BUUCTF [BSidesCF 2019] Mixer

原创

Senimo_ 于 2021-01-08 18:02:02 发布 95 收藏 1

分类专栏： BUUCTF WEB Writeup 文章标签： BUUCTF BSidesCF 2019 Mixer writeup CTF

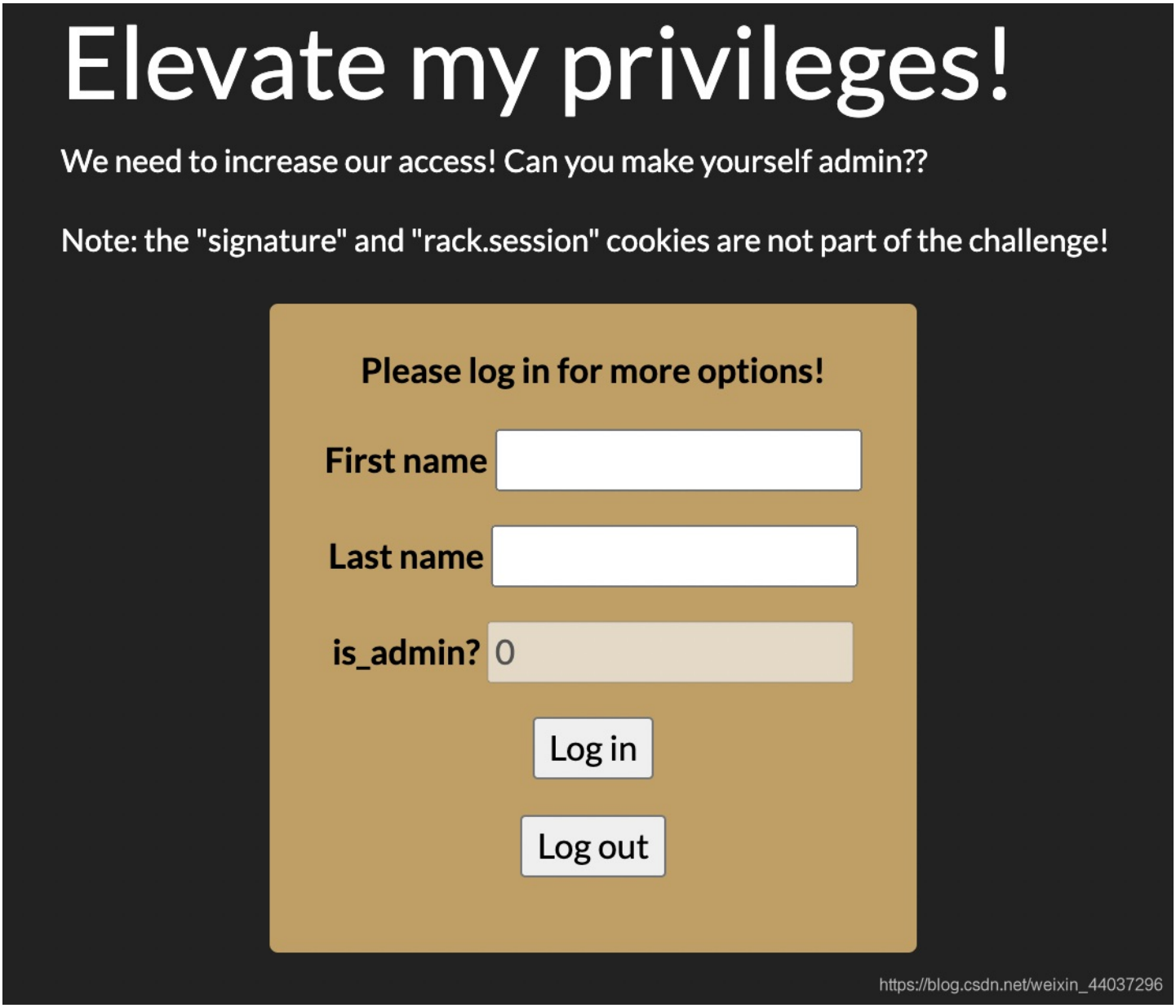
本文链接：https://blog.csdn.net/weixin_44037296/article/details/112370510

BUUCTF [BSidesCF 2019] Mixer

考点：

1. 图解分组密码五大工作模式
2. **ECB**模式加密拆解构造

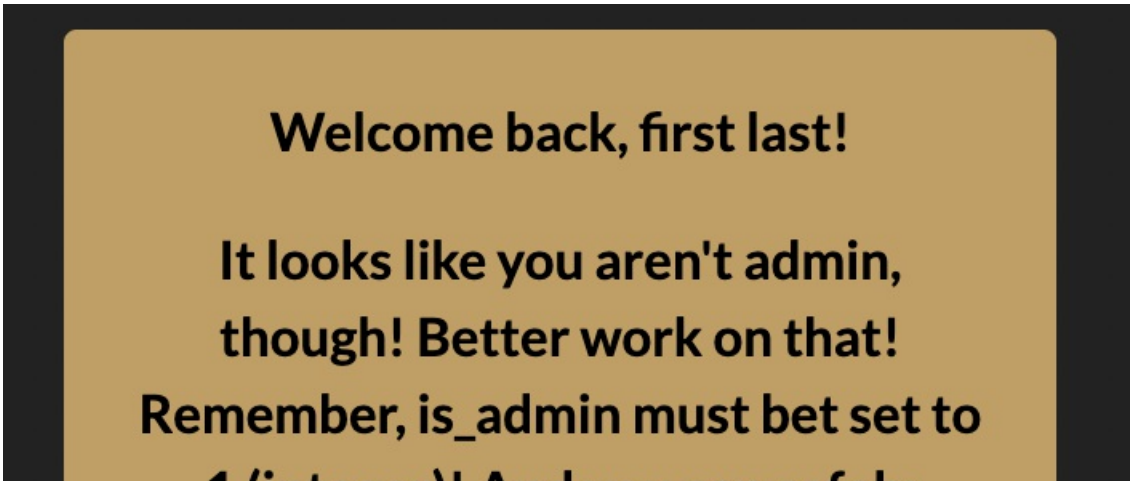
启动环境：

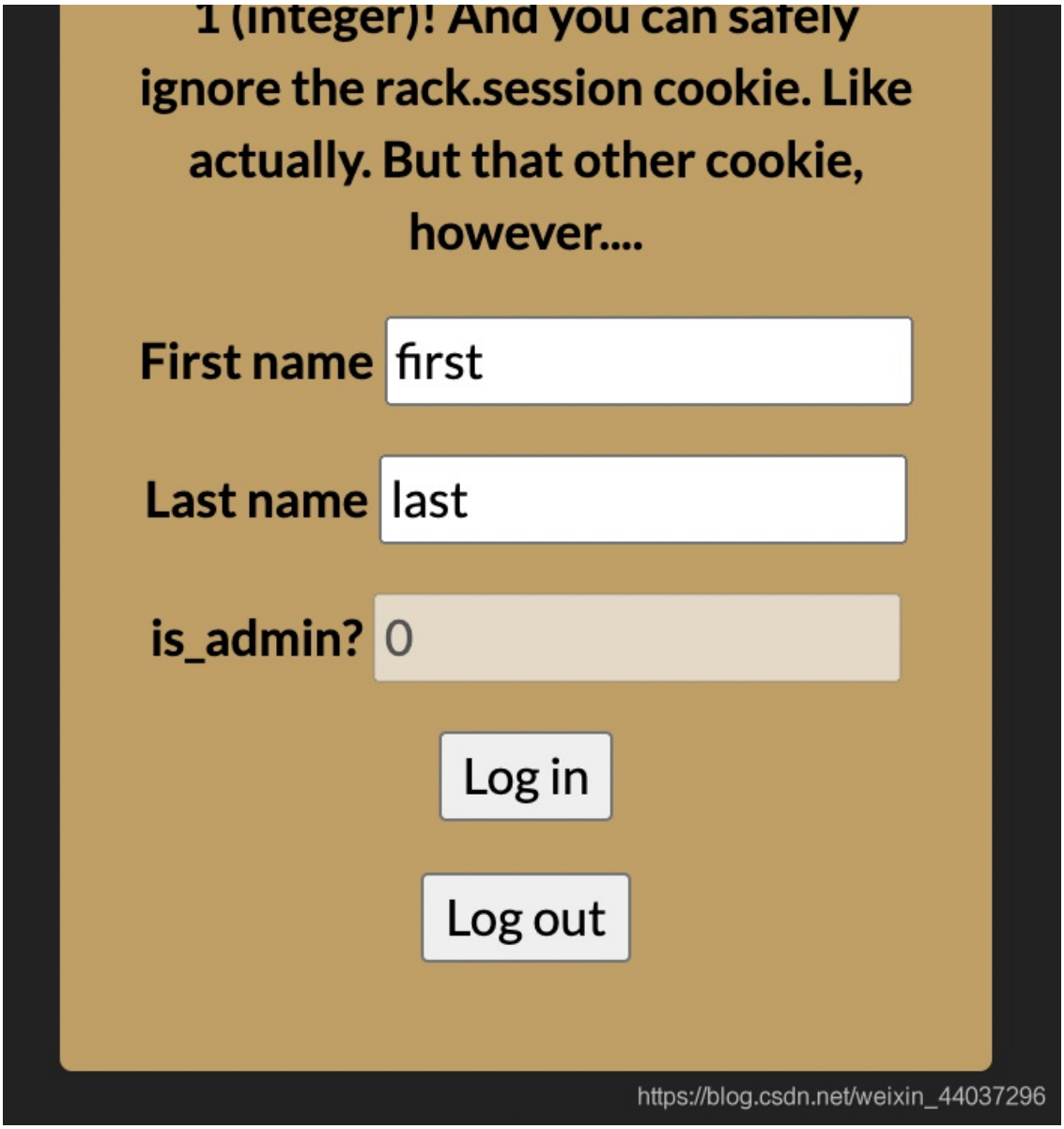


页面提示需要提升权限，并且提到了`cookie`，输入框中还有不能输入的`is_admin`属性
修改前端，使其值等于`1`：

```
<input type="text" class="name" disabled="1" value="1">
```

随意输入`First name`与`Last name`查看回显：





提示了需要以`admin`身份登陆，并将`is_admin`设置为`1`，使用**BurpSuite**抓取数据包：

Request

Raw | Params | Headers | Hex

```
1  GET /?action=login&first_name=first&last_name=last HTTP/1.1
2  Host: 36032f23-3e61-4aa8-8f87-9ee90f20e883.node3.buuoj.cn
3  Upgrade-Insecure-Requests: 1
4  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 11_1_0) AppleWebKit/537.36
   (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
5  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a
   png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
6  Referer: http://36032f23-3e61-4aa8-8f87-9ee90f20e883.node3.buuoj.cn/
7  Accept-Encoding: gzip, deflate
8  Accept-Language: zh-CN,zh;q=0.9
9  Cookie: UM_distinctid=176cc63994071c-0a3bb0b090c52f-6d112d7c-13c680-176cc639941f6e
10 Connection: close
11
12
```


使用**Repeater**发送后，获得回显：

Response

Raw Headers Hex

```
HTTP/1.1 302 Found
Server: openresty
Date: Fri, 08 Jan 2021 09:21:52 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 0
Connection: close
Location: http://36032f23-3e61-4aa8-8f87-9ee90f20e883.node3.buuoj.cn/
Set-Cookie: user=
c2fd13c78005fca26fb76dc643784a7cb0a41e9852beb1b062f2ae2bd169f740bf600f9caf9da5bfe
d45f1405bb5efe800f602c498ddd7a8251b7b664887c2fc;
domain=36032f23-3e61-4aa8-8f87-9ee90f20e883.node3.buuoj.cn; path=/; HttpOnly
Set-Cookie: rack.session=
BAh7B0kiD3Nlc3Npb25faWQGOgZFVEkiRTRkY2I3YTYyODU2OTNhYzlkODI3%0AZDM5ZGFjOGE1YmQ4MG
RmOWMwNzI3MjdjODNlOGFkOGNhMGZhNzYxMzZmMDQG%0AOwBGSSIMYWVzX2tleQY7AEYiJWmHXKegligb
Tz1DV2oaaOVDKe2Km9hsWvxL%0AME4vaRKZ%0A--4cf1964e7b7854e352dcccc5df2f78e84d45cb2d;
path=/; HttpOnly
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Xss-Protection: 1; mode=block
```


其存在`Set-Cookie`字样，将回显页面刷新，重新抓取数据包：

得到了其**Cookie**：

```
user=263b220db546fce7fa8f2ff161f836c51beb4a501c28b01d69dfcba62631bfd6a1f3ec4b16eeb7039c044a87d3206ecb15fd49a93d1
0d8c3917b024cd534c17b; rack.session=BAh7B0kiD3Nlc3Npb25faWQGOgZFVEkiRWFmNDg5NzQ1YjM0MjUxMmFkNzc0%0AMWYzZTE5YzI3O
DEzNDgyMmRlOTA1OGJiYjM2Y2M1MjA2NjFkOGZiYzUwODYG%0AOwBGSSIMYWVzX2tleQY7AEYiJY9KXebdt%2FZ1upUj4nMb%2FBNDIPff8k5riH
uK%0AJqBZHv0U%0A--b23edaf66d480964e13a229e4f409d8754a94835
```

因为其提示和`rack.session`无关，所以尝试修改`user`的前三位为`111`，查看回显：

```
34
35        <div class="alert alert-warning" role="alert">
36
37            <p>Error parsing JSON: 765: unexpected token at
    '&????1?s??.?O???irst","last_name":"last","is_admin":0}'</p>
38
39        </div>
40
41    </div>
```


成功得到报错内容，因为改动了开头，所以只有后半部分是完整的，相同方式获取前面的内容，也就是修改偏后方的三位为`111`，得到回显：

```
37            <p>Error parsing JSON: 765: unexpected token at
    '{"first_name":"first","last_name":"last","is_admL?}2F?????n???)'</p>
38
39        </div>
40
```

完整的`user`解密为：

```
{"first_name":"first","last_name":"last","is_admin":0}
```

查阅资料：图解分组密码五大工作模式
本题也就是每块内容被分成固定的大小块单独加密，推测为**ECB**模式，若是**CBC**模式，修改前面内容，后面内容会变成乱码。

ECB加密是`16`位一组，每组相互独立，加密后每组为32位，尝试整块替换，并且在**json**中`1.00 == 1`

首先构造被加密的字符串：

```
{"first_name":"A1.00000000000000","last_name":"last","is_admin":0}
```

可以将字符串拆分为`5`组，也就是：

```
# 第一组
{"first_name":"A

# 第二组
1.00000000000000

# 第三组
","last_name":"l

# 第四组
ast","is_admin":

# 第五组
0}
```

将加密后的第二组放到第四组的后面，构成：

```
"is_admin":1.00000000000000
```

完成了构造，那么就需要加密后的`cookie`，因为没有加密所需的`key`，所以通过原页面完成，构造登陆内容：

```
first name = A1.00000000000000
last name = last
```

提交表单后，获取到`user`的值：

```
b20e97737c12bfa4aa4b2426c48527681bf69e842fec030ee35bcfe578e91b1157a3a96575f74f2ab7b289a3e4da30f32c85e2383de1b089
993d200a66e2faa4fed061c9eaac7b3a19b1399cd05010cf
```

前四组不用动，也就是到`128`前，将`32`到`64`位作为第二组内容取出，并拼接至第四组后，使用Python3：

```
s = "b20e97737c12bfa4aa4b2426c48527681bf69e842fec030ee35bcfe578e91b1157a3a96575f74f2ab7b289a3e4da30f32c85e2383de
1b089993d200a66e2faa4fed061c9eaac7b3a19b1399cd05010cf"

res = s[:128] + s[32:64] + s[128:]
print(res)
```

得到构造好的flag：

```
b20e97737c12bfa4aa4b2426c48527681bf69e842fec030ee35bcfe578e91b1157a3a96575f74f2ab7b289a3e4da30f32c85e2383de1b089
993d200a66e2faa41bf69e842fec030ee35bcfe578e91b11fed061c9eaac7b3a19b1399cd05010cf
```

将其替换到`user`中：

Request

Raw | Params | Headers | Hex

```
1  GET / HTTP/1.1
2  Host: 36032f23-3e61-4aa8-8f87-9ee90f20e883.node3.buuoj.cn
3  Cache-Control: max-age=0
4  Upgrade-Insecure-Requests: 1
5  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 11_1_0) AppleWebKit/537.36
   (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
6  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/a
   png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
7  Referer: http://36032f23-3e61-4aa8-8f87-9ee90f20e883.node3.buuoj.cn/
8  Accept-Encoding: gzip, deflate
9  Accept-Language: zh-CN,zh;q=0.9
10 Cookie: UM_distinctid=176cc63994071c-0a3bb0b090c52f-6d112d7c-13c680-176cc639941f6e;
   user=
   b20e97737c12bfa4aa4b2426c48527681bf69e842fec030ee35bcfe578e91b1157a3a96575f74f2ab7b
   289a3e4da30f32c85e2383de1b089993d200a66e2faa41bf69e842fec030ee35bcfe578e91b11fed061
   c9eaac7b3a19b1399cd05010cf; rack.session=
   BAh7B0kiD3Nlc3Npb25faWQGOgZFVEkiRWFmNDg5NzQ1YjM0MjUxMmFkNzc0%0AMWYzZTE5YzI3ODEzNDgy
   MmRlOTA1OGJiYjM2Y2M1MjA2NjFkOGZiYzUwODYG%0AOwBGSSIMYWVzX2tleQY7AEYiJY9KXebdt%2FZ1up
   Uj4nMb%2FBNDIPff8k5riHuK%0AJqBZHv0U%0A--b23edaf66d480964e13a229e4f409d8754a94835
11 Connection: close
12
```


发送数据包，得到flag：

```
    <p>Welcome back, A1.00000000000000 last!</p>


      <p>And it looks like you're admin, too! Congrats! Your flag is <span
'highlight'>flag{8e9c7c05-f5b6-4a68-8435-ff643e3ac9f6}</span></p>
```