2026/07/02, 2026 보안 기초 세미나

암호학과 네트워크 보안

- 10장 비대칭 키 암호 -

김 민 식 (minsik@pel.sejong.ac.kr)

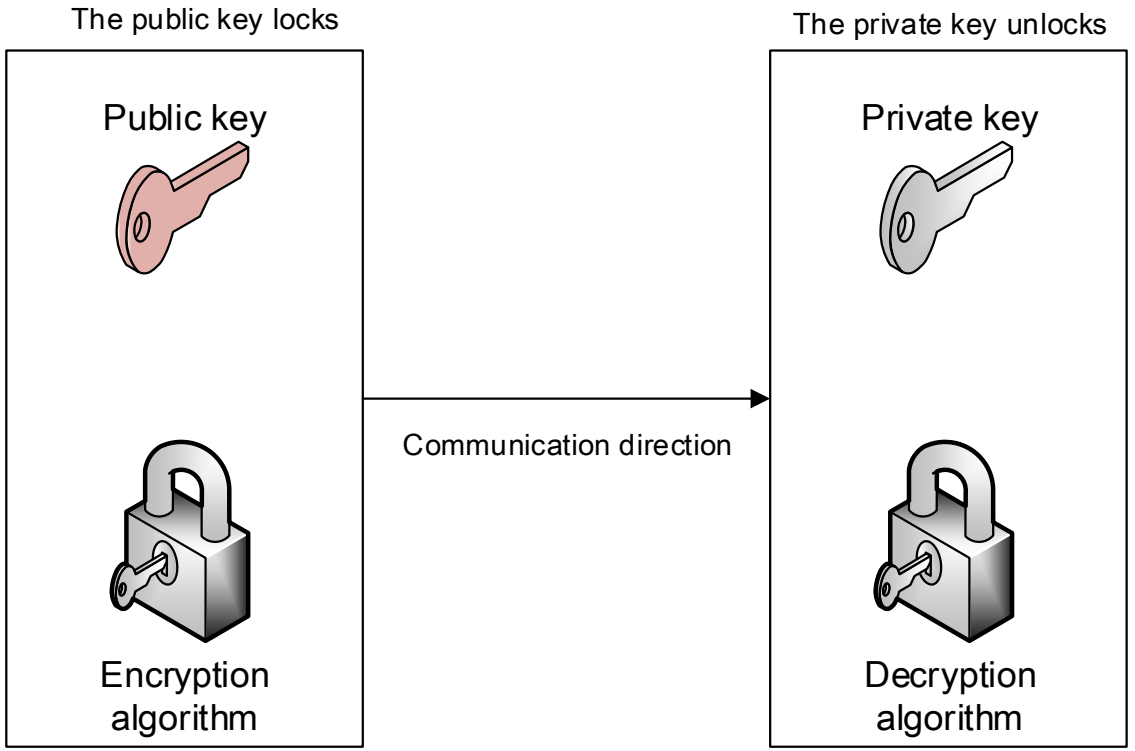
세종대학교 프로토콜공학연구실

목 차

- 비대칭 키(Asymmetric-key Cryptography)
- 배낭 암호(Knapsack Cryptosystem)
- RSA(Rivest, Shamir, Adleman)
- Rabin
- ElGamal
- 타원 곡선 기반 암호시스템(ECC, ElipticCurve Cryptography)

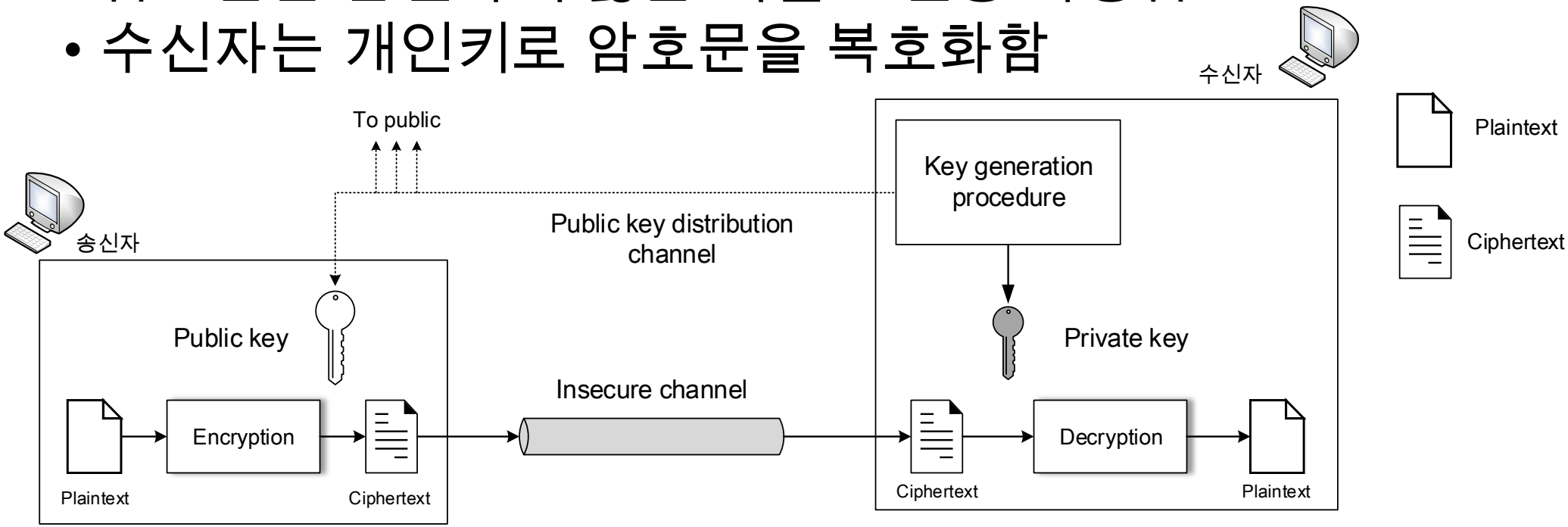
비대칭 키

- 정의
  - 서로 다른 두 개의 키인 공개키와 개인키를 사용하여 암호화와 복호화를 수행하는 방식



비대칭 키

- 암호화·복호화 구조
  - 수신자는 공개키와 개인키 한 쌍을 생성함
  - 공개키는 공개 채널(인터넷, 이메일 등)로 배포 가능함
  - 송신자는 수신자의 공개키로 평문을 암호화함
  - 암호문은 안전하지 않은 채널로 전송 가능함
  - 수신자는 개인키로 암호문을 복호화함



비대칭 키

- 대칭 키와 비대칭 키 암호시스템 비교

구분	대칭 키 암호시스템	비대칭 키 암호시스템
키 구조	송신자와 수신자가 같은 키 사용	공개키와 개인키 서로 다른 키 쌍 사용
키 개수	통신 쌍마다 1개의 비밀키 필요	사용자마다 공개키 1개, 개인키 1개 필요
키 배포	비밀키를 안전하게 공유해야 하므로 어려움	공개키는 공개 채널로 배포 가능함
속도	빠름	상대적으로 느림
주요 용도	대용량 데이터 암호화	공개키로 암호화하는 경우: 암호화에 사용할 비밀키 전달 개인키로 암호화하는 경우: 전자서명 생성, 인증서 서명

비대칭 키

- 일방향 함수(OWF, one-way function)
  - 정방향 계산은 쉽지만 역방향 계산은 매우 어려운 함수
    - e.g., 소인수분해, 이산대수, 타원곡선 이산대수, 해시함수
- 트랩도어(Trapdoor)
  - 특정 비밀 정보를 알고 있을 때만 어려운 역연산을 효율적으로 수행할 수 있게 해주는 정보
- 트랩도어 일방향 함수(Trapdoor one-way function)
  - 3개의 성질을 만족하는 일방향 함수의 일종임
    - 성질
      1. 주어진 x가 있다면 $y = f(x)$를 계산하는 것은 쉬움
      2. 주어진 y가 있다면 $x = f^{-1}(y)$를 계산하는 것은 어려움
      3. y와 트랩도어가 주어지면 x를 쉽게 계산할 수 있음

비대칭 키

- 트랩도어 일방향 함수(Trapdoor one-way function)
  - 예제 10.1

$p = 61, q = 53$일 때, $n = p \times q$를 구하고 반대로 $n = 3233$만 주어졌을 때 p와 q를 구하시오.

    - 풀이
      - $n = 61 \times 53 = 3233$
      - 소인수분해가 필요하므로 역계산은 어려움
  - 예제 10.2

n이 매우 큰 합성수일 때, 함수 $y = x^k \, mod \, n$이 트랩도어 일방향 함수가 되는 이유를 설명하시오.

    - 풀이
      - x, k, n이 있다면 y를 구하는 것은 쉬우나 n, k, y 이 주어졌을 때는 x를 구하는 것이 어려움
      - 단, k의 역원 $k^{-1} \, mod \, \varphi(n)$을 알면 x를 쉽게 계산할 수 있음

목 차

- 비대칭 키(Asymmetric-key cryptography)
- 배낭 암호(Knapsack cryptosystem)
- RSA(Rivest, Shamir, Adleman)
- Rabin
- ElGamal
- 타원 곡선 기반 암호시스템(ECC, ElipticCurve Cryptography)

배낭 암호

- 정의
  - 평문 비트를 0과 1의 선택 벡터x로 간주하고, 선택된 수열 원소a들의 합s를 암호문으로 사용하는 공개키 암호 방식

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

- 특징
  - 선택 벡터 x와 배낭 수열 a가 주어지면 합 s 계산은 쉬움
  - 공개키 수열 a와 합 s만 주어졌을 때 s를 만드는 선택 벡터 x $(inv_knapsackSum(s,a))$를 찾는 것은 어려움
- 키 구성
  - 공개키:수열 a
  - 개인키:초증가 수열 b, 비밀값 r, n

배낭 암호

- 초증가 순서 짝(Superincreasing tuple) (1/2)
  - 정의
    - 각 원소가 이전 원소들의 합보다 큰 수열
    - $a_i > a_1 + a_2 + \ldots + a_{i-1}$
  - 배낭암호의 활용
    - (암호화)수열 a의 각 원소와 선택 벡터 x의 각 원소를 곱하여 총합 s를 구함으로써, $s = knapsackSum(a, x)$계산이 가능함
    - (복호화)수열 a의 가장 큰 원소a_{i-1}부터 역순으로 총합과 비교하며, 선택 벡터 x에서 각 원소의 포함 여부를 찾아내어 $x = inv\_knapsackSum(s, a)$ 계산이 쉬워짐

배낭 암호

- 초증가 순서 짝(Superincreasing tuple) (2/2)
  - $knapsackSum$과 $inv\_knapsackSum$ 알고리즘

knapsackSum$(x[1 \ldots k], a[1 \ldots k])$
{
  $s \leftarrow 0$
  for$(i = 1\ to\ k)$
  {
   $s \leftarrow s + a_i \times x_i$
  }
  return s
}

inv_knapsackSum$(s, a[1 \ldots k])$
{
  for$(i = k$ down to $1)$
  {
  if $s \geq a_i$
  {
    $x_i \leftarrow 1$
    $s \leftarrow s - a_i$
  }
   else $x_i \leftarrow 0$
  }
  return $x\ [1 \ldots k]$
}

배낭 암호

- 예제 10.3

$a = [17,25,46,94,201,400]$이고 $s = 272$일 때, $inv_knapsackSum$를 나타내라

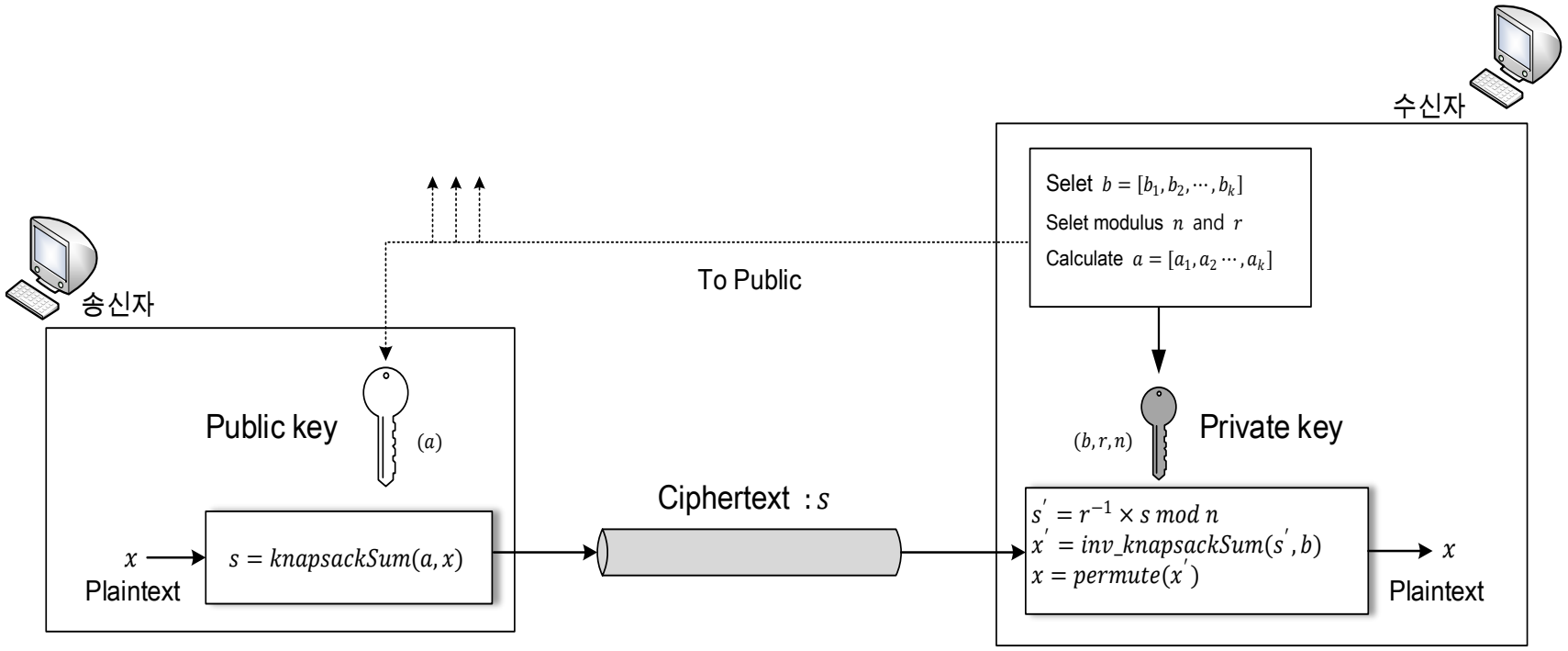
- 풀이

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

  - 이 경우 $x = [0,1,1,0,1,0]$이므로 $25, 46, 201$이 암호문 s에 포함되어 있음

배낭 암호

- 동작 원리 (1/3)


수신자
Selet $b = [b_1, b_2, \cdots, b_k]$
Selet modulus n and r
Calculate $a = [a_1, a_2 \cdots, a_k]$
To Public
송신자
Public key
(a)
(b, r, n)
Private key
Ciphertext : s
x
Plaintext
$s = knapsackSum(a, x)$
$s' = r^{-1} \times s \bmod n$
$x' = inv\_knapsackSum(s', b)$
$x = permute(x')$
x
Plaintext

배낭 암호

- 동작 원리 (2/3)
  - 키 생성
    1. 초증가 k순서 짝 $b = [b_1, b_2, \cdots, b_k]$ 를 만듦
    2. $n > b_1 + b_2 + \cdots + b_k$를 만족하는 모듈로 n을 선정함
    3. n하고 서로소면서 $1 \leq r \leq n-1$인 임의의 정수 r을 선택함
    4. $t_i = r \times b_i \, mod \, n$을 만족하는 임시 k 순서짝 $t = [t_1, t_2, \cdots, t_k]$ 을 생성함
    5. 임시 k 순서짝 t의 원소들을 순서에 따라 재배열한 $a = permute(t)$를 구함
    6. k순서짝 a를 공개 키로 사용하고 n,r,b는 개인 키의 인자로 사용함

배낭 암호

- 동작 원리 (3/3)
  - 암호화
    - 선택 벡터 x와 수열 a를 이용해 합 s 계산함
    - $s = knapsackSum(a, x) = x_1a_1 + x_2a_2 + \ldots + x_ka_k$
  - 복호화
    - 암호문 s로부터 선택 벡터 x를 복호화함
    - 비밀값 r^{-1}과 n을 이용하여 s' 계산
      - $s' = r^{-1} \times s \bmod n$
    - 초증가 수열 b를 이용하여 선택 벡터 x' 복원함
      - $x' = inv\_knapsackSum(s', b)$
    - 수열을 순서에 따라 재배열 하여 원래 평문 x 복원함
      - $x = permute\ (x')$

배낭 암호

- 예제 10.4 (1/2)

Bob은 초증가 수열 $b = [7,11,19,39,79,157,313]$와 모듈로 $n = 900$ 및 승수 $r = 37$ 과 치환표 (4,2,5,3,1,7,6)를 사용하여 공개키를 생성하였다. 문자 “g”(ASCII: 1100111)를 평문으로 사용할 때,

1. 공개키 a를 구하시오.
2. 암호문 s를 구하시오
3. 복호화를 수행하여 원래 평문을 복원하시오.

- 풀이
  1. 공개키 a 생성
     - $a_i = r \times b_i \; mod \; n \; (b = [7,11,19,39,79,157,313] \; r = 37, n = 900)$
     - $a = [543, 407, 223, 703, 259, 781, 409]$
  2. 암호문 s 계산
     - 문자 g= $(1100111)_2$, $x = [1, 1, 0, 0, 1, 1, 1]$
     - $s = knapsackSum(a, x) = 543 + 407 + 259 + 781 + 409 = 2399$
     - 암호문 $s = 2399$

배낭 암호

- 예제 10.4 (2/2)

Bob은 초증가 수열 $b = [7,11,19,39,79,157,313]$와 모듈로 $n = 900$ 및 승수 $r = 37$ 과 치환표 (4,2,5,3,1,7,6)를 사용하여 공개키를 생성하였다. 문자 “g”(ASCII: 1100111)를 평문으로 사용할 때,

1. 공개키 a를 구하시오.
2. 암호문 s를 구하시오
3. 복호화를 수행하여 원래 평문을 복원하시오.

i	a_i	s	$s \geq a_i$	x_i	$s \leftarrow s - a_i$
7	313	527	True	$x_7 = 1$	214
6	157	214	True	$x_6 = 1$	57
5	79	57	False	$x_5 = 0$	57
4	39	57	True	$x_4 = 1$	18
3	19	18	False	$x_3 = 0$	18
2	11	18	True	$x_2 = 1$	7
1	7	7	True	$x_1 = 1$	0

- 풀이
  3. 평문 x 복원
     - $37 \times r^{-1} \equiv 1 \; mod \; 900 \rightarrow r^{-1} = 73 \; mod \; 900$
     - $s' = r^{-1} \times \; s \; mod \; n \; = 73 \times 2399 \; mod \; 900 = 527$
     - $inv\_knapsackSum(527, b)$ 수행
       - $x = permute(x') = [1,1,0,0,1,1,1]$을 계산하여 $(1100111)_2$를 얻음

배낭 암호

- 안전성
  - 부분집합 합 문제
    - 공개키 수열 a와 암호문 s만으로 선택 벡터 x를 찾기 어려움
  - 트랩도어 구조
    - 수신자는 초증가 수열 b와 비밀값 r, n을 알고 있어 복호화가 쉬움
    - 공격자는 공개키 a만 보고 b를 찾기 어려워야 함
  - 현대에서 사용하지 않는 이유
    - 공개키 수열 a가 무작위 수열처럼 보이도록 만들지만, 실제로는 초증가 수열 b에서 변환된 구조를 가지고 있음
    - 공격자는 이 구조를 이용해 공개키 수열 a로부터 숨겨진 초증가 수열 또는 선택 벡터를 복원할 수 있음

목 차

- 비대칭 키(Asymmetric-key cryptography)
- 배낭 암호(Knapsack cryptosystem)
- RSA(Rivest, Shamir, Adleman)
- Rabin
- ElGamal
- 타원 곡선 기반 암호시스템(ECC, ElipticCurve Cryptography)

RSA

- 정의
  - 두 소수 p, q의 곱 $n = pq$와 공개 지수 e를 공개 키의 인자로 사용하고, p, q를 모르면 복호화가 어려운 소인수분해 문제 기반의 암호방식
  - 암호화

$$C = P^e \bmod n$$

  - 복호화

$$P = C^d \bmod n$$

- 키 구성
  - 공개 키(e,n),개인 키 (d)

*e: 공개 지수
*n: 공개 모듈러스
*d: 복호화 지수(개인키, 트랩도어 정보)

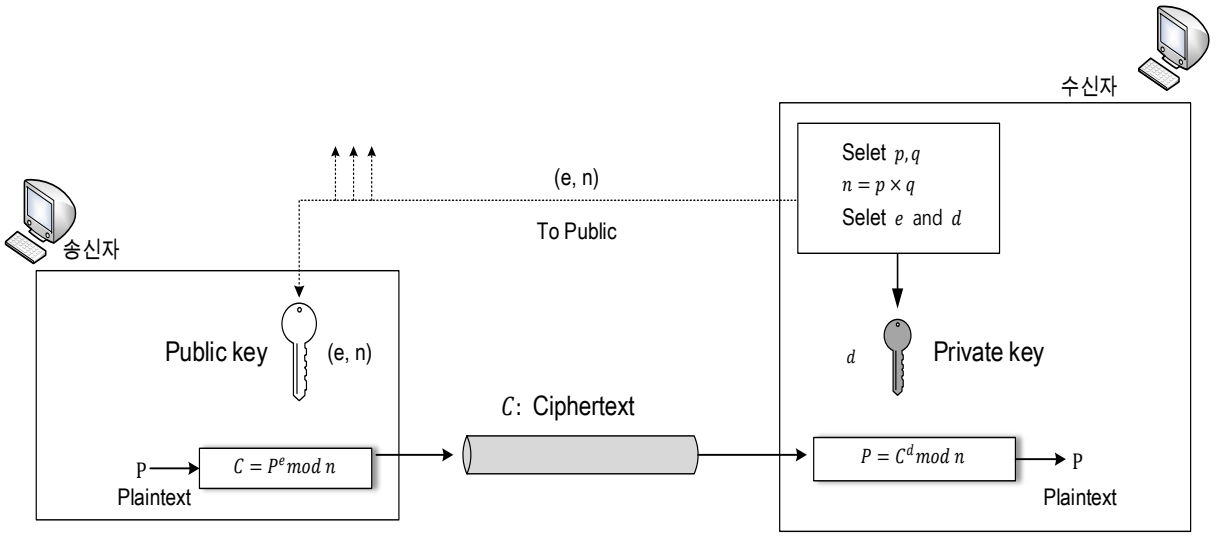
RSA

- 대수구조
  - 암호화/복호화
    - 공개되는 가환환 $R =< Z_n, +, \times >$ 을 사용함
    - 여기서 $Z_n = \{0, 1, 2, \ldots, n-1\}$ 이며, 모든 연산은 $mod\ n$에서 수행됨
    - 이때, n은 공개된 값이므로 해당 환의 구조는 누구나 알 수 있음
  - 키 생성
    - 비밀값 $\varphi(n)$을 이용해 군 $G =< Z_{\varphi(n)^*}, \times >$ 을 사용함
    - 이때, $\varphi(n)$은 공개되지 않은 비밀값임
      - $\varphi(n)$을 알면 d를 계산할 수 있으므로, $\varphi(n)$은 p, q와 함께 비밀로 유지해야 함

RSA

- 동작 원리 (1/2)
  - 키 생성
    - 순서 쌍 (e,n)을 공개 키, d를 개인키의 인자로 함
    - 보안을 위해 권장되는 각 소수 p와 q의 크기는 512비트임
      - p와 q는10진수로 약 154자릿수이며, n은 1024비트로서 10진수 309 자리가 됨

RSA_Key_Generation
{
두 개의 $p \neq q$ 큰 소수 p와 q를 선택한다
$n \leftarrow p \times q$
$\varphi(n) \leftarrow (p-1) \times (q-1)$
$1 < e < \varphi(n)$ 이면서 $\varphi(n)$과 서로소인 e를 선택한다.
$d \leftarrow e^{-1} \bmod \varphi(n)$
Public_key ← (e,n)
Private_key ← d
return Public_key and Private
}



RSA

- 동작 원리 (2/2)
  - 암호화
    - $C = \mathrm{P}^e \ mod \ n$
    - 평문P 크기는 n보다 작아야함
      - 만약 평문 길이가 n보다 크면 평문을 길이가 n보다 작은 블록으로 나누어 암호화함
    - 시간 복잡도는 $O(\log e \cdot k^2)$

```
RSA_Encryption(P, e, n)
{
   C ← Fast_Exponentiation(P, e, n)
   return C
}
```

  - 복호화
    - $P = C^d \ mod \ n$
    - 시간 복잡도는 $O(k^2)$

```
RSA_Decryption(C, d, n)
{
   P ← Fast_Exponentiation(C, d, n)
   return P
}
```

RSA

- 암·복호화 역관계 (1/3)
  - 증명
    - 오일러 정리 활용

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

- 복호화한 평문이 P_1이라고 하고 이것이 P와 같음

$$\mathrm{P}_1 = \mathrm{C}^d \ mod\ n = (\mathrm{P}^e \ mod\ n)^d \ mod\ n \ = \ \mathrm{P}^{ed} \ mod\ n$$
$$ed = k\varphi(n) + 1$$
$$\mathrm{P}_1 = \ \mathrm{P}^{ed} \ mod\ n \ \rightarrow \mathrm{P}_1 = \mathrm{P}^{k \times \varphi(n)+1} \ mod\ n \ = \mathrm{P} \ mod\ n$$

RSA

- 암·복호화 역관계 (2/3)
  - 예제 10.5

$p = 7, q = 11, e = 13$일 때, 평문 $P = 5$를 RSA 방식으로 암호화하고 복호화하시오.

- 풀이
  - 키생성
    - $n = 7 \times 11 = 77, \varphi(n) = (7-1)(11-1) = 60$
    - 개인 키 d 계산 $e \times d \equiv 1 \, mod \, \varphi(n), 13 \times d \equiv 1 \, mod \, 60$
    - $d = 37, 13 \times 37 = 481 \equiv 1 \, mod \, 60$
  - 암호화
    - $C = P^e \, mod \, n = 5^{13} mod \, 77 = 26$
  - 복호화
    - $P = C^d \, mod \, n = 26^{37} \, mod \, 77 = 5$

RSA

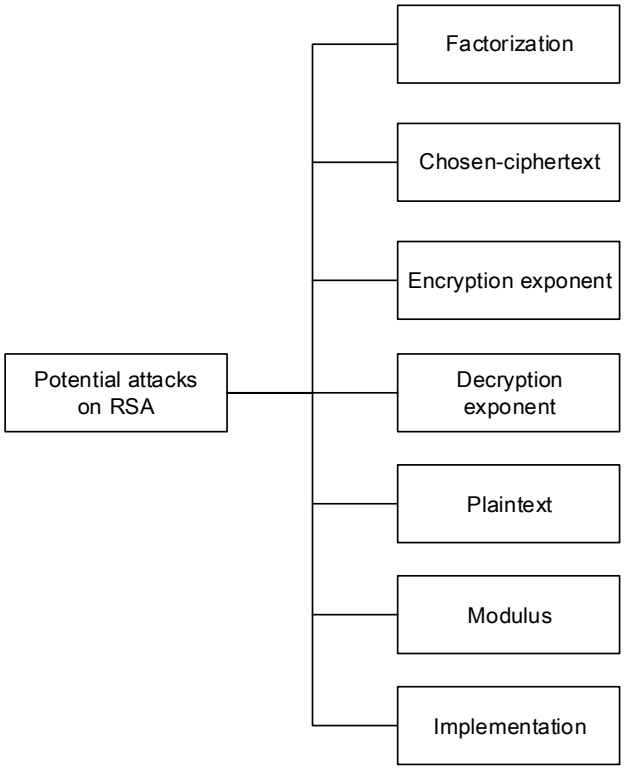
- 암·복호화 역관계 (3/3)
  - 예제 10.6

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

- 풀이
  - Jennifer에게 보내고자 하는 메시지가 “NO”라고 가정함
  - Ted는 각 문자를 두 자리 숫자로 된 코드(00부터 25까지)로 바꿈 (N=13, O=14)
  - 바꾼 문자코드 이어 붙여 4자리 숫자를 만듦(NO=1314)
  - 암호화(C): $1314^{343} = 33677\ mod\ 159197$
  - Jennifer는 33677을 받게 될 것이고 이를 개인 키 d를 사용하여 복호화(P): $33677^{12007} = 1314\ mod\ 159197$
  - Jennifer는 1314를 문자로 변환하여 “NO”를 얻음

RSA

- 공격
  - RSA는 기본적으로 높은 안전성을 제공하여 공격이 어려우나, 구현방식(e.g., 키 길이, 지수, 모듈러스, 패딩)이 부적절한 경우 다양한 취약점에 노출되어 공격이 발생할 수 있음
  - 종류
    - 소인수분해 공격
    - 선택 암호문 공격
    - 암호화 지수에 대한 공격
    - 복호화 지수에 대한 공격
    - 평문 공격
    - 일반 모듈로 공격
    - 실행에 대한 공격



RSA

- 공격:소인수분해 공격
  - 정의
    - n의 비트 길이가 짧거나 p, q가 취약하게 생성된 경우, 공개된 n을 소인수분해하여 p, q와 개인키 d를 구하는 공격
  - 공격방법
    - 공격자가 n을 소인수분해하여 p와 q를 구하면 $\varphi(n)$을 계산할 수 있고 $\varphi(n)$과 공개 지수 e를 이용하면 개인키 d를 계산할 수 있음
  - 보안 방법
    - 안전성을 위해 RSA는 모듈로 값 n을 1024비트 이상의 값으로 사용해야함
      - 최근에는 모듈로 값 n을 4096비트까지 사용하기를 권장함

RSA

- 공격:선택 암호문 공격
  - 정의
    - RSA의 곱셈 성질을 악용하여 공격자가 평문을 알아내는 공격
  - 공격방법
    a. 공격자는 $Z_n{}^*$에 속하는 임의의 정수 X 선택함
    b. 공격자는 $Y = C \times X^e mod\ n$를 계산함
    c. 공격자는 정상적인 송신자로 위장하여, Y를 수신자에게 송신해 복호화하고 $Z = Y^d mod\ n$를 얻음 (공격이 이루어지는 단계)
    d. 공격자는 아래의 수식을 활용해서 P를 구함
       - $Z = Y^d\ mod\ n = (C \times X^e)^d mod\ n = \left(\mathrm{C}^d \times X^{ed}\right) mod\ n =$ $\left(\mathrm{C}^d \times X\right) mod\ n = (\mathrm{P} \times X) mod\ n$
       - $Z = (\mathrm{P} \times X) mod\ n \rightarrow \mathrm{P} = (Z \times X^{-1})\ mod\ n$
  - 보안 방법
    - RSA-OAEP와 같은 안전한 패딩 적용

공격자의 능력
- 암호문 C와 공개 지수 e, 공개 모듈러스 n을 알고 있음
- 임의의 선택 암호문 Y를 생성할 수 있음
- 선택 암호문 Y에 대한 평문 Z를 얻을 수 있음

RSA

- 공격:암호화 지수에 대한 공격 (1/7)
  - 정의
    - 암호화 지수 e가 작고, 평문이 짧거나 패딩이 없는 경우 암호문으로부터 평문이 노출될 수 있는 공격
  - 공격 방법
    - $P^e < n$이면 $C = P^e mod\ n = P^e$가 되어 C의 e 제곱근을 계산하여 평문 P를 복원할 수 있음
  - 보안 방법
    - RSA-OAEP와 같은 안전한 패딩 적용
    - 공개 지수 $e = 2^{16} + 1 = 65537$를 사용함
      - 65537은 이진수에서 1의 개수가 적어 계산이 빠르기 때문에 사용함
      - $e = 3$보다 작은 지수 공격에 대한 위험을 줄일 수 있음

RSA

- 공격:암호화 지수에 대한 공격 (2/7)
  - 종류
    - Coppersmith 정리 공격
    - 브로드캐스트 공격
    - 관련된 메시지 공격
    - 짧은 패드 공격

RSA

- 공격:암호화 지수에 대한 공격 (3/7)
  - Coppersmith 정리 공격
    - 정의
      - $C = P^e mod\ n$ 에서 평문의 일부 정보가 알려졌을 때 나머지 평문을 구하는 공격
    - 공격 방법
      - 평문을 $P = A + x$로 표현함(A: 알려진 평문 부분, x: 모르는 부분)
      - $f(x) = (A + x)^e - C \equiv 0\ mod\ n$ 형태의 다항식을 구성함
      - 모듈로가 n이고 차수가 e인 다항식 $f(x)$에서 근 x가 $|x| < n^{(1/e)}$ 범위에 있으면, Coppersmith 정리를 이용해 x를 구할 수 있음
      - 구한 x를 이용해 원래 평문 $P = A + x$를 복원함
    - 보안 방법
      - RSA-OAEP와 같은 안전한 패딩 적용
      - 평문 일부가 예측 가능하거나 고정된 형식이 되지 않도록 함

공격자의 능력
- 암호문 C와 공개 지수 e, 공개 모듈러스 n을 알고 있음
- 평문의 일부 정보 A를 알고 있음

RSA

- 공격:암호화 지수에 대한 공격 (4/7)
  - 브로드캐스트 공격
    - 정의
      - 동일한 평문 P에 대한 암호문을 여러 수신자에게 브로드캐스트한 환경을 악용하여, 작은 공개 지수 e로 생성된 암호문들을 공격자가 수집한 뒤, CRT(Chinese Remainder Theorem)를 통해 평문 P를 알아내는 공격
    - 공격 방법
      - 송신자가 한 메시지 P를 3명의 수신자에게 보내는데 작은 공개 지수 $e = 3$을 사용하고 모듈로 값은 각각 n_1, n_2, n_3을 사용한다고 가정함
      - $\mathrm{C}_1 = \mathrm{P}^3 \, mod \, n_1 \quad \mathrm{C}_2 = \mathrm{P}^3 \, mod \, n_2 \quad \mathrm{C}_3 = \mathrm{P}^3 \, mod \, n_3$
      - CRT를 사용하여 P^3을 모듈로 $n_1 \times n_2 \times n_3$에 대해 재구성함
      - $N = n_1 \times n_2 \times n_3$ 라고 할 때, CRT로 $C' \equiv P^3 \, mod \, N$ 을 만족하는 C'를 구함
      - 평문 P는 각 n_i보다 작고, $e = 3$인 경우 $P^3 < N$ 이 성립함
      - 따라서 $C' = P^3$ 이 되며, 공격자는 $\mathrm{P} = \sqrt[3]{C}$을 계산해 평문 P를 복원함
    - 보안 방법
      - 동일한 평문을 여러 수신자의 공개키로 그대로 암호화하지 않음

RSA

- 공격:암호화 지수에 대한 공격 (5/7)
  - 관련된 메시지 공격(Related Message Attack)
    - 정의
      - 서로 다른 평문들이 완전히 독립적이지 않고 $P_2 = P_1 + r$와 같은 알려진 관계를 가질 때 발생할 수 있는 공격임
    - 공격 방법
      - 두 암호문 C_1, C_2와 평문 간 관계를 이용해 다항식을 구성하고, 두 다항식의 공통 근을 찾아 평문을 복원함
        - e.g., Franklin-Reiter 공격
          ($P_2 = P_1 + r$이고 같은 공개키 (n, e)로 암호화된 경우)
        - $C_1 = P_1^e \ mod \ n$, $C_2 = P_2^e \ mod \ n$일 때 $C_2 = (P_1 + r)^e \ mod \ n$임
        - $f_1(x) = x^e - C_1, \ f_2(x) = (x + r)^e \ - \ C_2$
        - 두 다항식의 공통 근 $x = P_1$을 찾아 평문을 복원함
    - 보안 방법
      - 같은 메시지 구조나 고정된 차이를 가지는 평문이 그대로 암호화 되지 않도록 함

RSA

- 공격:암호화 지수에 대한 공격 (6/7)
  - 짧은 패드 공격(Short Pad Attack)

*t:패드 r의 비트 길이

    - 정의
      - 같은 평문 P에 서로 다른 짧은 패드 r_1, r_2를 붙여 같은 RSA 공개키 (n, e)로 암호화할 때 발생할 수 있는 공격
    - 공격 방법
      - 패딩된 평문을 $M_1 = 2^t P + r_1, M_2 = 2^t P + r_2$ 라고 하면 두 패딩된 메시지는 $M_2 = M_1 + \Delta$ 관계를 가짐
        - $\Delta = r_2 - r_1$
      - 공격자는 r_1, r_2를 직접 알지 못하지만, 패드 길이 t가 짧으면 $|\Delta| < 2^t$ 범위에 있음을 알 수 있음
      - 패드 길이 t가 짧으면 Δ 의 범위가 작으므로, Coppersmith 방법으로 Δ를 찾을 수 있음
      - Δ를 구한 뒤에는 관련 메시지 공격을 적용하여 M_1을 복원하고, 패드를 제거해 원래 평문 P를 구할 수 있음

RSA

- 공격:암호화 지수에 대한 공격 (7/7)
  - 짧은 패드 공격(Short Pad Attack)
    - 보안 방법
      - RSA-OAEP와 같은 안전한 패딩 적용
      - 같은 평문을 같은 RSA 공개키로 반복 암호화하지 않음

RSA

- 공격:복호화 지수에 대한 공격 (1/3)
  - 정의
    - 복호화 지수 d가 노출되거나 너무 작은 값으로 설정되어 개인키 인자 d를 복원하고 암호문을 복호화하는 공격
  - 종류
    - 공개된 복호화 지수 공격
    - 작은 복호화 지수 공격

RSA

- 공격:복호화 지수에 대한 공격 (2/3)
  - 공개된 복호화 지수 공격
    - 정의
      - 공격자가 복호화 지수 d를 알게 되었을 때 발생하는 공격
    - 공격 방법
      - 공격자가 암호문 C와 공개 모듈러스 n, 복호화 지수 d를 알면 평문 P를 직접 계산할 수 있음$(P = C^d mod\ n)$
    - 보안 방법
      - 복호화 지수 d를 공개하지 않고 안전하게 보관함
        - d가 노출된 경우 기존 키를 폐기하고 새로운 p, q, n, e, d를 생성해야 함

RSA

- 공격:복호화 지수에 대한 공격 (3/3)
  - 작은 복호화 지수 공격
    - 정의
      - 복호화 속도를 높이기 위해 개인키 인자 d를 너무 작게 선택했을 때 발생하는 공격
    - 공격 방법
      - 개인키 인자 d가 매우 작으면 e/n이 k/d에 가까운 근사값이 됨
      - 대표적으로 Wiener 공격은 $d < \left(\frac{1}{3}\right) n^{1/4}$ 일 때 연속분수 방법으로 d를 복원함
      - d가 복원되면 공격자는 $C^d \ mod \ n$을 계산하여 암호문을 복호화 할 수 있음
    - 보안 방법
      - 복호화 지수 d는, $d \geq \left(\frac{1}{3}\right) n^{1/4}$ 이 성립해야 안전함
      - RSA 키 생성 알고리즘을 사용하여 충분히 큰 d가 생성되도록 함

RSA

- 공격:평문 공격 (1/4)
  - 정의
    - 공격자가 평문에 관련된 일부 정보(e.g., 평문 길이,패턴)를 알고 있을 때 발생하는 공격
  - 종류
    - 짧은 메시지 공격
    - 순환 공격
    - 감춰지지 않는 공격

RSA

- 공격:평문 공격 (2/4)
  - 짧은 메시지 공격
    - 정의
      - 평문의 길이가 짧거나 가능한 후보가 적을 때, 공격자가 평문 후보를 암호화하여 암호문과 비교함으로써 원래 평문을 찾는 공격
    - 공격 방법
      - 가능한 평문 후보들을 하나씩 RSA 공개키로 암호화함 $(C_i = P_i{}^e \ mod\ n)$
      - 실제 암호문 C와 비교하여 일치하는 후보를 찾음
      - 일치한 후보를 원래 평문 P로 판단함
    - 보안 방법
      - 짧거나 예측 가능한 평문을 그대로 RSA로 암호화하지 않음
      - RSA-OAEP와 같은 안전한 패딩 적용

공격자의 능력
- 평문 후보의 범위가 작다는 사실을 알고 있음
- 암호문 C와 공개 지수 e, 공개 모듈러스 n을 알고 있음

RSA

- 공격:평문 공격 (3/4)

공격자의 능력
- 암호문 C와 공개 지수 e, 공개 모듈러스 n을 알고 있음

  - 순환 공격
    - 정의
      - 암호문 C를 반복적으로 암호화했을 때 발생하는 순환 구조를 이용해 평문 P를 찾는 공격
    - 공격 방법
      - 동일한 암호문이 나올 때까지 반복함

가로챈 암호문:C

$$C_2 = C_1{}^e \ mod\ n$$
$$\vdots$$
$C_k = C_{k-1}{}^e \ mod\ n \ \rightarrow \ C_k = C$ 라면, 여기서 멈춤, 그러면 $P = C_{k-1}$임

    - 보안 방법
      - 충분히 큰 RSA 모듈러스 n을 사용함
      - RSA-OAEP와 같은 안전한 패딩 적용

RSA

- 공격:평문 공격 (4/4)
  - 감추어지지 않는 메시지 공격
    - 정의
      - 암호화를 수행했음에도 암호문이 평문과 같아지는 경우를 이용하는 공격
    - 공격 방법
      - 특정 평문 P에 대해 $C = P^e mod\ n = P$가 되면, 공격자는 암호문 C만 보고도 평문 P를 알 수 있음
        - e.g., $n = 33, e = 3, P = 10$일 때 $C = 10^3 mod\ 33 = 1000\ mod\ 33 = 10$ 암호문 C와 평문 P가 같아짐
    - 보안 방법
      - 암호화 결과가 평문과 같은 경우에는 해당 평문을 사용하지 않아야 함
      - RSA-OAEP와 같은 안전한 패딩 적용

RSA

- 공격:일반 모듈로 공격
  - 정의
    - 여러 사용자가 동일한 모듈러 값 n을 사용할 경우 발생하는 공격
  - 공격 방법
    - 한 사용자의 개인키 d가 노출되면 동일한 n을 사용하는 다른 사용자의 개인키도 계산될 수 있음
      - e.g., A :$e_1 = 7, n = 77, d_1 = 43$, B :$e_2 = 13, n = 77, d_2 = ?$
        - A의 개인키 d_1이 노출되면 $\varphi(n)$ 을 추정할 수 있고, 공개된 e_2를 이용해 B의 개인키 d_2도 계산 가능함: $e_2 \times d_2 \equiv 1 \, mod \, \varphi(n)$
        - $n = 77 = 7 \times 11$ 이므로 $\varphi(n) = 60$, $e_2 \times d_2 \equiv 1 \, mod \, \varphi(n)$에 대입하면 $13 \times d_2 \equiv 1 \, mod \, 60 \rightarrow d_2 = 37$
  - 보안 방법
    - 사용자마다 서로 다른 n값을 사용해야 함
    - RSA-OAEP와 같은 안전한 패딩 적용

RSA

- 공격:실행에 대한 공격 (1/5)
  - 정의
    - RSA 알고리즘 자체의 수학적 취약점이 아닌, 구현 및 실행 과정에서 발생하는 정보를 이용하는 공격
  - 종류
    - 타이밍 공격(Timing Attack)
    - 파워 공격(Power Attack)

RSA

- 공격:실행에 대한 공격 (2/5)
  - 타이밍 공격
    - 정의
      - 고속 지수 알고리즘에 기초한 공격으로, 암호화 또는 복호화 수행 시간의 차이를 이용하여 개인키 d를 추정하는 공격
    - 공격 방법
      - 공격자는 암호문 $C_1 \sim C_m$의 복호화 시간 $T_1 \sim T_m$을 측정함
      - 각 암호문 C_i에 대해 내부 곱셈 연산 시간을 t_i라고 함
        - t_i: $Result = Result \times \mathrm{C}_i \; mod \; n$ 수행 시간
      - 시간 차이를 분석하여 개인키 d의 비트 정보를 추정할 수 있음

RSA

- 공격:실행에 대한 공격 (3/5)
  - 타이밍 공격
    - 공격 방법
      1. 첫 번째 비트 설정 $d_0 = 1$
      2. 각 암호문 C_i에 대한 전체 복호화 시간 T_i를 측정
      3. $d_0 = 1$에 해당하는 곱셈 시간 t_i를 T_i에서 제거
      4. d_1부터 d_{k-1}까지 각 비트를 순서대로 추정
      5. 현재 비트 d_j가 1이라고 가정하고 곱셈 시간 t_i를 다시 계산
      6. $d_j = 1$이라고 가정했을 때의 보정 시간 D_i를 계산
         $D_i = T_i - t_i$
      7. 보정 전 시간 T_i와 보정 후 시간 D_i의 분산 차이를 계산
      8. 분산 차이를 이용하여 현재 비트 d_j를 0 또는 1로 결정하고 결정된 비트 d_i의 영향을 T_i에서 제거하고 다음 비트 분석으로 이동

*t_i:현재 추정 중인 비트가 1이라고 가정했을 때 발생하는 내부 곱셈 연산 시간
*D_i:전체 복호화 시간 T_i에서 예상 곱셈 시간 t_i를 제거한 값

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

RSA

- 공격:실행에 대한 공격 (4/5)
  - 타이밍 공격
    - 보안 방법
      - 계산 과정에 랜덤한 지연 시간을 추가하여 공격자가 실행 시간 차이로 개인키를 추정하기 어렵게 함
        - 시간 측정값에 잡음을 추가함
      - 암호문을 복호화하기 전에 난수로 곱하는 블라인딩 기법 추가
        - 복호화 입력 자체를 난수로 바꿔 원래 암호문과 실행 시간의 관계를 숨김
          1. 비밀로 된 난수 r을 1과 $n-1$ 사이에서 선택
          2. $C_1 = C \times r^e \ mod \ n$을 계산
          3. $\mathrm{P}_1 = \mathrm{C}_1{}^d \ mod \ n$을 계산
          4. $\mathrm{P} = \mathrm{P}_1 \times r^{-1} mod \ n$을 계산

RSA

- 공격: 실행에 대한 공격 (5/5)
  - 파워 공격
    - 정의
      - 암호화 또는 복호화 과정에서 소비되는 전력을 측정하여 개인키 d를 추정하는 공격
    - 공격 방법
      1. 공격자는 암호화 또는 복호화 과정에서 발생하는 전력 소비 파형을 측정함
      2. 연산 과정에서 입력값이나 개인키 비트에 따라 전력 소비 패턴이 달라지는지 분석함
      3. 특정 연산이 수행된 구간과 수행되지 않은 구간의 전력 차이를 비교함
      4. 전력 파형의 차이를 이용해 개인키 d의 비트 정보를 추정함
      5. 추정한 비트들을 조합하여 개인키 d를 복원함
    - 보안 방법
      - 개인키 비트에 따라 연산 흐름이 달라지지 않도록 구현함

RSA

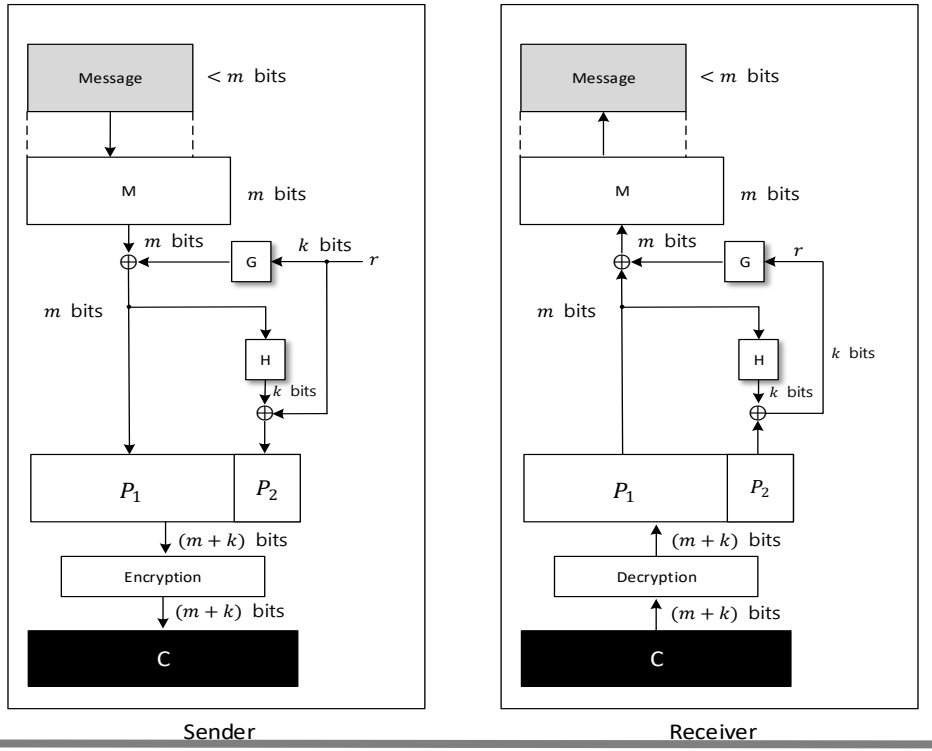
- 보안 설정

구분	권장사항	의미
키 길이	n은 최소 1024비트 이상 사용	모듈로 n이 작으면 소인수분해가 쉬워질 수 있음
소수 크기	p, q는 각각 최소 512비트 이상 사용	p, q가 작으면 $n = pq$를 분해하기 쉬워짐
소수 선택	p, q는 서로 너무 가까운 값이면 안 됨 ($\|p - q\| > 2^{\frac{nlen}{2} - 100}$)	두 소수가 가까우면 특수한 소인수분해 공격에 취약함
소수 조건	$p-1, q-1$은 큰 소인수를 가져야 함 (RSA-2048 기준 140비트 이상의 큰 소인수 권장)	$p-1, q-1$ 구조를 이용한 공격(Pollard $p-1$)을 어렵게 함
비율 조건	p와 q 사이에 예측 가능한 관계가 없어야 함	p, q 사이의 관계를 이용한 공격을 방지함
모듈로 사용	같은 n을 여러 사용자가 공유하면 안 됨	한 사용자의 키 노출이 다른 사용자에게도 영향을 줄 수 있음
공개 지수	$e = 2^{16} + 1 = 65537$ 또는 근처 값 사용	너무 작은 공개 지수 사용을 피하기 위함
키 노출 대응	개인키 d가 노출되면 n, p, q, e, d 모두 변경	관련 키 정보가 함께 노출될 수 있음
패딩	메시지는 OAEP로 패딩	평문 관련 공격과 선택 암호문 공격을 어렵게 함

RSA

- 방어 기법 (1/3)
  - 최적 비대칭 암호 패딩(OAEP, Optimal Asymmetric Encryption Padding) (1/3)
    - 패딩된 메시지 M에 난수 r을 섞어 RSA 평문을 랜덤화하게 암호화 하는 방식

M:Padded message
r:One-time random number
P: Plaintext($P_1||P_2$)
C:Ciphertext
G:Public function($k-bit\ to\ m-bit$)
H:Public function($m-bit\ to\ k-bit$)



RSA

- 방어 기법 (2/3)
  - 최적 비대칭 암호 패딩(OAEP, Optimal Asymmetric Encryption Padding) (2/3)
    - 암호화 절차
      1. m비트 메시지가 되도록 패딩을 붙임
         - 이것을 M이라고 부름
      2. k비트를 갖는 난수 r을 선택함
         - r은 한 번만 사용할 것이며 사용 후에는 폐기함
      3. r을 정수 입력으로 m비트 정수를 출력하는 공개된 일방향 함수 $G(r)$를 사용
         - 평문의 첫 번째 부분인 $P_1 = M \oplus G(r)$ 생성
      4. m비트를 입력하여 r비트를 내보내는 공개된 함수 $H(P_1)$ 를 사용
         - 평문의 두 번째 부분인 $P_2 = H(\mathrm{P}_1) \oplus r$ 생성
      5. 송신자는 $C = \mathrm{P}^e = (P_1 || P_2)^e$를 생성한 다음 C를 수신자에게 전송

RSA

- 방어 기법 (3/3)
  - 최적 비대칭 암호 패딩(OAEP, Optimal Asymmetric Encryption Padding) (3/3)
    - 복호화 절차
      1. $\mathrm{P} = \mathrm{C}^{d} = (\mathrm{P}_1 || \mathrm{P}_2)$
      2. $\mathrm{H}(\mathrm{P}_1) \oplus \mathrm{P}_2 = \mathrm{H}(\mathrm{P}_1) \oplus \mathrm{H}(\mathrm{P}_1) \oplus r = r$을 이용하여 r 값을 재생성함
      3. 패딩된 메시지 값을 재생성하기 위해서 $\mathrm{G}(\mathrm{r}) \oplus \mathrm{P} = \mathrm{G}(r) \oplus \mathrm{G}(r) \oplus \mathrm{M} = \mathrm{M}$을 사용함
      4. M으로부터 패딩을 제거하고 원래의 메시지를 구함

RSA

- 안전성
  - 소인수분해 문제
    - RSA의 보안성은 $n = p \times q$에서 p, q를 찾기 어렵다는 점에 기반함
    - p, q를 알면 $\varphi(n)$을 계산할 수 있고, 이를 통해 개인키 d를 구할 수 있음
    - 따라서 현대 RSA는 충분히 큰 n을 사용하여 소인수분해를 현실적으로 어렵게 만듦
  - 키 길이
    - 현대 RSA는 보통 2048비트 이상의 n을 사용하여 p, q를 현실적으로 찾기 어렵게 만듦
      - 이때 p와 q는 각각 약 1024비트 크기의 소수로 선택됨

목 차

- 비대칭 키(Asymmetric-key cryptography)
- 배낭 암호(Knapsack cryptosystem)
- RSA(Rivest, Shamir, Adleman)
- Rabin
- ElGamal
- 타원 곡선 기반 암호시스템(ECC, ElipticCurve Cryptography)

Rabin

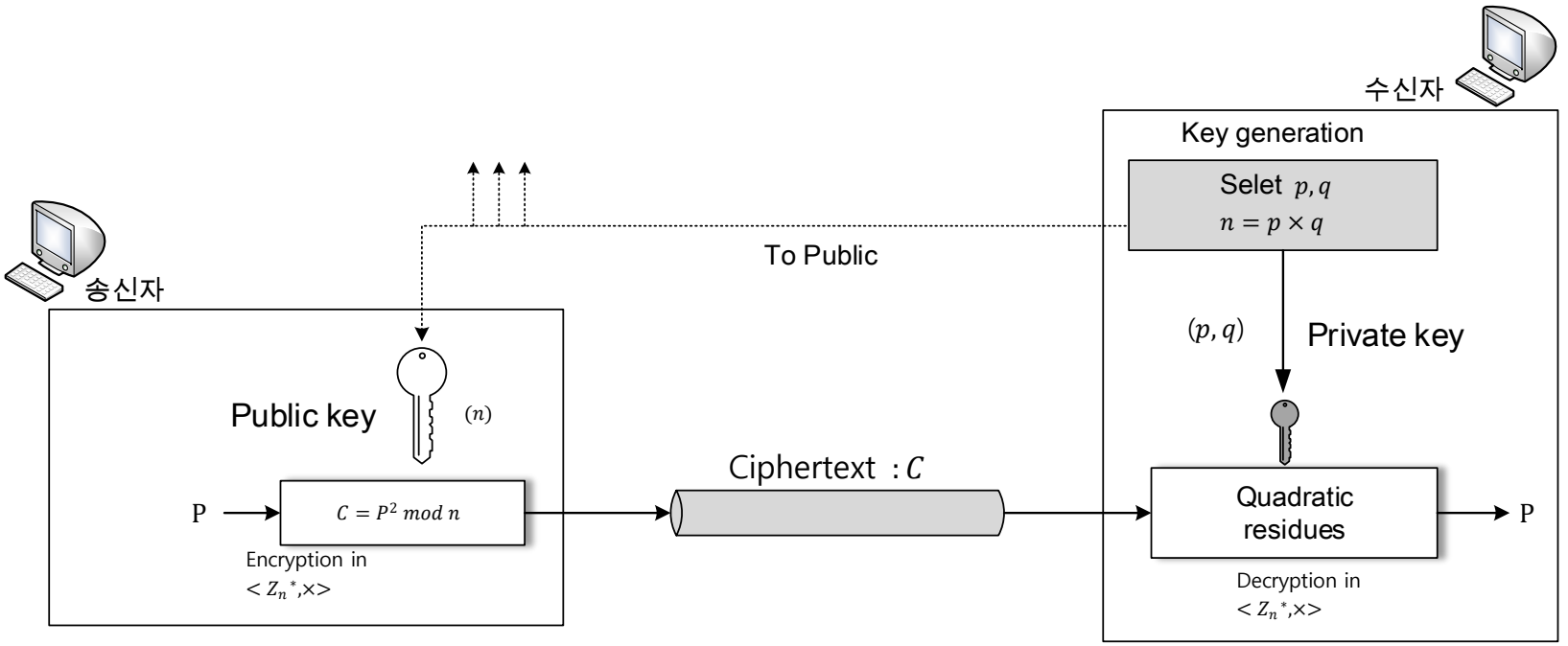
- 정의
  - 평문을 제곱하여 암호화하고, 소인수분해의 어려움에 기반하는 공개키 암호 방식

$$c = P^e \bmod n$$

- 키 구성
  - 공개 키 (n), 개인 키 (p, q)
    - 두 개의 소수 p와 q는 $4k+1$ 이나 $4k+3$ 형식을 가짐
      - $4k+1$ 형식일 경우에는 Tonelli-Shanks 알고리즘으로 제곱근 계산은 가능하지만, $4k+3$ 형태보다 복호화 과정이 복잡함
    - RSA 관점에서는 $e = 2, d = 1/2$로 간주할 수 있으나, 실제 개인키는 d가 아니라 (p, q)임

Rabin

- 동작 원리 (1/4)



Rabin

- 동작 원리 (2/4)
  - 키 생성

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 $\leftarrow (p, q)$
return Public_key and Private_key
}

1. 두 개의 큰 소수 p, q를 선택함($4k + 3$ 형태, $p \neq q$)
2. 선택한 두 소수 p, q를 곱해서 n을 만듦
3. 공개키를 n으로 설정함
4. 개인키를 (p, q)로 설정함
5. 공개키와 개인키를 반환함

Rabin

- 동작 원리 (3/4)
  - 암호화
    1. 평문 P를 제곱한 뒤 $mod\ n$연산을 수행하여 암호문 C를 계산함
    2. 계산된 암호문 C를 반환함

```
Rabin_Encryption(n, P)
{
   C ← P^2 mod n
   return C
}
```

  - 복호화
    1. 개인키 p, q를 이용하여 암호문 C의 제곱근을 각각 $mod\ p, mod\ q$에서 구함
    2. 네 가지 조합을 중국인의 나머지 정리로 결합하여 네 개의 평문 후보 P_1, P_2, P_3, P_4를 구함
    3. 후보 중 올바른 평문 P를 선택함

```
Rabin_Decryption (p, q, C)
{
   a1 ← +(C^((p+1)/4) mod p
   a2 ← −(C^((p+1)/4) mod p
   b1 ← +(C^((q+1)/4) mod q
   b2 ← −(C^((q+1)/4) mod q
   // 중국인 나머지 정리 알고리즘을 4번
      호출한다.
   P1 ← Chinese_Remainder (a1, b1, p, q)
   P2 ← Chinese_Remainder (a2, b2, p, q)
   P3 ← Chinese_Remainder (a2, b1, p, q)
   P4 ← Chinese_Remainder (a2, b2, p, q)

   return P1, P2, P3 and P4
}
```

Rabin

- 동작 원리 (4/4)
  - 복호화
    - 2차 합동에 대한 근에 근거하고 있음
      - P는 $Z_n{}^*$ 의 값이기 때문에 C는 $Z_n{}^*$ 안에 근을 반드시 가지고 있음
    - 복호화 결과가 유일하지 않음
      - 복호화는 4개의 평문 후보가 생성될 수 있음
      - 올바른 평문 선택을 위해 패딩 또는 메시지 형식 검사가 필요함
        - e.g., 특정 패턴, 중복 비트, 문자 형식 등

Rabin

- 예제 10.7 (1/2)

Rabin 암호에서 $p = 23, q = 7$이고 평문 $P = 24$일 때, 암호문 C를 구하고, 이를 복호화하여 가능한 평문 후보 4개를 구하시오.

- 공개키 (161), 개인 키(23, 7)
- 송신자가 평문 $\mathrm{P} = 24$를 수신자에게 보냄(161과 24는 서로소)
- 송신자는 $C = 24^2 = 93\ mod\ 161$을 계산해서 수신자에게 보냄
- 수신자는 93을 수신하고 다음 4개의 값을 계산함
  - $a_1 = +\left(93^{\frac{23+1}{4}}\right) mod\ 23 = 1\ mod\ 23$
  - $a_2 = -\left(93^{\frac{23+1}{4}}\right) mod\ 23 = 22\ mod\ 23$
  - $b_1 = +\left(93^{\frac{7+1}{4}}\right) mod\ 7 = 4\ mod\ 7$
  - $b_2 = -\left(93^{\frac{7+1}{4}}\right) mod\ 7 = 3\ mod\ 7$

Rabin

- 예제 10.7 (2/2)

Rabin 알고리즘을 설명하기 위해 간단한 예를 들어보기로 하자.

- 수신자는 4개의 가능한 답 $(a_1, b_1), (a_1, b_2), (a_2, b_1), (a_2, b_2)$을 선택하고 4개의 가능한 평문인 $116, 24, 137, 45$를 구함 (4개 모두 161과 서로소)

- 제곱을해서 모듈로 값을 구해보면 전부 암호문 93이 나옴
  - $116^2 = 93 \; mod \; 161$
  - $24^2 = 93 \; mod \; 161$
  - $137^2 = 93 \; mod \; 161$
  - $45^2 = 93 \; mod \; 161$
- 암호문만으로는 원래 평문을 유일하게 결정하기 어렵기 때문에 올바른 평문 선택을 위해 패딩 또는 중복 정보가 필요함

Rabin

- 안전성
  - 소인수분해 문제
    - Rabin 암호의 보안성도 $n = p \times q$에서 p, q를 찾기 어렵다는 점에 기반함
    - p, q를 알고 있는 수신자는 제곱근을 쉽게 계산할 수 있지만, 공격자는 n만으로 p, q를 찾기 어려움
  - 키 길이
    - Rabin도 RSA처럼 $n = p \times q$ 구조에 기반하므로 현대Rabin 암호는 2048비트 이상의 n을 사용하여 p, q를 현실적으로 찾기 어렵게 만듦

목 차

- 비대칭 키(Asymmetric-key cryptography)
- 배낭 암호(Knapsack cryptosystem)
- RSA(Rivest, Shamir, Adleman)
- Rabin
- ElGamal
- 타원 곡선 기반 암호시스템(ECC, ElipticCurve Cryptography)

ElGamal

- 정의
  - 이산대수 문제의 어려움에 기반하며, 매 암호화마다 난수 r을 사용해 암호문을 (C_1, C_2) 형태의 쌍으로 생성하는 공개키 암호 방식

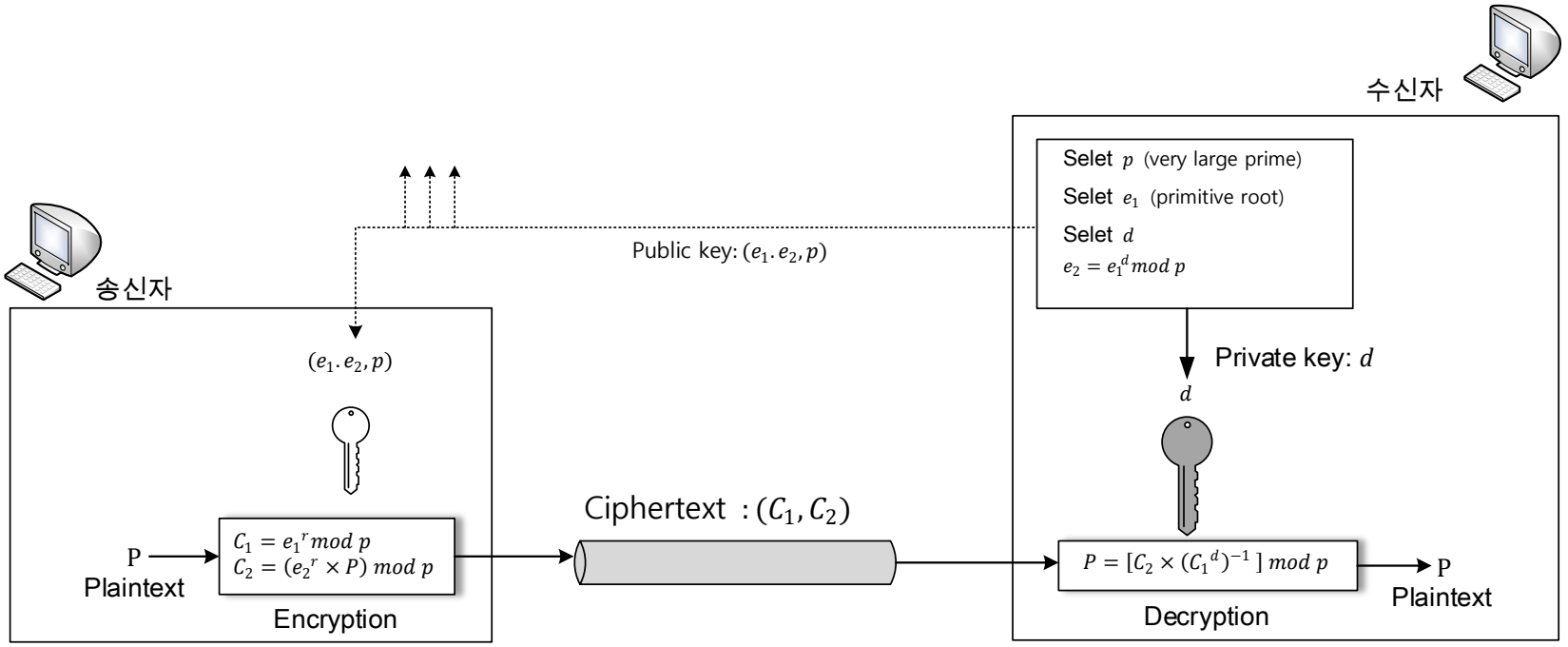
$$C_1 \leftarrow {e_1}^r \bmod p$$
$$C_2 \leftarrow (P \times {e_2}^r) \bmod p$$

- 키 구성
  - 공개 키(e_1, e_2, p),개인 키 d

*e_1: 원시근
(공개키 e_2와 암호문 C_1을 생성하기 위한 기준값)
*e_2: ${e_1}^d \bmod p$로 계산된 공개키 값
(암호화 시 ${e_2}^r$을 만들어 평문 P를 숨기는 데 사용됨)
*p:소수 모듈러스
*d: 개인키이자 복호화를 가능하게 하는 비밀 지수

ElGamal

- 동작 원리 (1/4)



ElGamal

- 동작 원리 (2/4)
  - 키 생성

ElGamal_key_Generation
{
Select a large prime p
Select d to be a member of the group $G = < Z_p^*, \times >$ such that $1 \leq d \leq p-2$
Select e_1 to be a primitive root in the group $G = < Z_p^*, \times >$
$e_2 \leftarrow {e_1}^d \ mod \ p$
Public_key $\leftarrow (e_1, e_2, p)$
Private_key $\leftarrow d$
return Public_key and Private_key
}

1. 큰 소수 p를 선택
2. 개인 키 d 선택 $(1 \leq d \leq p-2)$
3. p의 원시근 e_1 선택함
4. 공개 값 $e_2 = {e_1}^d \ mod \ p$ 계산
5. 공개키:(e_1, e_2, p), 개인키: d

ElGamal

- 동작 원리 (3/4)
  - 암호화

ElGamal_Encryption $(\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{p}, \boldsymbol{P})$
{
  Select a random integer r in the group $G = < Z_p^*, \times >$
  $C_1 \leftarrow {e_1}^r \ mod\ p$
  $C_2 \leftarrow (P \times {e_2}^r)\ mod\ p$
  return C_1 and C_2
}

1. 공개키 (e_1, e_2, p)와 평문 P 사용
2. 임의의 난수 r 선택
3. $C_1 = {e_1}^r \ mod\ p$ 계산
4. $C_2 = P \times {e_2}^r mod\ p$ 계산
5. 암호문 (C_1, C_2) 생성

ElGamal

- 동작 원리 (4/4)
  - 복호화

EIGamal_Decryption $(\boldsymbol{d}, \boldsymbol{p}, \mathbf{C_1}, \mathbf{C_2})$
{
    P ← $[\mathrm{C}_2(\mathrm{C}_1{}^{\mathrm{d}})^{-1}]\ mod\ p$
    return P
}

1. 개인키 d와 암호문 (C_1, C_2) 사용
2. ${C_1}^d\ mod\ p$ 계산
3. $({C_1}^d)^{-1}\ mod\ p$ 계산
4. $P = C_2 \times ({C_1}^d)^{-1}\ mod\ p$ 로 복호화

ElGamal

- 예제 10.8 (1/2)

Bob은 $p = 11$, 원시근 $e_1 = 2$, 개인키 $d = 3$을 선택하고 $e_2 = e_1{}^d mod\ p = 8$을 계산하였다.Alice는 $r = 4$와 평문 $P = 7$ 사용하여 암호화할 때 암호문(C_1, C_2)을 계산하라.

- 풀이
  - 암호화
    - 평문 $P = 7$
    - $C_1 = {e_1}^r\ mod\ 11 = 16\ mod\ 11 = 5\ mod\ 11$
    - $C_2 = (\mathrm{P} \times {e_2}^r)\ mod\ 11 = (7 \times 4096)\ mod\ 11 = 6\ mod\ 11$
    - 암호문: $(5, 6)$

ElGamal

- 예제 10.8 (2/2)

Bob은 $p = 11$, 원시근 $e_1 = 2$, 개인키 $d = 3$을 선택하고 $e_2 = e_1{}^d mod\ p = 8$을 계산하였다.Alice는 $r = 4$와 평문 $P = 7$ 사용하여 암호화할 때 암호문(C_1, C_2)을 계산하라

- 풀이
  - 복호화
    - 암호문 : $(C_1, C_2) = (5, 6)$, 개인키 $d = 3$
    - $C_1{}^d\ mod\ p\ =\ 5^3\ mod\ 11 =\ 125\ mod\ 11 =\ 4$
    - $\left(C_1{}^d\right)^{-1}\ mod\ 11 = 4^{-1}\ mod\ 11\ =\ 3$
    - $P\ =\ C_2 \times \left(C_1{}^d\right)^{-1}\ mod\ p = 6 \times 3\ mod\ 11\ =\ 7$
    - 평문 $P =\ 7$

ElGamal

- 공격 (1/2)
  - 알려진 평문 공격
    - 정의
      - 공격자가 평문 P와 대응되는 암호문 (C_1, C_2)를 알고 있을 때 암호화에 사용된 숨김값 ${e_2}^{\mathrm{r}}$을 알아내는 공격
    - 공격 방법
      - ElGamal 암호문은 $C_2 = P \times {e_2}^{\mathrm{r}} \, mod \, p$ 형태이므로 P의 역원을 이용해 숨김값 ${e_2}^{\mathrm{r}}$을 계산할 수 있음
        - ${e_2}^{\mathrm{r}} = C_2 \times P^{-1} mod \, p$
      - 만약 같은 난수 r로 암호화된 다른 암호문 (C_1, C_2')가 존재하면, 같은 숨김값 ${e_2}^{\mathrm{r}}$이 사용됨
      - 공격자는 C_2'에서 ${e_2}^{\mathrm{r}}$을 제거하여 다른 평문 P'를 복원할 수 있음
        - $P' = C_2' \times ({e_2}^{\mathrm{r}})^{-1} \, mod \, p$
    - 보안 방법
      - 난수 r은 암호화할 때마다 새롭게 생성해야 함

ElGamal

- 공격 (2/2)
  - 작은 모듈로 공격
    - 정의
      - 모듈러스 p가 너무 작을 때, 이산대수 문제를 쉽게 풀어 개인키 d 또는 난수 r을 구할 수 있는 공격
    - 공격 방법
      - $e_2 = e_1^{\,d} \, mod \, p$ 관계에서 이산로그를 계산하여 d를 구함
        - $d = \log_{e_1} e_2 \, mod \, p$
      - 또는 $C_1 = e_1{}^{\mathrm{r}} \, mod \, p$에서 이산로그를 계산하여 r을 구함
      - d또는 r을 알면 $e_2{}^{r}$ 또는 $C_1{}^{d}$를 계산할 수 있으므로 평문 P를 복호화 할 수 있음
    - 보안 방법
      - p는 최소 1024비트(약 300자리 10진수) 이상 사용해야 함

ElGamal

- 안전성
  - 이산대수 문제
    - ElGamal의 보안성은 이산대수 문제의 어려움에 기반함
    - 공개값 e_1, e_2, p가 주어져도 $e_2 = {e_1}^d \ mod\ p$를 만족하는 개인키 d를 찾기 어려움
  - 난수 r 사용
    - 암호화할 때마다 새로운 난수 r을 사용하여 같은 평문을 암호화해도 서로 다른 암호문이 생성됨
  - 키 길이
    - 현대적인 보안 수준에서는 최소 2048비트 이상의 p를 사용함

목 차

- 비대칭 키(Asymmetric-key cryptography)
- 배낭 암호(Knapsack cryptosystem)
- RSA(Rivest, Shamir, Adleman)
- Rabin
- ElGamal
- 타원 곡선 기반 암호시스템(ECC, ElipticCurve Cryptography)

타원 곡선 기반 암호시스템

- 정의
  - 타원곡선 기반 암호시스템은 유한체 위에서 정의된 타원곡선의 점 연산을 이용하는 공개키 암호 기법들의 프레임워크임
- 특징
  - 기준점 G와 개인키 d를 이용해 공개키 Q를 생성함

$$Q = dG$$

  - $Q = dG$계산은 쉽지만, G와 Q만 보고 d를 찾는 것은 어려움
  - 타원곡선 이산로그 문제의 어려움에 기반함

*Q: 공개 키
*d: 개인 키
*G: 기준점
*$Q = dG$:공개키 생성식

타원 곡선 기반 암호시스템

- 종류

종류	용도	분류	간단 설명
ECDH	키 교환	비대칭 키 기반 키 교환	타원곡선 공개키와 개인키를 이용해 양쪽이 같은 공유 비밀값을 생성함
ECDSA	전자서명	비대칭 키 전자서명	개인키로 서명하고 공개키로 서명을 검증함
EdDSA	전자서명	비대칭 키 전자서명	Edwards 곡선을 사용하는 전자서명 방식으로, ECDSA보다 구현이 단순하고 안전성이 좋음
EC-ElGamal	암호화/복호화	비대칭 키 암호 방식	ElGamal 암호를 타원곡선 위의 점 연산으로 구현한 방식임
ECIES	암호화/복호화	하이브리드 암호 방식	ECC로 공유 비밀값을 만들고, 실제 메시지는 대칭키 암호로 암호화함

타원 곡선 기반 암호시스템

- 타원 곡선 방정식
  - 일반 타원 곡선 방정식

$$y^2 = b_1xy + b_2y = x^3 + a_1x^2 + a_2x + a_3$$

  - 유한체 $GF(p)$ 위에서 사용하는 타원 곡선 방정식

$$y^2 \equiv x^3 + ax + b \ (mod\ p)$$

    - $4a^3 + 27b^2 \neq 0$이면 정칙 타원 곡선이고 $4a^3 + 27b^2 = 0$이면 특이 타원 곡선이 됨

타원 곡선 기반 암호시스템

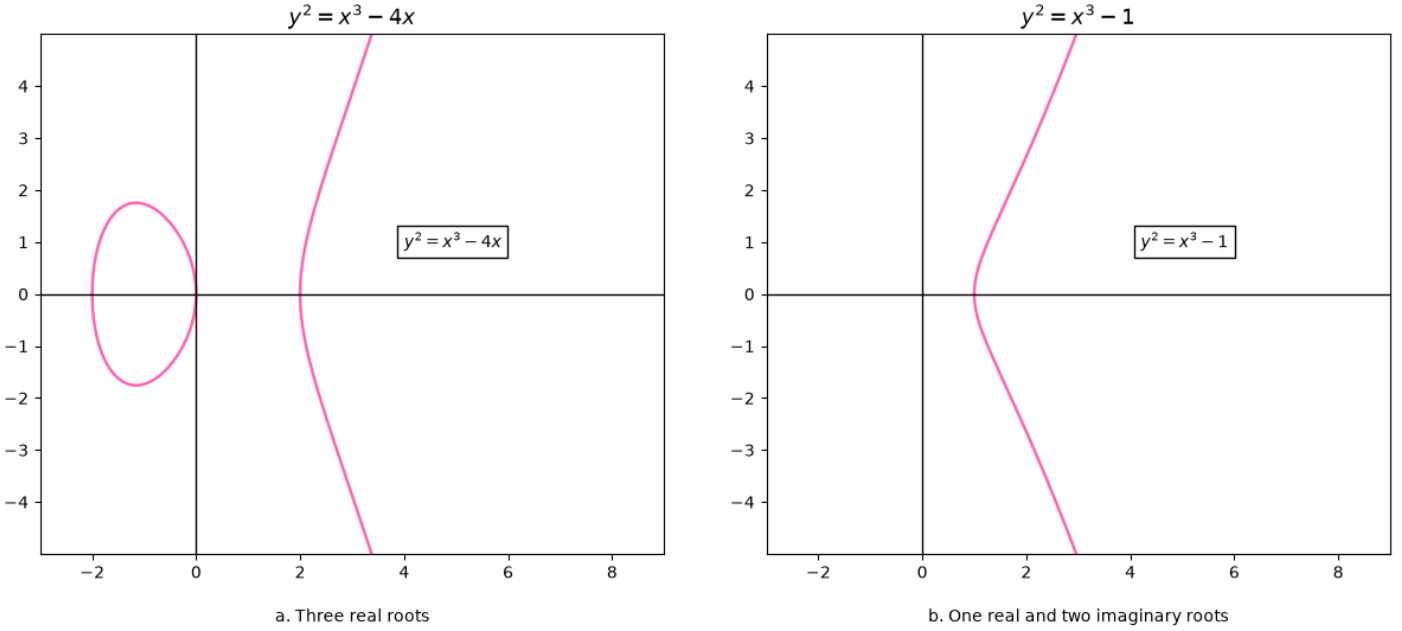
- 타원 곡선 종류
  - 정칙 타원 곡선
    - 정의
      - $y^2 = x^3 + ax + b$ 에서 특이점이 없는 타원곡선
    - 조건
      - $GF(p)$ 위에서는 $4a^3 + 27b^2 \neq 0 \ mod \ p$
  - 특이 타원 곡선
    - 정의
      - $y^2 = x^3 + ax + b$ 에서 특이점이 존재하는 타원곡선
    - 조건
      - $GF(p)$ 위에서는 $4a^3 + 27b^2 = 0 \ mod \ p$

*특이점:곡선 위의 한 점에서 접선이 유일하게 정의되지 않는 점

타원 곡선 기반 암호시스템

- 예제 10.9

두 타원 곡선 방정식의 그래프를 통해 실근의 개수와 곡선 형태의 차이를 확인하시오.



- 두 개 다 정칙 타원 곡선임 첫 번째 곡선은 3개의 실근$(-2, 0, 2)$을 갖고 두번째 곡선은 오직 1개의 실근(1)을 갖고 2개의 복소근을 가짐
- 실근에 개수에 따라 그래프의 형태가 달라짐

타원 곡선 기반 암호시스템

- 대수구조
  - 아벨 군
    - 타원곡선 위의 점들이 점 덧셈에 대해 닫힘성, 결합법칙, 항등원, 역원, 교환법칙을 만족하는 구조
  - 집합
    - $E(K) = \{ (x_1, y_1) \in K^2 \mid y^2 = x^3 + ax + b \} \cup \{ O \}$
      - O은 무한원점을 의미함
  - 덧셈 연산
    - 실수 상에서 정의하는 덧셈하고는 다르며, 곡선 상의 두 점을 더했을 때 곡선 상의 다른 한 점을 찾는 규칙을 의미함

$R = P + Q, P = (x_1, y_1), Q = (x_2, y_2)$이고 $R = (x_3, y_3)$

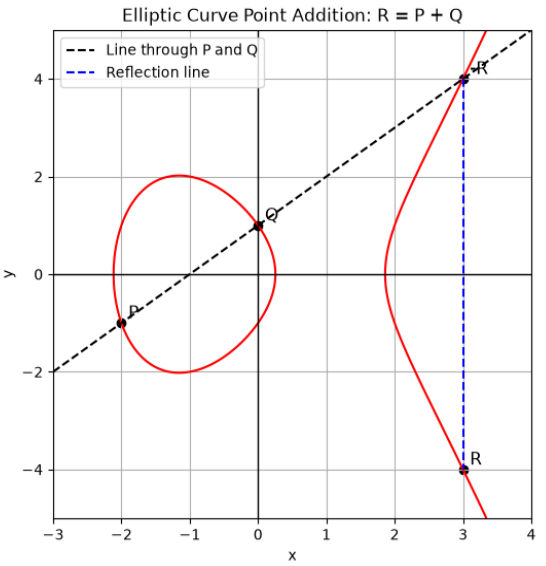
타원 곡선 기반 암호시스템

- 대수구조:아벨 군
  - 연산의 성질
    1. 닫힘:두 점 P, Q의 덧셈 결과 R이 다시 곡선 상에 있다는 성질
    2. 결합법칙:$(\mathrm{P}+\mathrm{Q})+\mathrm{R}=\mathrm{P}+(\mathrm{Q}+\mathrm{R})$
    3. 교환법칙:$\mathrm{P}+\mathrm{Q}=\mathrm{Q}+\mathrm{P}$
    4. 항등원의 존재성: 모든 점 P에 대해 $\mathrm{P}+\mathrm{O}=\mathrm{O}+\mathrm{P}=\mathrm{P}$를 만족하는 항등원 O가 존재하는 성질
    5. 역원의 존재성: 임의의 점 P에 대해 $\mathrm{P}+(-\mathrm{P})=O$를 만족하는 역원 $-\mathrm{P}$가 존재하는 성질

타원 곡선 기반 암호시스템

- 대수구조:덧셈 연산 (1/2)
  - 타원 곡선에서의 3가지 덧셈 유형
    1. 두 점 $P = (x_1, y_1), Q = (x_2, y_2)$의 각 좌표 값인 x좌표와 y좌표 값이 서로 다른 경우$(R = P + Q)$
       - P와 Q를 연결하는 직선은 곡선 상의 다른 한점인 $-R$에서 교차하게 됨
       - $R = (x_3, y_3)$를 구하기 위해서는 아래와 같은 절차를 따름

$$\lambda = (y_2 - y_1)/(x_2 - x_1)$$
$$x_3 = \lambda^2 - x_1 - x_2 \quad y_3 = \lambda(x_1 - x_3) - y_1$$

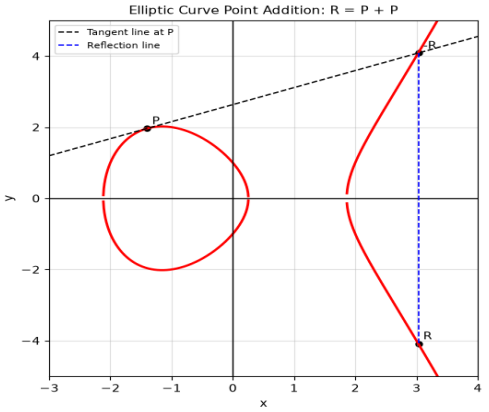


타원 곡선 기반 암호시스템

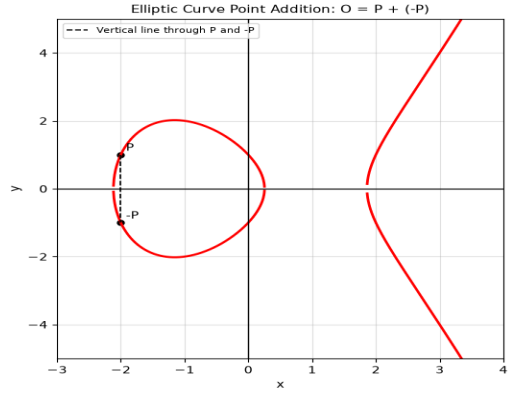
- 대수구조:덧셈 연산 (2/2)

2. 두 점이 겹치는 경우임($R = P + P$)
   - 기울기 λ 와 점 R의 좌표 값은 아래와 같은 식으로 구할 수 있음

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



3. 두 점이 합에 대한 역원 관계인 경우임($O = P + (-P)$)
   - 첫 번째 점이 $P = (x_1,\ y_1)$라면 두 번째 점은 $Q = (x_1, -y_1)$임
   - 두 점을 연결하는 직선은 곡선의 다른 한점에서 교차하지 않음
     - 무한원점 O혹은 영점이라고 부름



타원 곡선 기반 암호시스템

- $GF(p)$상의 타원 곡선 (1/4)
  - $y^2 \equiv x^3 + ax + b \ (mod\ p)$를 만족하는 점들의 집합 $E_p(a, b)$
  - 역원 구하기
    - 점(x, y)의 역원은 $(x, -y)$임
      - 여기서 $-y$는 y의 덧셈에 대한 역원임
  - 곡선 상의 점 찾기
    1. 처음 x값을 0으로 설정함
    2. x가 p보다 작은 경우 반복함
    3. x값을 $w \equiv x^3 + ax + b \ (mod\ p)$에 대입함$\left(y^2 \equiv w \ (mod\ p)\right)$
    4. w가 제곱수라면 그 y값들을 이용해 곡선 위의 점을 출력
    5. x값 검사가 끝났으므로 다음 x값으로 넘어감

ellipticCurve_points $(\boldsymbol{p}, \boldsymbol{a}, \boldsymbol{b})$
{
 $x \leftarrow 0$
 while $(x < p)$ {
  $w \leftarrow (x^3 + ax + b)\ mod\ p$
  if(w is perfect square in Z_p) output$(x, \sqrt{w})$ $(x, -\sqrt{w})$
  $x \leftarrow x + 1$
 }
}

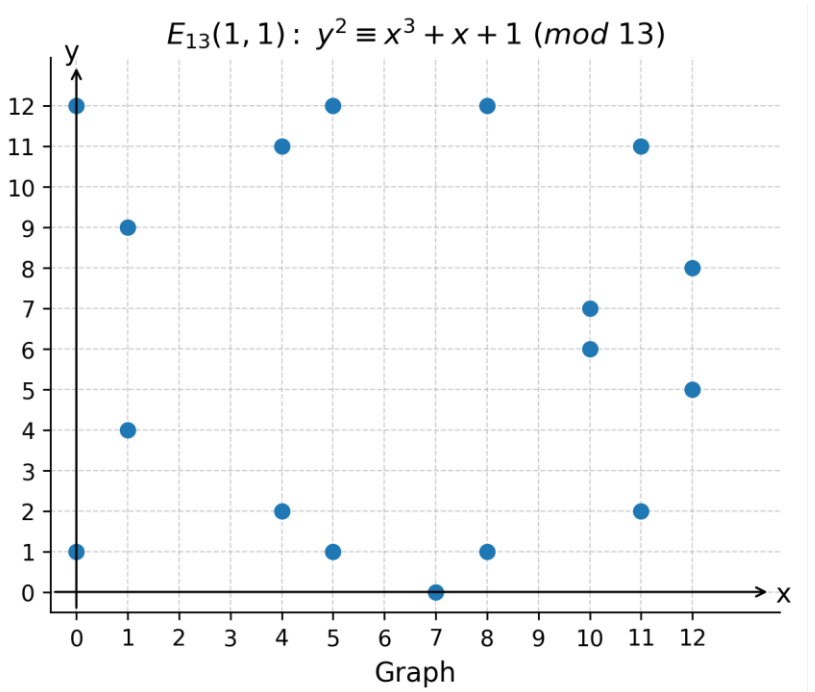
타원 곡선 기반 암호시스템

- $GF(p)$상의 타원 곡선 (2/4)
  - 예제 10.10

타원 곡선 $E_{13}(1,1)$: $y^2 = x^3 + x + 1\ (mod\ 13)$에 대해 $GF(13)$ 상의 곡선 위 점들을 구하고 그래프로 나타내시오.

  - 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)



타원 곡선 기반 암호시스템

- $GF(p)$상의 타원 곡선 (3/4)
  - 예제 10.10

타원 곡선 $E_{13}(1,1)$: $y^2 = x^3 + x + 1\ (mod\ 13)$에 대해 $GF(13)$ 상의 곡선 위 점들을 구하고 그래프로 나타내시오.

  - 사실
    a. 어떤 y^2값은 모듈로 13 상에서 제곱근을 갖지 못함
       - e.g., $x = 2, x = 3, x = 6, x = 9$인 점들은 곡선 상에 존재하지 않음
    b. 곡선에 정의된 모든 점은 역원을 가짐
       - e.g., $(1,4)$의 역원은 $(1,9)$
         - $4 + 9 = 13 \equiv 0\ mod\ 13$ 이므로 $(1,4) + (1,9) = O$
    c. 역원들은 그래프에서 같은 수직선 상에 나타남
       - e.g., $(1,4)$와 $(1,9)$는 x좌표가 모두 1이므로 같은 수직선 $x = 1$ 위에 존재함
         - $(0,1)$과 $(0,12)$, $(4,2)$와 $(4,11)$도 같은 방식으로 역원 관계임

타원 곡선 기반 암호시스템

- $GF(p)$상의 타원 곡선 (4/4)
  - 예제 10.11

두 점을 더해보자. $\mathrm{P} = (4, 2), \mathrm{Q} = (10, 6)$일 때 $\mathrm{R} = \mathrm{P} + \mathrm{Q}$를 구해보자.

    - 풀이
      - $\lambda = (6 - 2) \times (10 - 4)^{-1} mod\ 13 = 4 \times 6^{-1}\ mod\ 13 = 5\ mod\ 13$
      - $x = (5^2 - 4 - 10) mod\ 13 = 11\ mod\ 13$
      - $y = [5(4 - 11) - 2] mod\ 13 = 2\ mod\ 13$
      - $R = (11,\ 2)$ 이것이 곡선 상의 점임

타원 곡선 기반 암호시스템

- $GF(2^n)$상의 타원 곡선 (1/2)
  - $y^2 + xy = x^3 + ax^2 + b$를 만족하는 점들의 집합 $E_{2^n}(a, b)$
    - $b \neq 0$이고 x, y, a, b값들은 n 비트 단어를 표현하는 다항식임
  - 역원 구하기
    - $P = (x, y)$라면 $-P = (x, x + y)$임
  - 곡선 상의 점 찾기
    - 기약 다항식 $f(x)$를 이용하여 $GF(2^n)$의 원소를 표현함
      - e.g., $GF(2^3)$의 원소 표현
        - 기약다항식 $f(x) = x^3 + x + 1$ 사용
        - $f(g) = 0$ 이므로 $g^3 + g + 1 = 0$, 즉 $g^3 = g + 1$ 이를 이용해 g^3 이상의 항을 2차 이하 다항식으로 표현함

<$GF(2^3)$의 원소 표현 예시>

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

<곡선 위의 점 예시>

$(0,1)$	$(0,1)$
$(g^2, 1)$	(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)

타원 곡선 기반 암호시스템

- $GF(2^n)$상의 타원 곡선 (2/2)
  - 덧셈 연산
    1. 만약 $P = (x_1, y_1)$과 $Q = (x_2, y_2), Q \neq -P$이고 $Q \neq P$라면, $R = (x_3, y_3) = P + Q$는 다음과 같이 구할 수 있음

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

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \qquad y_3 = \lambda(x_1 + x_3) + x_3 + y_1$$

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

    2. 만약 $Q = P$라면 $R = P + P$(혹은 $R = 2P$)은 다음과 같이 구할 수 있음

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

$$x_3 = \lambda^2 + \lambda + a \qquad y_3 = {x_1}^2 + (\lambda + 1)x_3$$

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

타원 곡선 기반 암호시스템

- ElGamal에 타원 곡선 적용(EC-ElGamal) (1/5)
  - 정의
    - EC-ElGamal은 ElGamal 암호를 타원곡선 위의 점 연산으로 구현한 공개키 암호 방식임

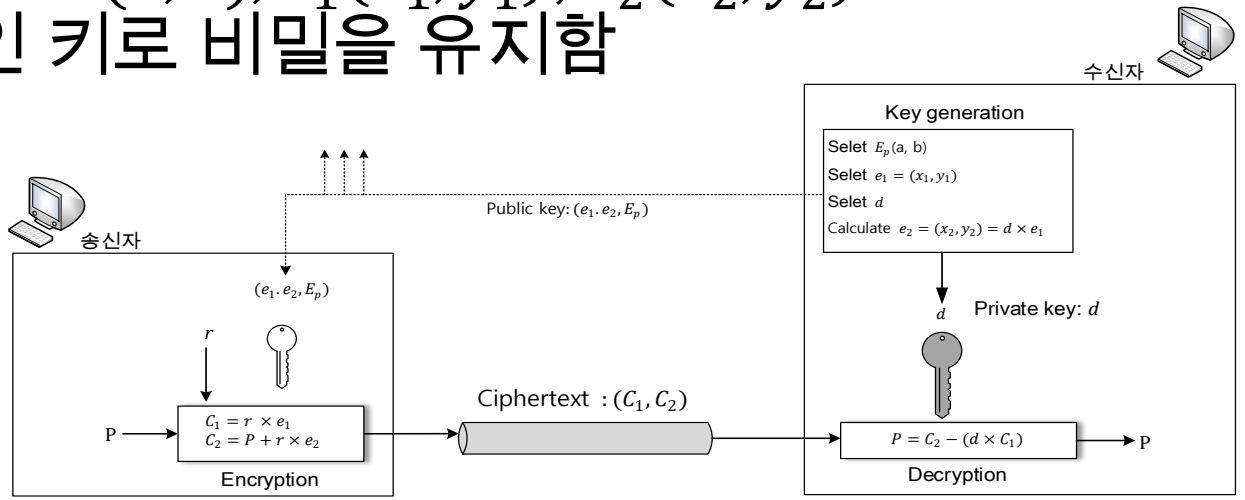
$$C_1 = r \times e_1 = rG$$
$$C_2 = P + r \times e_2 = P + rQ$$

  - 특징
    - EC-ElGamal은 타원곡선 위 점들의 덧셈군에서 동작함
    - 기존 ElGamal의 거듭제곱 연산은 스칼라 곱으로 대응됨
  - 키 구성
    - 공개 키: $\left(E_p(a,b), e_1, e_2\right)$, 개인키: d

*$e_1 = G$: 기준점
*$e_2 = Q = dG$: 공개키
*r: 암호화 시 사용하는 난수
*P:타원곡선 위의 점으로 표현된 평문

타원 곡선 기반 암호시스템

- ElGamal에 타원 곡선 적용 (2/5)
  - 키 생성
    1. 수신자는 $GF(p)$ 또는 $GF(2^n)$위에서 정의된 타원 곡선 $E(a,b)$를 선택함
    2. 수신자는 곡선상의 한점 $e_1 = (x_1, y_1)$와 d를 선택함
    3. 수신자는 $e_2(x_2,y_2) = d \times e_1(x_1, y_1)$을 계산함 (점을 여러 차례 더하는 것을 의미함)
    4. 수신자는 $E(a,b), e_1(x_1,y_1), e_2(x_2,y_2)$을 공개 키로 선언하고 d는 개인 키로 비밀을 유지함



타원 곡선 기반 암호시스템

- ElGamal에 타원 곡선 적용 (3/5)
  - 암호화
    - 평문에 해당되는 곡선 상의 한 점 P를 선택함
    - EC-ElGamal은 타원곡선 위의 점 덧셈을 사용하므로, 평문 점 P에 숨김점 $r \times e_2$를 더해 암호화함

$$C_1 = r \times e_1 \qquad C_2 = P + r \times e_2$$

  - 복호화
    - 수신자는 개인키 d를 이용해 C_1으로부터 $r \times e_2$를 계산함

$$d \times c_1 = d \times (r \times e_1) = r \times e_2$$

    - C_2에서 $r \times e_2$를 제거하여 평문 점 P를 복원함

$$P = C_2 - (d \times C_1)$$

      - 여기서 마이너스 역산은 더한다는 의미임

타원 곡선 기반 암호시스템

- ElGamal에 타원 곡선 적용 (4/5)
  - 예제 10.12

GF(p)상의 타원 곡선을 사용한 암호에 대한 예제를 살펴보자.

  - 키 생성
    - $GF(p)$상의 타원 곡선으로 $E_{67}(2,3)$을 선택함
    - $e_1 = (2, 22),\ d = 4$를 선택함
    - $e_2 = (13, 45)$를 선택함
      - 여기서 $e_2 = d \times e_1$임
    - 공개 키 (E, e_1, e_2)
  - 암호화
    - 평문 $P = (24, 26)$을 보내려고 하고, $r = 2$를 선택함
    - $C_1 = r \times e_1 = (35, 1)$
    - $C_2 = P + r \times e_2 = (21, 44)$

타원 곡선 기반 암호시스템

- ElGamal에 타원 곡선 적용 (5/5)
  - 예제 10.12

GF(p)상의 타원 곡선을 사용한 암호에 대한 예제를 살펴보자.

    - 복호화
      - 개인키 $d = 4$를 이용하여 $d \times C_1(35, 1) = (23,25)$를 계산함
      - $(23,25)$의 역원인 $(23,42)$를 구함
      - 암호문 $C_2 = (21, 44)$ 에 역원을 더해 평문을 복원함
        - $P = C_2 - (d \times C_1) = (21, 44) + (23, 42) = (24, 26)$
      - 따라서 원래 평문은 $P = (24, 26)$임

타원 곡선 기반 암호시스템

- 안전성
  - 타원 곡선 이산 로그 문제
    - 타원 곡선 위에서 점 곱셈을 계산하는 것은 쉽지만 계산된 결과만 보고 사용된 개인키나 난수 값을 역으로 찾는 것은 어려움
      - 풀 수 있는 유일한 방법은 Pollard p 알고리즘이 있지만 r값이 크고, $GF(2^n)$의 n이나 $GF(p)$의 p가 크다면 효과가 없음
    - 보안성은 타원 곡선 연산의 역추적이 어렵다는 점에 기반함
  - 키 길이
    - $GF(2n)$상의 ECC의 160비트는 RSA의 1024비트 크기의 모듈로가 제공하는 안전성과 동일함

Thanks!

김 민 식(minsik@pel.sejong.ac.kr)

부록 #1

- 공개키 암호 비교표

구분	RSA	Rabin	ElGamal	ECC
기반 난제	소인수분해	모듈러 제곱근/ 소인수분해	이산대수	타원곡선 이산대수
암호화 시간복잡도	$O(loge \cdot L^2)$	$O(L^2)$	$O(logr \cdot L^2)$	스칼라 곱 기반, $O(\log r)$번 점 연산
복호화 시간복잡도	$O(logd \cdot L^2)$	CRT 기반 제곱근 계산, RSA 복호화와 유사	$O(logd \cdot L^2)$	스칼라 곱 기반, $O(\log d)$번 점 연산
보안성	큰 n, 안전한 패딩 필요	복호화 후보 4개 문제 존재	r 재사용 금지	안전한 곡선, 난수 관리 필요
실생활 활용 예시	인증서, 전자서명, PKI	실무 사용은 거의 없고 이론·교육용	OpenPGP, 전자투표 일부	TLS의 ECDHE/ECDSA, 모바일 인증, 블록체인, 전자서명