

Whctf 2017 -UNTITLED- Writeup

转载

[weixin_30786617](#) 于 2017-09-18 09:11:00 发布 78 收藏

文章标签: [python](#)

原文链接: <http://www.cnblogs.com/WangAoBo/p/7541481.html>

版权

Whctf 2017 -UNTITLED- Writeup

转载请表明出处<http://www.cnblogs.com/WangAoBo/p/7541481.html>

分析:

下载下来的附件是一个py脚本, 如下

```

1 from Crypto.Util.number import getPrime,long_to_bytes,bytes_to_long
2 import primefac
3 import time
4 from os import urandom
5 import hashlib
6 import sys
7 class Unbuffered(object):
8     def __init__(self, stream):
9         self.stream = stream
10    def write(self, data):
11        self.stream.write(data)
12        self.stream.flush()
13    def __getattr__(self, attr):
14        return getattr(self.stream, attr)
15 import sys
16 sys.stdout = Unbuffered(sys.stdout)
17 def gen_args():
18     p=getPrime(1024)
19     q=getPrime(1024)
20     n=p*q
21     e=0x10001
22     d=primefac.modinv(e,(p-1)*(q-1))%((p-1)*(q-1))
23     return (p,q,e,n,d)
24 def proof():
25     salt=urandom(4)
26     print salt.encode("base64"),
27     proof=raw_input("show me your work: ")
28     if hashlib.md5(salt+proof.decode("base64")).hexdigest().startswith("0000"):
29         print "checked success"
30         return 1
31     return 0
32
33 def run():
34     if not proof():
35         return
36     m=int(open("/home/bibi/PycharmProjects/work/whctf/flag", "r").read().encode("hex"),16)#flag{*}
37     (p,q,e,n,d)=gen_args()
38     c=pow(m,e,n)
39     print "n:",hex(n)
40     print "e:",hex(e)
41     print "c:",hex(c)
42     t=int(hex(m)[2:][0:8],16)
43     u=pow(t,e,n)
44     print "u:",hex(u)
45     print "===="
46     x=int(hex(m)[2:][0:8]+raw_input("x: "),16)
47     print "===="
48     y=int(raw_input("y: "),16)
49     if (pow(x,e,n)==y and pow(y,d,n)==t):
50         print "s:",hex(int(bin(p)[2:][0:568],2))
51 run()

```

nc连上之后如下:

```
{8:41}~/Desktop/whctf/untitled ▸ nc 118.31.18.75 20013
+/DIHw== IS Learning IS Tools TrickyPC Tools Programming
show me your work: 
首页 公告 赛题 排行
```

结合py代码分析，首先要通过**proof()**函数的验证，即要满足：

```
hashlib.md5(salt+proof.decode("base64")).hexdigest().startswith("0000")
```

写一个脚本爆破，使盐值和输入内容的base64解码的md5开头四位为0000

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
__Author__ = 'M4x'

import hashlib
import string

#base64编码后的范围
dic = string.ascii_letters + string.digits + "+/"

#盐值
salt = '+/DIHw=='.decode('base64')

#先尝试爆破4位
for a in dic:
    for b in dic:
        for c in dic:
            for d in dic:
                proof = a + b + c + d
                try:
                    if hashlib.md5(salt + proof.decode("base64")).hexdigest().startswith("0000"):
                        print proof
                        exit(0)
                except:
                    pass
```

很快就爆破出多组结果

```
{8:42}~/Desktop/whctf/untitled ↵ python md5.py
ahTh
aR1n
aYh4
bnYG
bpdY
bUy0
bUEU
bY7i
ck40 006AWYXBly1fjnf8608y0j30eh02rjrp.jpg 复制
cAu+
cGbs
cI51
cUFs
dh+V large/006AWYXBly1fjnf8608y0j30eh02rjrp.jpg 复制
dnv9
dJ0t
cn/large/006AWYXBly1fjnf8608y0j30eh02rjrp.jpg 复制
```

随便找一组提交，通过了proof函数的验证，得到了n, e, c, u

```
{8:41}~/Desktop/whctf/untitled ↵ nc 118.31.18.75 20013
+/DIHw== IS Learning IS Tools TrickyPC Tools Programming CTF OJ About Me
show me your work: ahTh
checked success 首页 公告 赛题 排行
n: 0x621725fc8ce7ce38c3ff9da9e7d4a9d8764eac78985f5abc52bbad15f172d76c0d9cc
4b08b1bbcd36590bc0050ab492f7df58404c0bca8b178e7e0f07c0c08e46ae63d824861f1cd
d3f6cfed6fcc348b62e1cb7b269fc800c77d303ae154e1ade78a7492158c80818b8b180699e
709764d31e08544e9c6dd75788d468ce1288927d5cea4336d6a76a9998731e15285c4695550
c4db7210d09168903774ccee5dda6f8d3a502f8eac38a97c0cd84b3c3be87751dfc9f3bbcbde
c881d20fc7cb0086f71a0146b2e11e688372f809e401b9f19c003f75920df962631127dbda8
4cc781870b7895382c02d726eabc8373e73aec38f0a1dad4b8d0060c47511ef75d3L
e: 0x10001
c: 0x207b6655efcca22c8ffff836c81c7400d23df0d6121d9f3ed349c8ca00acd9b3464ae17
cdcfc1f64bab3f73ce10596527f20f60823e227edd12cbef4667bd8a7fd3318dfb78ad51de4
5228aa5e2db0ecf57eb8b839c643760abf760e223cc25fa1a55320ddd5c9407a6d1edc562f6
2176535f8ce619e2b9b33ee1b49bae1a398e9c02f947ad2c6d48a9a4681f9bb518280ad298b
8bf7e68a6580f6f66e14445a637429805b09926bb6d17581dcf9857828423e3b1dfe6f00450
f1da63b9bb709e456569e3f08879393de2544ff883aa7743adb669f5dccb9059ca494692d4a
ddd2c7619db0b07e08d6a3d6bf18897da74653cbbd0e84012f15a6c7b675a2d07f4L
u: 0x53442deb1908191ef5f42aef38f7d1903960cbcec70d0c9b3289bbd612bd7b6c01ce90
48e5c06549a04167beadec7028985174b476b61a1491729aee60b7499dc73dee360fdcf350c
6bcf0db780932ad8440138408a746b8491a0fca767a1135b370192b89caecb6f1dc1cf83be5
4d11a1e7a53f298c3bb799269239b68f6525fa59bc39ae3b1f1b309118d595d057a5ee07f92
2291d018a39c65fd90e7eb3ed2e73949aee4b583d375dff0ed32163f33087ec692675ed5d5d
355bfae791d04317c82005901b5adf45315d26e2e828e92594625b7b702c65e23c41d35da6f
2ff781c3fb7f24866199d0c057d7d7b5cf3d59aea968602b6c92f65c29ac4e293eeL
====
X: Nu1L
```

再分析题目给的py脚本，可以看出

c即为**flag**经过**RSA**加密的密文，其中**n**和**e**已知

u为**m**的前8位（根据**flag**形式，即为**f**）经过**RSA**加密后的值

```

x=int(hex(m)[2:][0:8]+raw_input("x: "),16)

y=int(raw_input("y: "),16)

pow(x,e,n)==y and pow(y,d,n)==t

```

以上三行连起来分析，很容易得出当我们输入的x为空，y为u时，即可通过最后一步if的验证，从而得到p的前568位（输入y时记得去掉最后的L）

```

c: 0x207b6655efcca22c8fff836c81c7400d23df0d6121d9f3ed349c8ca00acd9b3464ae17
cdcfc1f64bab3f73ce10596527f20f60823e227edd12cbef4667bd8a7fd3318dfb78ad51de4
5228aa5e2db0ecf57eb8b839c643760abf760e223cc25fa1a55320ddd5c9407a6d1edc562f6
2176535f8ce619e2b9b33ee1b49bae1a398e9c02f947ad2c6d48a9a4a81f9bb518280ad298b
8bf7e68a6580f6f66e14445a637429805b09926bb6d17581dcf9857828423e3b1dfe6f00450
f1da63b9bb709e456569e3f088879393de2544ff883aa7743adb669f5dccf9059ca494692d4a
ddd2c7619db0b07e08d6a3d6bf18897da74653cbbd0e84012f15a6c7b675a2d07f4L
u: 0x53442deb1908191ef5f42aef38f7d1903960cbcec70d0c9b3289bbd612bd7b6c01ce90
48e5c06549a04167beadec7028985174b476b61a1491729aee60b7499dc73dee360fdcf350c
6bcf0db780932ad8440138408a746b8491a0fca767a1135b370192b89caecb6f1dc1cf83be5
4d11a1e7a53f298c3bb799269239b68f6525fa59bc39ae3b1f1b309118d595d057a5ee07f92
2291d018a39c65fd90e7eb3ed2e73949aee4b583d375dff0ed32163f33087ec692675ed5d5d
355bfae791d04317c82005901b5adf45315d26e2e828e92594625b7b702c65e23c41d35da6f
2ff781c3fb7f24866199d0c057d7d7b5cf3d59aea968602b6c92f65c29ac4e293eeL
====
x:
====
y: 0x53442deb1908191ef5f42aef38f7d1903960cbcec70d0c9b3289bbd612bd7b6c01ce90
48e5c06549a04167beadec7028985174b476b61a1491729aee60b7499dc73dee360fdcf350c
6bcf0db780932ad8440138408a746b8491a0fca767a1135b370192b89caecb6f1dc1cf83be5
4d11a1e7a53f298c3bb799269239b68f6525fa59bc39ae3b1f1b309118d595d057a5ee07f92
2291d018a39c65fd90e7eb3ed2e73949aee4b583d375dff0ed32163f33087ec692675ed5d5d
355bfae791d04317c82005901b5adf45315d26e2e828e92594625b7b702c65e23c41d35da6f
2ff781c3fb7f24866199d0c057d7d7b5cf3d59aea968602b6c92f65c29ac4e293ee
s: 0xb447abcd768378f05675b98f4724e934b1a7251749b14b11d3af19d3a47e98dbf90b94
a77a01ab76e6a7f99d5b79cfce8e9edfcc7b626ed0f1699d743fa78bd73ff4a03f904bdeL

```

至此整理一下我们得到的信息：

- flag加密后的密文c
- 加密所用到的n和e
- p的前568位

很容易联想到恢复p，从而算出q，再解RSA就能拿到flag

步骤：

这个时候想到了国赛的一道类似题目**Partial**，也是知道n，e和高位p，需要恢复p，因此也选用相同的方法**Coppersmith Attack**(<https://github.com/Gao-Chuan/RSA-and-LLL-attacks#factoring-with-high-bits-known>)，但**Coppersmith Attack**方法需要我们最少知道576位p，已知568位，差了二个16进制数，根据官方给的hint

python 解RSA的姿势 (<http://www.cnblogs.com/WangAoBo/p/7513811.html>)

运行，即可得到flag

```
{9:06}~/Desktop/whctf/untitled ➤ python rsa.py  
flag{rs4_y0ok_s0_m2ch_1n_c7f_qu4ls_c0mp7t1t10n}  
{9:06}~/Desktop/whctf/untitled ➤
```

转载于:<https://www.cnblogs.com/WangAoBo/p/7541481.html>