Mobile Communication

LTE Attachment

201502085 이규봉

201502094 이재호

1. 실습 개요

1. 실습 일시 및 장소

   실습은 416호실을 예약해서 2번 (2번째 과제, 3번째 과제), 일반 강의실을 예약해서 1번, 총 세 번을 진행했습니다.

   날짜는 각각 11월 30일 (강의실), 12월 2일, 12월 8일 입니다.

2. 실습 목적

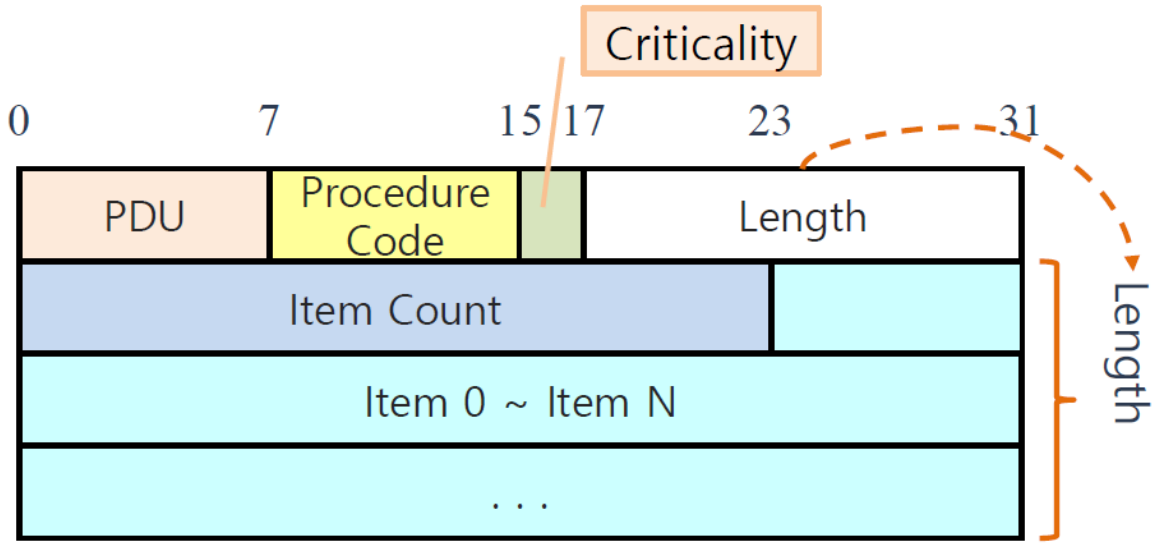
   LTE 네트워크에 대한 배경 습득 및 간단한 프로그램 구현을 통해 이론에 대한 이해도 향상

   실습을 통해 UE, eNB, MME 컴포넌트 간 시그널링을 구현해 LTE 네트워크의 접속 과정 학습

2. 프로토콜 설명

1. S1-AP

MME와 eNB 간의 시그널링 프로토콜이다.



S1-AP Header

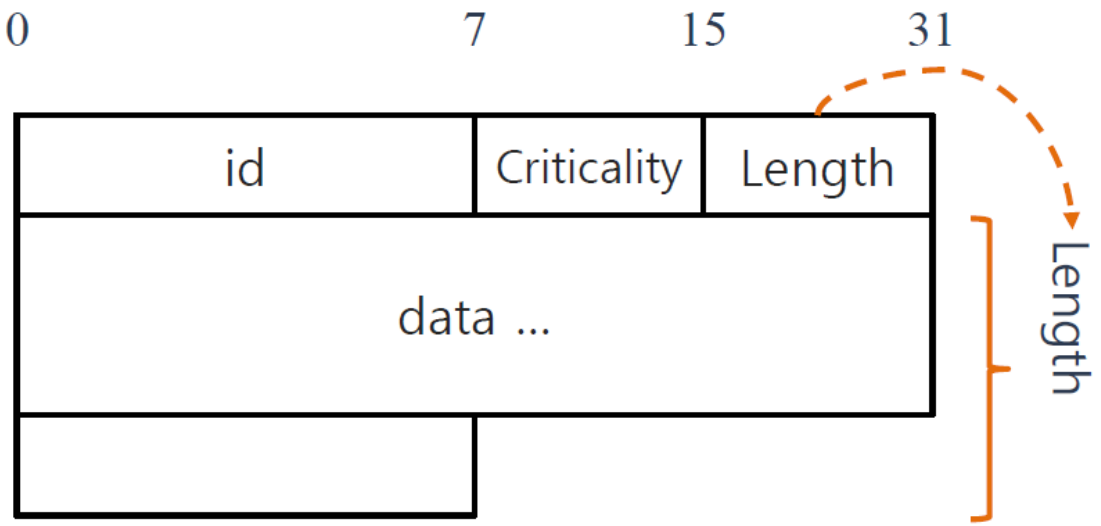
PDU : PDU의 타입을 표시

Procedure Code : 어떠한 Procedure을 진행 중인지 표시

Criticality : Criticality 정보를 표시

Length : Item Count부터 Item만큼의 길이

Item Count : Item의 개수



S1-AP Item

Id : Item의 유형을 식별하기 위한 field

Criticality : Criticality 정보를 표시

Length : Item의 data 길이

Data : Item에 들어있는 Data

2. SCTP

Bits	Bits 0 - 7	8 - 15	16 - 23	24 - 31
+0	Source port		Destination port	
32	Verification tag 0x00000000			
64	Checksum 0x00000000			
96	Chunk 1 type	Chunk 1 flags	Chunk 1 length	
128	Chunk 1 data			
...	...			
...	Chunk N type	Chunk N flags	Chunk N length	
...	Chunk N data			

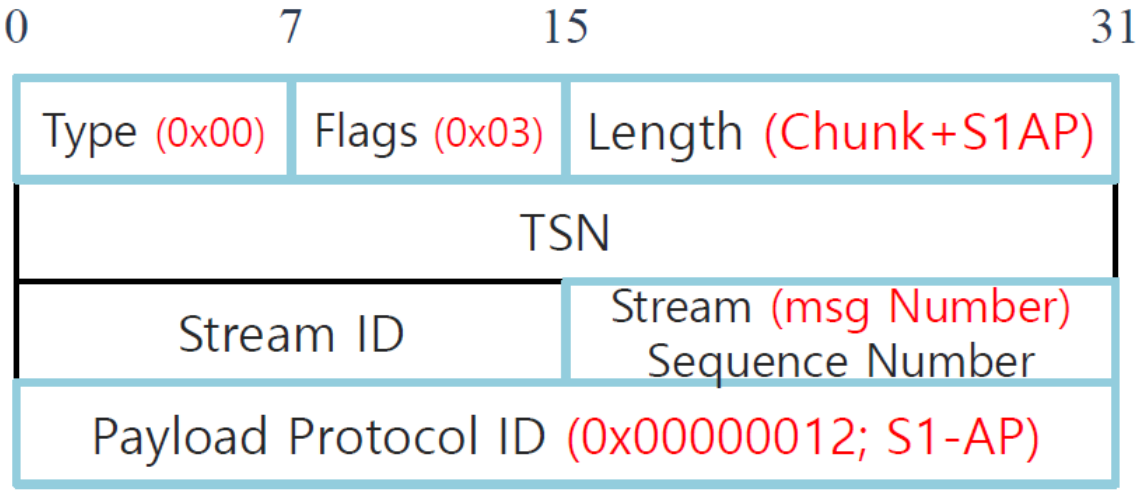
SCTP Header

Source port : 송신자의 Port 주소

Destination port : 수신자의 Port 주소

Verification tag : Association 별로 할당되는 세션 식별자

Checksum : 전체 패킷에 대한 Checksum



SCTP Chunk

Type : Chunk의 타입을 표시

Flags : I, U, B, E Chunk 플래그

Length : Chunk의 길이(최소 17)

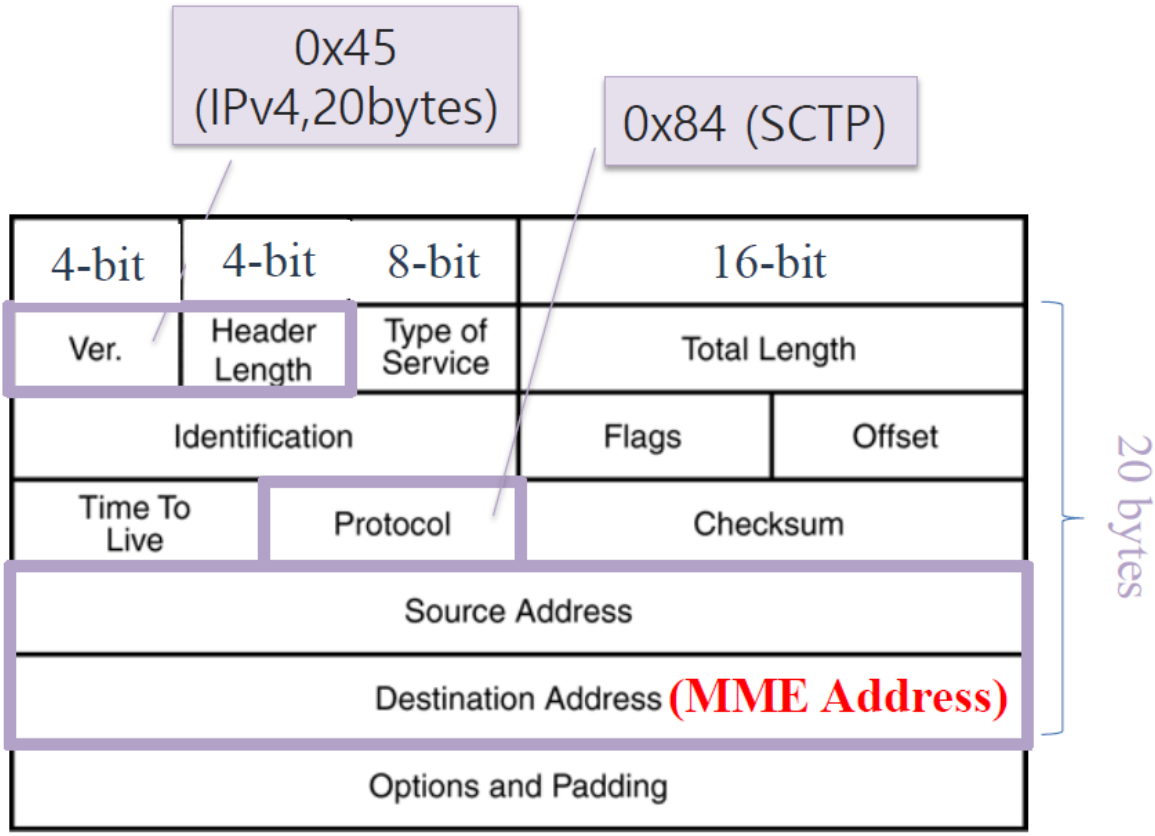
TSN : 전체 데이터 스트림을 위한 Sequence Number

Stream ID : Chunk가 어느 데이터 스트림에 속하는지 표시

SSN : 데이터 스트림 중 몇 번 스트림인가 표시

Payload Protocol ID : 어플리케이션 별 프로토콜 식별자

3. IP



Ver. : Internet Protocol의 버전을 나타내는 부분

Header Length : Header의 길이를 나타내는 부분

Type of Service : 서비스의 우선순위를 위한 부분

Total Length : 전체 IP 패킷의 길이를 나타내는 부분

Identification : Fragment가 발생했을 때, 결합을 위한 식별자를 표시하는 부분

Flags : Fragment의 여부를 표시하는 Flag

Offset : Fragment에 저장된 원래 데이터의 바이트 범위

Time To Live : 데이터가 이동할 수 있는 단계의 수

Protocol : 상위 계층 프로토콜을 표시

Checksum : 헤더의 Checksum

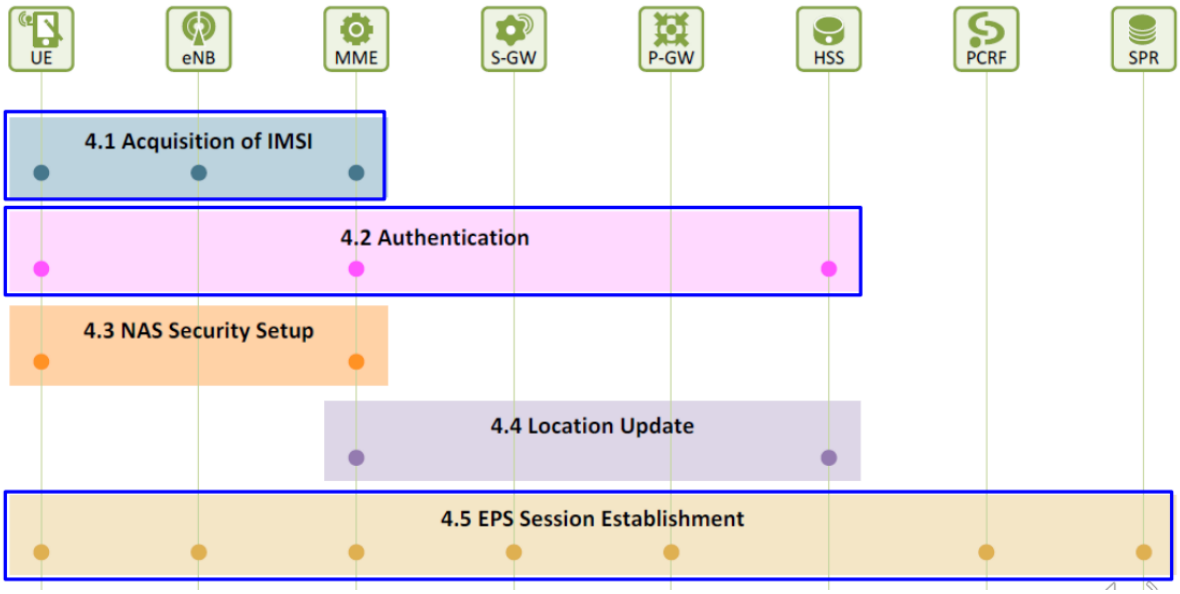
Source Address : 보내는 곳의 IP 주소

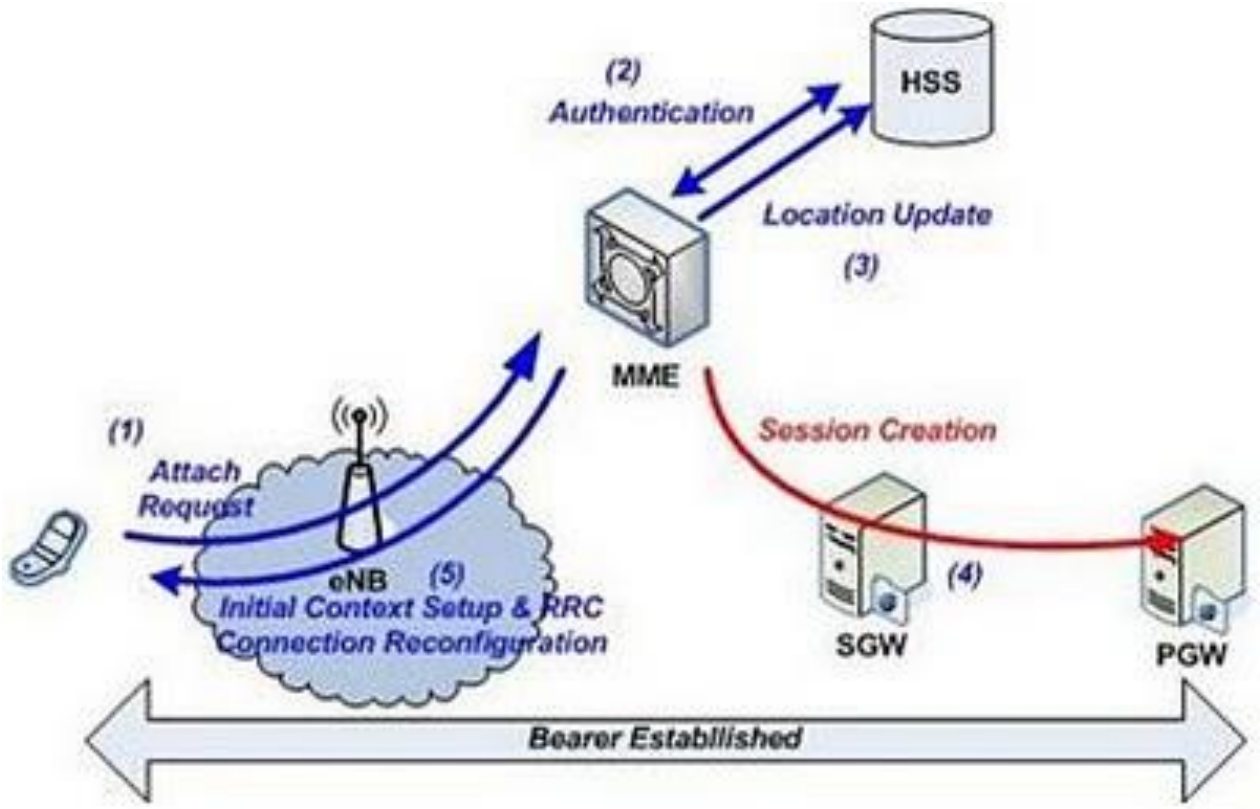
Destination Address : 도착지의 IP 주소

Options and Padding : 추가 처리 옵션을 위한 부분

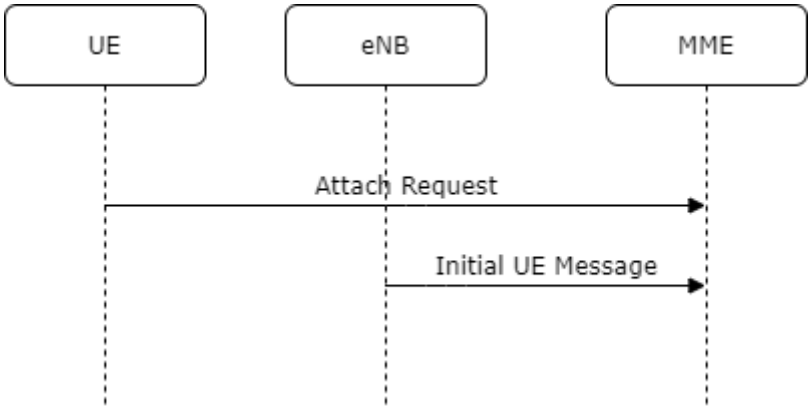
3. LTE Attachment 과정 설명

아래에선 Initial attach 과정이 어떻게 이뤄지는지에 대해 설명한다.





1. Acquisition of IMSI (4.1)



Attach request 하는 함수이다.

```
void CS1APLayer::AttachReqMsg(int nlength)
{
	m_sHeader.s1ap_pdu = S1AP_PDU_INITIAL_MESSAGE; // a type of PDU
	m_sHeader.s1ap_proc_code = S1AP_PROC_CODE_INITIAL_UE_MSG; // procedure code

	m_sHeader.s1ap_crit = 0x40;
	m_sHeader.s1ap_length = nlength + 3;

	((CSCTPLayer*)GetUnderLayer())->SetFlags(SCTP_FLAGS_B | SCTP_FLAGS_E);
	((CSCTPLayer*)GetUnderLayer())->SetSSN(0);
}
```

Attach 할 때 아이템을 붙이는 함수이다.

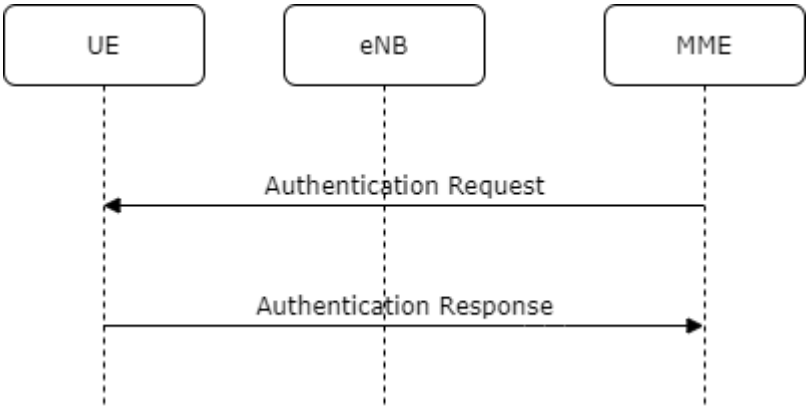
```
u_char* CPacketSenderDlg::attachReqItems()
{
	u_char temp[93] = {
		// item0 id-eNB-UE-S1AP-ID
		0x00,0x08,0x00,0x02,0x00,0x00,

		// Item1 id-NAS-PDU
		0x00,0x1a,0x00,0x35,0x34,0x17,0x7c,0x2a,0x01,0x33,0x06,0x07,0x41,0x51,0x0b,0xf6
		,0x54,0xf0,0x60,0x80,0x01,0x01,0xdb,0x00,0x15,0x3b,0x02,0xe0,0xe0,0x00,0x14,0x02
		,0x01,0xd0,0x31,0xd1,0x27,0x0d,0x80,0x00,0x0d,0x00,0x00,0x03,0x00,0x00,0x0c,0x00
		,0x00,0x01,0x00,0x52,0x54,0xf0,0x60,0x01,0xf4

		// Item2 id-TAI
		,0x00,0x43,0x00,0x06,0x00,0x54,0xf0,0x60,0x01,0xf4

		// Item3 id-EUTRAN-CGI
		,0x00,0x64,0x40,0x08,0x00,0x54,0xf0,0x60,0x00,0x00,0x10,0xe0

		// Item4 id-RRC-Establishment-Cause
		,0x00,0x86,0x40,0x01,0x30

		// padding
		,0x00,0x00,0x00
	};

	return temp;
}
```

MME가 IMSI를 획득하는 과정이다.

UE는 Attach Request 메시지를 보내며 자신의 IMSI를 전달한다.

eNB는 MME에게 UE가 어떤 Cell과 TAI에 있는지 Initial UE Message를 통해 전달한다.

2. Authentication (4.2)



Authentication request에 해당하는 함수이다.

```
void CS1APLayer::AuthenticationRepMsg(int nlength)
{
        m_sHeader.s1ap_pdu = S1AP_ID_NAS_PDU; // a type of PDU
        m_sHeader.s1ap_proc_code = S1AP_PROC_CODE_UPLINK_NAS_TRANS; // procedure code

        m_sHeader.s1ap_crit = 0x40;
        m_sHeader.s1ap_length = nlength + 3;

        ((CSCTPLayer*)GetUnderLayer())->SetFlags(SCTP_FLAGS_B | SCTP_FLAGS_E);
        ((CSCTPLayer*)GetUnderLayer())->SetSSN(2);
}
```

Authentication response에 해당하는 함수이다.

```
u_char* CPacketSenderDlg::authenticationRspItems()
{
        u_char temp[57] = {
                // item0 id-MME-UE-S1AP-ID
                0x00,0x00,0x00,0x03,0x40,0x12,0x86

                // Item1 eNB-UE-S1AP-ID
                ,0x00,0x08,0x00,0x02,0x00,0x00

                // Item2 NAS-PDU
                ,0x00,0x1a,0x00,0x12,0x11,0x17,0x38,0x6f,0x95,0x5b,0x08,0x07,0x53,0x08,0xaa,0x7a,0xdf,0x21,0x9c,0xa2,0x52,0x82

                // Item3 id-EUTRAN-CGI
                ,0x00,0x64,0x40,0x08,0x00,0x54,0xf0,0x60,0x00,0x00,0x10,0xe0

                // Item4 id-TAI
                ,0x00,0x43,0x40,0x06,0x00,0x54,0xf0,0x60,0x01,0xf4
        };

        return temp;
}
```

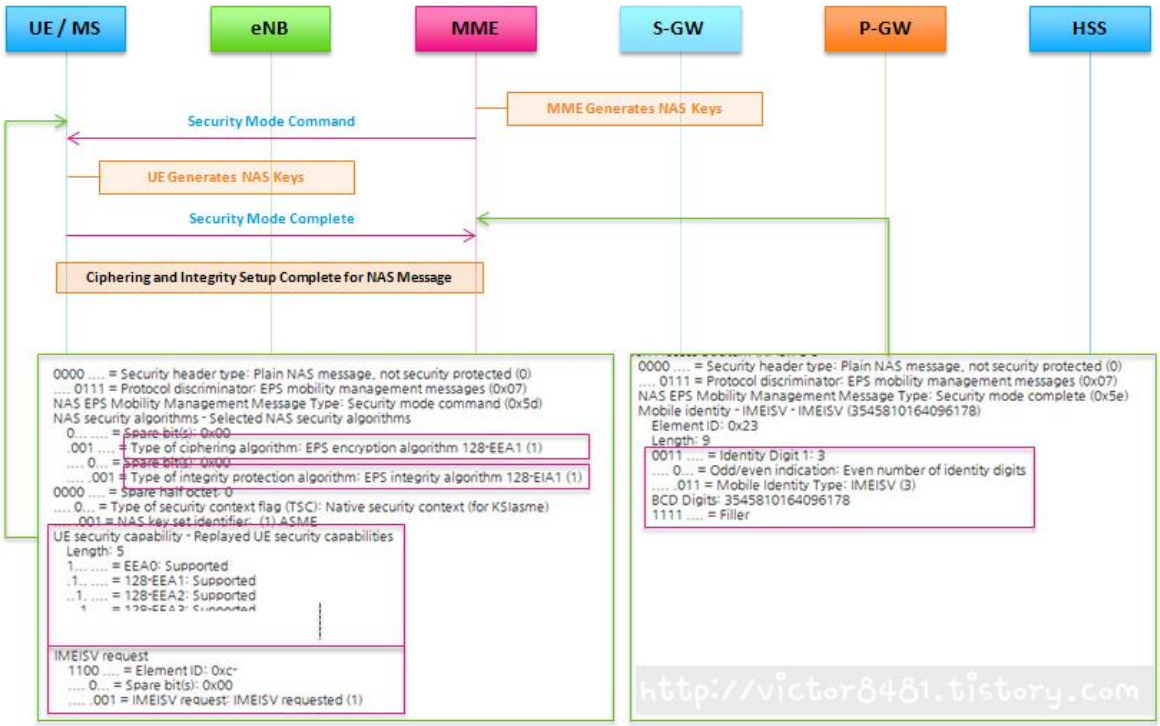
Authentication는 UE와 MME가 서로 인증하는 단계이다.

MME는 인증할 준비가 완료되면 UE에 Authentication Request를 보내 인증을 위한 정보를 전달하고, UE는 정보를 확인후 신뢰할 수 있는 망임을 확인한 뒤, 망 인증을 시도한다.

UE는 Authentication Response를 통해 인증 정보를 전달하고 MME는 자신이 확인할 수 있는 정보와 비교해서 망에 접속해도 괜찮은 단말임을 확인, 인증을 한다.

3. NAS Security Setup (4.3)

무선 구간에서 NAS 메시지를 전달할 때, 보안을 위한 Setup 과정을 진행해 NAS 메시지를 안전하게 한다. MME는 UE가 전송한 Attach Request 메시지로부터 NAS 메시지에 적용할 알고리즘을 선택하고, NAS 메시지에 적용한 키를 생성한다. 선택한 알고리즘을 Security Mode Command를 통해 UE에 알리면 UE가 NAS Security 키를 생성하게 되고, 이 키로 무결성 검사를 수행한다.

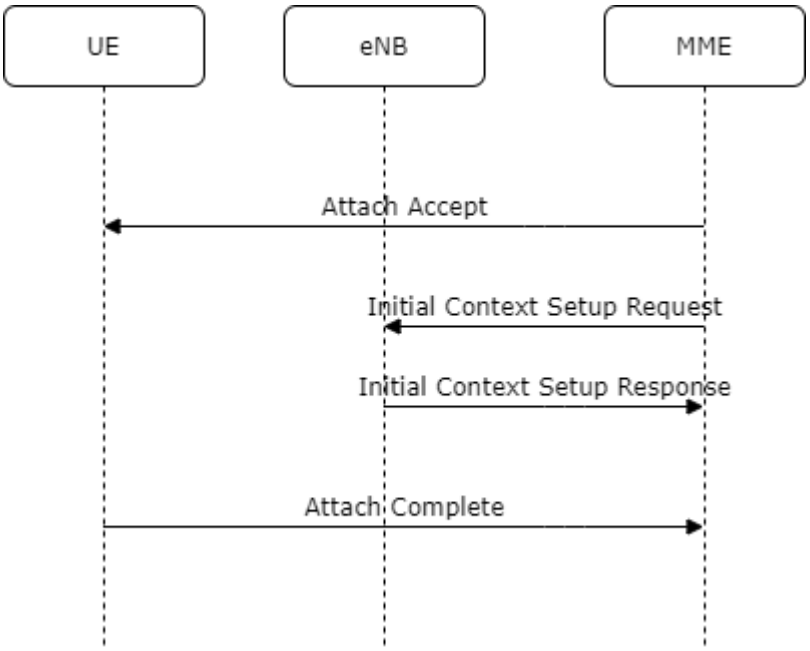


4. Location Update (4.4)

MME는 HSS에 UE의 접속 사실을 전달하고 HSS는 가입 사용자의 가입 정보를 반환한다.

HSS는 접속 정보를 받고 UE의 위치를 등록한다 그 후 MME에 UE의 QoS 정보를 전달한다.

5. EPS Session Establishment (4.5)



아래는 관련 함수들이다.

```
void CS1APLayer::attachCompleteItems(int nlength)
{
        // begin: 알맞은 값을 채우시오
        m_sHeader.s1ap_pdu = S1AP_ID_NAS_PDU; // a type of PDU
        m_sHeader.s1ap_proc_code = S1AP_PROC_CODE_UPLINK_NAS_TRANS; // procedure code
        // end

        m_sHeader.s1ap_crit = 0x40;
        m_sHeader.s1ap_length = nlength + 3;

        ((CSCTPLayer*)GetUnderLayer())->SetFlags(SCTP_FLAGS_B | SCTP_FLAGS_E);
        ((CSCTPLayer*)GetUnderLayer())->SetSSN(7);
}
```

```
u_char* CPacketSenderDlg::initialContextSetupResponseItems()
{
	// warning: length should be 32
	u_char temp[33] = {
		// item0, id-MME-UE-S1AP-ID (7)
		0x00,0x00,0x00,0x03,0x40,0x12,0x86

		// item1, id-eNB-UE-S1-AP-ID (6)
		,0x00,0x08,0x00,0x02,0x00,0x00

		// item2, id-E-RABSetupListCtxtSURes (Msg type: 0x32) (19)
		,0x00,0x33,0x40,0x0f,0x00,0x00,0x32,0x40,0x0a,0x0a,0x1f,0x04,0x05,0x01,0x11,0x01,0x00,0x00,0x0a
	};

	return temp;
}



u_char* CPacketSenderDlg::attachCompleteItems()
{
	u_char temp[57] = {
		// item0, id-MME-UE-S1AP-ID
		0x00,0x00,0x00,0x03,0x40,0x12,0x86

		// item1, id-eNB-UE-S1AP-ID
		,0x00,0x08,0x00,0x02,0x00,0x00

		// item2, id-NAS-PDU (Msg type: 0x43)
		,0x00,0x1a,0x00,0x0e,0x0d,0x27,0x40,0x73,0x5f,0x51,0x02,0x07,0x43,0x00,0x03,0x52,0x00,0xc2

		// item3, id-EUTRAN-CGI
		,0x00,0x64,0x40,0x08,0x00,0x54,0xf0,0x60,0x00,0x00,0x10,0xe0

		// item4, id-TAI
		,0x00,0x43,0x40,0x06,0x00,0x54,0xf0,0x60,0x01,0xf4
	};

	return temp;
}
```

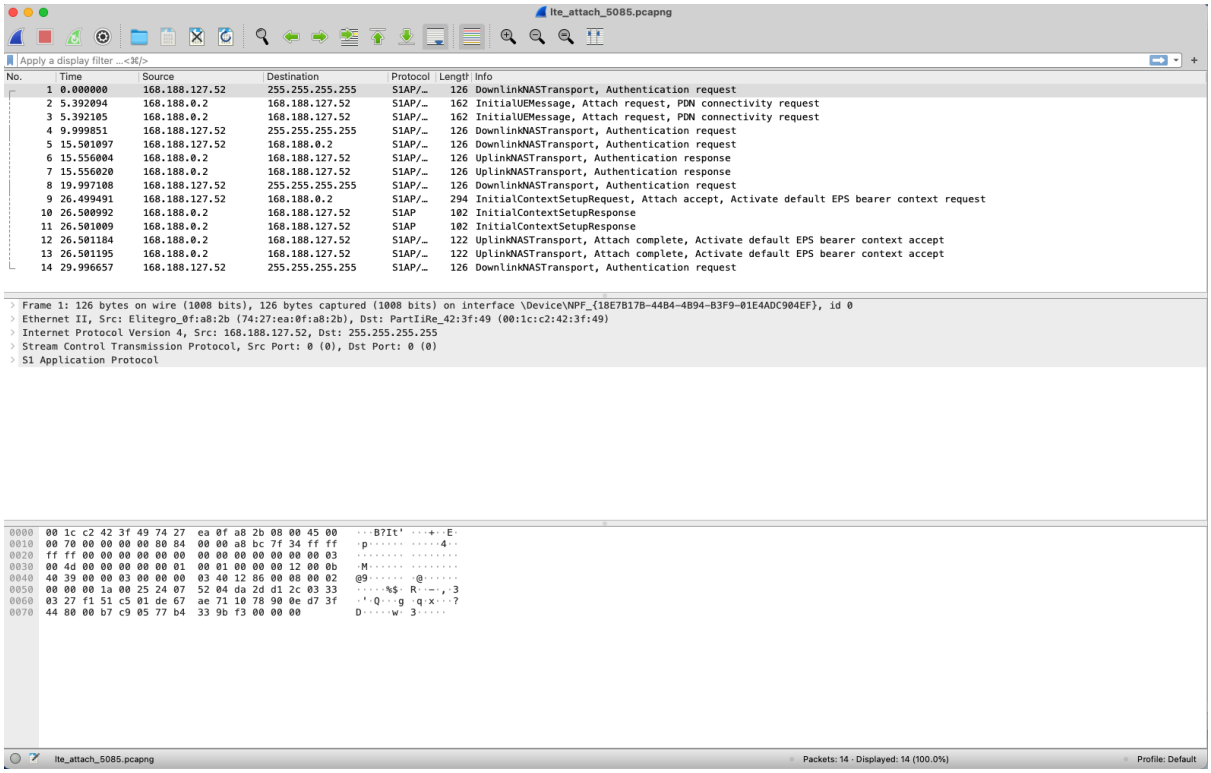
MME가 Attach Accept 메시지를 통해 UE에게 할당된 IP 주소와 QoS정보를 전달한다.

eNB는 MME로부터 Initial Context Setup Request를 통해 DRB를 설정할 수 있도록 정보를 전달받는다.

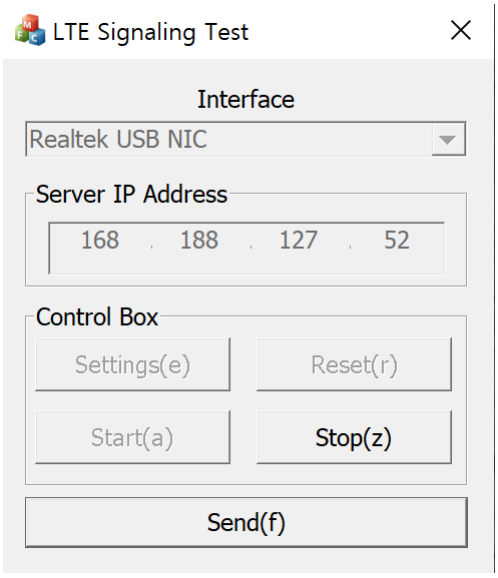
MME는 eNB로부터 Initial Context Setup Response 메시지를 받아 DRB가 Initial Context Setup을 마쳤음을 S-GW로 알릴 수 있다.

4. 실행 결과

1. Wireshark Packet Capture 화면



2. 프로그램 결과 화면



5. 총평

1. 실습 과정에서 발생한 문제점

실습 과정에서 프로토콜 헤더에 잘못된 값을 넣으면 wireshark의 info 컬럼에 잘못된, 의도하지 않은 값이 찍혔다. (예를 들어 handover required 등)

2. 이를 해결한 방법

해당 값이 어떤 헤더의 에러로 인해 찍히게 된 것인지 생각해보고 의심이 가는 부분을 계속 바꿔보면서 원하는 패킷이 나올 때 까지 반복해서 해결할 수 있었다.

3. 새로 발견한 사항 또는 새로운 아이디어

wireshark 쓰는 것에 더 익숙해지게 되었다.