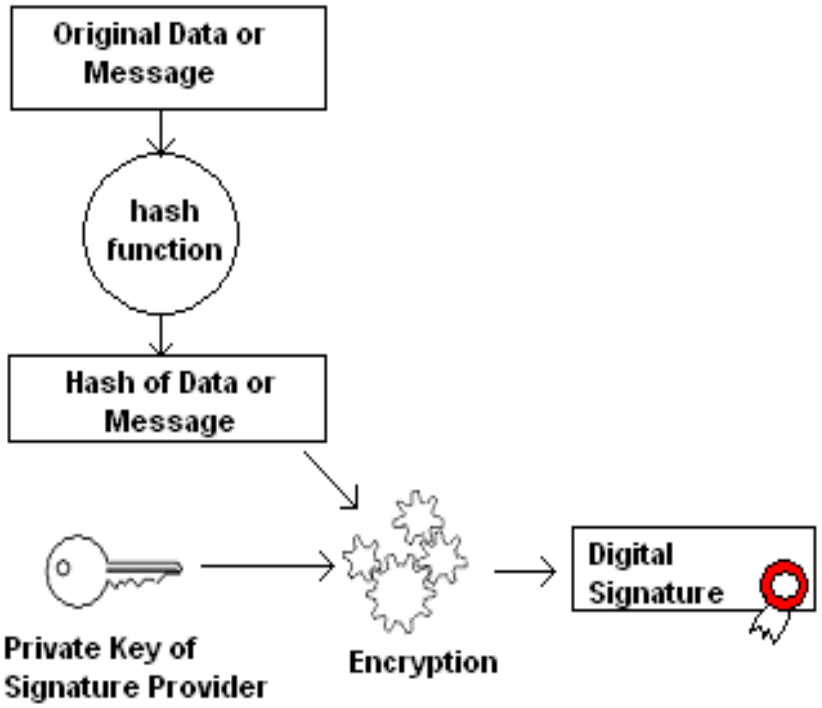
Admin

- Hope you enjoyed Quiz 1. Quiz 2 will have more calculation questions.
- Assignments will be explained today
- Try to form your group quickly (can partner with people from other groups)
- There will be a makeup lecture and tutorial next week

Web security

- Still remember digital signature?



Web Security Considerations

- The WEB is easily accessible worldwide

  more vulnerability
- It's not uni-directional like emails, it involves client and server.
- May trigger to execute software.
- A Web server can be exploited as a launching pad into a corporation's entire computer complex.

Web Security Approaches: Network layer

In the future weeks, you will study **IPSec**:

- Provides a general purpose solution.
- Transparent to end users and applications.

However, today, Web Security Approaches: Just above TCP

- Implement Just above Transport layer
- Provides a general purpose solution.
- SSL can also be embedded in applications. (Explorer browsers are equipped with SSL.)

A question in the future: if SSL /TLS is above Transport layer, why we say IPsec is on the network layer?

Web Security Approaches: application level

Application Level:

- Security services are embedded within an application.
- Security service can be tailored for specific needs of an application.
- Example: Secure Electronic Transaction (**SET**). **Your project!**

Secure Socket Layer (SSL)

- **Serving three security goals**
  1. Entity Authentication.
  2. Confidentiality.
  3. Message integrity.
- Provides secure key exchange between a browser (client) and server.
- Provides security parameters negotiation.

SSL Architecture

- SSL runs on the top of TCP to provide reliable and secure end-to-end service

The question is: is SSL responsible for re-arrange the packets if they arrive in different order?

- Consists of two layers (shown in next Slide).

SSL Architecture

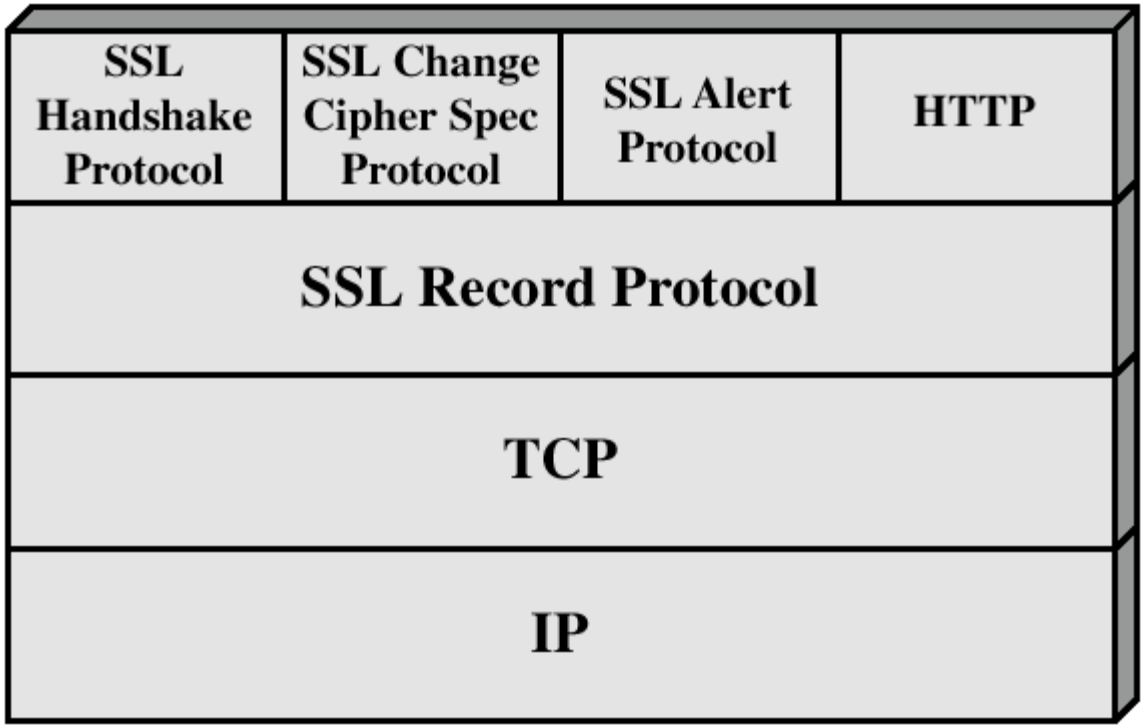


Figure 7.2 SSL Protocol Stack

SSL Record Protocol

- Provides two services for SSL connections:

1. Confidentiality:

   A shared secret key used for conventional encryption of SSL payload.

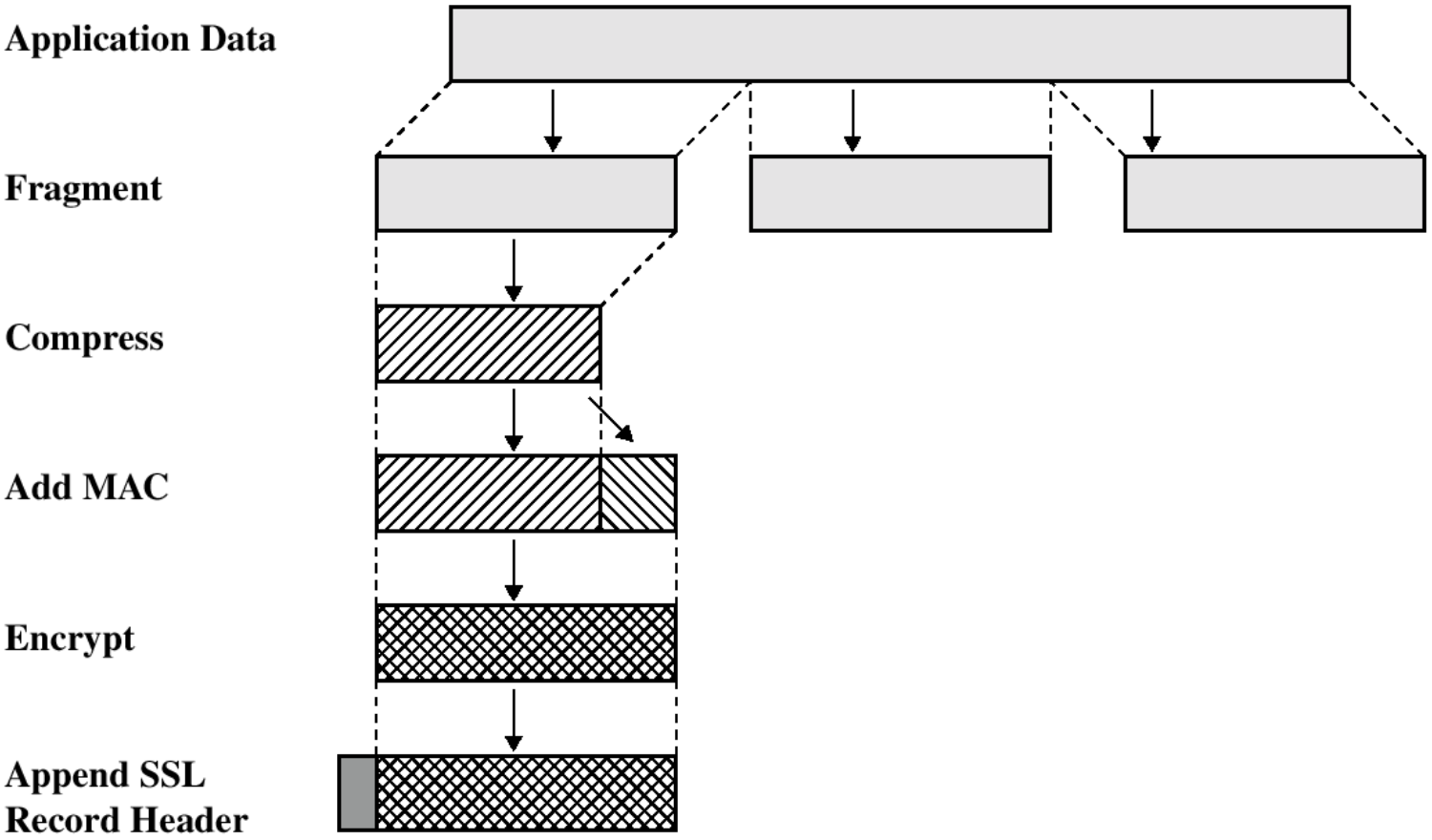
2. Message Integrity:

   A shared secret key is used to construct a message authentication code (MAC)

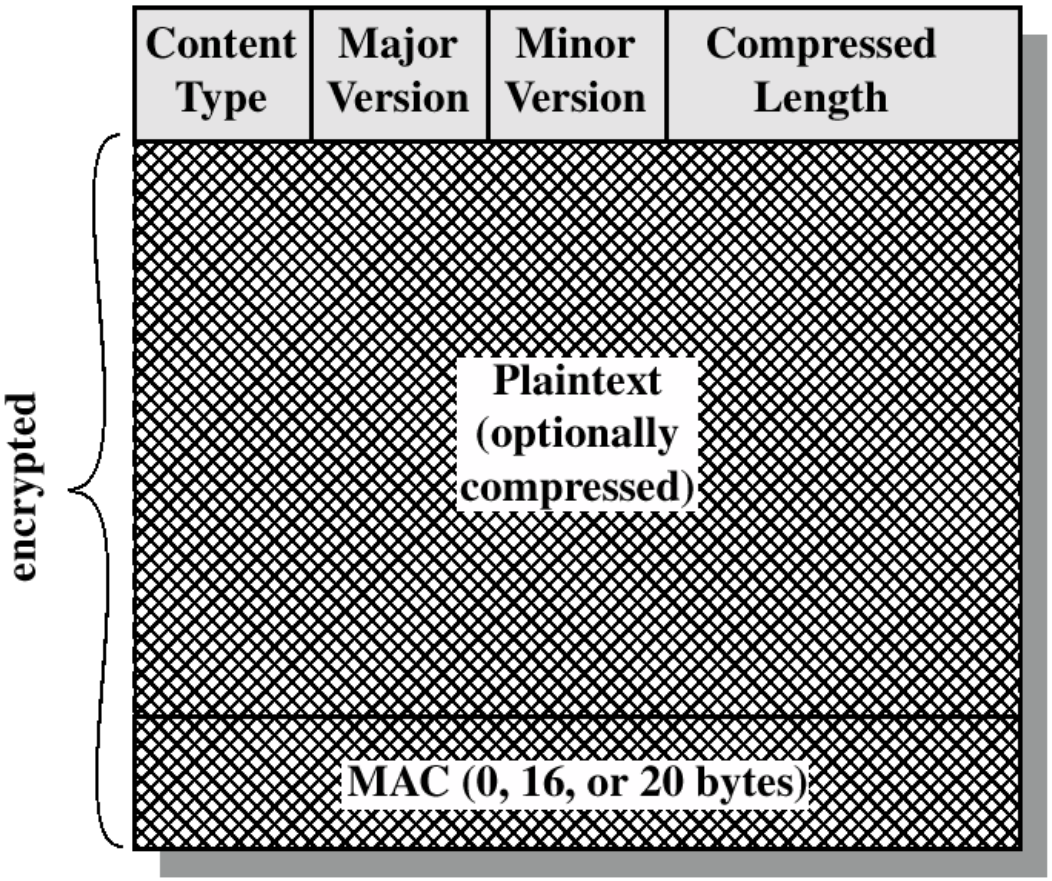
SSL Record Protocol…

- Record protocol takes an application message and performs the following operations:
- Fragmentation
- Compression
- Add a MAC (a shared secret key is used)
- Encryption (symmetric encryption)
- Appends an SSL record header.

SSL Record Protocol Operation



SSL Record Format



SSL Record Header

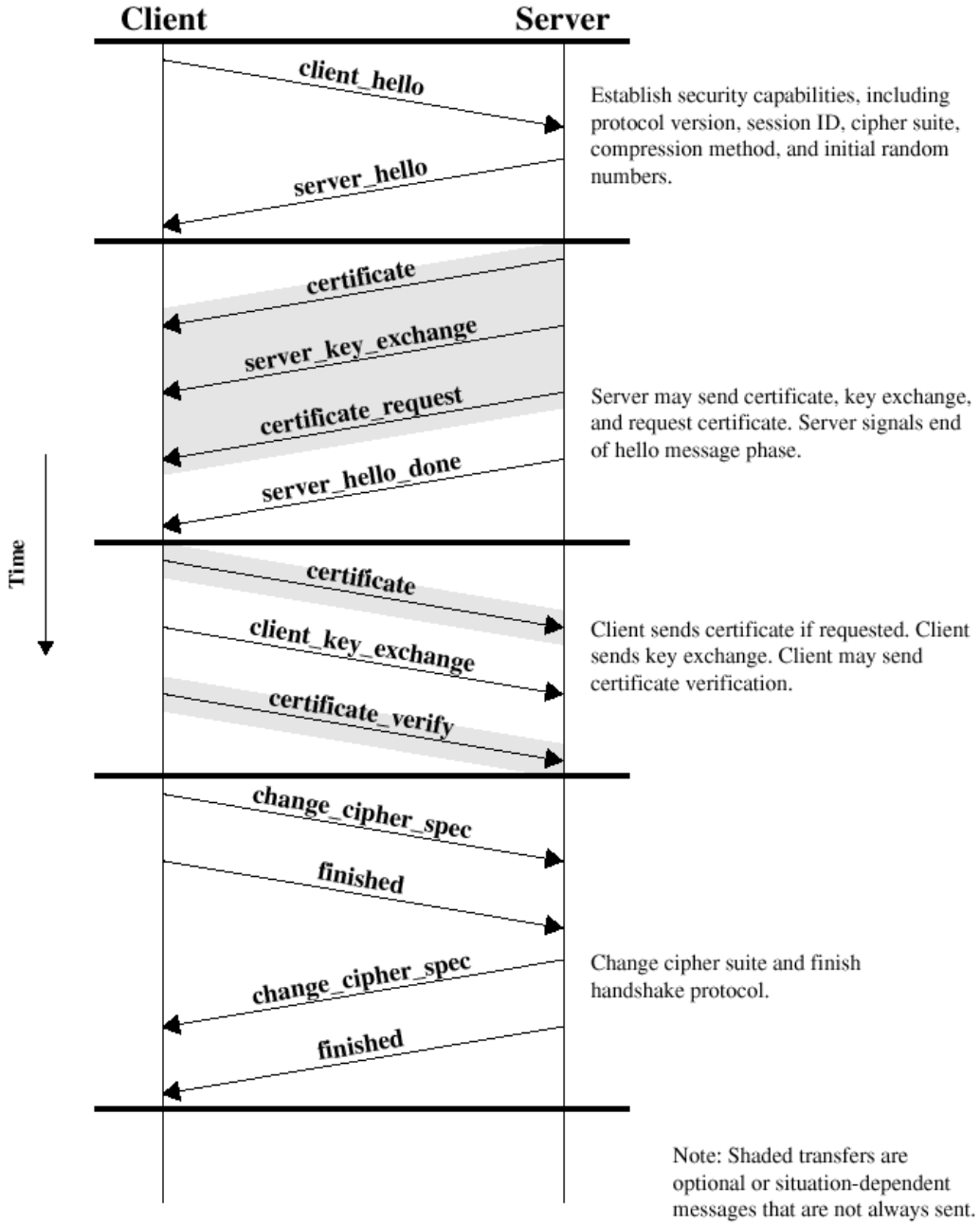
- Content Type (8): Used by higher layers to process the enclosed fragment.
- Major Version (8): Indicates the major version of SSL used.
- Minor Version (8): Indicates the minor version of SSL used.
- Compressed length (16): The length of fragment in bytes.

Handshake Protocol

- The most complex part of SSL.
- Allows the server and client to

  - authenticate each other.

  - negotiate encryption, MAC algorithm and cryptographic keys.
- Used before any application data are transmitted.


Client
Server
client_hello
server_hello
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random numbers.
certificate
server_key_exchange
certificate_request
server_hello_done
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.
Time
certificate
client_key_exchange
certificate_verify
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.
change_cipher_spec
finished
change_cipher_spec
finished
Change cipher suite and finish handshake protocol.
Note: Shaded transfers are optional or situation-dependent messages that are not always sent.

1. 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

1. 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

Change Cipher Spec protocol

- consists of a single message to tell other party in the SSL/TLS session, who is also known is the peer, that the sender wants to change to a new set of keys.
- The key is computed from the information exchanged by the Handshake sub-protocol.

1. Hello and Negotiate Parameters…

- After responding to the hello message, the server sends the client its digital certificate.

You should all know by now what client does with this certificate.

A trusted CA signed this certificate.

- The client uses the trusted CA's public key to decrypt the certificate and obtains server's public key and verifies the server.

2. Key Agreement and Exchange

- 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 keys used for cryptographic purpose

2. Key Agreement and Exchange…

Generation of six shared secret keys:

- Random values exchanged.
- Pre-master secret.
- Pseudo-random function generator.

Example:

PRF(pre-master secret, random1+ random2)

Computed repeatedly.

3. Authentication

The client authenticates the server:

- The clients sends the server a message that is encrypted with the generated secret keys.

  called the “finished handshake” message

- The server responds with its own encrypted finished handshake message.

  The clients is now convinced that it is communicating with right server.

pre-master secret could only be decrypted with the server’s private key.

4. 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.

Transport Layer Security (TLS)

- The same record format as the SSL record format.
- Defined in RFC 2246.
- Similar to SSLv3.
- Differences:
  - version number

  For current version of TLS, the major version is 3 and minor version is 1.
  - message authentication code

  TLS differs in actual algorithm and scope of the MAC calculation.

Transport Layer Security (TLS)

* TLS uses HMAC algorithm.

(difference is how padding bits are used.)

- TLS also covers the field ”TLSCompressed.version” field in MAC calculation.

– pseudorandom function

TLS makes use of a different function.

(objective is to expand secret into blocks of data for purpose of key generation.)

Transport Layer Security (TLS)

- alert codes
- TLS does not support ”no_certificate”.
- In addition, TLS supports some additional alerts.

– cipher suites

– TLS does not support ”Fortezza” method of key exchange.

– TLS does not support ”Fortezza” method of encryption.

Transport Layer Security (TLS)

- client certificate types
- TLS does not support ”Fortezza”.
- certificate_verify and finished message
- In TLS, for certificate_cerify message,

MD5 and SHA-1 hashes are calculated only over handshake_messages.

(In SSL, hash calculation also includes the master secret and pads.)

Transport Layer Security (TLS)

For finished_message, the calculation of hash is based on a different function.

- cryptographic computations
- Master secret computation in TLS uses different computation.

(uses the same parameters as in SSL)

Padding

Can be of any size (<=255 bytes) so that the total length is a multiple of cipher's block length.

Secure Electronic Transactions

- **Put all your studies in action!**
- An open encryption and security specification.
- Designed to protect credit card transaction on the Internet.
- Companies involved:
  - MasterCard, Visa, IBM, Microsoft, Netscape, RSA, Terisa and Verisign
- Not a payment system.
- Set of security protocols and formats

(enables users to employ existing Credit card (CC) payment infrastructure securely in an open environment).

SET Services

- Provides three services:

1. Provides a secure communication channel among all parties involved in a transaction.
2. Provides trust by the use of X.509v3 digital certificates.
3. Ensures privacy: information is only available to involved parties.

SET Overview

- Key Features of SET:
  - Confidentiality of information
  - Integrity of data
  - Cardholder account authentication
  - Merchant authentication

The settings…

- The customer has a certificate. (obviously contains customer's public key)
- Merchants have their own certificates. (Two certificates: one for signing messages and the other for key exchange.)
- The order and payment are sent together. Payment information is encrypted in such a way that it can not be read by the merchant.

Dual Signature

- Objective: to link two messages that are intended for two different recipients.
- Customer wants to send:

1. Order Information (OI) to merchant.
2. Payment information (PI) to bank.

Customer wants to link these two items and also wants to keep them separate.

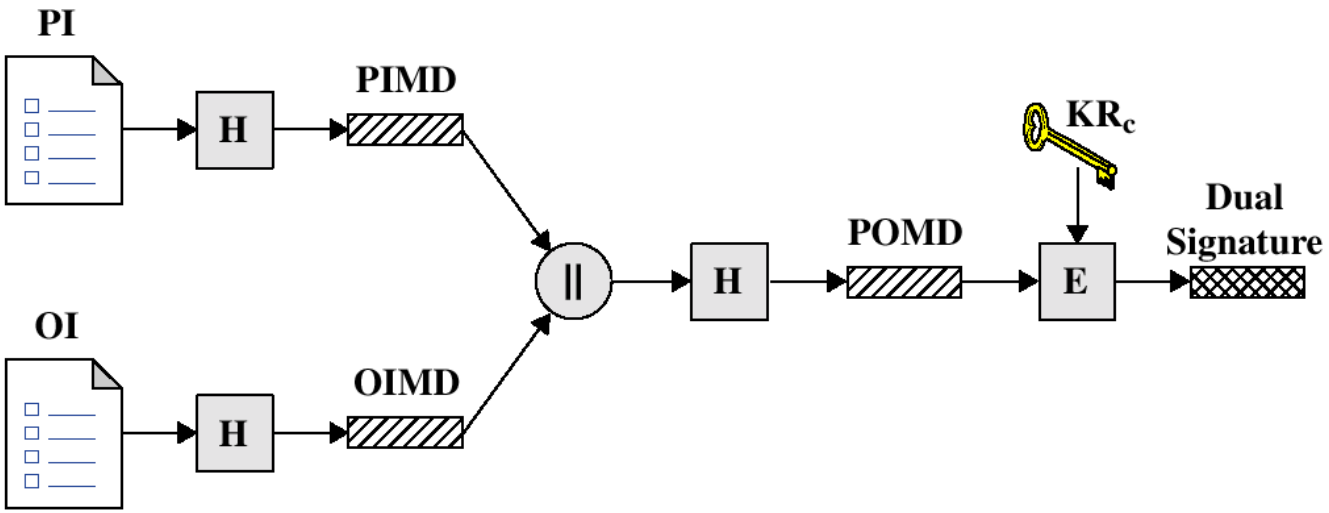
Generation of Dual Sign.

- Customer takes the hash (SHA-1) of PI.
- Customer takes the hash of OI.
- Concatenates these two and takes hash of the result.
- Customer signs the final hash with his private key.

$$DS = EKRc[H(H(PI)||H(OI))]$$

Dual Signature

$$DS = E_{KR_c}[H(H(PI) \| \mathrm{H(OI)})]$$



PI = Payment Information
OI = Order Information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment Order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

Dual Signature

- Merchant has DS, OI, and PIMD.
- Merchant computers H(PIMD||H(OI)).
- Merchant decrypts DS using customer's public key.
- If both these items are equal, the merchant has verified the DS.

PI is not send to the Merchant

Dual Signature

- The bank has DS, PI, and OIMD.
- The bank computers H(H(PI)||OIMD).
- The bank decrypts DS using customer's public key.
- If both these items are equal, the merchant has verified the DS.

OI is NOT send to the bank