# RSA由密文解密的奇偶性确定原始明文

原创

M3ng@L 于 2022-03-28 22:18:06 发布 378 收藏

分类专栏： CTF比赛复现 文章标签： python Crypto



本文链接：https://blog.csdn.net/qq_51999772/article/details/123808105



CTF比赛复现 专栏收录该内容

31 篇文章 0 订阅

订阅专栏

## RSA由密文解密的奇偶性确定原始明文

*keywords*: `XCTF 4th-QCTF-2018-babyrsa`

## *Problme*

```
'''
Baby RSA
e = 0x10001
n = 0x0b765daa79117afe1a77da7ff8122872bbcbddb322bb078fe0786dc40c9033fadd639adc48c3f2627fb7cb59bb0658707fe516967464439bdec2d6479fa3745f57c0a5ca255812f0884978b2a8aaeb750e0228cbe28a1e5a63bf0309b32a577eecea66f7610a9a4e720649129e9dc2115db9d4f34dc17f8b0806213c035e22f2c5054ae584b440def00afbccd458d020cae5fd1138be6507bc0b1a10da7e75def484c5fc1fcb13d11be691670cf38b487de9c4bde6c2c689be5adab08b486599b619a0790c0b2d70c9c461346966bcbae53c5007d0146fc520fa6e3106fbfc89905220778870a7119831c17f98628563ca020652d18d72203529a784ca73716db
c = 0x4f377296a19b3a25078d614e1c92ff632d3e3ded772c4445b75e468a9405de05d15c77532964120ae11f8655b68a630607df0568a7439bc694486ae50b5c0c8507e5eecdea4654eeff3e75fb8396e505a36b0af40bd5011990663a7655b91c9e6ed2d770525e4698dec9455db17db38fa4b99b53438b9e09000187949327980ca903d0eef114afc42b771657ea5458a4cb399212e943d139b7ceb6d5721f546b75cd53d65e025f4df7eb8637152ecbb6725962c7f66b714556d754f41555c691a34a798515f1e2a69c129047cb29a9eef466c206a7f4dbc2cea1a46a39ad3349a7db56c1c997dc181b1afcb76fa1bbbf118a4ab5c515e274ab2250dba1872be0
'''
```

提供了`nc`链接地址，简单把`c`传入得到

```
C:\Users\Menglin>nc 111.200.241.244 62431
---------------------------- baby rsa -----------------------------
Come and Decode your data
If you give me ciphertext, I can tell you whether decoded data is even or odd
You can input ciphertext(hexdecimal) now
0x4f377296a19b3a25078d614e1c92ff632d3e3ded772c4445b75e468a9405de05d15c77532964120ae11f8655b68a630607df0568a7439b
c694486ae50b5c0c8507e5eecdea4654eeff3e75fb8396e505a36b0af40bd5011990663a7655b91c9e6ed2d770525e4698dec9455db17db3
8fa4b99b53438b9e09000187949327980ca903d0eef114afc42b771657ea5458a4cb399212e943d139b7ceb6d5721f546b75cd53d65e025f
4df7eb8637152ecbb6725962c7f66b714556d754f41555c691a34a798515f1e2a69c129047cb29a9eef466c206a7f4dbc2cea1a46a39ad33
49a7db56c1c997dc181b1afcb76fa1bbbf118a4ab5c515e274ab2250dba1872be0
odd
```

# *Analysis*

提供的`n,c,e`是正常的RSA的公钥以及密文；那么突破重点便是`nc`链接服务端的作用

> If you give me ciphertext, I can tell you whether decoded data is even or odd

以上是服务端脚本的作用解释，其作用就是把使用本题中的`(n,e)`作为公钥生成的密文解密得到的数值判断奇偶性并返回

但是这里关于`(n,e)`生成的密文只有`c`，我们怎么使用这个条件呢

在模`n`的条件下改变`c`的大小，并且能够改变其奇偶性：$c=$

上述等式的作用是**解出的明文相当于原来明文的两倍**

$$c \equiv$$

如果奇数乘以$2$，结果一定为偶数；但是如果是在模$n$的情况下则不一定，当

$$(c \cdot 2 ) =$$

这样区别开来的作用是

当传入

$c \equiv$

当传入

$c \equiv$

---

当计算完$c$之后，在$c$的基础上继续进行

$c =$

并且缩小的范围相当于

注意一点是这里服务端的脚本将传入的数值进行了转换，可能是这样的

```
input = int(input,16)
```

那么就意味着传入的数值`c`必须是`hex`十六进制形式的

怎么看出来的呢？因为

```
>>>int("0x4f377296a19b3a25078d614e1c92ff632d3e3ded772c4445b75e468a9405de05d15c77532964120ae11f8655b68a630607df05
68a7439bc694486ae50b5c0c8507e5eecdea4654eeff3e75fb8396e505a36b0af40bd5011990663a7655b91c9e6ed2d770525e4698dec945
5db17db38fa4b99b53438b9e09000187949327980ca903d0eef114afc42b771657ea5458a4cb399212e943d139b7ceb6d5721f546b75cd53
d65e025f4df7eb8637152ecbb6725962c7f66b714556d754f41555c691a34a798515f1e2a69c129047cb29a9eef466c206a7f4dbc2cea1a4
6a39ad3349a7db56c1c997dc181b1afcb76fa1bbbf118a4ab5c515e274ab2250dba1872be0",16)
10000168328337586723723875929199426704755546201195615734523254693814039765906148543232611129765509088000736956851520022537085658495744583517648184152998168309758485546037306528320166722634787751788563573939958617061265638476529095277032847794703409537505203594204567151328850116513909550356213958708966751554276850944050261394701167302379584172517413175668997227366580985191901168830374739189734903482208079680206788635600160218616793118994764212910484101984483101655772133593536065667175866557356425850372486695596296455668658676150338420608042931515767744544733279970819343643596659567320755262783035393928843439072
---
C:\Users\Menglin>nc 111.200.241.244 62431
---------------------------- baby rsa -----------------------------
Come and Decode your data
If you give me ciphertext, I can tell you whether decoded data is even or odd
You can input ciphertext(hexdecimal) now
10000168328337586723723875929199426704755546201195615734523254693814039765906148543232611129765509088000736956851520022537085658495744583517648184152998168309758485546037306528320166722634787751788563573939958617061265638476529095277032847794703409537505203594204567151328850116513909550356213958708966751554276850944050261394701167302379584172517413175668997227366580985191901168830374739189734903482208079680206788635600160218616793118994764212910484101984483101655772133593536065667175866557356425850372486695596296455668658676150338420608042931515767744544733279970819343643596659567320755262783035393928843439072
even
```

得到的结果与传入十六进制时不同，所以得到上述结果

而Python会自动把十六进制转换为十进制再进行进一步操作，所以最后上传时需要再一次手动转换十六进制才行

```
>>> c = 0x4f377296a19b3a25078d614e1c92ff632d3e3ded772c4445b75e468a9405de05d15c77532964120ae11f8655b68a630607df05
68a7439bc694486ae50b5c0c8507e5eecdea4654eeff3e75fb8396e505a36b0af40bd5011990663a7655b91c9e6ed2d770525e4698dec945
5db17db38fa4b99b53438b9e09000187949327980ca903d0eef114afc42b771657ea5458a4cb399212e943d139b7ceb6d5721f546b75cd53
d65e025f4df7eb8637152ecbb6725962c7f66b714556d754f41555c691a34a798515f1e2a69c129047cb29a9eef466c206a7f4dbc2cea1a4
6a39ad3349a7db56c1c997dc181b1afcb76fa1bbbf118a4ab5c515e274ab2250dba1872be0
>>> c
10000168328337586723723875929199426704755546201195615734523254693814039765906148543232611129765509088000736956851520022537085658495744583517648184152998168309758485546037306528320166722634787751788563573939958617061265638476529095277032847794703409537505203594204567151328850116513909550356213958708966751554276850944050261394701167302379584172517413175668997227366580985191901168830374739189734903482208079680206788635600160218616793118994764212910484101984483101655772133593536065667175866557356425850372486695596296455668658676150338420608042931515767744544733279970819343643596659567320755262783035393928843439072
```

所以我们需要

```
c = 0x...
io.sendline(hex(c))
```

另外由于时常有网络波动（本题需要不断重复链接服务端），所以加入了`try,except`语句保证因为网络波动断开了连接之后继续进行连接计算

## *Solving code*

```python
from Crypto.Util.number import *
import gmpy2
from pwn import *
e = 0x10001
n = 0x0b765daa79117afe1a77da7ff8122872bbcbddb322bb078fe0786dc40c9033fadd639adc48c3f2627fb7cb59bb0658707fe5169674
64439bdec2d6479fa3745f57c0a5ca255812f0884978b2a8aaeb750e0228cbe28a1e5a63bf0309b32a577eecea66f7610a9a4e720649129e
9dc2115db9d4f34dc17f8b0806213c035e22f2c5054ae584b440def00afbccd458d020cae5fd1138be6507bc0b1a10da7e75def484c5fc1f
cb13d11be691670cf38b487de9c4bde6c2c689be5adab08b486599b619a0790c0b2d70c9c461346966bcbae53c5007d0146fc520fa6e3106
fbfc89905220778870a7119831c17f98628563ca020652d18d72203529a784ca73716db
c = int("0x4f377296a19b3a25078d614e1c92ff632d3e3ded772c4445b75e468a9405de05d15c77532964120ae11f8655b68a630607df0
568a7439bc694486ae50b5c0c8507e5eecdea4654eeff3e75fb8396e505a36b0af40bd5011990663a7655b91c9e6ed2d770525e4698dec94
55db17db38fa4b99b53438b9e09000187949327980ca903d0eef114afc42b771657ea5458a4cb399212e943d139b7ceb6d5721f546b75cd5
3d65e025f4df7eb8637152ecbb6725962c7f66b714556d754f41555c691a34a798515f1e2a69c129047cb29a9eef466c206a7f4dbc2cea1a
46a39ad3349a7db56c1c997dc181b1afcb76fa1bbbf118a4ab5c515e274ab2250dba1872be0",16)
m_upper = n
m_lower = 0
while True:
    try:
        io = remote("111.200.241.244","62431")
        io.recvuntil(b"now")
        c = (c * pow(2,e,n)) % n
        io.sendline(hex(c))
        output = io.recvall().strip().decode()
        if output == "even":
            m_upper = (m_upper + m_lower) // 2
        elif output == "odd":
            m_lower = (m_upper + m_lower) // 2
        print(m_upper)
        if m_upper == m_lower:
            print(long_to_bytes(m_upper))
            print(long_to_bytes(m_lower))
            break
    except:
        io = remote("111.200.241.244","62431")
        io.recvuntil(b"now")
        c = (c * pow(2,e,n)) % n
        io.sendline(hex(c))
        output = io.recvall().strip().decode()
        if output == "even":
            m_upper = (m_upper + m_lower) // 2
        elif output == "odd":
            m_lower = (m_upper + m_lower) // 2
        print(m_upper)
        if m_upper == m_lower:
            print(long_to_bytes(m_upper))
            break
```

当然也可以`def`解密函数，代码会更简洁一些

# *Reference*


QCTF 2018线上赛 writeup - 蝉时雨 - 博客园 (cnblogs.com)

XCTF-4th-QCTF-2018-babyrsa - 芒果的小站 (mangofeng.cn)