[2023 JBUCTF] crypto

show_me_the_flag

Write-Up

문제 개요

제공 파일 : show_me_the_flag.py

```python
from Crypto.Util.number import getPrime

flag = open('/flag', 'rb').read()

p , q = getPrime(1024), getPrime(1024)
n = p * q
n_1 = n - 1
phi = (p - 1) * (q - 1)
e = 0x10001
d = pow(e, -1, phi)

print(f'{n=}')
print(f'{e=}')

try:
    data1 = int(input('Input data1 : '))
    data2 = int(input('Input data2 : '))
    if  data1 < 2**1024 or data1 >= 2**2048:
        raise ValueError
    if  data2 < 2**1024 or data2 >= 2**2048:
        raise ValueError
except ValueError:
    print('Value Error')
    exit()
except Exception as e:
    print(e)
    exit()

enc1, enc2 = pow(data1, d, n), pow(data2, e, n)

if int(b'show_me_the_flag'.hex(), 16) == (enc1 * enc2) % n:
    print(f'flag : {flag.decode()}')
else:
    print('Falied')

```

RSA 개인키 (d, n), 공개키 (e, n)을 생성한 후, n과 e를 공개한다. 그리고

2**1024 보다 크거나 같고, 2**2048 보다 작은 두개의 정수 data1, data2를 입력 받는다.

입력 받은 정수 data1, data2로 enc1 = pow(data1, d, n), enc2 = pow(data2, e, n)을 계산한 후,

(enc1 * enc2) % n 이 b'show_me_the_flag'를 정수로 변환한 값과 같은 지 확인하고 같으면 flag를 출력한다.

문제 풀이

pow(data1, d, n) 같은 경우 data1^d mod n을 계산한다. 따라서 data1의 값을 a^e로 입력하게 되면

(a^e)^d mod n = a^ed mod n = a mod n 이므로 enc1의 값을 원하는 값으로 정해줄 수 있다.

RSA 암복호화 참고 : https://url.kr/56ngfw

pow(data2, e, n)은 data2^e mod n을 계산한다. d값을 모르므로 위와 같은 방법으로는 할 수 없다.

data2의 값을 n-1로 두면 (n-1)^e mod n을 계산하게 되는데 이때 e의 값이 홀수(65537) 이므로

(n-1)^e = n^e – n^(e-1) + … -1 이 되고, 이를 법을 n으로 해서 계산하면 -1 mod n = (n-1) mod n이 된다.

정리하자면 enc1의 값은 2**1024 보다 크거나 같고, 2**2048 보다 작은 정수 범위 안에서 값을 정할 수 있고, enc2는 n-1로 값을 정할 수 있다.

int(b'show_me_the_flag'.hex(), 16)를 A라고 할 때, A < n이고,

(enc1 * enc2) mod n = a * (n-1) mod n = A = A mod n

a * (n-1) mod n = A mod n에서 n-1, n, A의 값을 알고 있으므로 양 변에 법 n에 대한 n-1의 역원을 곱하면 a mod n = A * (n-1)^-1 mod n으로 a의 값을 구할 수 있다.

이제 구한 a 값을 a^e를 계산해서 data1에 입력하고, data2에는 n-1을 입력하면

if int(b'show_me_the_flag'.hex(), 16) == (enc1 * enc2) % n 이 참이 되어 flag를 얻을 수 있다.

data1, data2 를 D_1, D_2 라 하자.
그리고 int(b'show_me_the_flag'.hex(), 16) 를 A 라 하자.

$D_1 = \alpha^e$, $D_2 = n - 1$ 일 때, $A \equiv (D_1^d \ mod \ n) \ \times \ (D_2^e \ mod \ n)$ 라고 가정 하면,

$A \equiv (D_1^d \ mod \ n) \ \times \ (D_2^e \ mod \ n)$
$A \equiv ((\alpha^e)^d \ mod \ n) \ \times \ ((n-1)^e \ mod \ n)$
$A \equiv ((\alpha \ mod \ n) \ \times \ ((n-1) \ mod \ n)$
$\because \alpha^{ed} \ mod \ n \equiv \alpha \ mod \ n$ (rsa 암복호화 참고),
$e = 65537$ 이므로 홀수이다. 따라서 $(n-1)^e \ mod \ n = (n^e + a_{e-1}n^{e-1} + \ldots + a_1 n^1 + -1) \ mod \ n \equiv (-1) \ mod \ n \equiv (n-1) \ mod \ n$ 이다.

$A \ \times \ ((n-1)^{-1} \ mod \ n) \equiv \alpha \ mod \ n$
$(A \times (n-1)^{-1} \ mod \ n) \equiv \alpha \ mod \ n$

$\therefore \alpha \ mod \ n \equiv (A \times (n-1)^{-1} \ mod \ n)$

exploit.py

```python
from pwn import *

p = remote('172.17.0.2', 10001)

n = int(p.recvline()[2:-1].decode())
e = int(p.recvline()[2:-1].decode())
c = int(b'show_me_the_flag'.hex(), 16)

data1 = (c * pow(n-1, -1, n)) % n
data1 = pow(data1, e, n)
data2 = n - 1

p.sendafter(b'Input data1 : ', str(data1) + '\n')
p.sendafter(b'Input data2 : ', str(data2) + '\n')

print(p.recvline().decode())

```

FLAG

scpCTF{9f83e4df3015a9a69af4d3e77bcbb0e8c6e075d0b657}