RSA의 설명

Seongjin Lee

May 20, 2019

1 RSA 단계 별 설명

이 문서는 RSA를 조금 더 쉽게 이해할 수 있도록 작성 되었다. RSA는 Rivest, Shamir, Adleman이라는 MIT 교수들의 이름의 앞 글자를 따서 만든 공개키 암호 알고리즘의 이름이다. RSA에는 크게 보면 5단계로 이루어져 있다.

1. 소수의 선정
2. 선정된 소수 2개의 곱으로 이루어진 n의 계산
3. $\Phi(n)$(Euler Totient Function, 파이 함수)의 계산
4. 암호를 위한 e의 계산
5. 복호를 위한 d의 계산

각 절에 단계를 이해하기 위해 필요한 부연 설명들을 달아 놓았다.

한 가지 기억하고 가야하는 중요한 사전 정보는 평문은 모두 십진수로 변환해야 한다는 것이다. 일반 글자들의 아스키 표현식을 따라 모두 변환할 수 있다. 아래의 소개된 모든 방법은 수체계를 따르기 때문에 숫자로 변환하지 않은 어떤 값도 사용할 수 없다.

2 두 개의 큰 소수의 선정

RSA가 제대로 동작하기 위해서는 매우 큰 두 개의 소수가 필요하다. 최근에 소수의 크기를 1024bit에서 4096bit로 키우는 사례들이 있다. 큰 소수를 구해야 하는 이유는 역 계산이 어렵도록 하기 위해서이다. 이에 대해서는 다음 단계에서 좀 더 알아보기로 하자. 여기에서는 예를 위해 두 개의 소수를 다음과 같이 정하기로 하자. 너무 큰 수를 사용하면 계산이 많아지니 핵심만 살펴보기 위해 매우 작은 두 수를 선정하였다.

$$p = 2 \tag{1}$$

$$q = 7 \tag{2}$$

2.1 n의 계산

이 단계에서 할 일은 앞서 얻은 p와 q의 곱을 구하는 것이다.

$$n = p \cdot q \tag{3}$$

$p = 2$, $q = 7$이므로 $n = 14$가 된다. p와 q가 충분히 크더라도 n을 구하는 것은 곱셈이므로 그렇게 어렵지 않다. 하지만, n을 알 때 역으로 p와 q를 구하는 것은 매우 어려운 일이 된다.

2.2 $\Phi(n)$의 계산

Euler Totient Function이라고도 알려진 $\Phi(n)$은 어떤 숫자들의 coprime(서로 소, relatively prime)의 갯수를 알려주는 함수이다.

$$\Phi(n) = (p-1)(q-1) \tag{4}$$

$$\Phi(n) = (2-1)(7-1) = 6 \tag{5}$$

이 식이 정말 동작하는지 예를 들어 살펴보자. 우리가 처음 구한 2와 7로 얻은 n은 14이다. 1부터 14까지의 수를 나열하면 다음과 같다.

$$1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14$$

이 중에서 14, 2, 7의 서로 소인 수들을 찾기 위해서는 2와 7로 나누어 떨어지는 수들을 모두 제거 하면 된다. 모두 제거하면 다음과 같은 수가 남는다.

$$1, 3, 5, 9, 11, 13$$

Coprime들의 수를 새면 6개가 나온다. 위에서 계산한 $\Phi(14) = 6$과 같은 결과이다.

2.3 e (ncryption)의 계산

이제 RSA 암호를 사용하기 위한 준비 작업은 끝이 난다. 정말 중요한 암호키를 생성할 차례이다. 암호키는 다음의 두 가지 성질을 만족해야 한다.

$$e = \begin{cases} 1 < e < \Phi(n) \\ Coprime\ with\ n,\ \Phi(n) \end{cases} \tag{6}$$

첫 번째 조건부터 살펴보자. e의 값의 범위는 다음과 같다. $1 < e < 6$ ($\because \Phi(14) = 6$) 그러므로 e로 사용할 수 있는 값은 다음 중 하나이다.

$$2, 3, 4, 5$$

두 번째 조건을 살펴보자. 14와 6의 서로 소인 수를 찾는 것이 할 일이다. 6의 공약수 인, 2, 3, 6은 제외가 되고 최종 후보는 5가 된다. 우리의 예는 매우 간단한 것이라 하나의 후보만 나왔다. 하지만, 정말 큰 p와 q를 사용하였다면 더 많은 후보가 남았을 것이다. 이렇게 얻은 $e = 5$라는 값과 $n = 14$라는 값을 쌍으로 하여 $(n, e) = (14, 5)$가 공개키가 된다.

2.4 d (ecryption)의 계산

복호화하기 위해 d를 계산해야 하는데, 이 때 사용하는 식은 다음과 같다.

$$(e \cdot d) mod \Phi(n) = 1 \tag{7}$$

$e = 5$라고 했으므로 $(5 \cdot d) mod 6 = 1$을 만족하는 d를 찾아야 한다. 이를 하나 씩 대입해서 계산해 보면 다음과 같다.

$$(5 \cdot 1) mod 6 = 5$$
$$(5 \cdot 2) mod 6 = 4$$
$$(5 \cdot 3) mod 6 = 3$$
$$(5 \cdot 4) mod 6 = 2$$
$$(5 \cdot 5) mod 6 = 1$$
$$(5 \cdot 6) mod 6 = 0$$
$$(5 \cdot 7) mod 6 = 5$$

식 2.4에서 식 7를 만족하는 값을 찾을 수 있다. 하지만, $d = 5$를 사용하는 경우 너무 간단하게 문제가 풀리기 때문에 훨씬 이후에 나오는 값을 사용하자. 여기서는 2 번째로 만족하는 값을 d로 사용하기로 하자. 그 다음으로 $(e \cdot d) mod n = 1$ 을 만족하는 값은 11이므로 $d = 11$로 사용하자.

이 과정을 큰 수에 대해서도 계산할 수 있는 간단한 방법을 알고자 한다면 Extended Eclidean Algorithm (Extended GCD algorithm, E-GCD)을 찾아 보기 바란다. E-GCD는 기본적인 Eclidean에서 $gcd(x, y) = (s)x + (t)y$를 만족하는 s와 t를 찾도록 하는 알고리즘이다. 이 때 s를 e라고 한다면 t가 d이 된다.

3 RSA 암/복호화 식과 증명

3.1 암호화 복호화 식

RSA 에서 사용하는 암호화 과정의 식은 다음과 같다.

$$c = m^e mod \phi(n) \tag{8}$$

이 때, c는 암호문이라고 하고 m은 평문이다. e와 $\Phi(n)$은 앞 장에서 설명을 하였다.

복호화를 위한 식은 다음과 같다.

$$m = c^d mod \phi(n) \tag{9}$$

여기서 d는 앞 장에서 설명하였다.

3.2 식의 성립 증명

다음 식의 전개를 통해 RSA가 동작하는 이유를 이해해 보자.

$$c^d mod n = (m^e mod n)^d mod n \tag{10}$$
$$= m^{ed} mod n \tag{11}$$
$$= m^{(edmodz)} mod n \tag{12}$$
$$= m^1 mod n \tag{13}$$
$$= m \tag{14}$$

식 12는 아래의 식과 같은 mod의 성질 때문에 그렇다.

$$\because x^y mod n = x^{(ymodz)} mod n \tag{15}$$