암호화 알고리즘의 성능 분석
- 암호화 방식에 따른 속도의 차이를 중심으로

20124 윤영서
20208 노승연
20224 채윤호
20519 이준혁
20818 이수민

Ⅰ. 서론

1. 연구의 필요성 또는 동기

현재 중요한 정보를 암호화하거나 서로를 확인하기 위한 방법으로 소인수분해를 사용하는 RSA 암호화 알고리즘이 대표적으로 사용되고 있다. 최근에는 타원곡선을 사용하는 새로운 암호화 알고리즘인 ECC 알고리즘이 공개되었는데 우리는 두 알고리즘의 차이점과 장단점이 무엇인지 궁금해져 이를 분석하기로 하였다.

2. 연구 문제

가. 알고리즘에 따른 속도의 차이가 있는가?

나. 각 알고리즘의 속도에 영향을 미치는 요소는 무엇인가?

Ⅱ. 이론적 배경

1. 암호화 알고리즘

가. 인간이 알아볼 수 있는, 즉 암호화되지 않은 데이터를 평서문이라고 한다.

나. 암호화되어 원래 내용을 알아볼 수 없게 만든 데이터를 암호문이라고 한다.

다. 복호화란 암호문을 다시 평서문으로 바꾸는 것 혹은 바꾸는 과정을 뜻한다.

라. 양방향 암호화란 평서문을 암호화한 뒤 다시 복호화할 수 있는 암호화를 뜻한다.

마. 공개키 암호화란 암호화와 복호화에 사용되는 키가 다른 암호화 방식을 뜻한다. 비대칭키 암호화라고도 한다.

2. RSA 알고리즘

가. RSA 알고리즘은 큰 두 소수의 곱을 구하기는 쉽지만 반대로 큰 두 소수의 곱을 소인수분해 해 두 소수를 알아내기는 어렵다는 성질을 이용한 알고리즘이다.

나. RSA 알고리즘이 암호화되는 간략한 과정은 다음과 같다.

1 임의의 두 소수 p, q를 정하고 그 둘을 곱해 n을 구한다. (n = p * q)

2 1부터 n까지의 정수 가운데 n과 서로소인 원소의 개수를 나타내는 Φ(n) 함수의 값을 구한다. 이때 Φ(n) = (p - 1) * (q - 1) 이다.

3 1 < e < Φ(n) 이고 Φ(n)와 서로소인 e를 정한다.

4 a를 b로 나누었을 때 나머지를 나타내는 모듈러 연산을 (a mod b) 활용해 d를 구한다. 이때 (e * d) mod Φ(n) = 1 이다.

5 평서문을 M, 암호문을 C라고 하면 $C = M^e \bmod n$ 이다.

6 $M = C^d \bmod n$ 을 통해 복호화한다.

다. 간단한 예시를 들자면 다음과 같다.

1 p = 5, q = 17; n = 85

2 Φ(n) = (5 - 1) * (17 - 1) = 64

3 1 < e < 64이고 e와 64는 서로소이므로 e = 3

4 3d mod 64 = 1 이므로 d = 43

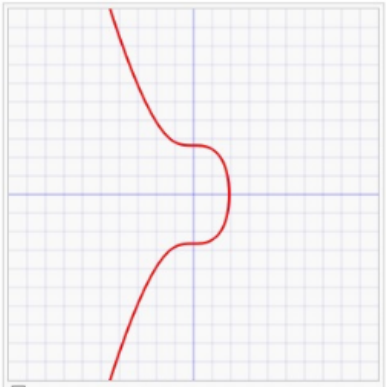
5 M = 15라고 했을 때, C = 15^3 mod 85 = 60

6 M = 60^{43} mod 85 = 15

3. ECC 알고리즘

가. ECC 암호화란 타원곡선 이론에 기반한 공개키 암호 방식으로, 암호화에 필요한 키는 정수여야 하므로 ECC 알고리즘에 사용되는 타원곡선 연산은 유한체 상에서 진행된다.

1 타원곡선 암호화의 대표적 예시



비트코인에 사용되는 **secp256k1curve ($y^2 = x^3 + 7$)**

2 타원곡선은 $y^2 = x^3 + ax + b$ 을 만족하는 (x,y)의 집합이다.

3 체란 대수적 구조 중 하나로 집합 내에서 덧셈. 뺄셈, 곱셈, 나눗셈의 사칙연산을 소화할 수 있는 집합이다. 체의 조건은 아래와 같다.

1) 집합의 원소로 연산하여 나온 결과가 집합의 원소이며 덧셈과 곱셈에 대하여 결합법칙과 교환법칙이 성립한다.

2) 임의의 원소에 또 다른 임의의 원소를 더했을 때 임의의 원소가 그대로 나오는 항등원이 존재하며 (2 + a = 2; a = 0; a는 항등원) 임의의 원소에 또 다른 임의의 원소를 곱했을 때 1이 나오는 역원이 (3 × a =1; a = $\frac{1}{3}$; a는 역원)이 존재한다.

4 표수란 유한체의 원소의 개수를 나타내는 수이다.

5 유한체(Z_p)란 유한개의 원소를 가지는 체이다. 정수 집합 내에서 체의 조건이 적용 되므로 원소의 개수는 유한하다. 유한체의 원소의 개수를 나타내는 표수는 반드시 소수이어야 한다. 표수가 소수이어야지만 역원이 존재한다.

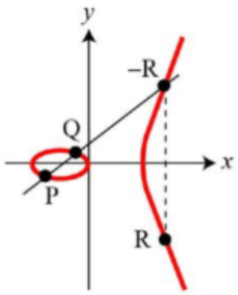
1) 표수가 소수가 아닌 경우의 예
표수(p)가 6일 때 Z_p={0, 1, 2, 3, 4, 5}
$(2 \times a) \bmod 6 = 1$을 만족하는 역원a가 집합 내에 존재하지 않는다.

6 난수란 정의된 범위 내에서 무작위로 만들어진 수를 말한다.

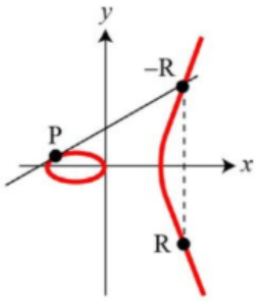
7 무한원점이란 직선이나 평면의 '끝'에 추가하는 가상의 점이다.

8 타원곡선의 연산

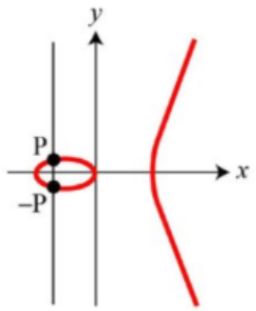
1) R=P+Q : P와 Q를 지나는 직선이 타원과 만나는 교점R을 x축으로 대칭시킨 점을 P+Q=R로 정의한다.



2) R=P+P : 덧셈 연산과 같이 P의 접점을 타원 곡선으로 이은 교점R을 x축으로 대칭시킨 점을 2P=R로 정의한다.



3) O=P+(-P) : P와 -P는 x축에 대칭되는 값이기 때문에 P+(-P)=0이다.



4) 타원 곡선 상의 덧셈 연산의 수학적 표시 (λ는 기울기)

- Addition $(P \neq Q)$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$
$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = (x_1 - x_3)\lambda - y_1$$

- Doubling $(P = Q)$

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

나. ECC알고리즘이 암호화되는 간략한 과정은 다음과 같다.

1 타원곡선 위의 임의의 점 G를 정한다. 이 점 G를 생성자라 칭한다.

2 임의의 수 k를 정한다. 이때 k는 비공개키가 된다.

3 타원곡선의 연산법을 이용하여 k * G를 구한다. 이때 kG는 공개키가 된다.

4 평서문 X(X_1, X_2)를 Y(Y_1, Y_2)로 암호화 할때 (r은 임의의 정수)
Y_1 = r * G; Y_2 = X + r * kG 이다.

5 암호문 Y(Y_1, Y_2)를 X(X_1, X_2)로 복호화 할때
X = Y_2 - (k * Y_1) 이다.

다. 간단한 예시를 들자면 다음과 같다.

1 G = (2, 7) 로 정한다.

2 k = 7 로 정한다.

3 kG = 7G; = 2G + 2G + 2G + G = (7, 2)

4 평서문 X(10, 9)를 Y(Y_1, Y_2)로 암호화하면, (r = 3)
Y_1 = 3G; Y_2 = X + 21G; Y(3G, X + 21G)

5 암호문 Y(3G, X + 21G)를 X(X_1, X_2)로 복호화하면,
X = X + 21G - (7 * 3G);
(이때 k(=7)이 없으면 21G를2 없애기 어렵다.)

Ⅲ. 연구 방법

1. 각 알고리즘의 암호화 과정을 키 생성을 위한 소수 준비, 공개키와 비밀키 생성, 암호화의 총 3개 과정으로 구분한다.

2. 암호화 코드를 세분화한 것에 따라 시간을 재는 코드를 삽입한다.

3. 각 코드를 10만 번 반복 실행시켜 걸린 시간의 평균값을 얻는다. 이때 반복실행 횟수를 10만 번으로 정한 근거는 횟수를 5만번부터 10만번까지 점차 올렸을 때 결괏값의 변동이 거의 없었기 때문에, 10만번 이상은 무의미한 것으로 판단하였다.

4. 위의 실험에서 얻을 결과를 바탕으로 알고리즘의 어떤 요소가 암호화 알고리즘의 성능에 어느 정도의 영향을 미치고 있는지 확인한다.

Ⅳ. 본론

1. RSA 데이터 분석

RSA 알고리즘은 키 생성을 위한 소수 준비 과정에서 0.000196 초를, 공개, 비밀키 생성 과정에서 0.220435초, 암호화 과정에서 0.000012초로 총 0.220644초가 소요되었다.

2. ECC 데이터 분석

ECC 알고리즘은 키 생성을 위한 소수 준비 과정에서 반올림 후 0초를, 공개, 비공개키 생성 과정에서 0.00198초, 암호화 과정에서 0.00001초로 총 0.00199초가 소요되었다.

3. 비교표

알고리즘 / 소요시간(초)	RSA	ECC
소수 준비	0.000196	0.000000
키 생성	0.220435	0.001980
암호화	0.000012	0.000010
합	0.220644	0.001990

Ⅴ. 결론

두 알고리즘 모두 총 암호화 시간 중에서 공개, 비밀키 생성 과정이 가장 큰 비율을 차지하였다. 이 과정에 비하면 나머지 소수 준비 과정과 암호화 과정은 무시할 수 있을 정도이다. RSA 암호화의 경우 매우 큰 소수 두 개의 소인수분해를 이용하는 데 반해 ECC 알고리즘은 미리 정의된 타원곡선을 이용하기 때문에 비교적 암호화에 사용되는 공개키와 비밀키 크기가 작고 이는 암호화 시간단축으로 이어진다. 따라서 ECC, RSA 알고리즘의 속도 차이는 암호화 방식에 따른 키의

크기의 차이 때문이다.
시간이 짧게 걸리므로 암호 해독에 필요한 시간도 짧아져 결국 보안성이 떨어지는 것 아닌가 하는 반론이 제기될 수 있으나 ECC 알고리즘은 여러 타원 곡선 방정식을 사용하여 공격자의 공격을 어렵게 한다. RSA 암호화의 경우 두 소수를 알아내면 암호가 풀리지만, ECC 암호의 경우 생성자, 유한체의 표수, 타원 곡선 방정식을 알아야 하므로 경우의 수가 기하급수적으로 늘어나 짧은 키로도 RSA 암호화와 동일한 수준의 보안 수준을 제공한다.

VI. 참고문헌

[암호학] 비대칭키-ECC, EIliptic Curve Cryptosystem, 티스토리, 2023.08.23. 인출

2023.08.22. 인출, 체(대수학), 나무위키

<정보통신기술용어해설>, 2023, http://www.ktword.co.kr/test/view/view.php?m_temp1=752
김현수•박석천, 2011, 음성 데이터 보안을 위한 효율적인 ECC 암호 알고리즘 설계 및 구현, <한국정보통신학회논문지>

최준백•신경욱, 2021, 타원곡선 기반 공개키 암호 시스템 구현을 위한 scalable ECC 프로세서, <한국정보통신학회논문지>

dev_mac-_-, "기초암호학(4) - ECC(타원곡선 암호화 알고리즘), 티스토리, 2019.04.13, https://developer-mac.tistory.com/83

2023년 8월 2일 인출, [암호학] 비대칭키 암호 - 공개 키 암호(Public-key cryptography) (tistory.com)

2023년 8월 2일 인출, [암호학] 비대칭키 암호 - RSA (공개키 암호시스템) (tistory.com)

2023년 8월 10일 인출, RSA 암호화 알고리즘 (1) - Mr.Soulware (mrsoulware.github.io)

2023년 8월 23일 인출, RSA암호의 원리 알아보기 | 보안 암호화 복호화 인수분해 (tistory.com)

2023년 8월 23일 인출, 톰 클란시스, 공개키 암호화 알고리즘과 원리에 대하여 RSA 공개키 암호화 알고리즘과 원리에 대하여 (tomclansys.com)