CTF-安恒19年二月月赛部分writeup

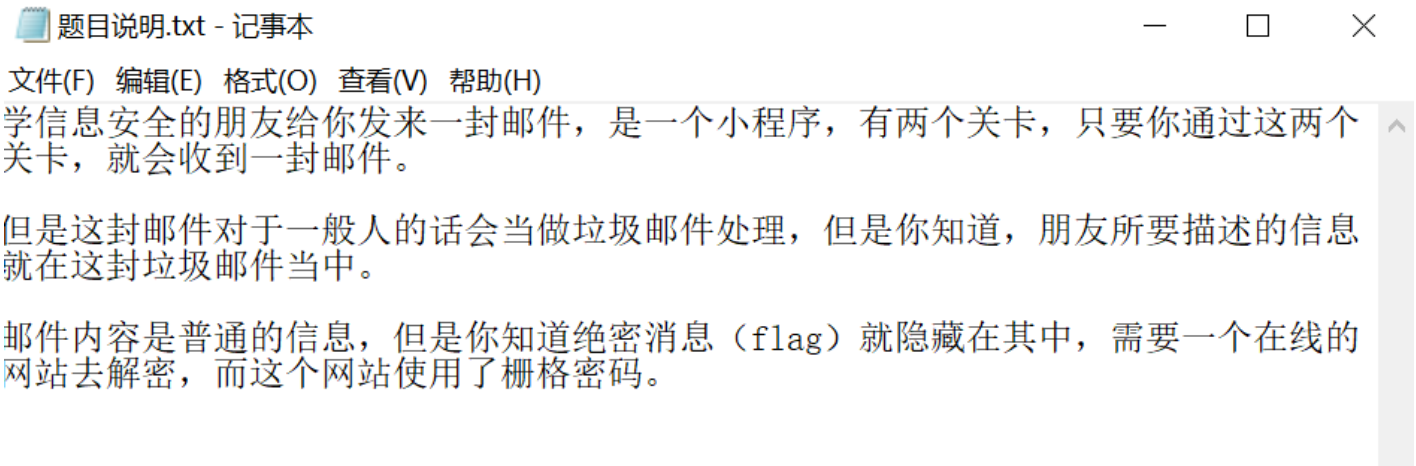
转载

weixin_33968104 于 2019-02-26 11:12:00 发布 1262 收藏
文章标签： python 游戏
原文链接：http://www.cnblogs.com/pureqh/p/10435229.html

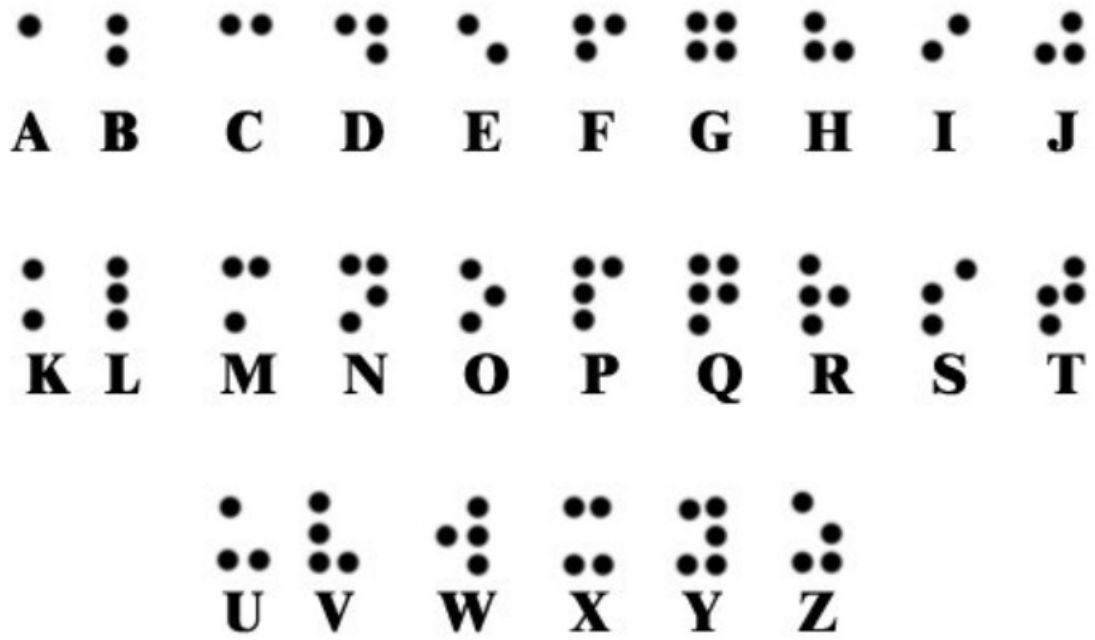
CTF-安恒19年二月月赛部分writeup

MISC1-来玩个游戏吧

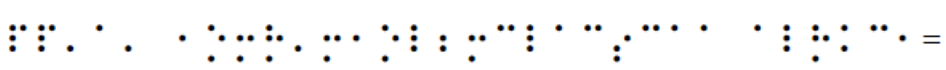
题目：



第一关，一眼可以看出是盲文，之前做过类似题目



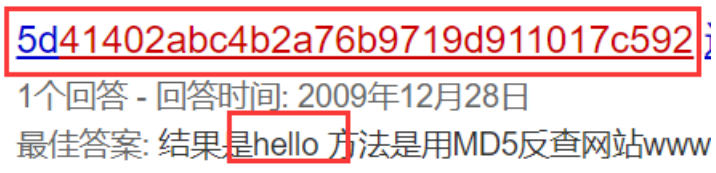
拿到在线网站解一下



加密 解密 使用密码

??41402abc4b2a76b9719d911017c592

??41402abc4b2a76b9719d911017c592，那么就奇怪了，这个？？是什么东西，数一下加上？？正好32位，应该是个MD5了，索性直接百度一下，



第一关答案出来了，试过了MD5值不对，hello是正确的，下一关。

第二关提示

两个文件的md5值一样就可以？

没见过这种的，还是百度一下，

原 使用fastcoll进行md5碰撞，两个不同的文件md5值一样。

2017年06月26日 23:10:12 sysprogram 阅读数：5317

生成两个文件

fastcoll_v1.0.0.5.exe -p C:\windows\notepad.exe -o D:\notepad1.exe D:\notepad2.exe

比较 md5 校验是一样的，但是文件内容不一样。

下载了这个脚本后执行命令

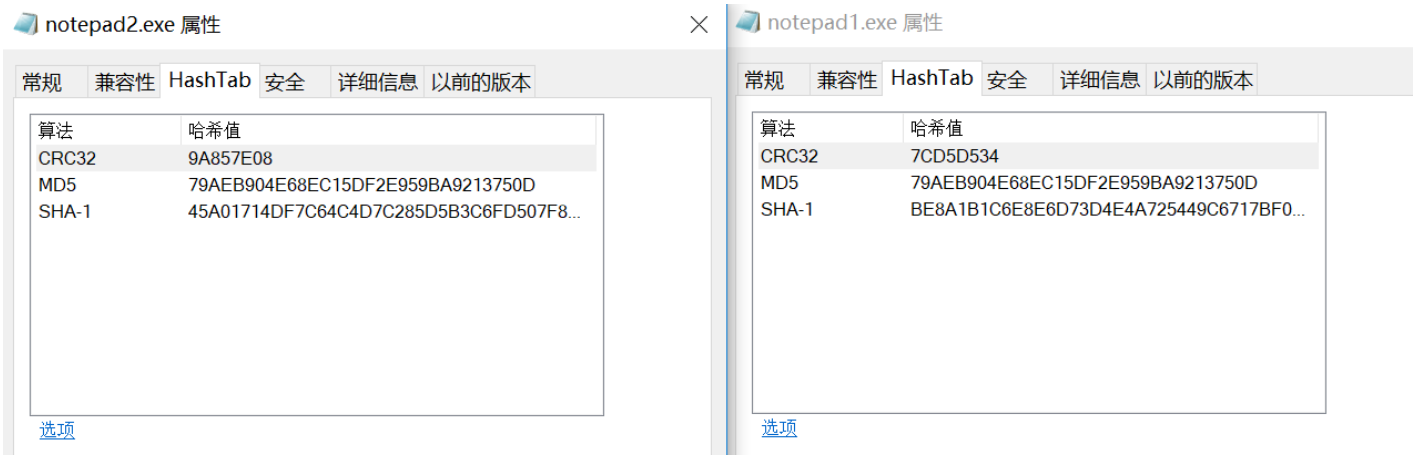
fastcoll_v1.0.0.5.exe -p C:\windows\notepad.exe -o D:\notepad1.exe D:\notepad2.exe

（因为没有规定文件名啥的就直接复制他的命令了）

```
D:\                                    >fastcoll_v1.0.0.5.exe -p C:\windows\notepad.exe -o D:\notepad1.exe D:\notepad2.exe
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'D:\notepad1.exe' and 'D:\notepad2.exe'
Using prefixfile: 'C:\windows\notepad.exe'
Using initial value: d51a7505887a678fec1558bd1f7e30be

Generating first block: ..........
Generating second block: S11...
Running time: 2.765 s
```



直接将文件路径复制到文本框即可


Get Flag
第一关
解出字符串' '的最终结果。（这是什么鬼啊？貌似在公交站经常见到！）
hello
提交
第二关
两个文件的md5值一样就可以？
输入第一个文件的路径
D:\notepad1.exe
输入第二个文件的路径
D:\notepad2.exe
提交
邮件接收区
送你一封包含flag的邮件：
Dear Professional ; Especially for you - this cutting-edge
intelligence ! If you no longer wish to receive our
publications simply reply with a Subject: of "REMOVE"
and you will immediately be removed from our club .
This mail is being sent in compliance with Senate bill
2216 , Title 9 ; Section 306 ! THIS IS NOT MULTI-LEVEL
MARKETING . Why work for somebody else when you can
become rich as few as 35 weeks . Have you ever noticed
more people than ever are surfing the web and people
will do almost anything to avoid mailing their bills

Dear Professional ; Especially for you - this cutting-edge intelligence ! If you no longer wish to receive our publications simply reply with a Subject: of "REMOVE" and you will immediately be removed from our club . This mail is being sent in compliance with Senate bill 2216 , Title 9 ; Section 306 ! THIS IS NOT MULTI-LEVEL MARKETING . Why work for somebody else when you can become rich as few as 35 weeks . Have you ever noticed more people than ever are surfing the web and people will do almost anything to avoid mailing their bills . Well, now is your chance to capitalize on this ! WE will help YOU decrease perceived waiting time by 120% & decrease perceived waiting time by 140% . You can begin at absolutely no cost to you . But don't believe us ! Mrs Jones of Minnesota tried us and says "I was skeptical but it worked for me" . We assure you that we operate within all applicable laws . Because the Internet operates on "Internet time" you must act now ! Sign up a friend and your friend will be rich too . Warmest regards . Dear Cybercitizen , We know you are interested in receiving red-hot announcement ! We will comply with all removal requests ! This mail is being sent in compliance with Senate bill 1619 ; Title 2 ; Section 301 . This is NOT unsolicited bulk mail ! Why work for somebody else when you can become rich within 53 MONTHS ! Have you ever noticed more people than ever are surfing the web and more people than ever are surfing the web . Well, now is your chance to capitalize on this . We will help you use credit cards on your website plus decrease perceived waiting time by 150% . The best thing about our system is that it is absolutely risk free for you ! But don't believe us ! Mrs Simpson of Washington tried us and says "Now I'm rich, Rich, RICH" . We assure you that we operate within all applicable laws ! We beseech you - act now ! Sign up a friend and your friend will be rich too . Thank-you for your serious consideration of our offer ! Dear Friend ; This letter was specially selected to be sent to you ! If you no longer wish to receive our publications simply reply with a Subject: of "REMOVE" and you will immediately be removed from our mailing list . This mail is being sent in compliance with Senate bill 2716 , Title 2 ; Section 306 ! This is a ligitimate business proposal . Why work for somebody else when you can become rich inside 33 weeks . Have you ever noticed more people than ever are surfing the web plus more people than ever are surfing the web . Well, now is your chance to capitalize on this ! WE will help YOU SELL MORE and process your orders within seconds . You can begin at absolutely no cost to you . But don't believe us ! Mrs Jones of Kentucky tried us and says "I was skeptical but it worked for me" ! This offer is 100% legal ! We implore you - act now . Sign up a friend and you'll get a discount of 50% . God Bless .

题目提示了：需要一个在线的网站去解密，而这个网站使用了栅格密码。

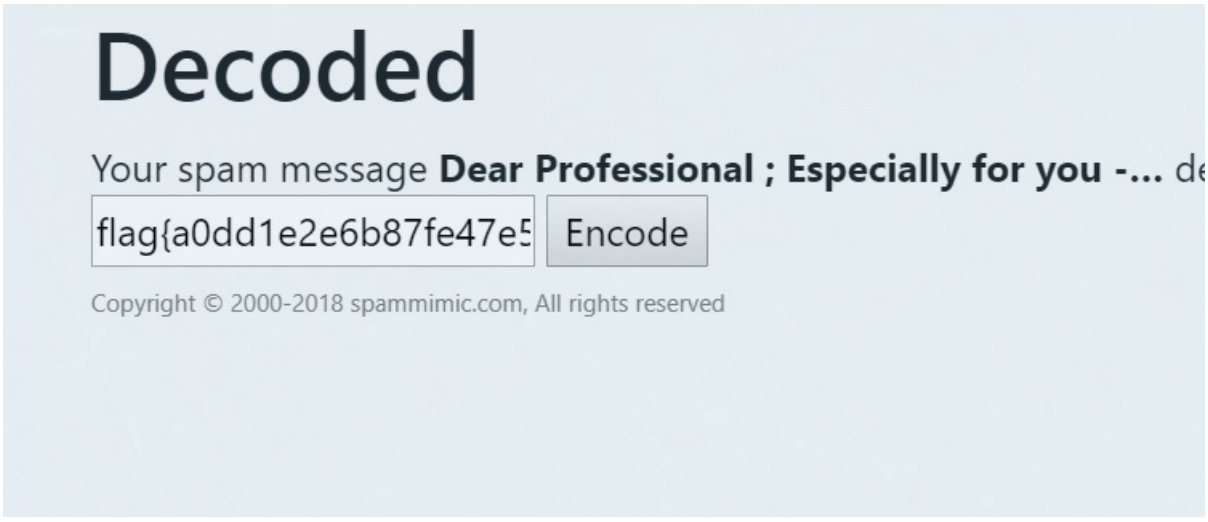
栅格密码也没听说过，还是百度

卡尔达诺栅格密码沿用至今，例如，“Spam Mimic”这个网站就使用了栅格密码，不过这个网站的程序不是用来发现隐藏消息的，而是用来加密和解密（如图1.12所示）。

图 1.12 Spam Mimic 解密

这个网站的目的是模拟一封看似是垃圾邮件却包含隐藏消息的邮件。人们每天都会收到大量的垃圾邮件，除非有人知道其中某封邮件会包含隐藏消息，否则就会将其当作垃圾邮件处理掉了。但密信接收者是知道的，他会登录spam Mimic网站，输入邮件内容，得到隐藏消息。

搜索关键字Spam Mimic到网站 http://www.spammimic.com/解码

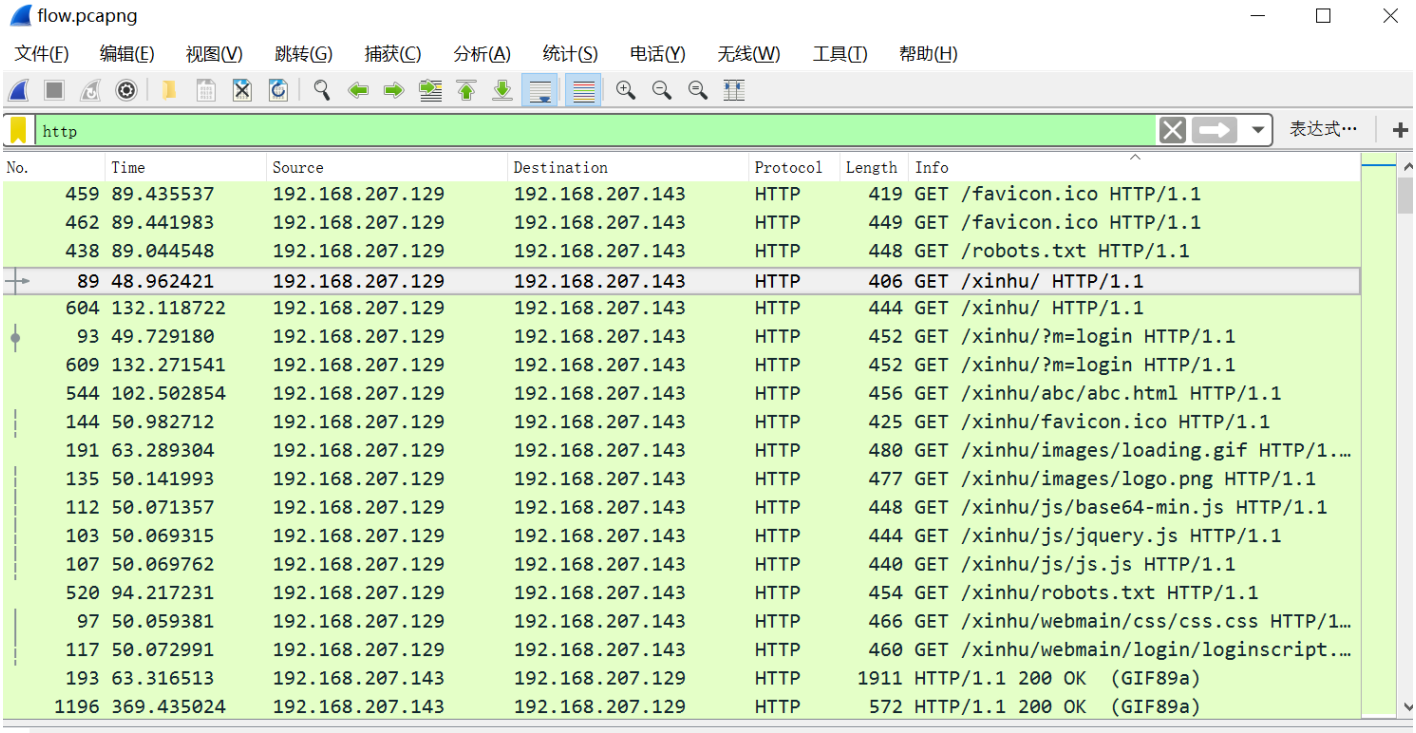


flag为：flag{a0dd1e2e6b87fe47e5ad0184dc291e04}

MISC2-简单的流量分析

题目：

过滤http协议，按照info排序一下



发现存在/xinhu/robots.txt

追踪http流到/xinhu/robots.txt

```
GET /xinhu/robots.txt HTTP/1.1
Host: 192.168.207.143
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:31.0) Gecko/20100101 Firefox/
31.0 Iceweasel/31.5.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: deviceid=1537883014341; PHPSESSID=8aqvfladb5epcd3ije75oc5163
Connection: keep-alive

HTTP/1.1 200 OK
Date: Tue, 25 Sep 2018 15:32:58 GMT
Server: Apache/2.2.22 (Debian)
Last-Modified: Tue, 25 Sep 2018 13:45:26 GMT
ETag: "6ca06-24-576b2516e6980"
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 53
Keep-Alive: timeout=5, max=97
Connection: Keep-Alive
Content-Type: text/plain

User-agent:*
Disallow: /abc/abc.html
```

发现abc.html,继续跟进

```
GET /xinhu/abc/abc.html HTTP/1.1
Host: 192.168.207.143
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:31.0) Gecko/20100101 Firefox/
31.0 Iceweasel/31.5.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: deviceid=1537883014341; PHPSESSID=8aqvfladb5epcd3ije75oc5163
Connection: keep-alive

HTTP/1.1 200 OK
Date: Tue, 25 Sep 2018 15:33:06 GMT
Server: Apache/2.2.22 (Debian)
Last-Modified: Tue, 25 Sep 2018 14:13:58 GMT
ETag: "143053-ac-576b2b7797580"
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 144
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html

md5 0x99a98e067af6b09e64f3740767096c96

DES 0xb19b21e80c685bcb052988c11b987802d2f2808b2c2d8a0d          (129->143)

DES 0x684a0857b767672d52e161aa70f6bdd07c0264876559cb8b          (143->129)
```

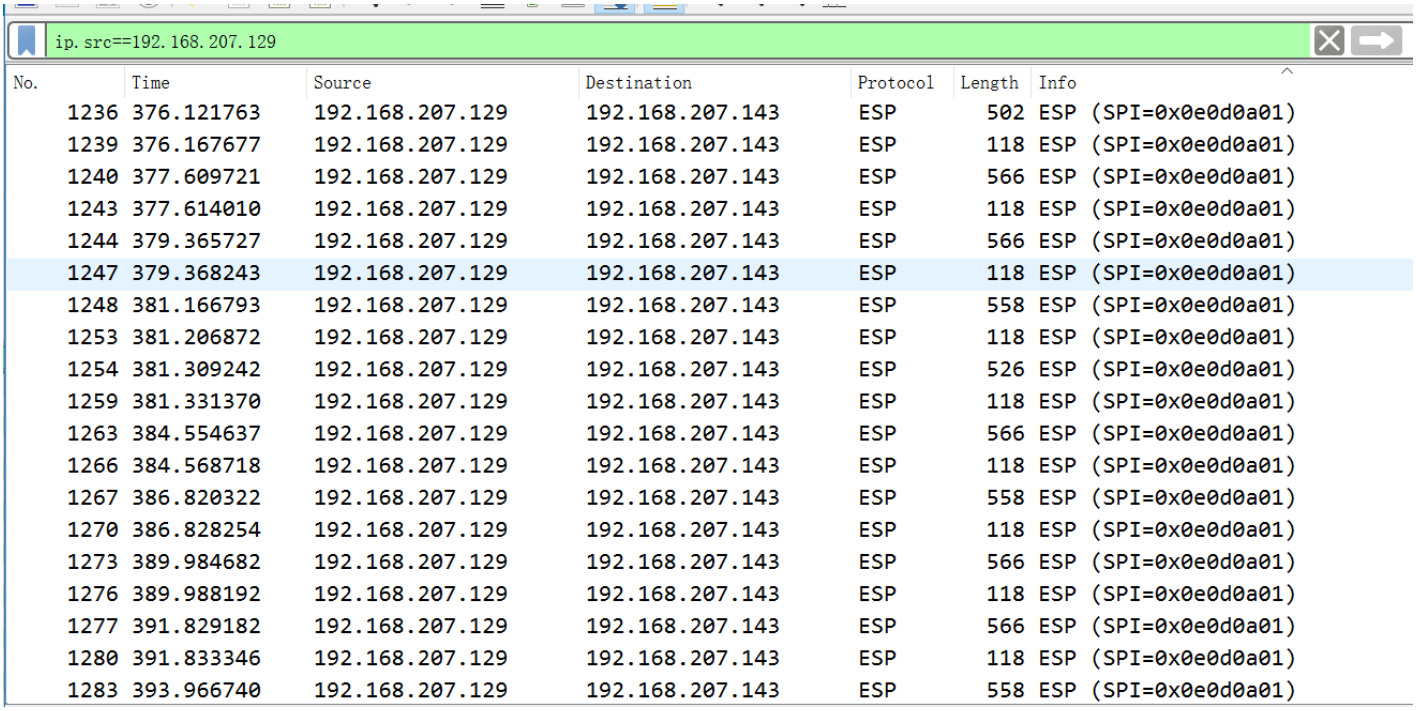
发现MD5和两串DES

```
md5 0x99a98e067af6b09e64f3740767096c96

DES 0xb19b21e80c685bcb052988c11b987802d2f2808b2c2d8a0d    (129->143)

DES 0x684a0857b767672d52e161aa70f6bdd07c0264876559cb8b    (143->129)
```

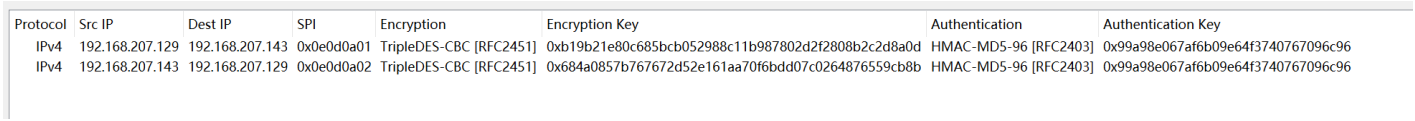
继续向下分析，发现都是IPSec加密后的流量，尝试使用前面给的MD5和DES解密



ip. src==192. 168. 207. 129

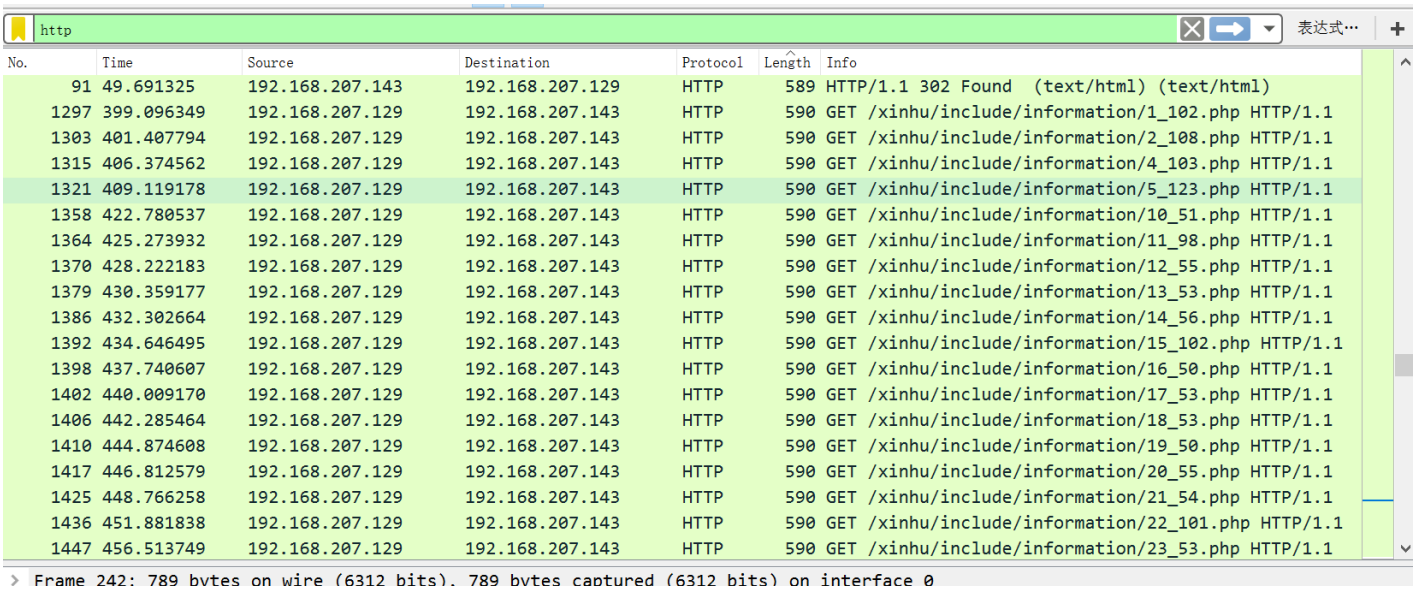
No.	Time	Source	Destination	Protocol	Length	Info
1236	376.121763	192.168.207.129	192.168.207.143	ESP	502	ESP (SPI=0x0e0d0a01)
1239	376.167677	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1240	377.609721	192.168.207.129	192.168.207.143	ESP	566	ESP (SPI=0x0e0d0a01)
1243	377.614010	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1244	379.365727	192.168.207.129	192.168.207.143	ESP	566	ESP (SPI=0x0e0d0a01)
1247	379.368243	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1248	381.166793	192.168.207.129	192.168.207.143	ESP	558	ESP (SPI=0x0e0d0a01)
1253	381.206872	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1254	381.309242	192.168.207.129	192.168.207.143	ESP	526	ESP (SPI=0x0e0d0a01)
1259	381.331370	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1263	384.554637	192.168.207.129	192.168.207.143	ESP	566	ESP (SPI=0x0e0d0a01)
1266	384.568718	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1267	386.820322	192.168.207.129	192.168.207.143	ESP	558	ESP (SPI=0x0e0d0a01)
1270	386.828254	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1273	389.984682	192.168.207.129	192.168.207.143	ESP	566	ESP (SPI=0x0e0d0a01)
1276	389.988192	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1277	391.829182	192.168.207.129	192.168.207.143	ESP	566	ESP (SPI=0x0e0d0a01)
1280	391.833346	192.168.207.129	192.168.207.143	ESP	118	ESP (SPI=0x0e0d0a01)
1283	393.966740	192.168.207.129	192.168.207.143	ESP	558	ESP (SPI=0x0e0d0a01)

wireshark进入Preference菜单下的Profile，找到ESP， 配置如下：



Protocol	Src IP	Dest IP	SPI	Encryption	Encryption Key	Authentication	Authentication Key
IPv4	192.168.207.129	192.168.207.143	0x0e0d0a01	TripleDES-CBC [RFC2451]	0xb19b21e80c685bcb052988c11b987802d2f2808b2c2d8a0d	HMAC-MD5-96 [RFC2403]	0x99a98e067af6b09e64f3740767096c96
IPv4	192.168.207.143	192.168.207.129	0x0e0d0a02	TripleDES-CBC [RFC2451]	0x684a0857b767672d52e161aa70f6bdd07c0264876559cb8b	HMAC-MD5-96 [RFC2403]	0x99a98e067af6b09e64f3740767096c96

此时再次过滤http发现有部分响应包带上了数字，102 108 转换为ASCII码则为f l 所以统一提取转换。



http

表达式…

No.	Time	Source	Destination	Protocol	Length	Info
91	49.691325	192.168.207.143	192.168.207.129	HTTP	589	HTTP/1.1 302 Found (text/html) (text/html)
1297	399.096349	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/1_102.php HTTP/1.1
1303	401.407794	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/2_108.php HTTP/1.1
1315	406.374562	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/4_103.php HTTP/1.1
1321	409.119178	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/5_123.php HTTP/1.1
1358	422.780537	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/10_51.php HTTP/1.1
1364	425.273932	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/11_98.php HTTP/1.1
1370	428.222183	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/12_55.php HTTP/1.1
1379	430.359177	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/13_53.php HTTP/1.1
1386	432.302664	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/14_56.php HTTP/1.1
1392	434.646495	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/15_102.php HTTP/1.1
1398	437.740607	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/16_50.php HTTP/1.1
1402	440.009170	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/17_53.php HTTP/1.1
1406	442.285464	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/18_53.php HTTP/1.1
1410	444.874608	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/19_50.php HTTP/1.1
1417	446.812579	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/20_55.php HTTP/1.1
1425	448.766258	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/21_54.php HTTP/1.1
1436	451.881838	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/22_101.php HTTP/1.1
1447	456.513749	192.168.207.129	192.168.207.143	HTTP	590	GET /xinhu/include/information/23_53.php HTTP/1.1

> Frame 242: 789 bytes on wire (6312 bits), 789 bytes captured (6312 bits) on interface 0

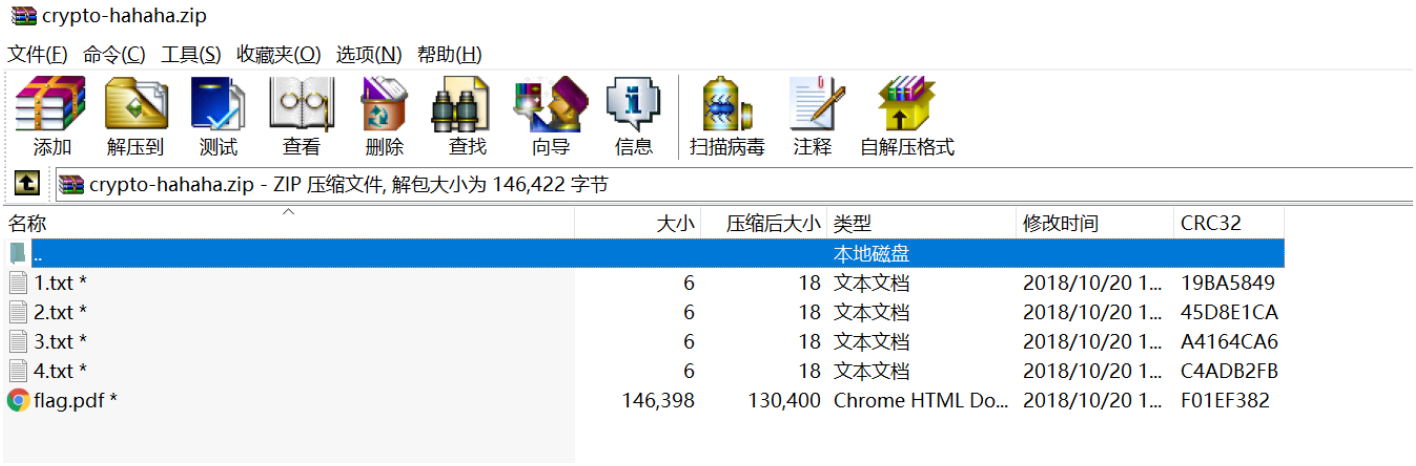
```
a =
[102,108,97,103,123,50,55,98,48,51,98,55,53,56,102,50,53,53,50,55,54,101,53,97,57,56,100,97,48,101,49,57,52,
55,98,101,100,125]
flag = ''
for i in a:
    flag +=chr(i)
print flag
```

```
flag{27b03b758f255276e5a98da0e1947bed}
[Finished in 0.1s]
```

flag：flag{27b03b758f255276e5a98da0e1947bed}

CRYPTO1-hahaha

题目：



压缩包题目，其实看到这压缩包里的短位CRC32应该就能猜出是CRC32爆破了

当然也可以一步一步排除一下

首先binwalk分析得出非伪加密，爆破的话没有提示，不理想。

所以直接上脚本

```
D:\渗透测试\PythonScript\crc32-master>python crc32.py reverse  0x19BA5849
4 bytes: {0xad, 0x91, 0x46, 0x6f}
verification checksum: 0x19ba5849 (OK)
alternative: 37mCIb (OK)
alternative: 3GQ2L2 (OK)
alternative: 7CL3MQ (OK)
alternative: BFpJos (OK)
alternative: E_wtEX (OK)
alternative: KPhDyV (OK)
alternative: N8GYgh (OK)
alternative: OTuEx5 (OK)
alternative: QK_w1l (OK)
alternative: SwZihq (OK)
alternative: bICsTv (OK)
alternative: o75QGt (OK)
alternative: sxiPSt (OK)
alternative: tanny_ (OK)
```

```
D:\渗透测试\PythonScript\crc32-master>python crc32.py reverse  0x45D8E1CA
4 bytes: {0xc8, 0x94, 0x8f, 0x4e}
verification checksum: 0x45d8e1ca (OK)
alternative: 3jAkZV (OK)
alternative: 5oH8jp (OK)
alternative: Aj5pSK (OK)
alternative: U30NJw (OK)
alternative: U_CcN3 (OK)
alternative: XMFlYu (OK)
alternative: daZyr4 (OK)
alternative: is_ver (OK)
alternative: njXHOY (OK)
alternative: pHc6nX (OK)
alternative: uPpZu6 (OK)
alternative: v14u72 (OK)
alternative: xBuUbp (OK)
alternative: yB4dyi (OK)
```

```
D:\渗透测试\PythonScript\crc32-master>python crc32.py reverse  0xA4164CA6
4 bytes: {0x47, 0xf9, 0x91, 0x31}
verification checksum: 0xa4164ca6 (OK)
alternative: 5nQesx (OK)
alternative: 6sKZXO (OK)
alternative: AwcqKW (OK)
alternative: DRaQ8a (OK)
alternative: LXw24I (OK)
alternative: PFIRLU (OK)
alternative: TBTSM6 (OK)
alternative: ZMKcq8 (OK)
alternative: axPHpR (OK)
alternative: faWvZy (OK)
alternative: hnHFfw (OK)
alternative: pIzkwP (OK)
alternative: pU57vD (OK)
alternative: r8RHCE (OK)
alternative: tMgjv3 (OK)
alternative: vLs8Gv (OK)
alternative: y_beau (OK)
```

```
D:\渗透测试\PythonScript\crc32-master>python crc32.py reverse  0xC4ADB2FB
4 bytes: {0x97, 0x02, 0xd5, 0x61}
verification checksum: 0xc4adb2fb (OK)
alternative: 3rHelj (OK)
alternative: GkzqTE (OK)
alternative: IdeAhK (OK)
alternative: QCW1y1 (OK)
alternative: VZPRSG (OK)
alternative: XUOboI (OK)
alternative: ikVxSN (OK)
alternative: nrQFye (OK)
alternative: pPj8Xd (OK)
alternative: qLdUBi (OK)
alternative: rM16h5 (OK)
alternative: tifu1_ (OK)
alternative: vUckhB (OK)
alternative: yFr6NA (OK)
```

所以加起来就是tanny_is_very_beautifu1_

哈哈

按照给的提示，flag应该是flag{1or! 2or@ sechn}

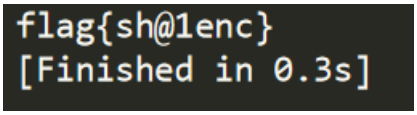
然后给了sha1值，应该是要爆破了。。。

当时做到这里就停了，因为不会写脚本了

下面献上一叶飘零大佬的脚本

```
import itertools
import hashlib

def sha1(str):
    sha = hashlib.sha1(str)
    encrypts = sha.hexdigest()
    return encrypts
a1 = '1!'
a2 = '2@'
a3 = '{'
a4 = '}'
for str1 in itertools.combinations(a1,1):
    for str2 in itertools.combinations(a2,1):
        str3 = str1[0]+str2[0]+'sechn'
        for i in itertools.permutations(str3):
            tmp = (''.join(i))
            res = 'flag{'+tmp+'}'
            # print sha1(res)
            if sha1(res) == 'e6079c5ce56e781a50f4bf853cdb5302e0d8f054':
                print res
                break
```



flag:flag{sh@1enc}

小结：web没做出来太菜，pwn刚起步，压根没看，密码2也没做出来，需要的脑洞太大了，另外膜飘零师傅。

参考：https://www.anquanke.com/post/id/171543

转载于:https://www.cnblogs.com/pureqh/p/10435229.html