**RSA 알고리즘**

RSA는 공개키 암호화 방식의 알고리즘이다. 공개키 암호화 방식은 6가지 특성이 있다. 첫째 암호화와 해독에 이용도는 키는 달라야한다. 둘째 누구든지 수신자가 되려면 개인키가 필요하다. 셋째 누구든지 수신자가 되려면 암호화 키 즉 공개키를 공개해야한다. 넷째 공개 키가 수신자의 공개키가 맞는지에 관한 신빙성을 보증할 장치가 제공돼야 한다. 다섯째 암호문과 공개키를 알아도 평문을 구할 수 없어야 한다. 여섯째 공개키에 대한 지식을 바탕으로 개인키를 구할 수 없어야 한다. 공개키 암호화는 일방향 함수를 사용한다. 일방향 함수는 두 개의 큰 소수의 곱셈을 구하는 방식과 큰 모듈을 사용한 모듈러 거듭제곱 방식이 있다. 두 소수의 곱이 주어지면 두 소수가 무엇인지 알아내기 위한 인수분해 과정은 매우 어려우며 두 소수가 클수록 인수를 찾기가 힘들다. 두 소수는 비밀키며 소수의 곱은 공개키가 된다. 언젠가는 매우 큰 수를 쉽게 인수분해 하는 시대가 와 공개키 암호화 방식이 무용지물이 될 것이다.

**RSA 공개키 생성방법**

**1. 모듈러스 생성 :** n = p\*q 이 때 p와 q는 매우 큰 소수이고 n은 더 큰 숫자다. 결과값 n은 RSA 모듈러스라 부른다.

**2. e 생성 :** e는 아무 숫자나 될 수 없다. e는 1 보다 크고 (p-1)\*(q-1)보다는 작아야 하고 (p-1)(q-1)로 나누는 숫자가 없어야 한다. 예를들어 p =3, q = 7 일 경우 (p-1)(q-1) = 2 \* 6 = 12 이다. 이 때 e는 1 이외의 숫자여야 하며 12로 나눠지는 숫자는 안된다. e가 2와 3이라면 12의 인수여서 안되며 2와 3의 배수인 4, 6, 8, 9, 10도 안된다. 따라서 e는 1보다 크고 (p-1)(q-1)의 값인 12보다 작은 5,7,11 이 가능하다.

**3. 공개 키 생성 :** n, e는 RSA 공개키를 생성한다. 개인키를 가진 사람에게 암호화 메시지를 보내고 싶어하는 사람은 누구나 공개키를 구할 수 있다. 하지만 소수 p와 q는 절대 노출되선 안된다. 왜냐하면 두 소수의 곱은 인수분해하기 매우 어려운데 두 소수가 노출되면 쉽게 개인키를 알아낼 수 있다. 따라서 수신자 외에는 누구도 알 수 없어야 한다.

# 파이썬을 사용한 RSA 암호화 예제코드

from Crypto.Cipher import PKCS1\_OAEP

from Crypto.PublicKey import RSA

def rsa\_encrypt(msg):

public\_key = RSA.generate(1024) # 공개키 생성

private\_key = public\_key.privatekey() # 개인키 생성

cipher = PKCS1\_OAEP.new(public\_key)

encdata = cipher.encrypt(msg) # 공개키를 사용해 암호화한다.

print(f'암호화 결과 : {encdata}') # 암호문 출력

cipher = PKCS1\_OAEP.new(private\_key)

decdata = cipher.decrypt(encdata) # 개인키를 사용해 복호화한다.

print(f'복호화 결과 : {decdata}') # 평문 출력

def main():

msg = 'RSA is difficult'

rsa\_encrypt(msg.encode('utf-8')) # RSA 암호화 실행

main()