从XCTF best_rsa 学习RSA共模攻击

原创

开火箭的β 于 2020-10-06 16:32:32 发布 253 收藏 1
分类专栏： 密码学 文章标签： 密码学

0x00 RSA原理

明文为m，密文为c
模n = p*q
欧拉函数值φ(n)，φ(n)=(p-1)(q-1)
公钥参数e和私钥参数d，可由欧拉函数值计算出，e≡d^-1 (mod φ(n))；
加密：m^e ≡ c (mod n)
解密：c^d ≡ m (mod n)

0x01 共模攻击的由来

所谓共模，就是明文m相同，模n相同，用两个公钥e1,e2加密得到两个私钥d1,d2和两个密文c1,c2
共模攻击，即当n不变的情况下，知道n,e1,e2,c1,c2 。可以在不知道d1,d2的情况下，解出m。
这里有个条件，即

```
gcd(e1,e2)=1
```

0x02 共模攻击原理

有整数s1,s2(一正一负）

```
e1*s1+e2*s2 = 1
```

根据扩展欧几里德算法，我们可以得到该式子的一组解（s1,s2），假设s1为正数,s2为负数。

0x02_1 欧几里得算法

要了解扩展欧几里得算法，最好先知道欧几里得算法
欧几里得算法：

```
d = gcd(a,b)
d = gcd(b,a mod b) //这里假设a>b
gcd(a,b) = gcd(b,a mod b)
上面就是一次辗转相除
一直辗转相除下去，可得：
gcd(a,b) = gcd(b,a mod b) = ... = gcd(m,0)
其中m为最大公约数
```

用例：

```
gcd(21,15)
= gcd(15,6)
= gcd(6,3)
= gcd(3,0)
即可得21与15的最大公约数为3
```

0x02_2 扩展欧几里得算法

```
对于不完全为 0 的非负整数 a，b
有gcd（a，b）
必然存在整数对 x，y ，使得 gcd（a，b）=a*x+b*y。
```

用代码表示：

```
def egcd(a, b):
  if a == 0:
    return (b, 0, 1)
  else:
    g, y, x = egcd(b % a, a)
    return (g, x - (b // a) * y, y)
```

因为

c1 = m^e1%n
c2 = m^e2%n
所以

$(c1^{s1*c2}s2)\%n = ((m^{e1\%n})s1*(m^{e2\%n})s2)\%n$
化简得
c1^s1 * c2^s2 = m

一个数的负次幂

在数论模运算中，要求一个数的负数次幂，与常规方法并不一样。

比如此处要求c2的s2次幂，就要先计算c2的模反元素c2r，然后求c2r的-s2次幂
所以我们有以下脚本：

0x03 解题脚本

```python
from Crypto.PublicKey import RSA
from Crypto.Util.number import *
from gmpy2 import *

def egcd(a, b):
  if a == 0:
    return (b, 0, 1)
  else:
    g, y, x = egcd(b % a, a)
    return (g, x - (b // a) * y, y)

f1 = open('publickey1.pem','rb').read()
pub1 = RSA.importKey(f1)

n = int(pub1.n)
e1 = int(pub1.e)

f2 = open('publickey2.pem','rb').read()
pub2 = RSA.importKey(f2)

e2 = int(pub2.e)

c1 = open('cipher1.txt','rb').read()
c1 = bytes_to_long(c1)
print(c1)

c2 = open('cipher2.txt','rb').read()
c2 = bytes_to_long(c2)
print(c2)

print(gcd(e1,e2))

s = egcd(e1, e2)
s1 = s[1]
s2 = s[2]

if s1 < 0:
    s1 = - s1
    c1 = invert(c1, n)
elif s2 < 0:
    s2 = - s2
    c2 = invert(c2, n)

m = pow(c1,s1,n)*pow(c2,s2,n) % n
print (long_to_bytes(m))
```