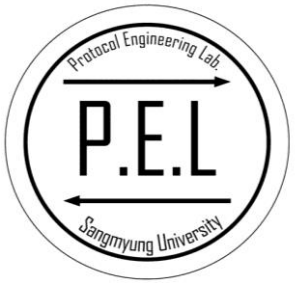


NETWORK SECURITY ESSENTIALS

- WAP -

Boo-Hyung Lee
(boohyung@pel.smuc.ac.kr)
Protocol **E**ngineering **L**ab., **Sangmyung** University

2015. 02. 26

Content

- WAP의 개요
- WML
- WAP 프로그래밍 모델
- WAP의 구조
- WTLS
- WAP 종단-대-종단 보안

WAP의 개요

- **WAP의 정의**

  - Wireless Application Protocol의 약자
  - Phone.com, Ericcson, Nokia, Motorola가 주축이 된 WAP Forum이 개발
  - 1998년 WAP 1.0발표, 2001년 WAP 2.0발표
  - Mobile Device로 무선 환경에서 인터넷 서비스를 이용할 수 있도록 하는 프로토콜

- **WAP이 필요한 이유**

  - 무선 네트워크의 특성과 일치

  1) 통신 환경 : 좁은 대역폭, 낮은 전송률

  2) 적은 Resource : 적은 메모리 및 배터리, 저사양 CPU사용

  3) 인터페이스 : 입력의 불편함, 응용프로그램의 제한

WML

- **WML의 정의**

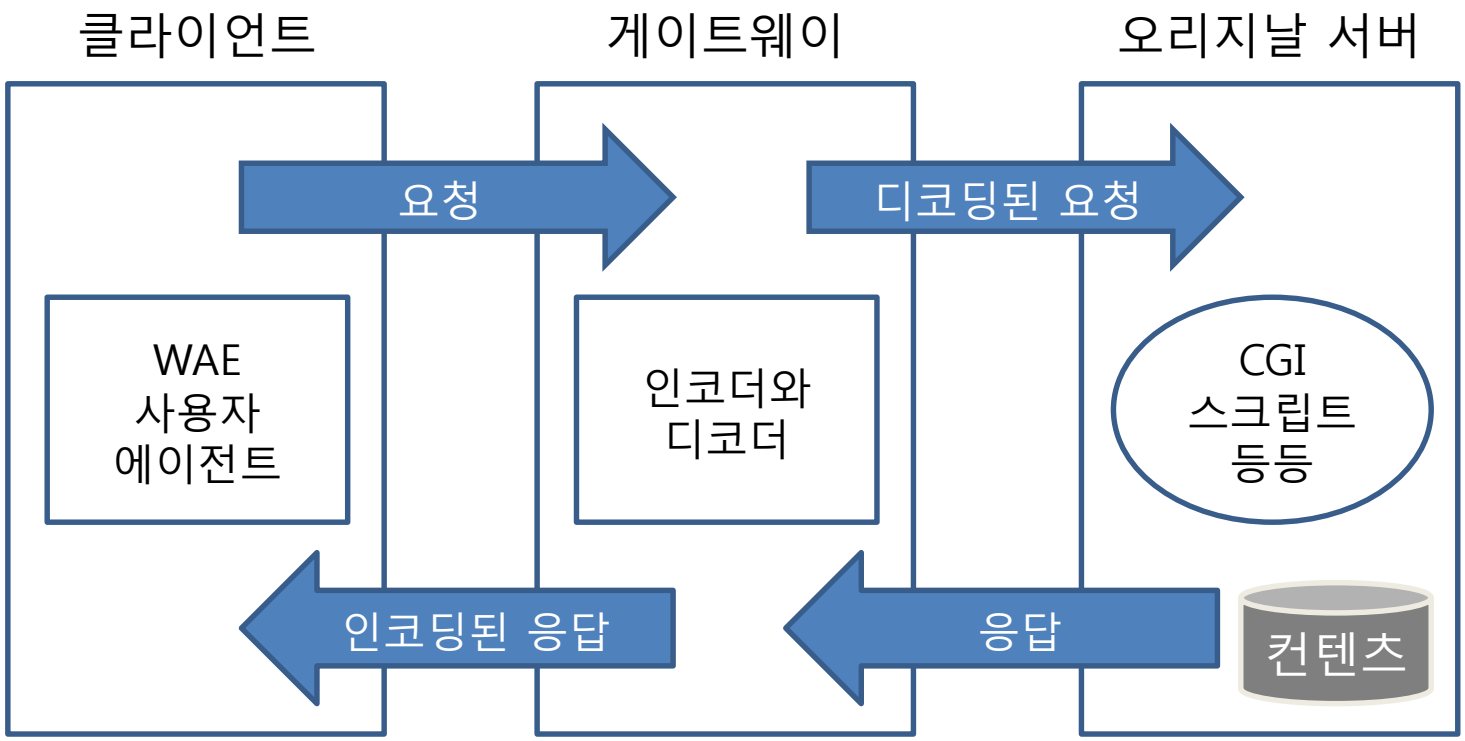
  - Wireless Markup Language의 약자

  - 제한된 사용자 입력 기능을 가진 장비(Mobile Device)에서 콘텐츠와 양식을 표현하기 위해 설계된 언어

- **HTML vs WML**

	HTML	WML
사용	Desktop	Moblie
Tag 개수	많음	적음
출력	Device와 무관	Device에 따라 다름
구성	여러 개의 Page가 하나의 Site를 구성	여러 개의 Card가 하나의 Deck을 구성

WAP 프로그래밍 모델 (1/3)



WAP 프로그래밍 모델 (2/3)

- **프로그래밍 모델 구성 요소**

  - 클라이언트 : 사용자

  - 게이트웨이 : WAP과 HTTP 사이의 변환, 인코딩/디코딩 수행; WAP(Wireless Access Point)에 위치함

  - 오리지날 서버 : 웹 서버

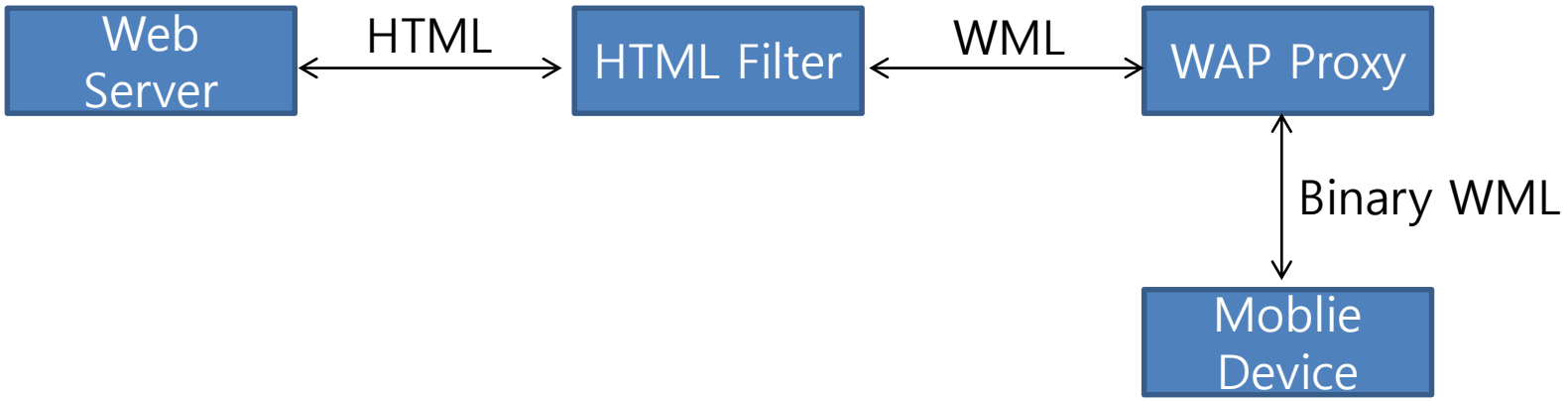
- **특징**

  - 게이트웨이와 오리지날 서버 사이의 트래픽은 HTTP로 전달

  - 무선 도메인에서는 게이트웨이가 프록시 서버로도 사용

WAP 프로그래밍 모델 (3/3)

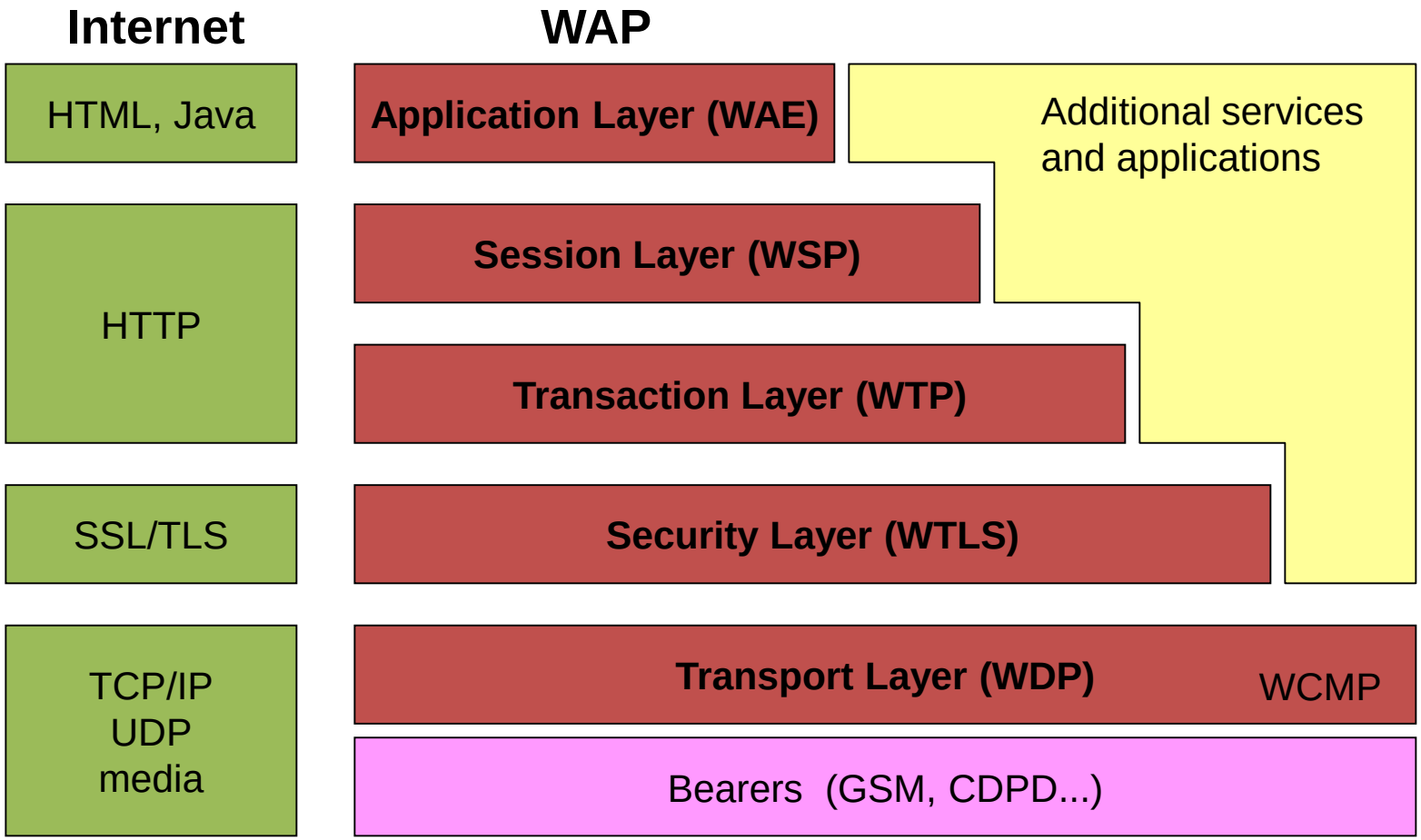
- **WAP 환경에서의 언어 변환**



- HTML Filter : HTML 컨텐츠를 WML 컨텐츠로 변환
- WAP Proxy : WML 컨텐츠를 이진 WML 컨텐츠로 변환하여 사용자에게 전달

WAP의 구조 (1/2)

- **WAP Stack**



WAP의 구조 (2/2)

- **무선 프로토콜의 종류**

  - WAE(Wireless Application Environment) : 무선 환경에 대한 어플리케이션 개발 환경 규격; 응용 프로그램과 장비 개발을 쉽게 하기 위한 도구와 형식의 집합
  - WSP(Wireless Session Protocol) : 세션을 정의하고, 관리하는 기능을 제공
  - WTP(Wireless Transaction Protocol) : 신뢰할 수 있는 데이터 전송 기능; TCP와 유사한 기능 제공
  - WTLS(Wireless Transport Layer Security) : 무선 보안 프로토콜; SSL/TLS와 유사한 기능
  - WDP(Wireless Datagram Protocol) : UDP와 유사한 기능

WTLS (1/18)

- **WTLS의 정의**

  - Mobile Device(클라이언트)와 WAP Gateway 간의 보안 서비스를 제공하는 보안 프로토콜
  - 웹 보안에 사용하는 표준 보안 프로토콜인 TLS에 기초
  - WAP Gateway와 Web Server 사이의 보안은 TLS를 사용

- **WTLS의 기능**

  - 데이터 무결성 : 메시지 인증
  - 기밀성 : 암호화
  - 인증 : 인증서를 통한 상호 인증

WTLS (2/18)

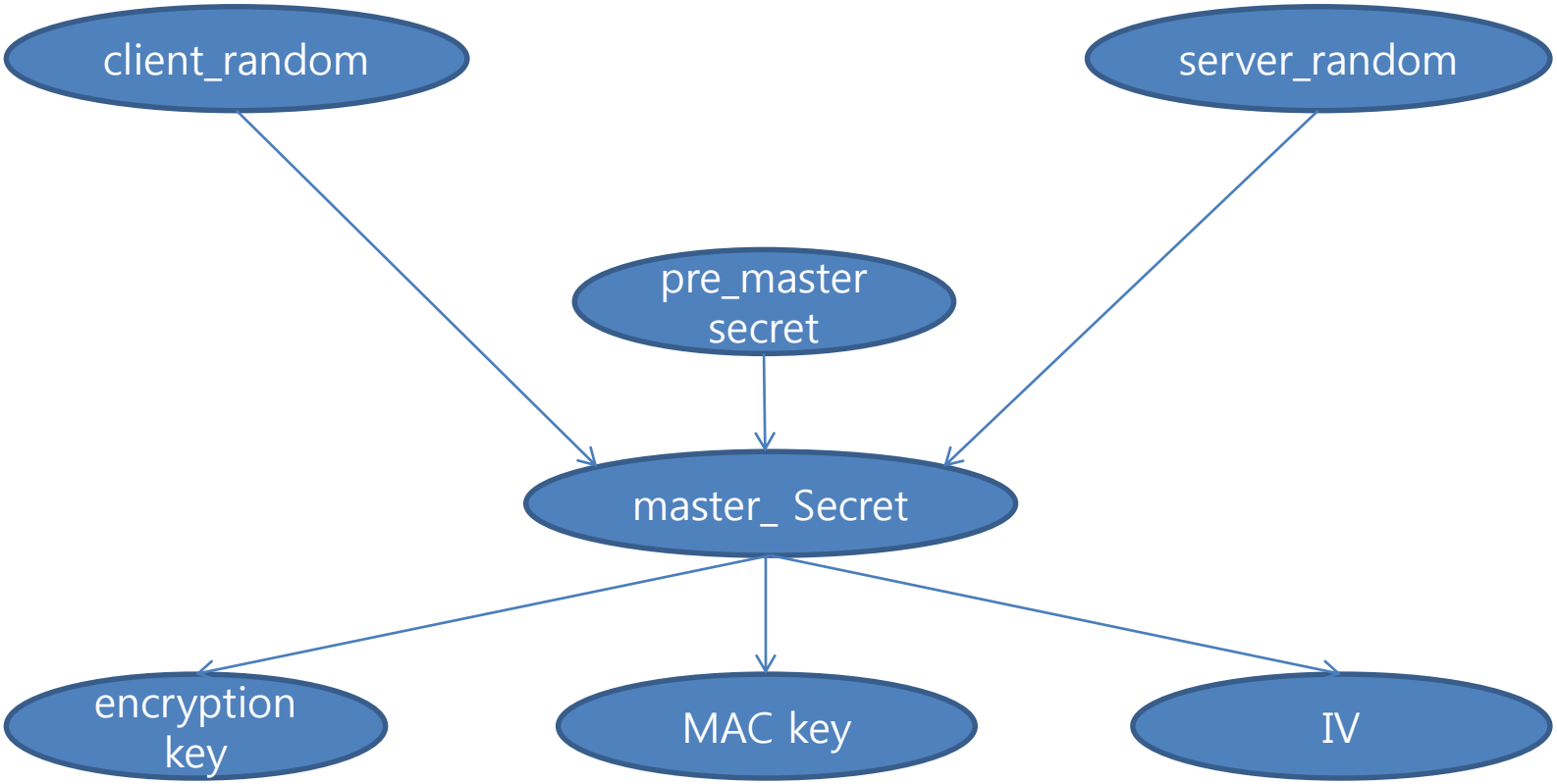
- **WTLS에서 사용하는 파라미터**

  - 프로토콜 버전(Protocol Version) : WTLS 프로토콜 버전 번호
  - 암호명세(CipherSpec) : 암호화 알고리즘, MAC 계산용 해시 알고리즘, 알고리즘에 따른 속성 값
  - 키 갱신(Key Refresh) : 암호화 키, MAC 비밀키, IV가 얼마나 자주 계산되는지를 정의
  - 클라이언트 난수(client_random) : 클라이언트가 제공하는 16비트 랜덤 값
  - 서버 난수(server_random) : 서버가 제공하는 16비트 랜덤 값
  - 사전 마스터 비밀(pre_master_secret) : 클라이언트가 만드는 20바이트 랜덤 값
  - 마스터 비밀(master_secret) : 클라이언트와 서버가 공유하는 20바이트 비밀 값

    → 클라이언트 난수, 서버 난수, 사전 마스터 비밀로 만듬
  - 클라이언트/서버 암호화 키(client/server encryption key)
  - 클라이언트/서버 MAC 비밀키(client/server MAC key)
  - 클라이언트/서버 IV(client/server IV)

WTLS (3/18)

- 파라미터 생성 (1/2)



WTLS (4/18)

- **파라미터 생성 (2/2)**

  - master_secret = PRF(pre_master_secret, “master secret”, client_random + server_random) [0 .. 19];

  - key_block = PRF(master_secret, expansion_label, seq_num + client_random + server_random);

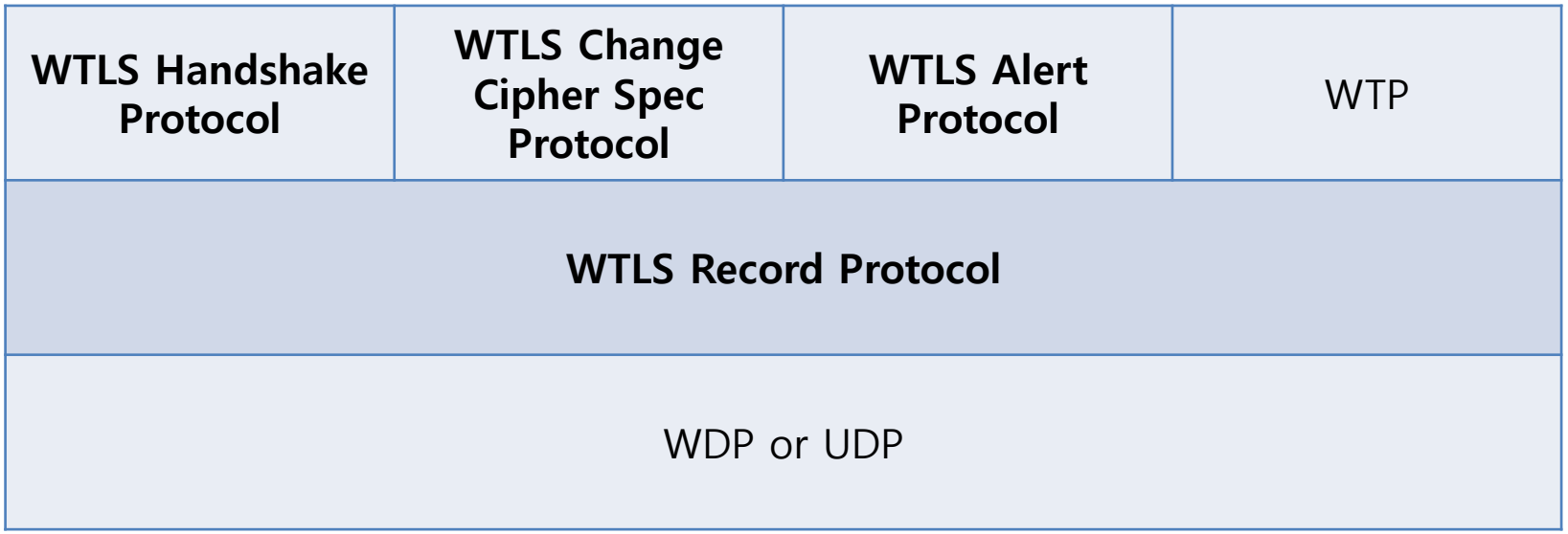
  - encryption key, MAC key, IV는 key_block을 통해 생성되고, key_refresh에서 정한 frequency에 따라 재생성

☞ expansion_label : key_block을 만들 때 사용되는 상수

☞ seq_num : key_refresh에 따라 달라지는 변수; 첫 번째 seq_num = 0

WTLS (5/18)

- **WTLS의 구조**

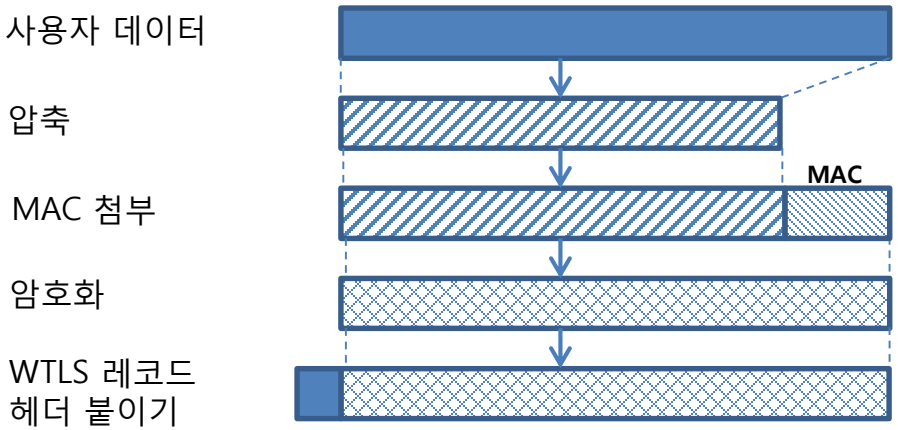


WTLS (6/18)

- **WTLS Record Protocol**

  - 기밀성 제공 : 암호화
  - 무결성 제공 : MAC

  - 전반적인 동작과정



☞ MAC 첨부 : HMAC 알고리즘 사용 (MD-5나 SHA-1)

☞ 암호화 : DES, 3DES, RC5, IDEA

WTLS (7/18)

- **WTLS Record Header (1/2)**

  - 레코드 종류(8비트)

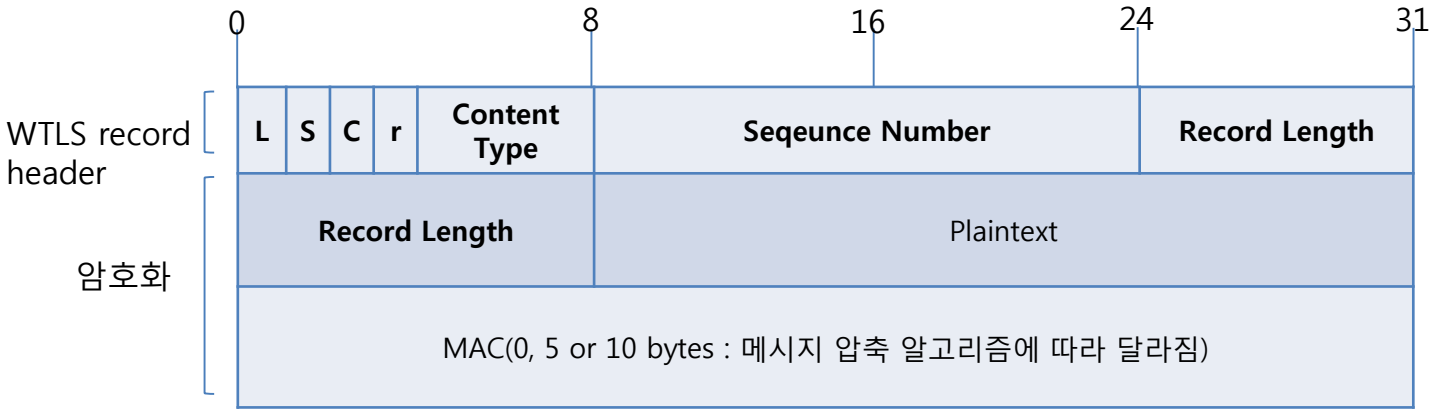
    1) L : 레코드 길이 필드 지시자(레코드 길이 필드의 사용 유무)

    2) S : 순서번호 길이 필드 지시자(다음 레코드 순서번호 필드의 사용 유무)

    3) C : 암호 스펙 지시자(압축 알고리즘의 사용 유무)

    4) r : 예약된 필드

    5) Content type : ChangeCipherSpec(1), Alert(2), Handshake(3), Application(4)

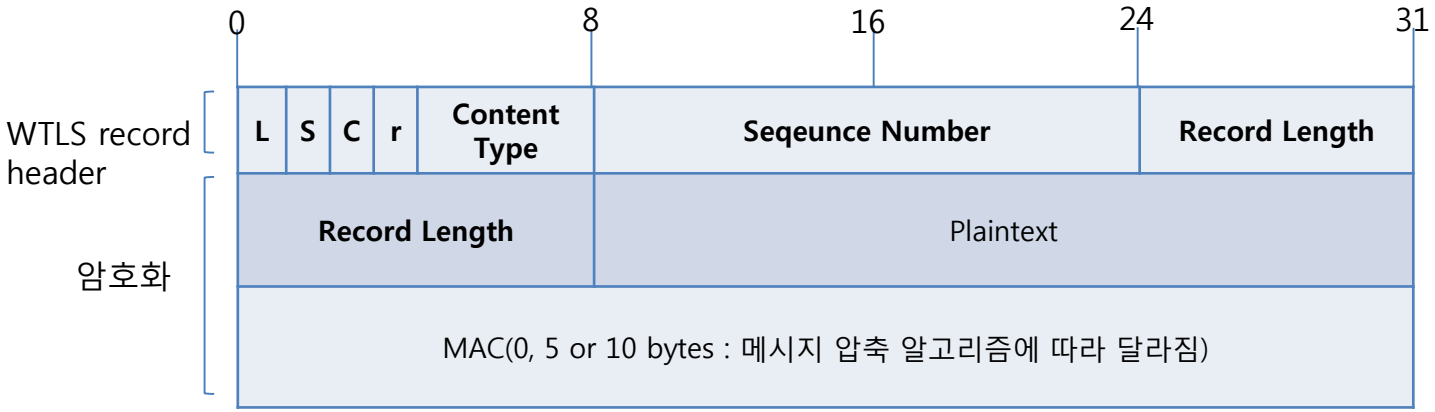


WTLS (8/18)

- **WTLS Record Header (2/2)**

  - 순서 번호(16비트) : 현재 레코드의 순서 번호; 레코드 사이의 구분을 위한 장치

  - 레코드 길이(16비트) : 평문(압축을 사용하는 경우 압축된 평문) 데이터의 바이트 단위 길이



WTLS (9/18)

- **WTLS Change Cipher Spec Protocol**

  - WTLS-지정 프로토콜 중 하나

  - Handshake 프로토콜에 의해 협상된 압축, MAC, 암호화에 쓰이는 알고리즘이 이후부터 적용됨을 수신자에게 알리는 역할

  - 1 byte이며, 값 1을 갖는 한 개의 메시지로 구성됨

WTLS (10/18)

- **WTLS Alert Protocol (1/2)**

  - 암호 오류, 압축 오류, 메시지 인증 오류, 인증 실패 등의 에러 발생을 수신자에게 알리는 역할

  - 메시지는 2 byte로 구성

    1) 첫번째 바이트(Level) : warning(1) or critical(2) or fatal(3)

    2) 두번째 바이트(Description) : 세부적인 에러코드

       Handshake, Change cipher spec, Record Protocol 수행 중 발생하는 오류메시지를 표현

  - 경고메시지는 압축되고 암호화

WTLS (11/18)

- **WTLS Alert Protocol (2/2)**

- Description의 예

1) session_close_notify : 송신자가 현재 세션을 통해서 메시지를 보내지 않겠다고 수신자에게 알림

2) unexpected_message : 예상하지 못한 메시지가 수신되었음

3) bad_record_mac : 부정확한 MAC이 수신됨

4) decompression_failure : 압축해제가 불가능하거나, 압축해제의 결과가 허용된 크기보다 큼

5) handshake_failure : 핸드셰이크 메시지의 한 필드가 허용된 값의 범위를 벗어남

6) connection_close_notify : 송신자가 현재 연결 상태에서는 메시지를 보내지 않겠다고 수신자에게 알림

7) bad_certificate : 수신한 인증서에 문제가 있음

8) unsupported_certificate : 지원되지 않는 형식의 인증서가 수신됨

9) certificated_revoked : 수신된 인증서는 취소된 인증서

10) certificated_expired : 수신된 인증서는 만료된 인증서

11) certificated_unknown : 수신된 인증서에 알 수 없는 문제가 있어 수신할 수 없음

WTLS (12/18)

- **WTLS Handshake Protocol**

  - 모든 응용 데이터를 전송하기 이전에 사용; 사용할 파라미터의 협상 역할

  - 클라이언트와 서버가 암호 통신에 사용할 공유 키와 MAC 알고리즘을 결정, 인증서를 이용한 인증

  - 클라이언트와 서버 사이의 연속된 여러 메시지 교환으로 구성

WTLS (13/18)

- **WTLS Abbreviated Handshake Protocol**

  - 매 세션 성립 시마다 인증서 및 키 교환을 하면 시간이 매우 오래 걸림

  - 최초에 세션이 성립된 상태라면, Client/ServerHello, ChangeCipherSpec, Finished 세 과정만을 진행하고 세션을 재개함

  - 과정

  1) 클라이언트가 현재 세션의 세션 ID를 이용해 ClientHello 메시지를 보냄

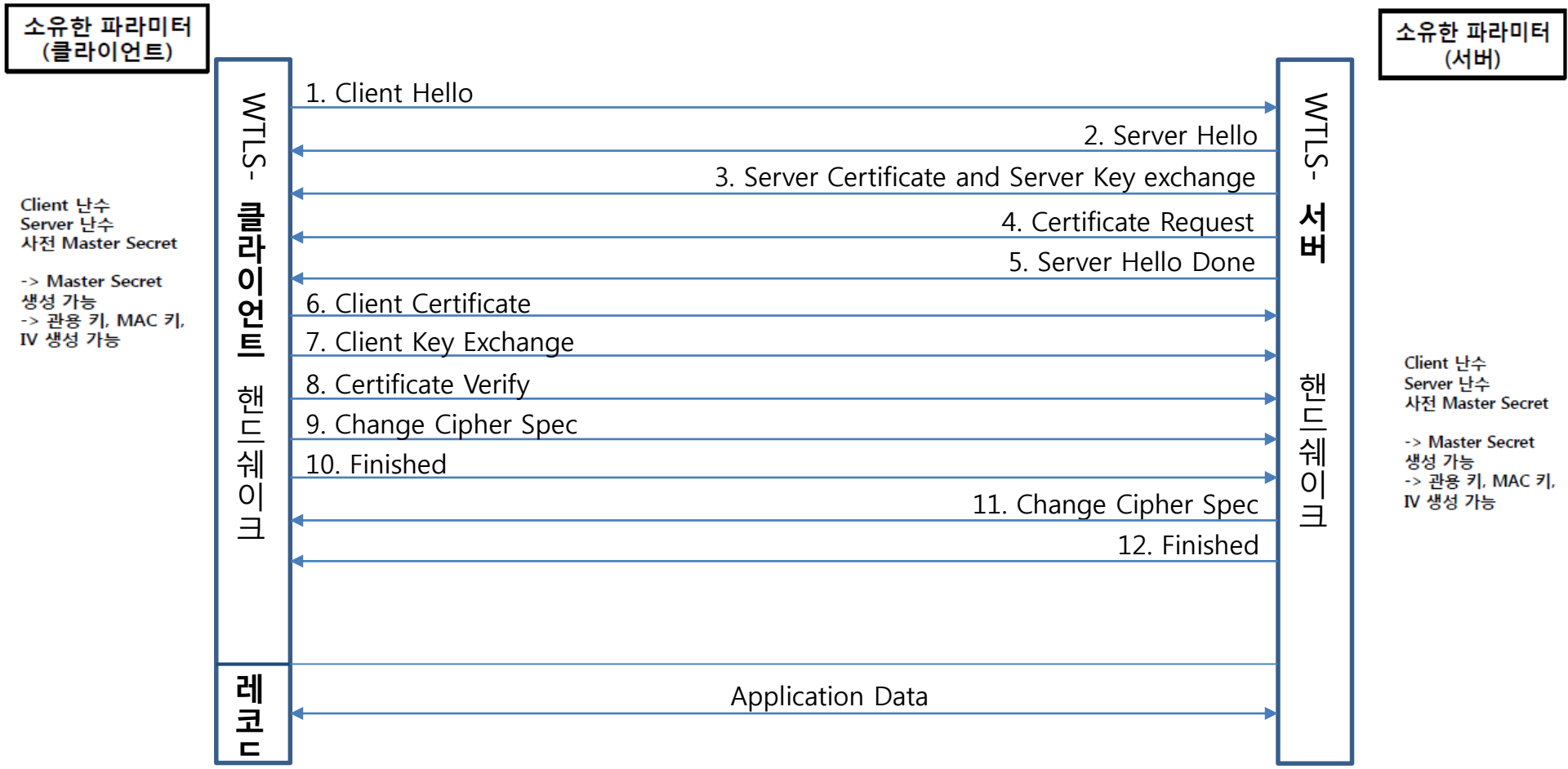
  2) 서버는 세션 캐쉬 안에 클라이언트가 보낸 세션 ID가 존재하는지 확인, 존재하면 동일한 세션 ID를 이용해 ServerHello 메시지를 보냄

  (만약, 세션 캐쉬 안에 클라이언트가 보낸 세션 ID가 존재하지 않으면 서버는 새로운 세션 ID를 만들고, 클라이언트와 서버는 Full Handshake를 수행해야 함)

  3) ChangeCipherSpec메시지와 Finished 메시지를 주고받아 암호화, 압축, MAC 알고리즘 파라미터를 결정하고 데이터 교환을 시작

WTLS (14/18)

- **WTLS Handshake Protocol 과정(최초 세션 성립; Full Handshake)**



WTLS (15/18)

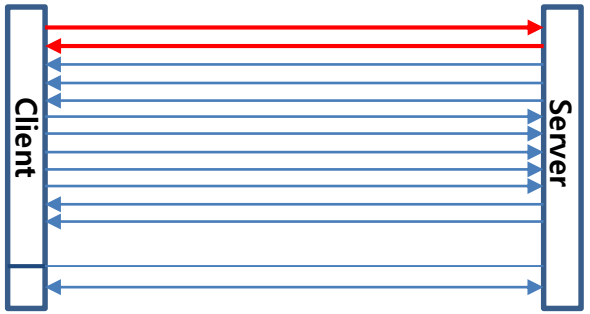
- **WTLS Handshake Protocol 과정 설명**

  - 단계 1(그림에서 1, 2번 과정) : 일종의 초기화 과정

  1) Client Hello : 클라이언트가 수용할 수 있는 WTLS 버전, 클라이언트 난수, 세션 ID, 지원 가능한 Cipher Suite 리스트와 압축방법 리스트 등의 정보를 서버에 전송

  2) Server Hello : Client Hello 메시지에 대한 응답으로 Server Hello 메시지를 전송; Server Hello 메시지 안에는 사용할 WTLS 버전, 서버 난수, 세션 ID, 클라이언트가 보낸 Cipher Suite 리스트와 압축방법 리스트 중 서버가 선택한 정보 등이 들어있음

☞ 세션 ID : 가변 길이의 세션 식별자



WTLS (16/18)

- **WTLS Handshake Protocol 과정 설명**

- 단계 2(그림에서 3, 4, 5번 과정) : 서버의 인증서를 보내고, 사전 Master Secret을 암호화하기 위한 키 교환, 클라이언트의 인증서를 요청, hello 메시지 단계를 끝내는 것을 알림

3) Server Certificate and Server Key Exchange : Server의 인증서를 전송하고 사전 Master Secret을 암호화할 때 이용될 키 교환용 공개키를 Client에게 전송

만약, Server의 인증서가 없을 경우 Key Exchange 과정 이후에 서명1을 통해 자신을 인증할 수 있음

4) Certificate Request : Client의 인증서를 요청함

5) Server Hello Done : Server의 Hello 절차가 완료되었음을 알림; 이제부터 클라이언트의 응답시작

☞ 서명1 : 클라이언트/서버 난수와 키 쌍 생성에 필요한 매개변수에 해시를 취하고, 서버의 개인키로 암호화를 하여 생성

WTLS (17/18)

- **WTLS Handshake Protocol 과정 설명**

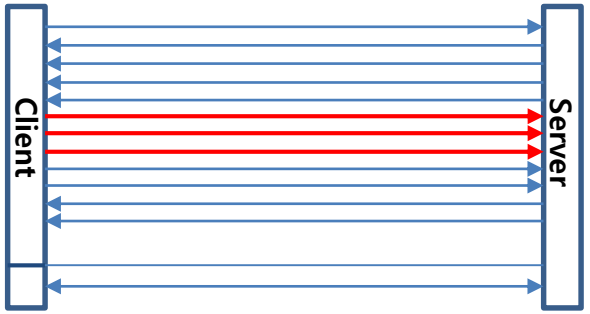
  - 단계 3(그림에서 6, 7, 8번 과정) : 클라이언트는 서버의 요청에 따라 인증서와 키 교환, 자신의 인증서에 대한 유효성 검증

  6) Client Certificate : Client의 인증서를 전송

  7) Client Key Exchange : 사전 Master Secret을 Server Key Exchange 과정에서 받은 공개키로 암호화하여 전송; 서버는 전송받은 데이터를 개인키로 복호화하여 사전 Master Secret을 얻을 수 있음

  8) Certificate Verify : Client가 자신의 인증서 유효성을 스스로 검증; 인증서에 서명2 포함

☞ 서명2 : Master Secret과 미리 정의된 상수에 해시를 취하고, 클라이언트의 개인키로 암호화



WTLS (18/18)

- **WTLS Handshake Protocol 과정 설명**

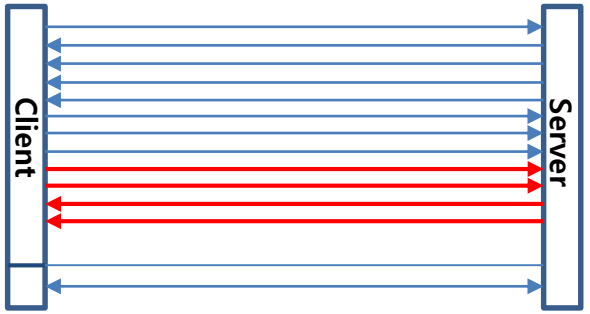
  - 단계 4(그림에서 9, 10, 11, 12번 과정) : 암호 알고리즘 정보를 서로 교환하고 프로토콜을 종료

  9) Change Cipher Spec : Server에게 Data 암호화에 사용될 알고리즘 정보 전달

  10) Finished : Server에게 완료 메시지 전송

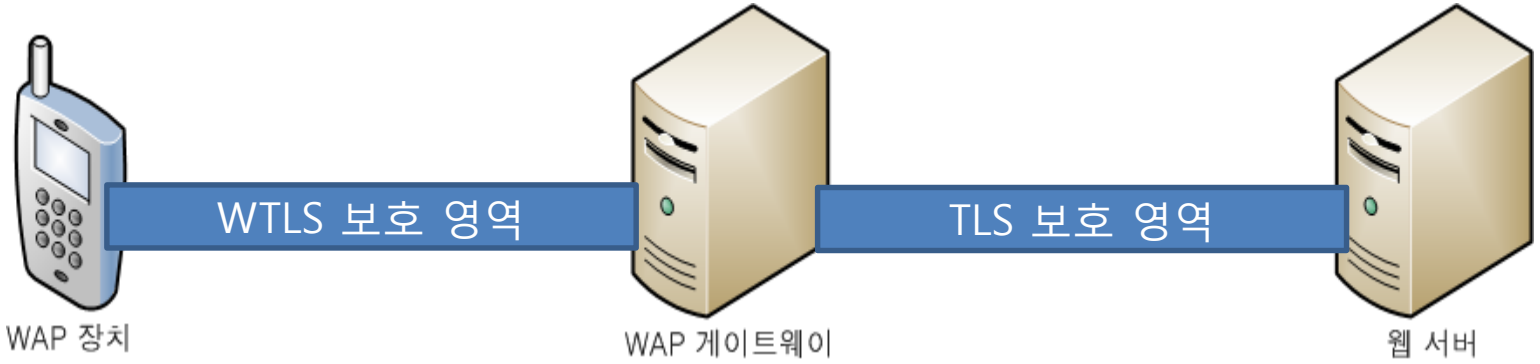
  11) Change Cipher Spec : Client에게 Data 암호화에 사용될 알고리즘 정보 전달

  12) Finished : Client에게 완료 메시지 전송



WAP 종단-대-종단 보안

- **WAP에서의 보안영역**



- WAP장치와 게이트웨이 간 : WTLS로 보호됨
- WAP 게이트웨이와 웹 서버 간 : TLS로 보호됨
- WAP 게이트웨이 안에서의 데이터는 암호화되지 않은 상태로 다루어짐

WAP 종단-대-종단 보안

- **종단-대-종단 보안을 제공하는 두 가지 방법**

  - TLS 기반 보안

    1) 두 종단 사이에 안전한 TLS 세션을 설치

    2) WAP 게이트웨이는 TCP계층의 게이트웨이로서 동작

    → 게이트웨이를 통과하는 동안 TCP 데이터는 암호화된 상태로 남아있기 때문에 종단 대 종단 보안이 유지될 수 있음

  - IPSec 기반 보안 : Host나 게이트웨이에서 IP 계층 단의 보안인 IPSec을 사용하면 모든 과정에서 데이터가 암호화되기 때문에 종단-대-종단 보안이 제공됨