

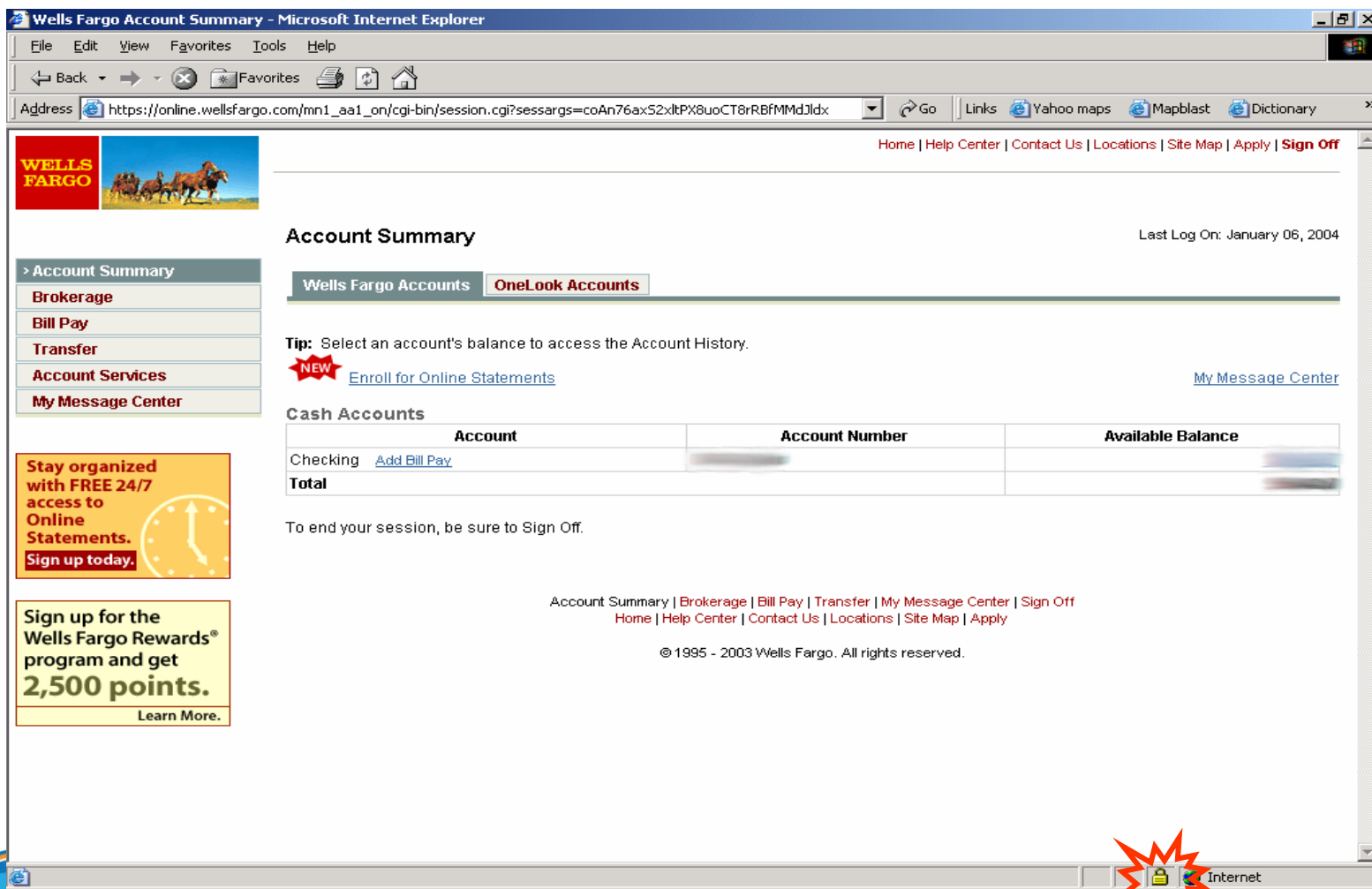
Chủ đề 10: Secured Socket Layer

PGS.TS. Trần Minh Triết



KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

SSL / TLS trong đời sống hằng ngày?



Wells Fargo Account Summary - Microsoft Internet Explorer

Address: https://online.wellsfargo.com/mn1_aa1_on/cgi-bin/session.cgi?sessargs=coAn76ax52xltPX8uoCT8rRBfMMdJldx

Home | Help Center | Contact Us | Locations | Site Map | Apply | Sign Off

Account Summary Last Log On: January 06, 2004

Wells Fargo Accounts | OneLook Accounts

Tip: Select an account's balance to access the Account History.

NEW [Enroll for Online Statements](#) [My Message Center](#)

Cash Accounts

Account	Account Number	Available Balance
Checking Add Bill Pay		
Total		

To end your session, be sure to Sign Off.

Account Summary | Brokerage | Bill Pay | Transfer | My Message Center | Sign Off
Home | Help Center | Contact Us | Locations | Site Map | Apply

© 1995 - 2003 Wells Fargo. All rights reserved.

Stay organized with FREE 24/7 access to Online Statements. Sign up today.

Sign up for the Wells Fargo Rewards® program and get 2,500 points. Learn More.



Sự phát triển của giao thức

- SSL 1.0
 - ▣ Nghiên cứu nội bộ trong Netscape (~đầu 1994?)
 - ▣ “Lost in the mists of time”
- SSL 2.0
 - ▣ Netscape công bố vào tháng 11 năm 1994
 - ▣ Còn tồn tại 1 số vấn đề
- SSL 3.0
 - ▣ Do Netscape và Paul Kocher thiết kế (11/1996)
- TLS 1.0
 - ▣ Chuẩn Internet dựa trên SSL 3.0 (01/1999)
 - ▣ Không hoạt động với SSL 3.0

Một số hạn chế trong SSL 2.0

- Độ dài khóa quá ngắn
 - ▣ Trong các chế độ đã được làm yếu trước khi đưa ra bên ngoài sử dụng, SSL 2.0 đã thu hẹp độ dài khóa để authentication xuống còn 40 bit.
- Tạo MAC yếu
- Có thể bị tấn công tính toàn vẹn thông tin
 - ▣ SSL 2.0 thêm các byte (padding) vào MAC trong các chế độ mã hóa theo khối, nhưng không kiểm soát để xác nhận độ dài padding. Điều này tạo điều kiện để người tấn công có thể xóa 1 số byte ở cuối thông điệp.
- Ciphersuite rollback attack
 - ▣ Người tấn công có thể sửa thông điệp ClientHello để “lừa” Server chọn dùng phiên bản cũ hay giải thuật yếu

Cơ bản về TLS

TLS gồm **hai** giao thức:

- **Giao thức Handshake**

- Sử dụng mã hóa bất đối xứng để thiết lập khóa bí mật dùng chung (shared secret key) giữa client và server

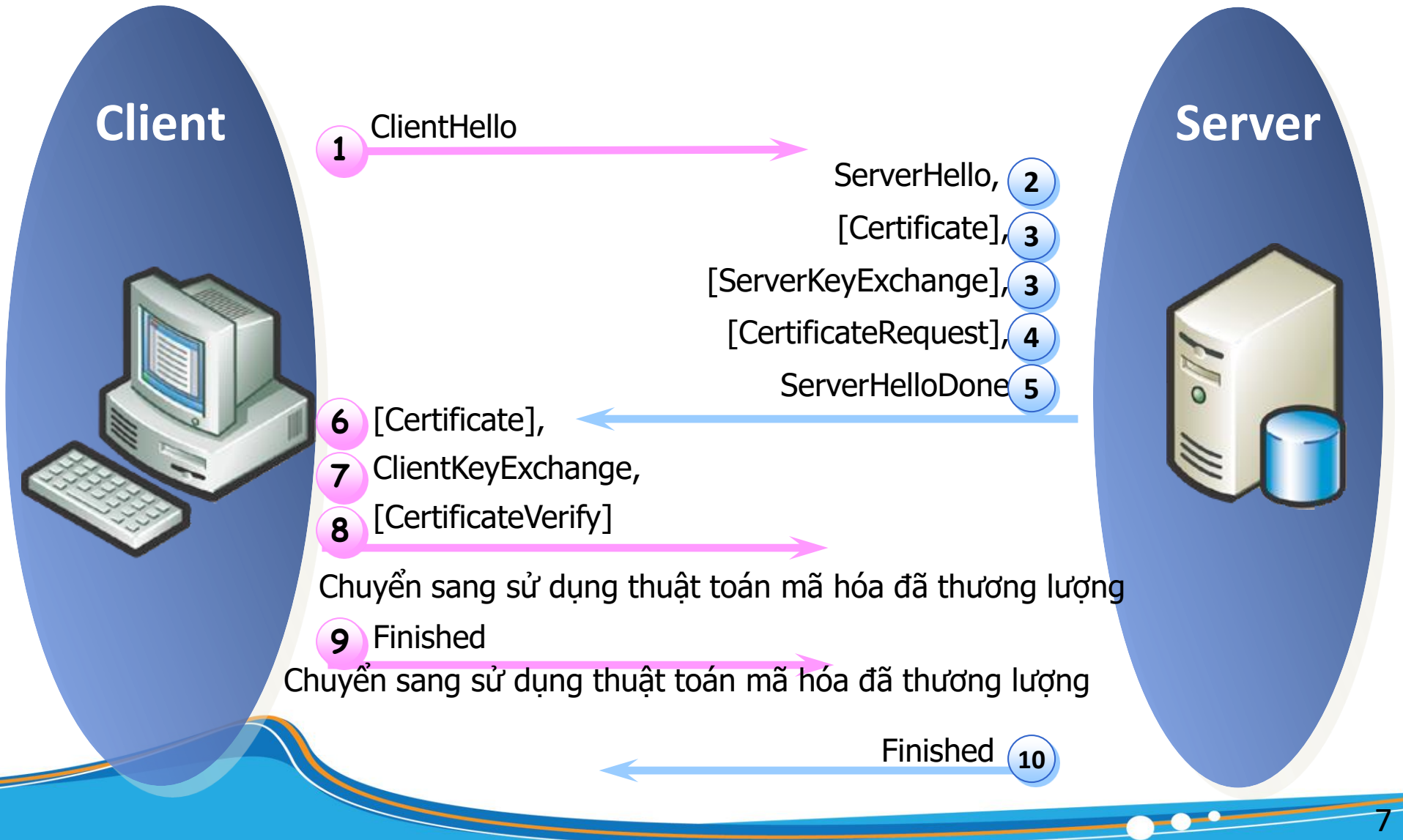
- **Giao thức Record**

- Sử dụng khóa bí mật đã được thiết lập trong giao thức handshake để bảo vệ truyền thông giữa client và server

TLS Handshake Protocol

- ☐ Client và Server
- ☐ Thương lượng về version của giao thức và tập các giải thuật mật mã được chọn sử dụng
 - ☐ Tính tương thích và phối hợp hoạt động giữa các cài đặt khác nhau của giao thức tổng quát
- ☐ Authenticate client và server (không bắt buộc)
 - ☐ Sử dụng chứng nhận điện tử để biết được public key của đối tác và xác nhận định danh của nhau
- ☐ Sử dụng public key để thiết lập thông tin bí mật chung

Cấu trúc giao thức Handshake (tổng quát)

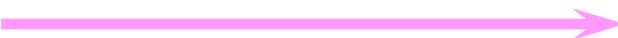


ClientHello

Client



1 ClientHello



Client thông báo (dữ liệu KHÔNG mã hóa):

- Version của Protocol đang dùng
- Các thuật toán mã hóa được hỗ trợ

Server



ClientHello (RFC)

```
struct {  
    TimeStamp    timestamp;  
    ProtocolVersion client_version;  
    Random       random;  
    SessionID    session_id;  
    CipherSuite  cipher_suites;  
    CompressionMethod compression_methods;  
} ClientHello
```

Version cao nhất của protocol được client hỗ trợ

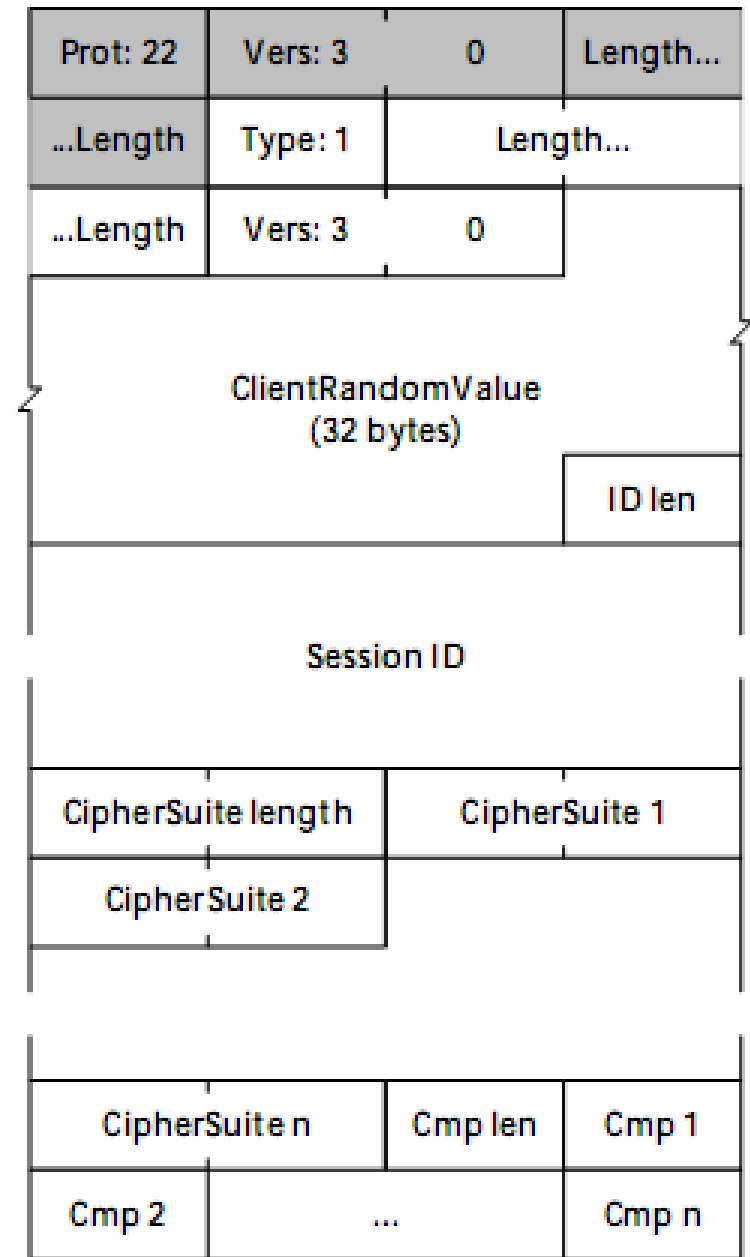
Session id (nếu client muốn resume một session cũ)

Các thuật toán mật mã được client hỗ trợ (ví dụ RSA hay Diffie-Hellman)

ClientHello

- ☐ 4 byte timestamp, 28 byte giá trị ngẫu nhiên
- ☐ session_id:
 - ☐ Khác 0 cho kết nối mới của session đã có
 - ☐ Bằng 0 cho kết nối mới của session mới
- ☐ client_version: giá trị version lớn nhất
- ☐ cipher_suites: danh sách có thứ tự các giải thuật mà client hỗ trợ
- ☐ compression_methods: danh sách có thứ tự các giải thuật nén mà client hỗ trợ

ClientHello



ServerHello

Client



1 $C, \text{Version}_C, \text{suite}_C, N_C$

← ServerHello 2

Server trả lời (dữ liệu KHÔNG mã hóa):

- Version cao nhất được server và client hỗ trợ
- Tập hợp các giải thuật mật mã mạnh nhất được client hỗ trợ

Server



ServerHello

- 32 byte giá trị ngẫu nhiên
- session_id:
 - Giá trị mới hay giá trị cũ dùng lại
- version: $\min\{\text{version}_{\text{client hỗ trợ}}, \text{version}_{\text{tối đa server hỗ trợ}}\}$
- cipher_suite list: danh sách các giải thuật được chọn (chỉ chọn 1 giải thuật trong mỗi loại)
- compression list: chọn lựa duy nhất từ các giải thuật nén mà client hỗ trợ

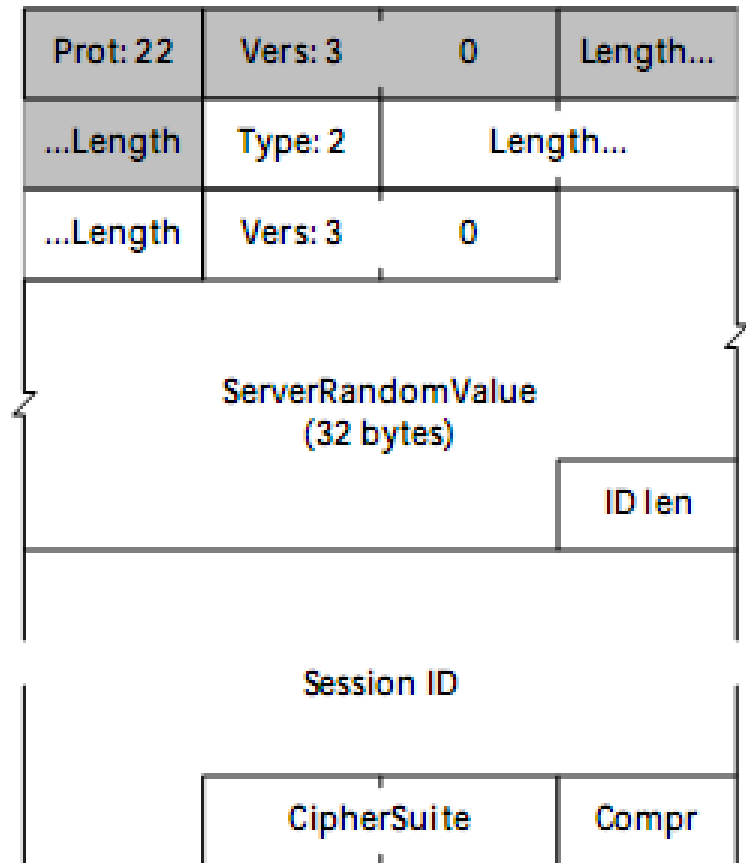
ServerHello

- Phương pháp trao đổi khóa
 - RSA: cần chứng nhận public key của người nhận
 - Fixed DH: cả 2 phía đều phải có chứng nhận public-key
 - Ephemeral DH: cả 2 phía đều cần khóa để ký và chứng nhận public-key
 - Anonymous DH: không xác thực khóa DH, có thể bị tấn công man-in-the-middle
 - Fortezza: rất ít sử dụng

ServerHello

- ☐ CipherAlgorithm: RC4, RC2, DES, 3DES, DES40, IDEA, Fortezza
- ☐ MACAlgorithm: MD5 hay SHA-1
- ☐ CipherType: stream hay block
- ☐ IsExportable: true hay false
- ☐ HashSize: 0, 16 hay 20 byte
- ☐ Key Material: dùng để phát sinh khóa ghi
- ☐ IV Size: kích thước của IV trong CBC

ServerHello

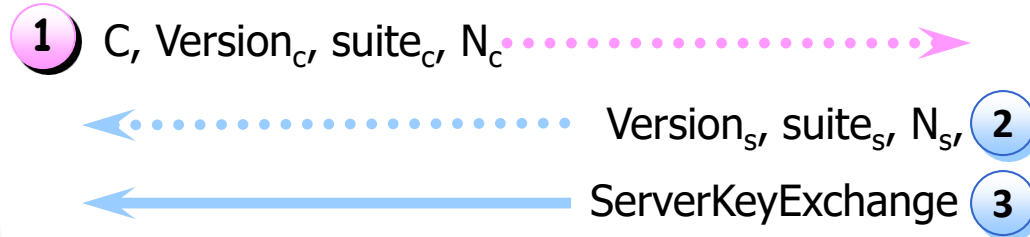


ServerKeyExchange

Client



Server



Server gửi certificate của public-key của mình (RSA hay Diffie-Hellman, tùy vào tập hợp các giải thuật đã chọn)

ServerKeyExchange

- ☐ Không cần dùng với RSA và Fixed DH
- ☐ Cần dùng với Anonymous DH, Ephemeral DH
- ☐ Cần dùng với RSA nếu server chỉ có khóa để ký. Khi đó, server gửi public key (RSA) tạm thời cho client

- ☐ Thông điệp ServerKeyExchange:
 - ☐ Được server ký
 - ☐ Chữ ký trên giá trị hash của:
 - ClientHello.random, ServerHello.random
 - Tham số của Server Key Exchange

ServerKeyExchange

Prot: 22	Vers: 3	0	Length...
...Length	Type: 12	Length...	
...Length	DH p length		DH p ...
...value		DH q length	
DH q value		DH Y _s length	
DH Y _s value			
Signed MD5 hash [if RSA signing] (16 bytes)			
Signed SHA hash [if RSA or DSA signing] (20 bytes)			

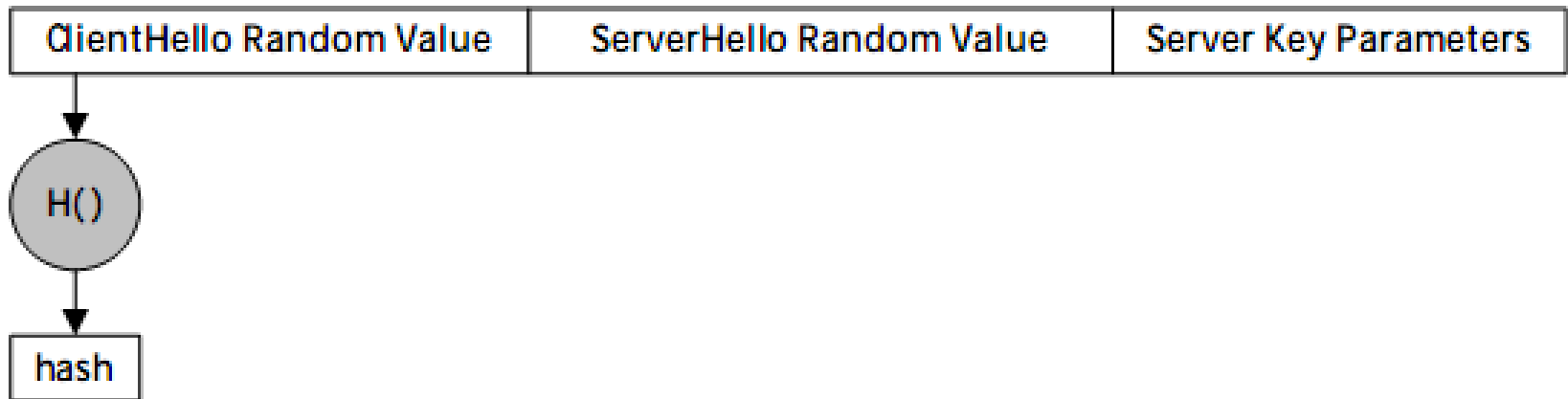
Sử dụng Diffie-Hellman

Prot: 22	Vers: 3	0	Length...
...Length	Type: 12	Length...	
...Length	RSA mod len		RSA ...
... mod value		RSA exp length	
RSA exp value			
Signed MD5 hash [if RSA signing] (16 bytes)			
Signed SHA hash [if RSA or DSA signing] (20 bytes)			

Sử dụng RSA

ServerKeyExchange

Server ký trên thông tin hash của ClientHelloRandomValue, ServerHello Random Value và các thông tin về public key của server

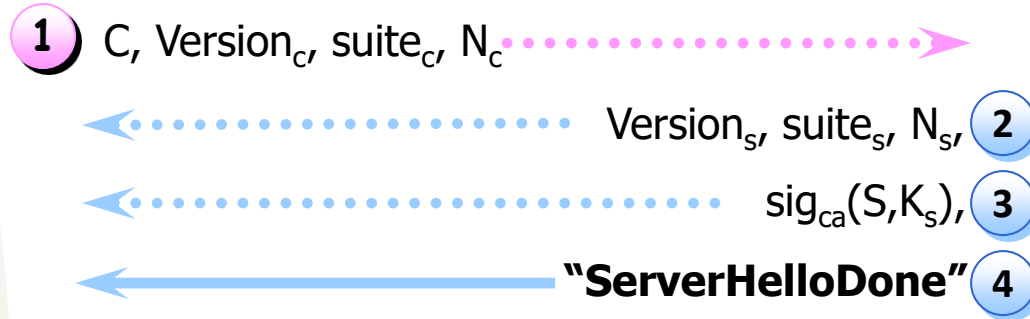


ClientKeyExchange

Client



Server

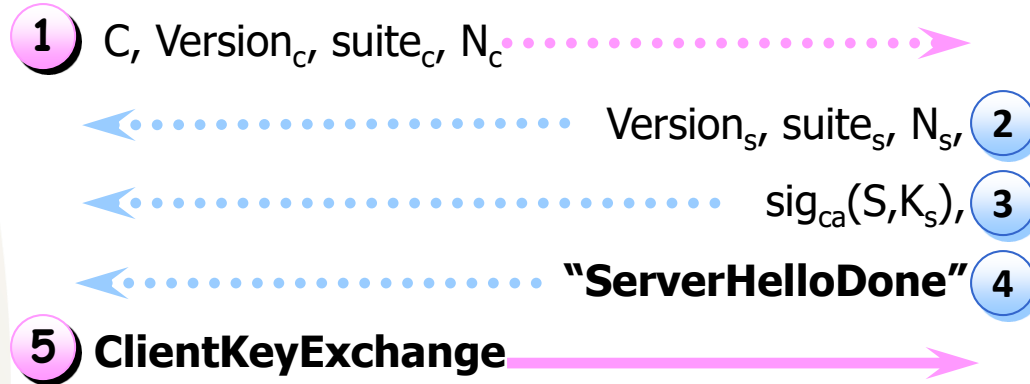


ClientKeyExchange

Client



Server



Client phát sinh khóa bí mật (đã mã hóa bằng public key của server) và gửi cho server

ClientKeyExchange (RFC)

```
struct {  
    select (KeyExchangeAlgorithm) {  
        case rsa: EncryptedPreMasterSecret;  
        case diffie_hellman: ClientDiffieHellmanPublic;  
    } exchange_keys  
} ClientKeyExchange
```

```
struct {  
    ProtocolVersion client_version;  
    opaque random[46];  
} PreMasterSecret
```

ClientKeyExchange (RFC)

Prot: 22	Vers: 3	0	Length...
...Length	Type: 16	Length...	
...Length	DH Y_c length		
DH Y_c value			

Sử dụng Diffie-Hellman

Prot: 22	Vers: 3	0	Length...
...Length	Type: 16	Length...	
...Length	Encrypted Premaster Secret		

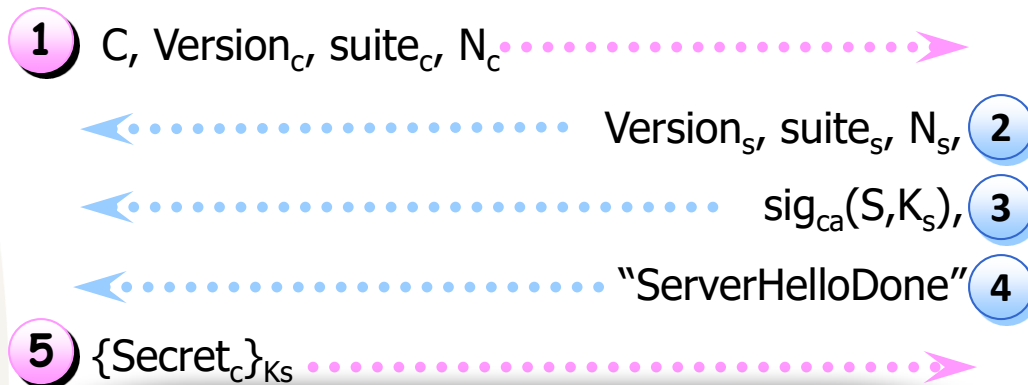
Sử dụng RSA

“Core” SSL

Client



Server

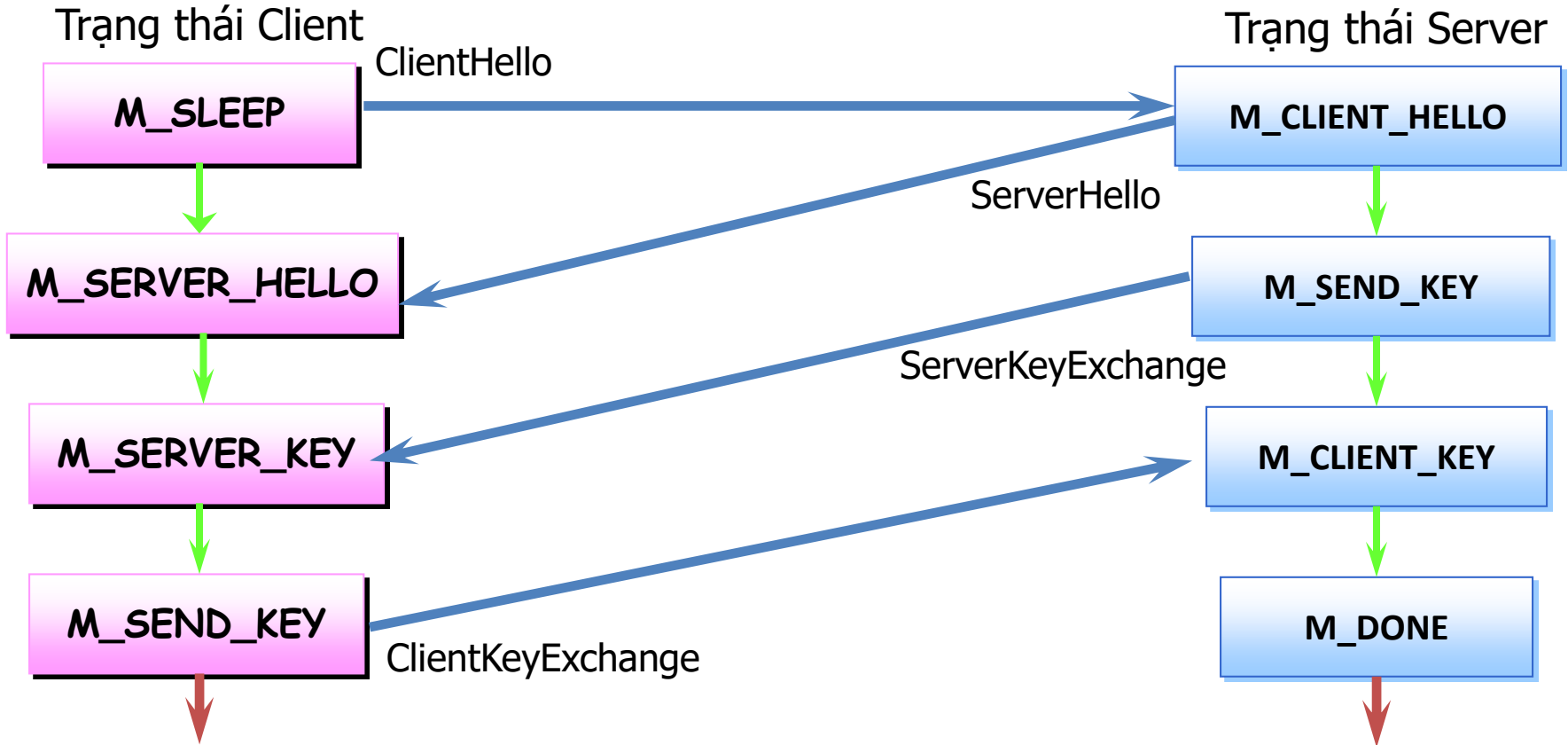


Nếu giao thức chính xác, từ lúc này,
C và S cùng chia sẻ khóa bí mật chung

Chuyển sang dùng khóa
suy ra từ secret_C

Chuyển sang dùng khóa
suy ra từ secret_C

Trạng thái của Client và Server

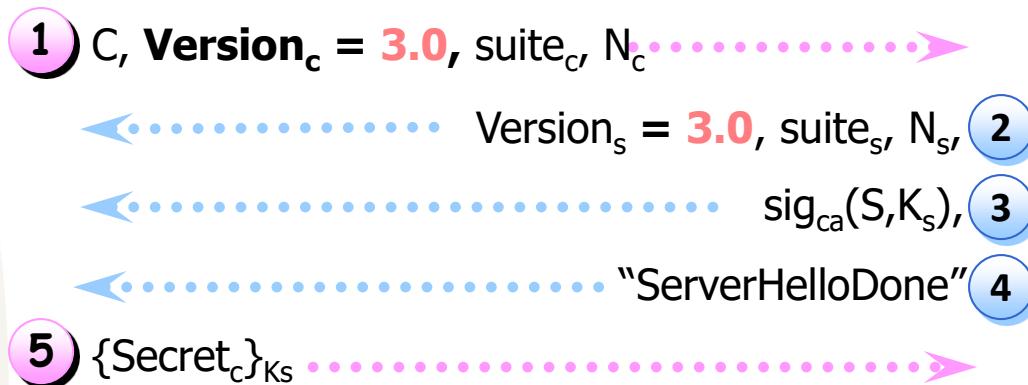


Phần chính của SSL 3.0

Client



Server



Nếu giao thức chính xác, từ lúc này,
C và S cùng chia sẻ khóa bí mật chung

Chuyển sang dùng khóa
suy ra từ secret_c

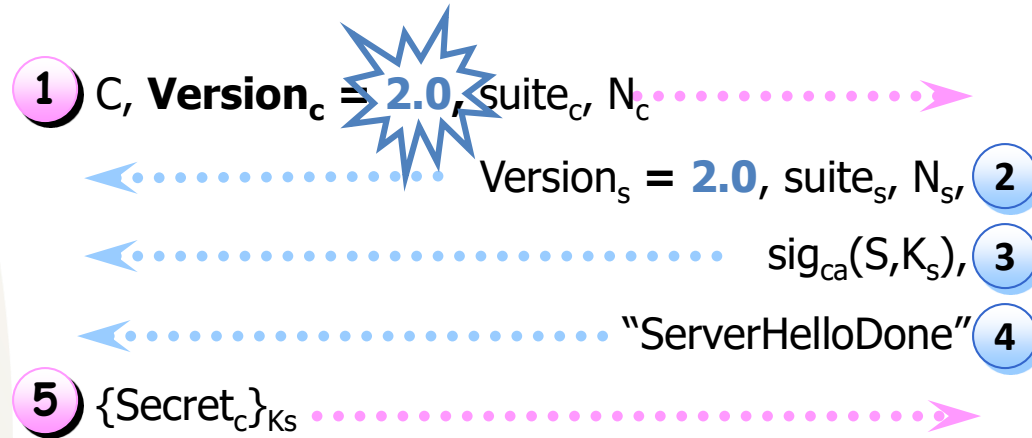
Chuyển sang dùng khóa
suy ra từ secret_c

Chọn lựa “nhầm” version cũ!

Client



Server

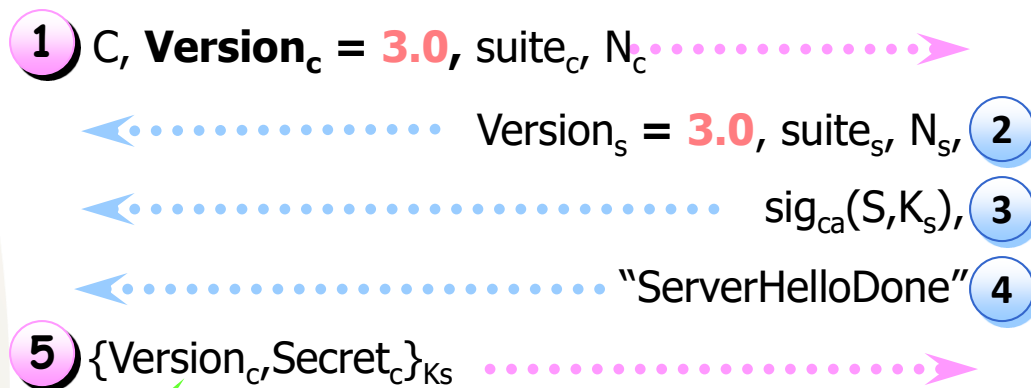


Server bị “lừa” và quyết định liên lạc
thiết lập liên lạc giữa S và C bằng version cũ

Điều chỉnh SSL

Client

Server



Nếu giao thức chính xác, từ lúc này,
C và S cùng chia sẻ khóa bí mật chung

Ngăn cản
tấn công
dùng version cũ

Bổ sung kiểm tra version
trùng với thông tin trong ClientHello

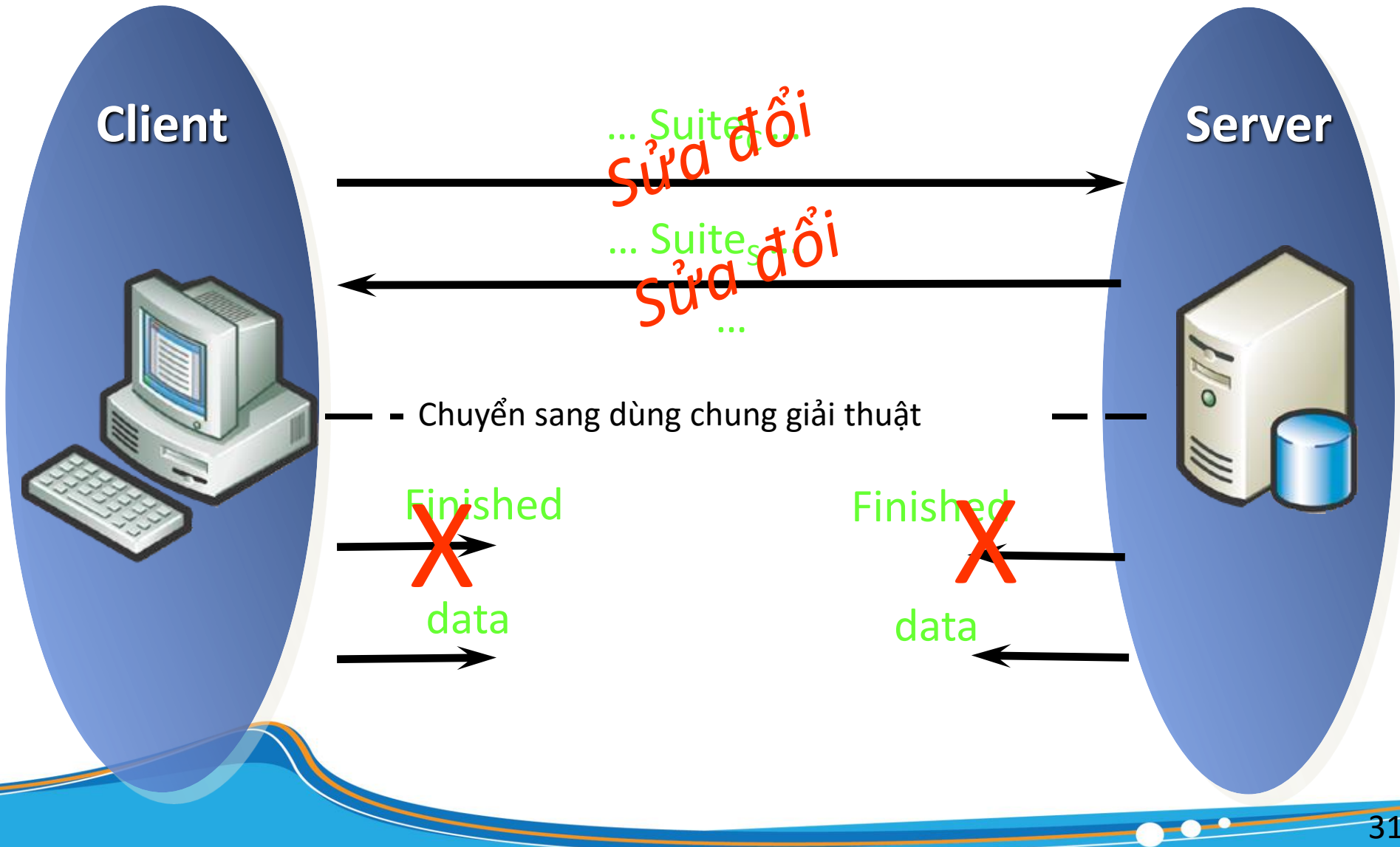
Chuyển sang dùng khóa
suy ra từ secret_c

Chuyển sang dùng khóa
suy ra từ secret_c

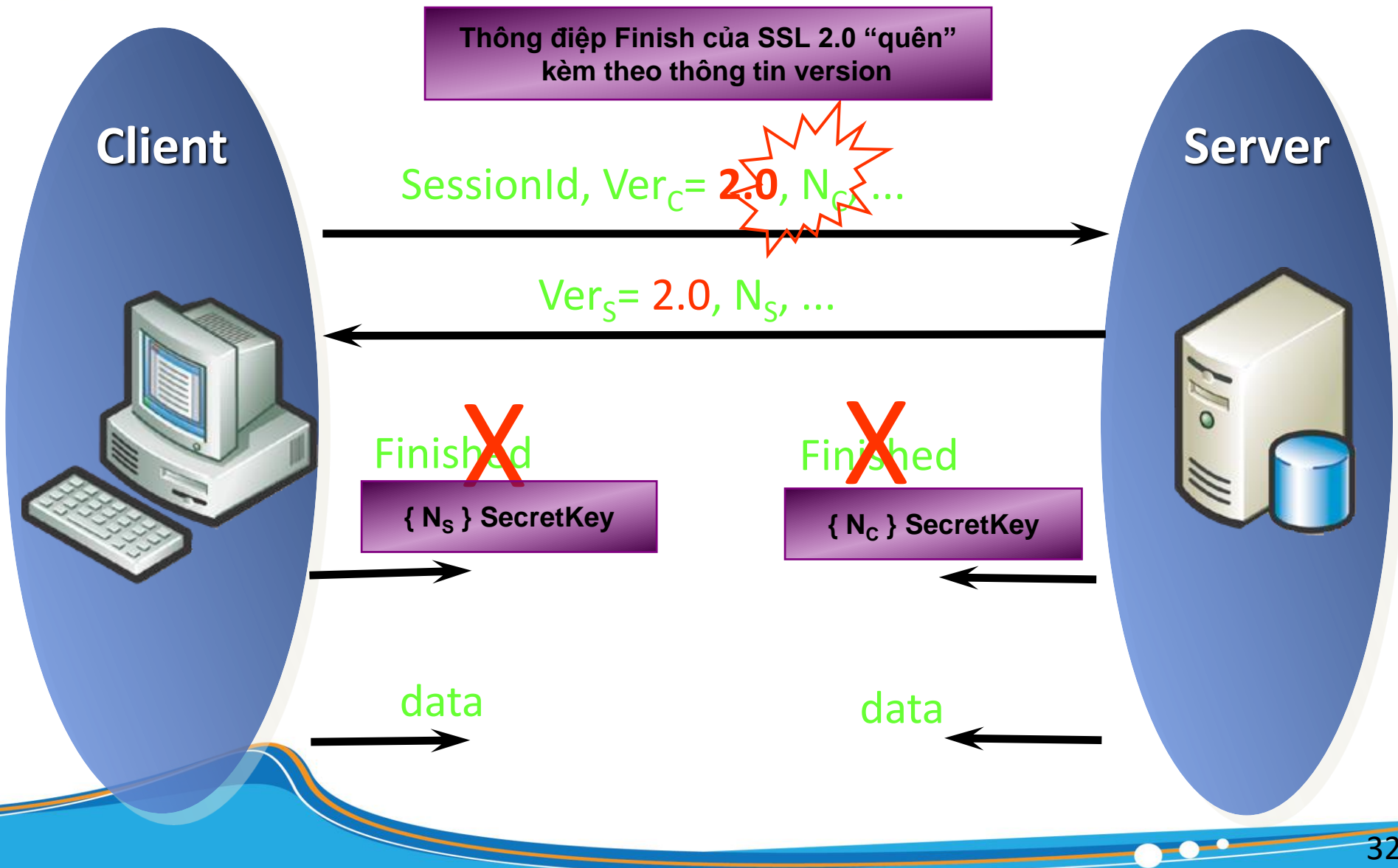
Tóm tắt

- ☐ A = protocol cơ bản
- ☐ C = A + certificate của public key
 - Authentication cho client và server
- ☐ E = C + thông điệp Xác nhận (Finished)
 - Ngăn cản việc tấn công “lừa” server chọn version cũ hay các giải thuật yếu
- ☐ F = E + Thông tin ngẫu nhiên (nonce)
 - Ngăn cản replay attack

Phát hiện sự bất thường bằng Finished



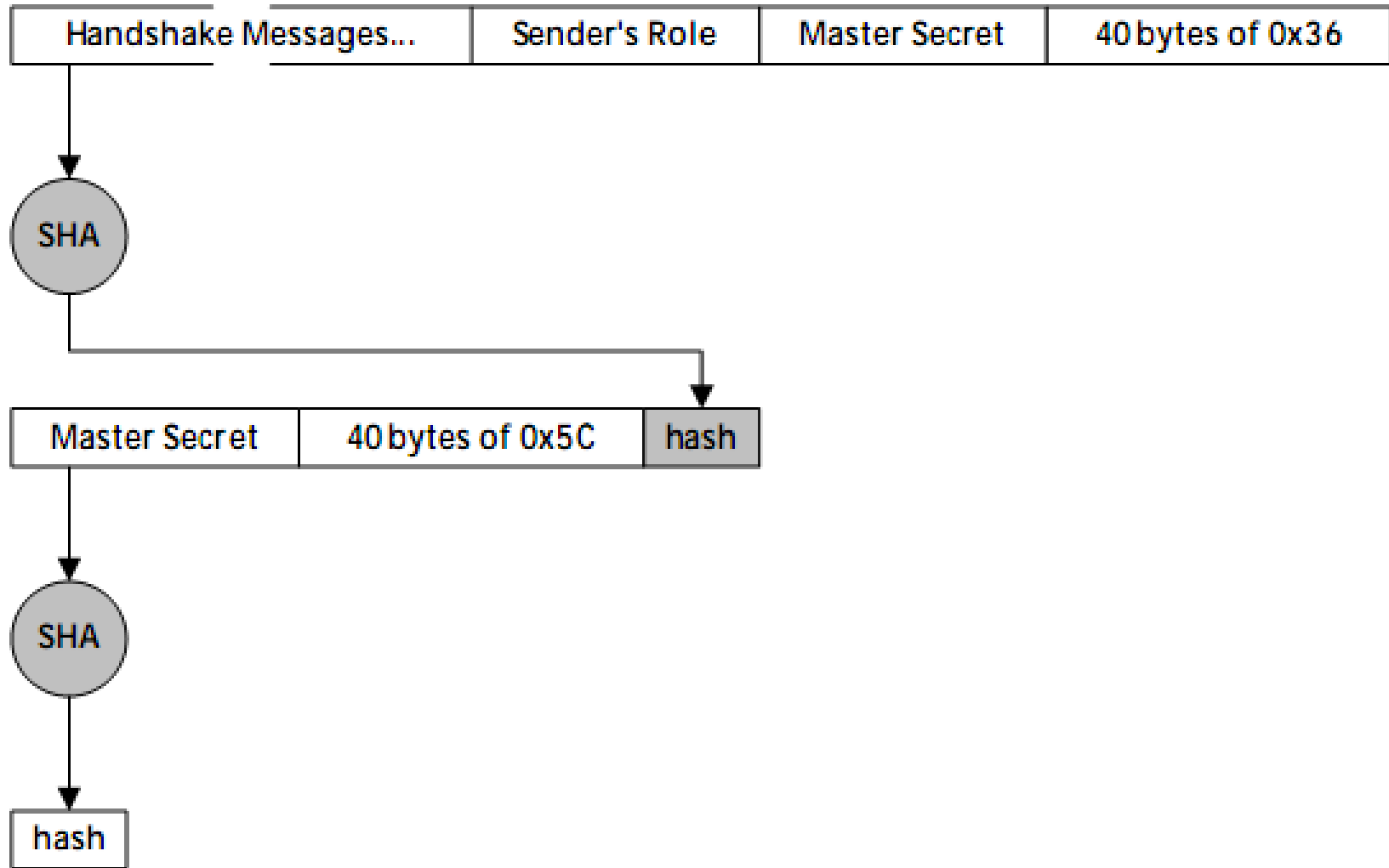
Ví dụ: Version Rollback Attack



Master secret

```
master secret = MD5(premaster secret + SHA('A' + premaster secret +  
ClientHello.random + ServerHello.random))  
  
+  
MD5(premaster secret + SHA('BB' + premaster secret +  
ClientHello.random + ServerHello.random))  
  
+  
MD5(premaster secret + SHA('CCC' + premaster secret +  
ClientHello.random + ServerHello.random))
```

Finish



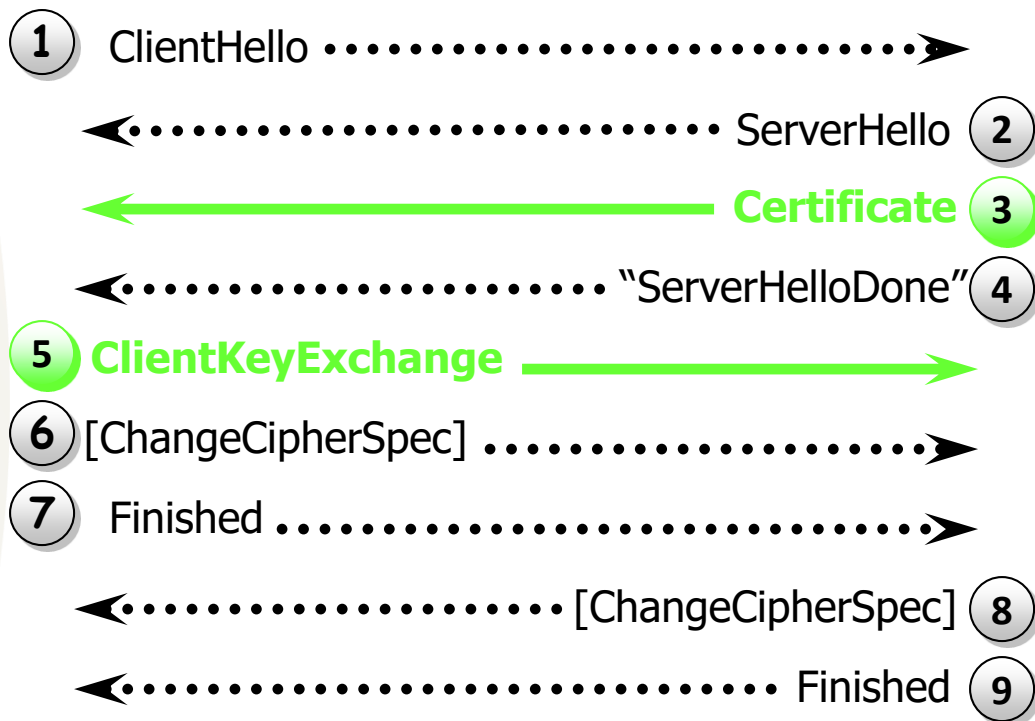
Server Authentication

Cần sử dụng giấy chứng nhận X.509 v3
trong trường hợp dùng RSA, Fixed
DH, Ephemeral DH

Client



Server



Server Authentication

- Cần sử dụng giấy chứng nhận X.509 v3 trong trường hợp dùng RSA, Fixed DH, Ephemeral DH

Server & Client Authentication

