

# ctfshow F5杯 部分WP(writeup) 超详细

原创

是Mumuzi  于 2021-02-24 11:38:59 发布  1724  收藏 8

分类专栏: [ctf ctfshow](#) 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/qq\\_42880719/article/details/114009106](https://blog.csdn.net/qq_42880719/article/details/114009106)

版权



[ctf 同时被 2 个专栏收录](#)

75 篇文章 28 订阅

订阅专栏



[ctfshow](#)

23 篇文章 8 订阅

订阅专栏

本文共4370字，147段落，全文看完预计用时10分钟  
这次F5杯的misc难度感觉比大吉杯难了许多，出题人的脑洞太大了  
在这里感谢各位大师傅群里的随缘hint（水群大胜利）  
写的非常详细，可以跟着实际操作，所以最后不会贴上静态flag

## WEB

lastward's website 100	eazy-unserialize ✓ 100	迷惑行为大赏之盲注 100	Web逃离计划 100
未完成的项目 100	eazy-unserialize-revenge ✓		

[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

## MISC

Just Another 拼图 100	大小二维码 ✓ 100	填字游戏 ✓ 100	过年了 100
天书奇谈 100	牛年大吉3.0 ✓ 100	两行代码一纸情书 ✓ 100	F5还会学中文 ✓ 100
F5也会LSB ✓ 100	GoodNight ✓ 100		

[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

## CRYPTO

网络是有记忆的 100	简单的古典密码 100	SoEZecdsa 100
----------------	----------------	------------------

[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

web

[eazy-unserialize & eazy-unserialize-revenge](#)

misc

[大小二维码](#)

[填字游戏](#)

[牛年大吉3.0](#)

[两行代码一纸情书](#)

[F5也会LSB](#)

[F5还会学中文](#)

[GoodNight](#)

WEB:

## eazy-unserialize & eazy-unserialize-revenge

一个payload打通两道，所以就放在一起

前半部分估计是某个登录页面的执行代码，为干扰项，重点在后半部分

```
class Happy{
    public $file='flag.php';

    function __destruct(){
        if(!empty($this->file)) {
            include $this->file;
        }
    }
}

function ezwaf($data){
    if (preg_match("/ctfshow/", $data)){
        die("Hacker !!!");
    }
    return $data;
}
if(isset($_GET["w_a_n"])) {
    @unserialize(ezwaf($_GET["w_a_n"]));
} else {
    new CTFSHOW("lookme", array());
}
```

存在文件包含漏洞，使用php为协议读取flag.php:

```
payload:?w_a_n=O:5:"Happy":1:{s:4:"file";s:57:"php://filter/read=convert.base64-encode/resource=flag.php";}
```

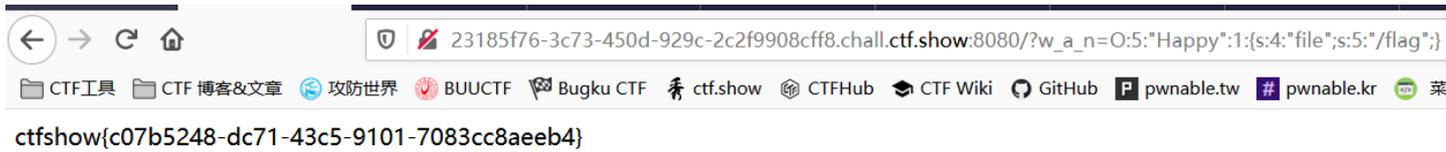
读取到  
PD9waHANCiFkZWZpbmVkkCdIYXBweScplCYmIGV4aXQoJ0FjY2VzcyBEZW5pZWQnKTsNCmVjaG8gZmlsZV9nZXRFY29udG  
VudHMoli9mbGFnlk7DQoNCj8+  
进行base64

```
<?php  
!defined('Happy') && exit('Access Denied');  
echo file_get_contents("/flag");  
  
?>
```

flag在/flag下，修改一下payload

```
?w_a_n=O:5:"Happy":1:{s:4:"file";s:5:"/flag";}  
好像这个payload被修了，那就和之前读flag.php一样用php协议读取  
payload:?w_a_n=O:5:"Happy":1:{s:4:"file";s:54:"php://filter/read=convert.base64-encode/resource=/flag";}
```

得到flag



[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

## MISC

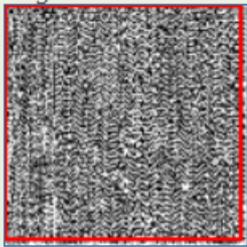
### 大小二维码

八神师傅的创意题，脑洞也还是大  
首先得到一张超大的二维码，用手机QQ扫码只能显示部分，但是开头是7z。猜测是将7z压缩包数据写进了二维码，  
使用barcode扫码将十六进制数据复制下来

<https://online-barcode-reader.inliteresearch.com/>

Barcode: 1 of 1      Type: QR      Page 1 of 1  
Length: 1037      Rotation: none  
Module: 13.0pix      Rectangle: {X=19,Y=19,Width=2287,Height=2287}

Barcode Text processing:  
Converted Character Set: ISO-8859-1  
Formatted: specialChar



Binary Data in barcode (Hex-ASCII display)

0000	37 7a bc af 27 1c 00 04	2c c5 12 90 ca 03 00 00	7z~'~,,~ ~~~~~
0010	00 00 00 00 23 00 00 00	00 00 00 00 eb d1 14 9c	~~~~#~ ~~~~~
0020	e0 14 c7 02 41 5d 00 44	94 05 c4 7a 27 f6 f7 ee	~~~~A]~D~ ~z'~ ~
0030	89 8e 50 90 88 b3 aa d5	50 25 8f 59 12 4a 87 79	~~P~ ~ ~P%~Y~]~y
0040	a5 b2 49 6a 46 6c 4f af	1a 7b 05 10 3e 2b 65 8d	~~IjF10~ ~{~ ~>+e~
0050	2e bc 8f 69 37 3c 64 69	f0 98 21 d8 fb 1d a5 9a	.~i7<di~ ~!~ ~~~~~
0060	65 33 49 d0 1e ab c1 e5	e7 90 a5 0b 7c 8d 45 60	e3I~ ~ ~ ~ ~ ~ ~ ~   ~E`
0070	23 59 2c e0 99 fb 79 3e	7d f2 d1 56 13 66 3a 0c	#Y,~ ~ ~y>}~ ~V~f:~

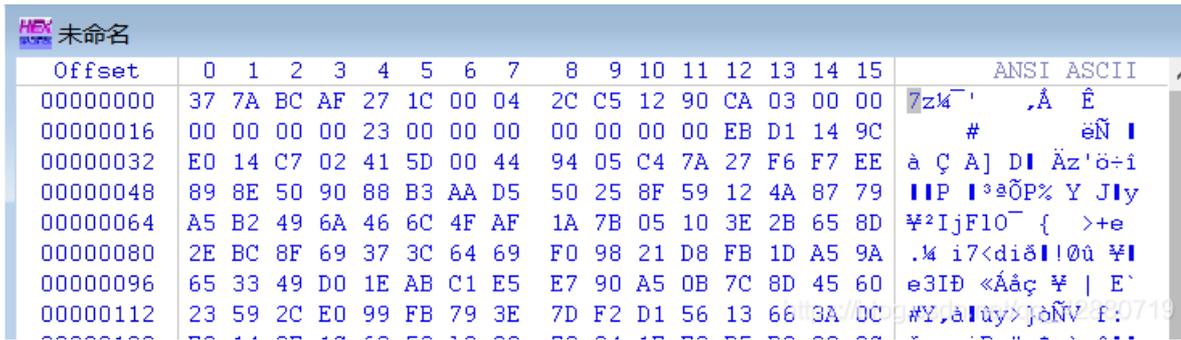
```

0080 f0 14 2e 1c 69 50 a0 22 72 24 1e e0 b5 b3 99 8c | ~.~iP~"r$~ ~ ~ ~ ~ |
0090 4b 14 35 f8 55 f5 c9 dd 91 e8 cf fd 44 44 95 fd | K~5~U~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
00a0 79 bd 8e 4b 0f ec 15 a7 5c e1 1c fa 14 db 83 ef | y~n~K~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
00b0 cd e2 ff b1 f3 c5 e7 5d 91 8d 3d 43 37 5c f4 a5 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
00c0 78 74 cb 8c 27 4c 21 44 99 89 98 90 49 ed 20 a1 | xt~'~L!~D~ ~ ~ ~ ~ ~ ~ ~ ~ |
00d0 f0 e7 65 ea 49 67 8b 2b 8e f9 19 44 3e d2 11 bd | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
00e0 d6 9f 06 cb 75 77 1d 3b 20 5f 85 3c ee df 1c 80 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
00f0 dc 54 ae b9 a5 c2 86 a9 ac 1a 9a b9 45 21 7a 4f | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
0100 15 8d f3 9a 99 0e 20 46 81 59 d7 e3 3c 2c 9a 24 | ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
0110 76 7c 3b 0b b0 29 9e 86 91 48 64 15 3d 60 75 0e | v|;~)~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ |
0120 a8 23 07 70 eb 71 21 7b 7d 53 6c fe f5 7a 26 5a | ~#~p~q!{~}S1~ ~ ~ ~ ~ ~ ~ ~ ~ |

```

将全部

复制下来，只保留十六进制信息，然后使用winhex写入，保存为7z文件。



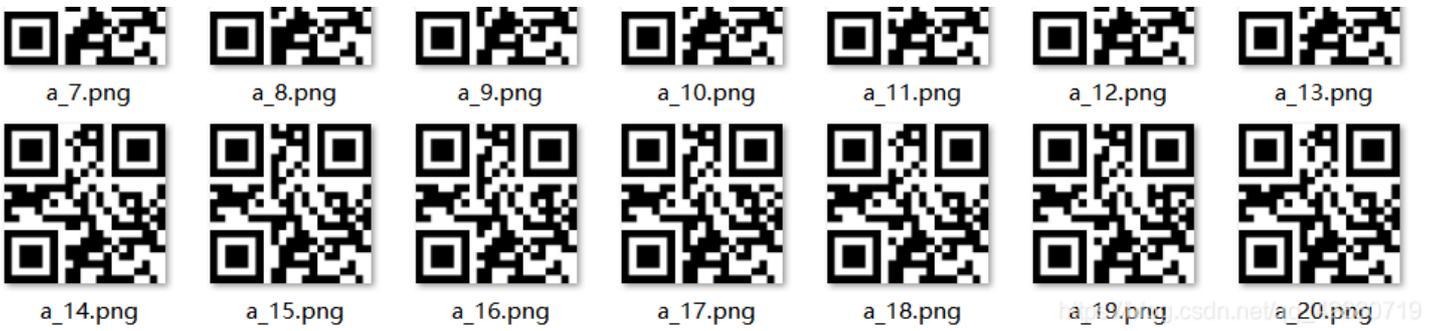
解压得到35个小二维

码



使用PS把小二维码放大，得到





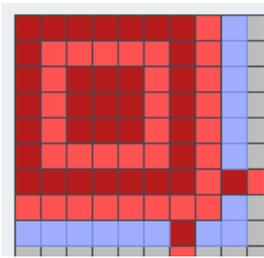
首先扫几张码，发现只有一张二维码内容是“nothing here~”，其他都为乱码

再次观察可以发现，所有的数据区和效验区都是一样的，唯一不相同的就是每个定位点旁边的掩码类型。

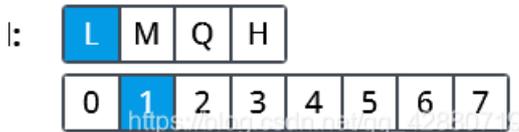
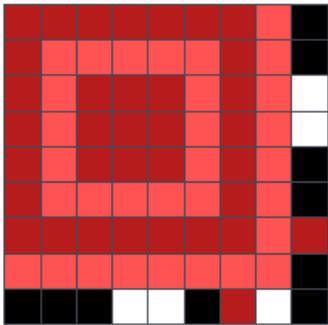
首先使用QRazyBo扫第一张二维码，查看他的Mask Pattern

<https://merricx.github.io/qrazybox/>

可以发现纠错等级都是L，但是显示掩码类型都为3，所以在这里生成一张不含任何信息的二维码，再去点击蓝色区域



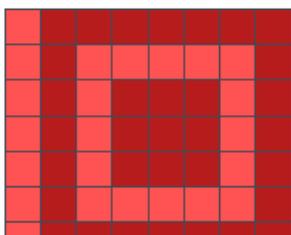
将纠错等级调为1，寻找掩码等级与刚刚那张图一样的

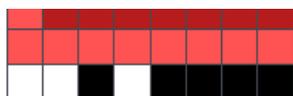


发现1与刚刚左上角的一样

再对比右上，发现4和之前的一样

Top Right ▾





L	M	Q	H
---	---	---	---

0	1	2	3	4	5	6	7
---	---	---	---	---	---	---	---

[https://blog.csdn.net/qz\\_42880719](https://blog.csdn.net/qz_42880719)

所以这里手动将35张图全部手动对比下来，得到一串0-7的数字，猜想是8进制，并且3位为一组并转为ascii，写个python脚本进行进制转换

```
import re
flag = ''
n = 0
p = input('input a octal number:\n')
arr = re.findall(r'.{3}', p)
try:
    for i in arr:
        print(chr(int(i,8)))
        flag += chr(int(i,8))
except:
    print("字符串中包含不能每3位进行转换的数字")
print(flag)
```

得到flag

```
9
-
H
a
0
}
ctfshow{Bu. }
Process finished with exit code 0
https://blog.csdn.net/qz\_42880719
```

## 填字游戏

U1S1，这次填字游戏非常良心，特别容易。首先用QQ的识别功能将英文全部提取下来

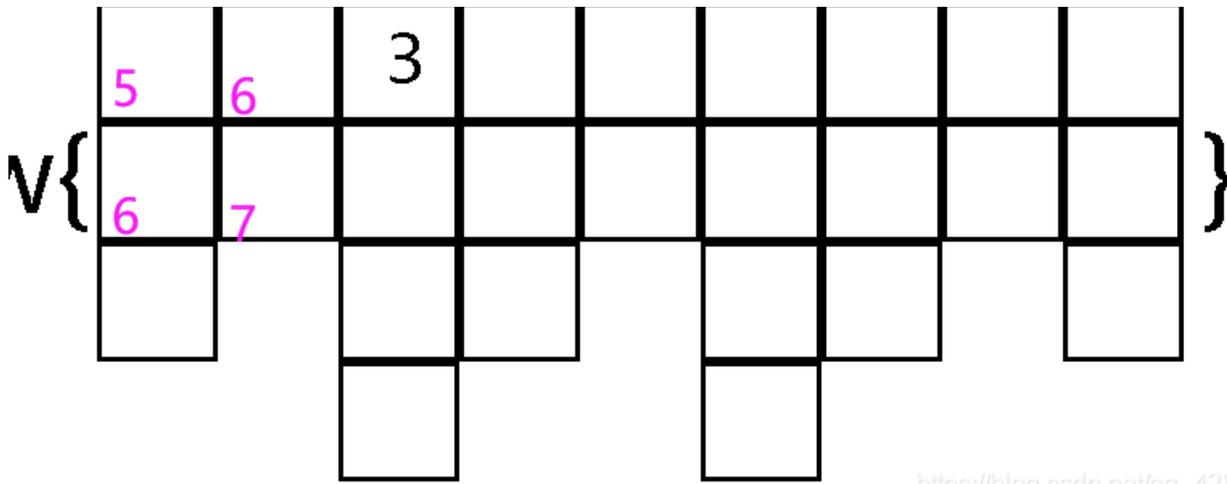
1.  $a(n) = n*(n^2+1)/2$
2. Number of balls in pyramid with base either a regular hexagon or a hexagon with alternate sides differing by 1
3. Initial members of prime sextuplets  $(p, p+4, p+6, p+10, p+12, p+16)$
4. Primes  $p$  such that neither  $p-2$  nor  $p+2$  is prime
5. Smith (or joke) numbers
6. the number of distinct reduced words of length  $n$  in the Coxeter group of "Apollonian reflections" in three dimensions
7. Hyperfactorials
8. Number of factorization patterns of polynomials of degree  $n$  over integers
9. Initial members of prime triples  $(p, p+4, p+6)$

这里随便拿两个来举例

首先拿第二个

因为是纯英文，就不用想百度了，直接上google。





[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

在此平台继续搜，特别注意，第一个也要在这个平台搜，不要自己代值进去，因为代值进去应该都不会想到先代数字0...

```
A006003      a(n) = n*(n^2 + 1)/2.
              (Formerly M3849)
0, 1, 5, 15, 34, 65, 111, 175, 260, 369, 504,
5335, 6095, 6924, 7825, 8801, 9855, 10990, 12205,
13495, 14860, 16290, 17785, 19345, 20970, 22660, 24415, 26235, 28120, 30070, 32085, 34165, 36310, 38520, 40795, 43135, 45540, 48010, 50545, 53145, 55810, 58540, 61335, 64195, 67120, 70110, 73165, 76285, 79470, 82720, 86035, 89415, 92860, 96370, 100045, 103885, 107790, 111760, 115795, 119895, 124060, 128290, 132585, 136945, 141370, 145860, 150415, 155035, 159720, 164470, 169285, 174165, 179110, 184120, 189195, 194335, 199540, 204810, 210145, 215545, 220910, 226340, 231835, 237395, 243020, 248710, 254465, 260285, 266170, 272120, 278135, 284215, 290360, 296570, 302845, 309185, 315590, 322060, 328595, 335195, 341860, 348590, 355385, 362245, 369170, 376160, 383215, 390335, 397520, 404770, 412085, 419465, 426910, 434420, 441995, 449635, 457340, 465110, 472945, 480845, 488810, 496840, 504935, 513095, 521320, 529610, 537965, 546385, 554870, 563420, 572035, 580715, 589460, 598270, 607145, 616085, 625090, 634160, 643295, 652495, 661760, 671090, 680485, 689945, 699470, 709060, 718715, 728435, 738220, 748070, 757985, 767965, 778010, 788120, 798295, 808535, 818840, 829210, 839645, 850145, 860710, 871340, 882035, 892795, 903620, 914510, 925465, 936485, 947570, 958720, 969935, 981215, 992560, 1003970, 1015445, 1026985, 1038590, 1050255, 1061980, 1073765, 1085610, 1097515, 1109480, 1121505, 1133590, 1145735, 1157940, 1170205, 1182530, 1194915, 1207360, 1219865, 1232430, 1245055, 1257740, 1270485, 1283290, 1296155, 1309080, 1322065, 1335110, 1348215, 1361380, 1374605, 1387890, 1401235, 1414640, 1428105, 1441630, 1455215, 1468860, 1482565, 1496330, 1510155, 1524040, 1537985, 1551990, 1566055, 1580180, 1594365, 1608610, 1622915, 1637280, 1651705, 1666190, 1680735, 1695340, 1710005, 1724730, 1739515, 1754360, 1769265, 1784230, 1799255, 1814340, 1829485, 1844690, 1859955, 1875280, 1890665, 1906110, 1921615, 1937180, 1952805, 1968490, 1984235, 1999940, 2015705, 2031530, 2047415, 2063360, 2079365, 2095430, 2111555, 2127740, 2143985, 2160290, 2176655, 2193080, 2209565, 2226110, 2242715, 2259380, 2276105, 2292890, 2309735, 2326640, 2343605, 2360630, 2377715, 2394860, 2412065, 2429330, 2446655, 2464040, 2481485, 2498990, 2516555, 2534180, 2551865, 2569610, 2587415, 2605280, 2623205, 2641190, 2659235, 2677340, 2695505, 2713730, 2732015, 2750360, 2768765, 2787230, 2805755, 2824340, 2842985, 2861690, 2880455, 2899280, 2918165, 2937110, 2956115, 2975180, 2994305, 3013490, 3032735, 3052040, 3071405, 3090830, 3110315, 3129860, 3149465, 3169130, 3188855, 3208640, 3228485, 3248390, 3268355, 3288380, 3308465, 3328610, 3348815, 3369080, 3389405, 3409790, 3430235, 3450740, 3471305, 3491930, 3512615, 3533360, 3554165, 3575030, 3595955, 3616940, 3637985, 3659090, 3680255, 3701480, 3722765, 3744110, 3765515, 3786980, 3808505, 3830090, 3851735, 3873440, 3895205, 3917030, 3938915, 3960860, 3982865, 4004930, 4027055, 4049240, 4071485, 4093790, 4116155, 4138580, 4161065, 4183610, 4206215, 4228880, 4251605, 4274390, 4297235, 4320140, 4343105, 4366130, 4389215, 4412360, 4435565, 4458830, 4482155, 4505540, 4528985, 4552490, 4576055, 4599680, 4623365, 4647110, 4670915, 4694780, 4718705, 4742690, 4766735, 4790840, 4814995, 4839210, 4863485, 4887810, 4912195, 4936640, 4961145, 4985710, 5010335, 5035020, 5059765, 5084570, 5109435, 5134360, 5159345, 5184390, 5209495, 5234660, 5259885, 5285170, 5310515, 5335920, 5361385, 5386910, 5412495, 5438140, 5463845, 5489610, 5515435, 5541320, 5567265, 5593270, 5619335, 5645460, 5671645, 5697890, 5724195, 5750560, 5776985, 5803470, 5830015, 5856620, 5883285, 5910010, 5936795, 5963640, 5990545, 6017510, 6044535, 6071620, 6098765, 6125970, 6153235, 6180560, 6207945, 6235390, 6262895, 6290460, 6318085, 6345770, 6373515, 6401320, 6429185, 6457110, 6485095, 6513140, 6541245, 6569410, 6597635, 6625920, 6654265, 6682670, 6711135, 6739660, 6768245, 6796890, 6825595, 6854360, 6883185, 6912070, 6941015, 6970020, 6999085, 7028210, 7057395, 7086640, 7115945, 7145310, 7174735, 7204220, 7233765, 7263370, 7293035, 7322760, 7352545, 7382390, 7412295, 7442260, 7472285, 7502370, 7532515, 7562720, 7592985, 7623310, 7653695, 7684140, 7714645, 7745210, 7775835, 7806520, 7837265, 7868070, 7898935, 7929860, 7960845, 7991890, 8022995, 8054160, 8085385, 8116670, 8148015, 8179420, 8210885, 8242410, 8273995, 8305640, 8337345, 8369110, 8400935, 8432820, 8464765, 8496770, 8528835, 8560960, 8593145, 8625390, 8657695, 8689960, 8722285, 8754670, 8787115, 8819620, 8852185, 8884810, 8917495, 8950240, 8983045, 9015910, 9048835, 9081820, 9114865, 9147970, 9181135, 9214360, 9247645, 9280990, 9314395, 9347860, 9381385, 9414970, 9448615, 9482320, 9516085, 9549910, 9583795, 9617740, 9651745, 9685810, 9719935, 9754120, 9788365, 9822670, 9857035, 9891460, 9925945, 9960490, 9995095, 10029760, 10064485, 10099270, 10134115, 10169020, 10203985, 10239010, 10274095, 10309240, 10344445, 10379710, 10415035, 10450420, 10485865, 10521370, 10556935, 10592560, 10628245, 10663990, 10709795, 10745660, 10781585, 10817570, 10853615, 10889720, 10925885, 10962110, 10998395, 11034740, 11071145, 11107610, 11144135, 11180720, 11217365, 11254070, 11290835, 11327660, 11364545, 11401490, 11438495, 11475560, 11512685, 11549870, 11587115, 11624420, 11661785, 11699210, 11736695, 11774240, 11811845, 11849510, 11887235, 11925020, 11962865, 12000770, 12038735, 12076760, 12114845, 12152990, 12191195, 12229460, 12267785, 12306170, 12344615, 12383120, 12421685, 12460310, 12498995, 12537740, 12576545, 12615410, 12654335, 12693320, 12732365, 12771470, 12810635, 12849860, 12889145, 12928490, 12967895, 13007360, 13046885, 13086470, 13126115, 13165820, 13205585, 13245410, 13285295, 13325240, 13365245, 13405310, 13445435, 13485620, 13525865, 13566170, 13606535, 13646960, 13687445, 13727990, 13768595, 13809260, 13849985, 13890770, 13931615, 13972520, 14013485, 14054510, 14095595, 14136740, 14177945, 14219210, 14260535, 14301920, 14343365, 14384870, 14426435, 14468060, 14509745, 14551490, 14593295, 14635160, 14677085, 14719070, 14761115, 14803220, 14845385, 14887610, 14929895, 14972240, 15014645, 15057110, 15099635, 15142220, 15184865, 15227570, 15270335, 15313160, 15356045, 15398990, 15442005, 15485080, 15528215, 15571410, 15614665, 15657980, 15701355, 15744790, 15788285, 15831840, 15875455, 15919130, 15962865, 16006660, 16050515, 16094430, 16138405, 16182440, 16226535, 16270690, 16314905, 16359180, 16403515, 16447910, 16492365, 16536880, 16581455, 16626090, 16670785, 16715540, 16760355, 16805230, 16850165, 16895160, 16940215, 16985330, 17030505, 17075740, 17121035, 17166390, 17211805, 17257280, 17302815, 17348410, 17394065, 17439780, 17485555, 17531390, 17577285, 17623240, 17669255, 17715330, 17761465, 17807660, 17853915, 17900230, 17946605, 17993040, 18039535, 18086090, 18132705, 18179380, 18226115, 18272910, 18319765, 18366680, 18413655, 18460690, 18507785, 18554940, 18602155, 18649430, 18696765, 18744160, 18791615, 18839130, 18886705, 18934340, 18982035, 19029790, 19077605, 19125480, 19173415, 19221410, 19269465, 19317580, 19365755, 19413990, 19462285, 19510640, 19559055, 19607530, 19656065, 19704660, 19753315, 19802030, 19850805, 19899640, 19948535, 19997490, 20046505, 20095580, 20144715, 20193910, 20243165, 20292480, 20341855, 20391290, 20440785, 20490340, 20539955, 20589630, 20639365, 20689160, 20739015, 20788930, 20838905, 20888940, 20939035, 20989190, 21039405, 21089680, 21140015, 21190410, 21240865, 21291380, 21341955, 21392590, 21443285, 21494040, 21544855, 21595730, 21646665, 21697660, 21748715, 21799830, 21850995, 21902220, 21953505, 22004850, 22056255, 22107720, 22159245, 22210830, 22262475, 22314180, 22365945, 22417770, 22469655, 22521590, 22573585, 22625640, 22677755, 22729930, 22782165, 22834460, 22886815, 22939230, 22991705, 23044240, 23096835, 23149490, 23202205, 23254980, 23307815, 23360710, 23413665, 23466680, 23519755, 23572890, 23626085, 23679340, 23732655, 23786030, 23839465, 23892960, 23946515, 24000130, 24053805, 24107540, 24161335, 24215190, 24269105, 24323080, 24377115, 24431210, 24485365, 24539580, 24593855, 24648190, 24702585, 24757040, 24811555, 24866130, 24920765, 24975460, 25030215, 25085030, 25139905, 25194840, 25249835, 25304890, 25359995, 25415160, 25470385, 25525670, 25581015, 25636420, 25691885, 25747360, 25802845, 25858340, 25913845, 25969360, 26024985, 26080720, 26136565, 26192420, 26248385, 26304360, 26360345, 26416340, 26472345, 26528360, 26584385, 26640420, 26696465, 26752520, 26808585, 26864660, 26920745, 26976840, 27032945, 27089060, 27145185, 27201320, 27257465, 27313620, 27369785, 27425960, 27482145, 27538340, 27594545, 27650760, 27706985, 27763220, 27819465, 27875720, 27931985, 27988260, 28044545, 28100840, 28157145, 28213460, 28269785, 28326120, 28382465, 28438820, 28495185, 28551560, 28607945, 28664340, 28720745, 28777160, 28833585, 28889920, 28946265, 29002620, 29058985, 29115360, 29171745, 29228140, 29284545, 29340960, 29397385, 29453820, 29510265, 29566720, 29623185, 29679660, 29736145, 29792640, 29849145, 29905660, 29962185, 30018720, 30075265, 30131820, 30188385, 30244960, 30301545, 30358140, 30414745, 30471360, 30527985, 30584620, 30641265, 30697920, 30754585, 30811260, 30867945, 30924640, 30981345, 31038060, 31094785, 31151520, 31208265, 31265020, 31321785, 31378560, 31435345, 31492140, 31548945, 31605760, 31662585, 31719420, 31776265, 31833120, 31889985, 31946860, 32003745, 32060640, 32117545, 32174460, 32231385, 32288320, 32345265, 32402220, 32459185, 32516160, 32573145, 32630140, 32687145, 32744160, 32801185, 32858220, 32915265, 32972320, 33029385, 33086460, 33143545, 33200640, 33257745, 33314860, 33371985, 33429120, 33486265, 33543420, 33600585, 33657760, 33714945, 33772140, 33829345, 33886560, 33943785, 34001020, 34058265, 34115520, 34172785, 34230060, 34287345, 34344640, 34401945, 34459260, 34516585, 34573920, 34631265, 34688620, 34745985, 34803360, 34860745, 34918140, 34975545, 35032960, 35090385, 35147820, 35205265, 35262720, 35320185, 35377660, 35435145, 35492640, 35550145, 35607660, 35665185, 35722720, 35780265, 35837820, 35895385, 35952960, 36010545, 36068140, 36125745, 36183360, 36240985, 36298620, 36356265, 36413920, 36471585, 36529260, 36586945, 36644640, 36702345, 36760060, 36817785, 36875520, 36933265, 36991020, 37048785, 37106560, 37164345, 37222140, 37279945, 37337760, 37395585, 37453420, 37511265, 37569120, 37626985, 37684860, 37742745, 37800640, 37858545, 37916460, 37974385, 38032320, 38090265, 38148220, 38206185, 38264160, 38322145, 38380140, 38438145, 38496160, 38554185, 38612220, 38670265, 38728320, 38786385, 38844460, 38902545, 38960640, 39018745, 39076860, 39134985, 39193120, 39251265, 39309420, 39367585, 39425760, 39483945, 39542140, 39600345, 39658560, 39716785, 39775020, 39833265, 39891520, 39949785, 40008060, 40066345, 40124640, 40182945, 40241260, 40299585, 40357920, 40416265, 40474620, 40532985, 40591360, 40649745, 40708140, 40766545, 40824960, 40883385, 40941820, 41000265, 41058720, 41117185, 41175660, 41234145, 41292640, 41351145, 41409660, 41468185, 41526720, 41585265, 41643820, 41702385, 41760960, 41819545, 41878140, 41936745, 42000000, 42063260, 42126520, 42189780, 42253040, 42316300, 42379560, 42442820, 42506080, 42569340, 42632600, 42695860, 42759120, 42822380, 42885640, 42948900, 43012160, 43075420, 43138680, 43201940, 43265200, 43328460, 43391720, 43454980, 43518240, 43581500, 43644760, 43708020, 43771280, 43834540, 43897800, 43961060, 44024320, 44087580, 44150840, 44214100, 44277360, 44340620, 44403880, 44467140, 44530400, 44593660, 44656920, 44720180, 44783440, 44846700, 44909960, 44973220, 45036480, 45099740, 45163000, 45226260, 45289520, 45352780, 45416040, 45479300, 45542560, 45605820, 45669080, 45732340, 45795600, 45858860, 45922120, 45985380, 46048640, 46111900, 46175160, 46238420, 46301680, 46364940, 46428200, 46491460, 46554720, 46617980, 46681240, 46744500, 46807760, 46871020, 46934280, 46997540, 47060800, 47124060, 47187320, 47250580, 47313840, 47377100, 47440360, 47503620, 47566880, 47630140, 47693400, 47756
```

8208208820

然后刚刚binwalk还分离出了熟悉的图片，这里看两张图片，能明显发现大小不一样但是差异很小，并且这里给了密钥，0通道没有LSB痕迹就排除了，麻了很久，这里使用的是OurSecret



两张都提取出来，发现大小也不相同，Demo.png略微比Demo7.png大，所以考虑盲水印。

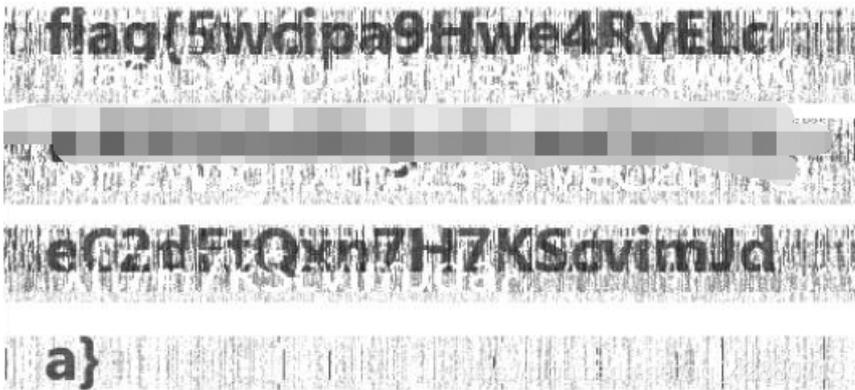
我目前只下载了两种

chishaxie/BlindWaterMark 是错的

linyacool /blind-watermark python3是错的

这里需要使用 linyacool /blind-watermark python2

得到



想要提交发现长度好像太长了，进行base58解码，

得到flag



两行代码一纸情书

Y4的超良心

下载dll附件，使用notepad++打开，直接搜ctf，发现一长串类似编码，复制下来进行base64解码，发现无乱码，一直向下解，得到flag

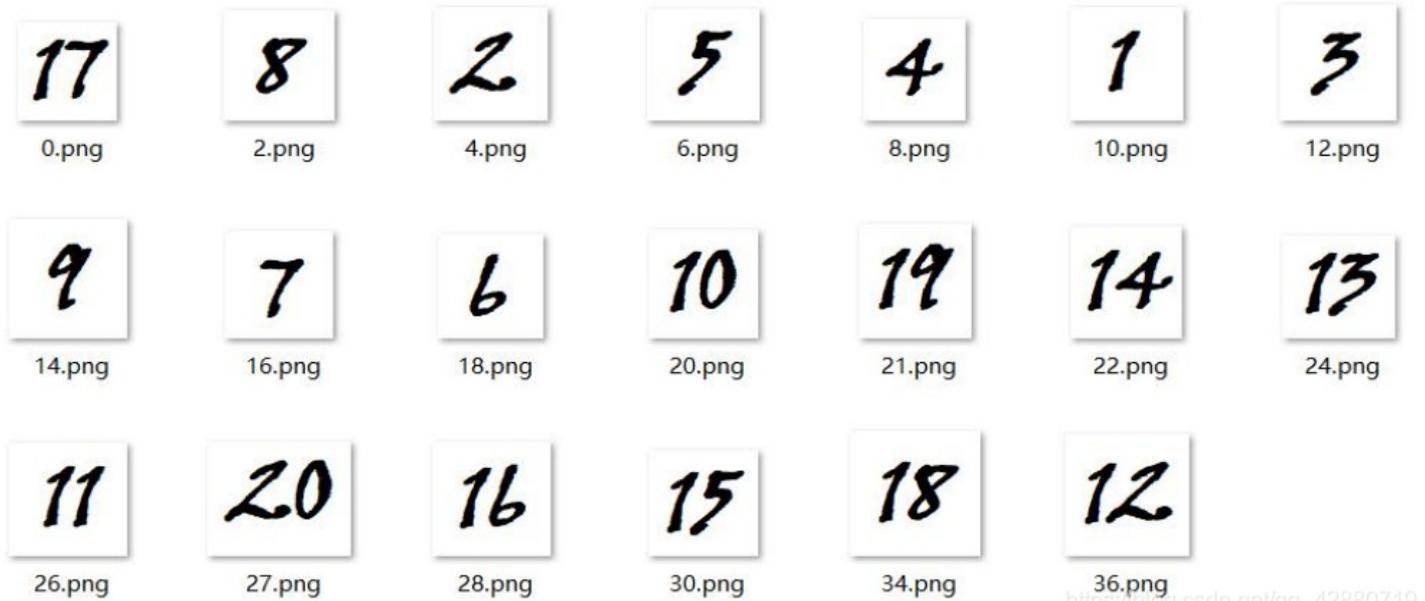
Base64 编码或解码的结果:

```
ctfshow{I_LC[REDACTED]ME}
```

## F5也会LSB

这道题也很良心

压缩包显示6long，爆破得到密码114514（恶臭），解压得到



结合LSB，以2.png为例，发现LSB有2字节长的数字信息，再查看图片1（即10.png），发现为504B030414，为ZIP文件头，所有信息提取出来

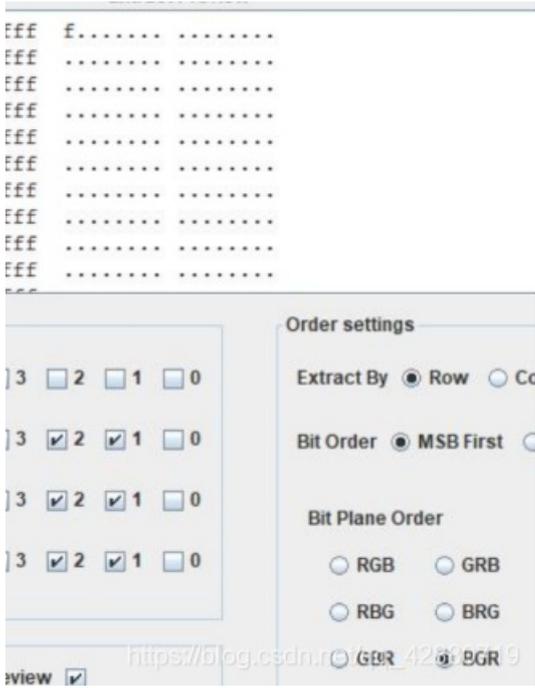
```
504B03041400010000007C4C4D524A0409E41D0000001100000008000000666C61672E747874268FDB736D3D74D7654C946555E  
F139A56337B2E90EE9D79DE3A0632B2504B01023F001400010000007C4C4D524A0409E41D000000110000000800240000000000  
000020000000000000000666C61672E7478740A002000000000001001800CBE6AA92A801D701F7A65B93A801D701A0FCEAE4A50  
1D701504B050600000000010001005A000000430000000000
```

得到zip包，发现有密码，再进行爆破，密码为7775，解压打开txt



BGR bit Planes 21

再去查看10.png的BGR 2 1通道



全部提取组合得到flag

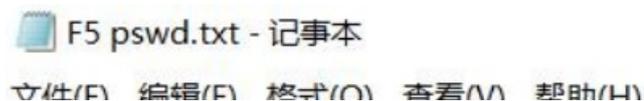
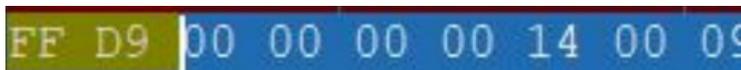
## F5还会学中文

也是一道凉心题

下载附件解压，得到 F5.jpg 和 flag.zip，flag.zip 需要密码，将 F5.jpg 放入 010 查看JPG结束后的"文件头"和文件尾



发现有PK字迹，所以需要修复zip包即00 00 00 00->50 4B 03 04，并且将09改为00



文件(F) 编辑(E) 格式(O) 语言(V) 帮助(H)

ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfsh  
ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF  
ctfshowF ctfshow5 ctfshow杯 ctfshow5 ctfshow5 ctfshowF ctfsh  
ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF  
ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfsh  
ctfshow杯 ctfshow5 ctfshowF ctfshow杯 ctfshowF  
ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfsh  
ctfshowF ctfshow5 ctfshowF ctfshow杯 ctfshowF  
ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfshowF ctfsh

很明显的Ook!

ctfshow—>Ook F—>. 5—>! 杯—>?

得到F5's password is f5alsogood

解压flag.zip发现错误，结合F5 pswd和jpg，可以猜测F5隐写。

```
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=
true
Huffman decoding starts
java.lang.ArrayIndexOutOfBoundsException: Index 64 out of bounds for leng
th 64
    at ortega.HuffmanDecode.Decode_ZZ(ortega/HuffmanDecode.java:306)
    at ortega.HuffmanDecode.Decode_AC_coefficients(ortega/HuffmanDeco
de.java:295)
```

F5隐写发现报错，原因是图片末尾有zip痕迹，将zip痕迹删掉只保留jpg，再提取，得到EveryOneL0veBeF5

## zip-password-is-EveryOneL0veBeF5

解压得到最难的部分，因为尝试GB2312编码，BIG5编码，GBK编码，GB18030编码，Unicode编码都是无解的

中国文化博大精深  
来瞧瞧下面的字吧  
毕竟F5也会学中文

插瑾舍聿悴丿忒法彘豕跌喷烤创

根据群里的随缘hint，和题目hint1：最后一步是转ascii

hint:GB2312-80，前两位为x码，后两位为x码

百度就能做

随缘hint@

GB2312-80, 前两位为x码, 后两位为x码  
直接搜“汉字GB2312-80”

## GB2312-80 《信息处理交换用汉字编码字符集 基本集》

GB2312将代码表分为94个区, 对应第一字节; 每个区94个  
为区号值和位号值加32 (20H), 因此也称为**区位码**。01-09

百度搜一个区位码批量查询

<http://www.jsj.net/index/gb2312.php>

输入汉字:

插瑾舍聿悴ワ怱法豕豕跌喷围创

区位码显示:

0003 7210 4165 7718 6718 0579 4343 2308 1371 8525  
2188 3771 1293 3757

将空格去掉 (注意最后一个数字后面的空格也要去掉)  
得到的数字转16进制, 再转字符, 得到flag

```
1 ctfshow{Ez_
```

## GoodNight

Hint1:题目的附件名字很重要  
Hint2:flag的内容要转为md5

又是套娃呜呜呜难死了

首先GoodNight很重要, 估计是某种加密的密码, 最后发现还是OurSecret

GoodNight.png Size: 137841 bytes

Step 2: Enter password

.....

 Unhide (double click to save)

Type	Name	Size (k)
File	Secert1	45

提取出来Secert1.rt1, 查看文件头, 发现BP, 考点是BPG, 可以查到bpg文件头是425047FB, B190是文件尾, 4250后面添加上的47FB, 然后搜索B190, 这里在第2个B190处截断, 因为后面有6050B405, 是反过来的504B0304

```
B1 90 00 00 00 00 01 70 00 00 00 FD 00 20 00 20  
00 00 00 00 60 50 B4 05 7A 09 5E 0B 88 5E D8 8B
```

然后改文件后缀, 用Simple BPG image viewer打开, 得到



解码得到fake flag...

```
flag(Th1s_i5_F4ke_f1a9)
```

再查看文件尾，将刚刚发现的zip提取出来

```
80 00 90 00 41 40 30 B4 05 F2 7
00 50 00 00 00 00 00 00 00 00 0
99 D3 00 00 00 00
```

504B0304，所以需要将十六进制数进行逆转

<http://tool.huixiang360.com/str/reverse.php>

转完后再放入winhex

00 00 00 00 00 3D 99 44 52 00 00 00 00 00 00 00	=!DR
00 00 00 00 00 05 00 00 00 66 6C 61 67 2F 50 4B	flag/PK
03 04 14 00 09 00 08 00 21 97 44 52 D6 36 06 29	!DRÖ6 )
90 0F 00 00 ED 56 00 00 13 00 23 00 66 6C 61 67	iV # flag
2F CE FB CE FB A3 AC CF EB B2 BB B5 BD B0 C9 75	/iüiü£-ië²»µ½°Eu
70 1F 00 01 43 56 31 EF 66 6C 61 67 2F E5 98 BB	p CVliflag/ã!»

对比正常的zip:

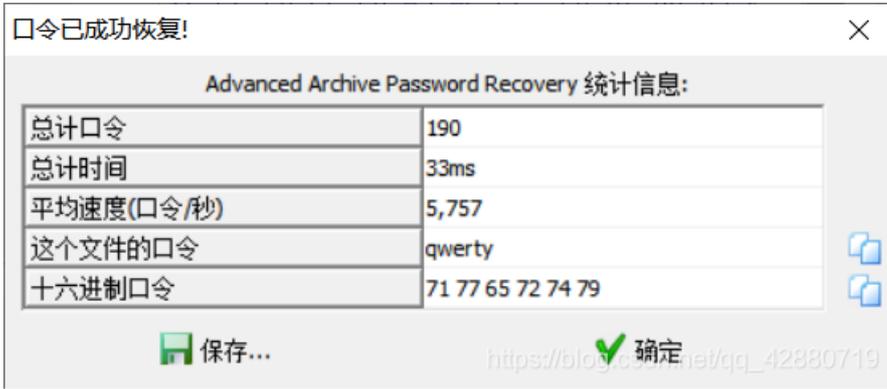
50 4B 03 04 14 00 00 00 08 00 4F 8B 57 52 04 BD	PK	O!WR ½
---	----	--------

可以发现就少了前面的

504B03041400，补上并保存

为加密的文件输入密码:  
flag\嘻嘻,想不到吧

需要密码(可恶), 这里我用的rockyou进行爆破



得到密码qwerty，解压得到



winhex查看

U	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	ANSI	ASCII
4	34	20	43	33	20	42	32	20	41	31	20	30	32	20	30	D4 C3 B2 A1 02 0	
30	20	30	34	20	30	30	20	30	30	20	30	30	20	30	30	0 04 00 00 00 00	
20	30	30	20	30	30	20	30	30	20	30	30	20	30	30	0D	00 00 00 00 00	
0A	46	46	20	46	46	20	30	30	20	30	30	20	30	31	20	FF FF 00 00 01	
30	30	20	30	30	20	30	30	20	44	30	20	46	45	20	34	00 00 00 D0 FE 4	
43	20	35	46	20	32	44	20	38	32	20	30	35	20	30	30	C 5F 2D 82 05 00	

需要将右边的数字导入winhex

然后放进kali能够识别出是个流量包，所以补上后缀pcapng  
打开后发现是一段英语对话文本

no.	time	source	destination	protocol	length	leftover	capture data	info
1	0.000000	VMware_a0:9e:01	Broadcast	ARP	60			Who has 192.168.80.1? Tell 192.168.80.132
2	0.000519	VMware_c0:00:08	VMware_a0:9e:01	ARP	42			192.168.80.1 is at 00:50:56:c0:00:08
3	0.000582	10.246.153.208	8.8.8.8	TCP	77			1234 → 81 [PSH] Seq=1 Win=53270 Len=23
4	0.000600	10.246.153.184	8.8.8.8	TCP	94			1234 → 81 [PSH] Seq=1 Win=53270 Len=40
5	0.000613	10.246.153.201	8.8.8.8	TCP	80			1234 → 81 [PSH] Seq=1 Win=53270 Len=26
6	0.000626	10.246.153.195	8.8.8.8	TCP	80			1234 → 81 [PSH] Seq=1 Win=53270 Len=26
7	0.000639	10.246.153.222	8.8.8.8	TCP	102			1234 → 81 [PSH] Seq=1 Win=53270 Len=48
8	0.000679	10.246.153.13	8.8.8.8	TCP	76			1234 → 81 [PSH] Seq=1 Win=53270 Len=22
9	0.000699	10.246.153.22	8.8.8.8	TCP	79			1234 → 81 [PSH] Seq=1 Win=53270 Len=25
10	0.000701	10.246.153.150	8.8.8.8	TCP	62			1234 → 81 [PSH] Seq=1 Win=53270 Len=8
11	0.000704	10.246.153.55	8.8.8.8	TCP	75			1234 → 81 [PSH] Seq=1 Win=53270 Len=21
12	0.000706	10.246.153.243	8.8.8.8	TCP	109			1234 → 81 [PSH] Seq=1 Win=53270 Len=55

```
> Frame 5: 80 bytes on wire (640 bits), 80 bytes captured (640 bits)
> Ethernet II, Src: VMware_a0:9e:01 (00:0c:29:a0:9e:01), Dst: VMware_c0:00:08 (00:50:56:c0:00:08)
> Internet Protocol Version 4, Src: 10.246.153.201, Dst: 8.8.8.8
> Transmission Control Protocol, Src Port: 1234, Dst Port: 81, Seq: 1, Len: 26
> Data (26 bytes)
```

```
0000 00 50 56 c0 00 08 00 0c 29 a0 9e 01 08 00 45 00  -PV....)....E-
0010 00 42 00 48 00 00 ff 06 06 9f 0a f6 99 c9 08 08  -B-H.....
0020 08 08 04 d2 00 51 00 00 00 00 00 00 00 50 08  -...Q...-...P-
0030 d0 16 14 e0 00 00 41 3a 48 6f 77 20 6f 6c 64 20  -...A: How old
0040 69 73 20 79 6f 75 72 20 64 61 75 67 68 74 65 72  is your daughter
```

[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

首先想是把长度提取出来转ascii，但是发现有不可打印字符长度，就放弃了这个思路。这道题的考点是TCP隐写

3	0.000582	10.246.153.208	8.8.8.8	TCP	77
4	0.000600	10.246.153.184	8.8.8.8	TCP	94
5	0.000613	10.246.153.201	8.8.8.8	TCP	80
6	0.000626	10.246.153.195	8.8.8.8	TCP	80
7	0.000639	10.246.153.222	8.8.8.8	TCP	102
8	0.000679	10.246.153.13	8.8.8.8	TCP	76

9	0.000699	10.246.153.22	8.8.8.8	TCP	79
10	0.000701	10.246.153.150	8.8.8.8	TCP	62
11	0.000704	10.246.153.55	8.8.8.8	TCP	75
12	0.000706	10.246.153.243	8.8.8.8	TCP	109

```

> Frame 3: 77 bytes on wire (616 bits), 77 bytes captured (616 bits)
> Ethernet II, Src: VMware_a0:9e:01 (00:0c:29:a0:9e:01), Dst: VMware_c0:00:08 (00:50:56:c0:00:08)
< Internet Protocol Version 4, Src: 10.246.153.208, Dst: 8.8.8.8
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 63
    Identification: 0x0040 (64)
  > Flags: 0x00
    Fragment Offset: 0
    Time to Live: 255
    Protocol: TCP (6)
    Header Checksum: 0x06a3 [validation disabled]
  
```

0000	00 50 56 c0 00 08 00 0c 29 a0 9e 01 08 00 45 00	-PV.....)	.....E.
0010	00 3f 00 40 00 00 ff 06 06 a3 0a f6 99 d0 08 08	.....@.....	.....
0020	08 08 04 d2 00 51 00 00 00 00 00 00 00 50 08	.....Q.....	.....P.
0030	d0 16 14 e0 00 00 41 3a 57 68 61 74 20 63 61 6e	.....A: What can	

将所有的字

符给提取出来，注意有重复字符，因舍弃。

No.	Time	Source	Destination	Prot
38	0.000802	10.246.153.149	8.8.8.8	TCP
39	0.000804	10.246.153.231	8.8.8.8	TCP
40	0.000806	10.246.153.88	8.8.8.8	TCP
41	0.000808	10.246.153.23	8.8.8.8	TCP
42	0.000856	10.246.153.64	8.8.8.8	TCP
43	0.000911	10.246.153.5	8.8.8.8	TCP
44	0.000959	10.246.153.128	8.8.8.8	TCP
45	0.001082	10.246.153.42	8.8.8.8	TCP
46	0.001129	10.246.153.172	8.8.8.8	TCP
47	0.001175	10.246.153.208	8.8.8.8	TCP
48	0.001229	10.246.153.103	8.8.8.8	TCP
49	0.001273	10.246.153.78	8.8.8.8	TCP

```

> Frame 49: 96 bytes on wire (768 bits), 96 bytes captured (768 bits)
> Ethernet II, Src: VMware_a0:9e:01 (00:0c:29:a0:9e:01), Dst: VMware_c0:00:08 (00:50:56:c0:00:08)
< Internet Protocol Version 4, Src: 10.246.153.78, Dst: 8.8.8.8
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 82
    Identification: 0x0050 (80)
  > Flags: 0x00
    Fragment Offset: 0
    Time to Live: 255
    Protocol: TCP (6)
    Header Checksum: 0x0702 [validation disabled]
  
```

0000	00 50 56 c0 00 08 00 0c 29 a0 9e 01 08 00 45 00	-PV.....)	.....E.
0010	00 52 00 50 00 00 ff 06 07 02 0a f6 99 4e 08 08	.....R.....	.....N.

然后进行base转码，只可能是base64以上的，最后发现base91成功转码

<https://ctf.bugku.com/tool/base91>

---

@iH<{\*;oUp/im"QPI`yR\*ie}NK;.D!Xu)b:J[Rj+6KKM7P

加密

解密

---

flag{wM...3A}|  
[https://blog.csdn.net/qq\\_42880719](https://blog.csdn.net/qq_42880719)

题目说flag的内容需要md5一下，将里面md5后提交即可

本文完