Asymmetric Cryptography

2025.08

자동차융합대학

GENERAL MOTORS
GM TECHNICAL CENTER KOREA

KMU 국민대학교
KOOKMIN UNIVERSITY

CONTENTS

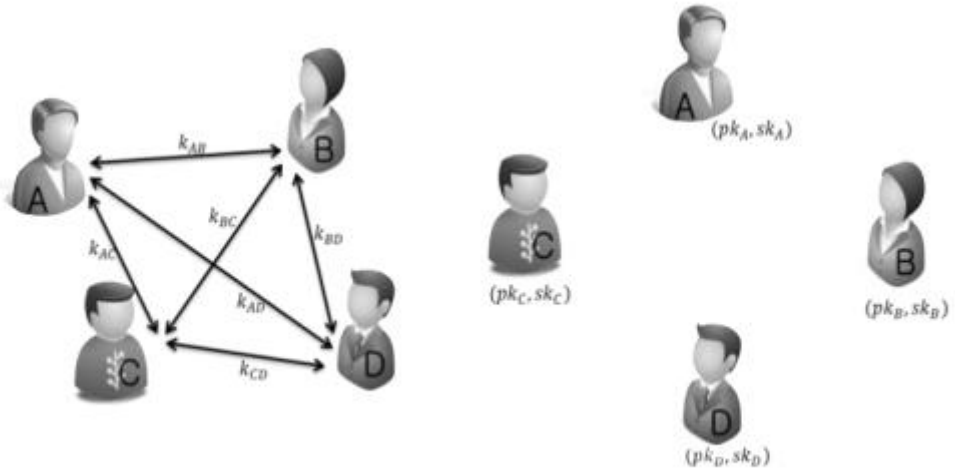
01

공개키 암호 개요

공개키 암호 개요

■ 대칭키 암호

- 안전한 채널을 통해서 사용자가 서로 동일한 키를 사전 공유
- N명이 서로 비밀 통신을 하기 위해서는 $\frac{n(n-1)}{2}$ 개의 키가 필요
- 송신자나 수신자의 부인방지를 제공하지 못함

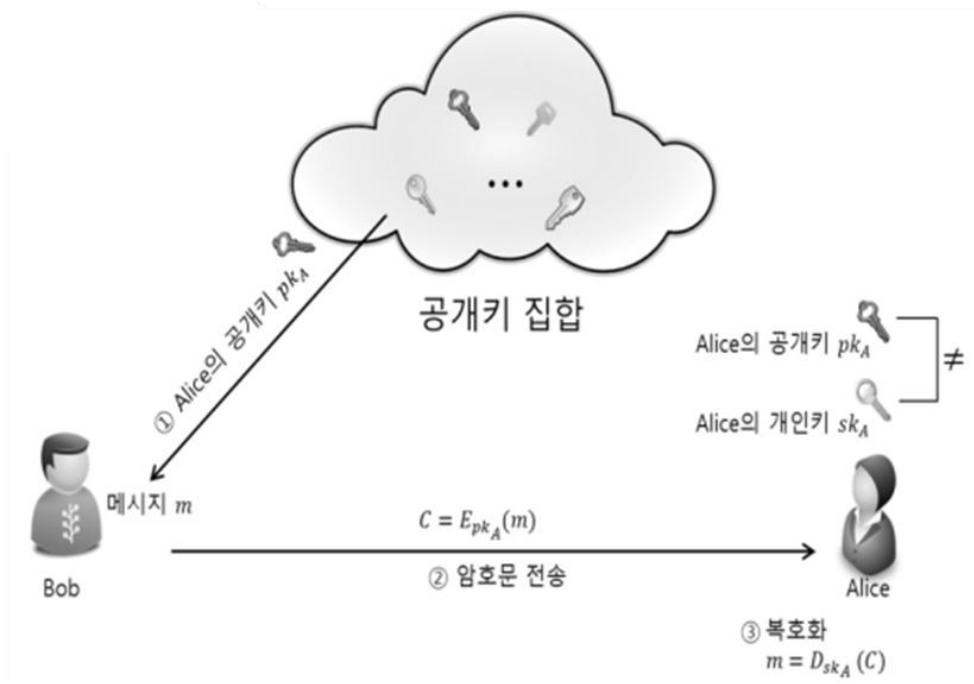


대칭키 암호시스템 공개키 암호시스템

공개키 암호 개요

■ 공개키 (or 비대칭키(Asymmetric) 암호시스템)

- Diffie와 hellman은 1976년 발표된 논문 “New Directions in Cryptography)”에서 공개키 암호시스템 소개
- 각 사람마다 한 쌍의 키(공개키 pk, 개인키 sk)
  - 공개키는 모두에게 공개되고, 개인키는 비밀로 보관
  - 공개키 pk로부터 개인키 sk를 도출하는 것은 계산적으로 불가능 (Computationally Infeasible)



공개키 암호 개요

■ 공개키 암호시스템과 대칭키 암호시스템의 차이점

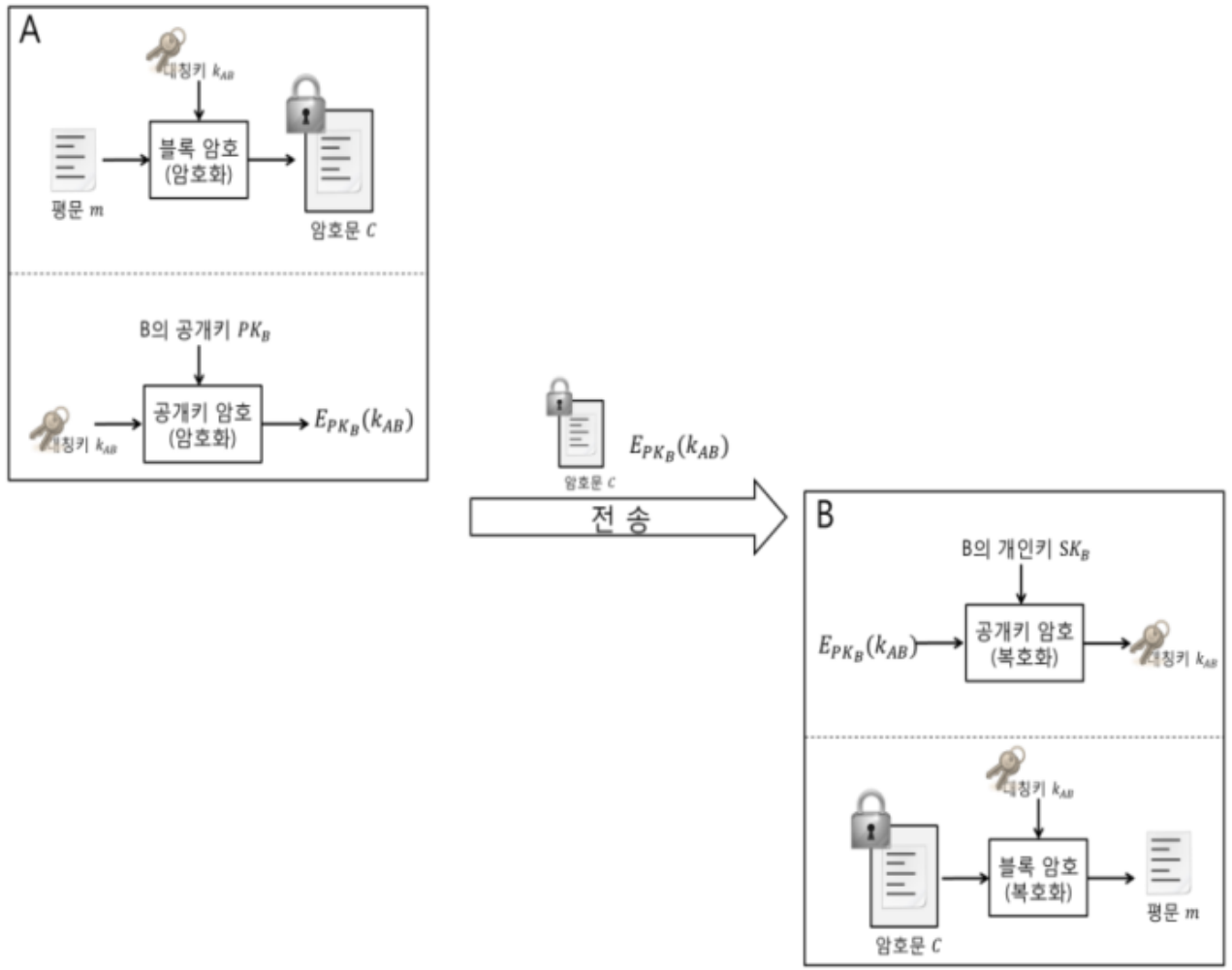
	대칭키 암호시스템	공개키 암호시스템
비밀키 분배	필요	불필요
보유 비밀키 개수 (n명이 비밀통신 하는 경우)	$(n-1)$개 (상대방별로 키가 필요)	1개 (자신의 비밀키만 보유)
암호화 & 복호화 속도	빠름	느림
대표 예	DES, AES, SEED, ARIA	RSA, ElGamal

공개키 암호 개요

■ 하이브리드 암호시스템

- 대용량의 데이터를 암호화하기 위해서 대칭키 암호 시스템에서 사용되는 secret key를 공개키 암호시스템으로 암호화($E_{pk}(secret\ key)$)하여 분배

- 수신자는 분배된 비밀키를 이용하여 대용량의 데이터를 대칭키 암호시스템으로 암호화



■ 일방향 함수 (One-Way Function) f

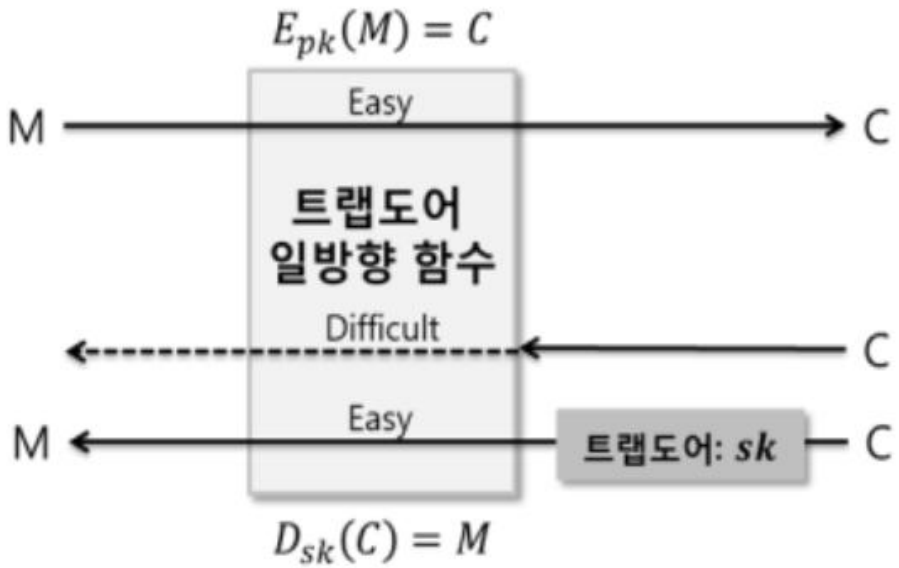
- f is easy to compute.
- f^{-1} is difficult to compute.

■ 소인수분해 문제

- When n is large, n = p * q is a one-way function.
- Given p and q, it is always easy to calculate n; given n, it is very difficult to compute p and q.
- 최근까지 알려진 결과로는 2009년에 232자리의 십진수를 수 백대의 컴퓨터를 사용하여 2년 만에 인수분해에 성공
  - 232자리 십진수를 이진수로 나타내면 768비트가 필요하며 위의 결과는 768비트 RSA의 경우 동일한 계산능력으로 2년 만에 평문이 복호화 될 수 있음을 의미

■ Trapdoor One-Way Function (TOWF)

- f is easy to compute.
- f^{-1} is difficult to compute.
- Given y and a trapdoor, x can be computed easily.



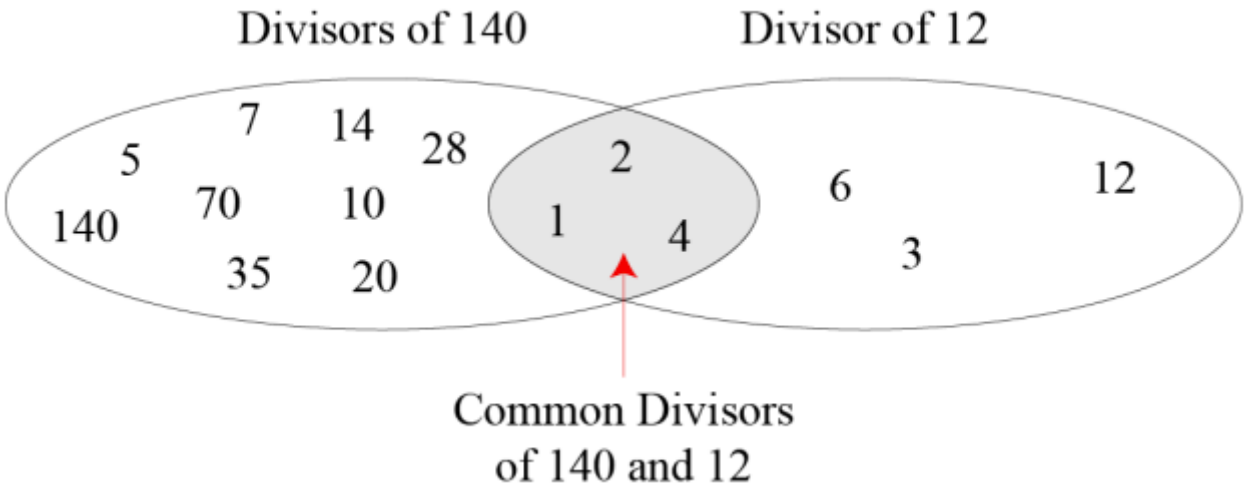
[그림] 공개키 설계의 기본 개념

02

수학적
배경 지식

■ 약수, 공약수, 최대공약수 (GCD: Greatest Common Divisor)

- $\gcd(a, b) = \gcd(140, 12) = 4$
- 0이 아닌 두 정수 a, b에 대하여, $\gcd(a, b) = 1$을 만족하면 a와 b는 서로소(relative prime)



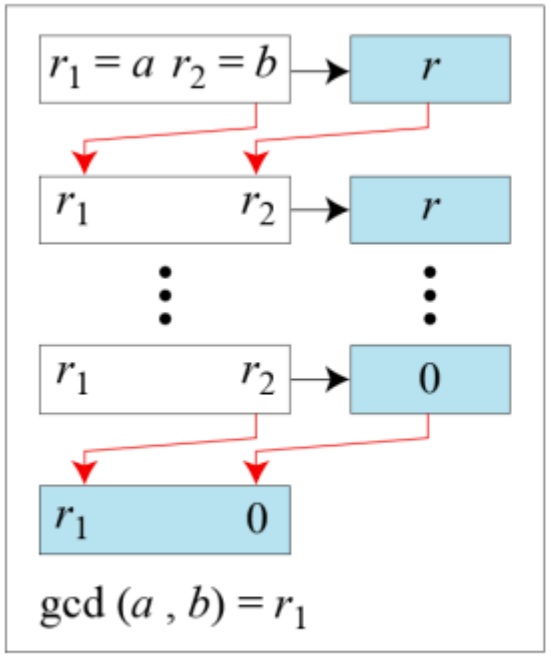
■ 유클리드 알고리즘(Euclidean Algorithm)

Fact 1: gcd (a, 0) = a
Fact 2: gcd (a, b) = gcd (b, r), where r is
the remainder of dividing a by b

q	r_1	r_2	r
1	2740	1760	980
1	1760	980	780
1	980	780	200
3	780	200	180
1	200	180	20
9	180	20	0
	20	0	

수학적 배경지식 (cont'd)

■ 유클리드 알고리즘(Euclidean Algorithm)



a. Process

$r_1 \leftarrow a;$ $r_2 \leftarrow b;$ (Initialization)

while $(r_2 > 0)$

{

$q \leftarrow r_1 / r_2;$

$r \leftarrow r_1 - q \times r_2;$

$r_1 \leftarrow r_2;$ $r_2 \leftarrow r;$

}

gcd $(a, b) \leftarrow r_1$

b. Algorithm

[문제] 유클리드 알고리즘

■ 다음 두 수의 최대공약수(GCD)를 유클리드 알고리즘을 이용하여 구하시오.

- gcd(675, 108) = ?

q	r1	r2	r
6	675	108	27
4	108	27	0
	27	0	

- gcd(1666, 6732) = ?

q	r1	r2	r
4	6732	1666	68
24	1666	68	34
2	68	34	0
	34	0	

수학적 배경지식 (cont'd)

■ 확장 유클리드 알고리즘 (Extended Euclidean Algorithm)

- 적어도 하나는 0이 아닌 두 정수 a와 b에 대하여 다음을 만족하는 s와 t가 존재한다.

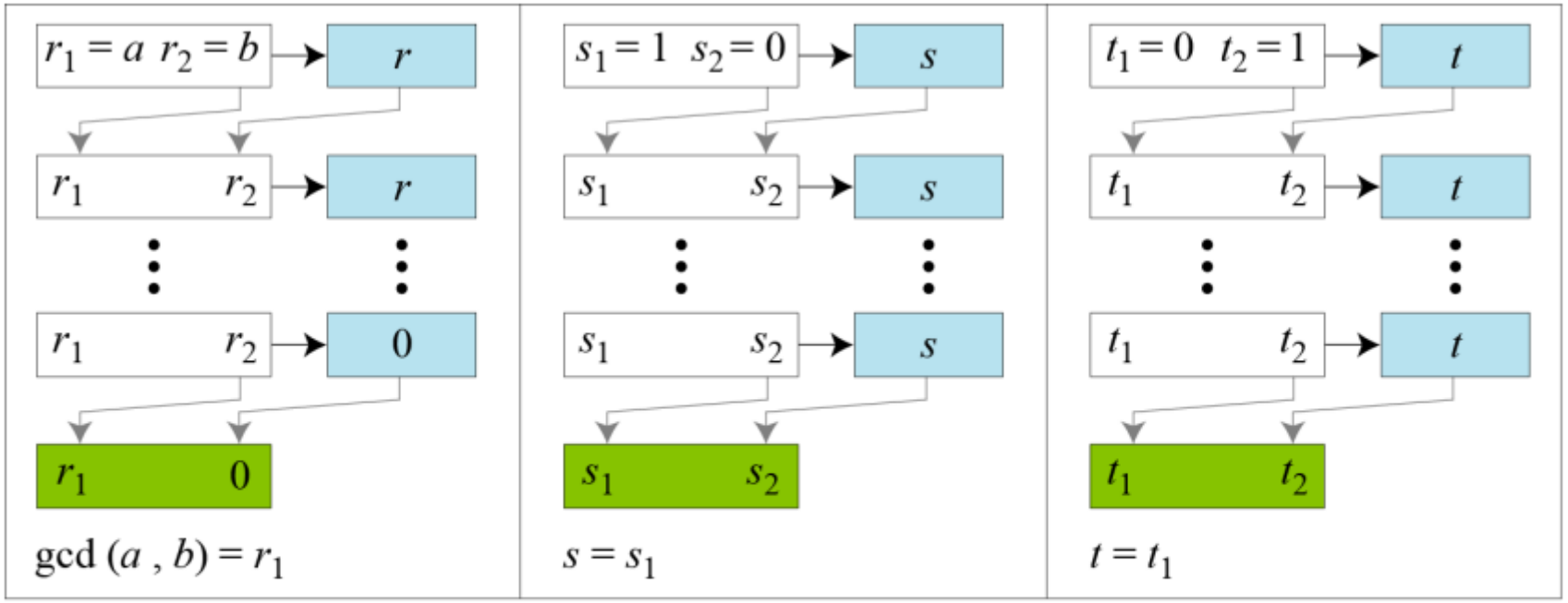
$$s \times a + t \times b = \gcd(a, b)$$

- a = 75, b = 20인 경우

$$75 \times (-1) + 20 \times (4) = \gcd(75, 20) = 5$$

- 확장 유클리드 알고리즘은 $\gcd(a, b)$ 뿐만 아니라 s와 t를 구해준다.

수학적 배경지식 (cont'd)



a. Process

$r = r_1 - q \times r_2, \; s = s_1 - q \times s_2, \; t = t_1 - q \times t_2$

수학적 배경지식 (cont'd)

$r_1 \leftarrow a;$ $r_2 \leftarrow b;$
$s_1 \leftarrow 1;$ $s_2 \leftarrow 0;$
$t_1 \leftarrow 0;$ $t_2 \leftarrow 1;$ (Initialization)

while $(r_2 > 0)$

{

$q \leftarrow r_1 / r_2;$

$r \leftarrow r_1 - q \times r_2;$
$r_1 \leftarrow r_2;$ $r_2 \leftarrow r;$ (Updating r's)

$s \leftarrow s_1 - q \times s_2;$
$s_1 \leftarrow s_2;$ $s_2 \leftarrow s;$ (Updating s's)

$t \leftarrow t_1 - q \times t_2;$
$t_1 \leftarrow t_2;$ $t_2 \leftarrow$ t; (Updating t's)

}

gcd $(a, b) \leftarrow r_1;$ $s \leftarrow s_1;$ $t \leftarrow t_1$

b. Algorithm

수학적 배경지식 (cont'd)

■ **Ex: Given a = 161 and b = 28, find gcd(a, g) and the values of s and t.**

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
5	161	28	21	1	0	1	0	1	−5
1	28	21	7	0	1	−1	1	−5	6
3	21	7	0	1	−1	4	−5	6	−23
	7	0		**−1**	4		**6**	−23	

$r = r_1 - q \times r_2$, $s = s_1 - q \times s_2$, $t = t_1 - q \times t_2$

■ 임의의 정수 a를 양의 정수 n으로 나누면 몫이 q가 되고 음이 아닌 나머지 r을 얻는다.

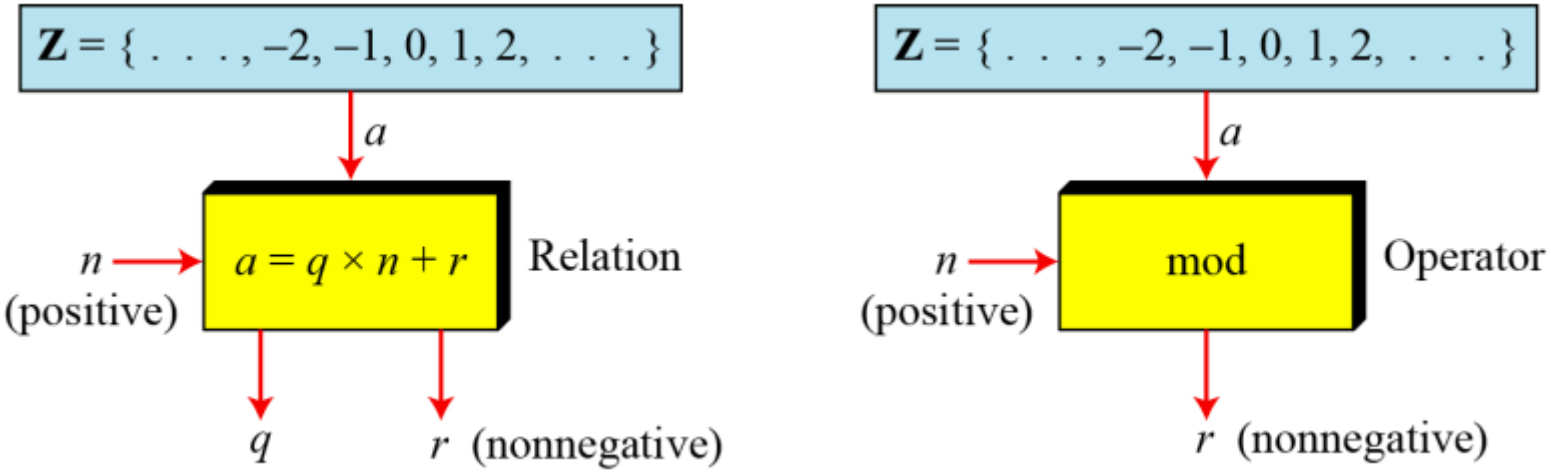
$$a = qn + r \quad (0 \le r < n)$$

$$23 = 4 \times 5 + 3; -17 = (-3) \times 5 + (-2) = (-4) \times 5 + 3$$

■ mod 연산

$$a \bmod n = r$$

$$23 \bmod 5 = 3; -17 \bmod 5 = 3$$



- mod 연산은 임의의 정수 a를 양의 정수 n으로 나누면 몫이 q가되고 음이 아닌 나머지 r을 얻는다.

$$a = qn + r \ (0 \leq r < n)$$

- mod 연산은 완전잉여계 Z_n을 만든다.

$$\mathbf{Z}_n = \{ 0, 1, 2, 3, \ldots, (n-1) \}$$

$$\mathbf{Z}_2 = \{ 0, 1 \} \quad \mathbf{Z}_6 = \{ 0, 1, 2, 3, 4, 5 \} \quad \mathbf{Z}_{11} = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 \}$$

- 합동(Congruence)

$$2 \equiv 12 \pmod{10} \qquad 13 \equiv 23 \pmod{10}$$
$$3 \equiv 8 \pmod{5} \qquad 8 \equiv 13 \pmod{5}$$

■ mod 연산의 성질

First Property: $(a + b) \bmod n = [(a \bmod n) + (b \bmod n)] \bmod n$

Second Property: $(a - b) \bmod n = [(a \bmod n) - (b \bmod n)] \bmod n$

Third Property: $(a \times b) \bmod n = [(a \bmod n) \times (b \bmod n)] \bmod n$

$$10 \bmod 3 = 1 \quad \rightarrow \quad 10^n \bmod 3 = (10 \bmod 3)^n = 1$$
$$10 \bmod 9 = 1 \quad \rightarrow \quad 10^n \bmod 9 = (10 \bmod 9)^n = 1$$
$$10 \bmod 7 = 3 \quad \rightarrow \quad 10^n \bmod 7 = (10 \bmod 7)^n = 3^n \bmod 7$$

$$a = a_n \times 10^n + \cdots + a_1 \times 10^1 + a_0 \times 10^0$$

For example: $6371 = 6 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 1 \times 10^0$

$$\begin{aligned} a \bmod 3 &= (a_n \times 10^n + \cdots + a_1 \times 10^1 + a_0 \times 10^0) \bmod 3 \\ &= (a_n \times 10^n) \bmod 3 + \cdots + (a_1 \times 10^1) \bmod 3 + (a_0 \times 10^0) \bmod 3 \\ &= (a_n \bmod 3) \times (10^n \bmod 3) + \cdots + (a_1 \bmod 3) \times (10^1 \bmod 3) + \\ &\quad (a_0 \bmod 3) \times (10^0 \bmod 3) \\ &= a_n \bmod 3 + \cdots + a_1 \bmod 3 + a_0 \bmod 3 \\ &= (a_n + \cdots + a_1 + a_0) \bmod 3 \end{aligned}$$

■ 덧셈상의 역원, 곱셈상의 역원

- Z_n상에서 덧셈상의 역원

$$a + b \equiv 0 \pmod{n}$$

- Z_n상에서 곱셈상의 역원

$$a \times b \equiv 1 \pmod{n}$$

- In modular arithmetic, an integer may or may not have a multiplicative inverse. Number a has the multiplicative Inverse if $\gcd(n, a) \equiv 1 \ (mod\ n)$

수학적 배경 지식 – 확장 유클리드 알고리즘 활용 예제#1

■ $85^{-1} \, mod \, 33 \equiv$

q	r1	r2	r	s1	s2	s	t1	t2	t
2	85	33	19	1	0	1	0	1	-2
1	33	19	14	0	1	-1	1	-2	3
1	19	14	5	1	-1	2	-2	3	-5
2	14	5	4	-1	2	-5	3	-5	13
1	5	4	1	2	-5	7	-5	13	-18
4	4	1	0	-5	7	-33	13	-18	85
	1	**0**		**7**	-33		**-18**	85	

수학적 배경 지식 – 확장 유클리드 알고리즘 활용 예제#2

■ $2145^{-1} \; mod \; 133 \equiv$

q	r1	r2	r	s1	s2	s	t1	t2	t
16	2145	133	17	1	0	1	0	1	-16
7	133	17	14	0	1	-7	1	-16	113
1	17	14	3	1	-7	8	-16	113	-129
4	14	3	2	-7	8	-39	113	-129	629
1	3	2	1	8	-39	47	-129	629	-758
2	2	1	0	-39	47	-135	629	-758	2145
	1	**0**		**47**	-135		**-758**	2145	

[문제] 확장 유클리드 알고리즘

■ 다음을 확장 유클리드 알고리즘을 이용하여 계산하시오.

- $131^{-1} \bmod 29 \equiv 2$

q	r1	r2	r	s	t
				1	0
4	131	29	15	0	1
1	29	15	14	1	-4
1	15	14	1	-1	5
14	14	1	0	2	-9

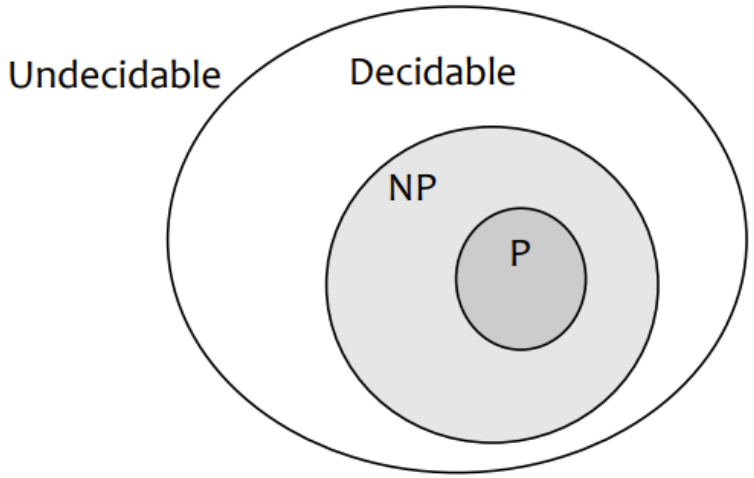
- $3714^{-1} \bmod 131 \equiv 94$

q	r1	r2	r	s	t
				1	0
28	3714	131	46	0	1
2	131	46	39	1	-28
1	46	39	7	-2	57
5	39	7	4	3	-85
1	7	4	3	-17	482
1	4	3	1	20	-567
3	3	1	0	-37	1049

■ P-NP 문제

- Undecidable
  - No algorithm that solves it.

- Decidable
  - If a problem can be solved in poly-time, it is tractable. Otherwise, it is intractable.
  - P: there exists a poly-time algorithm
  - NP: We don't know if there exists a poly-time algorithm and nobody insists that it cannot be solvable in poly-time.



■ NP 문제

- 비결정적 단계 (Nondeterministic Phase)
  - Guess

- 결정적 단계 (Deterministic Phase)
  - 다항식 시간 검증

- e.g.,) 소인수 분해 문제
  - 입력: 합성수 N
    - ✓ 비결정적 단계: p와 q를 Guess
    - ✓ 결정적 단계: p * q =? N을 다항식시간 안에 검증

■ 오일러 함수 (Euler's Phi Function)

- 오일러 함수 $\varphi(\cdot)$는 1부터 n까지 n과 서로소인 정수의 개수

$$\varphi(n) = |\{a \in \mathbb{N} | \gcd(a, n) = 1\}|$$

- p가 소수일 때, $\varphi(p) = p - 1$
- 서로소인 정수 m, n에 대하여, $\varphi(m \times n) = \varphi(m) \times \varphi(n)$

■ e.g., $\boldsymbol{\varphi(10)}$

- $gcd(1,10) = 1,\ gcd(2,10) = 2,\ gcd(3,10) = 1$
- $\gcd(4,10) = 2,\ \gcd(5,10) = 5,\ \gcd(6,10) = 2$
- $\gcd(7,10) = 1,\ \gcd(8,10) = 2,\ \gcd(9,10) = 1$
- $\boldsymbol{\varphi(10) = 4}$

■ 오일러 정리 (Euler's Theorem)

- $n \in \mathbb{Z}, \quad \forall a \in Z_n^* \quad \Rightarrow \quad a^{\varphi(n)} \equiv 1 \pmod{n}$
- $n \in \mathbb{Z}, \quad \forall a \in \mathbb{Z}_n^*, \quad \Rightarrow \quad a^{\varphi(n)+1} \equiv a \pmod{n}$
- 예) $3^{-1} \bmod 14$
  - $\varphi(14) = 6 \rightarrow 3^6 \equiv 1 \ (mod\ 14)$
  - $3 \times 3^5 \equiv 1 \ (mod\ 14)$: 곱셈상의 역원에 대한 정의와 동일
  - $3^{-1} \equiv 3^5 \pmod{14}$
  - $3^{-1} \equiv 3^5 \equiv 243 \equiv 5 \pmod{14}$

■ 소수의 개수

- 가장 큰 소수 : 6,320,430자리의 소수 (MSU)
- 소수의 개수는 무한
- n보다 작은 소수의 개수 : $f(n)$

$$[\frac{n}{ln\, n}] < f(n) < [\frac{n}{ln\, n - 1.08366}]$$

  - n의 값이 커질수록, 그 수가 소수일 확률도 $\frac{1}{\ln n}$의 분포를 따라서 작아짐
- 1,000,000보다 적은 소수의 개수는?
  - 72,383 < f(1,000,000) <78,543.
  - 실제 78,498개의 소수

- 선택된 수 k 가 소수일 확률

$$P(k \text{ is prime}) \approx \frac{1}{ln(k)}$$

03

RSA 암호

■ 가장 많이 사용되고 있는 공개키 암호시스템

- Rivest, Shamir, and Adleman의 이름에서 RSA
- Clifford Cocks, an English mathematician working for the UK intelligence agency, described an equivalent system in 1973, but it was mostly considered a curiosity and, as far as is publicly known, was never deployed. His discovery, however, was not revealed until 1998 due to its top-secret classification, and Rivest, Shamir, and Adleman devised RSA independently of Cocks' work.

■ 키 생성

1. 서로 다른 두 소수 p와 q 선택 (크기가 동일한 1024비트 이상의 수로 선택) ; ($P(k \text{ is prime}) \approx \frac{2}{\ln(2^{1024})} = \frac{2}{1024\ln(2)} \approx \frac{1}{355}$)
2. $n = p \times q$값을 계산.
3. $\varphi(n) = (p-1)(q-1)$
4. $1 < e < \varphi(n) - 1$의 범위에서 $\varphi(n)$과 서로소인 e를 선택
5. $d = e^{-1} \bmod \varphi(n)$ (확장 유클리드 알고리즘)

- **(e,n) : public-key**
- **(d,n) : private-key**

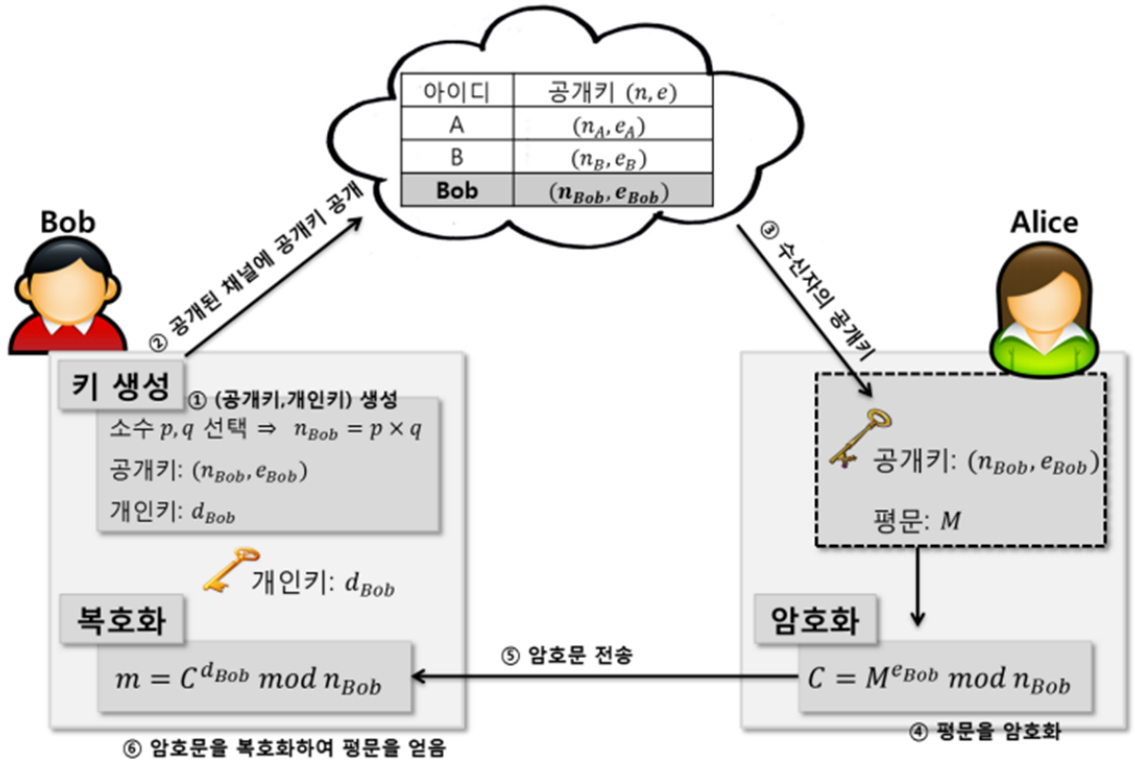
예)

1. $p = 127,\ q = 131$
2. $n = p \times q = 127 \times 131 = 16637$
3. $\varphi(n) = \varphi(p \times q) = \varphi(p) \times \varphi(q) = (p-1) \times (q-1) = 126 \times 130 = 16380$
4. 공개키 e는 집합 $\mathbb{Z}^*_{\varphi(n)}$에서 $gcd(e, \varphi(n)) = 1$을 만족하는 $e = 17$로 선택
5. $d \equiv e^{-1} \equiv 17^{-1} \equiv 14453\ (mod\ 16380)$
6. 공개키 $(\boldsymbol{n = 16637, e = 17})$, 개인키 $(\boldsymbol{d = 14453})$

■ 암/복호화

- Encryption: $c = m^e \ mod \ n$
  - Note m < n (for uniqueness)
- Decryption: $m = c^d \ mod \ n$

- (e,n) : public-key
- (d,n) : private-key



■ RSA 암호의 정확성 (correctness)

× **Decryption** : $c^d \equiv (m^e)^d \equiv m^{ed} \pmod{n}$

We know that $ed = k\varphi(n) + 1$

Case 1: GCD(m, p)= 1 and GCD(m, q)= 1

$m^{ed} \equiv m^{k\varphi(n)+1} \equiv (m^{\varphi(n)})^k \cdot m \equiv 1^k \cdot m \equiv m \pmod{n}$ (∵ 오일러 정리 $m^{\varphi(n)} \equiv 1 \bmod n$)

Case 2 : m = i x p and GCD(m, q)= 1

$m^{k\varphi(n)} \equiv \left(m^{\varphi(n)}\right)^k \equiv \left(m^{\varphi(p)\varphi(q)}\right)^k \equiv \left(m^{(q-1)}\right)^{k(p-1)} \equiv 1^{k(p-1)} \ (\boldsymbol{mod\ q})$

(∵ 페르마 소정리 $m^{q-1} \equiv 1 \bmod q$)

$m^{k\varphi(n)}$ = (j x q) + 1)

$m^{k\varphi(n)+1}$ = ((j x q) + 1) x m = (j x q x m) + m = (j x q x i x p) + m

= (j x i x n) + m

⇒ $m^{k\varphi(n)+1} \equiv m \pmod{n}$

Case 3 : m = i x q and GCD(m, p)= 1 : *similar to Case* 2

■ RSA 암호 예시

p = 47 and q = 71, $n = p*q$ = 3337

- $(p\text{-}1)*(q\text{-}1)$ = 46 * 70 = 3220, $GCD\ (\ e, (p\text{-}1)*(q\text{-}1)) = 1$
- Choose ***e*** at random to be 79
- ***d*** = 79^{-1} mod 3220 = 1019
- To encrypt message ***m* = 6882326879666683**
- m_1 = 688 m_2 = 232 m_3 = 687
  m_4 = 966 m_5 = 668 m_6 = 003
- $c_1 = m_1^{\,e}\ mod\ n = 688^{79}\ mod\ 3337$ = 1570
- ***c* = 1570 2756 2091 2276 2423 158**
- To decrypt, $m_1 = c_1^{\,d}\ mod\ n = 1570^{1019}\ mod\ 3337$ = 688

■ RSA 암호의 안전성

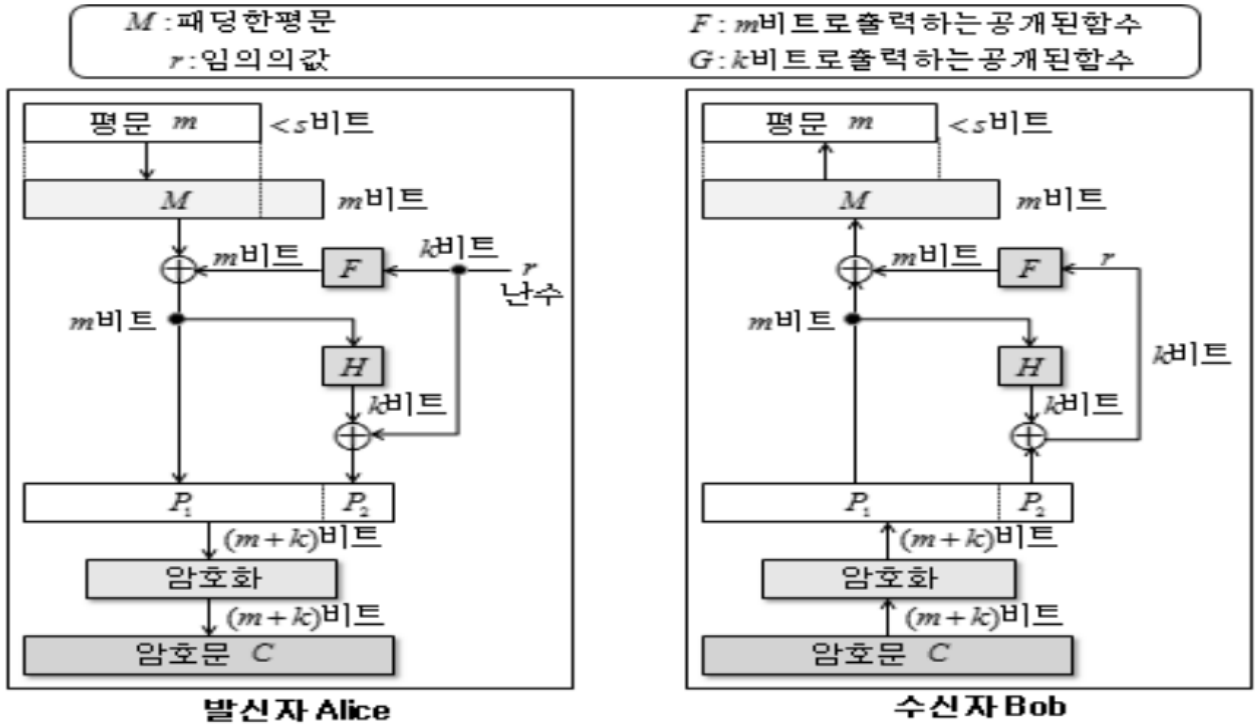
- RSA 문제
  - $c \equiv m^e \ (mod\ n)$가 주어졌을 때의 c의 $e^{th}\ root$를 구하는 문제
    - ✓ 인수분해 문제 → RSA 문제
      - $n = p \times q$을 인수분해 → $\varphi(n)(= (p-1)(q-1)) \rightarrow e$의 곱셈상의 역원 d를 계산
    - ✓ RSA 문제 → 인수분해 문제
      - Not known yet

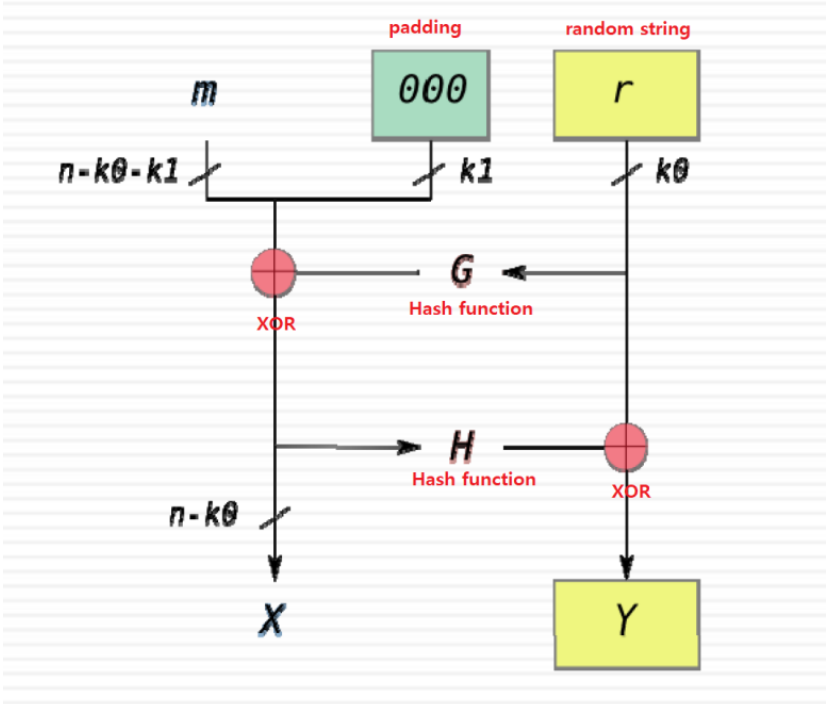
■ RSA 이용 시 권고사항

- N의 비트는 적어도 (서명의 경우) 2048비트가 되어야 한다.
- 서로 다른 두 소수 p와 q는 적어도 1024비트 이상이 되어야 한다.
- 서로 다른 두 소수 p와 q가 너무 가까이 있는 소수는 선택하지 않는다.
- p-1과 q-1은 적어도 하나의 큰 소인수를 가져야 한다.
- 비율 p/q가 작은 분자나 작은 분모를 갖는 유리수와 가까이 있으면 안된다.
- N을 공통적으로 이용하지 않는다.
- 공개키 e는 $2^{16} + 1 = 65537$을 이용하거나 혹은 65537과 가까이 있는 값을 이용한다.
- 만약 개인키 d가 노출되었을 경우, 수신자는 반드시 공개키 n과 e, 개인키 d를 즉시 교체해야 한다.
- OAEP를 이용

■ OAEP (Optimal Asymmetric Encryption Padding)

- P1 || P2 where P1 = $(m||0^{k1}) \oplus G(r)$, P2 = $H(P1) \oplus r$
- $|P1| = k - k_0$, $|P2| = k_0$





Thank You

- **Lab:** https://mose.kookmin.ac.kr
- **Email:** sh.jeon@kookmin.ac.kr