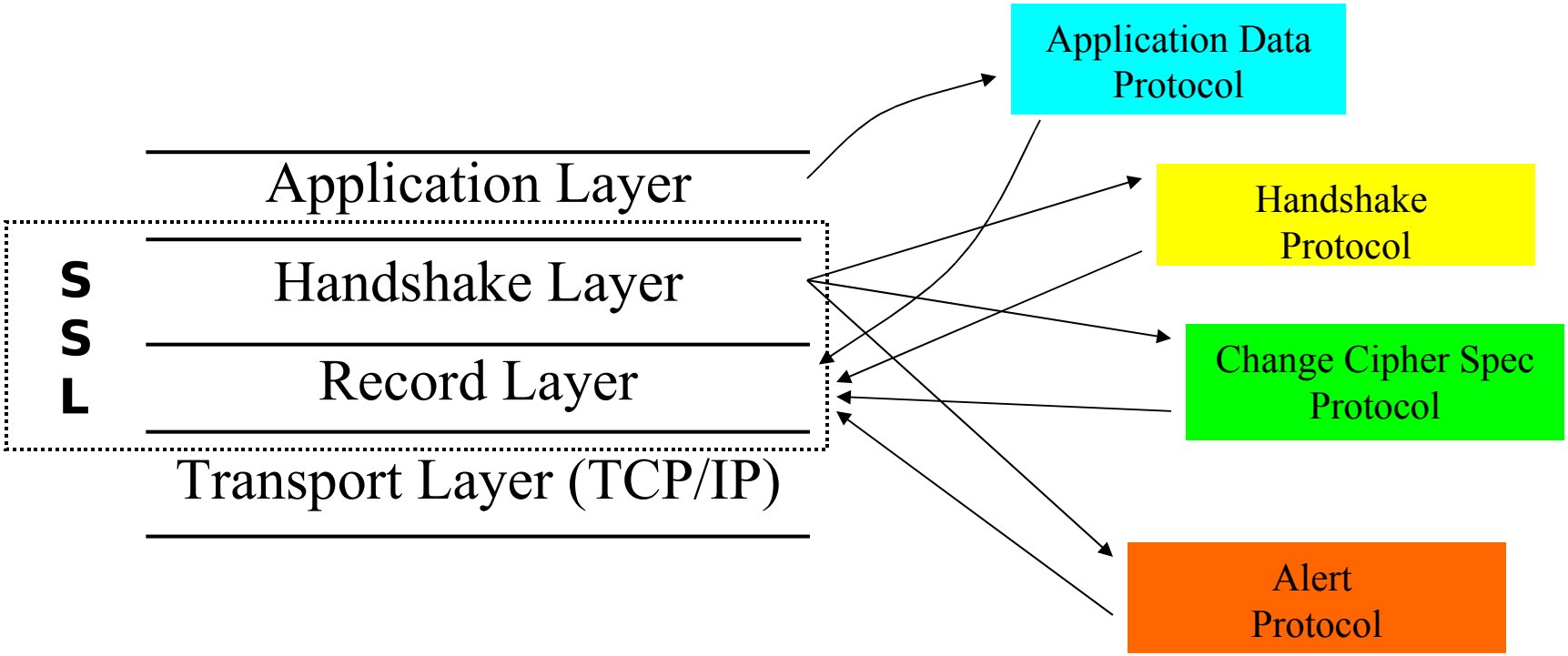
SSL3.0 / TLS1.0

Secure Communication
over
Insecure Line

What does “Secure” Mean?

+ Confidentiality
  - Prevent eavesdropping
+ Authenticity
  - Communication line cannot be overtaken
+ Integrity
  - Messages cannot be modified during transport
– Non-reputiation
  - “you can’t prove I said *that*!”

Layered Architecture



SSL is a *Statefull* Protocol

- SSL handles multiple simultaneous sessions
- Each session handles multiple connections

Session State
- Session ID
- Peer certificate
- Compression method
- Cipher Spec.
- *Master Secret*
- Is resumable (flag)

Connection State
- Random data
- MAC Secrets
- Transport Keys
- IVs (for CBC block ciphers)
- Sequence numbers

TLS' State Model

Session State (?)

- Session ID
- Peer certificate
- Is resumable (flag)

Connection State

- Master Secret
- Random data
- CipherSpec
- Compression Method
- MAC Secrets
- Transport Keys
- IVs (for CBC block ciphers)
- Sequence numbers

State Structure

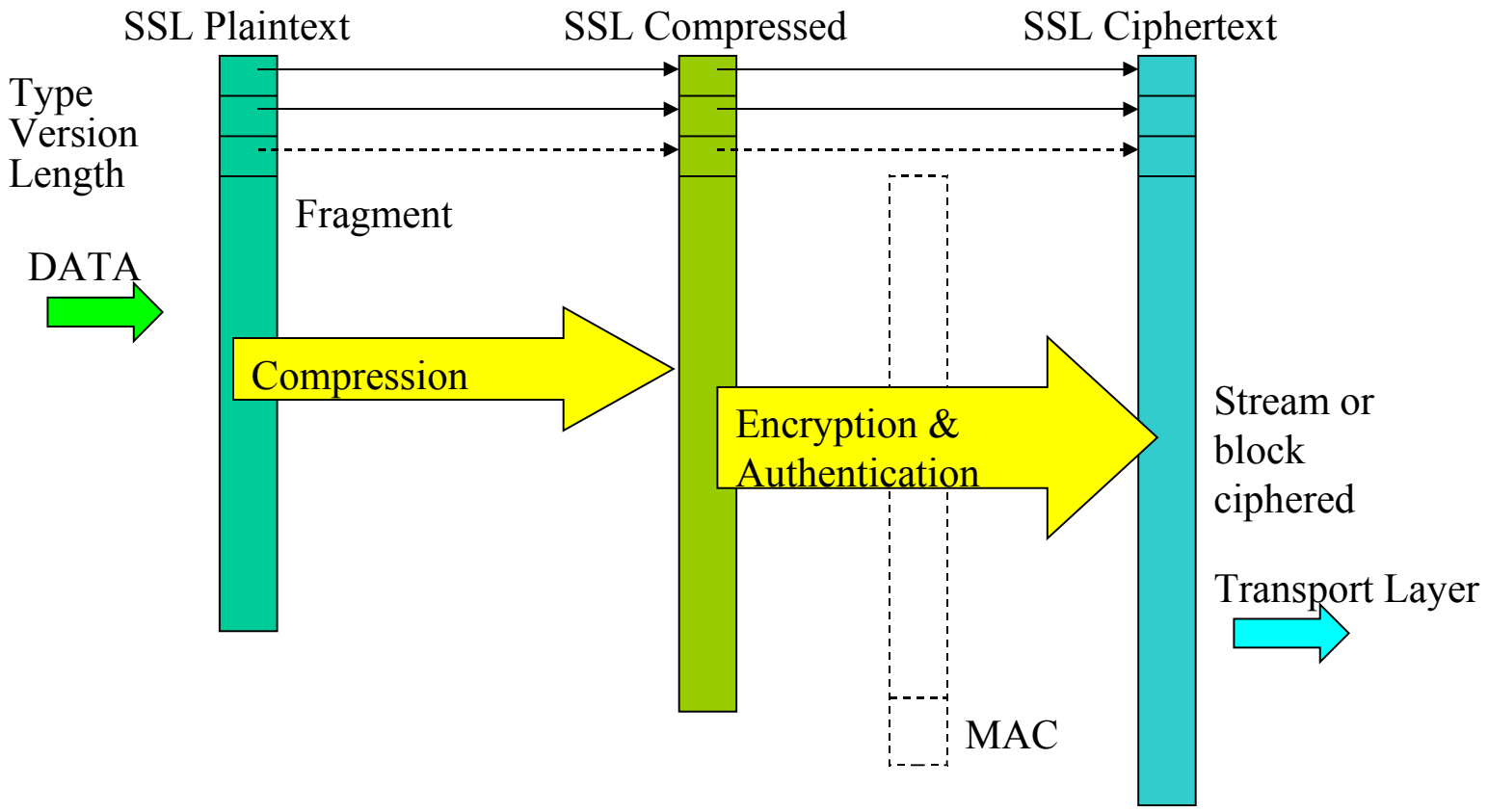
Client

	Read	Write
Pending	RSA, 3DES, SHA, Keys, …	RSA, 3DES, SHA, Keys, …
Current	Null, Null, Null, …	Null, Null, Null, …

Server

	Read	Write
Pending	RSA, 3DES, SHA, Keys, …	RSA, 3DES, SHA, Keys, …
Current	Null, Null, Null, …	Null, Null, Null, …

Record Layer

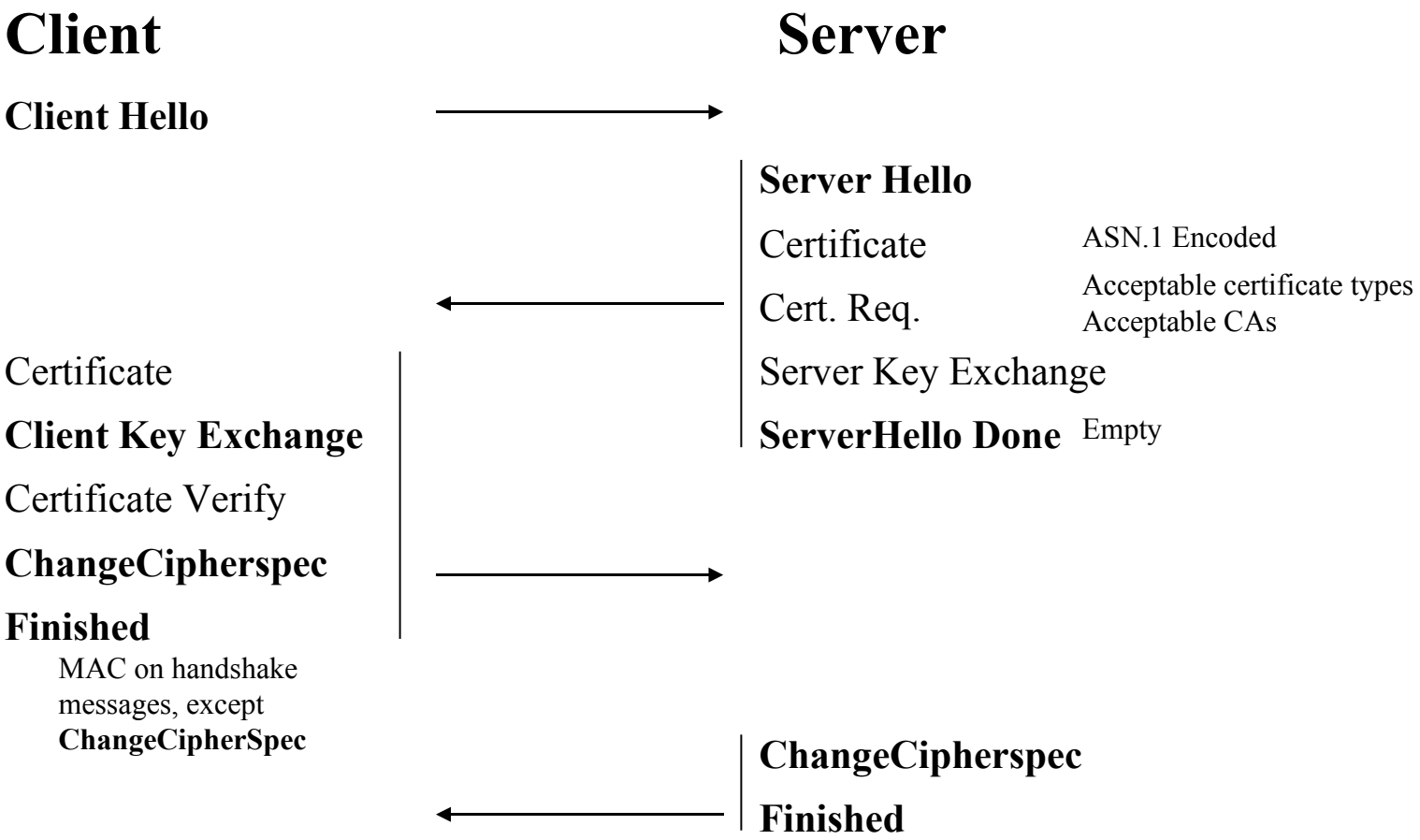


MAC Computation

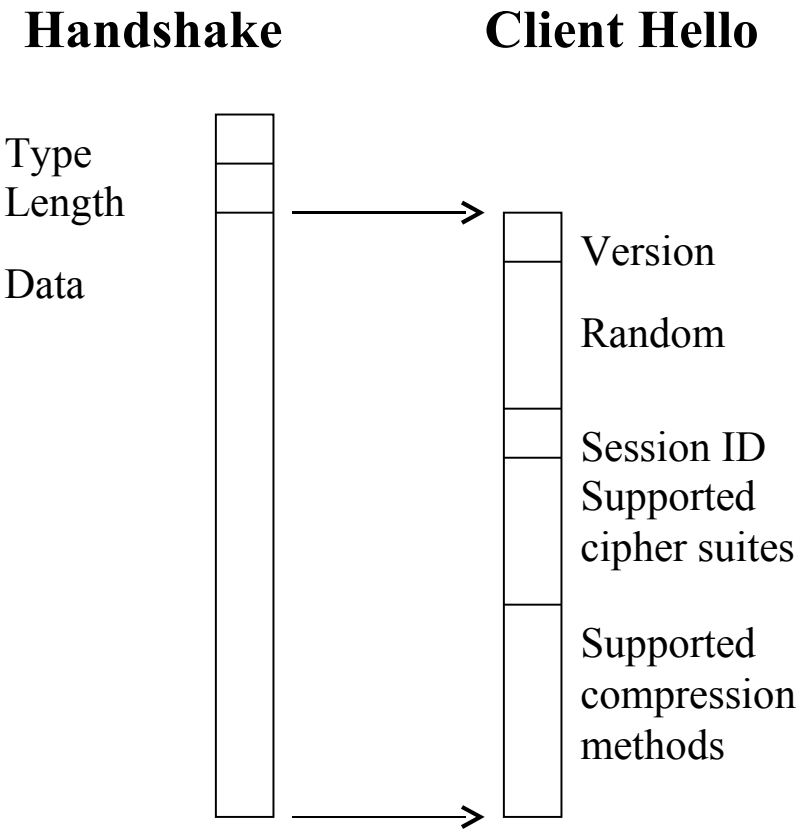
MAC = hash(MAC_secret + pad_2 +
hash(MAC_secret + pad_1 + seq_num + length + content))

hash = SHA-1 or MD5
pad_1 = 0x36 repeated 40 times (SHA-1) of 48 times (MD5)
pad_2 = 0x5C repeated as above

Handshake Protocol



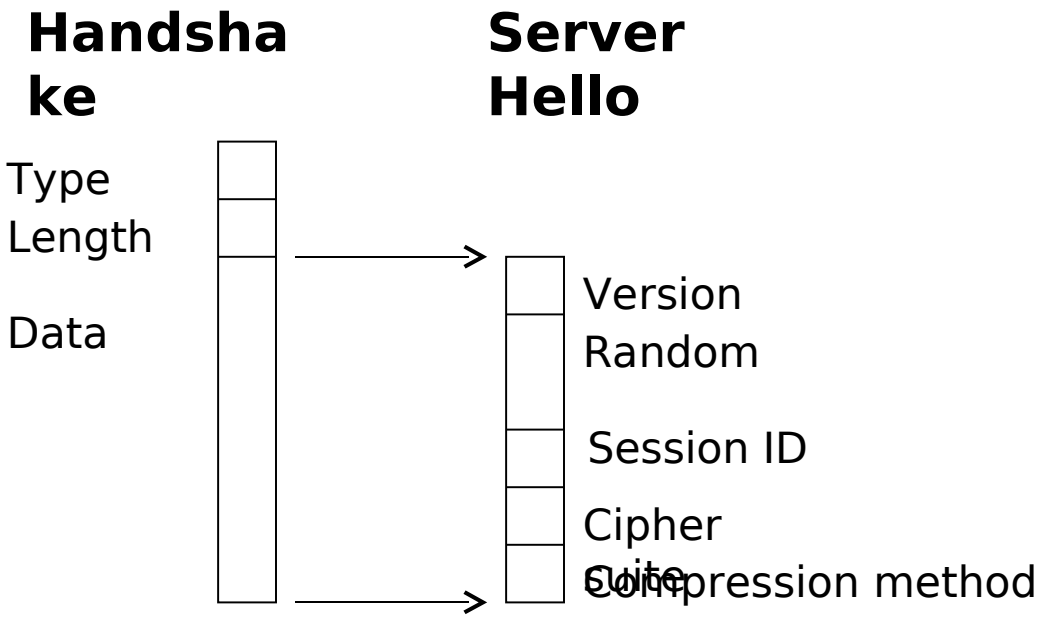
Client Hello



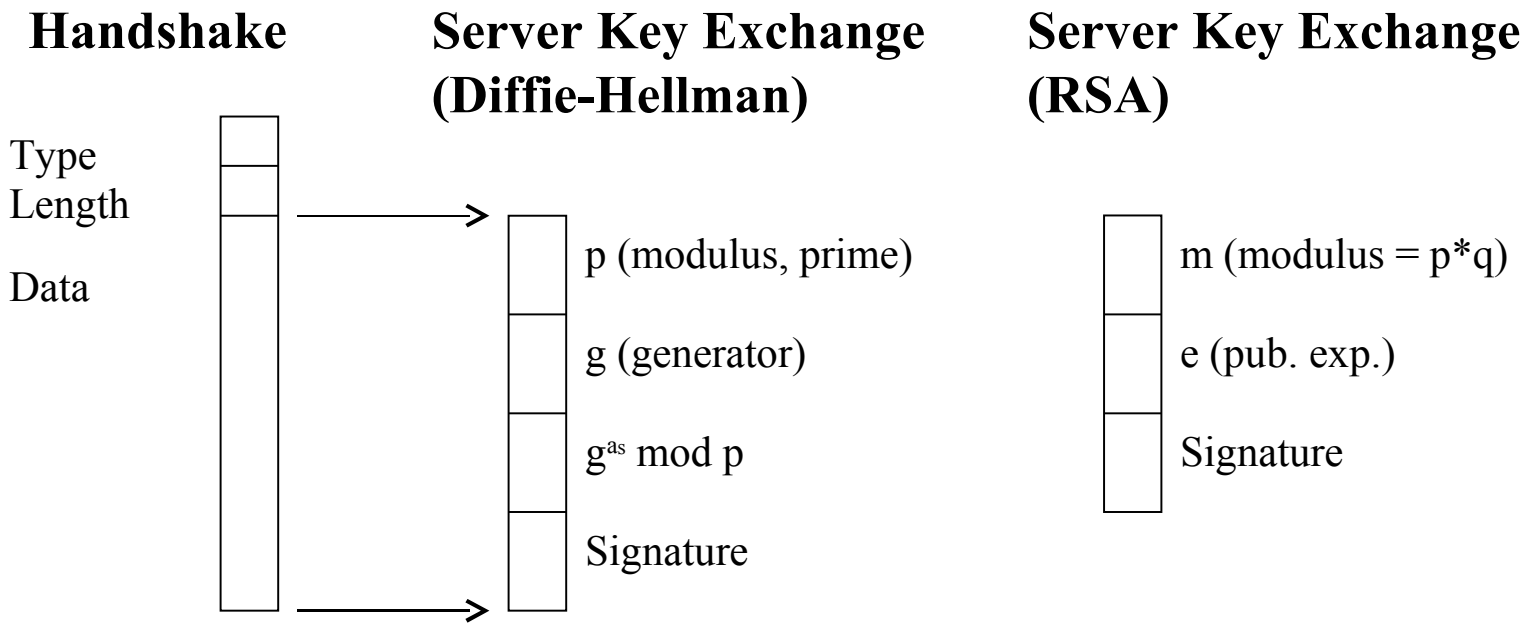
Example Cipher Suites:

SSL_RSA_WITH_DES_CBC_SHA
SSL_RSA_EXPORT_WITH_RC4_40_MD5
SSL_DHE_RSA_WITH_DES_CBC_SHA
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA
…

Server Hello



Server Key Exchange



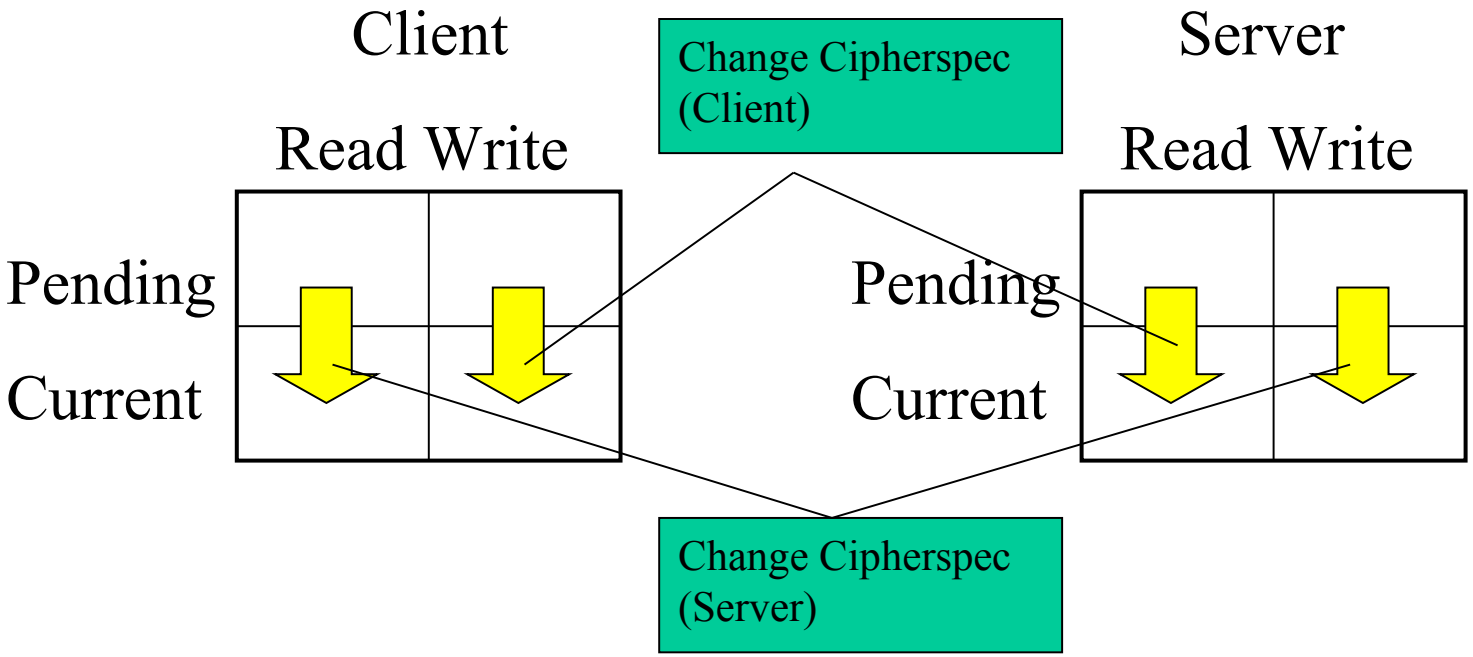
Diffie-Hellman

Client Computes: PreMasterSecret = $(g^{as})^{ac}$ mod p
Client Sends : g^{ac} to server
Server Computes: PreMasterSecret = $(g^{ac})^{as}$ mod p

RSA

Client Computes: y = $PreMasterSecret^{e}$ mod p
Client sends : y to server
Server Computes: PreMasterSecret = y^{d} mod p

Change Cipherspec



Key Generation

ms = MD5(pms + SHA('A' + *pms* + client_random + server_random))
+ MD5(pms + SHA('BB' + *pms* + client_random + server_random))
+ MD5(pms + SHA('CCC' + *pms* + client_random + server_random))

Where

ms = "Master Secret" (48 bytes)
pms = "Pre Master Secret" (48 bytes)

Key material computed by

(Pseudo) random data =
MD5(ms + SHA('A' + *ms* + client_random + server_random))
+ MD5(ms + SHA('BB' + *ms* + client_random + server_random))
+ MD5(ms + SHA('CCC' + *ms* + client_random + server_random))
+ …

Dropping **ChangeCipherSpec**

Attack:

Remove the ChangeCipherSpec message entirely from the communication, forcing communication to proceed with initial cipher spec (null).

SSL defence:

Implementation must check that a **ChangeCipherSpec** message has been received before a **Finished** message is received.

Key Exchange Rollback

Attack:

Make client use a prime modulus for its RSA key exchange by tampering with cipher suite specs in **ClientHello** and **ServerHello.**

SSL defence:

Implementation must check that the number of fields in the **ServerKeyExchange** message matches the chosen cipher suite.

Experts in cryptography

info@cryptomathic.com
www.cryptomathic.com