2017/01/03, 2017 보안 기초 세미나

Network Security Essentials

- Chapter_5 전송-레벨 보안 -

임연주(yeonjoo@pel.smuc.ac.kr)

상명대학교 프로토콜공학연구실

목 차

웹 보안

- 웹 (WWW: World Wide Web)
  - 개요
    - 인터넷과 TCP/IP 인트라넷 상에서 운영하는 기본적인 서버/클라이언트를 구현하는 응용 프로그램
    - 웹 서비스를 구현함에 따라 보안서비스 적용에 필요성이 제기
  - 웹 보안의 필요성
    - 활성화된 웹 서버를 통한 관리 서버 침투 위험성 증가
      - HTTP 프로토콜의 발견되지 않은 구조적인 보안 취약점이 많음
    - 보안정보를 알 필요 없이 충분한 보안성을 제공 받아야 하는 일반 사용자가 많음

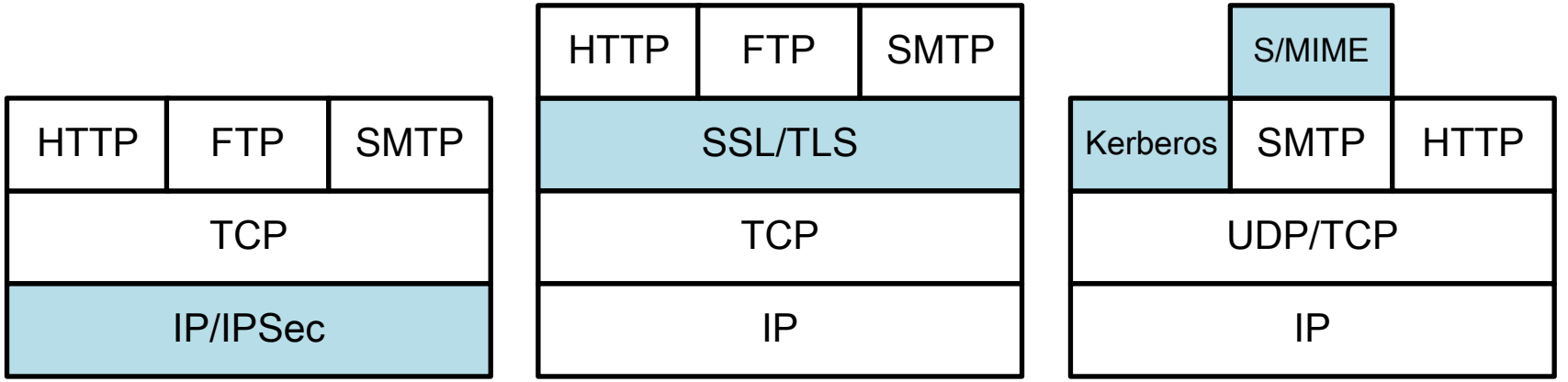
웹 보안

- 웹 보안 위협 비교 표

	위협	피해 사항	대응 방법
무결성 (Integrity)	• 사용자 데이터 변경 • 트로이 목마 브라우저 • 메모리의 변경 • 전송 중 메시지 트래픽 변경	• 정보의 손실 • 하드웨어(기계) 침해 • 다른 위협에 대한 취약성	• 암호학적 검사합 (Check sum)
기밀성 (Confidentiality)	• Net 도청 • 서버 정보 갈취 • 클라이언트의 데이터 갈취 • 네트워크 구성 정보 습득 • 서버와 통신 중인 클라이언트 정보 습득	• 정보의 손실 • 기밀성 침해	• 암호화 • 웹 프록시
서비스 거부 (Dos)	• 사용자 스레드 중단 • 다수의 가짜 위협 전송 • 메모리나 디스크 영역 차지 • DNS 공격을 통한 고립화	• 서버의 서비스 제공 방해 • 사용자의 작업 방해	• 예방하기 어려움
인증 (Authentication)	• 신분위장 • 데이터 위조	• 사용자 식별 오류 • 가짜 정보를 진짜로 오인	• 암호학적 기술

웹 보안

- 웹 트래픽 보안 방법
  - 웹 응용 범위나 TCP/IP 프로토콜마다 다양한 보안 방법이 존재
    - TCP/IP 프로토콜 스택에서 보안 기능의 위치 그림



웹 보안

- 웹 트래픽 보안 방법
  - 계층별 보안 방법 비교 표 추가하기
    a. 네트워크 레벨
       - 응용 프로그램에 투명하고 범용적인 해결책 제공
       - 선택된 트래픽만 처리하는 필터링 기능 제공

    b. 전송 레벨
       - 특정 패키지처럼 포함되어 있는 경우가 다수
         - e.g, 인터넷 익스플로우, 크롬 등
       - 기본 프로토콜의 일부로 제공 가능

    c. 응용 레벨
       - 특정 응용 프로그램 안에 포함
       - 응용프로그램의 보안 요구사항 수용 용이

목 차

- 웹 보안
- 안전 소켓 계층 (SSL)
- 전송 계층 보안 (TLS)
- HTTPS
- SSH

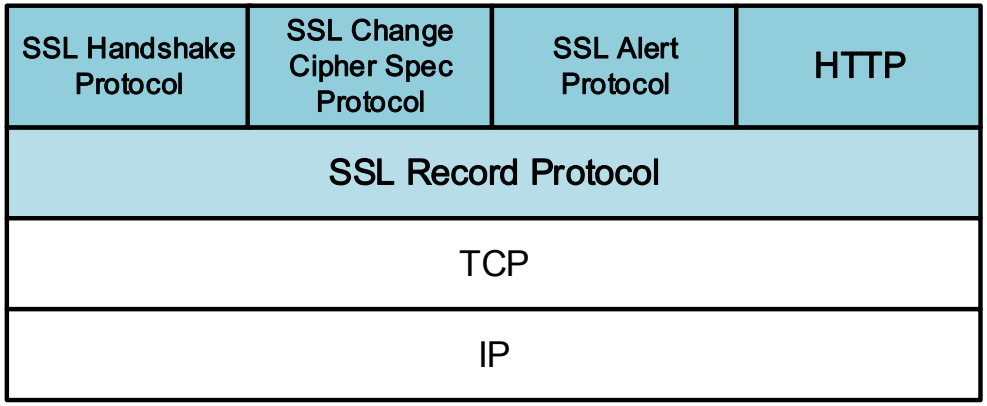
안전 소켓 계층 (SSL)

- 개요
  - SSL (Secure Socket Layer) 정의
    - 1993년 Netscape에서 웹 내의 안전한 통신을 위해 개발
    - 기존의 TCP/IP에 없는 보안 세션을 추가
      - 신뢰성 높은 서비스 제공을 하기 위해 TCP 사용
        - UDP 통신 보안을 제공하는 Datagram Transport Layer Security(DTLS)도 존재
    - 두 개의 계층으로 구성
  - SSL과 TLS
    - SSL이 표준 제안되면서 IETF 내에서 개발
    - TLSv1과 SSLv3.1은 거의 유사하므로 호환됨

안전 소켓 계층 (SSL)

- SSL 구조

1. SSL Handshake Protocol
   - 통신의 이전에 필요한 보안속성 협상을 수행
     - 세션 키, 암호화 알고리즘, 인증 등을 파라미터 정의
2. SSL Change Cipher Spec Protocol
   - 협상의 적용상태를 알림
3. SSL Alert Protocol
   - 세션의 종료 또는 오류 발생시 알림
4. SSL Record Protocol
   - 데이터를 암/복호화
     - 무결성 제공



안전 소켓 계층 (SSL)

- SSL의 개념: 세션과 연결
  - 세션 (Session)
    - 클라이언트와 서버 사이의 연관
      - 암호적 파라미터 교환과 인증
    - 다수의 연결이 파라미터를 공유할 수 있음
      - 각 연결마다 불필요한 보안 협상을 피할 수 있음
    - SSL Handshake Protocol
  - 연결 (Connection)
    - 데이터 전송 서비스 수행
    - 암호화와 무결성을 제공
    - SSL Record Protocol

안전 소켓 계층 (SSL)

- SSL의 개념: 세션과 연결
  - 세션 상태 (Session State) 파라미터 정리 표

파라미터	의미
Session identifier	세션의 상태를 나타내는 임의의 바이트 열(서버가 선택)
Peer certificate	서버와 클라이언트의 대등 X.509v3 인증서 (NULL도 가능)
Compression method	암호화 하기 전 압축에 사용되는 알고리즘
Cipher spec	MAC 계산에 사용되는 암호 또는 해시 알고리즘과 해시크기 등을 정의
Master secret	클라이언트와 서버가 공유하는 48-바이트 비밀 값
Is resumable	기존 세션의 새 연결 시작가능여부를 나타내는 플래그

  - 연결 상태 (Connection State) 파라미터 정리 표

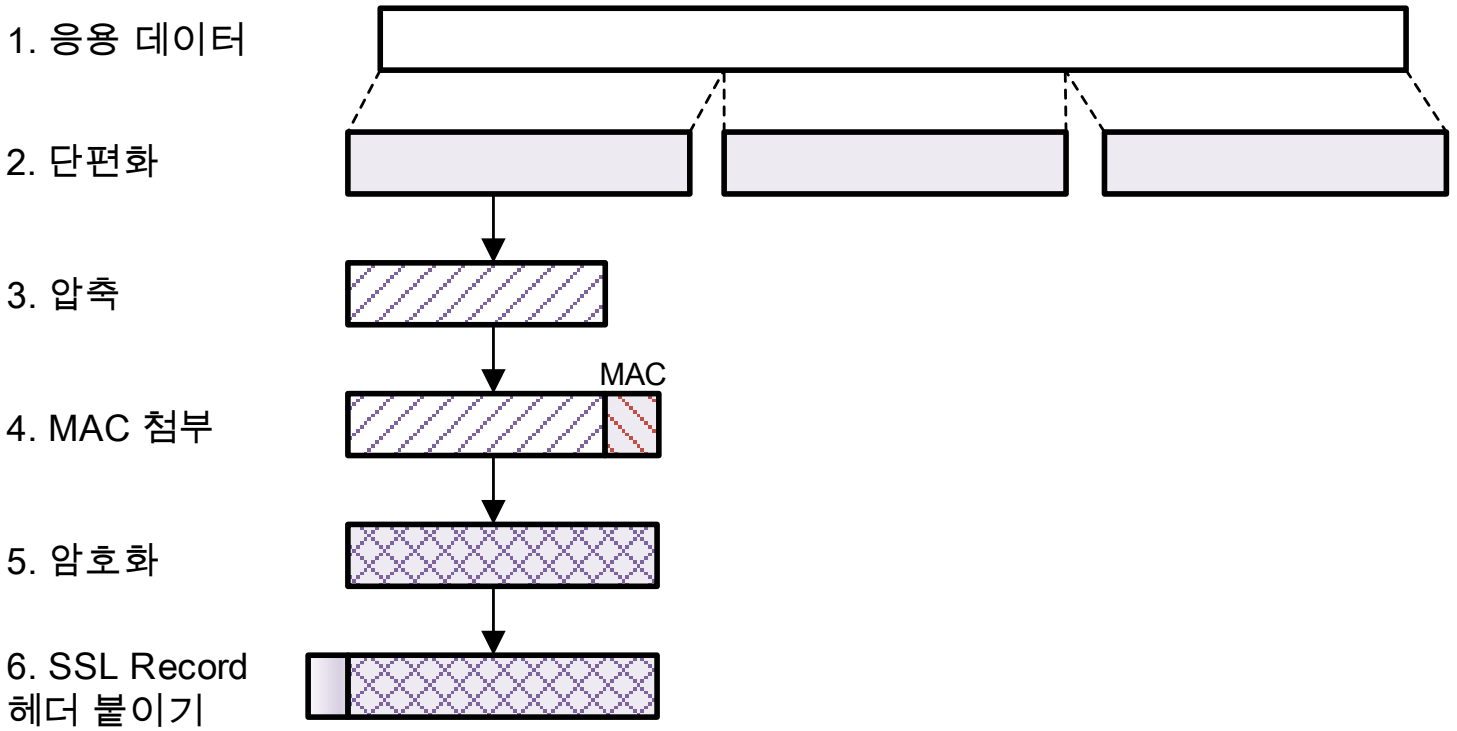
파라미터	의미
server/client random	각 연결에 사용하는 임의의 바이트 열(서버/클라이언트가 선택)
Server write MAC secret	MAC 계산시 사용하는 비밀키(**서버가 보낸** 데이터)
Client write MAC secret	MAC 계산시 사용하는 비밀키(**클라이언트가 보낸** 데이터)
Server write key	**서버**가 데이터를 암호화, **클라이언트**가 복호화할 때 사용하는 대칭 키
Client write key	**클라이언트**가 데이터를 암호화, **서버**가 복호화할 때 사용하는 대칭 키
Initialization vectors	CBC 모드로 블록 암호를 사용할 때 사용되는 초기화 백터 값
Sequence numbers	각 연결마다 송/수신하는 메시지에 대한 순서번호 (최대: $2^{64} - 1$)

안전 소켓 계층 (SSL)

- SSL Record Protocol (1/4)
  - 정의 및 기능
    - 상위 계층 메시지를 TCP로 전달
    - 기밀성, 메시지 무결성 제공
      - 기밀성 (Confidentiality)
        1. 핸드셰이크 프로토콜에서 정의한 공유된 비밀 키와 대칭 암호화 과정
        2. 암호화 전에 메시지 압축 과정: CRC 알고리즘을 이용해 데이터 오류 검사
      - 메시지 무결성 (Intergrity)
        - 공유된 비밀 키와 MAC
          - HMAC과 패딩만 다르고 비슷함

안전 소켓 계층 (SSL)

- SSL Record Protocol (2/4)
  - 전반적인 동작 그림

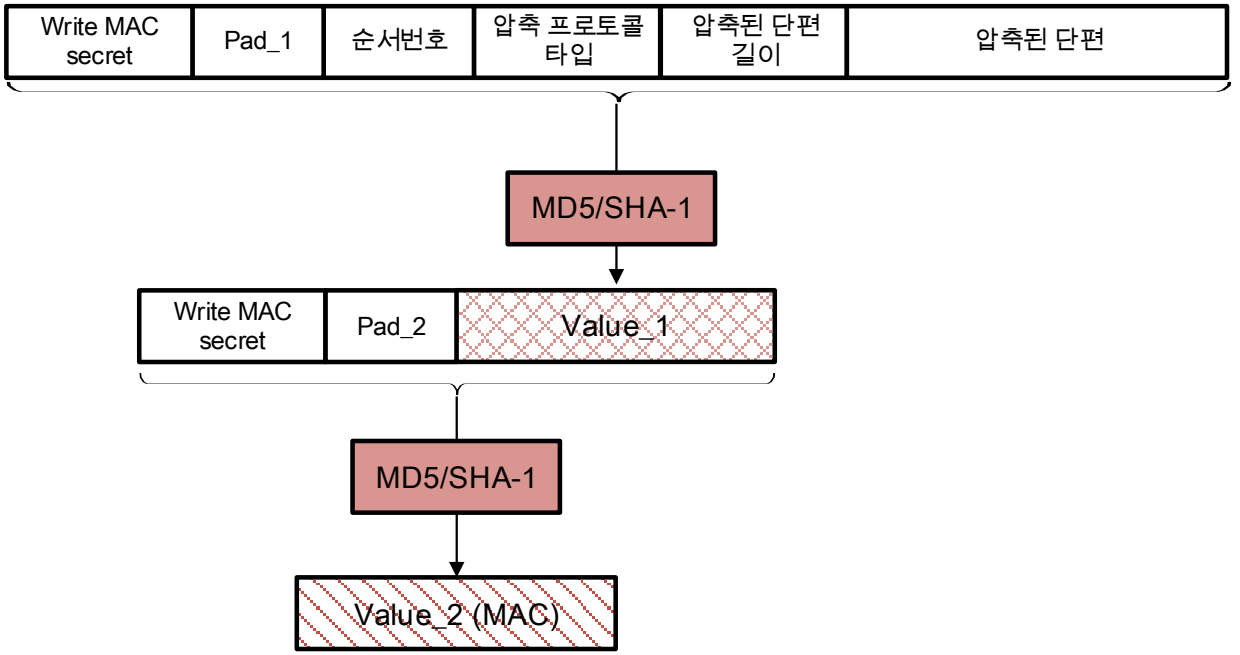


안전 소켓 계층 (SSL)

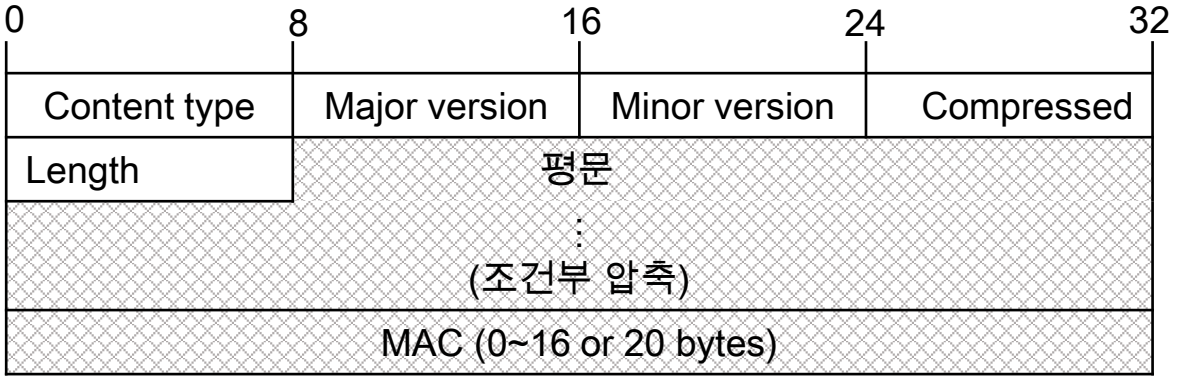
- SSL Record Protocol (3/4)
  - 동작 설명
    1. 단편화
       - 2^{14} byte 또는 없음
    2. 압축
       - 옵션사항
         - SSLv3과 TLS에는 없음
    3. MAC 첨부
       - 해시 함수의 값을 메시지 뒤에 첨부
    4. 암호화
       - 협상한 알고리즘을 이용해 암호화
         - IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128을 지원

안전 소켓 계층 (SSL)

- SSL Record Protocol (4/4)
  - MAC 계산과정 그림

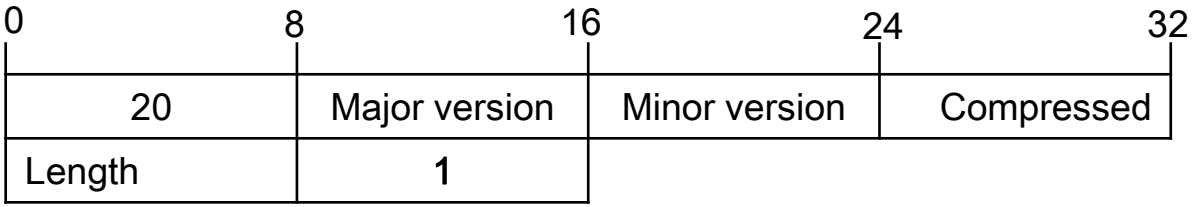


  - 포맷 그림



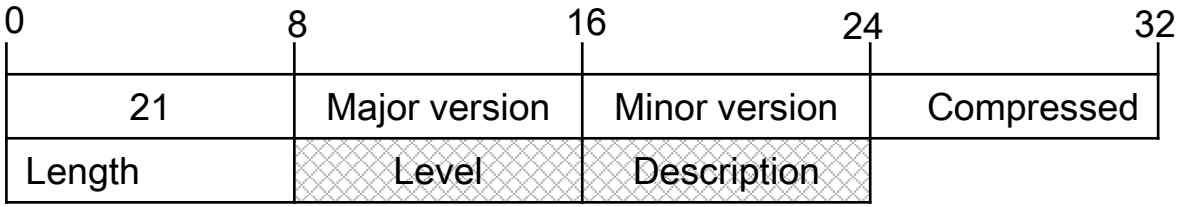
안전 소켓 계층 (SSL)

- Change Cipher Spec Protocol
  - 주요 기능
    - Handshake 프로토콜로 협상된 내용이 이후 통신부터 적용됨을 알림
  - 포맷 그림



안전 소켓 계층 (SSL)

- Alert Protocol (1/2)
  - 주요 기능
    - 암호 오류, 압축 오류, 메시지 인증 오류, 인증 실패 등을 경고 필드에 표기한 뒤 전송
  - 포맷 그림



안전 소켓 계층 (SSL)

- Alert Protocol (2/2)
  - 항상 심각(Always Fatal) 경고의 종류 표

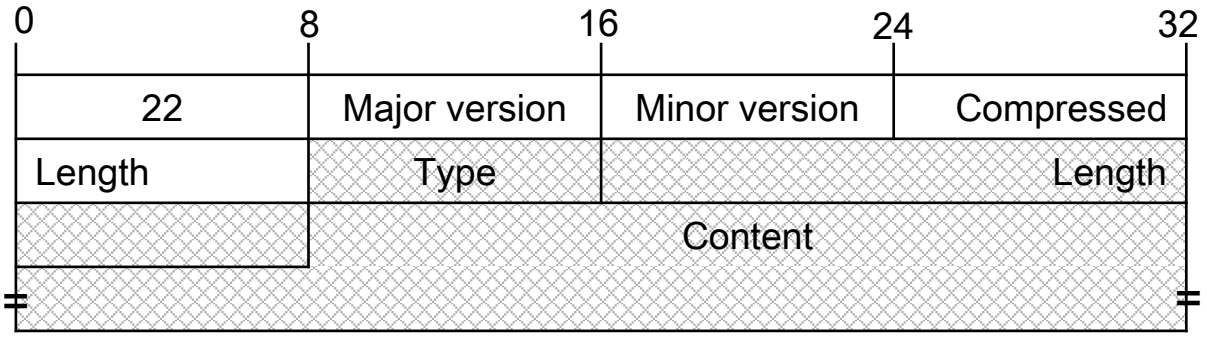
유형	의미
Unexpected_message	적합하지 않은 메시지의 수신
Bad_record_MAC	부정확한 MAC 수신
Decompressed_failure	압축해제함수에 적합하지 않은 입력 (압축풀기 불가 또는 최대 허용길이보다 큰 경우 등)
Handshake_failure	사용할 수 있는 옵션이 주어졌지만 송신자와 협상 불가
Illegal_parameter	핸드셰이크 메시지 안의 한 필드가 범위 밖이거나 다른 필드와 맞지 않음

  - 일반적인 경고의 종류 표

유형	의미
Close_notify	연결 종료 전에 이 경고를 보냄으로써 송신자가 이 연결에서 더 이상 메시지를 보내지 않을 것을 수신자에게 알림
오류_certificate	No, Bad, Unsupported등 의 수신된 인증서에 대한 경고
Certificate_상태	Revoked, Expired, Unknown 등의 인증서 문제가 발생

안전 소켓 계층 (SSL)

- Handshake Protocol (1/21)
  - 정의 및 기능
    - 한 세션 동안 이용되는 파라미터 생성
      - 사용되는 비밀정보를 공유
    - 서버와 클라이언트간의 상호인증을 수행
    - 사용할 키 교환 방식, 대칭키 암호 방식, HMAC 방식, 압축방식 등의 보안속성을 협상
  - 포맷 그림



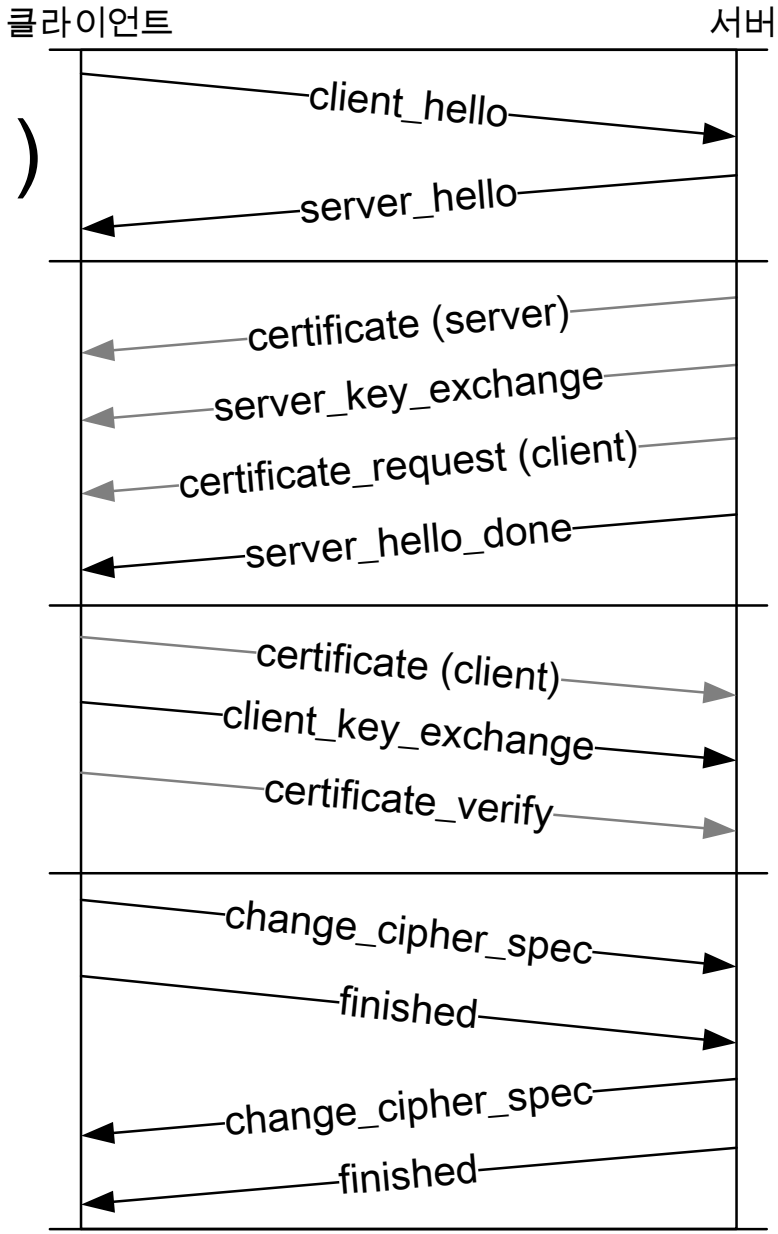
안전 소켓 계층 (SSL)

- Handshake Protocol (2/21)
  - 메시지 유형 정리 표

type	메시지	파라미터
0	hello_request	null
1	client_hello	version, random, session id, cipher suite, compression method
2	server_hello	
11	certificate	Chain of X.509v3 인증서
12	server_key_exchange	parameters, signature
13	certificate_request	type, authorities
14	server_done	null
15	certificate_verify	Signature
16	client_key_exchange	parameters, signature
20	Finished	hash value

안전 소켓 계층 (SSL)

- Handshake Protocol (3/21)
  - 전반적인 동작 그림



단계 1
보안 기능 수집: 프로토콜 버전, 세션 ID, 암호 조합, 압축 방법, 조기 랜덤 넘버를 포함

단계 2
서버가 필요하다고 생각되면, 인증서, 키 교환을 보내고 인증서를 요청
Hello 메시지 단계를 끝내는 것을 알림

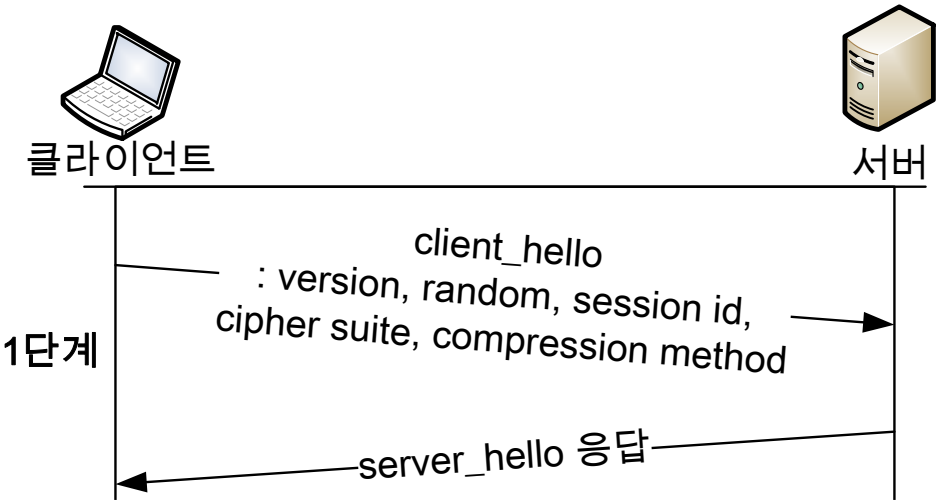
단계 3
클라이언트는 요청된 경우, 인증서를 보냄
클라이언트는 인증서에 대한 확인(서명)을 보낼 수도 있음

단계 4
암호 조합을 교환하고 핸드셰이크 프로토콜을 종료

- 음영이 들어간 화살표는 옵션이거나 상황에 따라 필요한 경우만 씀

안전 소켓 계층 (SSL)

- Handshake Protocol (4/21)
  - 1단계: 보안 기능 설정 (1/3)

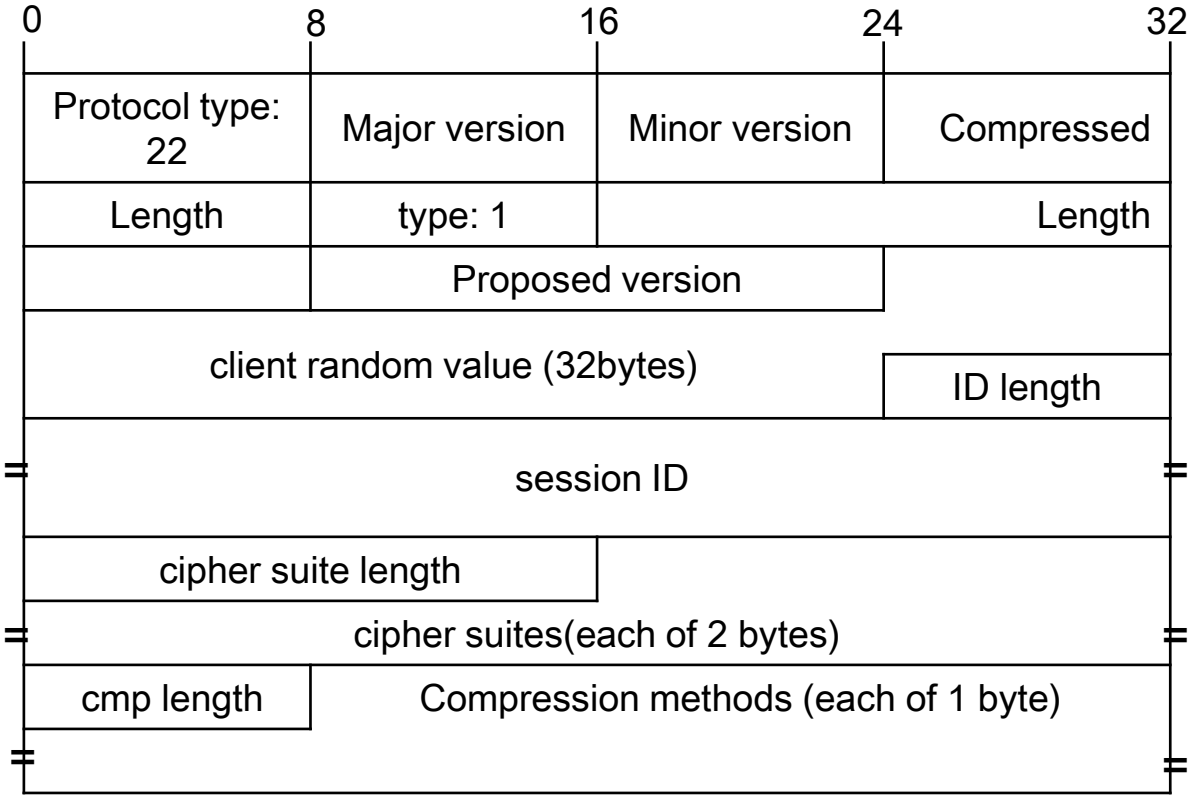


암호 명세	의미
Cipher algorithm	RC4, DES 등
MAC Algorithm	MD5 또는 SHA-1
Cipher type	스트림 또는 블록
Is exportable	참 또는 거짓
Hash size	0, 16(MD5) 또는 20(SHA-1) bytes
Key material	Write 키 생성에 사용할 데이터를 포함하는 데이터 열
IV Size	CBC 암호화에 사용하는 IV의 크기

이름	역할
version	클라이언트가 수용할 수 있는 가장 높은 버전
random	32비트 타임 스탬프와 28 바이트 난수 (nonce)
session ID	값이 0이 아니면 동일한 암호 값을 사용, 0이면 새로운 세션을 위한 값을 설정
cipher suite	클라이언트가 지원하는 암호 알고리즘 목록과 교환 방식을 나열
compression method	압축 방법

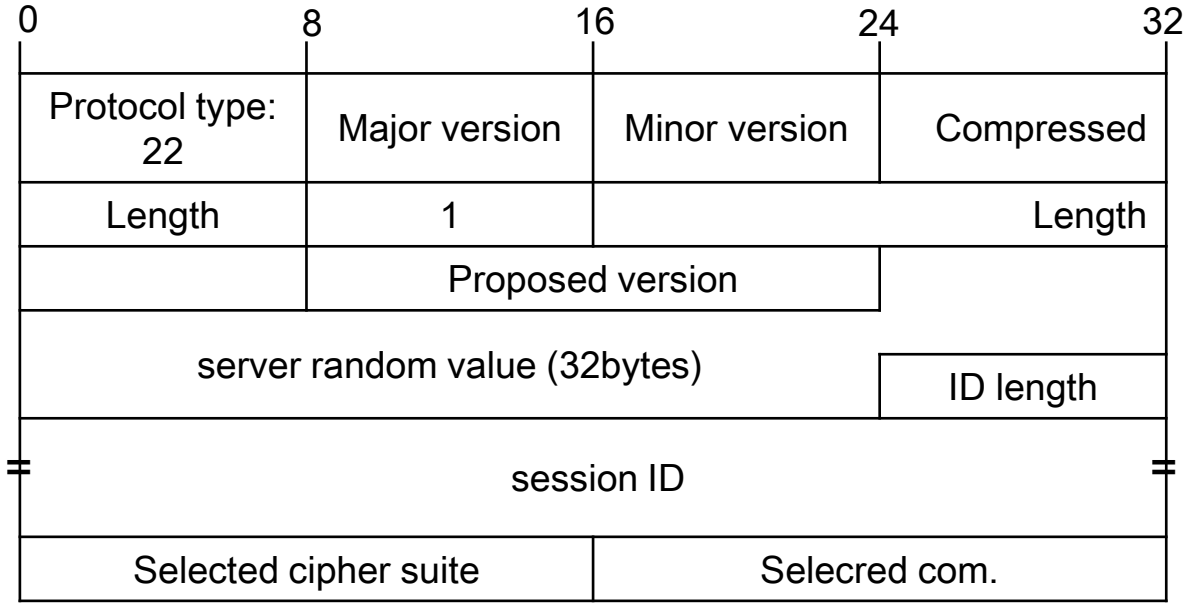
전송 계층 보안 (TLS)

- Handshake Protocol (5/21)
  - 1단계: 보안 기능 설정 (2/3)
    - client_hello 메시지 포맷



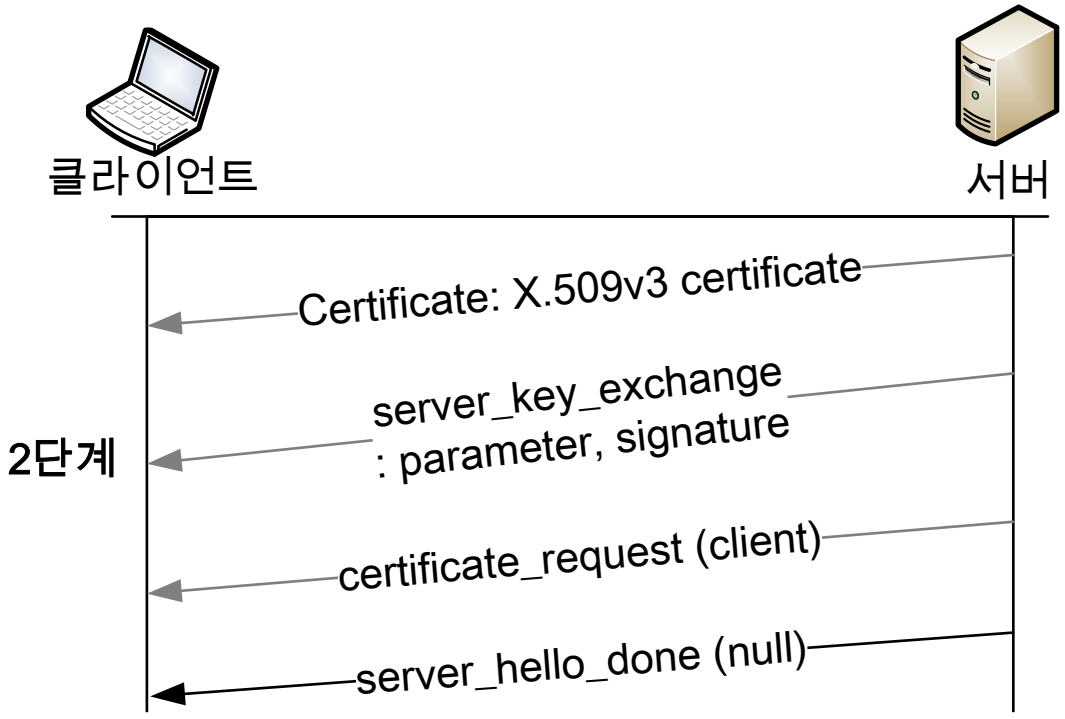
전송 계층 보안 (TLS)

- Handshake Protocol (6/21)
  - 1단계: 보안 기능 설정 (3/3)
    - server_hello 메시지 포맷



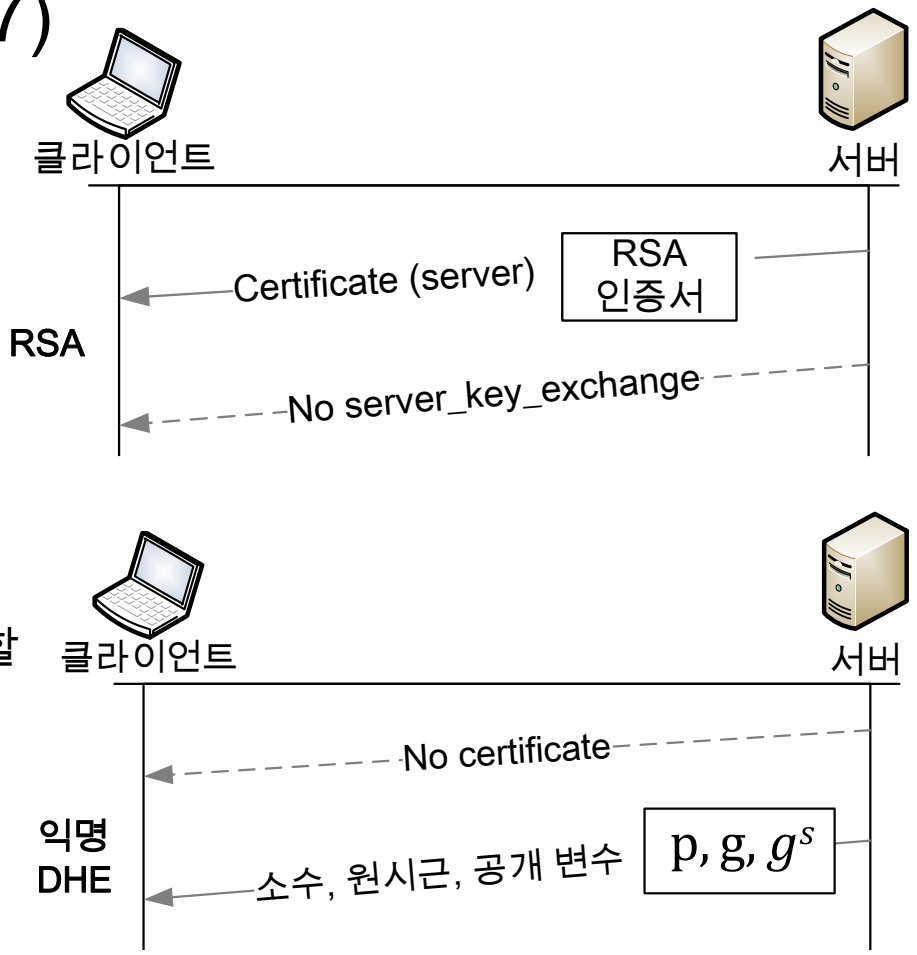
안전 소켓 계층 (SSL)

- Handshake Protocol (7/21)
  - 2단계: 서버 인증과 키 교환 (1/7)
    - 대표적 키 교환 알고리즘 (4가지)
      1. RSA
      2. 익명 DHE
      3. 임시 DHE
      4. 고정 DHE



안전 소켓 계층 (SSL)

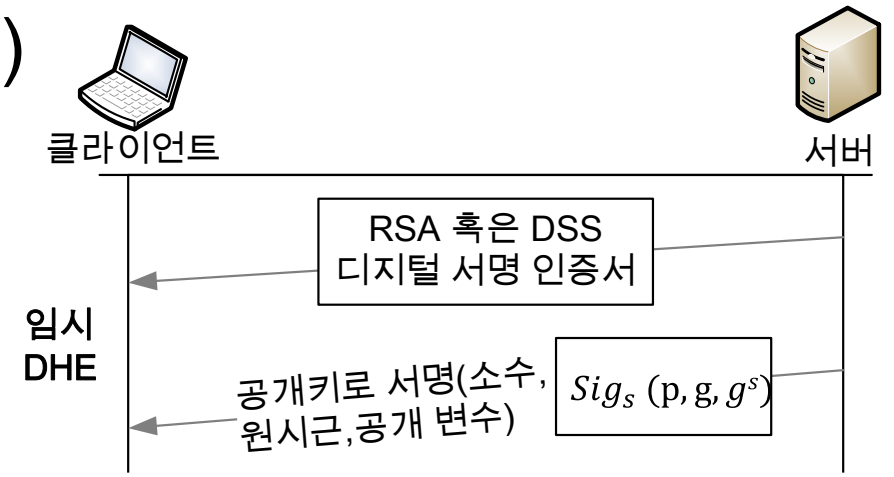
- Handshake Protocol (8/21)
  - 2단계: 서버 인증과 키 교환 (2/7)
    - 4가지 키 교환과 인증 유형
      1. RSA
         - 서버의 인증서를 전송
      2. 익명 DHE
         - p, a, g^x 를 전송
         - 교환된 키는 인증되지 않았기 때문에 중간자 공격에 취약
           - set_cipher_list에서 !ADH를 설정할 수 있음
         - pre_master_secret: $g^{cs}\ mod\ p$



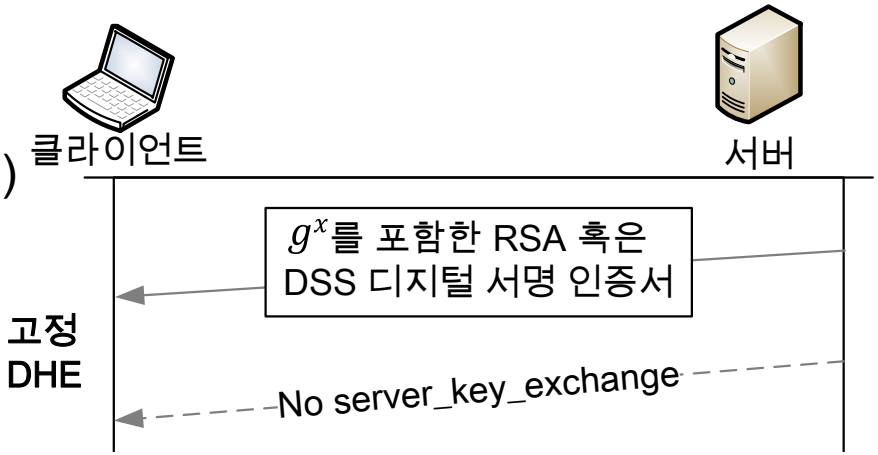
안전 소켓 계층 (SSL)

- Handshake Protocol (9/21)
  - 2단계: 서버 인증과 키 교환 (3/7)
    - 4가지 키 교환과 인증 유형

3. 임시 DHE
   - 클라이언트/서버의 임시 공개 키로 p, a, g^x 를 서명해 전송
     - 각 인스턴스 또는 프로토콜 실행은 다른 공개 키를 사용
       - Perfect Forward Secrecy(PFS)제공
   - pre_master_secret: g^{cs}
4. 고정 DHE
   - 인증서의 내용에 g^x (public 파라미터)를 포함
   - 인증기관의 개인키로 서명하여 인증
     - RSA 또는 DSS 인증서 활용
   - pre_master_secret: $g^{cs}\ mod\ p$

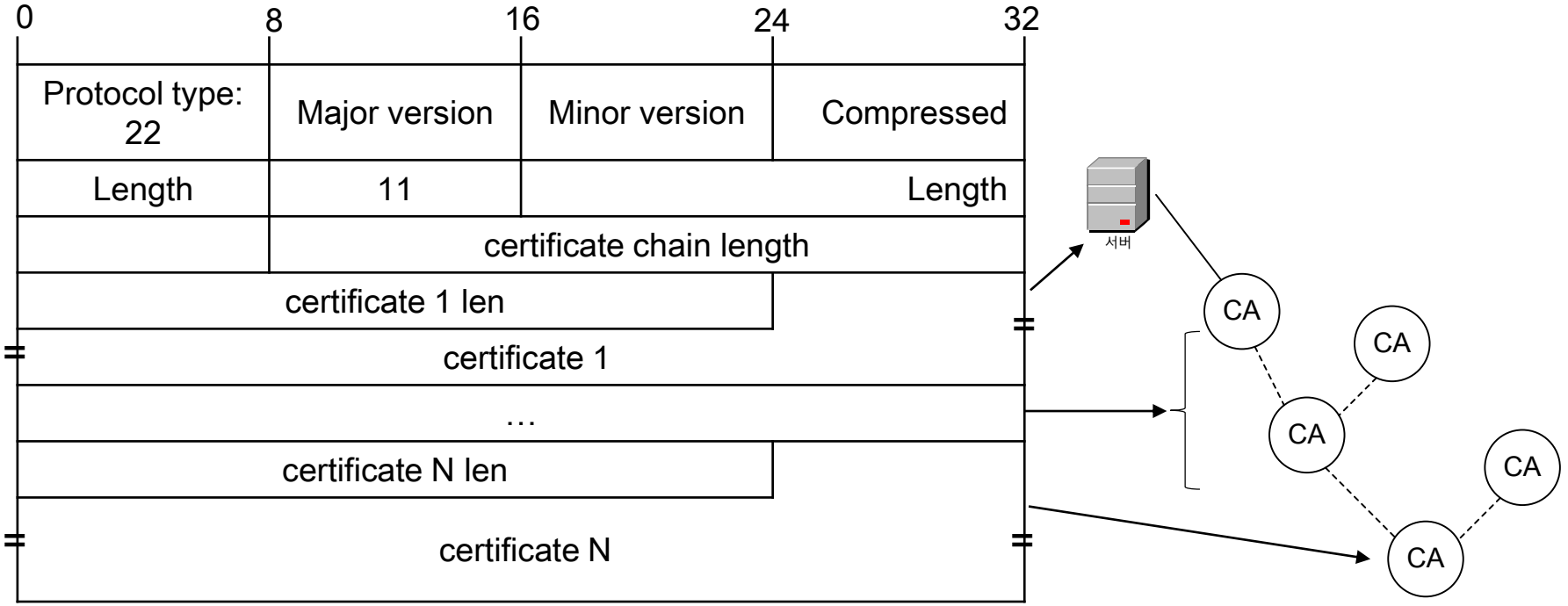


Sig_s = Signed with sever public key



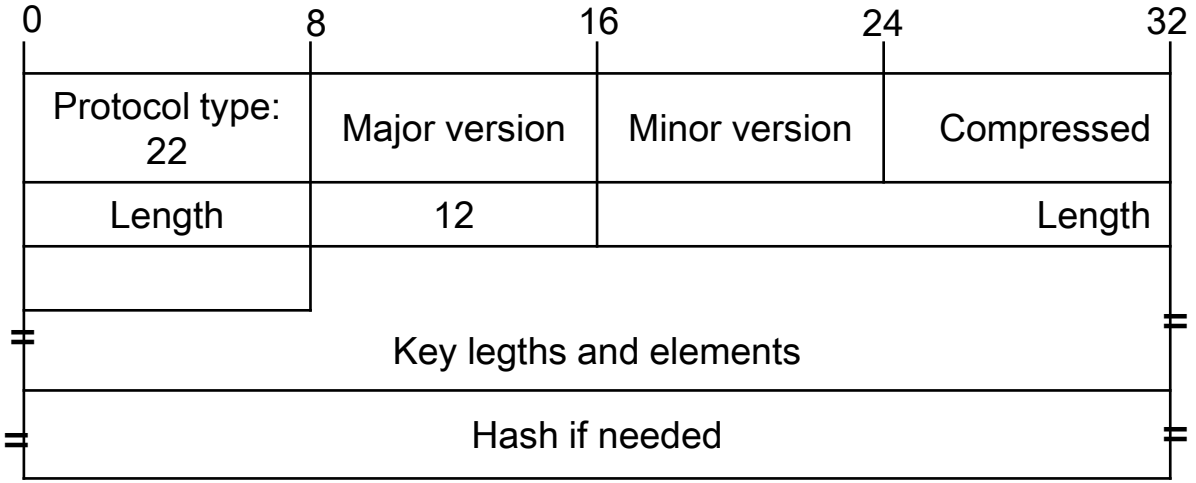
전송 계층 보안 (TLS)

- Handshake Protocol (10/21)
  - 2단계: 서버 인증과 키 교환 (4/7)
    - certificate 메시지 포맷



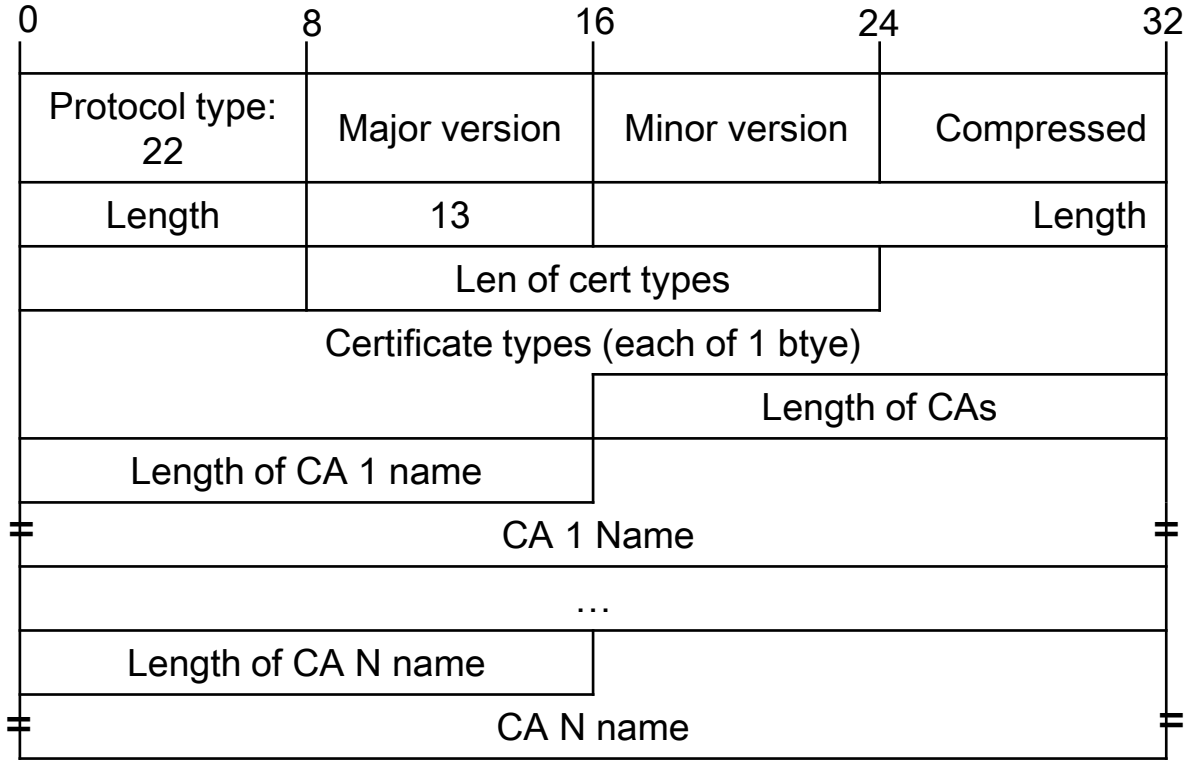
전송 계층 보안 (TLS)

- Handshake Protocol (11/21)
  - 2단계: 서버 인증과 키 교환 (5/7)
    - server_key_exchange 메시지 포맷



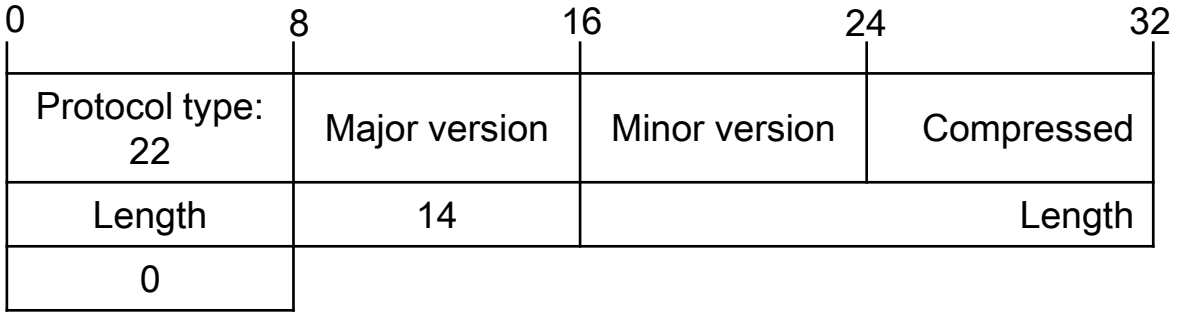
전송 계층 보안 (TLS)

- Handshake Protocol (12/21)
  - 2단계: 서버 인증과 키 교환 (6/7)
    - certificate_request 메시지 포맷



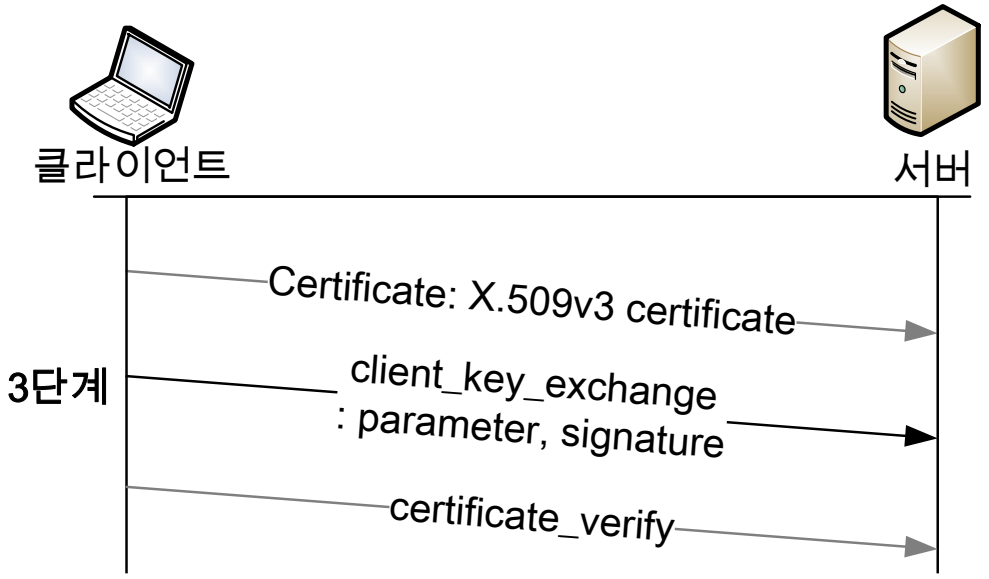
안전 소켓 계층 (SSL)

- Handshake Protocol (13/21)
  - 2단계: 서버 인증과 키 교환 (7/7)
    - server_hello_done 메시지 포맷



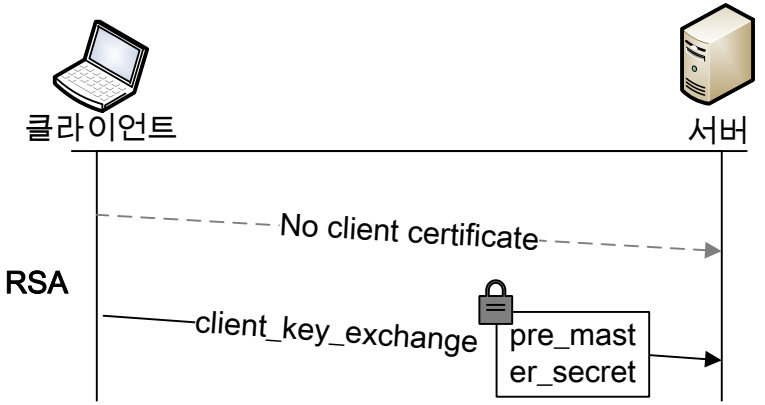
안전 소켓 계층 (SSL)

- Handshake Protocol (14/21)
  - 3단계: 클라이언트 인증과 키 교환 (1/5)
    - 클라이언트가 자신의 인증서가 유효함을 스스로 검증
    - 대표적 키 교환 알고리즘 (4가지)
      1. RSA
      2. 익명 DHE
      3. 임시 DHE
      4. 고정 DHE

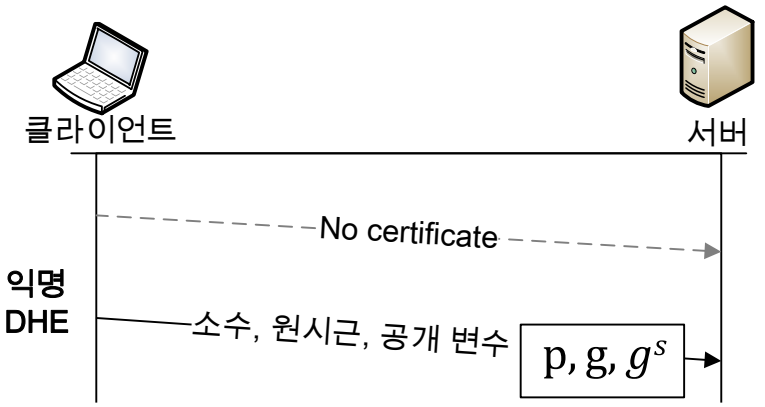


안전 소켓 계층 (SSL)

- Handshake Protocol (15/21)
  - 3단계: 클라이언트 인증과 키 교환 (2/5)
    - 4가지 키 교환과 인증 유형
      1. RSA
         - 서버의 공개 키로 PM 값을 암호화하여 전송
         - pre_master_secret: client_version (2bytes) + nonce (4bytes)



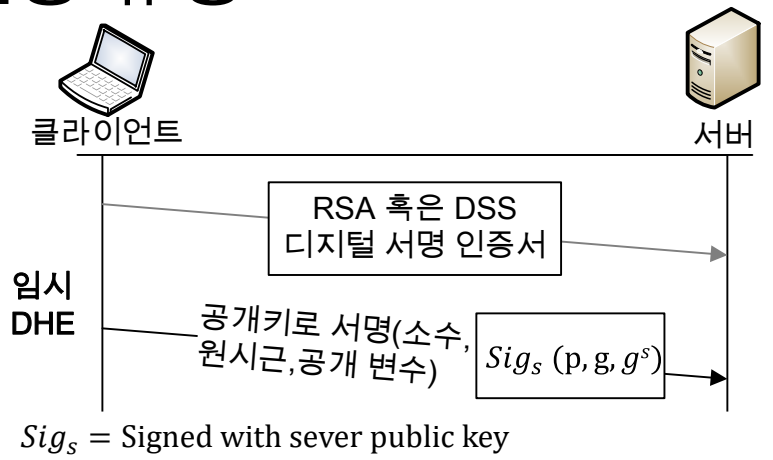
      2. 익명 DHE



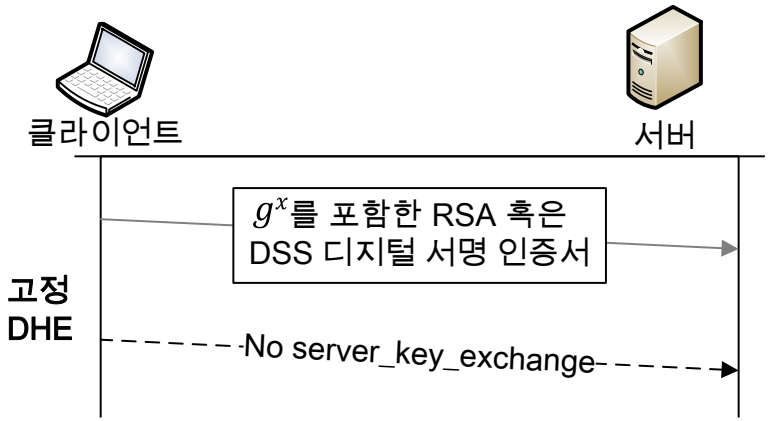
안전 소켓 계층 (SSL)

- Handshake Protocol (16/21)
  - 3단계: 클라이언트 인증과 키 교환 (3/5)
    - 4가지 키 교환과 인증 유형

3. 임시 DHE

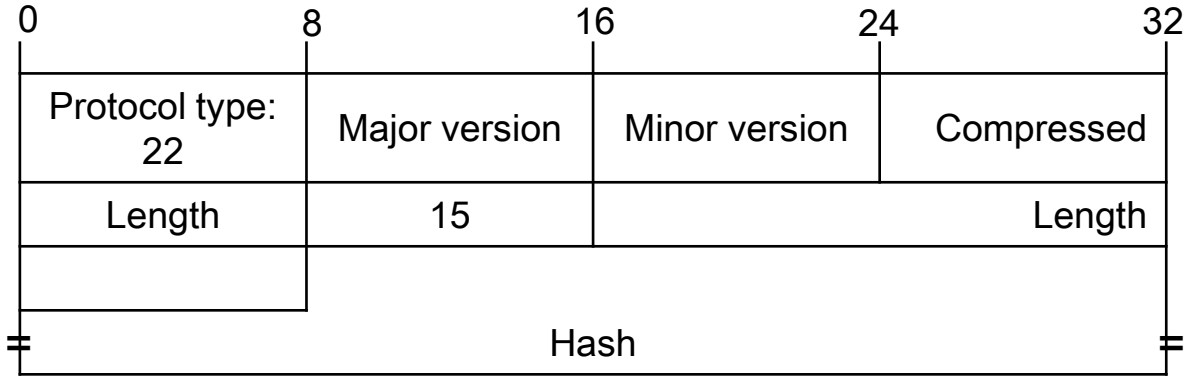


4. 고정 DHE



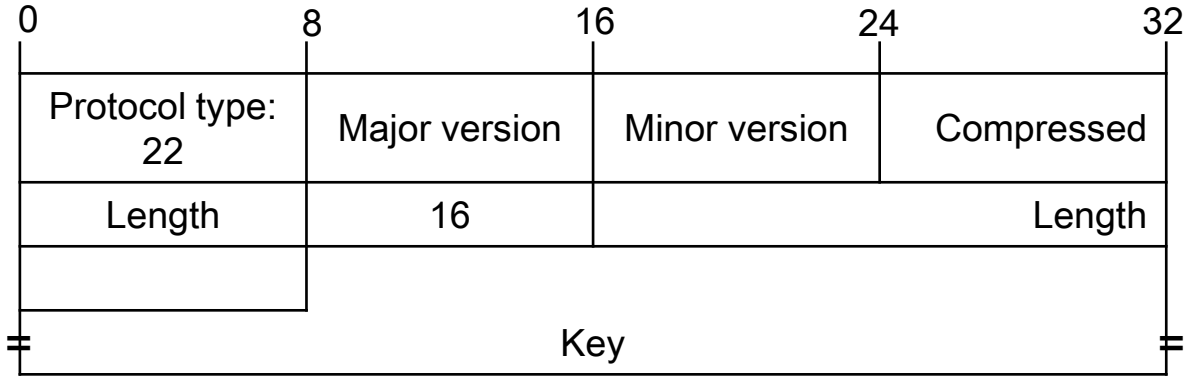
안전 소켓 계층 (SSL)

- Handshake Protocol (17/21)
  - 3단계: 클라이언트 인증과 키 교환 (4/5)
    - certificate_verify 메시지 포맷



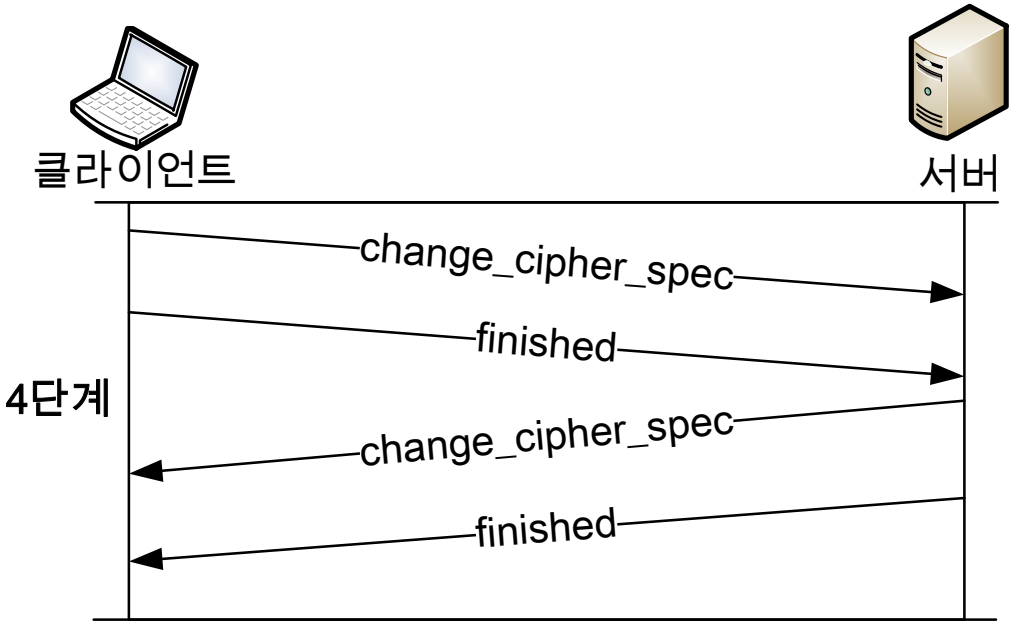
안전 소켓 계층 (SSL)

- Handshake Protocol (18/21)
  - 3단계: 클라이언트 인증과 키 교환 (5/5)
    - client_key_exchange 메시지 포맷



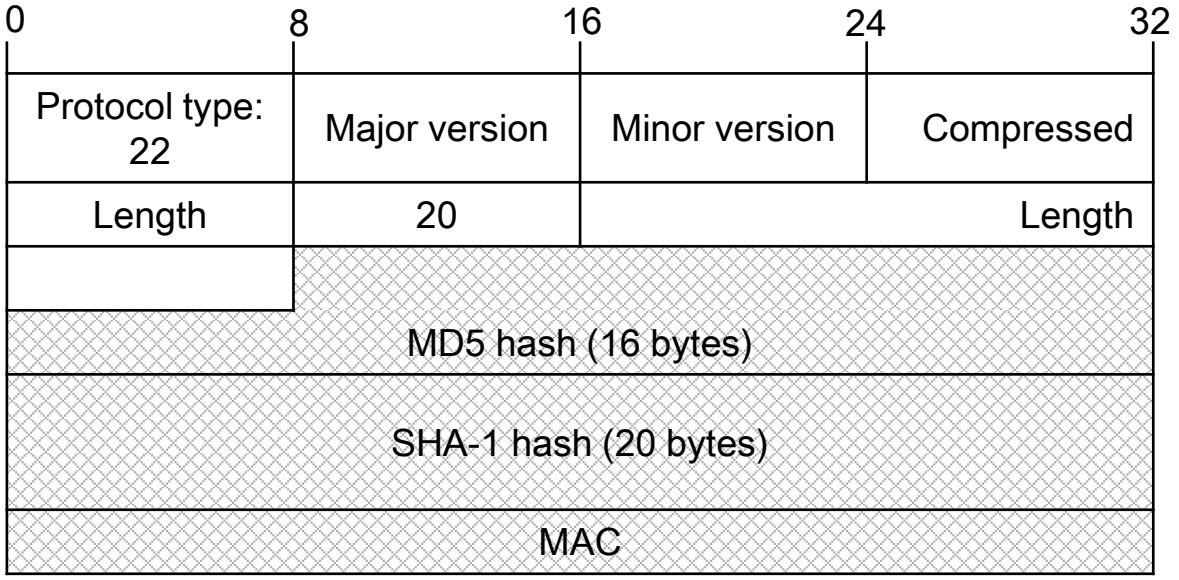
안전 소켓 계층 (SSL)

- Handshake Protocol (19/21)
  - 4단계: 종료 (1/3)
    - 안전한 연결 종료
    - Finished 메시지는 키 교환과 인증 과정이 성공적임을 확인 (해시 값)



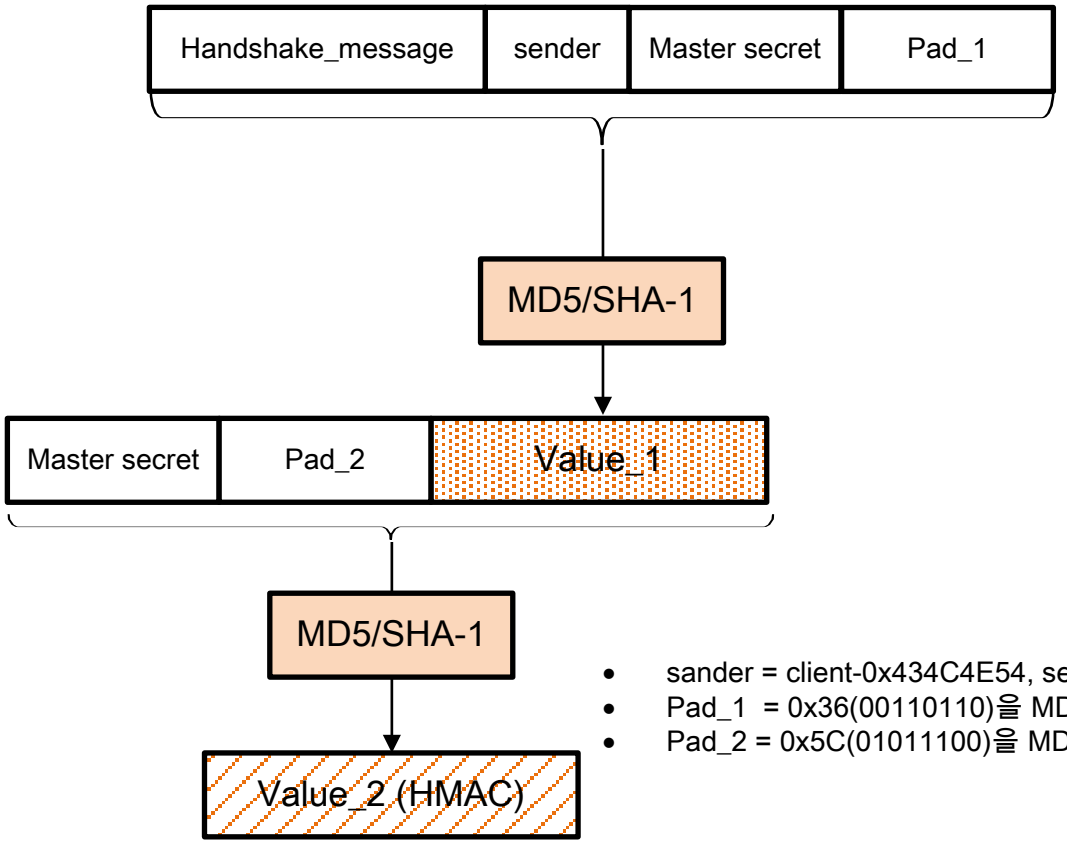
안전 소켓 계층 (SSL)

- Handshake Protocol (20/21)
  - 4단계: 종료 (2/3)
    - Finished 메시지 포맷



안전 소켓 계층 (SSL)

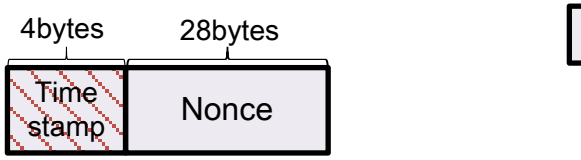
- Handshake Protocol (21/21)
  - 4단계: 종료 (3/3)
    - Finished 메시지의 해시 계산 과정 그림



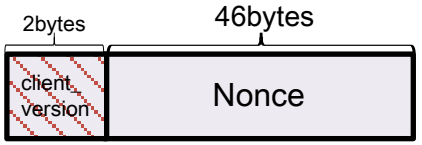
- sander = client-0x434C4E54, server-0x53525652
- Pad_1 = 0x36(00110110)을 MD5는 48번, SHA-1은 40번 반복
- Pad_2 = 0x5C(01011100)을 MD5는 48번, SHA-1은 40번 반복

안전 소켓 계층 (SSL)

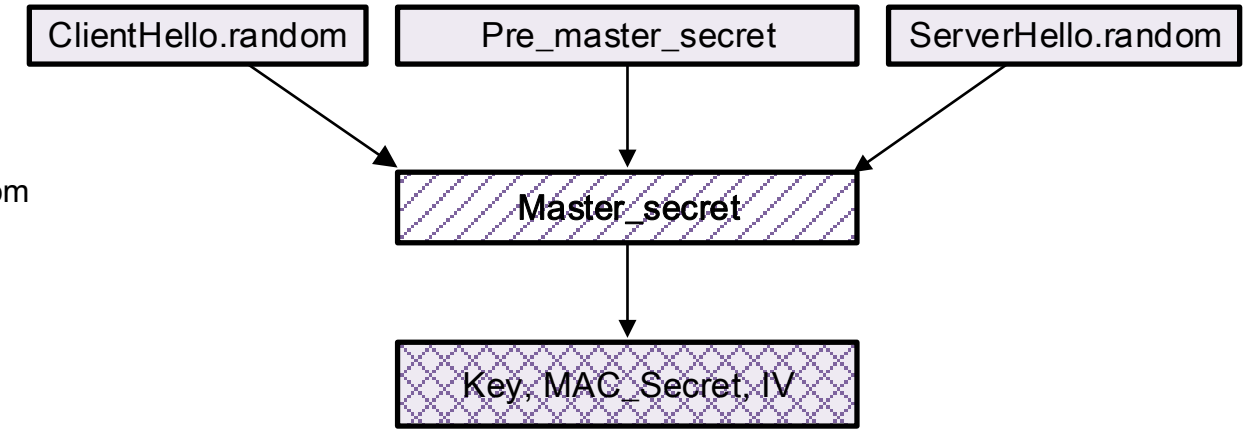
- 암호 계산 (1/4)
  - 파라미터의 생성
    - ClientHello.random/ServerHello.random
      - PRF를 이용해 만든 32bytes 난수
    - Pre_master_secret
      - Client가 PRF를 이용해 만든 48bytes 세션 키
    - CR/SR, PM 구조 그림



ClientHello.random/ServerHello.random



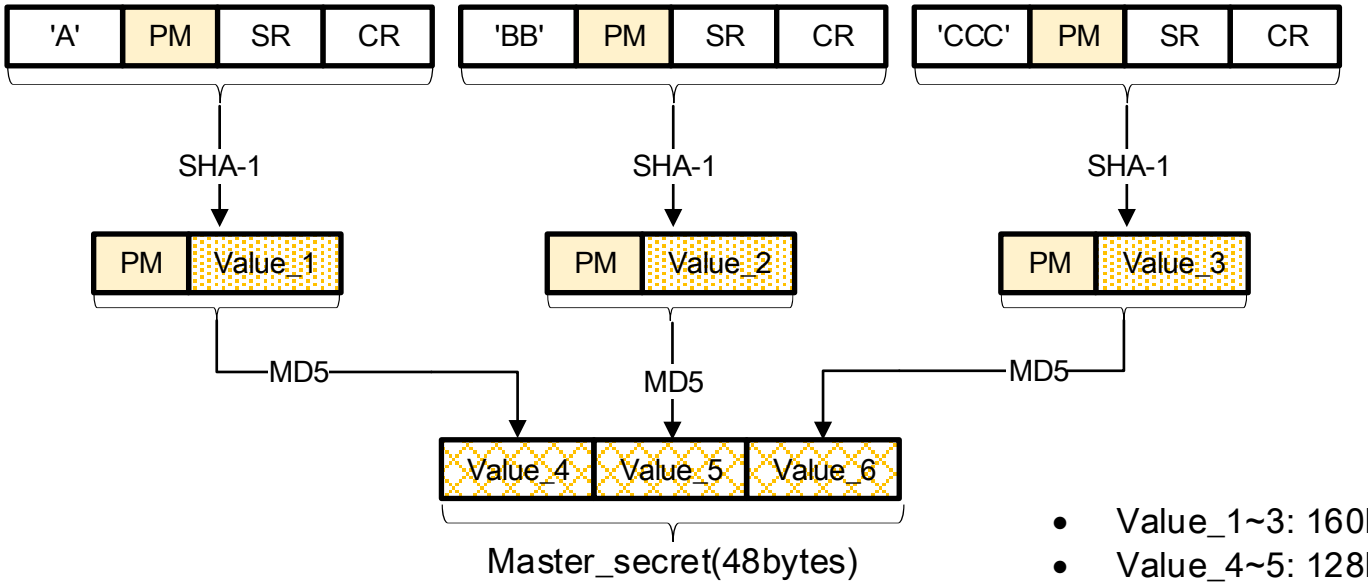
Pre_master_secret



안전 소켓 계층 (SSL)

- 암호 계산 (2/4)
  - 파라미터의 생성
    - Master_secret
      - PM, CR, SR 3가지를 이용해 생성
      - 생성 과정 그림

- PM = Pre_master_secret
- CM = ClientHello.random
- SM = ServerHello.random

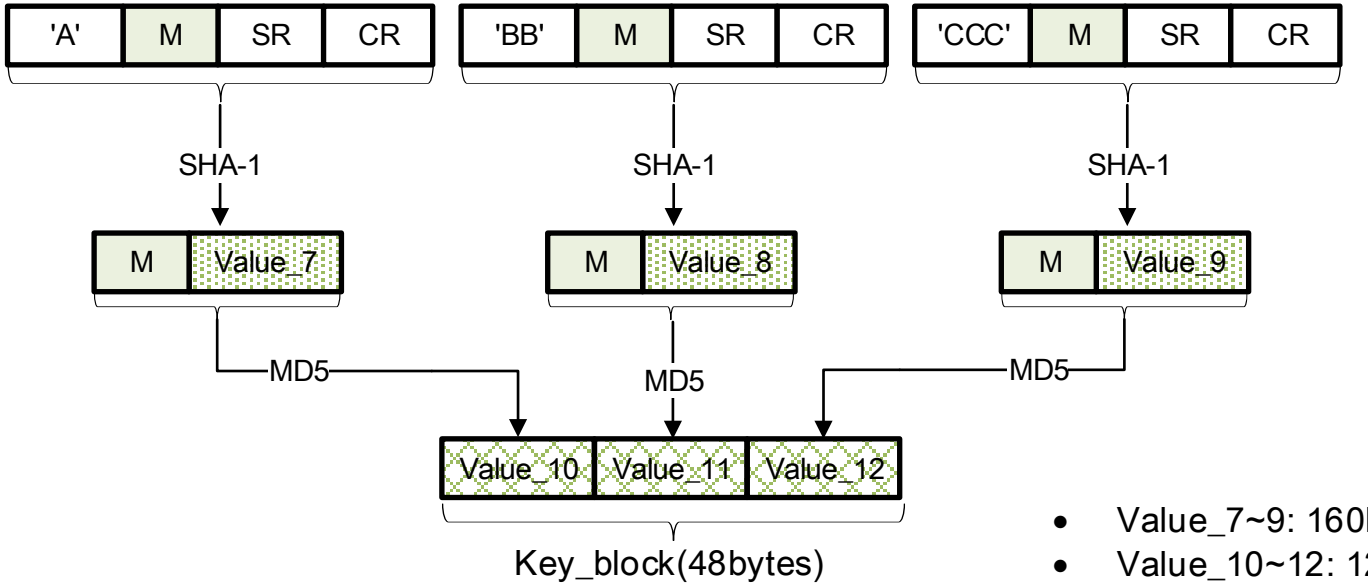


- Value_1~3: 160bits
- Value_4~5: 128bits

안전 소켓 계층 (SSL)

- 암호 계산 (3/4)
  - 파라미터의 생성
    - Key_block
      - Master_secret 생성과정과 유사함
      - 생성 과정 그림

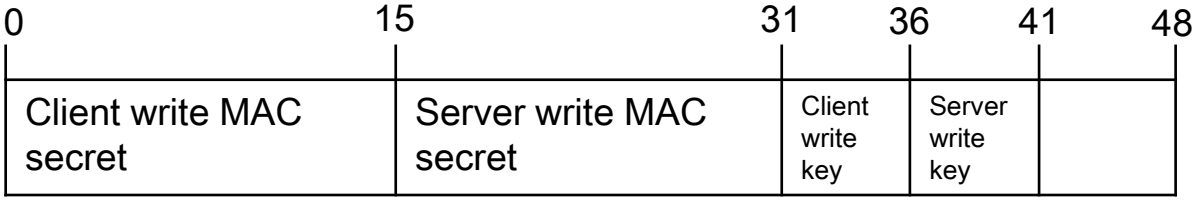
- M = Master_secret
- CM = ClientHello.random
- SM = ServerHello.random



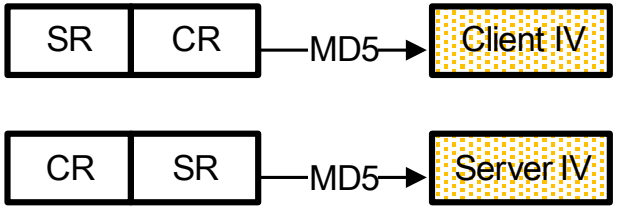
- Value_7~9: 160bits
- Value_10~12: 128bits

안전 소켓 계층 (SSL)

- 암호 계산 (4/4)
  - 파라미터의 생성
    - Key_block
      - 구조 그림



      - Client/Server IV 계산 과정 그림



- CM = ClientHello.random
- SM = ServerHello.random

목 차

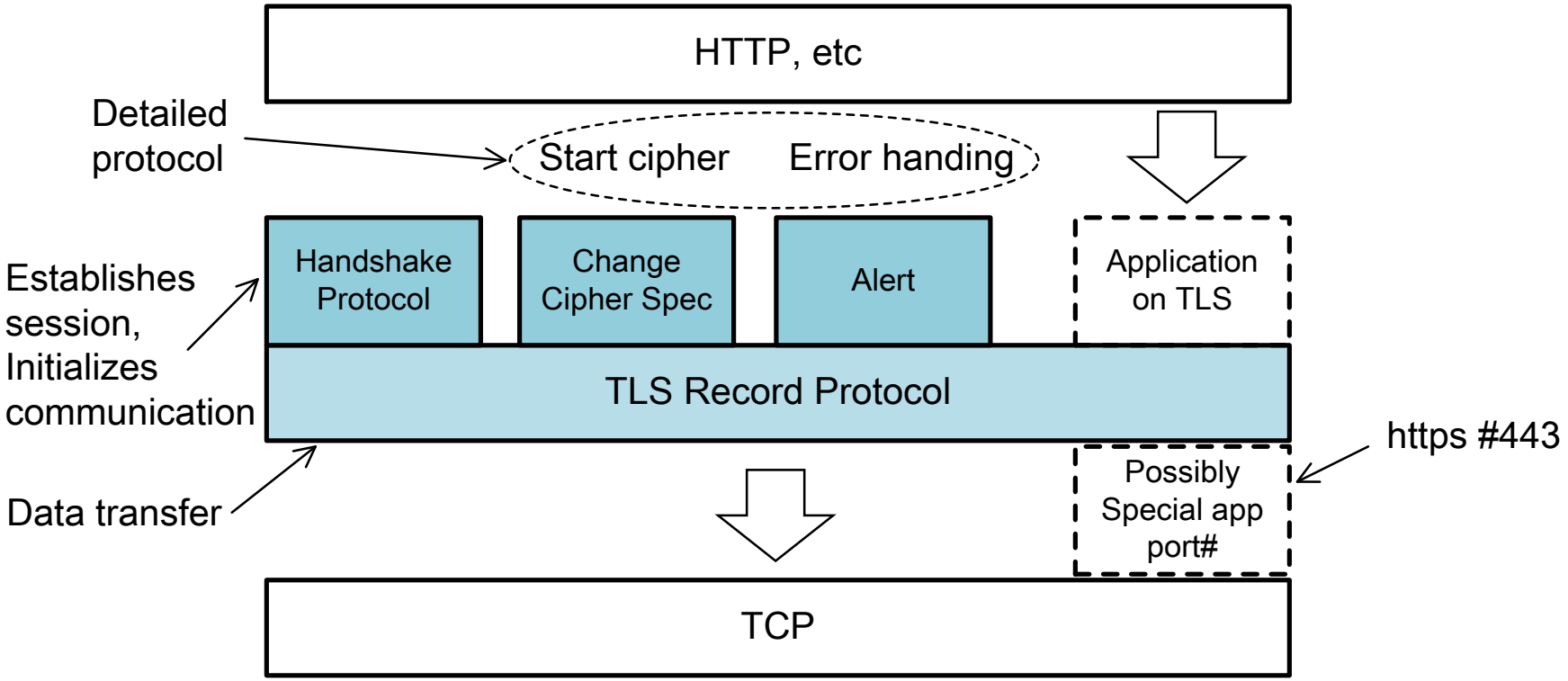
- 웹 보안
- 안전 소켓 계층 (SSL)
- 전송 계층 보안 (TLS)
- HTTPS
- SSH

전송 계층 보안 (TLS)

- 개요
  - TLS (Transport Layer Security) 정의
    - 표준 개발을 위해 SSL을 기반한 IETF (Internet Engineering Task Force) 표준
      - 1999년 1월에 RFC 2246으로 제안된 인터넷 표준
  - SSL과 TLS 주요 차이점
    1. 레코드 형식
       - 버전 필드
    2. MAC 계산 알고리즘과 범위
       - HMAC (XOR)
    3. Master Secret 계산식, 핸드셰이크 프로토콜 일부분
       - PRF(Rseudo Random Function) 사용

전송 계층 보안 (TLS)

- 개요
  - TLS Protocol stack 그림



전송 계층 보안 (TLS)

- 개요
  - TLS 사용 와이어샤크 캡쳐

```
▲ Secure Sockets Layer
  ▲ TLSv1.2 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 512
    ▲ Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 508
        Version: TLS 1.2 (0x0303)
      ▲ Random
          GMT Unix Time: Apr  1, 2018 19:35:27.000000000 ���ϫα�  g�ڰ�
          Random Bytes: d6a4f472f3964bf8a7d14c22941dbbc961c83a334aacacdd...
        Session ID Length: 32
        Session ID: 894ee456a1519875314816897e4be7a2ea08ebeeec97ab8b...
        Cipher Suites Length: 36
      ▲ Cipher Suites (18 suites)
```

```
0030  40 3d 78 09 00 00 16 03  01 02 00 01 00 01 fc 03   @=x..... ........
0040  03 5a c0 b5 ef d6 a4 f4  72 f3 96 4b f8 a7 d1 4c   .Z...... r..K...L
0050  22 94 1d bb c9 61 c8 3a  33 4a ac ac dd ee ea 4e   "....a.: 3J.....N
0060  a0 20 89 4e e4 56 a1 51  98 75 31 48 16 89 7e 4b   . .N.V.Q .u1H..~K
0070  e7 a2 ea 08 eb ee ec 97  ab 8b fe 53 3d 4d 0f f6   ........ ...S=M..
0080  8f 6f 00 24 5a 5a c0 2b  c0 2f c0 2c c0 30 cc a9   .o.$ZZ.+ ./.,.0..
0090  cc a8 cc 14 cc 13 c0 09  c0 13 c0 0a c0 14 00 9c   ........ ........
00a0  00 9d 00 2f 00 35 00 0a  01 00 01 8f ba ba 00 00   .../.5.. ........
00b0  ff 01 00 01 00 00 00 00  1d 00 1b 00 00 18 77 77   ........ ......ww
00c0  77 2e 67 6f 6f 67 6c 65  61 64 73 65 72 76 69 63   w.google adservic
00d0  65 73 2e 63 6f 6d 00 17  00 00 00 23 00 e4 a4 bd   es.com.. ...#....
00e0  fa 67 16 c3 36 b6 dd 93  3d 39 a6 a2 31 a0 65 99   .g..6... =9..1.e.
00f0  6a 97 9b 0f ea 50 c2 3c  19 69 6e d5 e4 60 b7 57   j....P.< .in..`.W
0100  9c b3 61 8c 6f c3 42 b9  76 b2 c3 67 63 be d0 46   ..a.o.B. v..gc..F
0110  99 3e 85 67 b5 72 7a 42  00 4e ef f1 62 55 19 21   .>.g.rzB .N..bU.!
0120  ec dd 86 8c 90 45 f2 2c  7c f2 96 d7 f4 34 31 34   .....E., |....414
0130  47 38 70 13 de 54 75 00  0d 74 86 a6 72 42 ca d2   G8p..Tu. .t..rB..
0140  9b 13 f1 7c 83 cb c9 48  6c 6e 4f 34 84 ee 68 bb   ...|...H lnO4..h.
0150  17 3a 9f 5b c2 c9 8d 3f  28 19 6c 1b df cc 2e a8   .:.[...? (.l.....
0160  e6 84 03 eb 9c df 98 4e  0b 9b 56 5e cf 20 be cb   .......N ..V^. ..
0170  4d 03 29 16 46 73 f2 49  77 a2 58 a7 97 ad e7 c0   M.).Fs.I w.X.....
```

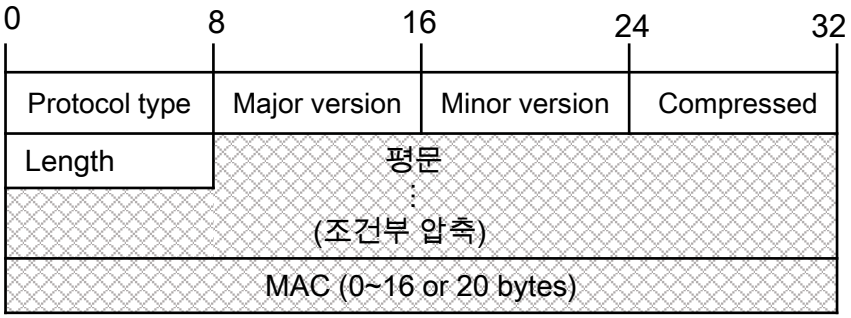
전송 계층 보안 (TLS)

- Cipher Suite (암호 그룹)
  - TLS에서 사용되는 Cipher Suite 정리 표

Cipher Suite	Key exchange	Encryption	Hash
TLS_NULL_WITH_NULL_NULL	NULL	NULL	NULL
TLS_RSA_WITH_NULL_MD5	RSA	NULL	MD5
TLS_RSA_WITH_NULL_SHA	RSA	NULL	SHA-1
TLS_RSA_WITH_RC4_128_MD5	RSA	RC4	MD5
TLS_RSA_WITH_RC4_128_SHA	RSA	RC4	SHA-1
TLS_RSA_WITH_IDEA_CBC_SHA	RSA	IDEA	SHA-1
TLS_RSA_WITH_DES_CBC_SHA	RSA	DES	SHA-1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	3DES	SHA-1
TLS_DH_anon_WITH_RC4_128_MD5	DH_anon	RC4	MD5
TLS_DH_anon_WITH_DES_CBC_SHA	DH_anon	DES	SHA-1
TLS_DH_anon_WITH_3DES_EDE_CBC_SHA	DH_anon	3DES	SHA-1
TLS_DHE_RSA_WITH_DES_CBC_SHA	DHE_RSA	DES	SHA-1
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	DHE_RSA	3DES	SHA-1
TLS_DHE_DSS_WITH_DES_CBC_SHA	DHE_DSS	DES	SHA-1
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DHE_DSS	3DES	SHA-1
TLS_DH_RSA_WITH_DES_CBC_SHA	DH_RSA	DES	SHA-1
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	DH_RSA	3DES	SHA-1
TLS_DH_DSS_WITH_DES_CBC_SHA	DH_DSS	DES	SHA-1
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	DH_DSS	3DES	SHA-1

전송 계층 보안 (TLS)

- SSL과의 다른 점 (1/7)



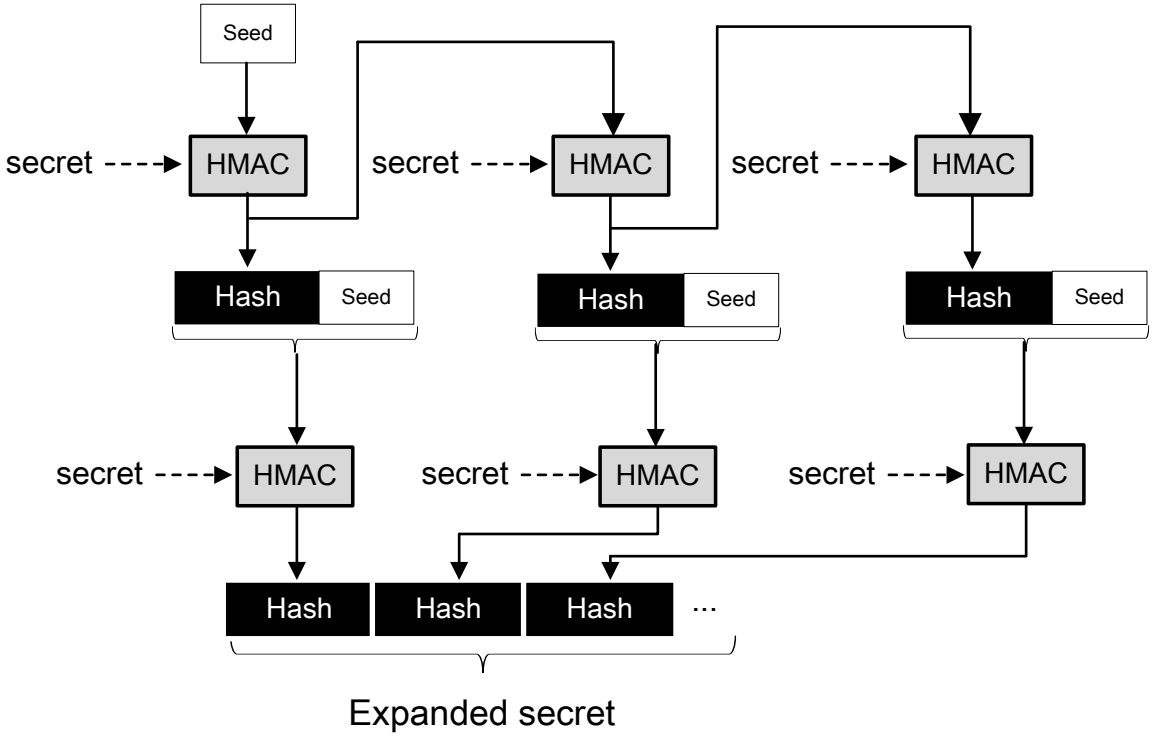
1. TLS 레코드 형식
   - SSL의 레코드 형식과 동일
     - 부 버전: 1로 표기

2. 암호 도구
   - 키 교환 알고리즘: fortezza만 제외하고 SSL과 동일
   - 대칭 암호 알고리즘: fortezza만 제외하고 SSL과 동일

전송 계층 보안 (TLS)

- SSL과의 다른 점 (2/7)

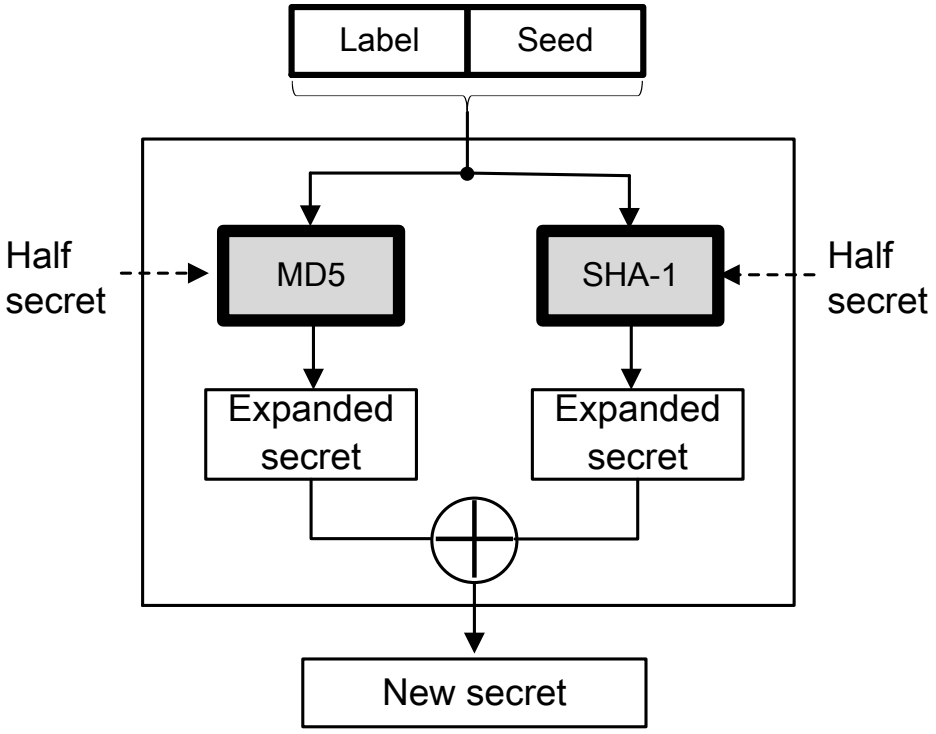
3. 데이터-확장 함수
   - 비밀을 보다 길게 확장하기 위해 미리 정의된 HMAC을 사용
   - 데이터-확장 함수 생성 과정 그림



전송 계층 보안 (TLS)

- SSL과의 다른 점 (3/7)

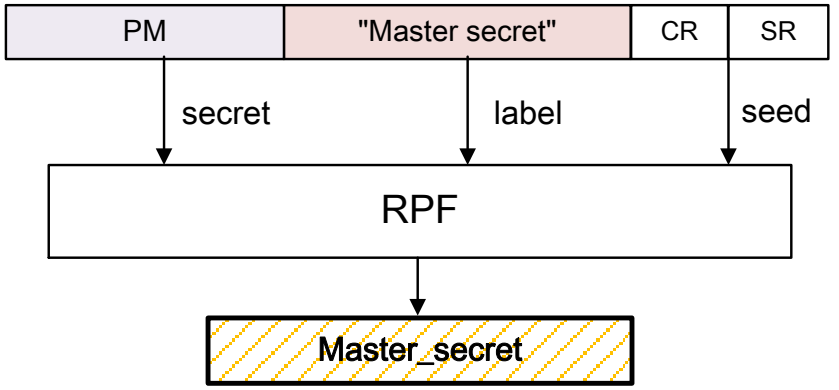
4. PRF(Rseudo Random Function) 사용
   - TLS는 PRF가 MD5/SHA-1와 데이터-확장 함수가 결합됨
   - PRF 개념 그림

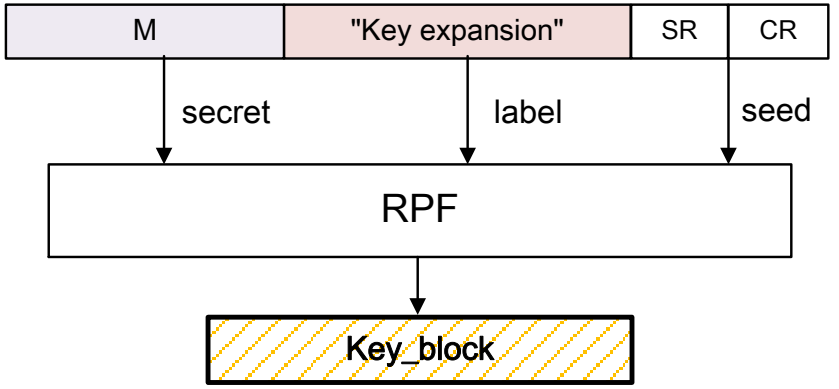


전송 계층 보안 (TLS)

- SSL과의 다른 점 (4/7)

5. Master Secret 암호 계산
   - Pre_master_secret 계산
     - SSLv3과 동일
   - Master_secret, Key_block 계산
     - 해시 함수 적용
       - 충분한 출력이 나올 때까지 수행

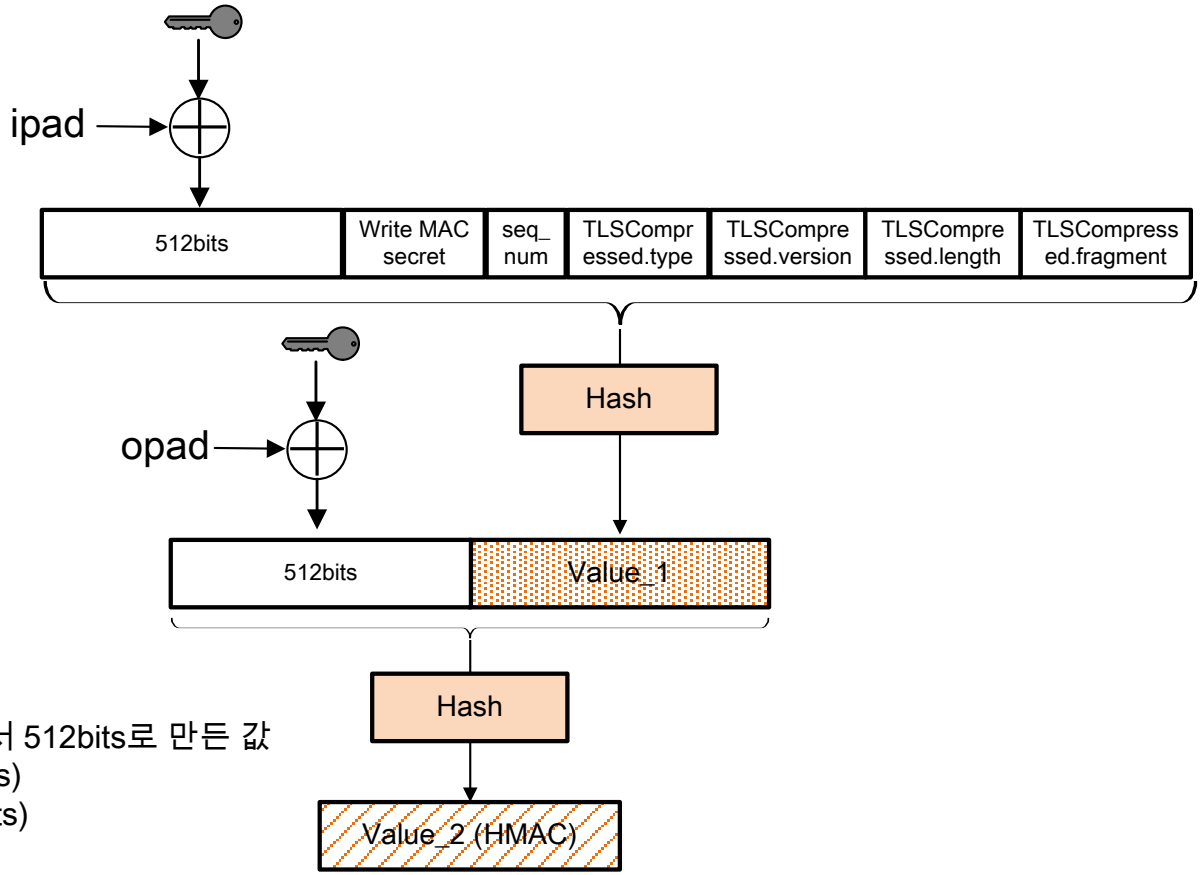




전송 계층 보안 (TLS)

- SSL과의 다른 점 (5/7)

6. 메시지 인증 코드
   - RFC 2104에 정의된 HMAC을 사용
   - 생성 과정 그림



- = MAC secret를 왼쪽에 0들을 패딩해서 512bits로 만든 값
- ipad = 0x36(00110110)을 64번 반복 (512bits)
- opad = 0x5C(01011100)을 64번 반복 (512bits)

전송 계층 보안 (TLS)

- SSL과의 다른 점 (6/7)

7. 유연한 패딩 가능
   - 암호 블록 길이의 배수만 되면 되므로 패딩의 길이 예상하기 어려움 (최대 255bytes)

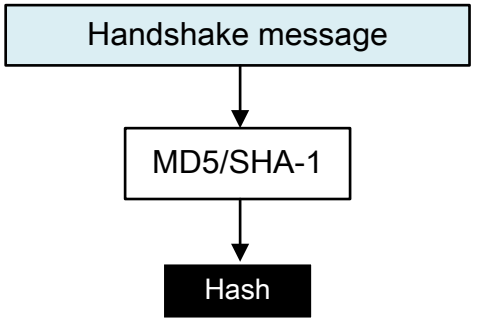
8. 경고 코드

유형	의미
Record_overflow	2^{14} + 2048bytes이 초과하는 페이로드를 수신하거나 복호화했을 때 그보다 큰 경우
User_canceled	프로토콜의 실패와는 무관하게 다른 이유로 핸드셰이크를 취소하는 경고
Access_denied	정확한 인증서가 수신되었지만 접근통제를 하면 송신자가 협상을 수행하지 않기로 결정했을 때 수행하는 경고
No_renegotiation	송신자가 재협상을 할 수 없다는 것을 나타내는 경고
Internal_error	내부 오류로 인해 연결을 계속할 수 없다는 것을 나타내는 경고
Insuffcient_security	더 안전한 암호를 요구하는 경우 협상을 실패해 handshake_failure 대신에 전송하는 경고

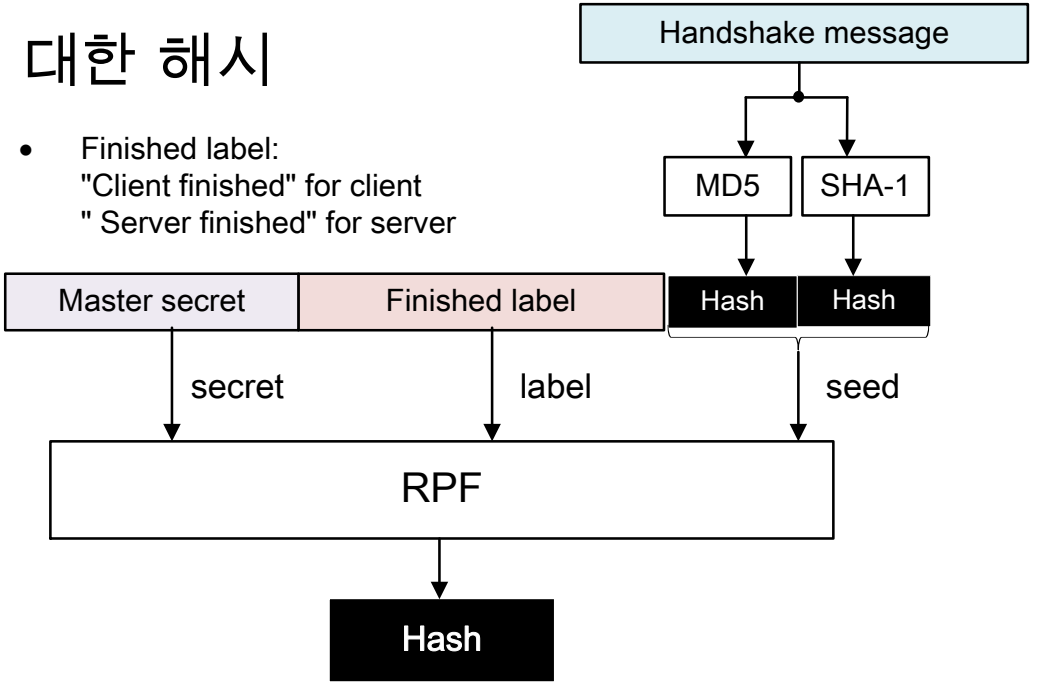
전송 계층 보안 (TLS)

- SSL과의 다른 점 (7/7)

9. Handshake 프로토콜
   - certificate_verify 메시지
     - TLS에서 인증서 검증 메시지에 대한 해시

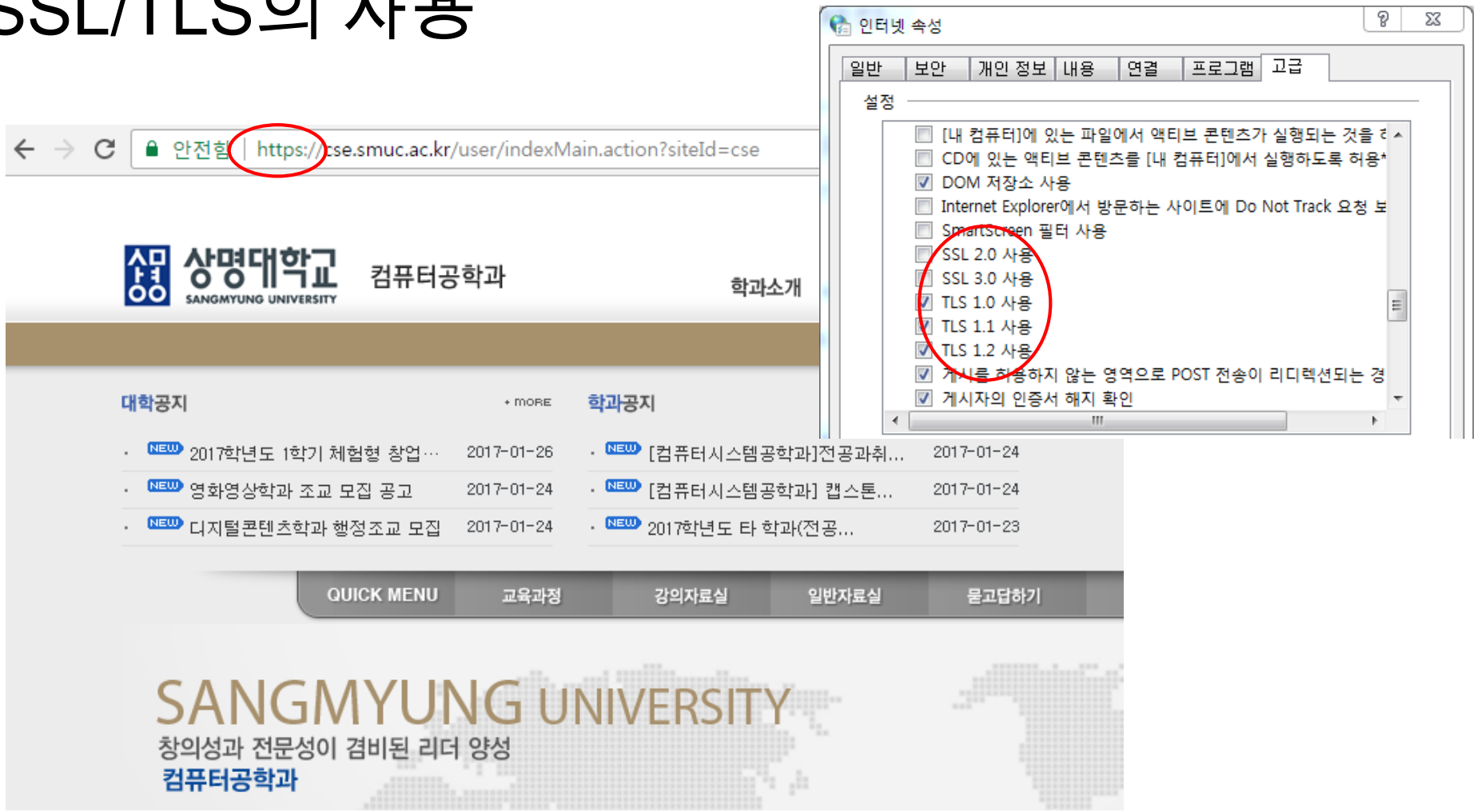


   - Finished 메시지
     - TLS에서 Finished 메시지에 대한 해시



전송 계층 보안 (TLS)

- SSL/TLS의 사용



목 차

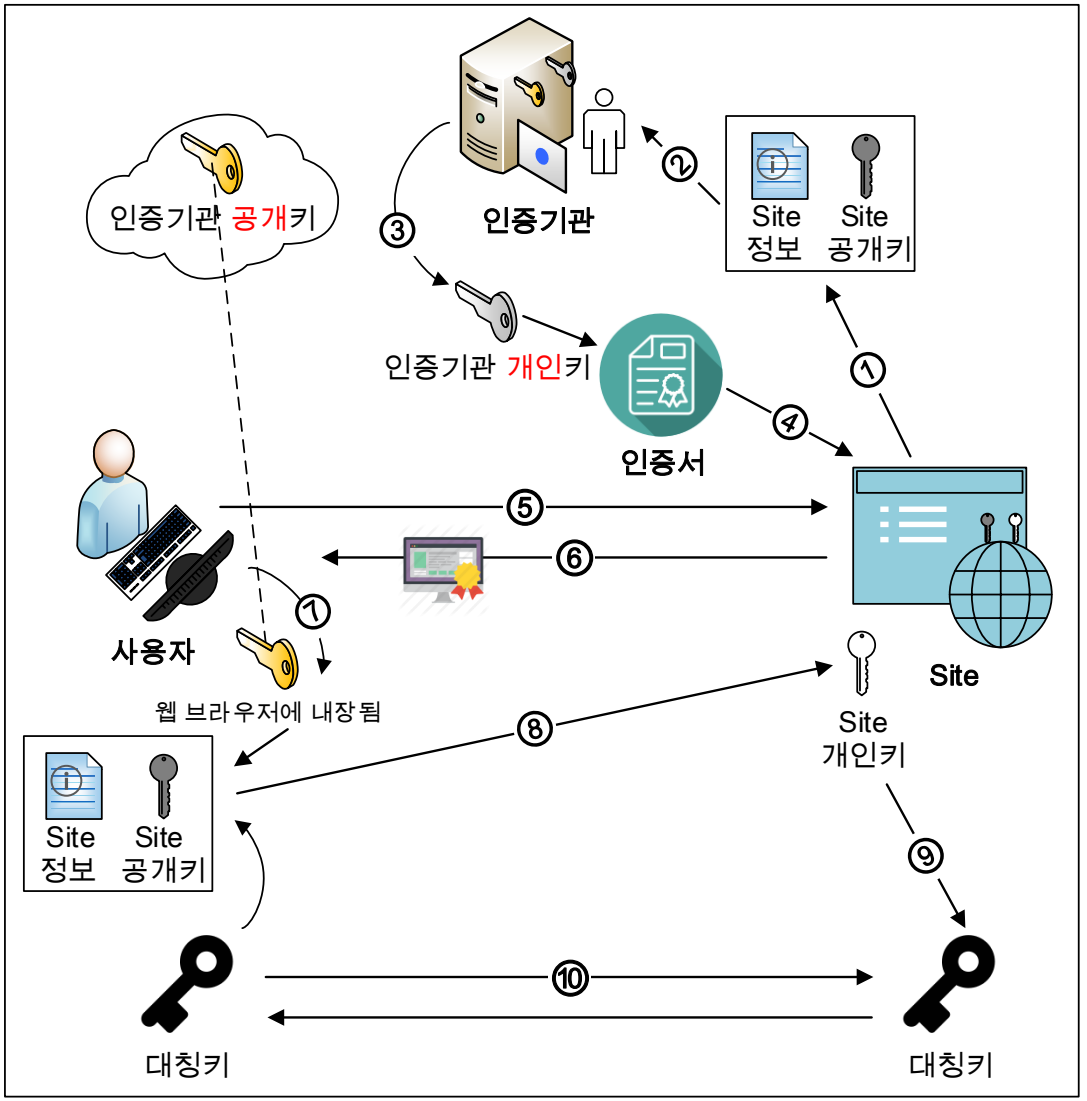
- 웹 보안
- 안전 소켓 계층 (SSL)
- 전송 계층 보안 (TLS)
- HTTPS
- SSH

HTTPS

- HTTPS (HTTP Over TLS/SSL)
  - 개요
    - 등장배경
      - 웹 브라우저와 서버간 안전한 통신을 구현하기 위함
        - Http는 단순 텍스트로 메시지가 교환됨
    - SSL/TLS와 HTTP의 결합
      - HTTPS 기능은 현재 모든 웹 브라우저에 내장
        - 웹 서버에 따라 다름
      - HTTPS의 포트는 443 사용 (HTTP 80)
        - SSL 호출
      - URL이 http:// -> https:// 로 시작

HTTPS

- HTTPS (HTTP Over TLS/SSL)
  - 개요
    - HTTPS 그림



HTTPS

- HTTPS (HTTP Over TLS/SSL)
  - 개요
    - 안전한 Cipher Suite
      - https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml
    - 우리 연구실 웹 서버는?
      - https://www.ssllabs.com/ssltest/analyze.html?d=pel.smuc.ac.kr

HTTPS

- HTTPS (HTTP Over TLS/SSL)
  - 개요
    - HTTPS의 암호화 요소
      1. 요청 문서 URL
      2. 문서의 내용
      3. 브라우저 양식 내용
      4. 브라우저가 송신한 쿠키, 서버가 송신한 쿠키
      5. HTTP 헤더의 내용

HTTPS

- HTTPS (HTTP Over TLS/SSL)
  - 기본 동작(1/2)
    - 연결 개시
      - 클라이언트는 SSL/TLS을 통해 연결 요청 메시지를 서버에게 연결을 요청함
        - TLS 클라이언트와 TLS 서버 사이의 세션이 설립
        - TLS의 요청은 TCP 개체간의 연결을 설립함으로써 시작
      - TLS 시작을 위해 client_hello 전송
      - TLS 핸드셰이크 완료 후 HTTP 요청시작
        - 모든 HTTP 데이터는 TLS 응용 데이터로서 전송됨
      - 이후 일반적인 HTTP 동작

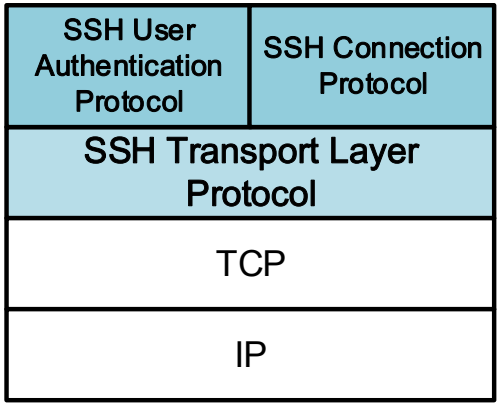
- HTTPS (HTTP Over TLS/SSL)
  - 기본 동작(2/2)
    - 연결 종료
      - 종료 전에 close_notify 경보를 교환해야 함
        - close_notify와 connection:close 지시자를 보내지 않고 종료되면 비정상 종료되었다고 함
      - 비정상 종료: 보안 경고(warning)를 발령해야 함
        - 하위 TCP 연결이 TLS 수준 연결 종료 보다 먼저 발생된 상황
        - 서버 프로그램 오류나 제 3자의 공격이 원인이 됨

목 차

- 웹 보안
- 안전 소켓 계층 (SSL)
- 전송 계층 보안 (TLS)
- HTTPS
- SSH

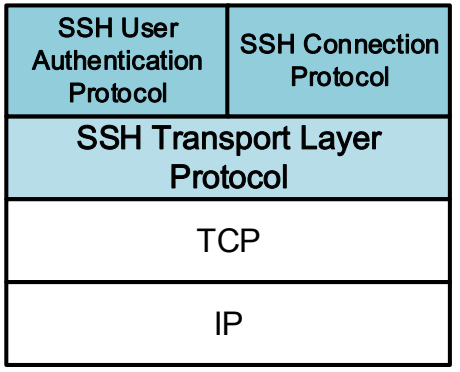
SSH

- SSH (Secure Shell)
  - 개요
    - 정의
      - 네트워크에서 다른 컴퓨터에 로그온, 원격 명령 등을 다른 시스템에 안전한 데이터 전송을 보장하는 네트워크 통신 프로토콜
      - 클라이언트/서버 구조의 TCP 보안 채널 제공
        - 포트번호: 22
    - 암호화되지 않은 기존의 Telnet 등을 대체하기 위해 설계
    - SSH 프로토콜 스택 그림



SSH

- SSH (Secure Shell)
  - 프로토콜의 구성
    1. 전송 계층 프로토콜(Transport Layer Protocol)
       - 서버의 인증, 데이터 기밀성/무결성을 제공
         - Perfect Forward Secrecy(PFS)을 만족함
       - 옵션으로 압축 제공



    2. 사용자 인증 프로토콜
       - 서버에게 사용자를 인증
    3. 연결 프로토콜
       - 논리채널 다중화
         - SSH 연결을 통해서 1개의 암호화된 터널 상에서 여러 개의 논리적 통신 채널을 다중화

SSH

1. 전송 계층 프로토콜 (Transport Layer Protocol)

- 특징
  - 서버 인증과 암호화를 통해 데이터 기밀성, 무결성을 보장
  - 암호화에 사용할 알고리즘 협상, 키 교환, 암/복호화 담당

① 동작 초기
- TCP 연결 수립
- 신원 확인용 스트링 교환
  - 클라이언트/서버 식별 문자열 교환
    - e.g., SSH-2.0-billsSSH_3.6.3q3<CR>

1. 전송 계층 프로토콜 (Transport Layer Protocol)

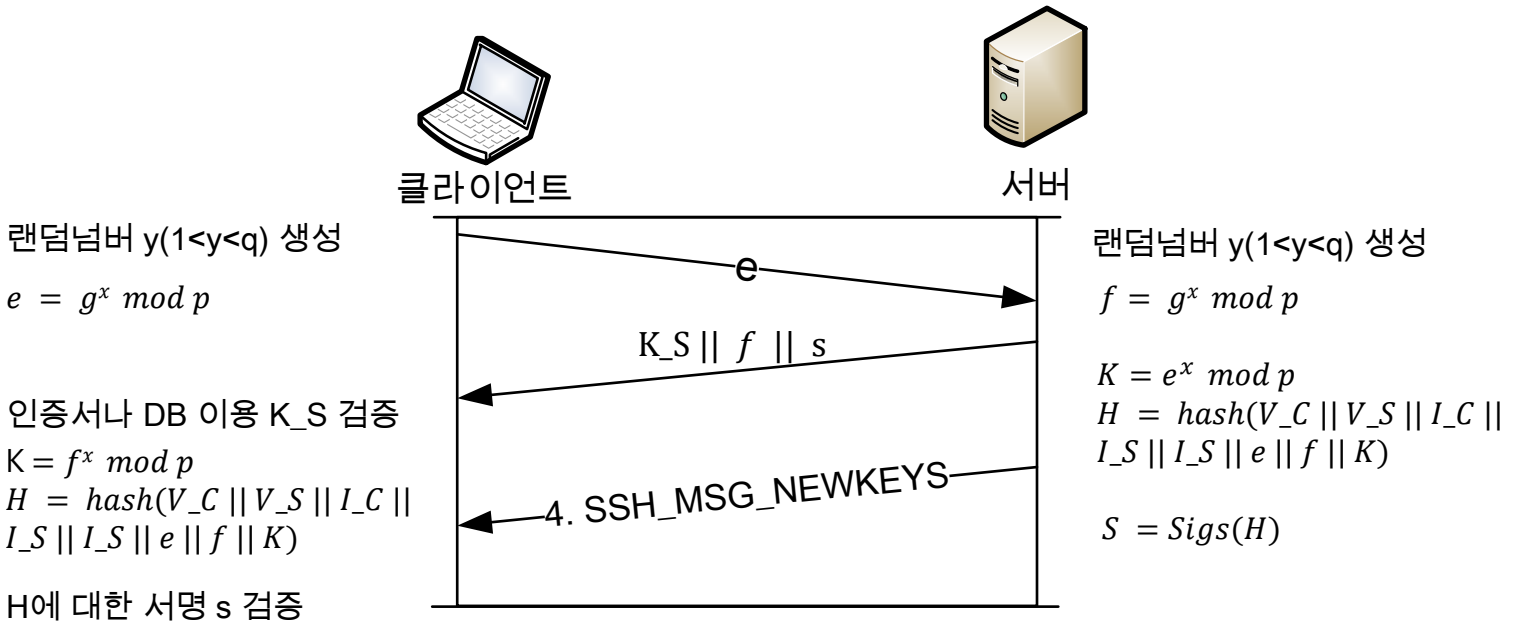
② 알고리즘 협상

- 지원가능한 알고리즘을 선호도 순으로 정렬한 목록을 전송
  - 암호 알고리즘
    - 다양한 키 길이로 CBC 모드의 DES(3DES), AES, Blowfish, Twofish, Serpent, Arcfour, Cast 등의 알고리즘 지원
  - MAC 알고리즘
    - 다양한 키 길이로 HMAC 사용을 위한 SHA-1, MD5 등의 알고리즘을 지원
  - 압축 알고리즘
    - 압축 알고리즘 zlib의 옵션을 지원

1. 전송 계층 프로토콜 (Transport Layer Protocol)

③ 협상한 키 교환 알고리즘을 이용해 키 교환 (1/2)

- 키 교환 알고리즘(Diffie-Hellman)
  - 키 교환 후 클라이언트 서버는 대칭 키 K를 공유



1. 전송 계층 프로토콜 (Transport Layer Protocol)

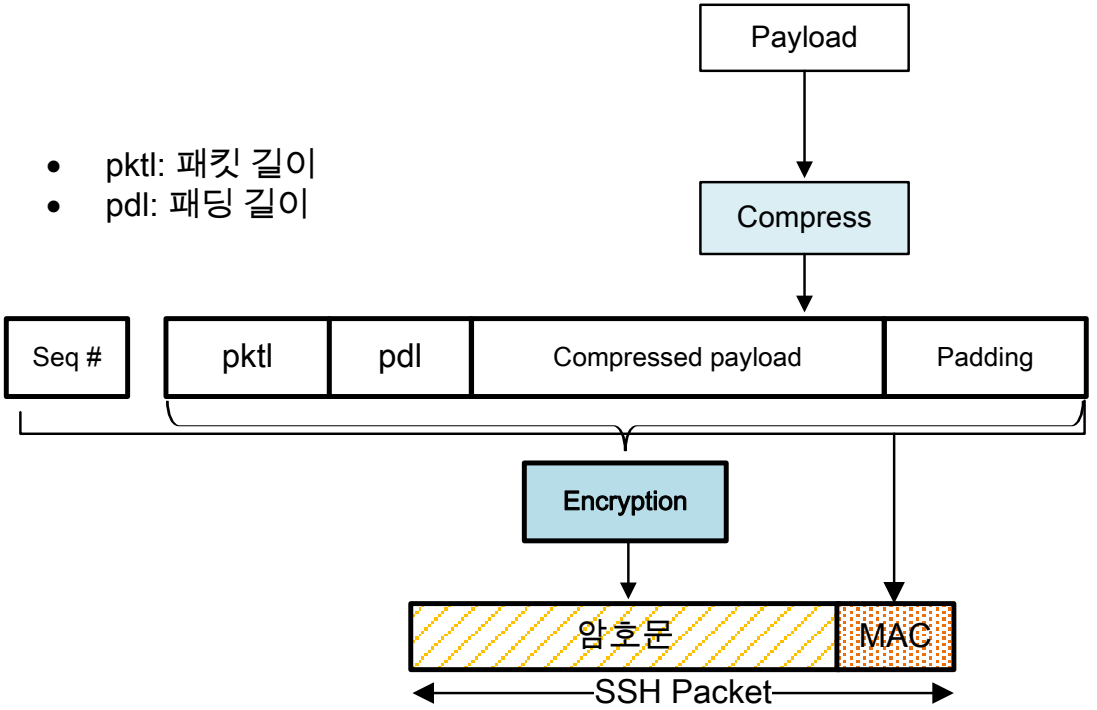
③ 협상한 키 교환 알고리즘을 이용해 키 교환 (2/2)

- 키 교환 생성(Diffie-Hellman)
  - 공유된 Master_secret K, 키교환 시 계산된 해시값 H, 세션 식별자를 이용해 암호화와 MAC에 사용할 키 생성 (키 생성 후 키 교환이 없으면 세션 식별자는 H)
    - 클라이언트가 서버에게 보내는 초기 IV: H(K || H || ‘A’ || session_id)
    - 서버가 클라이언트에게 보내는 초기 IV: H(K || H || ‘B’ || session_id)
    - 클라이언트가 서버에게 보내는 암호화 키: H(K || H || ‘C’ || session_id)
    - 서버가 클라이언트에게 보내는 암호화 키: H(K || H || ‘D’ || session_id)
    - 클라이언트가 서버에게 보내는 무결성 키: H(K || H || ‘E’ || session_id)
    - 서버가 클라이언트에게 보내는 무결성 키: H(K || H || ‘F’ || session_id)

1. 전송 계층 프로토콜 (Transport Layer Protocol)

④ 서비스 요청

- 프로토콜 수립 요청과 수립 이후의 패킷 교환
  - 패킷 형식 그림

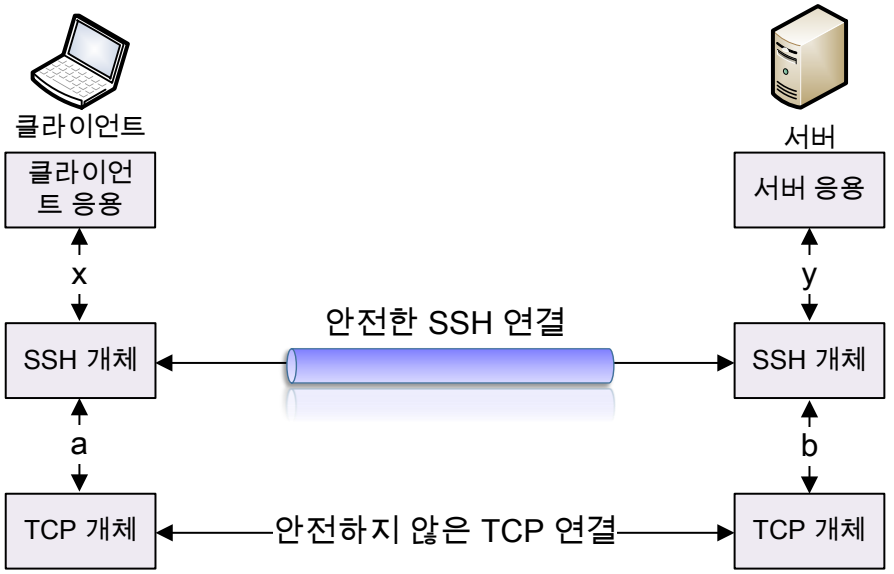


2. 사용자 인증 프로토콜 (User Authentication Protocol)

- 서버가 클라이언트에게 인증요청 하는 것
- 사용자 인증 방법
  - 비밀번호 인증
    - 사용자 ID-패스워드 인증
  - 공개 키 인증
    - 공개키 인증서 기법을 사용한 인증
  - 호스트 기반
    - 호스트가 인증(공개키) 되어있고 호스트에서 클라이언트를 인증해 주면 서버는 호스트를 믿고 클라이언트에게 서비스 제공

3. 연결 프로토콜 (Connection Protocol)

- SSH의 안전한 인증 연결이 된 전송 계층 프로토콜 상에서 수행
  - 안전한 인증 연결(터널) = 전송 계층이 수립되고 이미 사용자 인증 프로토콜이 수행된 이후
  - 안전한 터널의 다수의 논리 채널로 다중화



3. 연결 프로토콜 (Connection Protocol)

- 채널 메커니즘
  - 채널 오픈
    - 채널에 로컬번호를 할당 후 메시지 전송
    - 원격지에서 채널 개시 할 수 있다면, SSH_MSG_CHANNEL_OPEN_CONFIRMATION 메시지 반환
    - 채널 개시 실패 시 원인 코드와 SSH_MSG_CHANNEL_OPEN_FAILURE 메시지 반환
  - 데이터 전송
    - SSH_MSG_CHANNEL_DATA 메시지를 전송
    - 채널이 열려있으면 양 방향 전송임
  - 채널 종료
    - 한 쪽이 종료를 원하면 수신자 채널 정보가 포함된 SSH_MSG_CHANNEL_CLOSE 전송

감사합니다!