# A立方CTF部分wp

原创

WustHandy 于 2021-01-10 18:00:48 发布 464 收藏

分类专栏： WriteUp 文章标签： crypto



本文链接：https://blog.csdn.net/weixin_45883223/article/details/112427962





## A立方CTF部分wp



## easy

修复zip头部，stegdetect，发现是jphide，key是另一个文件里的md5在线网站解密

## cpt1

把key用base64解密，再维吉尼亚，再base32

## cpt2

```
from Crypto.Util.number import *
import gmpy2

msg1 = '**********************************'
msg2 = '**********************************'
hex_msg1=int(msg1.encode("hex"),16)
hex_msg2=int(msg2.encode("hex"),16)

p=getPrime(512)
q=getPrime(512)
n1=p*q
e1=0x10001
e2=getPrime(10)
n2=n1
c1=pow(hex_msg1,e1,n1)
c2=pow(hex_msg2,e2,n2)

phi=(p-1)*(q-1)
d1=gmpy2.invert(e1,phi)
print("d1=",hex(d1),"e1=",hex(e1),"n1=",hex(n1),"c1=",hex(c1))
'''
('d1=', '0x7d12e57b1aa157038ebe5c45b56256270671e6984b0dcdf10a2ea07ce480143240c9a3e1c60870e499306a717073f157476aa
88e99a7bdf1e2a4adf8ce21025cc6c05035c4a1d7e3b6f061464872e65118384999f0154f3c1761fa68d4685126b7fc98f4c2cdc41c98aa4
e099a868a89099dd2170664647efca2c8d8e06a2e49',
'e1=', '0x10001',
'n1=', '0x96ed2727e4446e26c84552a9a19640c7d720c9b6e661cfcfec03463e92a9d0b228ddc9847c0daa137a19db67294626c535fe71
c388f6ea3eb8cb5dbf09a84374eb021c9297a29394cf77da157c1b8be77b09a4fcbe54bf3dc93d33539e842766ad8e38369093ddc034ac32
583a48e299a4d8b31b606b1729298ee136664b8b77L',
'c1=', '0x6c435db37217bc4da3f225a8f1a0501e03a97d2cbc4fa249df051ed66c1559b68885f4fa181bdd9e98242441f463dbbc1c26d1
eea2c5774a0a905b366c8775bce8e52182dc32a93647c9b8842b74abc434e5b84eeae679a3b19cb7a1ef6ae8f65d22ce6ab438a16119805e
ee83408a68207bbdfde5181a8bd8b4794c711d33c4L')
'''

print("e2=",hex(e2),"n2=",hex(n2),"c2=",hex(c2))
'''
('e2=', '0x3f1',
'n2=', '0x96ed2727e4446e26c84552a9a19640c7d720c9b6e661cfcfec03463e92a9d0b228ddc9847c0daa137a19db67294626c535fe71
c388f6ea3eb8cb5dbf09a84374eb021c9297a29394cf77da157c1b8be77b09a4fcbe54bf3dc93d33539e842766ad8e38369093ddc034ac32
583a48e299a4d8b31b606b1729298ee136664b8b77L',
'c2=', '0x8cb5d8861e5838f41910d6eaf142a8d47b92e0c6b1b1e9e25896f7169644bbb726ccfdc82ba50932fbc45f00c53dda42f8efc3
58a5108cde8aaa9f38b493aa3417c9522924f06847ba4a3dd26f005a610f7633877fbe89e090df5cb3a7a5ebae0fbe72eabb339b21fa2ddd
33844a5cb53e39491fc472721ed676ae07b33c8d6eL')
'''
```

已知d，分解n

```
from gmpy2 import *
from libnum import *
from random import *
def getpq(n,e,d):
    while True:
        k = e * d - 1
        g = randint(0, n)
        while k%2==0:
            k=k//2
            temp=powmod(g,k,n)-1
            if gcd(temp,n)>1 and temp!=0:
                return gcd(temp,n)
d1 = 0x7d12e57b1aa157038ebe5c45b56256270671e6984b0dcdf10a2ea07ce480143240c9a3e1c60870e499306a717073f157476aa88e9
9a7bdf1e2a4adf8ce21025cc6c05035c4a1d7e3b6f061464872e65118384999f0154f3c1761fa68d4685126b7fc98f4c2cdc41c98aa4e099
a868a89099dd2170664647efca2c8d8e06a2e49
e1 = 0x10001
N = 0x96ed2727e4446e26c84552a9a19640c7d720c9b6e661cfcfec03463e92a9d0b228ddc9847c0daa137a19db67294626c535fe71c388
f6ea3eb8cb5dbf09a84374eb021c9297a29394cf77da157c1b8be77b09a4fcbe54bf3dc93d33539e842766ad8e38369093ddc034ac32583a
48e299a4d8b31b606b1729298ee136664b8b77
c1 = 0x6c435db37217bc4da3f225a8f1a0501e03a97d2cbc4fa249df051ed66c1559b68885f4fa181bdd9e98242441f463dbbc1c26d1eea
2c5774a0a905b366c8775bce8e52182dc32a93647c9b8842b74abc434e5b84eeae679a3b19cb7a1ef6ae8f65d22ce6ab438a16119805eee8
3408a68207bbdfde5181a8bd8b4794c711d33c4
m1 = pow(c1, d1, N)
e2 = 0x3f1
c2 = 0x8cb5d8861e5838f41910d6eaf142a8d47b92e0c6b1b1e9e25896f7169644bbb726ccfdc82ba50932fbc45f00c53dda42f8efc358a
5108cde8aaa9f38b493aa3417c9522924f06847ba4a3dd26f005a610f7633877fbe89e090df5cb3a7a5ebae0fbe72eabb339b21fa2ddd338
44a5cb53e39491fc472721ed676ae07b33c8d6e
p = getpq(N, e1, d1)
q = N // p
phin = (p-1)*(q-1)
d2 = invert(e2,phin)
m2 = pow(c2, d2, N)
print(n2s(m1))
print(n2s(m2))
# flag part one is :2295b774c4467c9a
# flag part two is :ca5c600783b9bde0
```

## cpt3

```
import random
from sympy import nextprime
from Crypto.Util.number import *
from gmpy2 import gcd
flag = 'DASCTF{******************************}'

def lcg(seed,params):
    (m,c,n)=params
    s = seed % n
    while True:
        s = (m * s + c) % n
        yield s


seed = getPrime(128)
m = getPrime(128)
c = getPrime(128)
n = getPrime(129)
```

```
key_stream = lcg(seed,(m,c,n))
num=[]
for _ in range(6):
 num.append(next(key_stream))

secret =  next(key_stream)

e = nextprime(secret)
p = getPrime(1024)
q = getPrime(1024)
r = getPrime(1024)


flag = bytes_to_long(flag)
flag ^= flag >> 10
print(p)
print(p*q*r)
print(pow(flag,e,p*q*r))

'''
[586059925024795371559439659045959212 73L, 525605798656979919420608964379033982804L, 607738431135489138748992347244318940466L, 631747898536603358381419028993140907216L, 134506587010017815645432193254862877 17L, 407826262741495712819054543462943222370L]
138092450043978032187397495330379791355629274237204650898232878263413301988536751004632087169676028049236253598677819980191406826529664613957150122788435561338344715937422320958238628877093605040078776555586363593650646481242888908171897232624141894446324625781720275455534977357099473212936612966142541689717
337350040978482181449013111844302829523144663891819187286682607866458685754549951011566472531169763234515686679508864088607470889304989629340827723212104520517258809131616474312417317774711006955339487212887050507627749883060115272356503093236066529886412246881238992329587364404129297747073698143961676767175222529716900170376716346041943101501401292474388182875376578845599650301803110391746187534690494130275230115209742736805258829179873267114334483321126781850732481246305836819932436728501965102800300076505467948523331776720174756871749373240686162656147010343648897859526732652187699511654247503086990422810301988861912327526300993156937561262679648575145379023102922439887955866891994082934159998204526008564573313706491061264085152512895116566032266665160394374961555950756388019154514423128287500449464366623478123144870281102604761323262299330763073799991805851159892210284086276160468058885831336410091475147640340522976224675404992818282038409183193065303107530049168092540620105120083552580372904554927069109321998620410524986748598618388761467715127564839742806614159382512978830563949967053562802375030363283879451081474764301602860367140250483857874335594802704634427276762179861996608105102610424633434334897307449739846880323406404392707133558068604318100709123534146480241087444970887061019249462781101352675143546896311167223705818928852008449453378657306584370462191508578973172358776091037853477313763351962019320345004699446615484807941331997999389076458388793677731615948701000240709318726951282045320759117339576251397079997283985811950158588527795425826948336313346024033986627235852243190425228625954232765873123256846599000827826522708611404206432633493770575804209798762338855618402273718094480766079299241303909751652645445506321816170450583788237801840068704315862850346527437470362425722250494861205523771581019005
'''
```

先根据lcg的六个数算出m,c,n进而算出第七个数，求得下一个素数即为e
因为p是1024位，m长度小于128字节，所以可以直接把p当作n
最后进行异或

```
from libnum import *
from gmpy2 import *
from sympy import nextprime
from binascii import *
s = [586059925024795371559439659045959212 73, 525605798656979919420608964379033982804, 607738431135489138748992347244318940466, 631747898536603358381419028993140907216, 134506587010017815645432193254862877 17, 407826262741495712819054543462943222370]
t = []
```

```
for i in range(6):
    t.append(s[i]-s[i-1])
all_n = []
for i in range(4):
    all_n.append(gcd((t[i+1]*t[i-1]-t[i]*t[i]), (t[i+2]*t[i]-t[i+1]*t[i+1])))
MMI = lambda A, n,s=1,t=0,N=0: (n < 2 and t%N or MMI(n, A%n, t, s-A//n*t, N or n),-1)[n<1] #逆元计算
for n in all_n:
    n=abs(n)
    if n==1:
        continue
    a=(s[2]-s[1])*MMI((s[1]-s[0]),n)%n
    ani=MMI(a,n)
    b=(s[1]-a*s[0])%n
    seed = (ani*(s[0]-b))%n
    # print(a)
    # print(b)
    # print(n)
# 38
# 38
# 247
# 359731160793570101748267293954921586931 2
# 991116632052036749507348612776709448638
# 395089967299688616248375694782996979450 6
# 304895213771629215412875483024241040557
# 332633353219222389093389121471714482887
# 658483278832814360413959491304994965751
m = 304895213771629215412875483024241040557
c = 332633353219222389093389121471714482887
n = 658483278832814360413959491304994965751
secret = (m * s[5] + c) % n
e = nextprime(secret)
p = 138092450043978032187397495330379791355629274237204650898232878263413301988536751004632087169676028049236253
598677819980191406826529664613957150122788435561338344715937422320958238628877093605040078776555586363593650646 4
812428889081718972326241418944632462578172027545553497735709947321293661296614254168971 7
cipher = 405229762246754049928182820384091831930653031075300491680925406201051200835525803729045549270691093219 9
862041052498674859861838876146771512756483974280661415938251297883056394996705356280237503036328387945108147476 4
301602860367140250483857874335594802704634427276762179861996608105102610424633434334897307449739846880323406404 3
927071335806860431810070912353414648024108744497088706101924946278110135267514354689631116722370581892885200844 9
453378657306584370462191508578973172358776091037853477313763351962019320345004699446615484807941331997999389076 4
583887936777316159487010002407093187269512820453207591173395762513970799972839858119501585885277954258269483363 1
334602403398662723585224319042522862595423276587312325684659900082782652270861140420643263349377057580420979876 2
338855618402273718094480766079299241303909751652645445506321816170450583788237801840068704315862850346527437470 3
624257222504948612055237771581019005
d = invert(e, (p-1))
flag = pow(cipher, d, p)
_flag = ''
for i in range(len(bin(flag)[2:])):
    j = i
    k = int(bin(flag)[2:][i])
    while j >= 10:
        j -= 10
        k ^= int(bin(flag)[2:][j])
    _flag += str(k)
print(n2s(int(_flag, 2)))
# DASCTF{b19998fb5acd197e441029b37bc246d7}
```

## cpt4

```
from Crypto.Util.number import *
import binascii
import gmpy2
flag = 'flag is :*******************************'
hex_flag=int(flag.encode("hex"),16)

p=getPrime(256)
q=getPrime(256)
n=p*q

e=0x3

c=pow(hex_flag,e,n)

m=((hex_flag>>120)<<120)

print("n=",hex(n))
print("e=",hex(e))
print("c=",hex(c))
print("m=",hex(m))

'''
('n=', '0x8bc8479ebf10dc1a4c24d6dfd6effd0437969eebf67654bc5c495bf2577f15226c15b9793ce9363c5986c485c2932fc7e7e6da
ac8dc108cca6d1b3850353fa2fL')
('e=', '0x3')
('c=', '0x87c05b7868cf54a58e19fe7a969a0213101f045e2afbf7547534564e918b62caa8187c773a8168ff464b20c28ce0e33383a600
351883bb0938b2ecf0c45d59f3L')
('m=', '0x666c6167206973203a373930666637353265333839383861353200000000000000000000000000000000L')
'''
```

coppersmith's method Stereotyped messages ， 已知m高位的攻击，脚本为sage

```
from Crypto.Util.number import *
e = 0x3
b = 0x666c6167206973203a373930666637353265333839383861353200000000000000000000000000000000
n = 0x8bc8479ebf10dc1a4c24d6dfd6effd0437969eebf67654bc5c495bf2577f15226c15b9793ce9363c5986c485c2932fc7e7e6daac8d
c108cca6d1b3850353fa2f
c = 0x87c05b7868cf54a58e19fe7a969a0213101f045e2afbf7547534564e918b62caa8187c773a8168ff464b20c28ce0e33383a6003518
83bb0938b2ecf0c45d59f3
kbits = 120
PR.<x> = PolynomialRing(Zmod(n))
f = (x + b)^e-c
x0 = f.small_roots(X=2^kbits, beta=1)[0]
m = x0 + b
print(long_to_bytes(m))
# b'flag is :790ff752e38988a521c73837942b1414'
```

## cpt5

```python
from Crypto.Cipher import DES
import random
import base64
from secret import flag

class _2DES:
    def __init__(self):
        pass

    def Get_key(self):
        k1="".join([random.choice('0123456789abcdef') for i in range(8)])
        return (k1[:4]*2,k1[4:]*2)

    def Enc(self,msg,key1,key2):
        k1,k2=key1,key2
        e1=DES.new(k1.encode(),DES.MODE_CBC,bytes(8))
        e2=DES.new(k2.encode(),DES.MODE_CBC,bytes(8))
        c=e2.decrypt(e1.encrypt(msg))
        return base64.b64encode(c)

if __name__ == "__main__":
    _2des=_2DES()
    k1,k2=_2des.Get_key()
    c=_2des.Enc(b'12345678',k1,k2)
    print(c)
    c=_2des.Enc(flag,k1,k2)
    print(c)

#b'D4meWwYAUE4='
#b'qMMGe3wORmFJePnQnM2ZeA=='
```

中间相遇攻击

```python
from Crypto.Cipher import DES
import random
import base64
c1 = b'D4meWwYAUE4='
c2 = b'qMMGe3wORmFJePnQnM2ZeA=='
lst = '0123456789abcdef'
lst = list(lst)
k1 = []
k2 = []
for a in lst:
 for b in lst:
  for c in lst:
   for d in lst:
    k1.append((a+b+c+d)*2)
    k2.append((a+b+c+d)*2)
E1 = []
for i in k1:
 e1=DES.new(i,DES.MODE_CBC,bytes(8))
 E1.append(e1.encrypt(b'12345678'))
E2 = []
for i in k2:
 e2=DES.new(i,DES.MODE_CBC,bytes(8))
 E2.append(e2.encrypt(base64.b64decode(c1)))
for i in E2:
 if i in E1:
  same = i
  break
for i in k1:
 e1=DES.new(i,DES.MODE_CBC,bytes(8))
 if e1.encrypt(b'12345678') == same:
  k1 = i
  break
for i in k2:
 e2=DES.new(i,DES.MODE_CBC,bytes(8))
 if e2.encrypt(base64.b64decode(c1)) == same:
  k2 = i
  break
c2 = base64.b64decode(c2)
e1=DES.new(k1,DES.MODE_CBC,bytes(8))
e2=DES.new(k2,DES.MODE_CBC,bytes(8))
print(e1.decrypt(e2.encrypt(c2)))
# b'_Meet_in_Middle_'
```