RSA (공개키 암호) (비대칭 키) (RSA-2048 / 3072 / 4096)

1. 키 생성 2. 암호화 3. 복호화

*키 생성은 복호화하는 사람이 한다. 암호화는 암호화하는 자, 복호화는 복호화 하는 자.

*이 알고리즘은 두 개의 큰 소수를 이용한다. 이 수들의 곱과 추가연산을 통해 하나는 공개키, 다른 하나는 개인키를 구성.

-RSA 알고리즘에서는 모든 사람이 고유한 n 값(두 소수의 곱, p와 q의 곱)을 가진다.

영희의 공개키인 n값은 모두에게 공개된다.

그렇다면 영희에게 메시지를 보내고 싶은 사람은 n값을 어떤 알고리즘으로 암호화한 후 영희에게 보낸다. 여기서 영희의 개인키는 p와 q이다.

여기서 b=e이다.

<키 생성>

RSA 암호화를 위해 2개의 큰 소수 p와 q를 선택하여 비밀로 한다. 그렇지만 그 곱인 $n=pq$는 공개한다.

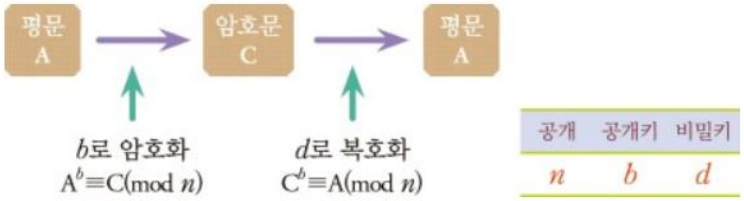
-$\phi(n)=\phi(pq)=\phi(p)\phi(q)=(p-1)(q-1)$이다. Compute $n = pq$ and $\phi = (p-1)(q-1)$.

1. 암호화 키로 쓰이는 수 b는 $\phi(n)$과 서로소인 수 중에서 선택하고, n과 마찬가지로 공개한다. Select a random integer e, $1 < e < \phi$, such that $\gcd(e, \phi) = 1$
2. 암호를 보낼 때에는 $A^b \equiv C \pmod{n}$를 계산하여 C를 보낸다.
3. 암호문을 받은 사람이 복호화하기 위해서는 키 d가 필요하다.

   D는 $bd \equiv 1 \pmod{\phi(n)}$, 을 만족하는 수이다. → d는 n,b와 달리 비밀키이다.

   암호문 C를 전해 받은 사람은 복호화 키 d를 이용하여 A를 알아낼 수 있다.

$$C^d \equiv (A^b)^d \equiv A^{bd}$$
$$\equiv A^{1+k\phi(n)} \equiv AA^{k\phi(n)} \quad (\because \text{오일러의 정리에 의해 } A^{\phi(n)} \equiv 1 \pmod{n})$$
$$\equiv A \pmod{n}$$



공개	공개키	비밀키
n	b	d

A's public key is (n, e); A's private key is d.

구체적인 수를 가지고 위의 과정을 밟아보자. 계산을 간편화하기 위해 A=2를 암호화한다고 가정하자. 두 소수 p와 q를 각각 3과 11로 선택하면, n=3×11=33이다. $\phi(33)=\phi(3)\phi(11)=(3-1)(11-1)=20$이므로 20과 서로소인 수 b=7을 선택하자. 이를 이용하여 암호화하면 $2^7=128\equiv29 \pmod{33}$이고, 29를 공개된 (33, 7)과 함께 보낸다.

받은 메시지 29를 복호화할 때는 비밀키 d를 알아야 한다. d는 $7 \cdot d\equiv1 \pmod{20}$을 만족하는 수로, $7 \cdot 3=21\equiv1 \pmod{20}$이므로 d=3이 된다. 이 키를 알면 $29^3\equiv(2^7)^3=2^{21}=2^{20} \cdot 2^1=2^{\phi(33)} \cdot 2^1\equiv1 \cdot 2\equiv2 \pmod{33}$이므로 처음에 암호화한 수가 2임을 쉽게 알 수 있다.

그러나 n을 소인수분해한 p와 q는 비밀로 되어 있기 때문에, $\phi(n)=(p-1)(q-1)$을 계산할 수가 없고, 따라서 $bd\equiv1 \pmod{\phi(n)}$을 만족하는 d를 알아내기 어렵다. 즉, 소인수분해하는 데 오랜 시간이 걸린다는 점 때문에 비밀키 d를 밝혀내기 어려운 것이다.

RSA가 2048비트라면 p와 q는 1024비트씩.

<암호화>

*공개키는 누구나 암호화할 수 있어야 하나 풀 수 있는 것은 alice만.

*public키는 bob들(모두를위해)을 위해. private키는 alice를 위해.

1. 숫자를 고른다 . 1과 n-1까지의 숫자중.(m)

Enc(m1)=C1=(m1^e) mod (n)

C1을 보낸다 → Alice는 c1에 d승(private key)을 하고 mod n 만 해주면 원래의 m1이 나온다.(복호화)

*alice에서 나오는 것은 RSA 전자서명. RSA 암호화 메시지는 BOB들이 만듦.

*공개키는 데이터가 짧을 때, 막대한 데이터에는 ㄷㅐ칭키.

<복호화>

Key d는 $ed \equiv 1 \pmod{\phi}$ 이것을 만족한다.

$ed = 1 + k\phi$. Now, if $\gcd(m, p) = 1$ 라면 $m^{p-1} \equiv 1 \pmod{p}$. 이다.

$m^{1+k(p-1)(q-1)} \equiv m \pmod{p}$ (k(q-1)승을 해주고 양 변에 m을 곱해준다.)

$m^{ed} \equiv m \pmod{p}$.

$m^{ed} \equiv m \pmod{q}$.

primes, it follows that

$m^{ed} \equiv m \pmod{n}$, → $c^d \equiv (m^e)^d \equiv m \pmod{n}$.

<DH KEY EXCHANGE>

공개적으로 교환할 발생기 generater로 **p를 소수 g를 생성자**라고 하겠습니다.

1) 엘리스는 개인키 a를 이용하여 **g^a mod p**를 생성합니다.

2) 밥은 개인키 b를 이용하여 **g^b mod p**를 생성합니다.

3) 엘리스는 밥에게 **g^a mod p**를 보내고 밥은 엘리스에게 **g^b mod p**를 송신합니다.

4) 엘리스는 자신의 개인키 a를 이용하여 **G^ab mod p**를 생성하고 밥은 자신의 개인키 b를 이용하여 **g^ab mod p**를 만듭니다.

5) 이제 엘리스와 밥은 **새롭게 생성된 키를 대칭키(비밀키)로 이용**합니다

// 이때 엘리스와 밥이 교환하여 결합한 값이 서로 같다는 것!

- 비대칭키(공개키) 알고리즘에서 사용되는 키 교환 방식이다.
- 상대방의 공개키와 나의 비밀키를 이용하여 비밀키를 생성한다.
- A의 공개키와 B의 개인키를 DH연산하면 B의 비밀키가 되고
- B의 공개키와 A의 개인키를 DH연산하면 A의 비밀키가 된다.
- 이산대수법에 의거한 수학적 공식에 의해 A의 비밀키와 B의 비밀키는 같아진다.
- 송신자와 수신자는 이 비밀키를 사용하여 데이터를 암호화한 후 전달한다.

디피-헬만법은 이산대수의 어려움을 이용한 알고리즘이다. 쉽게 말해 주어진 g, x, p 를 이용하여 y = g^x mod p 를 구하기는 쉽지만 g, y , p 값을 이용하여 원래의 x를 찾기 어렵다는 원리를 이용한 것이다.

An appropriate prime p and generator α of $\mathbb{Z}_p^*$ $(2 \leq \alpha \leq p-2)$

➔ P-1은 원시근이 될 수 없다. 제곱하면 1이고 세제곱하면 -1 / 1도 원시근이 될 수 없다

$$A \rightarrow B : \alpha^x \bmod p \quad (1)$$
$$A \leftarrow B : \alpha^y \bmod p \quad (2)$$

➔ MESSAGE

(a) A chooses a random secret x, $1 \leq x \leq p-2$, and sends B message (1).
(b) B chooses a random secret y, $1 \leq y \leq p-2$, and sends A message (2).
(c) B receives α^x and computes the shared key as $K = (\alpha^x)^y \bmod p$.
(d) A receives α^y and computes the shared key as $K = (\alpha^y)^x \bmod p$.

➔ ACTION

Could use $K = g^{ab} \bmod p$ as symmetric key

트루디가 엘리스와 밥 중간에서 엘리스의 a값을 t로 변경하여 밥에게 송신하고 밥에게 받은 b의 값을 t로 바꾸어 엘리스에게 송신하고 있습니다.

즉, **g^at mod p**와 **g^bt mod p**를 만들었습니다. 하지만 **엘리스와 밥은 이 사실을 인지하지 못하고 트루디가 준 g^t mod p를 이용하여 서로 g^at mod p와 g^bt mod p를 생성**합니다. 이로 인하여 엘리스와 밥 그리고 트루디는 모두 같은 대칭키를 가지게 되었습니다. 즉 트루디는 공격을 할 수 있게 되었습니다. 이런 MITM(중간자공격)을 어떻게해야 방지할 수 있을까요?

1) **대칭키**로 디피헬만 교환값을 암호화
2) **공개키**로 디피헬만 교환값을 암호화
3) 디피헬만 교환값을 **개인키로 서명**

<RSA 전자서명> : 메시지 무결성, 출처인증 보장 (출처인증은 부인방지)

전자서명

1. 개인키(비밀키)를 가진 사람만이 서명을 생성할 수 있어야 함.

2. 서명은 문서마다 변경되어야 함.

3. **'서명생성'**은 입력으로 **메시지 x**와 **개인키** 를 갖는 함수로 구현

4. **'서명검증'**은 입력으로 **공개키**와 **메시지x**를 갖는 함수로 구현

전자서명의 조건

· 위조불가 (unforgeable)

· 서명자 인증(user authentication)

· 부인불가 (non-repudiation)

· 변경불가 (unalterable)

· 재사용 불가 (not reusable)

*** 전자서명은 기밀성을 제공하지 않는다.**

-키 생성

*사용자 PUBLIC KEY(n,e)

: 임의의 두소수 p와 q의 곱인 n을 공개하고 (p-1)*(q-1)의 서로수인 random숫자 e를 공개한다.

Compute $n = pq$ and $\phi = (p-1)(q-1)$. Select a random integer e, $1 < e < \phi$, such that $\gcd(e, \phi) = 1$.

*사용자 private key(d)

: d, $1 < d < \phi$, such that $ed \equiv 1 \pmod{\phi}$.

-서명 & 검증

*서명 : $S = M^d \ mod \ n$: 어떤 메시지 M에 대하여 개인키 d를 사용하여 서명 생성.

*검증 : $M = S^e \ mod \ n$: signature에 e승을 해서 맞는지 확인.

*서명 검증은 일반인의 몫, 서명 생성은 alice

*RSA는 서명을 검증하는데 빠르고 DSA는 서명을 생성하는데 빠르다.

<DSA 전자서명> : 자기가 남긴 메시지가 내꺼다라고 증명하고 싶음.

DSA는 이산대수문제, RSA는 소인수분해

-키 생성

소수 p의 크기 $2^{159} \le p \le 2^{160}$, 160비트
소수 q의 크기 $2^{511} \le q \le 2^{1024}$, 512비트부터 최대 1024비트, 64의 배수

*q는 p-1로 나누어져야 한다. Q는 (p-1)의 약수이다.

$\alpha = g^{(p-1)/q} \bmod p.$ → 여기서 g는 원시근. (알파는 생성자.) 1승부터 q승까지. 그룹군 중에 하나를 고름.

*private key: random integer a such that $1 \le a \le q - 1$.

*public key: $y = \alpha^a \bmod p.$

A's public key is (p, q, α, y); A's private key is a.

전자서명 : 분리가능, 비밀키 복제(hard), digital
인감도장 : 분리불가 , 도장 복제 (easy), analog

-서명 생성 (서명생성, 키생성 모두 alice)

0에서 q사이에서 k를 하나 고름.
지수만다룰때는 mod q
전체를 다룰때는 mod p.

k는 나중에 지수에 쓰일 것이다.

1. Select a random secret integer $k, 0 < k < q.$

2. $r = (\alpha^k \bmod p) \bmod q$ **계산**

3. $s = k^{-1}\{h(m) + ar\} \bmod q.$ **계산. → 서명생성할 때 k도 안다는 것을 증명하기 위해 k^-1을 곱함.**

A's signature for m is the pair (r, s).

메시지 m에 대하여 DSA 전자서명 값 (r, s)의 각각 크기는 160비트
즉, 임의의 길이 메시지에 대해 실제 DSA 서명 값은 겨우 320비트임

-서명 검증

1. 메시지 m과 전자서명 (r,s)를 얻은 후
2. Obtain A's authentic public key (p, q, α, y).
3. Verify that $0 < r < q$ and $0 < s < q$; if not, then reject the signature.
4. Compute $w = s^{-1} \bmod q$ and $h(m)$.
5. Compute $u_1 = w \cdot h(m) \bmod q$ and $u_2 = rw \bmod q$.
6. Compute $v = (\alpha^{u_1} y^{u_2} \bmod p) \bmod q$.
7. v와 r이 같아야 한다.

실제 검증자는 서명에 사용된 난수 k를 알지 못함
실제 검증자는 서명에 사용된 비밀키 a를 알지 못함
수신한 메시지 m′에 대하여 검증자는 서명자의 공개키를 이용해서 메시지 m′가 위.변조 되었는지? (메시지의 integrity 확인 가능)
그 서명이 정말 서명자가 맞는 지? (메시지 출처 인증 가능)

$k = \frac{H(M) + xr}{s}$ 으로 k를 정리해볼 수 있습니다.

$u_1 = \frac{H(M)}{s}$
$u_2 = \frac{r}{s}$ 검증자는 이 두 값을 검증하기 위해 임의로 만들어 줍니다.

$$v = ((g^{u_1} \cdot Y^{u_2}) \bmod p) \bmod q = g^{\frac{H(M)}{s}} \cdot Y^{\frac{r}{s}} = g^{\frac{H(M)}{s}} \cdot (g^x)^{\frac{r}{s}} = g^{\frac{H(M)+xr}{s}} = g^k$$

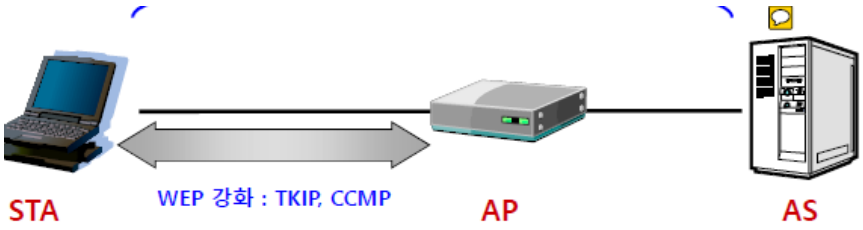
v 값이 서명할 때 만들었던 r값과 같으면 서명검증이 됩니다.

(여기서 g는 알파. a는 x)

<무선통신과 암호 및 정보보안>

-무선랜(LAN, local area) 과 암호 및 정보보안

*STA (무선 단말기), AP(통신사) : WEP 강화 → WEP 키를 주기적으로 변경함으로서 무선랜의 보안성을 향상시키는 방식.



<LTE Security & Network Architecture>

-LTE ARCHITECTURE

: 이전 시스템인 회로 교환 (circuit-switched) 모델과 비교하여 LTE는 패킷교환만 지원하도록 디자인 되었다.

-LTE Network

: E-UTRAN(3G) + EPC(Evolved Packet Core)=Evolved Packet System(EPS)=All – IP Network

*UE : 사용자 단말 (smartphone)

*eNB: LTE 기지국. UE와 LTE간에 무선연결을 제공하는 장비.

*MME : LTE망의 두뇌 , UE를 인증. 인증 프로토콜은 EPS-AKA. UE를 인증하기 위한 KEY는 HSS에 들어 있다. KEY정보를 HSS로부터 받아서 UE를 인증.

*HSS : LTE망의 중앙 DB. 각 UE(가입자)별로 인증을 위한 KEY정보와 가입자 프로파일을 가지고 있다.

*S-GW(Serving Gateway) : E-UTRAN과 EPC의 종단점.

Enb 간 핸드오버 (eNB에서 다른 Enb로 넘어갈 때) 시 anchoring point가 된다.

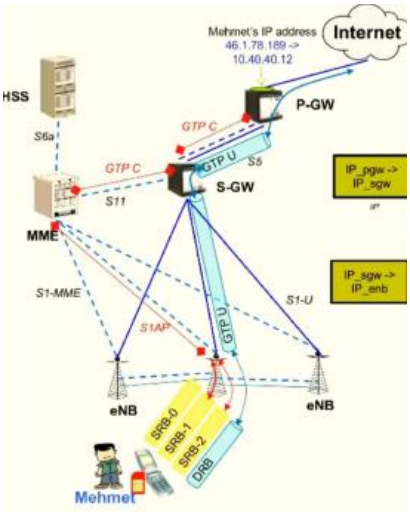
*P-GW(PDN Gateway)

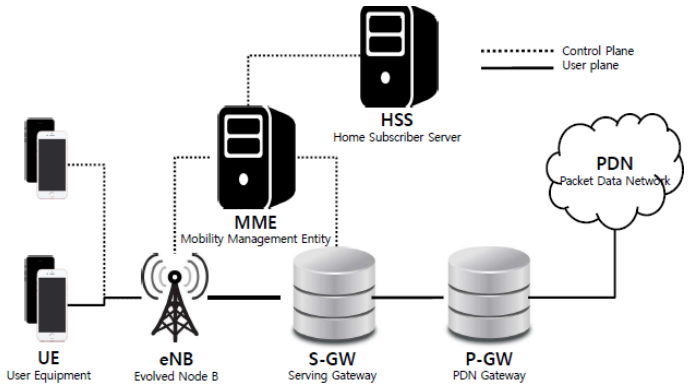
: 각 가입자 별로 언제 접속했고 데이터를 얼마나 사용했는지 기록.

UE를 외부 PDN 망과 연결.

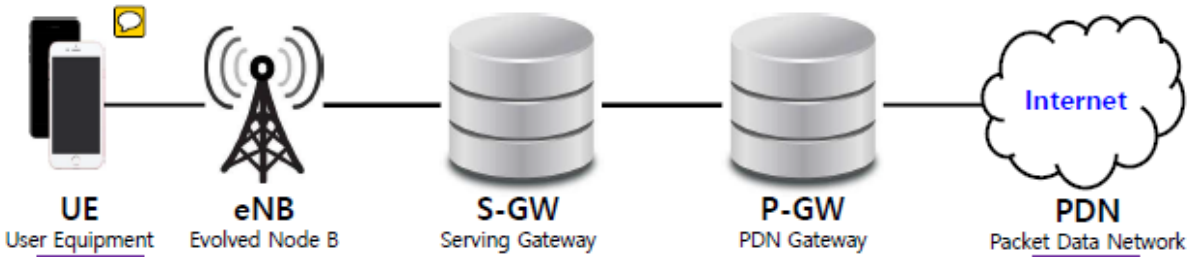
S-GW의 핸드오버시 anchoring point.

UE에 IP주소 할당.





-LTE Aechitecture-User plane (All-IP, 모든 것을 IP를 통해서)



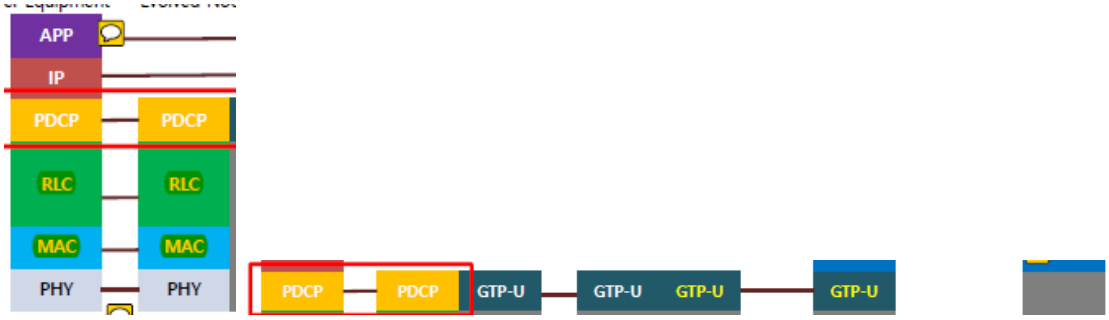
*UE와 Enb만 무선. 여기를 제어하는 것이 MAC, MAC을 제어하는 것이 RLC -

*RLC와 MAC은 암복호화 필요.(AES, ZUC, SNOW) -> 키 일치가 필요.

*PDCP : IP패킷이 무선링크를 통해 효율적으로 전송될 수 있도록.

*RLC : 무선링크 제어

*MAC : 무선 자원을 UE들에게 동적 할당.



*UE가 보낸 IP 패킷은 eNB에서 P-GW까지 GTP 터널을 통해 전송된다. 이 말은 단말이 IP패킷의 DESTINATION IP 주소에 어떤 값을 기록하던 상관없이 단말이 보낸 패킷은 항상 Enb를 통해 P-GW까지 전송이 된다는 의미이다. GTP-U 터널이라고 "U"를 붙인건 User Plane의 약자로 즉 사용자 데이터가 흐르는 것

*무선 구간인 UE와 eNB사이에 기밀성이 있다. 무선구간에서 암호화 한다.

→ 블록암호 사용 (AES,ZUC,SNOW) (대칭키 암호, UE가 암호화하고 eNB가 복호화함)

대칭 키 암호이기 때문에 복호화와 암호화할 때 키가 같음. 키 일치과정이 필요.

1. 인증(개체인증, 메시지인증, 3세대때부터 양방향 인증(MUTUAL))
2. 키일치 (AKA : 1+2)
3. 기밀성(무선구간보호)
4. 무결성 (MAC, 메시지 인증 코드)
5. 출처인증

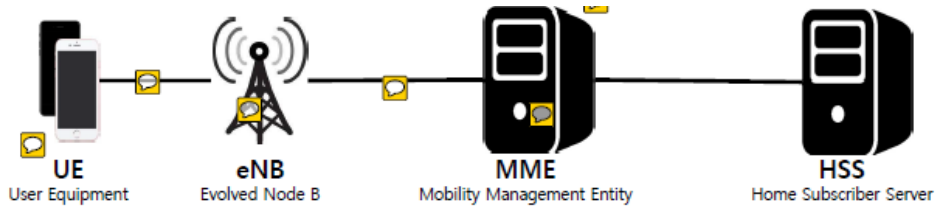
➔ LTE에 들어 있는 보안, (1번과 2번은 USIM / 3,4,5는 단말기)

기밀성을 제공해주는 암호화 알고리즘 필요.
사용자의 개인 정보들을 암호화.
-암호화알고리즘(AES,SNOW3G,ZUC)
eNB안에 이 세가지 알고리즘이 들어있음.
CONTROL PLANE 통신사에서 어떤 알고리즘을 쓸지 결정.

encapsulation : 캡슐화.
여기까지.
내가 원하는 데이터는 PDN에 있다.
통신사가 보안을 도와주지 않음.
데이터를 전달하는 역할만 함.
LTE는 이삿짐센터역할만 함.
이동통신사들은 자기의 LTE를 위한 보안과 사용자들의 보안만 걱정.
그것만 고민하면 됨.

P-GW까지.

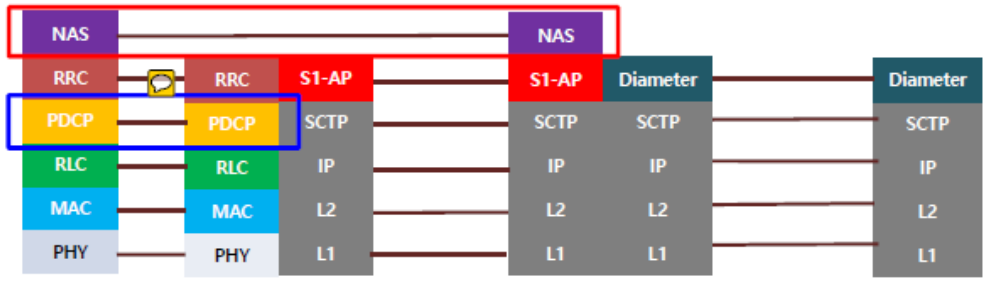
-LTE Aechitecture - Control plane



*usim의 home은 hss. (hss와 말을 하고 싶지만 거리가 멀어서 도와주는 것이 eNB와 MME)

*MME : 인증했다고 도장을 찍어줌, 이동한다고 알려줌, 이동성을 관리.

*eNB와 MME는 현지. 우리의 HOME은 KT, 유플러스 .



배달온사람이 eNB. 우리가보는사람
MME는 전화를 받는 사람.
NAS -> 짜장면 주세요
RRC(RADIO, 무선)-> 배달 총괄
NAS에서 NON ACCESS
MAC은 ACCESS과정.

eNB와 MME가 AKA
eNB가 USIM에게 누구냐고 물어봄.
HSS가 증표를 줌으로서
그 증표를 만들 수 있는 사람이 HSS밖에 없기 때문에
USER는 eNB와 MME를 믿게 됨.

*NAS : NON ACCESS, UE와 MME 간의 메시지 전송.

*RRC : 무선 자원 제어 계층

*PDCP : CP, UP데이터 전송

-LTE Security ARCHITECTURE (user to network security) : EPS-AKA

*EPS-AKA는 E-UTRAN에서 사용해야 하는 인증 및 키 일치 과정이다.

*EPS-AKA는 RRC, NAS무결성 보호 키 뿐만 아니라 UP, RRC, NAS암호 키에 대한 기반을 형성해야 한다.

- **UE-MME : NAS Signaling – Integrity, Ciphering**
- **UE-eNB : RRC Signaling – Integrity, Ciphering**
- **UE-eNB : User traffic - Ciphering**

*NAS Signaling : 짜장면이요 하고 신호를 준다.

*eNB : 직접배달, AES 암호 하나로 INTEGRITY(무결성), CIPHERING(암호화) 다됨.

*실제 트래픽은 암호화만. INTEGRITY까지 하면 서비스 속도가 떨어짐.

*모든 무선 통신망(Wi-Fi, WiBro 그리고 LTE)은 "가입자 인증(User Authentication)"과 "무선구간에서의 보안(무결성 확인 및 암호화)" 기능을 제공

*LTE 는 EPS-AKA 라는 방식을 사용하여 인증한다. 여기서 인증은 상호인증!

UE 는 MME 에게 Attach request, MME 는 UE 에게 User Authentication request

UE 와 HSS 모두에 가입자 고유 식별자인 IMSI 와 LTE K 값이 있어야 한다. (USIM 안에 들어있다.)

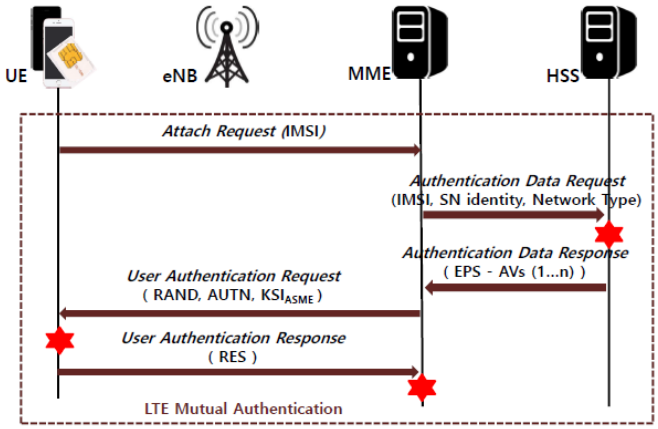
가입자가 전원을 키면 UE 는 망에 인증을 요청(attach request)

이 메시지를 수신한 MME 는 HSS 에게 해당 가입자(IMSI 로 식별)를 인증하기 위한 인증정보를 받아옴.

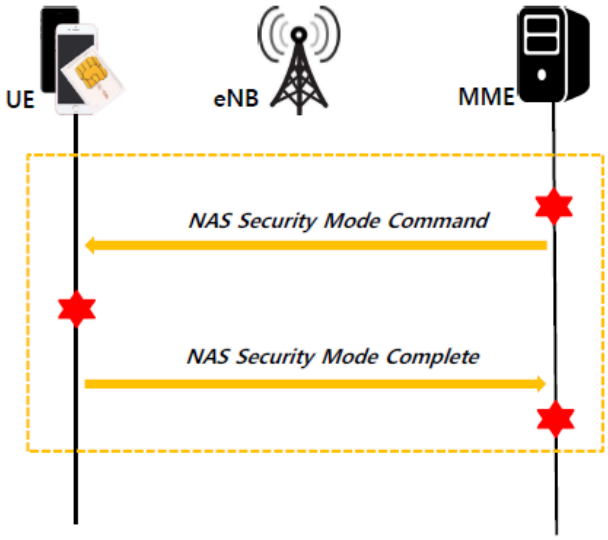
MME 는 이 인증정보를 이용하여 가입자를 인증.

*인증이 끝난 후 무선 구간 보안.

; 인증이 끝나면 결과로 master key 가 나온다. 이 master key 를 이용하여 복잡한 알고리즘을 돌려 무선구간에서 무결성과 암호화를 할 수 있는 key 를 생성. 이 Master Key를 K_{ASME}



- K_{ASME} 는 UE에 전달되지 않고, UE가 직접 생성
  하지만 이에 대응하는 인덱스로 KSI_{ASME} 전송

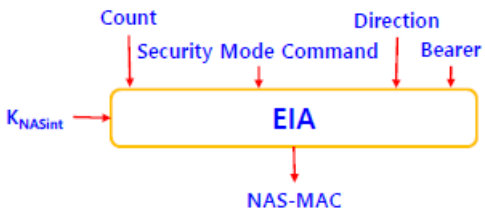


- UE와 MME는 상호 인증을 마치고 K_{ASME} 를 공유한 후 NAS Security Setup 절차를 시작한다.
- NAS 시그널링 메시지들을 암호화하고 무결성 보호하기 위해

NAS Security Setup

*EIA(EPS INTEGRITY ALGORITHM)

- KSI_{NASint}
- Replayed UE Security Capability
- NAS Ciphering Algorithm (MME selected)
- NAS Integrity Protection Algorithm (MME selected)



- MME는 UE에게 전송할 Security Mode Command 메시지를 구성하고, NAS-MAC을 생성한다.
- 생성된 NAS-MAC을 Security Mode Command에 포함시켜 UE에 전송하고 암호화는 없음

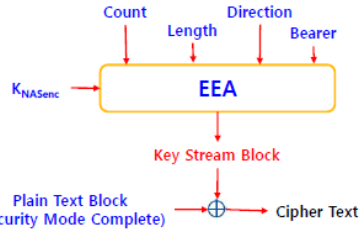
무결성확인/보호(Integrity Check): 송신측에서 자신이 가지고 있는 Key값과 메시지내용을 어떤 알고리즘을 돌려 32바이트(예를 들어 32바이트)짜리 어떤 값을 얻습니다. 그리고 그 값(A)을 메시지 맨 뒤에 붙여서 보냅니다. 그러면 이를 수

*MME와 마찬가지로 K_{NASint} 와 K_{NASenc} 생성

UE는 수신한 Security Mode Command에 포함된 NAS-MAC을 검증

- UE가 MME와 같은 방법으로 XNAS-MAC 생성 후 비교

*EEA(EPS ENCRYPTION ALGORITHM



- 검증이 완료되면 UE는 MME에 전송할 Security Mode Complete 메시지를 구성하고 암호화 한다.

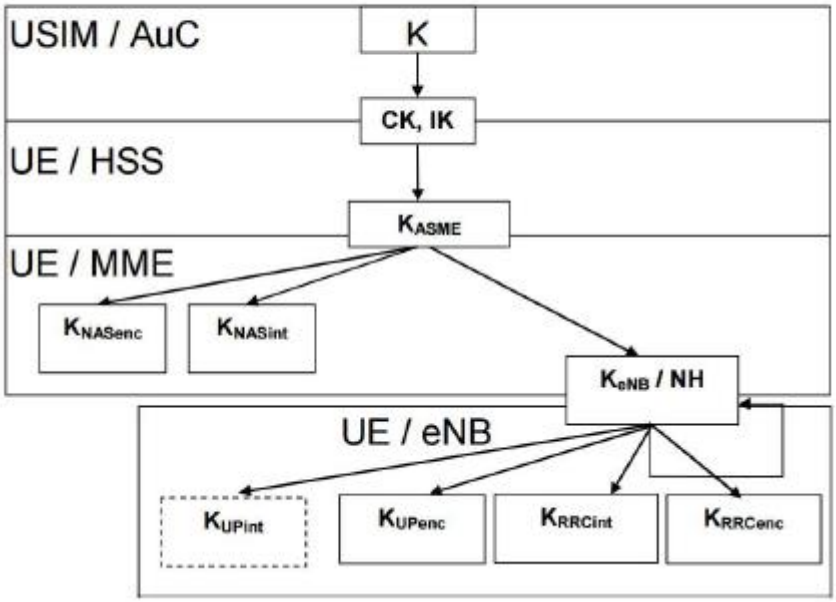
)

그런 후 이번에는 UE와 eNB간에 주고 받는 모든 control 메시지에 대한 무결성확인과 암호화를 위한 Key인 K_{RRCint}(RRC=RRC 메시지, int=Integrity)와 K_{RRCenc}(enc=encryption)를 생성하고, 이를 통해 UE와 MME간 control 메시지는 무결성보호 & 암호화 되어 안전하게 주고 받을 수 있게 됩니다.

- eNB는 UE에게 전송할 Security Mode Command 메시지를 생성하고 K_{RRCint} 를 사용하여 MAC-I 생성

- AS Security Setup 절차는 RRC 시그널링 메시지와 IP 패킷을 안전하게 전달하기 위한 키를 생성하는 절차이다.

- **LTE Security KEYs**



Nas, rrc→ control 메시지만

Up(user plane) → 데이터 메세지