

K0i CTF 第一期 官方 writeup

作者：李逢天@k0i

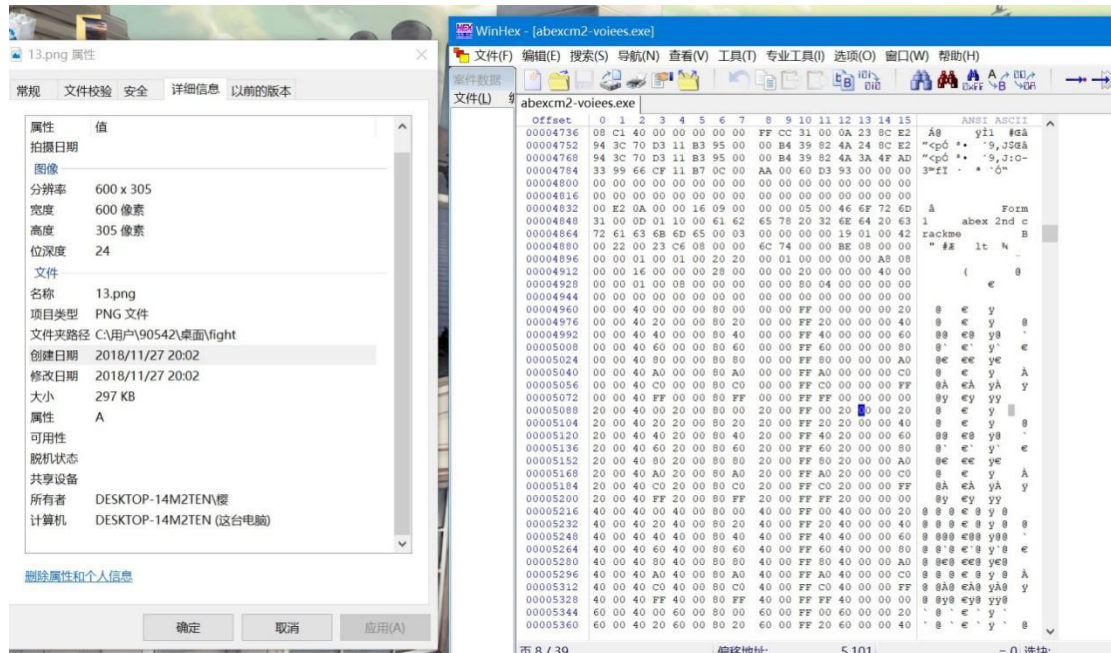
目录

#MISC 1.....	2
#MISC 2.....	2
# MISC 3.....	3
# MISC 4.....	5
# MISC 5.....	6
# MISC 6.....	6
# MISC 7.....	10
# MISC 8.....	12
# WEB 很简单的.....	16
# WEB 你输对了吗?	16
# WEB 基础\$_GET.....	16
# WEB POST.....	17
# WEB 头上张犄角.....	18
# WEB 简单代码审计.....	18
# WEB 等量登陆.....	19
# WEB 你知道 sql 注入吗?	19
# Crypto 早餐吃培根.....	20
# Crypto 签到.....	20
# Crypto 滴滴嗒嗒.....	20
# Crypto 这键盘手感不错.....	20
# Crypto 困在栅栏里的凯撒.....	20
# Crypto 回转 13 位.....	20
# Crypto 密钥生成.....	20
# Crypto 绚丽的 ASCII.....	22
# Re Easy_vb.....	22
# Re Easy_re.....	23
# Re Hello,RE!.....	23
# Re 简单逆向.....	24

#MISC 1

此题为图片类隐写，简单的图片类隐写一般在 16 进制源码，详细信息，其他图片通道中直接隐藏 flag。

首先用 winhex 打开图片，并查看图片的详细信息，并没有发现 flag



然后就要考虑是否在图片的其他通道中隐藏了 flag。



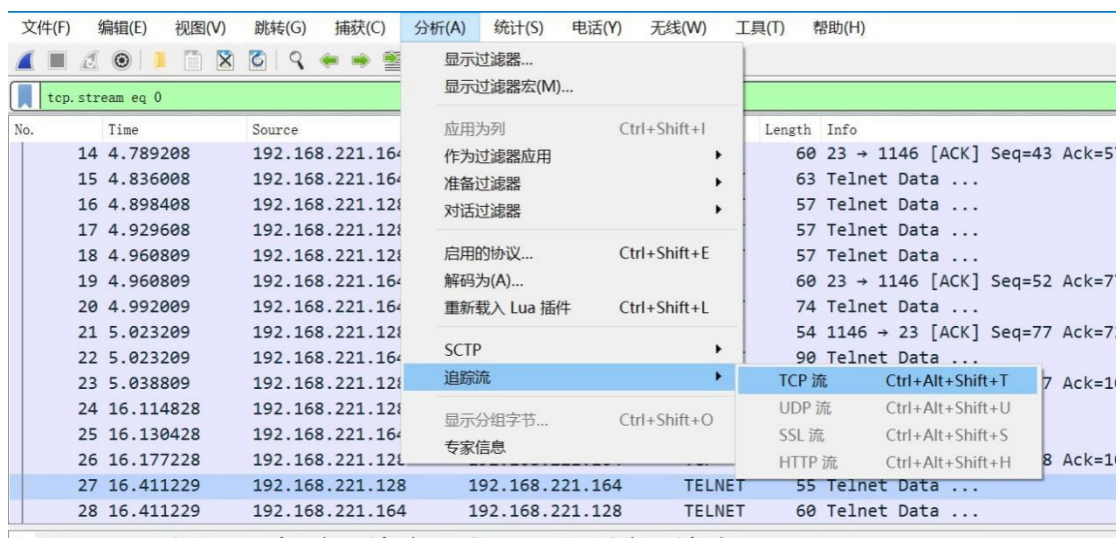
可以发现在图片的 BLUE 0 通道中隐藏了一张二维码。

这里大家可能用的手机扫描的，这里推荐用电脑上的 QRsearch 扫描，因为如果你以后参加比赛的话，手机不让用，而且平常做题 QRsearch 也会更方便一些。

关于此题的具体原理可以参考这个网站：<https://segmentfault.com/a/1190000016223897>

#MISC 2

最基础的数据包分析题。



可以追踪 TCP 流，也可以用 winhex 直接搜索 flag。

MISC 3

发现 gif 图片打不开，题目题型这张 GIF 破损了，请你修复它。

这里需要知道 GIF 图片的文件头为 GIF89，文件尾为分号；

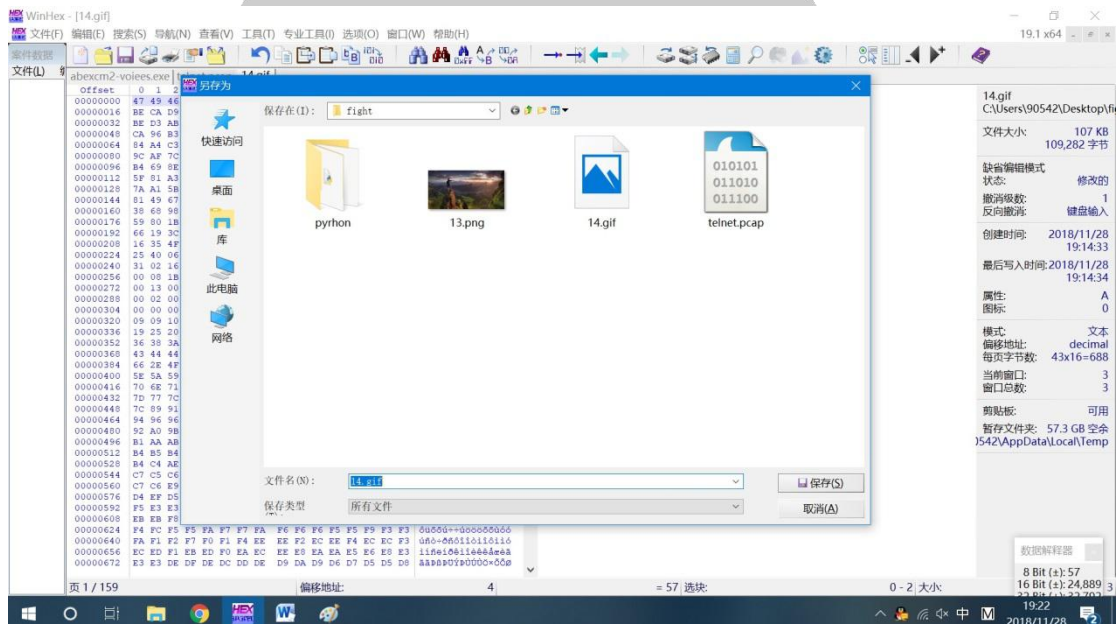
用 winhex 打开文件，在第一个字节处添加三个字节。



这里改成 3，就可以在最前面添加 3 个空字节了。



把前三个字节改为 GIF8，另存为

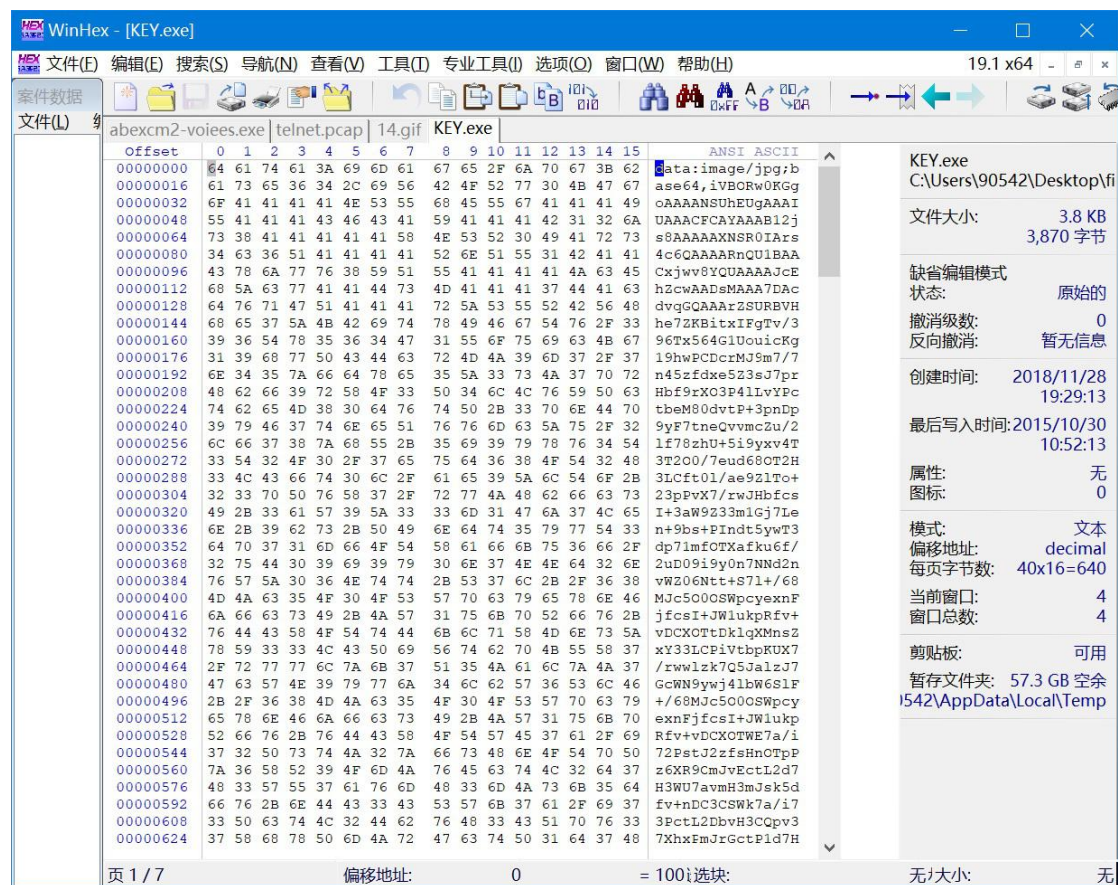


打开图片后是一张动图，flag 分开在每一帧中，如果要直接读出来的话有点困难，这里可以用 stegoslove 打开图片，点击 Frame Browser，就可以分帧查看图片了。



MISC 4

解压出来一个可执行 exe 文件，但是并不能打开，题目提醒我们换个方式打开，我们先来用 winhex 打开看看。



可以看到，这个文件虽然后缀名是 exe，但是其源代码却什么文件结构特征都不具备。根据文件头部的 base64 与 image 提醒，我们可以知道这是一个 base64 的图片编码。

此题可以在网站上搜索 base64 转图片在线工具。



此题如果是在比赛时无法连接网络就不能用在线工具了，这里再介绍一种 python 解法。源码：

```
import base64
```

a=题目中的那串 base64 代码，注意去掉无关字符，只保留 base64'

```
open('1.png','wb').write(base64.b64decode(a))
```

用 QRsearch 扫描得到 flag。

MISC 5

又是一道图片隐写题，我再重复一遍，对于这种简单的图片隐写题，隐藏 flag 的地方可以在 16 进制源码中（用 winhex），图片通道中（用 stegosolve），详细信息中（鼠标右键属性）。这三步做完之后，我们并没有什么发现，那么这里就要想它是否隐藏了其他的文件。要检查它是否隐藏了其它的文件，就用到 kali 里面的 binwalk 命令了。



binwalk 这个命令的作用是分析文件结构的。由图中可以得知这个文件可能隐藏了多个 JPG 图片。

接下来我们用 foremost 分离文件。这里要注意，output 文件夹要清空才能正常运行这条命令。

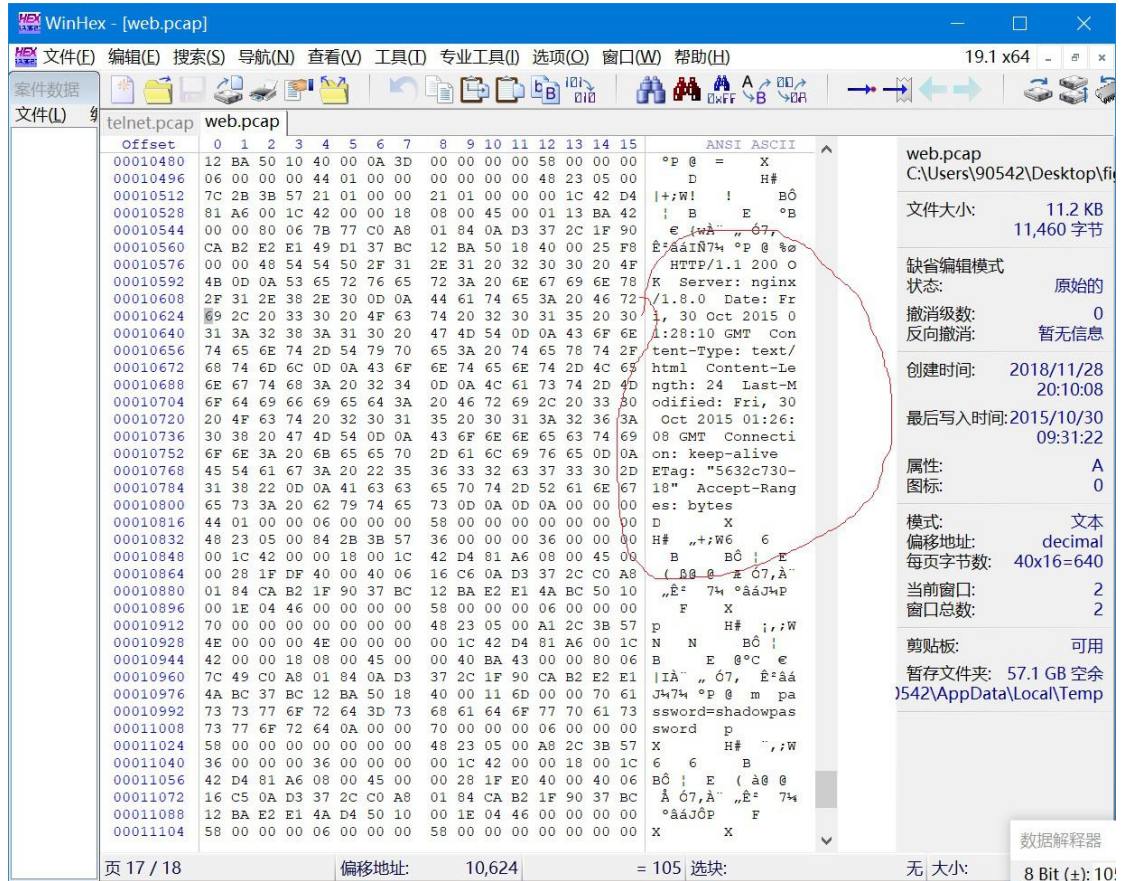


执行后，可以在 output 文件夹中发现多了一张 flag 的 jpg 图片。

MISC 6

解压缩，一个没有后缀名的文件，我们用 winhex 打开，可以查看它的文件头，以确定它到底是什么文件。

k0i CTF 第一期官方 writeup



在文件的中间部分可以发现有一个 HTTP 协议包，再联系文件名为 web，于是可以推断出它可能是一个 pcap 数据包文件，改后缀名为 pcap，成功打开。
接下来进行分析。

No.	Time	Source	Destination	Protocol	Length	Info
10	10.015870	192.168.1.132	10.211.55.44	TCP	54	[TCP Dup ACK 2#1] [TCP ACKed unseen segment] 8080 → 51604 [ACK] Seq=1 Ack=1 Win=0 Len=0
11	16.768800	10.211.55.44	64.233.162.83	TCP	54	[TCP Dup ACK 5#1] 60897 → 80 [ACK] Seq=1 Ack=1 Win=30 Len=0
12	16.768157	64.233.162.83	10.211.55.44	TCP	54	[TCP Dup ACK 6#1] [TCP ACKed unseen segment] 80 → 60897 [ACK] Seq=1 Ack=1 Win=0 Len=0
13	16.991982	10.211.55.44	64.233.162.83	TCP	54	[TCP Dup ACK 7#1] 60902 → 80 [ACK] Seq=1 Ack=1 Win=30 Len=0
14	16.992132	64.233.162.83	10.211.55.44	TCP	54	[TCP Dup ACK 8#1] [TCP ACKed unseen segment] 80 → 60902 [ACK] Seq=1 Ack=1 Win=0 Len=0
15	17.348682	10.211.55.44	192.168.1.132	TCP	74	51833 → 8080 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2626 TSecr=0
16	17.348957	192.168.1.132	10.211.55.44	TCP	62	8080 → 51833 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460 WS=2
17	17.348974	10.211.55.44	192.168.1.132	TCP	54	51833 → 8080 [ACK] Seq=1 Ack=1 Win=29696 Len=0
18	17.349005	10.211.55.44	192.168.1.132	HTTP	147	GET /Key.zip HTTP/1.0
19	17.349079	192.168.1.132	10.211.55.44	TCP	54	8080 → 51833 [ACK] Seq=1 Ack=94 Win=32768 Len=0
20	17.349203	192.168.1.132	10.211.55.44	TCP	314	8080 → 51833 [PSH, ACK] Seq=1 Ack=94 Win=32768 Len=260 [TCP segment of a ...]
21	17.349219	10.211.55.44	192.168.1.132	TCP	54	51833 → 8080 [ACK] Seq=94 Ack=261 Win=30720 Len=0
22	17.349316	10.211.55.44	192.168.1.132	TCP	54	51833 → 8080 [FIN, ACK] Seq=94 Ack=261 Win=30720 Len=0
23	17.349378	192.168.1.132	10.211.55.44	TCP	54	8080 → 51833 [ACK] Seq=261 Ack=95 Win=32768 Len=0
24	17.349933	192.168.1.132	10.211.55.44	HTTP	288	HTTP/1.1 206 Partial Content (application/zip)

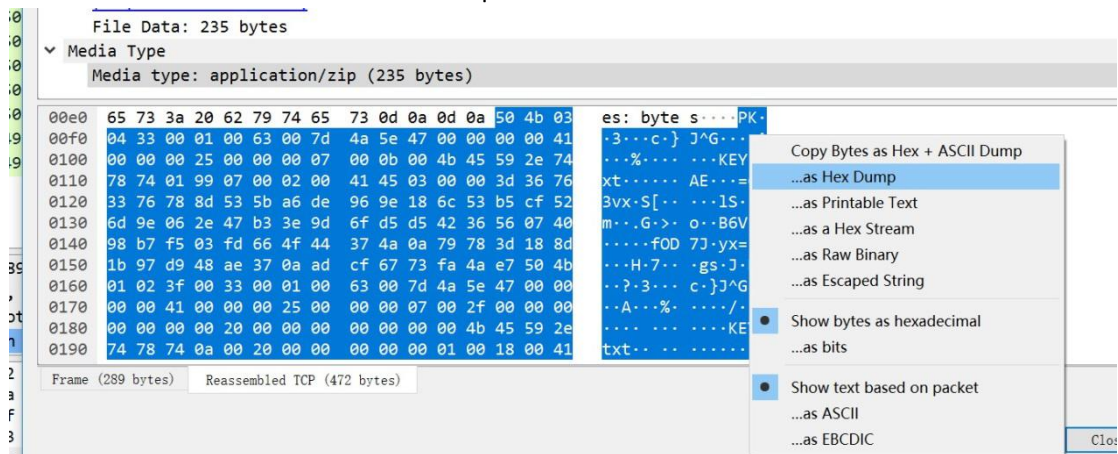
黑色的属于连接失败的，在分析时可以忽略。在中间可以发现有一个 Key.zip，在这里的意思是通过 http 方式传输了一个 key.zip 文件。到这里我们的思路就很清晰了，这个文件很可能就隐藏了 flag，我们把它分离出来就解决这个问题了。
下面来分离这个文件。

在上面的框中输入 http，过滤出所有的 http 包。

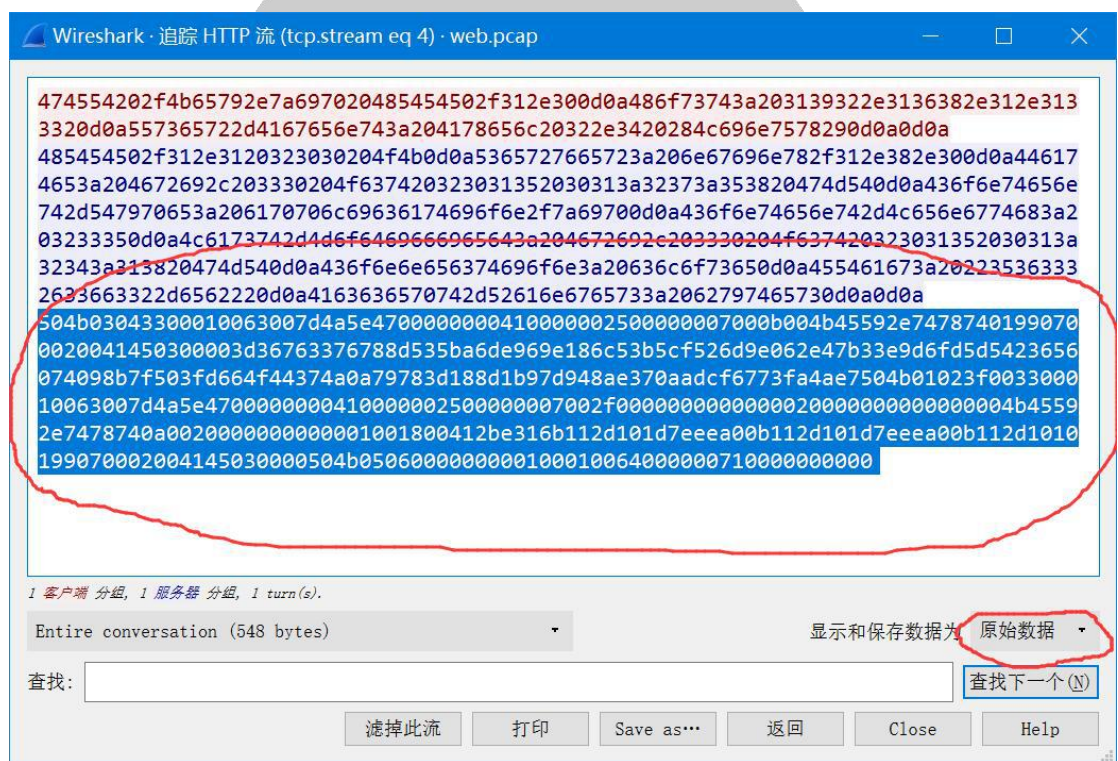
No.	Time	Source	Destination	Protocol	Length	Info
18	17.349005	10.211.55.44	192.168.1.132	HTTP	147	GET /Key.zip HTTP/1.0
24	17.349933	192.168.1.132	10.211.55.44	HTTP	288	HTTP/1.1 206 Partial Content (application/zip)
33	17.350321	10.211.55.44	192.168.1.132	HTTP	130	GET /Key.zip HTTP/1.0
36	17.350408	10.211.55.44	192.168.1.132	HTTP	152	GET /Key.zip HTTP/1.0
41	17.350559	192.168.1.132	10.211.55.44	HTTP	289	HTTP/1.1 200 OK (application/zip)
44	17.350576	10.211.55.44	192.168.1.132	HTTP	152	GET /Key.zip HTTP/1.0
46	17.350669	192.168.1.132	10.211.55.44	HTTP	374	HTTP/1.1 206 Partial Content (application/zip)
49	17.350701	10.211.55.44	192.168.1.132	HTTP	151	GET /Key.zip HTTP/1.0
51	17.350783	192.168.1.132	10.211.55.44	HTTP	374	HTTP/1.1 206 Partial Content (application/zip)
55	17.350852	192.168.1.132	10.211.55.44	HTTP	113	HTTP/1.1 206 Partial Content (application/zip)
84	29.849153	10.211.55.44	192.168.1.132	HTTP	357	GET / HTTP/1.1
88	29.849601	192.168.1.132	10.211.55.44	HTTP	78	HTTP/1.1 200 OK (text/html)

k0i CTF 第一期官方 writeup

逐个分析，在其中一个包中，发现了 zip 的文件头 PK。

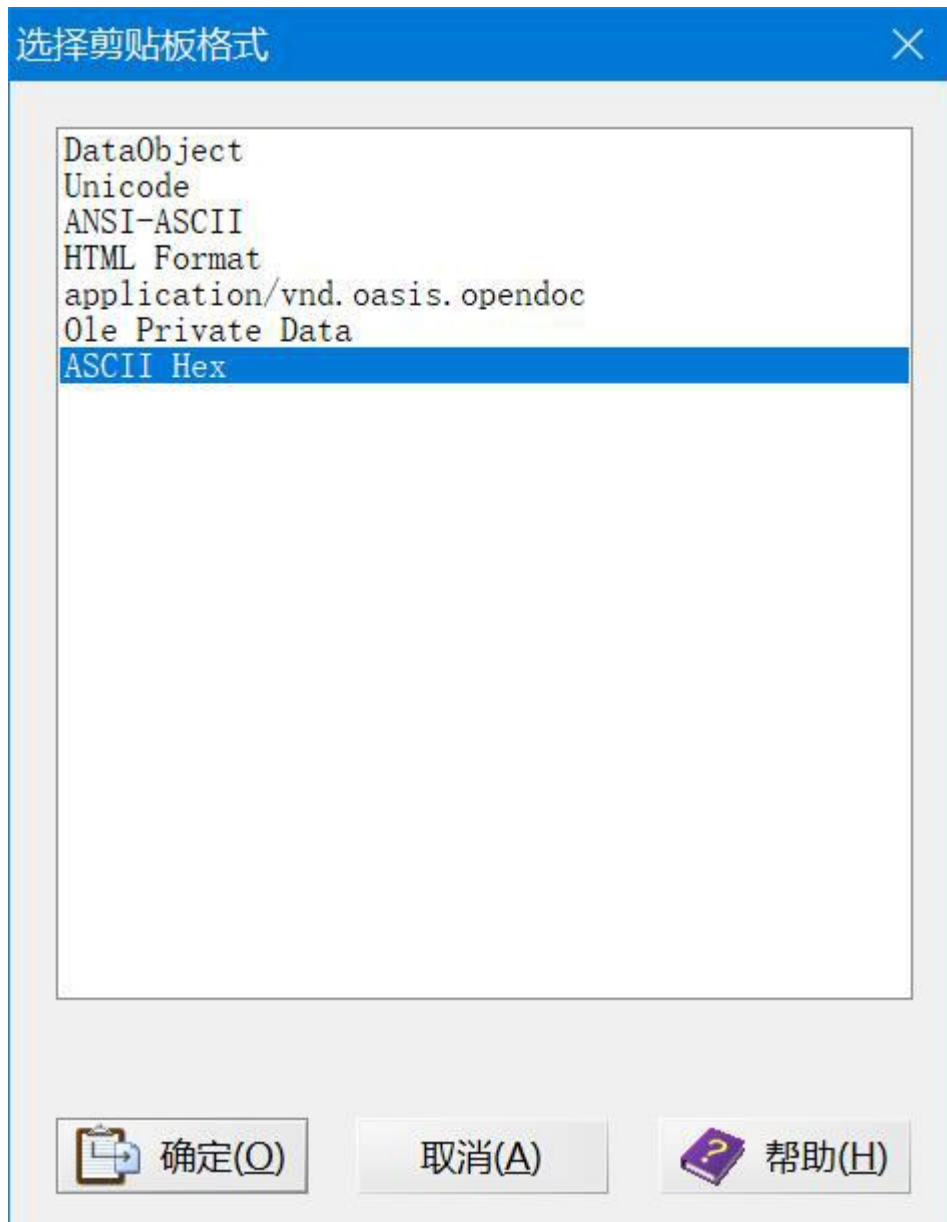


追踪 http 协议。



选择原始数据后复制这串代码，可以看到 504B0304 这串字符，这是 zip 文件的文件头。

接下来打开 winhex 新建文件，作为 hex 黏贴



保存为 zip 文件。发现里面有 KEY，但是需要密码。
然后再回到 wireshark 中找密码。



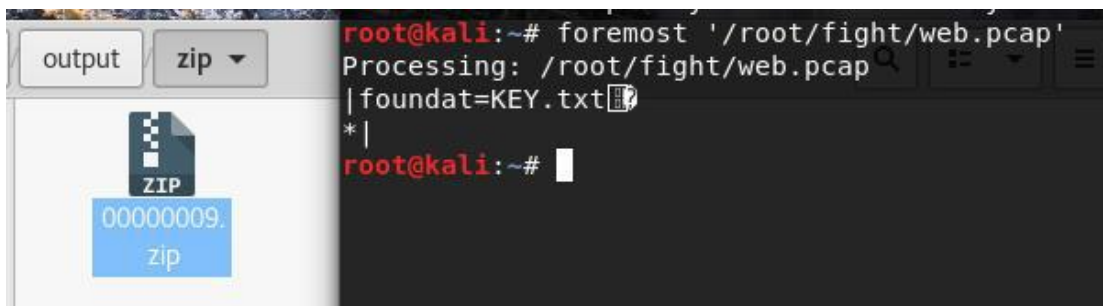
在其中一个 http 包中发现密码。输入后打开压缩包得到 flag。
以上是这个题的正规做法。
下面再来介绍一下这个题的简单解法。
直接把文件放到 kali 里面，用 binwalk 查看文件结构。

```

root@kali:~# binwalk '/root/flight/web.pcap'
123      8888.jpg
DECIMAL      HEXADECIMAL      DESCRIPTION
-----
2862      0xB2E      End of Zip archive, footer length: 22
4890      0x131A      Zip archive data, encrypted compressed size: 65, u
ncompressed size: 37, name: KEY.txt
5103      0x13EF      End of Zip archive, footer length: 22
5968      0x1750      End of Zip archive, footer length: 22
root@kali:~#

```

可以看到文件隐藏了一个 ZIP 文件。接下来用 foremost 分离。



```

root@kali:~# foremost '/root/flight/web.pcap'
Processing: /root/flight/web.pcap
| foundat=KEY.txt
*|
root@kali:~#

```

由于这个压缩包是加密的，我们得想办法找到它的密码。打开 winhex，按 alt+f10 搜索字符串 pass。

B2	E2	E1		I	A	..	Ó	7,	E	=	á	á
00	70	61	J	4	7	4	°	P	@	m	pa	
70	61	73	s	s	w	o	r	d	=	s	h	a
00	00	00	s	w	o	r	d		p			
2C	3B	57	X				H	#				
18	00	1C	6		6		B					

成功找到，输入密码后得到 flag。

MISC 7

拿到图片，我们首先还是要看一下它的 16 进制源码（用 winhex），图片通道（用 stegosolve），详细信息（鼠标右键属性）。这之后并没有发现有价值的信息。

这个题对新手来说还是想不到的，但是如果有了经验，通过图片上向上指的手势不难猜出要修改图片的高度。所以接下来我们需要修改图片的高度。

要修改 PNG 的高度，需要用到 PNG 文件头的知识，首先用 winhex 打开图片。

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	ANSI ASCII	
89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	%PNG	IHDR
00	00	06	1E	00	00	03	1E	08	06	00	00	00	27	CA	40		'Ê@
F2	00	00	0C	12	69	43	43	50	49	43	43	20	50	72	6F	ò	iCCPICC Pro
66	69	6C	65	00	00	48	89	95	57	07	54	93	C9	16	9E	file	H%•W T"É ž
BF	A4	10	12	5A	20	02	52	42	EF	48	AF	D2	7B	11	90	¿	z RBiH-ò{
0E	36	42	12	20	94	00	09	41	C5	AE	2C	2A	B8	76	11	6B	" AA@,*v
C5	8A	AE	80	28	B8	16	40	16	1B	16	2C	2C	02	F6	FA	ÅŠ@€(, @ , , ou	
40	45	65	65	5D	2C	D8	50	79	93	02	BA	BE	76	DE	3D	@Eee],ØPy" °¼vP=	
67	FF	FF	72	FF	FF	3B	FF	9D	DC	99	33	03	80	A2	0D	ghùrcF•B üw3 €@	

图中划红线的地方即是设置图片高度的地方，可以看到，这里图片的高度是十六进制下的 031E，转成十进制也就是 798，这一点可以在图片的详细信息中得到验证。如下图所示。

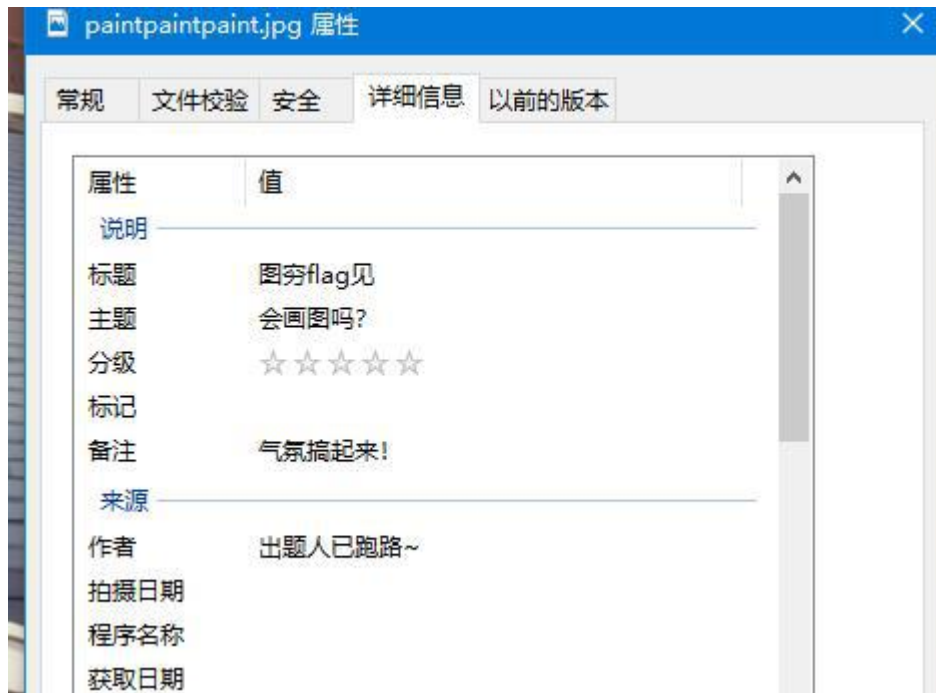


在修改的时候我们可能会不知道到底要改成多大，如果改的太大，图片就会不能正常显示，改的太小，隐藏的信息会显示不全。这里分享一下我的做法，把 031E 的 03 增大 3 位，变为 06。改好之后保存，就可以看到隐藏在下面的 flag 了。

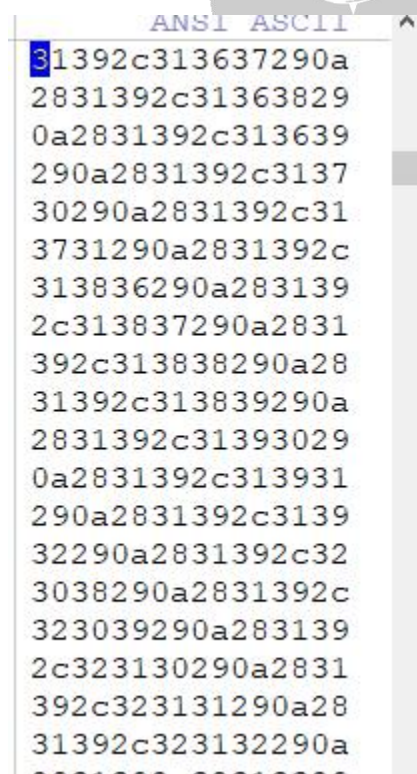


MISC 8

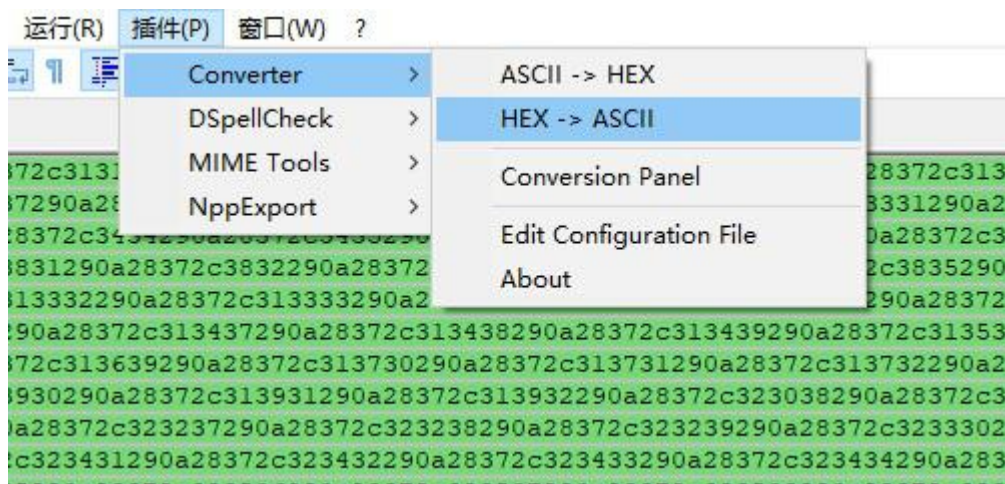
首先拿到图片，我们首先还是要看一下它的 16 进制源码（用 winhex），图片通道（用 stegosolve），详细信息（鼠标右键属性）。在详细信息中的确发现了 hint，但并没有太大价值。



再来查看文件的 16 进制源码，发现在后半部分有大量 16 进制的文本。



我们不妨把这一串代码转成字符串试试。用 notepad++ 中工具，如图所示



转换完之后。

```
(7,128)
(7,129)
(7,130)
(7,131)
(7,132)
(7,133)
(7,134)
(7,135)
(7,136)
(7,137)
(7,138)
```

是一连串坐标。这里如果联系题目和之前在详细信息中 **hint** “会画图吗？”可以猜想到应该是要把这一连串坐标转换成图片。

可以想象有一张白纸，对应着这串坐标，在每一个相应的地方点上一个黑点，当所有的坐标都点完后，就会出现最终的图片（猜想是二维码或者是一串字什么的）。这里如果我们要手工自己点的话，35000 多个点显然是非常困难的。所以这里就要用到 **kali** 中自带的一个绘图工具 **gnuplot**。

在这之前，我们先来处理一下这些坐标，把所有坐标的括号和中间的，去掉。这里在 **notepad++** 中按 **ctrl+f** 打开搜索窗口。如图所示。



点击全部替换。处理之后的文本才能被 `gnuplot` 处理。在 `kali` 的终端中启动 `gnuplot`，然后输入“`plot 文件地址`”即可。注意文件路径两边加单引号。

```
root@kali:~# gnuplot

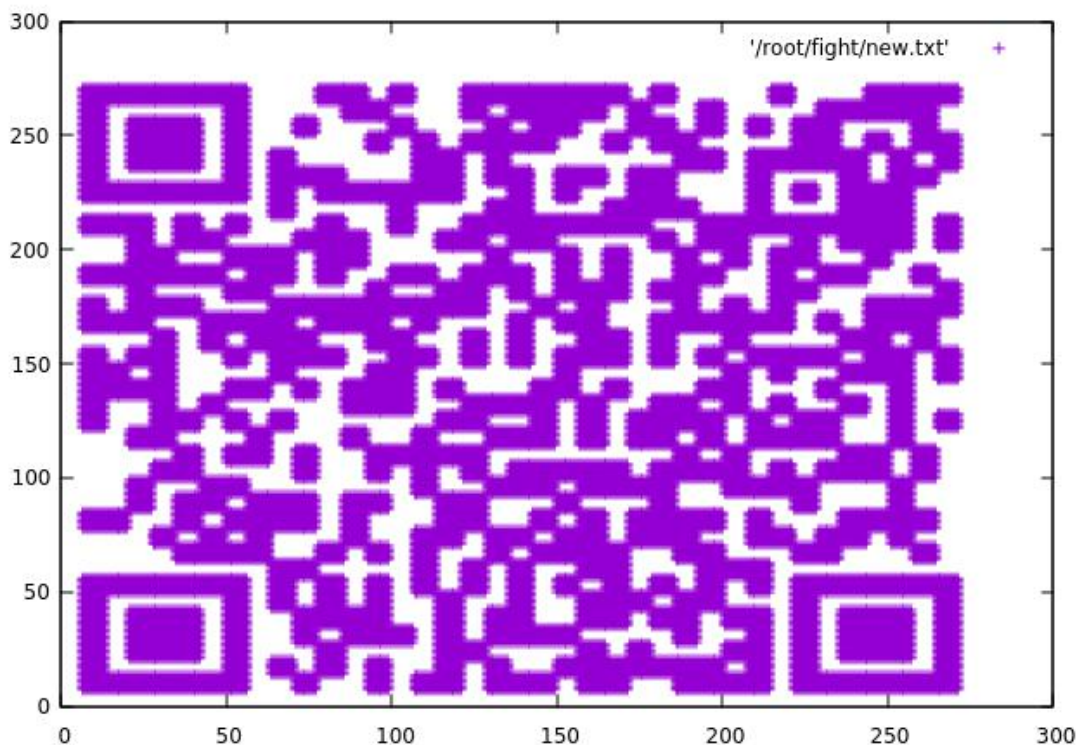
G N U P L O T
Version 5.2 patchlevel 2    last modified 2017-11-01

Copyright (C) 1986-1993, 1998, 2004, 2007-2017
Thomas Williams, Colin Kelley and many others

gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:    type "help FAQ"
immediate help:    type "help" (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> plot '/root/fight/new.txt'
gnuplot>
```

得到一张二维码，`qrsearch` 扫描后得到 `flag`。



WEB 很简单的

Pass:推荐大家使用 firefox 做 web 类题目，原因是用它操作方便。

鼠标右击查看源代码得到 flag。

WEB 你输对了吗？

尝试输入 zhimakaimen，发现只能输入 10 位，最后一位无法输入。这里需要一点 web 前端的知识。大家这里先知道前端代码（通常是 html，JavaScript）是运行在客户端的，它是可以被用户修改的，利用这一点，我们可以绕过只能输入 10 位字符的限制。

鼠标右键查看元素，在对应的输入窗口中可以发现有一个 maxlength。



改为 11。输入 zhimakaimen 后得到 flag。

WEB 基础\$_GET

关于 GET 协议和 POST 协议的具体知识大家自行百度学习。这里我主要说一下怎么区分二者。GET 协议用来传输一些非敏感信息，比如文件路径，用户名。最直观的认识就是在 URL 地址里，? 后面跟的就是 GET 协议传输的内容。因为它可见，所以是不安全的。

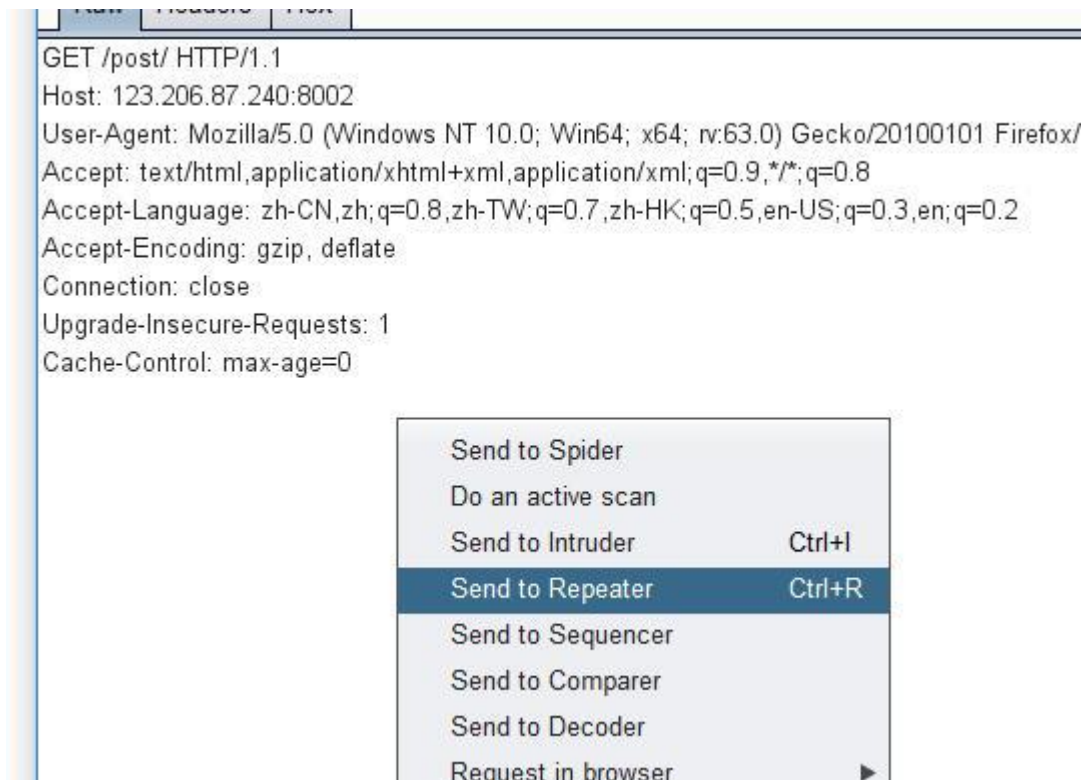
POST 协议用来传输一些敏感信息，比如密码。它是无法直接查看的，因此是安全的。

在这个题中，构造 payload 为?what=flag，黏贴在源地址后面即可。

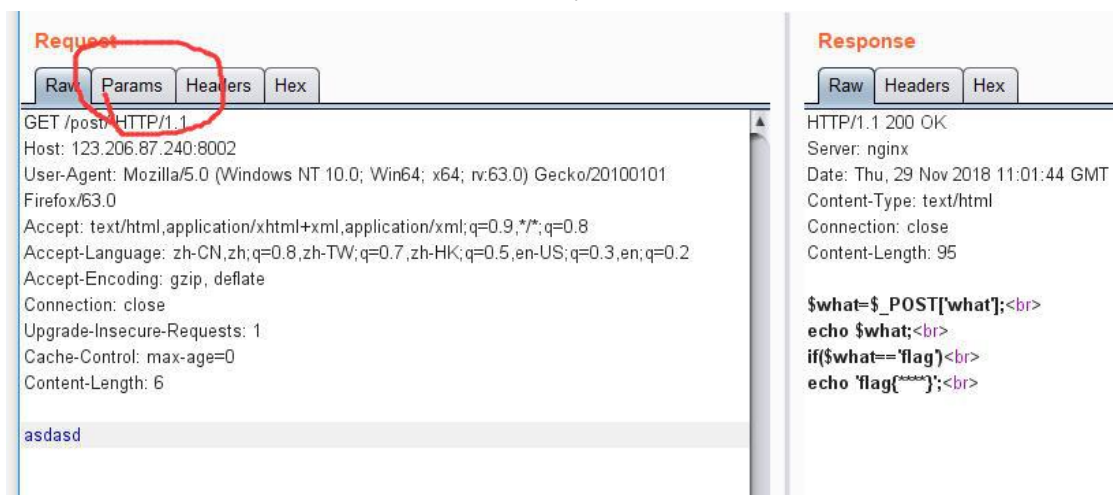
（payload 可以大体的理解为有效信息，以后的学习中会经常出现，不理解的去百度）

WEB POST

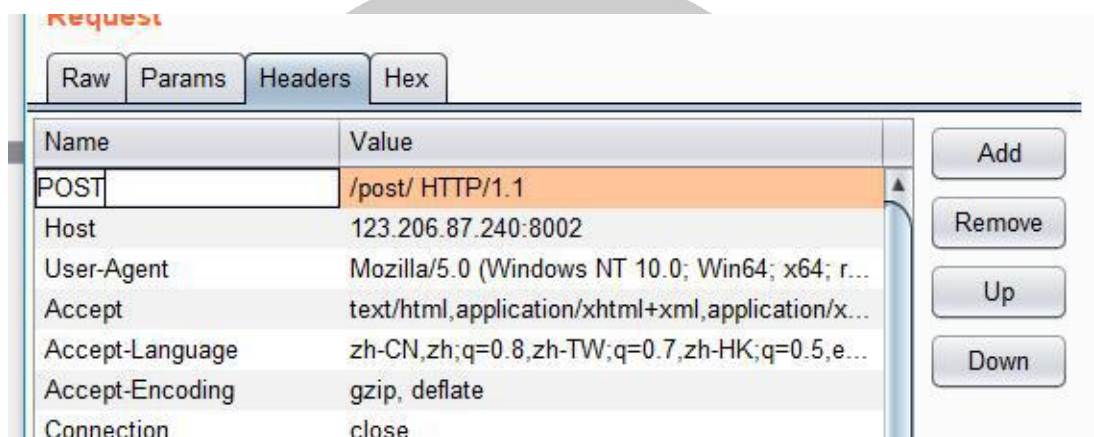
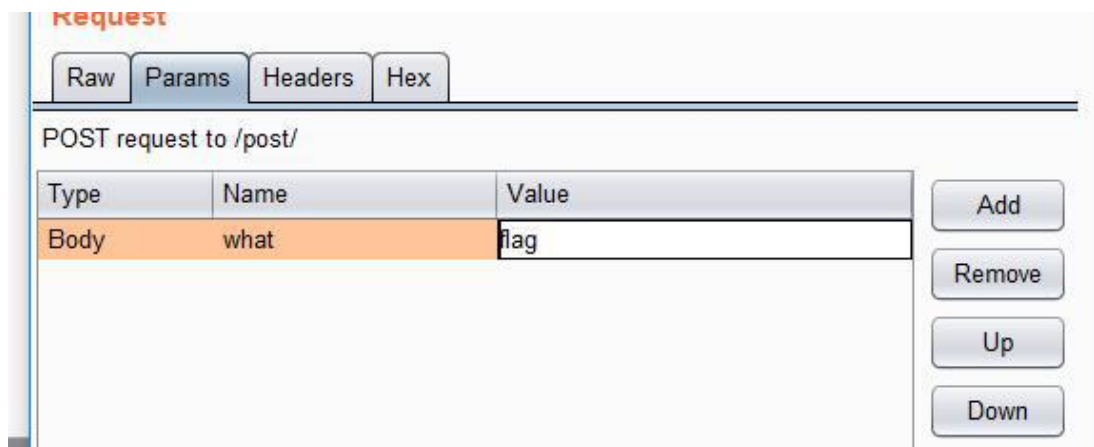
GET 协议可以直接加? 在 URL 里传输, 但是 POST 协议就无法这样做了。这里需要用到工具。推荐使用 burpsuite, 不推荐用 hackbar, 因为前者的功能完全覆盖了后者, 掌握了 burpsuite 也就没必要用后者了。在就是浏览器推荐用 firefox, 因为它在改代理的时候非常方便。如何配置环境使用 burpsuite 抓包自行百度, 这里主要介绍一下这个题的解法。抓到包后 send to repeater



随便传输一串数据, 目的是让抓取的包显示出 params 模块, 如下图所示。



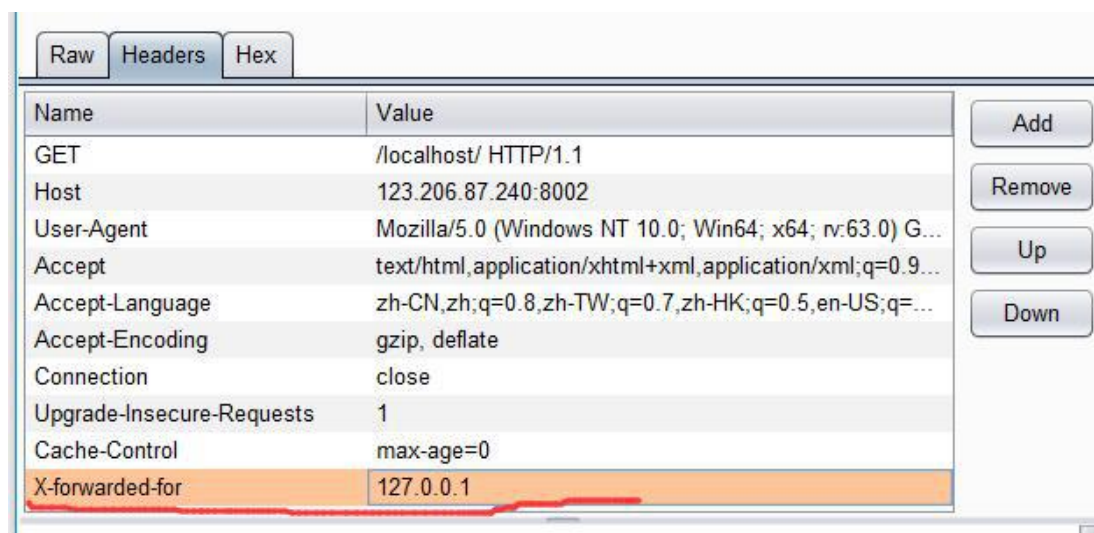
接下来进行改包



改完这两处后，点击 GO，得到 flag。

WEB 请从本地访问

抓到包后 send to repeater。点击 Add 添加一行，按照下图改包，这里需要说明一下的就是 127.0.0.1 是计算机的回环地址，即本地地址。



改好后点击 go，得到 flag。

WEB 头上张犄角

此题暗示了“头”，由此猜想 flag 可能隐藏在返回包的 header 中。抓到包后 send to repeater。点击 go 后再 response 中的 header 中发现 flag。

WEB 简单代码审计

从代码中我们得知要想获得 flag，就要进入 if（）语句，而进入 if（）语句的条件是我们要输入与 flag 一模一样的字符串。这样就引起了矛盾，我们如果知道了 flag，就没有做的必要了，但是不知道又没法显示出 flag。所以这里我们需要想办法让 strcmp（）函数返回 0。要用到的就是这个函数的一个漏洞：它无法处理数组，如果给它传递一个数组就会返回 0，这样我们的目的就达到了。

Payload: ?a[]=1

WEB 等量登陆

打开网址发现 Your password can not be your name. 查看网页源代码，在注释里发现有价值信息。

```

1 <center>
2 <h1>Your password can not be your name.</h1>
3 </center>
4 <br>
5 <br>
6 <br>
7 <!--index.php-->
8 </html>
9

```

那么我们就去这个网页看看。转到这里 <http://ctf.whaledu.com:10010/web16/index.php> 再次查看源代码，发现是一串 PHP。

```

1 <?php
2 error_reporting(0);
3 $flag = '*****';
4 if (isset($_GET['name']) and isset($_GET['password'])) {
5     if ($_GET['name'] == $_GET['password'])
6         print 'name and password must be diffirent';
7     else if (sha1($_GET['name'])=== sha1($_GET['password']))
8         die($flag);
9     else print 'invalid password';
10 }
11 ?>

```

观察源代码。我们要想得到 flag，要传递不同的 name 和 password，但是要求他们的 sha1 值要相同。这样就矛盾了，因为不同的值，它 sha1 加密后也是不同的。所以这里需要我们绕过。用到了 sha1（）函数无法处理数组（跟上题一样）。

Payload: ?name[]=1&password[]=2

WEB 你知道 sql 注入吗？

此题属于 sql 注入题目中较简单的一类，但由于大家第一次接触，再加上用到的 mysql 语法太多，又缺少 web 后端的知识，所以会无从下手。

下面说一下这个题的做法，看不懂的可以再百度，实在不理解就先放过吧，因为真的涉及太多知识。

首先我们输入正常数据 1, 2, 3。可以发现页面正常显示出某人的成绩。

龙龙龙的成绩单

Math	English	Chinese
60	60	70

接下来就要 sql 注入了（何为 sql 注入？就是黑客构造特殊的语句，这串语句被带入到数据库中执行以达到黑客入侵的目的）。

首先第一步：

Payload:

```
1' order by 1 # (正常显示)
1' order by 2 # (正常显示)
1' order by 3 # (正常显示)
1' order by 4 # (正常显示)
1' order by 5 # (非正常显示)
```

有此可以判断有四个字段。

二：爆出数据库名字

注意这里要用-1，因为如果输入 1, 2, 3 的话页面会正常显示搜索结果，只有让它显示失败，才能够运行后面的 select 结果。

```
-1' union select 1,2,3,group_concat(database()) #
```

得到数据库名字 skctf_flag

三：爆出表名

```
-1' union select 1,2,3,group_concat(table_name) from information_schema.tables where
table_schema = 'skctf_flag' #
```

得到表名 fl4g, sc

四：爆出字段名

```
-1' union select 1,2,3,group_concat(column_name) from information_schema.columns where
table_name = 'fl4g' #
```

得到字段名 skctf_flag

五：爆出 flag

```
-1' union select 1,2,3,skctf_flag from fl4g #
```

Crypto 早餐吃培根

对应培根密码表解密,或者直接用 ctftools 自动解密。注意是小写。

Crypto 签到

直接放到 ctftools 里 base64 解密

Crypto 滴滴嗒嗒

直接放到 ctftools 里摩斯密码解密

Crypto 这键盘手感不错

按照一组字母的顺序观察键盘，可以看到中间围了一个字母，依次观察，得到 flag。

Crypto 困在栅栏里的凯撒

先在 ctftools 用栅栏解密，cixd{Qzj_Pbzrofqv}，可以看到这一串字母，很像 flag 的格式。再联系题目，我们知道要再进行一次凯撒解密。做完后得到 flag

Crypto 回转 13 位

先在 ctftools 里面用 rot13 解密，之后进行 base64 解密得到 flag

Crypto 密钥生成

RSA 的详细原理自行百度，这里说一下各个参数的含义。

p:任意一个大的质数

q:任意一个大的质数

n:p*q，用来加密或解密 mod 用的

r:(p-1)*(q-1),p 和 q 的欧拉函数结果相乘

e:随机选择和 r 互质的数字，实际中通常选择 65537

d:d 是 e 关于 r 的模反元素， $de \equiv 1 \pmod r$

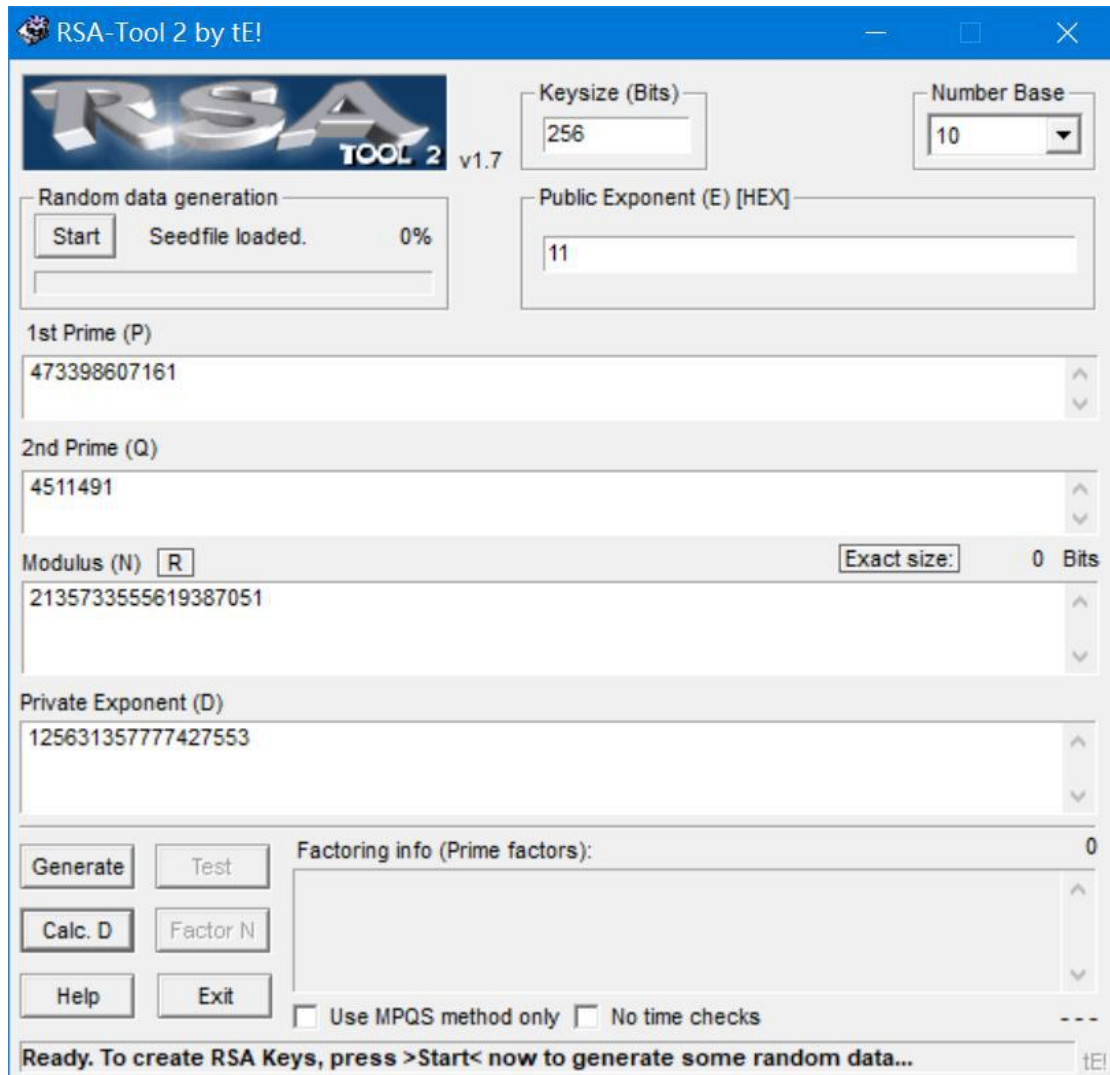
python 脚本如下:

```
p=473398607161
q=4511491
e=17
n=long(p*q)
r=long((p-1)*(q-1))
i=1
while(True):
    x=(i*r)+1
    if(x%e==0):
        d=x/e
        break
    i=i+1
print d
```

以上这串代码可以帮助大家理解 RSA 算法，但是这串代码效率很低，对于太大的数就没办法了。再分享另一个简单点的，不过得要安装 gmpy2，安装方法自行百度。

```
import gmpy2
p=473398607161
q=4511491
e=17
n=p*q
r=(p-1)*(q-1)
d=gmpy2.invert(e,r)
print d
```

上面代码大家争取看懂。最后再来用工具 rsatools 解这个题，如下图.

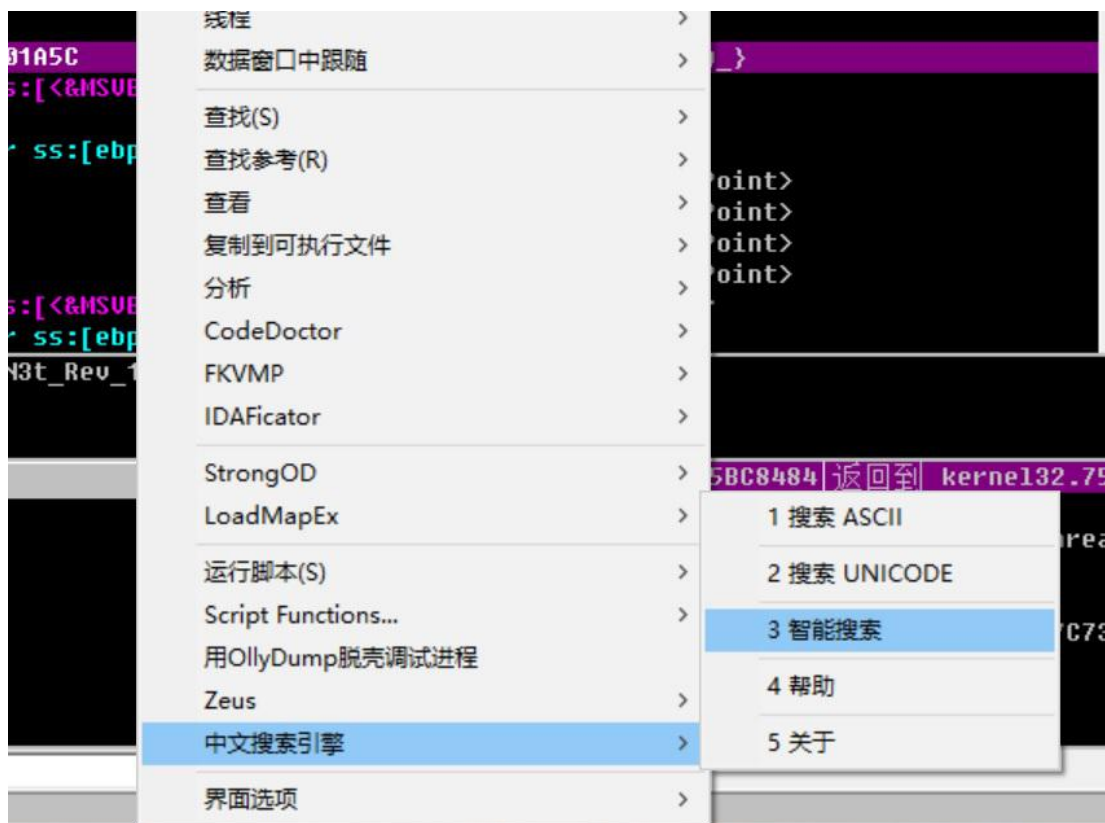


Crypto 绚丽的 ASCII

这里提示+1s，如果把 e 加一位后变为 f，k 加一位变为 l，f 加一位变为 g，到这里思路就有了。就是把每个字符都往后移动一位。（注意这里是在 ascii 表里往后移动的）

Re Easy_vb

首先打开程序运行，点击按钮无反应。放到 OD 中，右键中文引擎搜索，智能搜索。



在注释里面发现 flag。

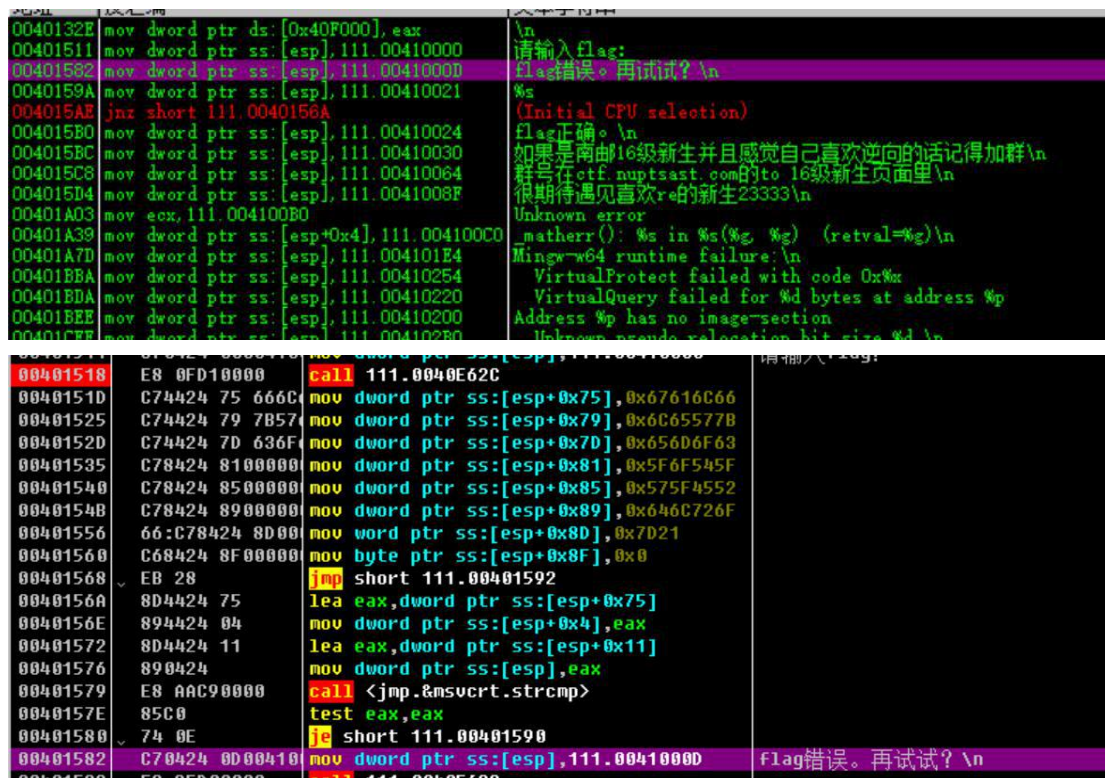
Re Easy_re

做法同上。

Re Hello,RE!

首先打开程序运行，输入任意字符串，提示“flag 错误，再试试”。

用 OD 打开，智能搜索，找到这串字符串后双击。如下图



可以看到在 401582 上面有一个 je 条件跳转,这是一个关键跳转,条件成立会跳转到打印“flag 正确”的地方,而错误就会向下打印“flag 错误”。我们接着往上分析,401579 有个 call 函数,strcmp 即为 string compare 字符串比较的意思,这里可以猜想到此函数的作用为把用户输入的字符串与程序内存中的 flag 做比较。接下来的思路就有了,我们如果找到储存在程序内存中的 flag,此题就解决了。我们再往上分析,发现从 40151D 到 401560 一直再用 mov 指令把数据内存为 0x67616c66 等值的数据移动到栈中。尝试在数据窗口中跟随,发现 flag。

0040151D	C74424 75 666C	mov dword ptr ss:[esp	注释	;	
00401525	C74424 79 7B57	mov dword ptr ss:[esp	断点(P)	>	
0040152D	C74424 7D 636F	mov dword ptr ss:[esp	RUN 跟踪	>	
00401535	C78424 81000000	mov dword ptr ss:[esp	此处为新 EIP	Ctrl+Gray *	
00401540	C78424 85000000	mov dword ptr ss:[esp	转到	>	
00401548	C78424 89000000	mov dword ptr ss:[esp	线程	>	
00401556	66:C78424 8D00	mov word ptr ss:[esp+	数据窗口中跟随	>	选择
00401560	C68424 8F000000	mov byte ptr ss:[esp+	内存地址(A)		
00401568	EB 28	jmp short 111.00401592	查找(S)	>	
0040156A	8D4424 75	lea eax,dword ptr ss:	查找参考(R)	>	
0040156E	894424 04	mov dword ptr ss:[esp	查看	>	
00401572	8D4424 11	lea eax,dword ptr ss:	复制到可执行文件	>	
00401576	890424	mov dword ptr ss:[esp	分析	>	
00401579	E8 AAC90000	call <jmp.&msvcrt.strc	CodeDoctor	>	
0040157E	85C0	test eax,eax	FKVMP	>	
00401580	74 0E	je short 111.00401590	IDAficator	>	
00401582	C70424 0D00410	mov dword ptr ss:[esp	StrongOD	>	00410021 ASC
00401589	E8 9ED00000	call 111.0040E62C	LoadMapEx	>	0065FE31
0040158E	EB 02	jmp short 111.00401592	运行脚本(S)	>	00000000
堆栈 ss:[0065FE95]=67616C66			Script Functions...	>	00401ED0 111
地址	数值	ASCII	注释		0065FE1C
0065FE95	67616C66	Flag			77C549B9 返回
0065FE99	6C65577B	{We1			
0065FE9D	656D6F63	come			
0065FEA1	5F6F545F	_To_			
0065FEA5	575F4552	RE_W			

Re 简单逆向

用 IDA 打开,搜索 flag。