CSC429 – Computer Security

LECTURE 12
NETWORK SECURITY

Mohammed H. Almeshekah, PhD
meshekah@ksu.edu.sa

Network Security

IPSec

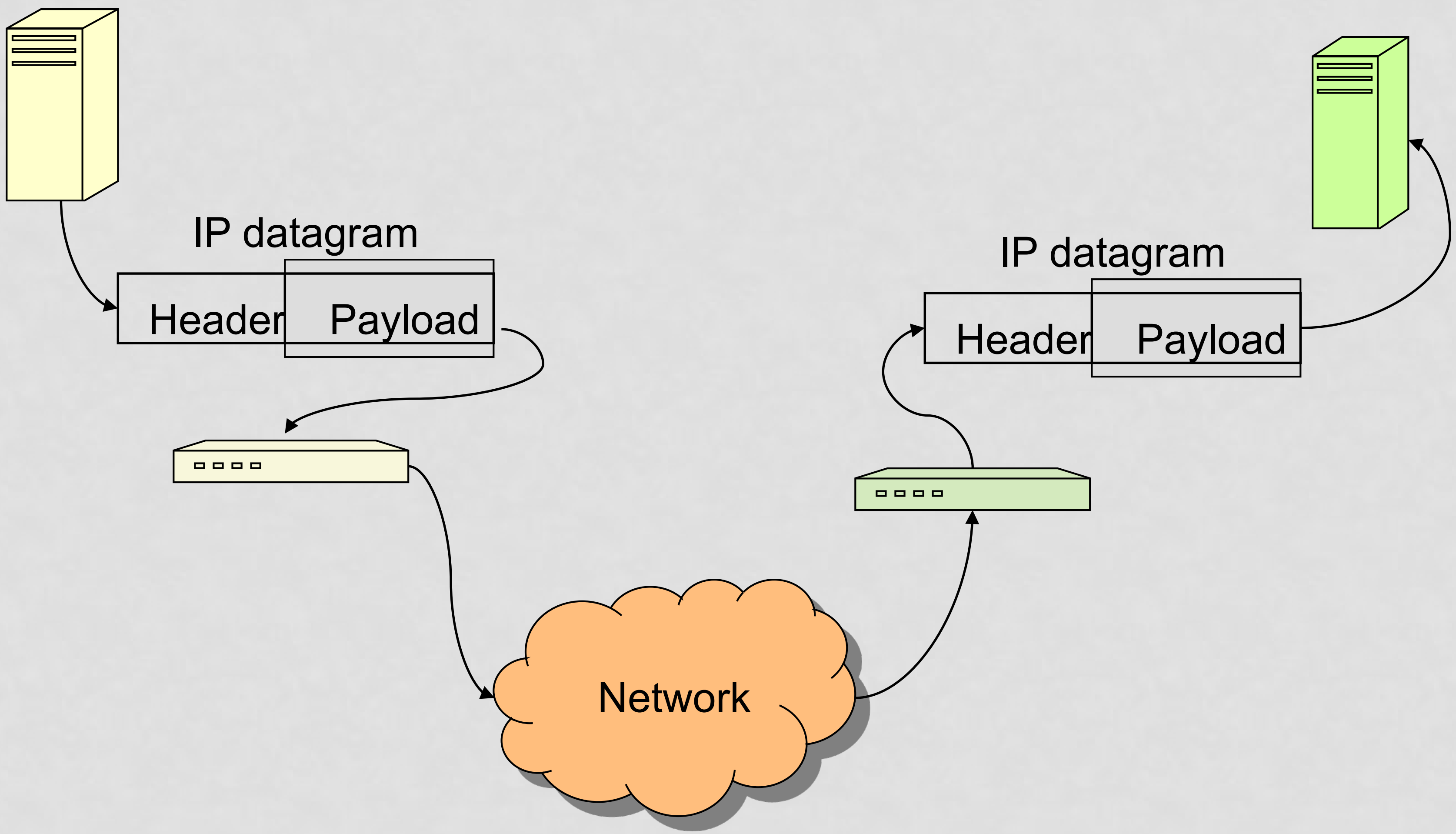
IPsec Basic Features

- IPsec provides two basic modes of use:
  - Transport mode: for IPsec-aware hosts as endpoints.
  - Tunnel mode: for IPsec-unaware hosts, established by intermediate gateways or host OS.
- IPsec provides authentication and/or confidentiality services for data
  - AH and ESP protocols.
- AH and ESP can each be applied multiple times (in tunnel or transport mode) to a given datagram.

IPsec Transport Mode

- Protection for upper-layer protocols.
- Protection covers IP datagram payload (and selected header fields).
  - Could be TCP packet, UDP, ICMP message,….
- Host-to-host (end-to-end) security:
  - IPsec processing performed at endpoints of secure channel.
  - So endpoint hosts must be IPsec-aware for transport mode.

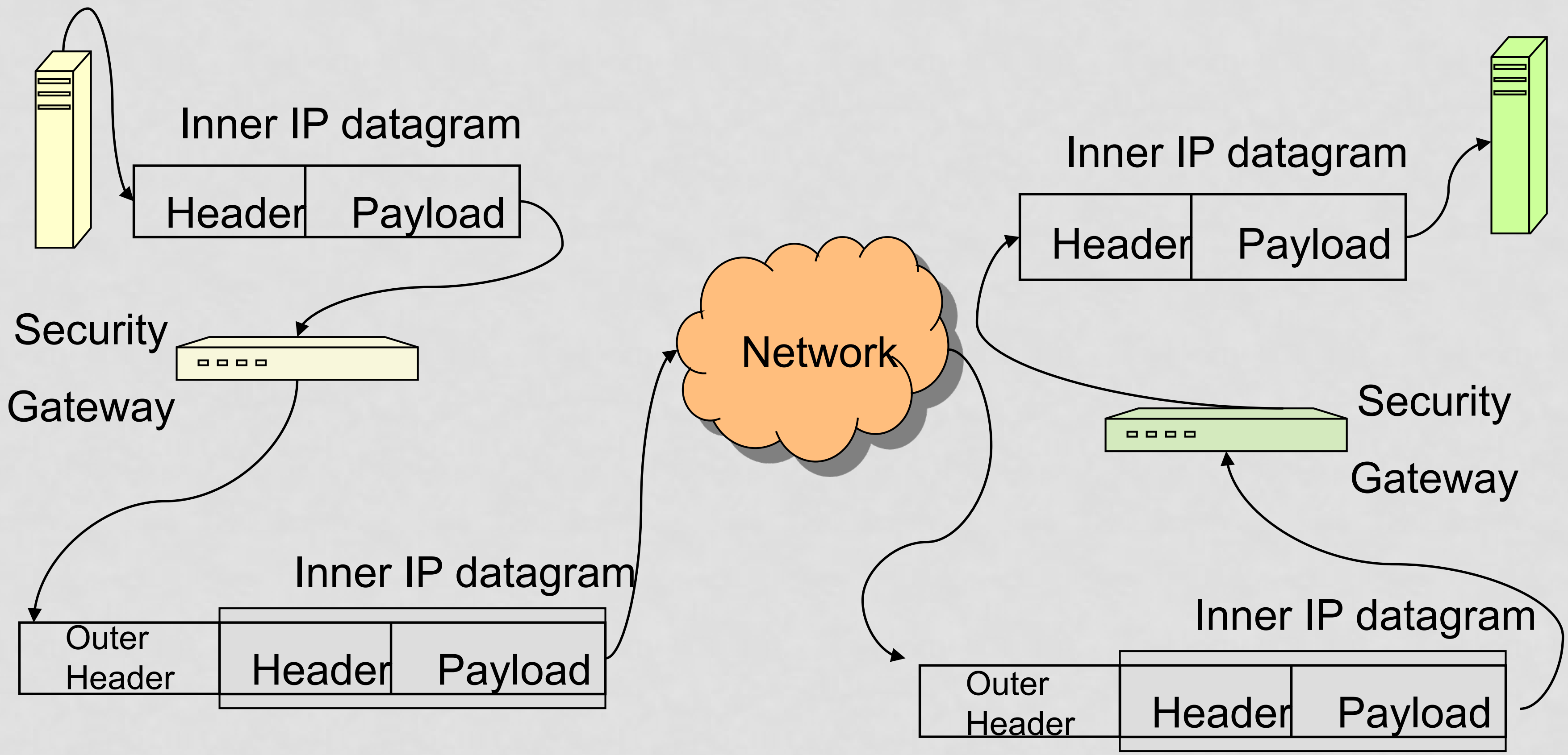
IPsec Transport Mode



IPsec Tunnel Mode

- Protection for entire IP datagram.
- Entire datagram plus security fields treated as new payload of "outer" IP datagram.
- IPsec processing is performed at *security gateways* on behalf of endpoint hosts.
    - Gateway-to-gateway rather than end-to-end security.
    - Hosts need not be IPsec-aware.
- Intermediate routers have no visibility of inner IP datagram.
    - Even original source and destination addresses encapsulated and so "hidden".

IPsec Tunnel Mode



AH Protocol

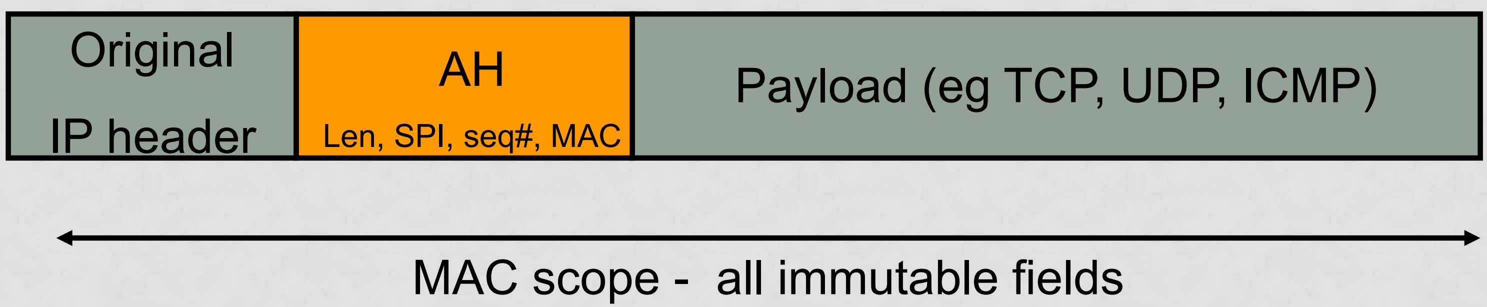
- AH = Authentication Header (RFC 4302).
- Provides data origin authentication and data integrity services.
- AH authenticates whole payload and most of header.
- Prevents IP address spoofing.
  - Source IP address is authenticated.
- Creates stateful channel.
  - Use of sequence numbers.
- Prevents replay of old datagrams.
  - AH sequence number is integrity protected.
- Uses MAC and symmetric key shared between endpoints.

AH Protocol

- AH specifies a header added to IP datagrams

- Fields in header include:
  - Payload length.
  - SPI = Security Parameters Index.
    - Identifies which algorithms and keys are to be used for IPsec processing (more later).
  - Sequence number.
  - Authentication data (the MAC value).
    - Calculate over immutable IP header fields (so omit TTL, checksum, fragmentation fields,…) and payload.

AH Protocol – Transport



ESP Protocol

- ESP = Encapsulating Security Payload (RFC 4303).

- Provides one or both of:
  - Confidentiality.
    - Protection for payload.
  - Authentication/integrity protection
    - Protection for payload.
    - Protection of ESP header and trailer fields (including sequence number).
    - But IP header fields (original header or outer header) are unprotected.

- Uses symmetric encryption and MACs based on secret keys shared between endpoints.

- Gives limited traffic-flow confidentiality in tunnel mode.

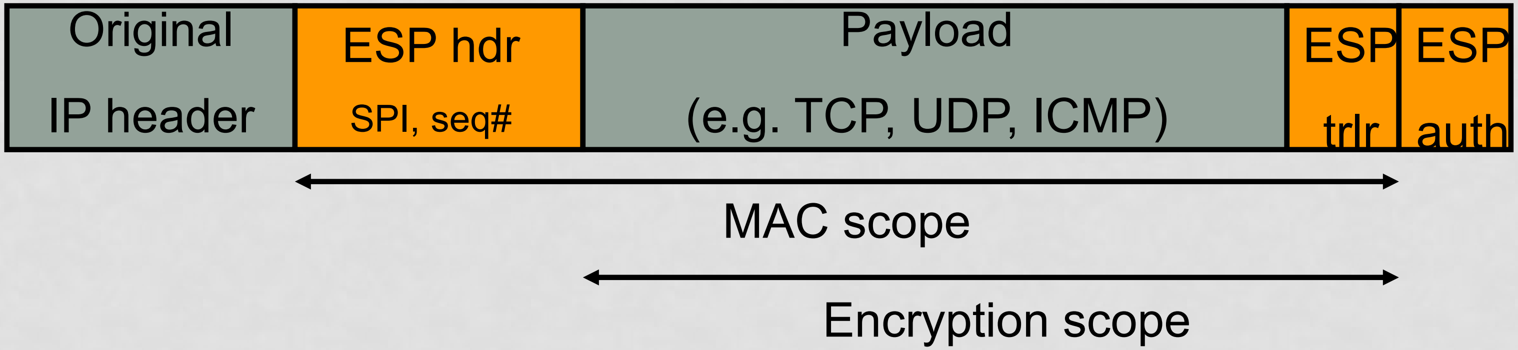
ESP Protocol

- ESP specifies a header and trailing fields to be added to IP datagrams.

- Fields in header include:
  - SPI.
  - Sequence number.

- Fields in trailers include:
  - Optional padding for traffic flow confidentiality (TFC).
  - Any padding needed for encryption algorithm (may also help disguise payload length).
  - Padding length.
  - Next header field.
  - Authentication data (if any) – the MAC value.

ESP Protocol – Transport



Integrity Protection in AH and ESP

- Separate existence of authentication/integrity protection in both AH and in ESP for performance and backwards-compatibility.
  - Original version of ESP (RFC 1827) had no integrity protection mechanism.
- Integrity protection has different scope in ESP and AH.

ESP (Encryption-only)

- IPsec allows selection of "encryption-only" configurations.
  - ESP using an encryption algorithm but no MAC algorithm.
- These are now known to be extremely insecure against active attacks.
- AH followed by ESP also has weaknesses in some configurations.
- Safest approach is to always use ESP with encryption and MAC algorithm.

Network Security

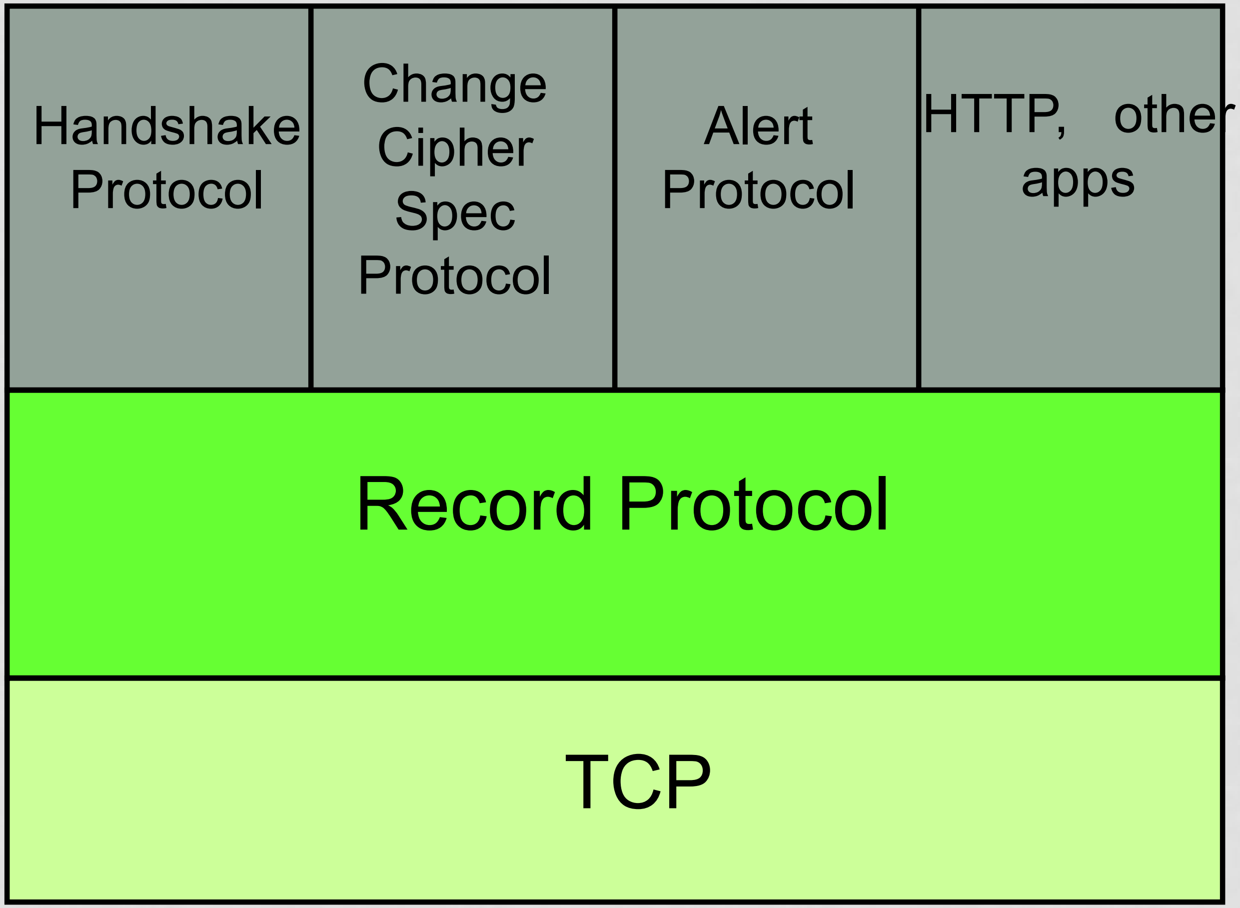
SSL/TLS

SSL/TLS Overview

- SSL/TLS widely used in Web browsers and servers to support ‘secure e-commerce’ over HTTP.
  - Use indicated by presence of browser lock.

- SSL/TLS architecture provides two layers:
  - Record Protocol
    - Provides secure, reliable channel to upper layer.
  - Upper layer carrying:
    - Handshake Protocol, Change Cipher Spec. Protocol, Alert Protocol, HTTP, any other application protocols.

SSL/TLS Protocol Architecture



SSL/TLS Record Protocol

- Session concept:
  - Sessions created by Handshake Protocol.
  - Session state defined by session ID and set of cryptographic parameters (encryption and hash algorithm, master secret, certificates) negotiated in Handshake Protocol.
  - Each session can carry multiple sequential *connections*.

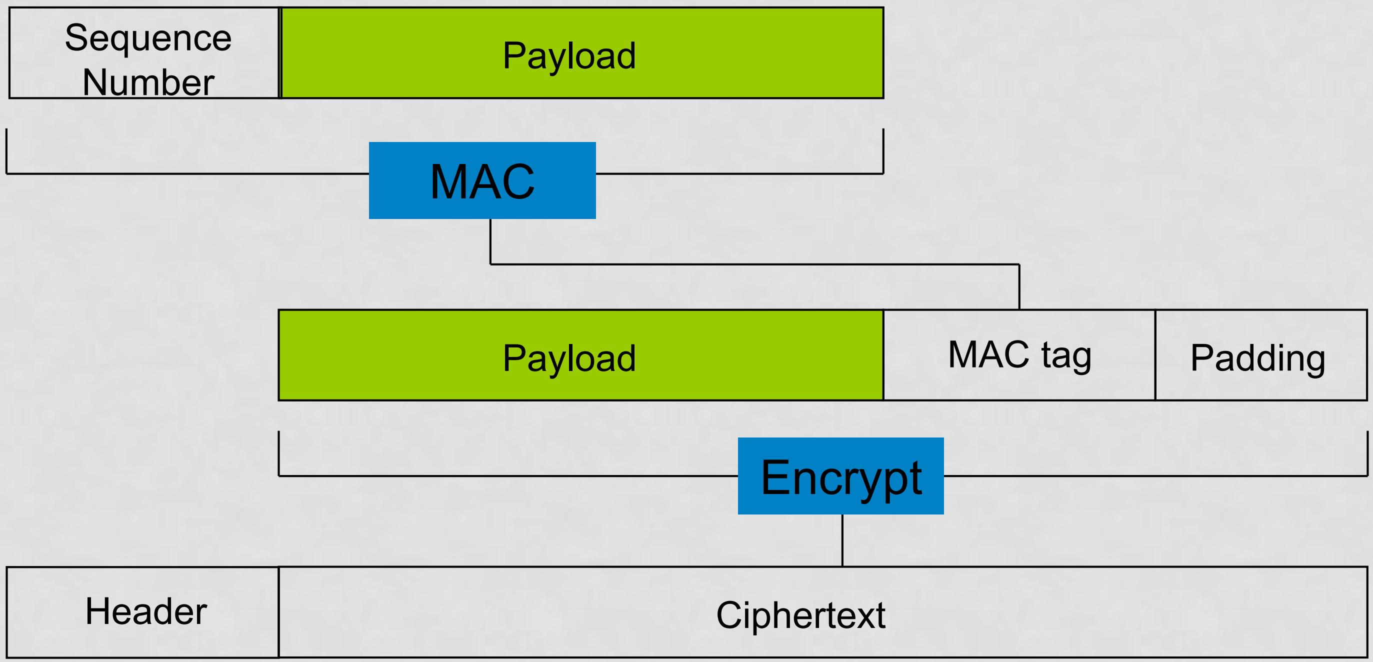
- Connection concept:
  - Keys for multiple connections derived from master secret created during single run of a session.
  - Avoids repeated use of expensive Handshake Protocol.

SSL/TLS Record Protocol

SSL/TLS Record Protocol provides:

- Data origin authentication and integrity.
  - MAC algorithm.
  - Algorithms supported in TLS1.2 are:
    - NULL, HMAC-MD5, HMAC-SHA1, HMAC-SHA256.
- Confidentiality.
  - Bulk encryption using symmetric algorithm.
  - Algorithms supported in TLS1.2 are:
    - NULL algorithm.
    - 3DES, AES-128, AES-256 block ciphers, all in CBC mode.
    - RC4-128 stream cipher.

SSL/TLS Record Protocol (Simplified)



SSL/TLS Handshake Protocol

- SSL/TLS consumes symmetric keys:
  - MAC and encryption algorithms at Record Layer.
  - Different keys in each direction.
- SSL/TLS also needs initialization vectors (IVs) for some encryption algorithms.
- These keys and IVs are established by the Handshake Protocol and subsequent key derivation.
- SSL/TLS Handshake Protocol is a complex protocol with many options.

SSL/TLS Handshake Protocol Security Goals

- Entity authentication of participating parties.
  - Participants are called ‘client’ and ‘server’.
    - Reflects typical usage in e-commerce.
  - Server nearly always authenticated, client more rarely.

- Establishment of a fresh, shared secret.
  - Shared secret used to derive further keys.
  - For confidentiality and authentication/integrity in SSL Record Protocol.

- Secure negotiation of all cryptographic parameters.
  - Encryption and hash algorithms.
  - Authentication and key establishment methods.

SSL/TLS Handshake Protocol – Key Exchange

- SSL/TLS supports several key establishment mechanisms.
- Method used is negotiated during the Handshake Protocol itself.
- Most common is RSA encryption.
  - Client chooses `pre_master_secret`, encrypts using public RSA key of server, sends to server.
- Can also create `pre_master_secret` from:
  - Diffie-Hellman
    - Server and Client exchange Diffie-Hellman components.

SSL/TLS Handshake Protocol – Entity Authentication

- SSL/TLS supports several different entity authentication mechanisms for clients and servers.
- Method used is negotiated along with key exchange method during the Handshake Protocol itself.
- Most common server authentication method is based on RSA.
  - Ability of server to decrypt `pre_master_secret` using its private key and then generate correct MAC in `finished` message using key derived from `pre_master_secret` authenticates server to client.

SSL/TLS Handshake Protocol Run

- An illustrative protocol run follows.

- We choose the most common use of SSL/TLS.
  - No client authentication.
  - Client sends `pre_master_secret` encrypted under Server's RSA public key
  - Server public key obtained from server certificate.
  - Server authenticated by ability to decrypt to obtain `pre_master_secret`, and construct correct `finished` message.

- Other protocol runs are similar.

SSL/TLS Handshake Protocol Run

M1: C → S: **`ClientHello`**

- Client initiates connection.
- Sends client version number.
- Sends `ClientNonce`.
- Offers list of ciphersuites.
  - Key exchange and authentication options, encryption algorithms, hash functions.
  - E.g. TLS_RSA_WITH_AES_256_CBC_SHA256.

SSL/TLS Handshake Protocol Run

M2: S → C: **`ServerHello,`** `ServerCertChain`

- Sends server version number.
- Sends `ServerNonce`.
- Selects single ciphersuite from list offered by client.
  - E.g. TLS_RSA_WITH_AES_256_CBC_SHA256.

SSL/TLS Handshake Protocol Run

M2: S → C: `ServerHello`, **`ServerCertChain`**

- Sends `ServerCertChain` message.
  - Allows client to validate server's public key back to acceptable root of trust.

SSL/TLS Handshake Protocol Run

M3: C → S: **`ClientKeyExchange`**, `ChangeCipherSpec`, `ClientFinished`

- `ClientKeyExchange` contains encryption of `pre_master_secret` under server's RSA public key.

SSL/TLS Handshake Protocol Run

M3: C → S: `ClientKeyExchange`, **`ChangeCipherSpec`**, **`ClientFinished`**

- `ChangeCipherSpec` indicates that client is now switching to use of ciphersuite agreed for this session.
  - Sent using SSL/TLS Change Cipher Spec. Protocol.
  - Technically, an upper layer protocol.
- Finally, `ClientFinished` message:
  - Computed as PRF applied to hash of all messages sent so far (by both sides).
  - Key for PRF is `master_secret`.
  - Provides protection of ciphersuite negotiation.

SSL/TLS Handshake Protocol Run

M4: S → C: **`ChangeCipherSpec, ServerFinished`**

- `ChangeCipherSpec` indicates that server is now switching to ciphersuite agreed for this session.
- Finally, `ServerFinished` message.
  - Computed as PRF applied to hash of all messages sent so far (by both sides).
  - Key for PRF is `master_secret`.
  - Server can only compute PRF if it can decrypt `ClientKeyExchange` in M3 to get `pre_master_secret` and then derive `master_secret`.
  - Provides server authentication and protection of ciphersuite negotiation.

SSL/TLS Handshake Protocol Run

Summary:

M1: C → S: `ClientHello`

M2: S → C: `ServerHello, ServerCertChain`

M3: C → S: `ClientKeyExchange, ChangeCipherSpec, ClientFinished`

M4: S → C: `ChangeCipherSpec, ServerFinished`

Next Lecture

- Wireless Network Security
- Reading for next lecture:
  - Andreson's book – section 21.4.5.2