

암호학과 네트워크 보안

- 10장 비대칭 키 암호 -

김 혜 정(hyejeong@pel.sejong.ac.kr)

세종대학교 프로토콜공학연구실

목 차

- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

목 차

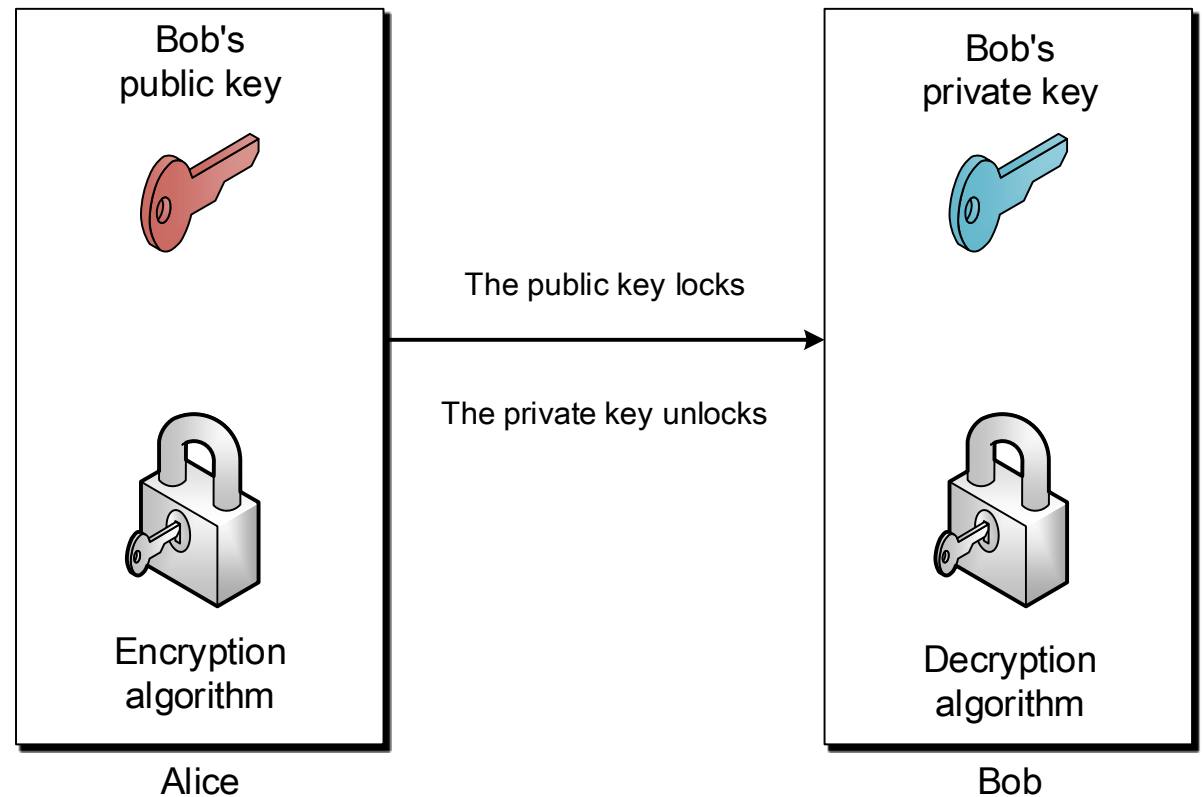
- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

비대칭 키 (1/7)

- 개요

- 정의

- 암호화 시 두 개의 서로 다른 키를 사용하는 암호 방식으로 공개 키라고도 함



비대칭 키 (2/7)

- 대칭 키와 비대칭 키

- 차이점

- 비밀 개수

- 대칭 키: $n(n-1)/2$

- e.g., a, b, c 3명이 사용하는 경우, 대칭 키는 a-b, b-c, c-a 간 각각 다른 암호 키를 사용함(즉, $nC_2 = n(n-1)/2$)

- 비대칭 키: $2n$

- e.g., a, b, c 3명이 사용하는 경우, 비대칭 키는 (공개 키, 개인 키) $\times n$

- 암호 방식

- 대칭 키: 기호(문자나 비트)를 대체하거나 치환하는 것에 중점

- e.g., 64비트의 평문을 대체, 전치를 시키는 DES(Data Encryption Standard)

- 비대칭 키: 숫자를 다른 숫자로 변경하는 것에 중점

- e.g., $n = p \times q$ 와 같이 하나의 복잡한 합성수나 소수를 쓰는 RSA(Rivest-Shamir-Adleman)

비대칭 키 (3/7)

- 대칭 키와 비대칭 키

- 하이브리드 암호 시스템

1. 비대칭 키 암호화

- 데이터 전송을 시작하기 전에 비대칭 암호화를 통해 비밀 키 (대칭 키)를 수신자의 공개 키로 암호화함

2. 대칭 키 암호화

- 송신자는 대칭 키를 사용하여 실제 데이터를 암호화함
- 암호화된 데이터와 대칭 키(비대칭 암호화로 암호화된)를 함께 수신자에게 전송함

3. 비대칭 키 암호화

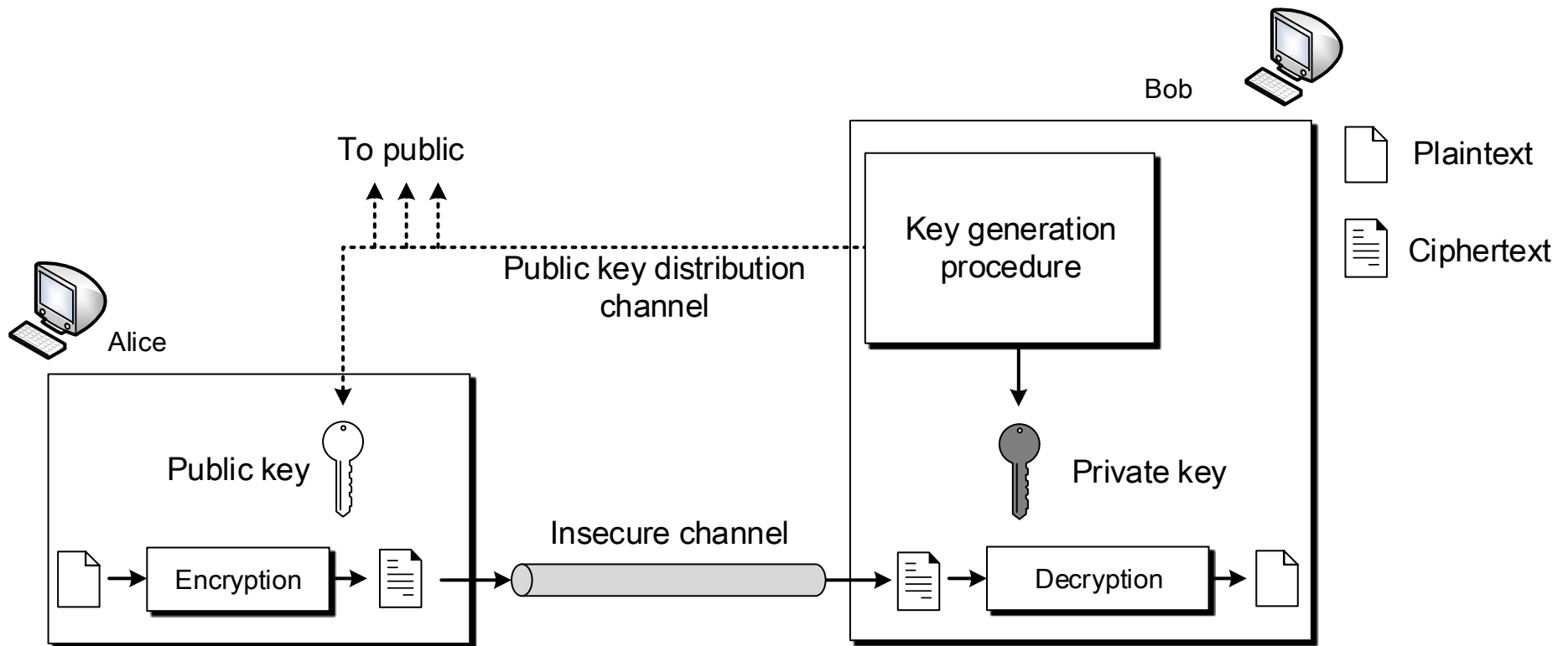
- 수신자는 자신의 개인 키를 사용하여 대칭 키를 복호화함
- 수신자는 복호화된 대칭 키를 사용하여 데이터를 복호화함

- e.g., PGP (Pretty Good Privacy)

비대칭 키 (4/7)

- 비대칭 키의 특징 (1/2)

- 수신자(Bob)가 책임을 지고 키 생성과 키 배분을 해야 함
 - 키 배분 채널은 무결성과 인증을 제공해야 함



비대칭 키 (5/7)

- 비대칭 키의 특징 (2/2)

- 양방향 통신에서 동일한 키 쌍을 사용할 수 없음
 - 송신자(Alice)가 메시지 m 을 수신자(Bob)에게 보내고, Bob이 메시지 m 에 대한 응답을 하기 위해선 Alice가 자신의 키 쌍을 가지고 있어야 함
- 대칭 키의 키 교환 문제를 해결함
 - 송신자는 공개 키(또는 개인 키)를 알고, 수신자는 개인 키(또는 공개 키)를 알면 되기 때문에 두 사람이 동일한 키를 공유하지 않음

비대칭 키 (6/7)

- 트랩도어 일방향 함수(TOWF, Trapdoor OWF)
 - 아래 성질 1, 2, 3을 만족하는 일방향 함수의 일종임
- 일방향 함수(OWF, One-Way Function)
 - 성질 1: 주어진 x 가 있다면 $y = f(x)$ 를 계산하는 것은 쉬움
 - 성질 2: 주어진 y 가 있다면 $x = f^{-1}(x)$ 를 계산하는 것은 어려움
- 트랩도어(Trapdoor)
 - “다락방 문”을 뜻하는 말로 시스템 보안이 제거된 비밀통로를 의미함
 - 성질 3: y 와 비밀(트랩도어)가 주어지면 x 를 쉽게 구할 수 있음

비대칭 키 (7/7)

- 트랩도어 일방향 함수(TOWF, Trapdoor OWF)

- 예제 10.1

n 이 매우 큰 수라면, $n = p \times q$ 는 일방향 함수이다

- 일방향 함수 $y = f(x)$ 에 적용해보자
- 소인수분해 문제에 기반하여 x 는 두 개의 소수로 이루어진 순서쌍 (p, q)
- y 는 n 임

- 예제 10.2

n 이 매우 큰 수라면, 함수 $y = x^k \bmod n$ 은 하나의 트랩도어 일방향 함수이다

- 주어진 x, k, n 이 있다면 y 를 구하는 것은 쉬우나 y, k, n 이 주어졌을 때는 x 를 구하는 것이 어려움 (이산 로그 문제)
- $k' = 1 \bmod \varphi(n)$ 이 되는 k' 를 알고 있다면 $x = y^{k'} \bmod n$ 을 이용하여 x 를 구할 수 있음

목 차

- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

배낭 암호 (1/9)

- 개요

- 배경

- Merkle과 Hellman이 만든 최초 공개 키 암호시스템으로, 오늘날의 표준에는 안전하지 않지만 추후 공개 키 암호에 토대가 됨

- 정의

- 배낭 속에 있는 정해진 수들의 집합들의 원소를 알고 원소들의 합을 구하는 것은 쉽지만 합을 알고 각 원소들을 알기는 어렵다는 문제를 기반으로 한 암호

배낭 암호 (2/9)

- 동작

- 키 구성

- 2개의 k 순서쌍 ($k \in \mathbb{Z}$)

- $a = [a_1, a_2 \cdots a_k]$: 사전에 정의된 집합
 - $x = [x_1, x_2 \cdots x_k]$: 0과 1로만 이루어진 집합으로, a 가 배낭에 들어갈지를 결정

- 함수

- 배낭 속 원소들의 합 표현

- $s = knapsackSum(a, x) = x_1 a_1 + x_2 a_2 + \cdots + x_k a_k$
 - $x = inv_knapsackSum(s, a)$

- a 와 x 를 알고 s 를 구하는 것은 쉬우나 s 와 a 를 알고 x 를 구하는 것은 어려움

- 함수 $knapsackSum(a, x)$ 는 일방향함수임

배낭 암호 (3/9)

- 동작

- 초증가 순서쌍(Superincreasing tuple)

- 집합 a 가 있을 때, $a_i \geq a_1 + a_2 + \dots + a_{i-1}$ 일 경우를 뜻함

```
knapsackSum( $x[1 \dots k]$ ,  $a[1 \dots k]$ ){
```

```
   $s \leftarrow 0$ 
```

```
  for( $i=1$  to  $k$ ){
```

```
     $s \leftarrow s + a_i \times x_i$ 
```

```
  }
```

```
  return  $s$ 
```

```
}
```

```
inv_knapsackSum( $s$ ,  $a[1 \dots k]$ ){
```

```
  for( $i=k$  down to  $1$ ){
```

```
    if ( $s \geq a_i$ ){
```

```
       $x_i \leftarrow 1$     //  $x_i = 1$ 이면 배낭 속에 존재한다는 의미
```

```
       $s \leftarrow s - a_i$  // 존재하므로 삭제
```

```
    }
```

```
    else  $x_i \leftarrow 0$ 
```

```
  }
```

```
  return  $x[1 \dots k]$ 
```

```
}
```

배낭 암호 (4/9)

- 동작

- 초증가 순서쌍(Superincreasing tuple)
 - 예제 10.3

$a = [17, 25, 46, 94, 201, 400]$ 이고 $s = 272$ 일 때, $knapsackSum$ 과 $inv_knapsackSum$ 를 계산하라.

i	a_i	s	$s \geq a_i$	x_i	$s \leftarrow s - a_i$
6	400	272	X	$x_6 = 0$	272
5	201	272	O	$x_5 = 1$	71
4	94	71	X	$x_4 = 0$	71
3	46	71	O	$x_3 = 1$	25
2	25	25	O	$x_2 = 1$	0
1	17	0	X	$x_1 = 0$	0

- 이 경우, $x = [011010]$ 이며, 배낭 속에는 25, 46, 201이 들어가있음을 의미함

배낭 암호 (5/9)

- 동작

- 키 생성

1. 초증가 k 순서짜 $b = [b_1, b_2 \cdots b_k]$ 를 만듦
2. $n > b_1 + b_2 + \cdots + b_k$ 인 모듈로 n 선정함
3. n 하고 서로소이면서 $1 \leq r \leq n - 1$ 인 임의의 정수 r 선택함
4. $t_i = r \times b_i \bmod n$ 을 만족하는 임시 k 순서짜 $t = [t_1, t_2 \cdots t_k]$ 생성함
5. k 개의 원소들과 매칭되는 새로운 순서짜 $a = \text{permute}(t)$ 를 구함
6. k 순서짜 a 를 공개 키로 사용하고 n, r, b 는 개인 키로 사용함

배낭 암호 (6/9)

- 동작

- 암호화

1. 메시지를 k 순서짜 $x = [x_1, x_2 \cdots x_k]$ 로 변환함
 - x_i 는 0 또는 1임
 - 순서짜 x 는 평문임
2. *knapsackSum* 함수를 사용하여 s 를 계산함
 - 원소 합 s 는 암호문임

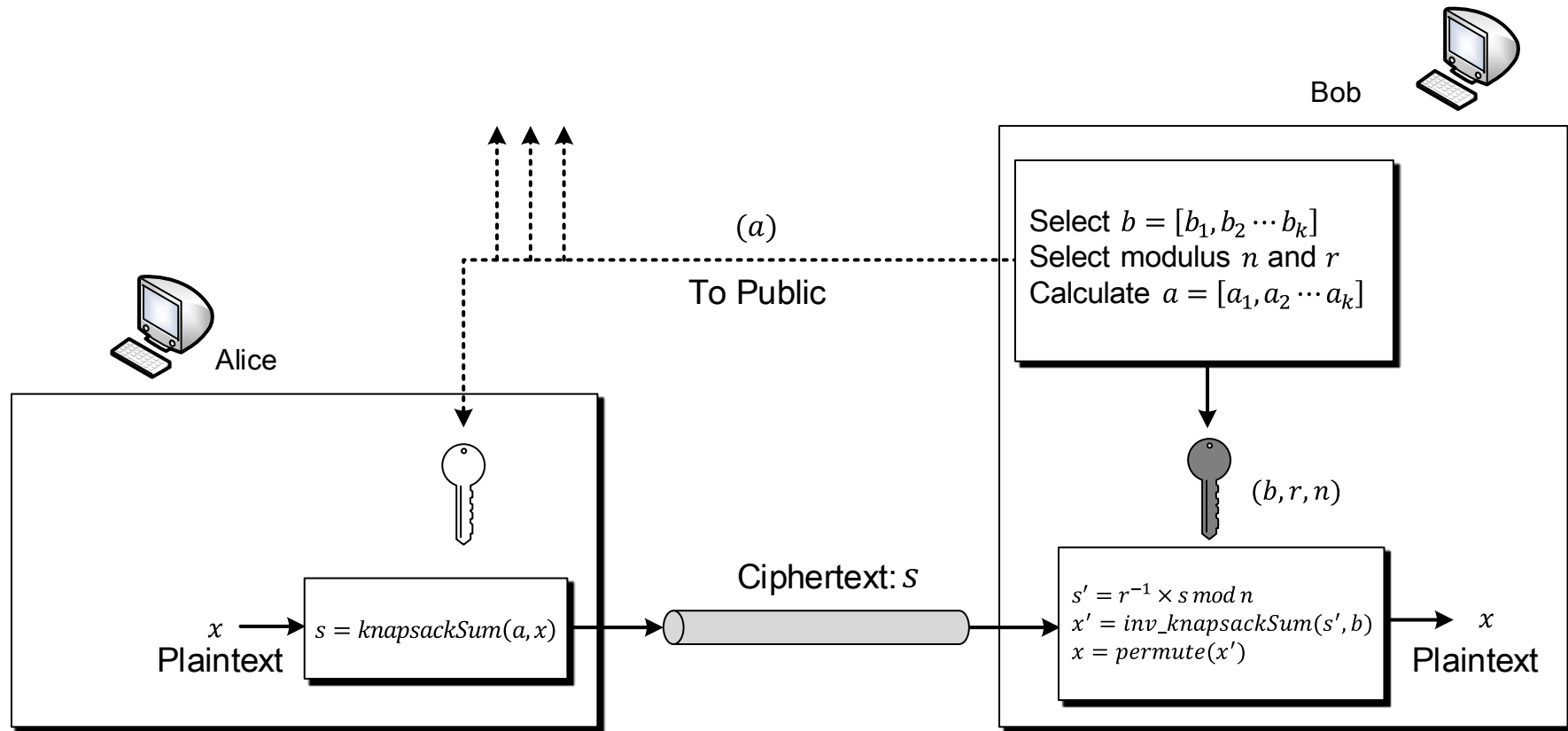
- 복호화

1. $s' = r^{-1} \times s \bmod n$ 계산함
2. *inv_knapsackSum*을 사용하여 x' 를 구함
3. x' 를 치환하여 x 를 구함

배낭 암호 (7/9)

- 동작

- Bob이 가지고 있는 트랩도어는 $s' = r^{-1} \times s \bmod n$ 와 초증가 순서쌍 b 임



배낭 암호 (8/9)

- 동작

- 예제 10.4 (1/2)

절차를 보여주기 위한 간단한 예를 들어보자.

- 키 생성

- Bob이 초증가 순서쌍 $b = [7, 11, 19, 39, 79, 157, 313]$ 을 만듦
- Bob이 모듈로 $n = 900, r = 37$ 을 선택하고 치환표로는 $[4\ 2\ 5\ 3\ 1\ 7\ 6]$ 을 선택함
- Bob은 순서쌍 $t = [259, 407, 703, 543, 223, 409, 781]$ 을 계산함
 - $t_i = r \times b_i \bmod n$
- Bob이 치환표를 가지고 $a = \text{permute}(t)$ 를 수행함
 - $t = [543, 407, 223, 703, 259, 781, 409]$
- Bob이 a 를 공개하고 n, r, b 는 비밀로 함

배낭 암호 (9/9)

- 동작

- 예제 10.4 (2/2)

절차를 보여주기 위한 간단한 예를 들어보자

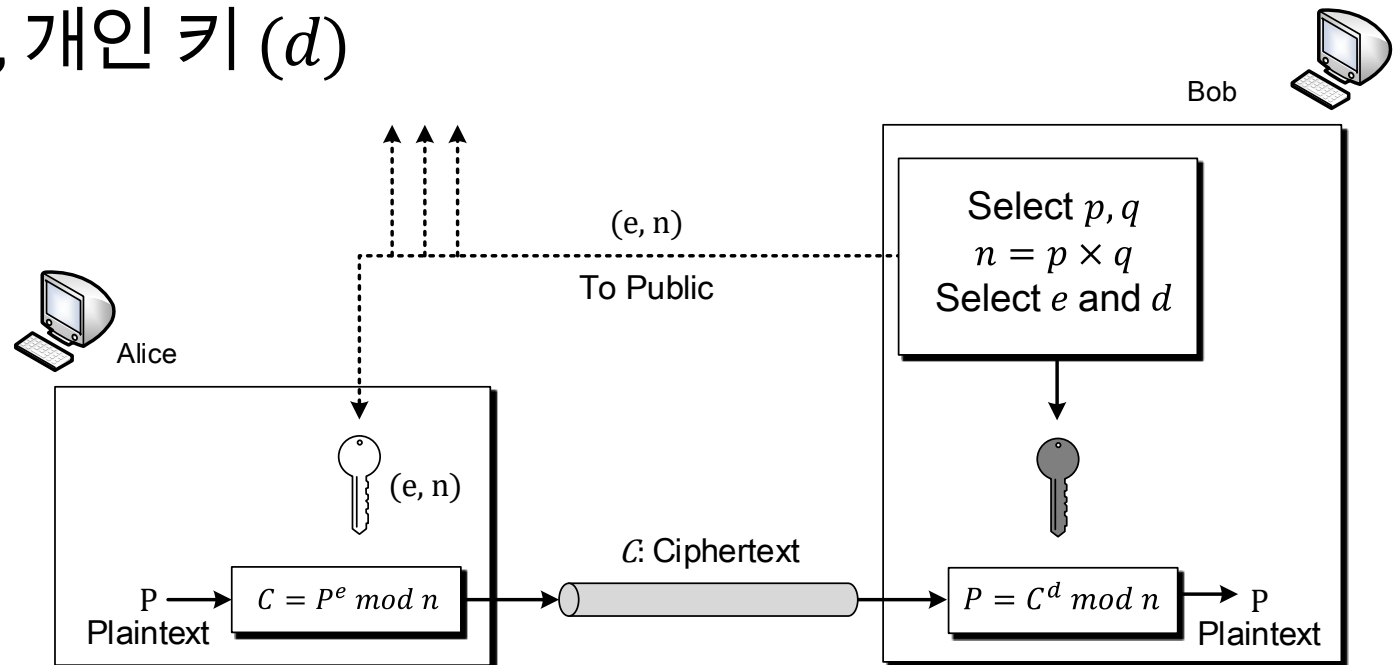
- Alice가 단 하나의 문자 g 를 Bob에게 보낼 것임
 - 7비트 ASCII 코드를 이용해 g 를 부호화함
 - $g = (1100111)_2$
 - 순서쌍 $x = [1, 1, 0, 0, 1, 1, 1]$ 생성함
 - x 가 평문임
 - S 를 계산함
 - $543 \times 1 + 407 \times 1 + 223 \times 0 + 703 \times 0 + 259 \times 1 + 781 \times 1 + 409 \times 1 = 2399$
- Bob이 s 를 사용하여 x 를 복호화함
 - $s' = r^{-1} \times s \bmod n = (37^{-1} \times 2399) \bmod 900 = 527$
 - $x = \text{inv_knapsackSum}(s', b) = [1, 1, 0, 1, 0, 1, 1]$ 를 계산함
 - $x = \text{permute}(x') = [1, 1, 0, 0, 1, 1, 1]$ 을 계산하여 x 를 얻음
 - x 를 ASCII 코드로 변환하면 g 가 나옴

목 차

- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

RSA (1/26)

- 개요
- 정의
 - 3명의 개발자(Rivest, Shamir, Adleman)의 성을 따온 이름으로 두 개의 큰 소수로 생성된 키 쌍을 사용한 공개 키 암호 시스템
- 키 구성
 - 공개 키 (e, n) , 개인 키 (d)



RSA (2/26)

- 동작

- 키 생성

- 곱셈군 $G = \langle Z_{\varphi(n)}^*, \times \rangle$ 사용

- 합성수 $n = p \times q$
 - 이때, $\varphi(n), p, q$ 는 비밀임

- 과정

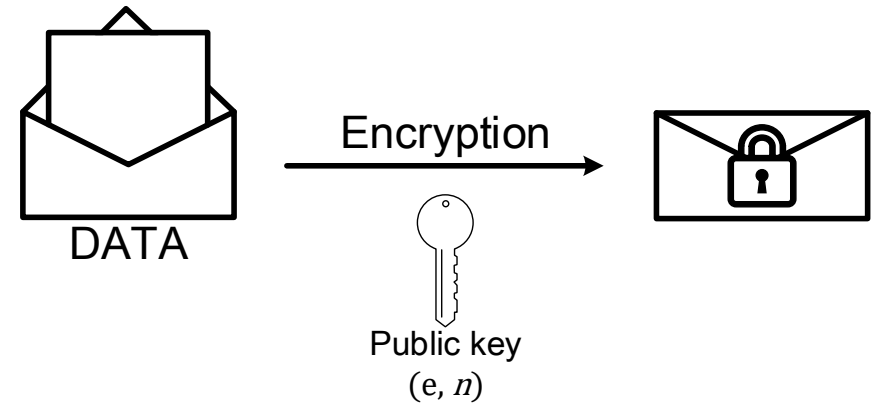
1. 1024bit 이상인 p 와 q 를 선택하여 n 생성
2. $\varphi(n)$ 과 서로소인 정수 e 선택 ($1 < e < \varphi(n)$)
3. $ed \bmod \varphi(n) = 1$ ($1 < d < \varphi(n)$)
 - e 의 곱셈의 역원인 d

RSA (3/26)

- 동작

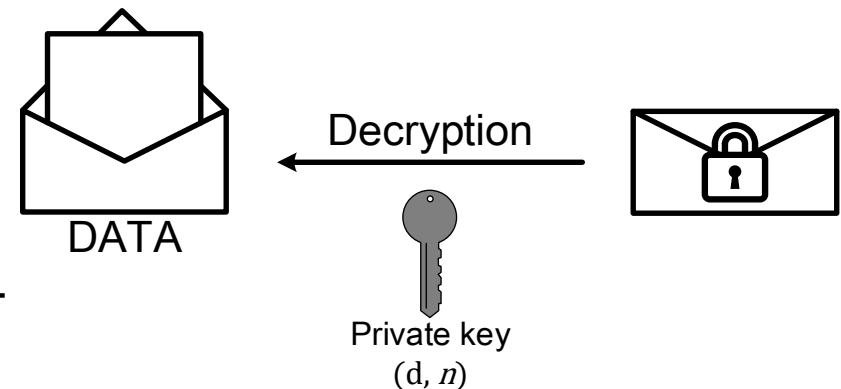
- 암호화

- $C = P^e \bmod n$
 - 메시지 P 가 n 보다 큰 경우
작은 블록으로 나누어 줌



- 복호화

- $P = C^d \bmod n$
 - 메시지 P 가 나누어진 경우
해당 변환방식이 공유되어야 함



RSA (4/26)

- 동작

- 암호·복호화 역관계 증명

- 오일러 정리 활용

- 만약 $n = p \times q$, $a < n$ 이고 k 는 정수라면, $a^{k \times \varphi(n) + 1} \equiv a \pmod{n}$ 이 성립된다.

- $ed \pmod{\varphi(n)} = 1$

- 즉, $ed = k \times \varphi(n) + 1$ 으로 표현 가능

- Bob이 복호화한 평문을 P_1 이라고 하고, 이것이 P 와 같음

- $P_1 = C^d \pmod{n} = (P^e \pmod{n})^d \pmod{n} = P^{ed} \pmod{n}$

- $P_1 = P^{ed} \pmod{n} \rightarrow P_1 = P^{k \times \varphi(n) + 1} \pmod{n} = P \pmod{n}$

RSA (5/26)

- 동작

- 예제 10.5

주어진 수를 통해 평문과 복호화를 거친 암호문이 같은지 살펴보자.

- Bob이 $p = 7, q = 11$ 을 선택했다고 가정하자
 $n = 7 \times 11 = 77, \varphi(77) = 6 \times 10 = 60$
- Z_{60}^* 에서 e 와 d 를 선정할 수 있다. e 를 13으로 선택했다면 d 는 37이 됨
 - $ed \bmod \varphi(n) = 13d \bmod 60 = 1$
 - 유클리드 확장 알고리즘을 사용해서 $1 = -23 \cdot 13 + 5 \cdot 60 \therefore -23$
 - 양수로 전환하면 $60 - 23 = 37 \therefore d = 37$
- Alice가 평문 5를 Bob에게 보낸다고 가정하자
- P: 5, C: $5^{13} = 26 \bmod 77$
- Bob은 자신의 개인 키(d)를 사용하여 복호화함
- C: 26 P: $26^{37} = 5 \bmod 77$

RSA (6/26)

- 동작

- 예제 10.7

Jennifer는 자신의 키 쌍을 만들기 위해 $p = 397, q = 401$ 을 선택하고 $e = 343$ 과 $d = 12007$ 을 선택했다. 만약에 Ted가 e 와 n 을 알면 메시지를 Jennifer에게 보낼 수 있다는 것을 보이시오.

- Jennifer에게 보내고자 하는 메시지가 “NO”라고 가정하자
- Ted는 각 문자를 숫자로 된 코드(00부터 25까지)로 바꿈
 $N \rightarrow 13, O \rightarrow 14$
- 바꾼 문자코드를 이어붙여 4자리 숫자를 만듦
 $NO \rightarrow 1314$
- e 와 n 을 사용하여 암호화함
$$C: 1314^{343} = 33677 \text{ mod } 159197$$
- Jennifer는 33677을 받게 될 것이고 이를 개인 키 d 를 사용하여 복호화함
$$P: 33677^{12007} = 1314 \text{ mod } 159197$$
- Jennifer는 이를 다시 문자로 변환하여 “NO”를 얻음

RSA (7/26)

- 응용

- 데이터 암호화

- 전자메일

- e.g., PGP(Pretty Good Privacy)

- 파일 암호화

- 파일을 암호화하여 저장하거나 전송할 때 사용됨

- 디지털 서명

- 소프트웨어 배포

- 소프트웨어에 디지털 서명을 추가하여 배포 시 사용

- 인증

- 웹 사이트 인증

- SSL/TLS 프로토콜의 일부로 사용되어 웹 사이트의 인증을 제공함

- 사용자 인증

- 사용자 비밀번호를 암호화하거나 두 단계 인증 시 사용함

RSA (8/26)

- 공격

- RSA에 대한 치명적인 공격은 아직 알려진 것이 없으나, 문제가 있는 평문이나 약한 매개변수 선정 혹은 적합하지 않은 암호적용 등에 대한 공격 방법이 존재함

- 종류

- 소인수분해 공격
- 선택 암호문 공격
- 암호화 지수에 대한 공격
- 평문 공격
- 일반 모듈로 공격
- 타이밍 공격

RSA (9/26)

- 공격

- 소인수분해 공격

- RSA의 안전성은 소인수분해가 어렵다는 것에 근거하기 때문에 n 값은 공개하더라도 p 와 q 는 비밀로 유지함
- 공격자가 p 와 q 를 알아냈을 경우, $\varphi(n)$ 를 계산할 수 있게 되고 e 가 공개되어 있기 때문에 d 까지 계산할 수 있음
- 이를 고려하여 n 을 300자리 이상의 십진수(대략 1024비트)를 사용해야하며, 최근에는 4096비트까지 사용할 것을 권장

RSA (10/26)

- 공격

- 선택 암호문 공격

- RSA의 곱셈성질을 이용한 공격으로 Bob이 Alice가 보낸 암호문이 아닌 공격자 Eve가 보낸 암호문을 복호화하는 공격
 - Eve가 Alice가 보낸 C 를 가로챈 다음, 아래와 같은 절차를 거침
 1. Eve는 Z_n^* 에 속하는 임의의 정수 X 선택함
 2. Eve는 $Y = C \times X^e \bmod n$ 를 계산함
 3. Eve는 Y 를 Bob에게 보내어 복호화를 부탁하여 $Z = Y^d \bmod n$ 를 얻음
 4. Eve는 다음과 같이 해서 쉽게 P 를 구함
 - $Z = Y^d \bmod n = (C \times X^e)^d \bmod n = (C^d \times X^{ed}) \bmod n$
 $= (C^d \times X) \bmod n = (P \times X) \bmod n$
 - $Z = (P \times X) \bmod n \rightarrow P = (Z \times X^{-1}) \bmod n$

RSA (11/26)

- 공격

- 암호화 지수에 대한 공격 (1/6)

- 암호화 지수 e 가 작을 때 발생하는 취약점을 노린 공격

- e.g., $e = 3$

- 과거에 많이 사용되었지만, 패딩이 불완전할 경우 보안 취약점이 있을 수 있음

- Coppersmith 정리 공격

- 모듈로가 n 이고 차수가 e 인 다항식 $f(x)$ 에서 만약 한 개의 근이 $n^{1/e}$ 보다 작으면 복잡도가 $\log n$ 인 알고리즘을 사용하여 근을 구할 수 있다

- 즉, 작은 e 와 그보다 더 작은 p 일 경우를 말함

- $C = P^e \bmod n$ 에서 P^e 가 n 보다 작으면 모듈로 연산이 필요 없음

- e.g., $e = 3, n = 1000, P = 5$

- $C = 5^3 \bmod 1000 = 125 \bmod 1000$

RSA (12/26)

- 공격

- 암호화 지수에 대한 공격 (2/6)

- 브로드캐스트 공격 (1/2)

- 다수의 수신자가 같은 메시지를 서로 다른 공개 키로 암호화하는데 이때, 동일한 작은 지수 e 를 사용할 경우에 가능한 공격
 - e.g., Alice가 한 메시지 P 를 3명의 수신자에게 보내는데 $e = 3$ 을 사용하고, 모듈로 값은 각각 n_1, n_2, n_3 를 사용한다고 가정함
 - $C_1 = P^3 \bmod n_1$ $C_2 = P^3 \bmod n_2$ $C_3 = P^3 \bmod n_3$
 - CRT를 사용하여 P^3 을 모듈로 $n_1 \times n_2 \times n_3$ 에 대해 재구성함
 - 재구성된 P^3 은 $n_1 \times n_2 \times n_3$ 보다 작거나 같으므로, $C = P^3$ 이 모듈로 상의 항등이 아니라 일반 산술적인 의미에서 항등이 될 수 있음

RSA (13/26)

- 공격

- 암호화 지수에 대한 공격 (3/6)

- 브로드캐스트 공격 (2/2)

- 예제 10.8

- Prob)

- $e = 3, P = 42$

- $n_1 = 61 \times 53 = 3233$

- $n_2 = 47 \times 59 = 2773$

- $n_3 = 43 \times 71 = 3053$

- Sol)

- 공격자가 C_1, C_2, C_3 와 각 n_i 를 알고 있기 때문에 다음을 만족하는 P^3 을 찾으면 됨

- $P^3 \equiv C_1 \pmod{n_1}, P^3 \equiv C_2 \pmod{n_2}, P^3 \equiv C_3 \pmod{n_3}$

- $C_1 = 42^3 \bmod 3233 = 74088 \bmod 3233 = 1392$

- $C_2 = 42^3 \bmod 2773 = 74088 \bmod 2773 = 1712$

- $C_3 = 42^3 \bmod 3053 = 74088 \bmod 3053 = 1986$

RSA (14/26)

- 공격

- 암호화 지수에 대한 공격 (4/6)

- 관련된 메시지 공격

- Franklin-Reiter이 발견한 방법으로, 두 개의 관련 메시지가 선형 관계에 있을 때 적용되는 공격

- e.g., $P_2 = a \times P_1 + b$ 란 관계를 알고 있고 $e = 3, a = 2, b = 1$

- $C_1 = P_1^3 \bmod n$

- $C_2 = (2P_1 + 1)^3 \bmod n$

RSA (15/26)

- 공격

- 암호화 지수에 대한 공격 (5/6)

- 짧은 패드 공격 (1/2)

- 패딩이 짧아서 평문 길이가 충분히 길지 않거나, 평문의 구조를 통해 공격자가 평문을 복구할 수 있는 상황에서 발생하는 공격

- 공격 전제 조건

- N, e : 공개 키
 - n : N 의 비트 길이
 - m : $\frac{n}{e^2}$ 로 정의되고, 평문 M 을 나눔
 - 높은 부분: $M_1 = 2^m M + r_1$ ($0 \leq r_1 \leq 2^m$)
 - 낮은 부분: $M_2 = 2^m M + r_2$ ($0 \leq r_2 \leq 2^m$)
 - M : 평문 메시지로, 길이가 $n - m$ 이하

RSA (16/26)

- 공격

- 암호화 지수에 대한 공격 (6/6)

- 짧은 패드 공격 (2/2)

- 공격 개요

- 1. 두 개의 다항식 정의

- $g_1(x) = x^e - C_1$
 - $g_2(x, y) = (x + y)^e - C_2$

- 2. 공통근 찾기

- $y = r_2 - r_1$ 로 설정하면, 두 다항식은 공통근 M_1 을 가짐

- 3. Resultant 계산

- $h(y) = \text{res}_x(g_1, g_2)$

- 4. Coppersmith의 정리 적용

- h 의 차수는 최대 e^2
 - $|r_2 - r_1| < 2^m < N^{1/e^2}$ 이므로, Coppersmith의 정리를 통해 y 를 구할 수 있음

- 5. Franklin Reiter 관련 메시지 공격 적용

RSA (17/26)

- 공격

- 평문 공격 (1/2)

- 짧은 메시지 공격

- 공격자가 이미 많은 평문을 알고 있을 때, 가로챈 암호문과 동일한 암호문이 나올 때까지 자신이 가지고 있는 평문을 다 암호화하는 공격
 - e.g., Alice가 4자리 숫자를 Bob에게 보낸다면, Eve는 0000부터 9999까지 다 암호화해보는 것을 의미함

- 감추어지지 않는 공격

- 암호화를 했음에도 도로 자기 자신이 되어 감추어지지 않는다는 메시지에 대한 공격
 - e.g., 암호화된 메시지가 원래의 평문과 유사한 구조를 가지는 경우

RSA (18/26)

- 공격

- 평문 공격 (2/2)

- 순환 공격

- 암호문이 평문의 치환이라는 것과 암호문을 연속해서 암호화하면 궁극적으로 평문에 도달하게 된다는 사실에 기초한 공격
 - 똑같은 암호문이 나올 때까지 반복함
 - e.g., 가로채 암호문: C , $P = C_{k-1}$

$$\begin{aligned}C_1 &= C^e \bmod n \\C_2 &= C_1^e \bmod n \\&\vdots \\C_k &= C_{k-1}^e \bmod n \rightarrow C_k = C \text{라면, 여기서 멈춤}\end{aligned}$$

RSA (19/26)

- 공격

- 일반 모듈로 공격

- 여러 개의 암호문이 동일한 모듈로를 사용할 경우, 공격자는 이 암호문들 간의 관계를 활용하여 원래의 평문을 복구하려는 공격

- e.g., 동일한 모듈로와 평문, 작은 지수 e_i 를 사용할 경우

- $C_1 = M^{e_1} \bmod n$ $C_2 = M^{e_2} \bmod n$
- $C_1^{e_2} \equiv M^{e_1 \cdot e_2} \bmod n$ $C_2^{e_1} \equiv M^{e_1 \cdot e_2} \bmod n$
- $\frac{C_1^{e_2}}{C_2^{e_1}} \equiv M^{e_1 \cdot e_2 - e_2 \cdot e_1} \bmod n$
- $\frac{C_1^{e_2}}{C_2^{e_1}} \equiv M^0 \bmod n$
- $\frac{C_1^{e_2}}{C_2^{e_1}} \equiv 1 \bmod n$
- 이후, 공통근을 찾음

RSA (20/26)

- 공격

- 타이밍 공격 (1/2)

- 암호화 시스템에서 특정 연산을 수행할 때 걸리는 시간을 변화를 관찰하고 분석하며, 시스템의 비밀정보를 추출하거나 암호화된 데이터를 해독하는 공격
 - e.g., 공격자 Eve가 암호문 C_1 부터 C_m 까지 가로챘다고 가정하자
 - Eve는 Bob이 각 암호문을 복호화하는 과정을 지켜보면서 해독 시간을 기록 (T_1 부터 T_m)
 - Eve는 하드웨어에서 곱셈을 하는 데 걸리는 시간을 알고있고 이를 기록 (t_1 부터 t_m)
 - 곱셈연산은 $Result = Result \times C_i \bmod n$ 을 수행하는 데 걸리는 시간

RSA (21/26)

- 공격

- 타이밍 공격 (2/2)

- 코드 설명

- 초기화 단계

- 비트 추정 단계

- 비밀 키 비트 크기 k 만큼 반복문을 돌림
 - $[D_1 \cdots D_m] \leftarrow [T_1 \cdots T_m] - [t_1 \cdots t_m]$
 - 새로 계산된 시간에서 곱셈 시간을 빼서 d_j 에 예상되는 시간 차이를 구함
 - $\text{var} \leftarrow \text{variance}([D_1 \cdots D_m]) - ([T_1 \cdots T_m])$
 - 시간 차이의 분산을 계산
 - $\text{if}(\text{var} > 0) d_j \leftarrow 1$
 - 분산 값이 양수이면, d_j 를 1로 설정
 - $[T_1 \cdots T_m] \leftarrow [T_1 \cdots T_m] - d_j \times [t_1 \cdots t_m]$
 - d_j 에 해당하는 시간 차이 조정

```
RSA_Timing_Attack( $[T_1 \cdots T_m]$ ){  
  
     $d_0 \leftarrow 1$   
    Calculate( $t_1 \cdots t_m$ )  
     $[T_1 \cdots T_m] \leftarrow [T_1 \cdots T_m] - [t_1 \cdots t_m]$   
    for(j from 1 to k-1)  
    {  
        Recalculate( $T_1 \cdots T_m$ )  
         $[D_1 \cdots D_m] \leftarrow [T_1 \cdots T_m] - [t_1 \cdots t_m]$   
         $\text{var} \leftarrow \text{variance}([D_1 \cdots D_m]) - ([T_1 \cdots T_m])$   
         $\text{if}(\text{var} > 0) d_j \leftarrow 1$   
         $[T_1 \cdots T_m] \leftarrow [T_1 \cdots T_m] - d_j \times [t_1 \cdots t_m]$   
    }  
}
```

RSA (22/26)

- 공격

- 타이밍 공격 방지

- 지수 계산을 할 때 무작위적으로 지연을 추가
 - 알고리즘의 실행 시간이 입력 데이터나 비밀 키의 비트에 따라 달라지도록 않게 함
 - e.g., 복호화 연산 중 특정 조건이 충족될 때, 1ms에서 5ms 사이의 무작위 지연을 추가
- 암호문을 복호화하기 전에 난수로 곱하는 블라인딩 기법 추가
 1. 비밀로 된 난수 r 을 1과 $n-1$ 사이에서 선택
 2. $C_1 = C^e \times r \bmod n$ 을 계산
 3. $P_1 = C_1^d \bmod n$ 을 계산
 4. $P = P_1 \times r^{-1} \bmod n$ 을 계산

RSA (23/26)

- 공격

- 공격 방지를 위한 권장사항

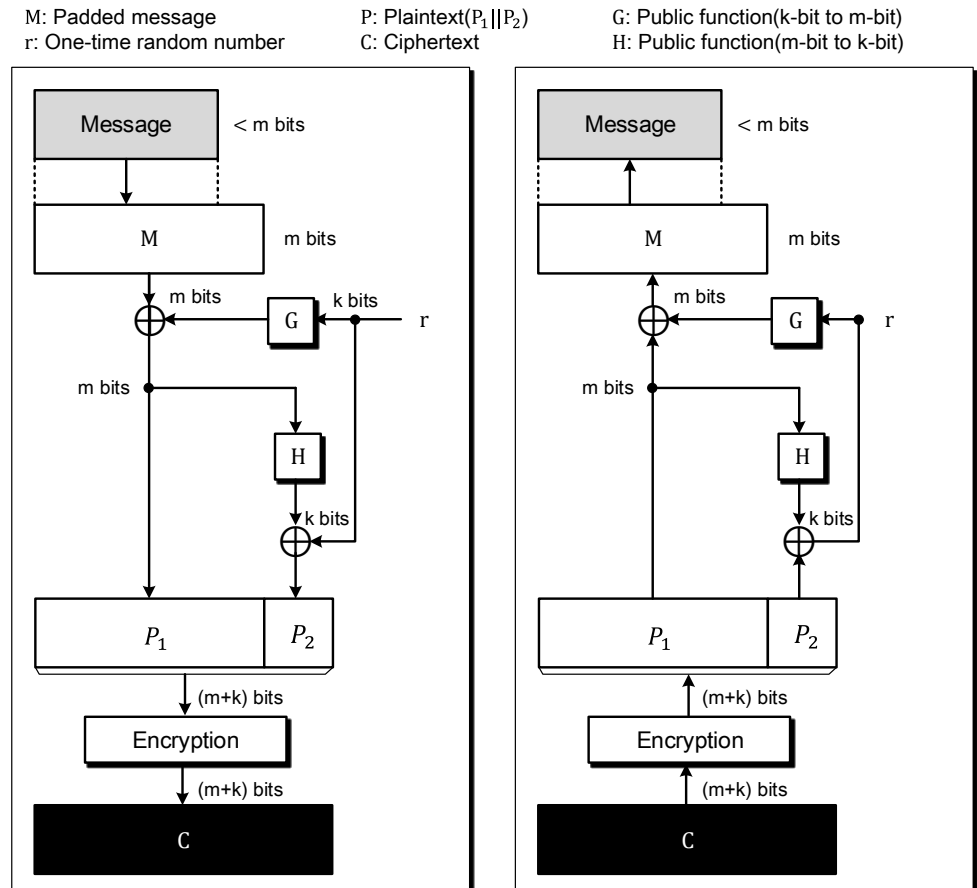
- n 의 비트 수는 적어도 2048비트. 즉, n 은 2^{2048} 근방에 있는 수로 선정
- p 와 q 는 서로 너무 가까이 있는 수가 되어서는 안 됨
- $p - 1$ 과 $q - 1$ 은 각각 적어도 하나의 큰 소인수를 가져야 함
- p/q 는 작은 분자나 분모를 갖는 유리수와 가까이 있어서는 안 됨
- 모듈로 n 을 공동으로 사용해서는 안 됨
- e 는 $2^{16} + 1$ 이거나 이 값에 가까운 정수이어야 함
- 만약 d 가 침해당했다면 n, p, q, e 와 d 모두 바꾸어야 함
- 메시지는 OAEP를 사용해서 패딩 되어야 함

RSA (24/26)

- 공격

- 최적 비대칭 암호 패딩(OAEP, Optimal Asymmetric Encryption Padding)

- 암호 시스템에서 사용하는 패딩 방식으로, 메시지를 암호화하기 전에 추가적인 정보를 넣는 방법



RSA (25/26)

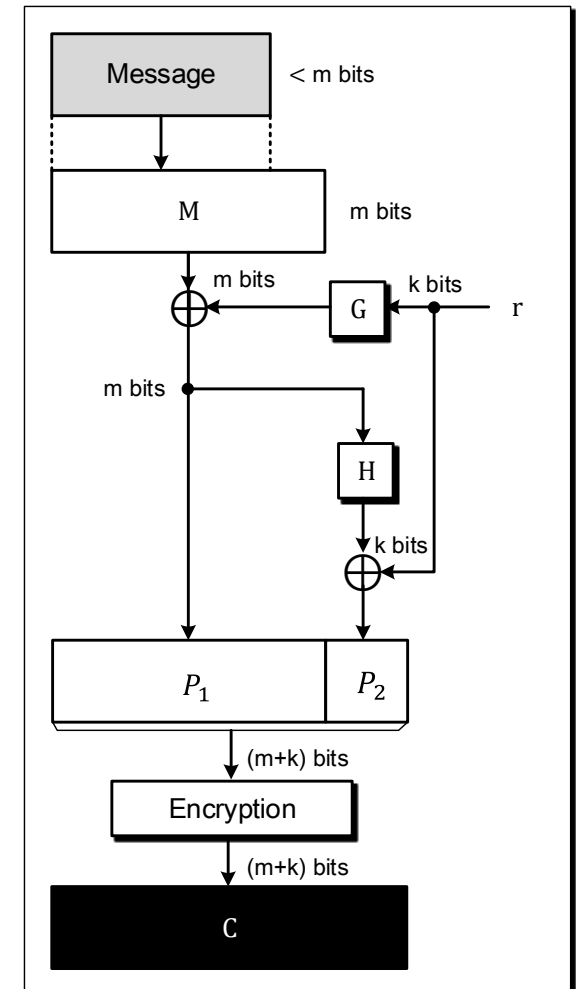
• 공격

• 최적 비대칭 암호 패딩(OAEP, Optimal Asymmetric Encryption Padding)

• 암호화 절차

- $P = P_1 || P_2$

1. m 비트 메시지가 되도록 패딩을 붙인 것을 M 이라고 부름
2. k 비트를 갖는 난수 r 을 선택함
3. r 비트 정수 입력으로 m 비트 정수를 출력하는 공개된 일방향 함수 $G(r)$ 사용
 - 평문의 첫 번째 부분인 $P_1 = M \oplus G(r)$ 생성
4. m 비트를 입력하여 k 비트를 내보내는 공개된 함수 $H(P_1)$ 를 사용
 - 평문의 두 번째 부분인 $P_2 = H(P_1) \oplus r$ 생성
5. Alice는 $C = P^e = (P_1 || P_2)^e$ 을 전송



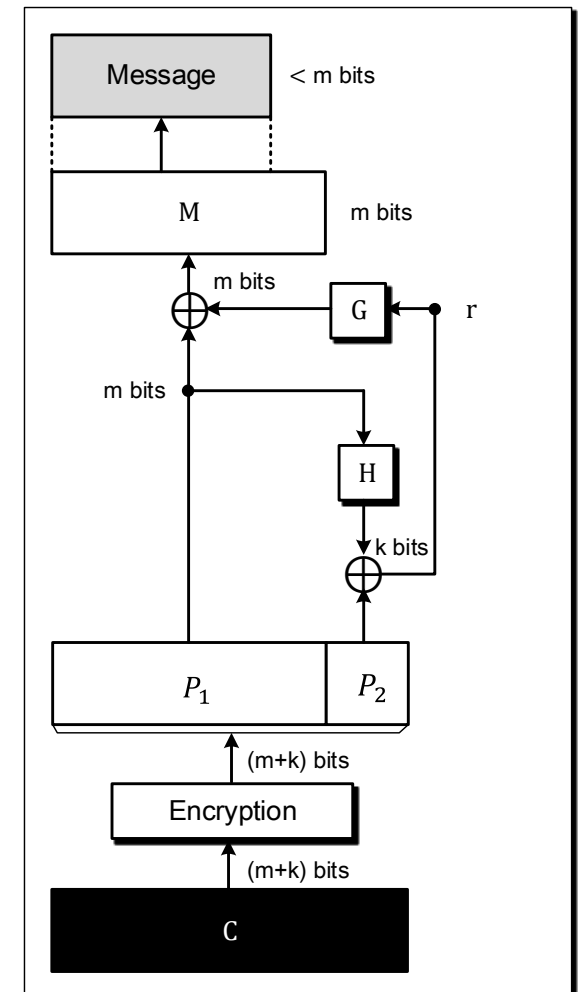
RSA (26/26)

- 공격

- 최적 비대칭 암호 패딩(OAEP, Optimal Asymmetric Encryption Padding)

- 복호화 절차

1. $P = C^d = (P_1 || P_2)$
2. $H(P_1) \oplus P_2 = H(P_1) \oplus H(P_1) \oplus r = r$ 을 이용하여 r 을 재생성
3. 패딩된 메시지 값을 재생성하기 위해 $G(r) \oplus P = G(r) \oplus G(r) \oplus M = M$ 을 이용
4. M 으로 부터 패딩을 제거하고 원래의 메시지를 구함



목 차

- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

Rabin (1/7)

- 개요
 - 정의
 - RSA 변형으로 평문의 제곱을 암호문으로 생성하는 공개 키 암호 시스템
- 키 구성
 - 공개 키 (n) , 개인 키 (p, q)
 - RSA의 $e = 2, d = \frac{1}{2}$ 로 고정임

Rabin (2/7)

- 동작

- 키 생성

- $k \in \mathbb{Z}, 4k + 3$ 인 서로 다른 두 소수 p 와 q 를 선택
 - $4k + 1$ 을 가질 수 있지만, 이 경우엔 복호화 과정이 어려워짐
- $n = p \times q$

Rabin_key_Generation{

Choose two large primes p and q in the form $4k+3$ and $p \neq q$.

$n \leftarrow p \times q$

Public_key $\leftarrow n$

Private_key $\leftarrow (q, p)$

return Public_key and Private_key

}

Rabin (3/7)

- 동작

- 암호화

- $C = P^2 \bmod n$

- 복호화

1. $P_p = \sqrt{C} \bmod p, P_q = \sqrt{C} \bmod q$

- $p \cdot y_p + q \cdot y_q = 1$ 을 만족하는 y_p 와 y_q 구하기

- $a_1 \equiv C^{\frac{p+1}{4}} \bmod p, \quad a_2 \equiv -C^{\frac{p+1}{4}} \bmod p$

- $b_1 \equiv C^{\frac{q+1}{4}} \bmod q, \quad b_2 \equiv -C^{\frac{q+1}{4}} \bmod q$

2. CRT를 이용

- $P_1 = \text{CRT}(a_1, b_1, p, q), P_2 = \text{CRT}(a_1, b_2, p, q),$
 $P_3 = \text{CRT}(a_2, b_1, p, q), P_4 = \text{CRT}(a_2, b_2, p, q)$

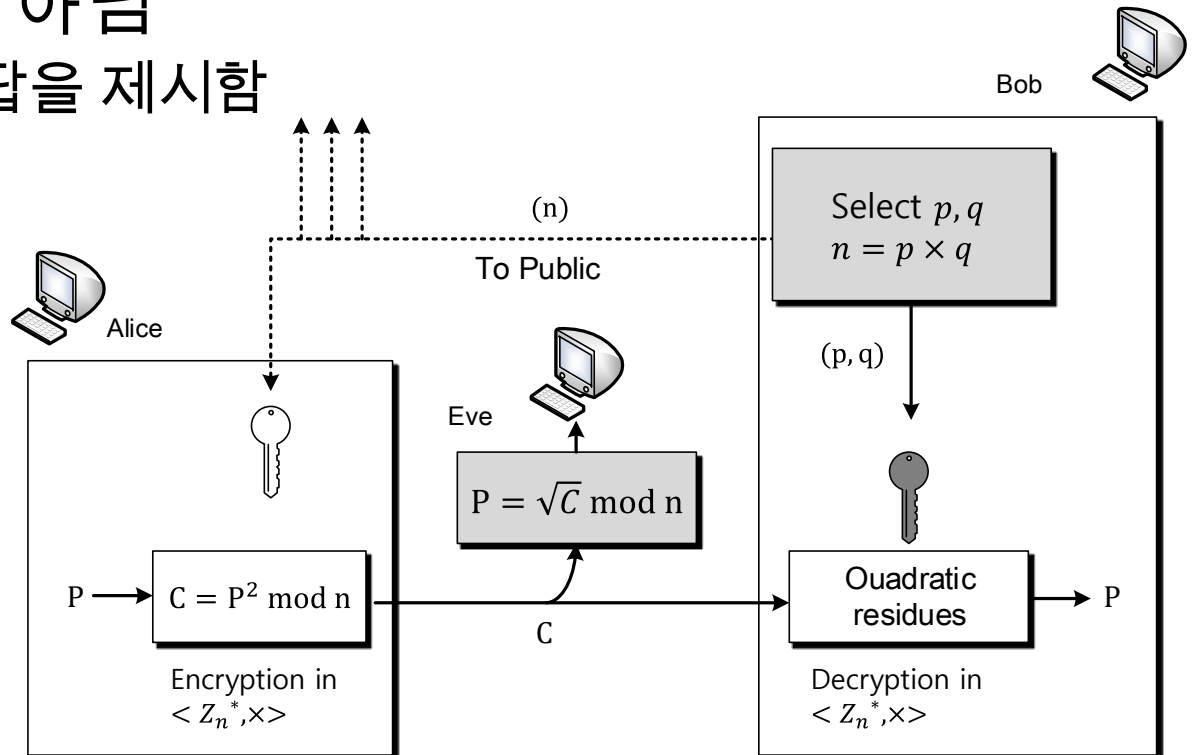
3. $\{P_1, P_2, P_3, P_4\}$ 중 하나가 평문

Rabin (4/7)

- 동작

- 복호화

- 2차 합동에 근을 가지고 있음
 - P 는 Z_n^* 의 값이기 때문에 C 는 Z_n^* 안에 근을 반드시 가지고 있음
- 결정적 알고리즘이 아님
 - 복호화에선 4개의 답을 제시함



Rabin (5/7)

- 동작

- 예제 10.9 (1/2)

Rabin 알고리즘을 설명하기 위해 간단한 예를 들어보기로 하자.

- Bob은 $p = 23, q = 7$ 을 선택함 (두 수는 모두 모듈로 4로 3과 합동임)
공개 키(161), 개인 키(23, 7)
- Alice가 Bob에게 평문 $P = 24$ 를 보냄(161과 24는 서로소로 $24 \in Z_{161}^*$)
$$C = 24^2 = 93 \text{ mod } 161$$
- Alice는 93을 Bob에게 보냄
- Bob은 93을 수신하고 CRT를 계산함
 - $a_1 = +\left(93^{\frac{23+1}{4}}\right) \text{ mod } 23 = 1 \text{ mod } 23$
 - $a_2 = -\left(93^{\frac{23+1}{4}}\right) \text{ mod } 23 = 22 \text{ mod } 23$
 - $b_1 = +\left(93^{\frac{7+1}{4}}\right) \text{ mod } 7 = 4 \text{ mod } 7$
 - $b_2 = -\left(93^{\frac{7+1}{4}}\right) \text{ mod } 7 = 3 \text{ mod } 7$

Rabin (6/7)

- 동작

- 예제 10.9 (2/2)

Rabin 알고리즘을 설명하기 위해 간단한 예를 들어보기로 하자.

- Bob은 4개의 가능한 답 $\{P_1, P_2, P_3, P_4\}$ 을 계산
 - $P_1 = \text{CRT}(a_1, b_1, p, q)$, $P_2 = \text{CRT}(a_1, b_2, p, q)$, $P_3 = \text{CRT}(a_2, b_1, p, q)$, $P_4 = \text{CRT}(a_2, b_2, p, q)$
 - $\{P_1, P_2, P_3, P_4\} = \{116, 24, 137, 45\}$
- 25%의 확률로 Alice가 보낸 평문 24를 고를 수 있음
 - 패딩의 유무, 중복 암호문의 유무 등의 메타데이터를 통해 선택함
- 4개의 답은 제공하여 모듈로 값을 구해보면 전부 암호문 93이 나옴
 - $116^2 = 93 \pmod{161}$
 - $24^2 = 93 \pmod{161}$
 - $137^2 = 93 \pmod{161}$
 - $45^2 = 93 \pmod{161}$

Rabin (7/7)

- RSA와 비교

- 효율성

- 암호화에서 Rabin은 제곱 모듈로를 계산하기 때문에 3차 이상을 계산하는 RSA에 비해 효율적임
- 암호화와 복호화가 상대적으로 간단하지만 복호화 과정에서 제곱근 문제로 인해 다소 복잡할 수도 있음

- 수학적 기반

- 두 암호 시스템 모두 소인수분해의 어려움에 기반해 보안성을 제공함
- 두 암호 시스템 모두 큰 키 길이를 요구함

	RSA	Rabin
키 생성	공개 키: (n, e) , 개인 키: d	공개 키: n , 비밀 키: (p, q)
암호화	$C = P^e \bmod n$	$C = P^2 \bmod n$
복호화	$P = C^d \bmod n$	$P = \sqrt{C} \bmod p \text{ or } \sqrt{C} \bmod q$

목 차

- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

ElGamal (1/7)

- 개요

- 정의

- 이산 로그 문제에 기반하고 난수를 사용하여 매번 다른 암호문을 생성하는 공개 키 암호 시스템

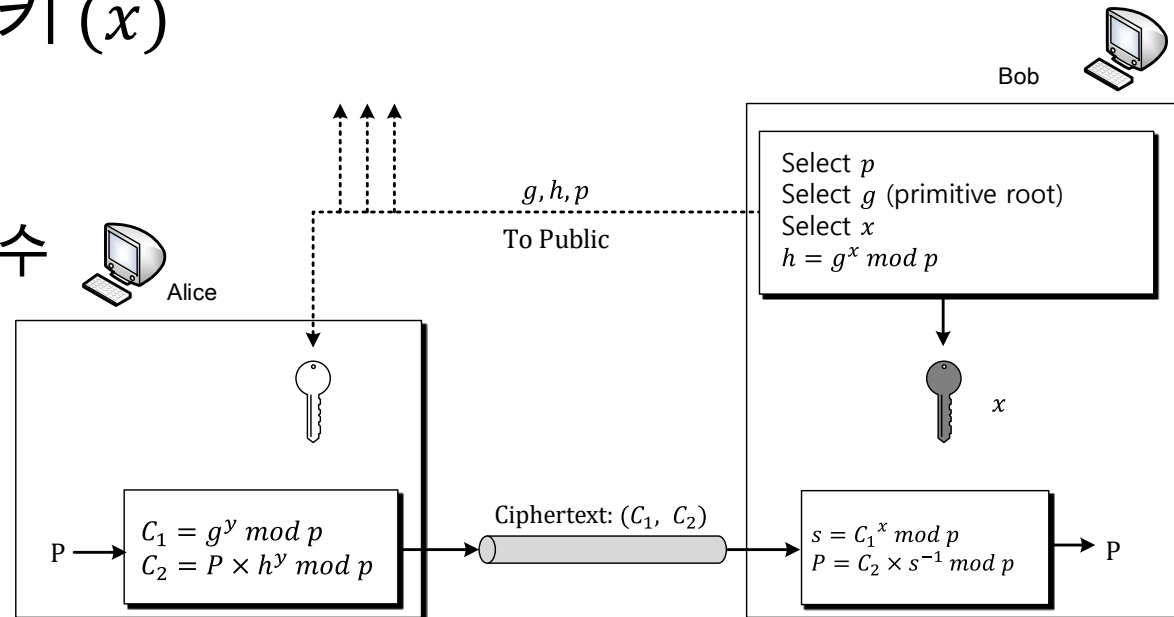
- 키 구성

- 공개 키 (p, g, h) , 개인 키 (x)

- g : p 의 원시근
- h : $g^x \bmod p$
- x : $1 \leq x \leq p - 2$ 의 정수

이산로그 문제

소수 p 와 p 의 원시근 정수 g , 그리고 g 의 거듭제곱 $y \equiv g^x \bmod p$ 에서 y 가 주어졌을 때, $x = \log_g y$ 를 찾는 문제



ElGamal (2/7)

- 동작

- 키 생성

1. 큰 소수 p 를 선택
2. p 의 원시근 g 를 선택
3. 1이상 $p - 2$ 이하의 정수 x 선택
 - $p - 1$ 일 경우, 보안적 측면과 수학적 측면에서 결함이 발생함
 - 보안적 측면: p 가 $2q + 1$ 과 같은 특정 구조일 경우 이산 로그 문제를 푸는 것이 쉬워짐
 - 수학적 측면: $p - 1$ 은 p 의 약수와 관련된 성질을 가져, 모듈로 연산에 영향을 미침
4. $h = g^x \bmod p$ 계산

ElGamal (3/7)

- 동작

- 암호화

1. 1 이상 $p - 2$ 이하 정수 y 선택
2. $C_1 = g^y \bmod p$ 계산
3. $C_2 = P \times h^y \bmod p$
4. 암호문 (C_1, C_2) 생성

```
ElGamal_Encryption( $g, h, p, P$ ){                                     //P는 평문

    Select a random integer  $y$  in the group  $G = \langle Z_p^*, \times \rangle$ 
     $C_1 \leftarrow g^y \bmod p$ 
     $C_2 \leftarrow P \times h^y \bmod p$ 
    return  $C_1$  and  $C_2$ 

}
```

ElGamal (4/7)

- 동작

- 복호화

1. $s = C_1^x \bmod p$ 계산
2. s 의 모듈러 역원 계산
 - $s \times s^{-1} \bmod p$
3. $P = C_2 \times s^{-1} \bmod p$ 로 복호화

```
ElGamal_Decryption(s, p, C1, C2) {
```

```
    P ← C2 × s-1 mod p  
    return P
```

```
}
```

- 증명

- $s = C_1^x \bmod p = g^{xy} \bmod p$
- $C_2 = P \times h^y \bmod p = P \times g^{xy} \bmod p$
- $P = P \times g^{xy} \times (g^{xy})^{-1} \bmod p$

ElGamal (5/7)

- 동작

- 예제 10.10 (1/2)

Bob은 $p = 11, g = 2 (2 \in Z_{11}^*)$ 를 선택하고 $x = 3, y = 4$ 를 선택했다. 평문 $P = 7$ 에 대한 암호문을 계산하라.

- 암호화

- $C_1 = g^y \bmod p \quad C_2 = P \times h^y \bmod p \quad h = g^x \bmod p$
- 공개 키 $(p, g, h) = (11, 2, 2^3)$, 개인 키 (3)
- $C_1 = 2^4 \bmod 11 = 16 \bmod 11 = 5 \bmod 11$
- $C_2 = 7 \times 8^4 \bmod 11 = 7 \times 4096 \bmod 11 = 6 \bmod 11$
- \therefore 암호문 $(5, 6)$

ElGamal (6/7)

- 동작

- 예제 10.10 (2/2)

Bob은 $p = 11, g = 2 (2 \in Z_{11}^*)$ 를 선택하고 $x = 3, y = 4$ 를 선택했다. 평문 $P = 7$ 에 대한 암호문을 계산하라.

- 복호화

- $P = C_2 \times s^{-1} \mod p \quad s = C_1^x \mod p$
- $P = C_2 \times s^{-1} \mod 11 = 6 \times (5^3)^{-1} \mod 11 = 6 \times 3 \mod 11 = 7 \mod 11$

- \therefore 평문 7

- 페르마 소정리에 의해 아래와 같이 표현 가능

- $P = C_2 \times (C_1^x)^{-1} \mod p \quad P = C_2 \times C_1^{p-1-x} \mod p$
- $C_2 \times C_1^{p-1-x} \mod p = 6 \times 5^{11-1-3} = 6 \times 5^7 \mod 11 = 7$

ElGamal (7/7)

- 공격

- 작은 모듈로 공격

- 만약 p 가 작아 $p - 1$ 이 작은 소수들의 곱으로 표현될 수 있는 경우, 이산로그 문제가 쉽게 풀림
 - p 는 적어도 300자리 이상의 10진수로 설정해야 함
 - e.g., $p - 1 = q_1 q_2 \cdots q_n$, g 가 작은 소수 부분군의 원시근이라면 g 의 거듭제곱이 작은 소수 부분 군 내의 모든 요소를 생성할 수 있음

- 알려진 평문 공격

- 공격자가 암호문과 대응하는 평문의 쌍을 알고 있는 경우, y 에 임의의 정수 k 를 넣어 해독함
 - y 는 암호화할 때마다 매번 다르게 사용해야 함
 - $C_1 = g^k \bmod p$ $C_2 = P \times h^k \bmod p$

목 차

- 비대칭 키
- 배낭 암호
- RSA
- Rabin
- ElGamal
- ECC

ECC (1/23)

- 개요

- 정의

- 타원 곡선 암호화(ECC, Elliptic Curve Cryptography)라고 하며 타원 곡선의 수학적 구조를 이용해 보안성을 높이고 키 크기를 줄이는 공개 키 암호 시스템

- 타원 곡선 방정식

- 일반적 타원 곡선식: $y^2 + b_1xy + b_2y = x^3 + a_1x^2 + a_2x + a_3$
- 정칙 타원 곡선: $y^2 = x^3 + ax + b$
 - $4a^3 + 27b^2 \neq 0$ 을 만족해야 함
 - 즉, 곡선 미분이 0이 되는 점이 없도록 하는 곡선을 사용함

ECC (2/23)

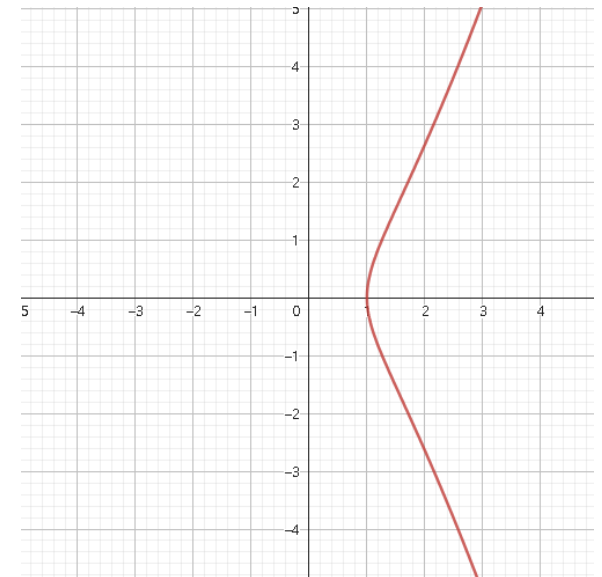
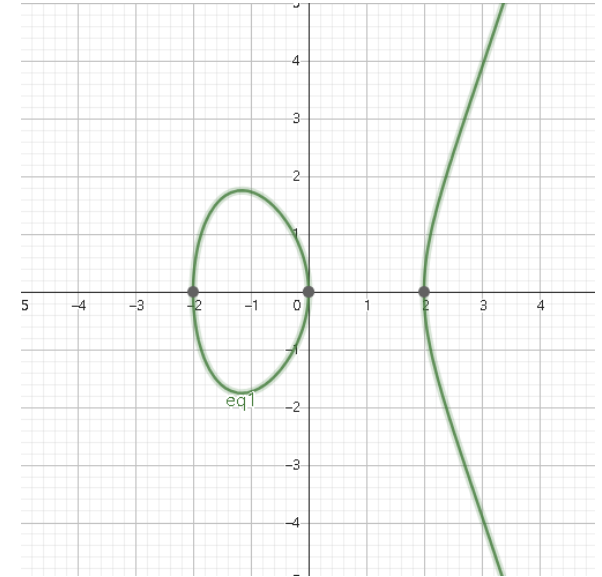
• 공격

• 예제 10.13

두 타원 곡선 방정식의 특이점을 살펴보자

- $y^2 = x^3 - 4x$
- $a = -4, b = 0$
- $4a^3 + 27b^2 = 4 \times (-4)^3 = -256 \neq 0$
- 해당 방정식은 특이점이 없는 정칙 타원 곡선임

- $y^2 = x^3 - 1$
- $a = 0, b = -1$
- $4a^3 + 27b^2 = 27 \times (-1)^3 = -27 \neq 0$
- 해당 방정식은 특이점이 없는 정칙 타원 곡선임



ECC (3/23)

- 아벨 군

- 타원 곡선의 점들로 아벨 군을 정의할 수 있음
 $G = \langle E(K), + \rangle$

- 집합($E(K)$)

- 곡선 상의 한 점을 좌표를 나타내는 쌍 $P = (x_1, y_1)$
- $E(K) = \{(x_1, y_1) \in K^2 \mid y^2 = x^3 + ax + b\} \cup \{O\}$
 - O 은 무한원점을 의미

- 연산($+$)

- 실수 상 정의하는 덧셈과는 다르며, 곡선 상의 두 점을 더했을 때 곡선 상의 다른 한 점을 찾는 규칙을 의미함
 - $R = P + Q$, 여기서 $P = (x_1, y_1)$, $Q = (x_2, y_2)$ 이고 $R = (x_3, y_3)$

ECC (4/23)

- 아벨 군

- 연산(+) (1/2)

- 다른 점들: $P + Q = -R$

- 두 점 P와 Q가 다르면 P와 Q를 지나는 직선을 그려 타원 곡선과 만나는 세 번째 점 R을 찾음

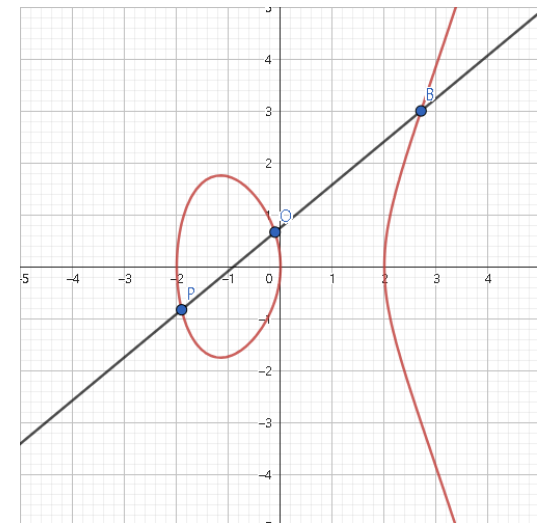
- R의 y좌표를 반전 시킨 점이 $P + Q$

- 기울기 구하기

- $x_3 = \lambda^2 - x_1 - x_2$

- $y_3 = \lambda(x_1 - x_3) - y_1$

- $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$



ECC (5/23)

• 아벨 군

• 연산(+) (2/2)

• 자기 자신과의 덧셈: $2P = -R$

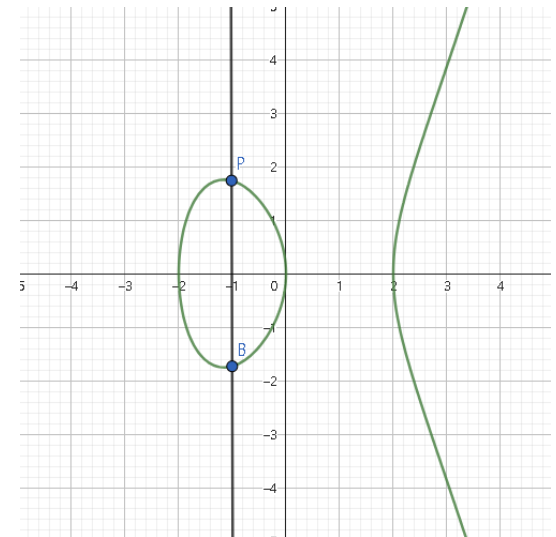
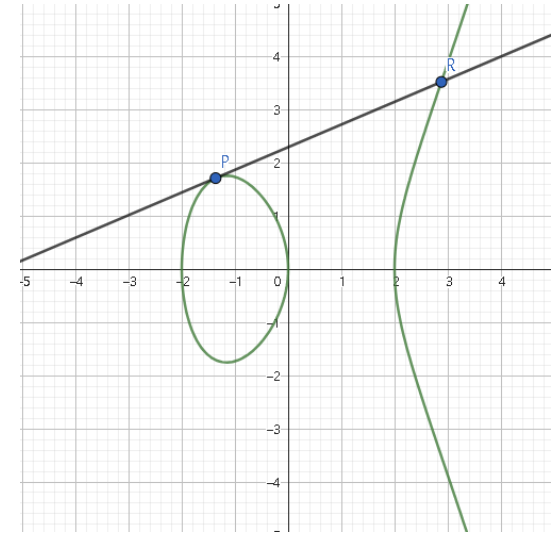
- 점 P에서 접선이 타원 곡선과 만나는 점 R를 찾음

• 기울기 구하기

- $x_3 = \lambda^2 - x_1 - x_2$
- $y_3 = \lambda(x_1 - x_3) - y_1$
- $\lambda = \frac{3x_1^2 + a}{2y_1}$

• 무한 원점: $P + O = P$

- 무한 원점 O를 덧셈의 항등원 역할을 함



ECC (6/23)

- 아벨 군

- 연산의 성질

- 닫힘

- 앞에서 정의한 덧셈 연산을 이용하면 두 점을 합친 것이 다시 곡선 상에 있다는 점을 증명할 수 있음

- 결합법칙

- $(P + Q) + R = P + (Q + R)$

- 교환 법칙

- $P + Q = Q + P$

- 항등원의 존재성

- 해당 덧셈 연산에 대한 항등원은 무한 영점 O 임

- 역원의 존재성

- 곡선상의 점에 대한 역원은 x 축에 대한 대칭 점임

- $P = (x_1, y_1)$ 과 $Q = (x_1, -y_1)$ 는 서로 역원 관계이며 $P + Q = O$

ECC (7/23)

- GF(p)상의 타원 곡선

- 앞선 아벨 군 같은 경우, 연산이 실수 위의 계산이었다면 점좌표상의 연산은 $p > 3$ 에 대한 체 GF(p)상의 연산임
 - 타원 곡선 군 $E_p(a, b)$ 로 표현

- 집합

- $E(\text{GF}(p)) = \{(x, y) \in \text{GF}(p)^2 \mid y^2 = x^3 + ax + b \pmod{p}\} \cup \{O\}$
 - (a, b) 는 GF(p)의 원소
 - O 은 무한원점 또는 점연산의 항등원을 의미함

```
ellipticCurve_points( $p, a, b$ ){                                     //  $p$ 는 암호문
     $x \leftarrow 0$ 
    while( $x < p$ ){
         $w \leftarrow (x^3 + ax + b) \pmod{p}$     //  $w$  is  $y^2$ 
        if( $w$  is perfect square in  $Z_p$ ) output( $x, \sqrt{w}$ ) ( $x, -\sqrt{w}$ )
         $x \leftarrow x + 1$ 
    }
}
```

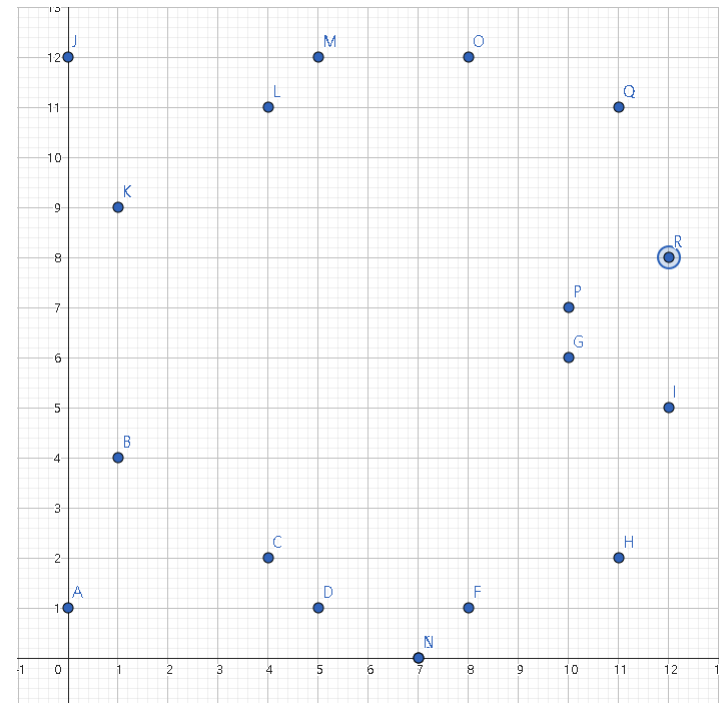
ECC (8/23)

- $GF(p)$ 상의 타원 곡선
- 예제 10.14 (1/2)

타원 곡선 $E_{13}(1,1)$ 을 정의하시오.

- 방정식은 $y^2 = x^3 + x + 1$ 이고 계산은 모듈로 13으로 연산함
- 곡선 상의 점들은 그림과 같이 표현 가능함

(0, 1)	(0, 12)
(1, 4)	(1, 9)
(4, 2)	(4, 11)
(5, 1)	(5, 12)
(7, 0)	(7, 0)
(8, 1)	(8, 12)
(10, 6)	(10, 7)
(11, 2)	(11, 11)
(12, 5)	(12, 8)



ECC (9/23)

- $GF(p)$ 상의 타원 곡선
- 예제 10.14 (2/2)

타원 곡선 $E_{13}(1,1)$ 을 정의하시오.

- 특징
 - 어떤 y^2 값은 모듈로 13 상에서 제곱근을 갖지 못함
 - y 가 0부터 12까지의 값일 때 y^2 를 계산하면 $GF(13)$ 에서 가능한 제곱 값은 $\{0, 1, 3, 4, 9, 10, 12\}$ 임
 - $x^3 + x + 1 \pmod{13}$ 값을 계산하여 제곱 값이 가능한지 확인함
 - 제곱근이 없는 값은 $\{2, 5, 7, 8, 9\}$ 임
 - 모든 점이 위에 값이 곡선 위에 있는 것은 아님을 의미함
 - 곡선에 정의된 모든 점을 역원을 가짐
 - e.g., $(7, 0)$ 은 자기 자신이 역원임
 - 역원들은 그래프에서 같은 수직선쌍에 나타남
 - e.g., 4가 y 라면 9가 $-y$

ECC (10/23)

- $GF(p)$ 상의 타원 곡선

- 연산

- 두 점을 더하기

- 덧셈과 곱셈에 대한 역원을 사용함

- e.g., $P + Q = -R$

- 1. 기울기를 계산함

- $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$

- 2. 기울기 역수 계산함

- $\lambda^{-1} \equiv \lambda^{p-2} \pmod{p}$

- 3. 새로운 좌표 계산함

- $x_3 = \lambda^2 - x_1 - x_2$

- $y_3 = \lambda(x_1 - x_3) - y_1$

ECC (11/23)

- GF(p)상의 타원 곡선

- 예제 10.15

$P = (4, 2), Q = (10, 6)$ 일 때, $R = P + Q$ 를 구하라 (예제 10.14 사용)

- $P = (x_1, y_1), Q = (x_2, y_2), R = (x_3, y_3)$
- $\lambda = \frac{6-2}{10-4} \pmod{13} = \frac{4}{6}$
- $\frac{4}{6} \equiv 4 \cdot 11 \equiv 44 \equiv 5 \pmod{13}$
- $\therefore \lambda = 5$
- $x_3 = \lambda^2 - x_1 - x_2 = 5^2 - 4 - 10 = 12 - 4 - 10 = 11 \pmod{13}$
- $y_3 = \lambda(x_1 - x_3) - y_1 = 5(4 - 11) - 2 = -37 \equiv -37 + 52 = 15 \equiv 2 \pmod{13}$
- $R = (x_3, y_3) = (11, 2)$

ECC (12/23)

- GF(p)상의 타원 곡선

- 연산

- 한 점에 상수를 곱하기

- 어떤 수에 k 번 곱한다는 것은 그 수를 k 번 더하는 산술연산과 마찬가지로 자기자신에게 k 번 더한다는 의미임

- e.g., $E_{13}(1,1)$ 에서 점 $P = (8,1)$ 에 3을 곱함

- $P + P = 2P = (1,11)$ 계산

- $\lambda = \frac{3x_1^2 + a}{2y_1} = \frac{3 \cdot 8^2 + 1}{2 \cdot 1} = \frac{193}{2} \equiv \frac{9}{2} \pmod{13}$

- $\lambda = 9 \cdot 7 = 63 \equiv 11 \pmod{13}$ ($\because 2$ 의 역원 7)

- $x_3 = \lambda^2 - 2x_1 = 11^2 - 2 \cdot 8 = 121 - 16 = 105 \equiv 1 \pmod{13}$

- $y_3 = \lambda(x_1 - x_3) - y_1 = 11(8 - 1) - 1 = 11 \cdot 7 - 1 = 76 \equiv 11 \pmod{13}$

- $2P + P$ 계산

- $\lambda = \frac{y_2 - y_1}{x_2 - x_1} = \frac{10}{-7} \pmod{13}$

- $\lambda = 10 \cdot 11 = 110 \equiv 6 \pmod{13}$ ($\because -7 \equiv 6, 6$ 의 역원 11)

- 이하 동일

- $3P = (1,2)$

ECC (13/23)

- $\text{GF}(2^n)$ 상의 타원 곡선
 - $\text{GF}(p)$ 와 다르게 모듈로 연산이 아닌 다항식으로 정의됨
 - 일반적으로 방정식 $y^2 + xy = x^3 + ax^2 + b$ 로 표현됨
- 집합
 - $E(\text{GF}(2^n)) = \{(x, y) \in \text{GF}(2^n) \times \text{GF}(2^n) \mid y^2 + xy = x^3 + ax^2 + b\} \cup \{O\}$
 - a 와 b 는 $\text{GF}(2^n)$ 의 원소임
 - O 은 무한원점 또는 영점, 덧셈의 항등원을 의미함
 - y^2 : 이진 체에서의 제곱 연산은 다항식의 비트 연산으로 처리됨
 - $x^3 + ax^2 + b$: 이항식의 계산은 비트연산과 다항식 연산을 통해 수행됨

ECC (14/23)

- GF(2^n)상의 타원 곡선

- 연산

- 역원

- $P = (x, y)$ 라면 $-P = (x, x + y)$ 임

- 곡선 상의 점을 찾기 위해선 다항식 생성원을 사용함

- e.g., 기약 다항식 $f(x) = x^3 + x + 1$ 을 사용하여 원소가 g^i 인 GF(2^8)

- $\{0, g, g^2, g^3, g^4, g^5, g^6\}$ 로 표현

- $g^3 + g + 1 = 0$ 으로 사용되거나 $g^3 = g + 1$ 로 사용할 수 있음

- e.g., $g^5 = g^3 \cdot g^2 = (g + 1) \cdot g^2 = g^3 + g^2 = g^2 + g + 1$

0	000	$g^3 = g + 1$	011
1	001	$g^4 = g^2 + g$	110
g	010	$g^5 = g^2 + g + 1$	111
g^2	100	$g^6 = g^2 + 1$	101

(0,1)	(0,1)
$(g^2, 1)$	(g^2, g^6)
(g^3, g^2)	(g^3, g^5)
$(g^5, 1)$	(g^5, g^4)
(g^6, g)	(g^6, g^5)

ECC (15/23)

- $GF(2^n)$ 상의 타원 곡선

- 연산

- 두 점을 더하기

- $GF(p)$ 의 규칙과 다르게 적용됨

- 1. $P = (x_1, y_1), Q = (x_2, y_2)$ 이고 $\pm P \neq Q$ 인 경우, $P + Q = R = (x_3, y_3)$

- $\lambda = \frac{y_1 + y_2}{x_1 + x_2}$

- $x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$

- $y_3 = \lambda(x_1 + x_3) + y_1$

- e.g., $P = (0, 1)$ 이고 $Q = (g^2, 1)$ 인 경우, $\lambda = 0$ (\because XOR), $R = (g^5, g^4)$

- 2. $P = (x_1, y_1), Q = (x_2, y_2)$ 이고 $P = Q$ 인 경우, $P + P = R$

- $\lambda = x_1 + \frac{y_1}{x_1}$

- $x_3 = \lambda^2 + \lambda + a$

- $y_3 = x_1^2 + (\lambda + 1)x_3$

- e.g., $P = (g^2, 1)$ 인 경우, $\lambda = g^2 + \frac{1}{g^2} = g^2 + g^5 = g + 1$ ($\because x^3 = x + 1$)

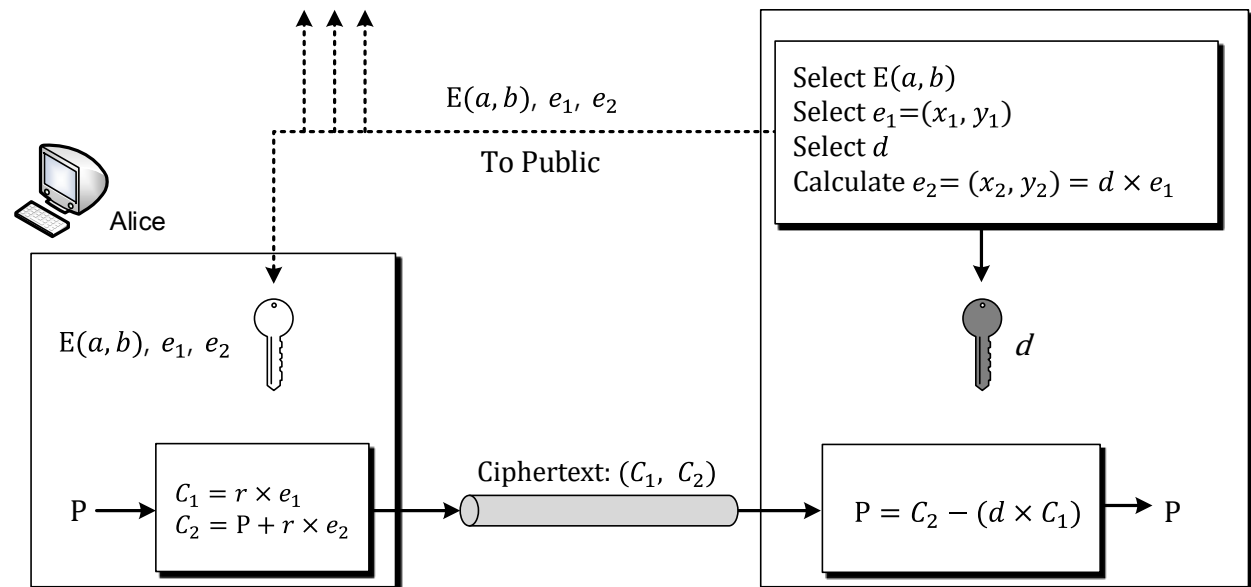
ECC (16/23)

• ElGamal에 타원 곡선 적용

• 키 생성

- $GF(2^n)$ 나 $GF(p)$ 상의 타원 곡선 하나와 $E(a, b)$ 선택함
- 곡선상의 한점 $e_1 = (x_1, y_1)$ 와 d 선택함
- $e_2 = d \cdot (x_1, y_1)$ 를 계산함 (일반 산술 연산이 아닌 GF상의 연산)
- $E(a, b)$, e_1 , e_2 을 공개 키로, d 는 개인 키로 선언함

Bob 



ECC (17/23)

- ElGamal에 타원 곡선 적용

- 암호화

- 평문에 해당하는 곡선 상의 한 점 P 를 선택함
- Alice가 암호문으로 사용할 두 점을 계산함
 - $C_1 = r \times e_1, C_2 = P + r \times e_2$
 - (C_1, C_2) 이 곡선 상 점일 수 있도록 Alice는 알고리즘을 사용해서 기호들과의 일대일 대응을 찾아야 함

- 복호화

- Bob은 수신받은 암호문을 해독함
 - $P = C_2 - (d \times C_1)$ (여기서 마이너스 기호는 역원을 더한다는 의미임)

- 증명

- $$P + r \times e_2 - (d \times r \times e_1) = P + (r \times d \times e_1) - (d \times r \times e_1)$$
$$= P + O = P$$

ECC (18/23)

- ElGamal에 타원 곡선 적용

- 예제 10.19 (1/2)

$GF(p)$ 상의 타원 곡선을 사용한 암호에 대한 예제를 살펴보자.

- 키 생성

- Bob이 $E_{67}(2,3)$ 을 선택함
 - $y^2 = x^3 + 2x + 3 \pmod{67}$
- Bob이 $e_1 = (2, 22)$, $d = 4$ 를 선택함
- Bob이 $e_2 = (13, 45)$ 를 선택함. 이때 $e_2 = d \times e_1$ 임
- \therefore 공개 키 (E, e_1, e_2)

- 암호화

- Alice는 평문 $P = (24, 26)$ 을 보내려고 하고, $r = 2$ 를 선택함
- $C_1 = r \times e_1 = (35, 1)$ $C_2 = P + r \times e_2 = (21, 44)$

ECC (19/23)

- ElGamal에 타원 곡선 적용
 - 예제 10.19 (2/2)

$GF(p)$ 상의 타원 곡선을 사용한 암호에 대한 예제를 살펴보자.

- 복호화
 - Bob이 전달받은 C_1 과 C_2 를 구함
 - $P = C_2 - (d \times C_1)$
 - Bob이 $(d \times C_1) = (23, 25)$ 의 역원인 $(23, 42)$ 를 구함
 - $\therefore (35, 1) + (23, 42) = P$

ECC (20/23)

- ElGamal에 타원 곡선 적용

- 기존 ElGamal vs 타원 곡선을 적용한 ElGamal

- 같은 키 길이에 비해 시뮬레이션이 더 높은 보안성을 가짐
- 타원 곡선 연산은 모듈러 곱셈보다 연산 비용이 낮아 전력 소모가 적은 편임

	기존 ElGamal	시뮬레이션
사용되는 군	$G = \langle Z_p^*, \times \rangle$	$E_p(a, b)$
지수 계산	$h = g^x \bmod p$	점 P
곱셈과 덧셈	$s = g^{xy} \bmod p$	$P + Q$
역원	$s^{-1} \bmod p$	$-P$
연산의 효율성	지수 계산	점 곱셈
연산의 대체	곱셈	덧셈
개인 키	정수	정수

ECC (21/23)

- 안전성

- 타원 곡선 이산 로그 문제

- 곡선 E와 타원 곡선 상의 두 점 P와 Q에 대하여, 정수 k를 찾아서 $Q = kP$ 를 만족시키는 k를 구하는 문제
 - P는 타원 곡선 상의 생성점이며, k는 이산 로그 문제의 해
- r 를 알아내기 위해선 r 를 알 수 있는 대표적인 알고리즘 Pollard's rho 알고리즘이 있음
 - $GF(2^n)$ 의 n이나 $GF(p)$ 의 p가 크다면 효과가 없음
- 전통적인 이산로그 문제보다 해결이 어려움

- 키 길이

- $GF(2^n)$ 상의 ECC의 160비트는 RSA의 1024비트 크기의 모듈로가 제공하는 안전성과 동일함

ECC(22/23)

• 비대칭 키 암호 비교

• 안전성

- 다른 암호 시스템보다 키 길이 대비 안전성이 강함

Date	Security Strength	Symmetric Algorithms	Factoring Modulus	Discrete Logarithm		Elliptic Curve
				Key	Group	
Legacy	80	2TDEA	1024	160	1024	160
2019-2030	112	3TDEA AES-128	2048	224	2048	224
2019-2030 &beyond	128	AES-128	3072	256	3072	256
2019-2030 &beyond	192	AES-192	7680	384	7680	384
2019-2030 &beyond	256	AES-256	15360	512	15360	512

<출처>Elaine Barker, "Recommendation for Key Management", Special Publication 800-57 Part 1 Rev.5, NIST, 05/2020.

ECC(23/23)

• 비대칭 키 암호 비교

• 효율성

- 다양한 암호 방식 설계가 용이해 SW 및 HW로 구현하기 쉬움
 - RSA는 인프라가 다소 구현된 환경에 적용된다면 ECC는 소형 모바일 환경에도 적용가능함

	ECC	ElGamal	Rabin	RSA
키 생성	타원 곡선의 원소를 기반으로 생성	큰 소수와 원시근을 기반으로 생성	큰 소수의 제곱을 기반으로 생성	두 큰 소수의 곱을 기반으로 생성
키 길이	일반적으로 RSA보다 짧음 (같은 보안 수준에서)	RSA와 비슷한 키 길이	RSA와 비슷한 키 길이	상대적으로 긴 키가 필요
수학적 기반	이산 로그 문제와 타원 곡선 문제 기반	이산 로그 문제 기반	소인수분해 문제에 기반	소인수분해 문제에 기반

Thanks!

김 혜 정(hyejeong@pel.sejong.ac.kr)