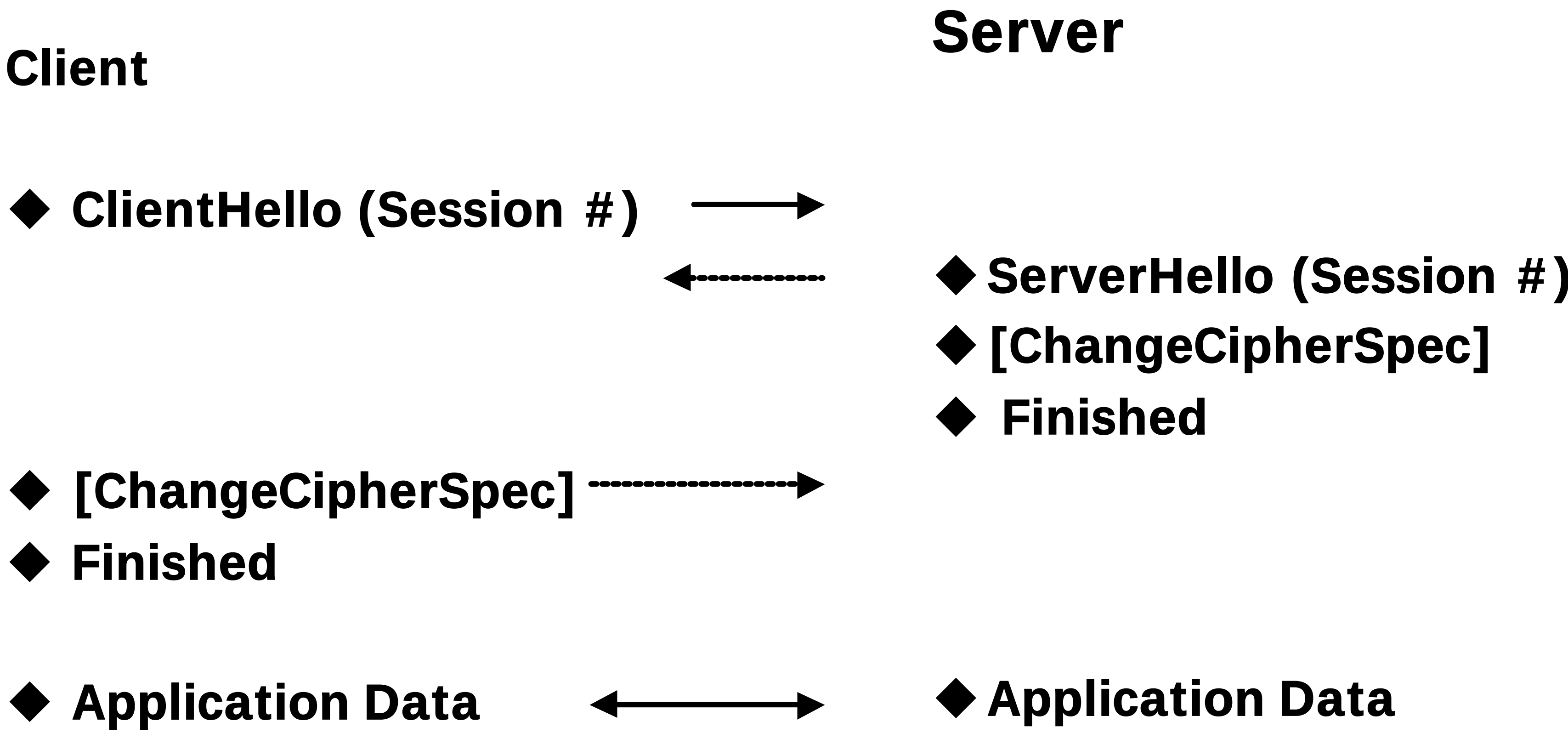
Computed repeatedly.

# Handshake protocol


Client
Server
(1) Send ClientHello
(2) Return ServerHello
(3) Send Digital Certificates(if required)
(4) Send ServerKeyExchange(if required)
(5) Send CertificateRequest(if required)
(6) Send ServerHelloDone
(7) Send Digital Certificates(if required)
(8) Send ClientKeyExchange
(9) Send CertificateVerify (if required)
(10) Send ChangeCipherSpec
(11) Send Finished
(12) Send ChangeCipherSpec
(13) Send Finished

**(a) Full version**

# SSL: Using a Session



# SSL: Change Cipher Spec/Finished

- Change Cipher Spec
  - Announce switch to negotiated algorithms and values (*pending to current state*)
- Finished
  - Send copy of handshake using new session
  - Permits validation of handshake

## Record Phase: Confidentiality and Integrity

- Client and server use the generated secret keys for confidential data transfer.
  - The client uses its secret key to generate a HMAC for the message.
  - The client encrypts message data + HMAC with its secret key and sends it to server.
  - The server decrypts the received message with its secret key.
  - The server checks the integrity of the message using HMAC.

# SSL: Implementation

- Cryptographic Libraries
  - RSARef
  - TLS/SSL packages
  - SSLeay
  - SSLRef

- References:

RFC 2246 T. Dierks, C. Allen “The TLS Protocol Version 1.0”, January 1999

SSL v3 Specification, Netscape http://www.netscape.com/eng/ssl3/