第一届AFCTF杂项出题

contact me

- 秦川
- QQ：*1040359429*
- Blog：https://blog.csdn.net/funkypants
- Github：https://github.com/FunkyPantss

1.黑客与画家

题目描述

不要被这本书的书名迷惑，这本书也可以叫做《程序员与画家与创业》，强烈推荐！你问我这堆数字是什么？画出来就知道了！

flag

afctf{Hackers and Pa!nters!}

出题思路

使用以下Python脚本提取图片像素值，并写入到一个压缩文件中

```
from PIL import Image
import zipfile
f_zip = zipfile.ZipFile('filename.zip', 'w', zipfile.ZIP_DEFLATED)

im = Image.open('flag.png')
pix = im.load()
width = im.size[0]
height = im.size[1]
f = open('result333.txt', 'w')
for x in range(width):
        for y in range(height):
                r, g, b = pix[x, y]
                f.write(str(r) + ' ')
                f.write(str(g) + ' ')
                f.write(str(b) + ' ')
                f.write('\n')

f_zip.write('result333.txt')
f_zip.close()
```

解题思路

使用以下Python脚本将像素值还原为图片

```
from PIL import Image

x = 887
y = 111

im = Image.new("RGB",(x,y))
file = open('result333.txt', 'r')
line = file.readlines()

for i in range(0, x):
        for j in range(0, y):
                l = line[i * y + j]
                # l = l.replace('(', '').replace(')', '').replace(' ', '')
                lst = l.split(" ")    # 分割成三个通道，r,g,b分别为三个图像对象
                im.putpixel([i, j], (int(lst[0]), int(lst[1]), int(lst[2])))    # 将对应的rgb写入相应的位置
im.show()
im.save("c.png")
```

选手writeup点评

题目中给出的resuslt333.txt只是我忘了改文件名，结果有很多人通过这个判读出了图片高度是333，而原先设定的y刚好是111……一种可行的做法是：将总行数98457分解为887x37x3，将37x3写为111，刚好得到原图片的长和高。

2.git_leak

题目描述

github.com，全球最大同性交友网站，快去学一下怎么用吧！

flag

afctf{git_is_useful}

出题思路

将flag在某次提交后，在下个版本中删除flag，可通过版本回退找到flag。最终版本的文件夹中包含的图片都是用作干扰。

```
秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git add flag.txt

秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git commit -m 'flag'
[master f8c105c] flag
 1 file changed, 1 insertion(+)
 create mode 100644 flag.txt

秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git log
commit f8c105cc910e22c51ec49c07959b8edb62c6126e (HEAD -> master)
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:13:08 2018 +0800

    flag

commit 6c59527b0520faf01bbded2500a2fa21f2e7cab2
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:11:34 2018 +0800

    first commit

秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git add .

秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git diff

秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git commit -m 'can you find the flag in this directory?'
[master 4efb1a9] can you find the flag in this directory?
 1 file changed, 1 deletion(-)
 delete mode 100644 flag.txt

秦川@Rye MINGW64 /f/AFCTF/git_leak (master)
$ git log
commit 4efb1a9fc47cf85fc22cea7e8bb0b1a85085a399 (HEAD -> master)
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:14:37 2018 +0800

    can you find the flag in this directory?

commit f8c105cc910e22c51ec49c07959b8edb62c6126e
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:13:08 2018 +0800

    flag

commit 6c59527b0520faf01bbded2500a2fa21f2e7cab2
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:11:34 2018 +0800

    first commit
```

解题思路

使用git reset –hard 版本号回退到flag所在版本

```
秦川@Rye MINGW64 ~/Desktop/git_leak (master)
$ git log
commit 4efb1a9fc47cf85fc22cea7e8bb0b1a85085a399 (HEAD -> master)
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:14:37 2018 +0800

    can you find the flag in this directory?

commit f8c105cc910e22c51ec49c07959b8edb62c6126e
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:13:08 2018 +0800

    flag

commit 6c59527b0520faf01bbded2500a2fa21f2e7cab2
Author: FunkyPantss <1040359429@qq.com>
Date:   Sun Apr 8 16:11:34 2018 +0800

    first commit

秦川@Rye MINGW64 ~/Desktop/git_leak (master)
$ git reset --hard f8c105
HEAD is now at f8c105c flag

秦川@Rye MINGW64 ~/Desktop/git_leak (master)
$
```

3.中英文比较文学

题目描述

在计算机的世界里，中文和英文的有很多种不同的表示方法，直到.....

flag

afctf{welcome_to_join_us}

出题思路

利用中英文的utf-8字符编码长度不同的特点，可将一段中英文混合的字符串生成不同长度的二进制数据，达到迷惑效果。

需要特别说明的是，对于中文来说，不用将其补全为8位或16位，直接

使用以下代码生成flag文件

```
import zipfile

text = '中南大学极光网络安全实验室成立于2016年，是一个以技术为导向的社团，旨在为中南大学校内信息安全爱好者提供一个相互交流的平台。现有实验室成员60余人,目前设有网

with open('bi.txt', 'w') as f:
        for i in text:
                print(bin(ord(i)).replace('0b', ''))
                f.write(bin(ord(i)).replace('0b', ''))
                f.write(' ')

#make a zip file
f_zip = zipfile.ZipFile('bi.zip', 'w', zipfile.ZIP_DEFLATED)
f_zip.write('bi.txt')
f_zip.close()
```

解题思路

将二进制数据转化为十进制，再打印其ASCII码。 解题代码

```
with open('bi.txt', 'r') as f:
        for i in f.read().split(' '):
                print(chr(int(i,2)), end='')
```

4.Word隐写1

题目描述

我昨天去看了头号玩家，记得主角是怎么通过第一关的吗？不要将思路局限在常规情况下，很多时候解杂项题都需要脑洞。

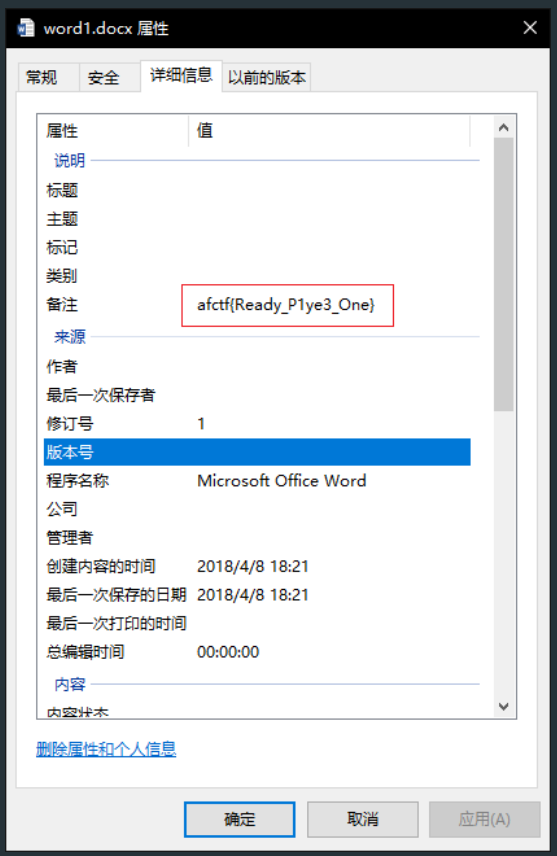
flag

afctf{Ready_P1ye3_One}

出题思路

将flag放到文件-详细信息中。

解题思路



5.Word隐写2

题目描述

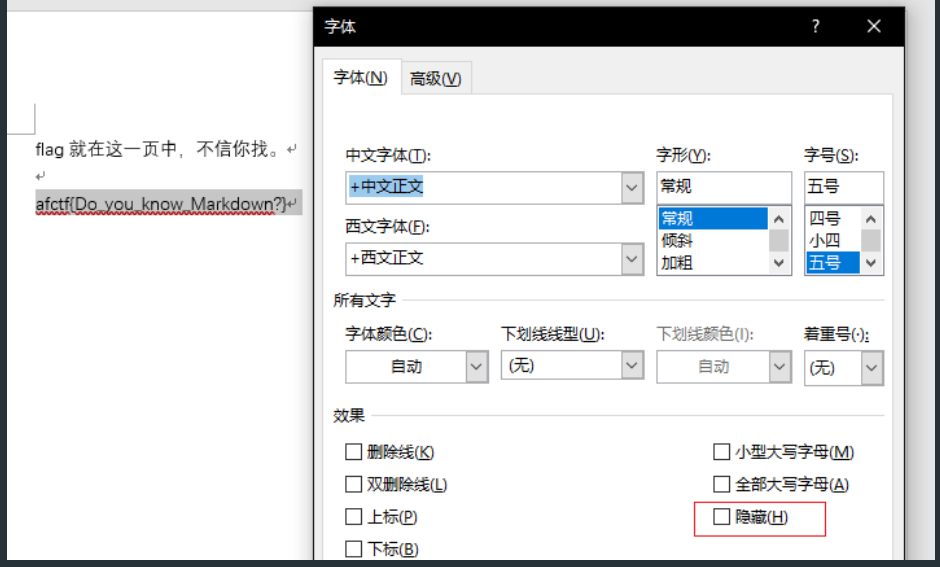
这是一份用神奇墨水写的实验报告吗？

flag

afctf{Do_you_know_Markdown?}

出题思路

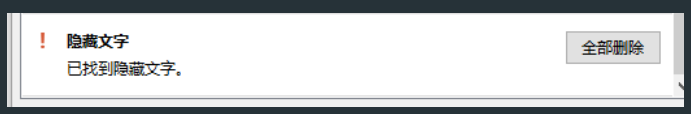
利用Word中的隐藏功能将内容隐藏起来，若未用过此功能，很难找到窍门。



解题思路

方法1

使用Word中的检查文档功能查找被隐藏文字。



全选文档，取消勾选“隐藏”可见flag。

方法2

使用Binwalk工具将Word文档拆分为xml文档，在其中寻找文本内容，这里不再演示。

6.流量分析

题目描述

日常浏览网页。

flag

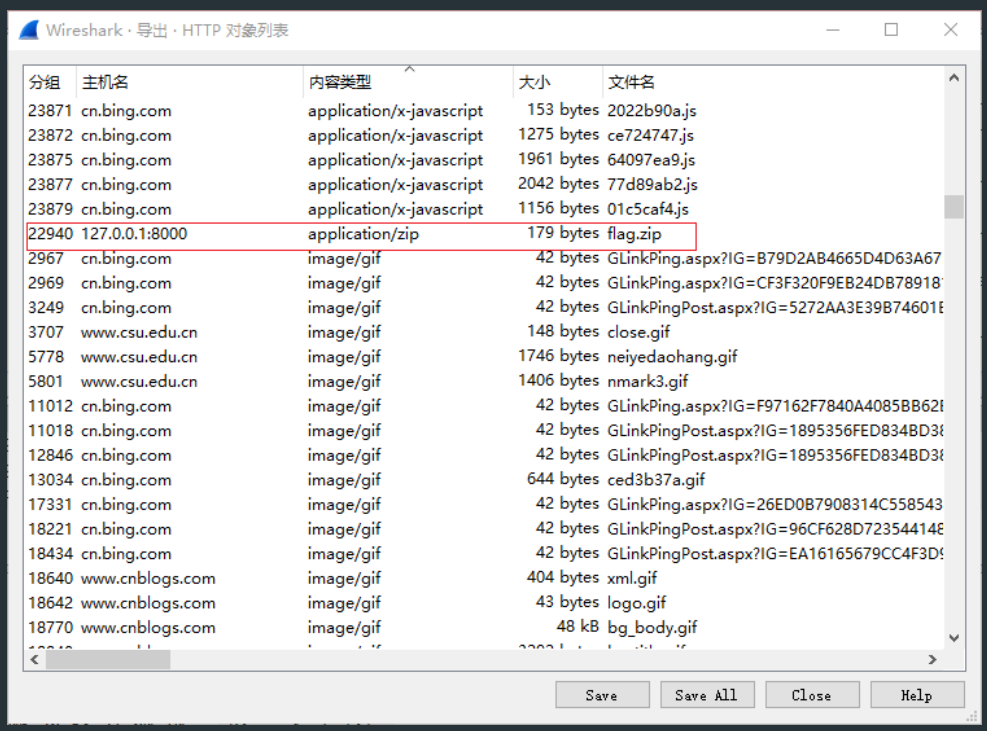
afctf{get_f1ag_from_a_fi1e}

出题思路

在本地开了个文件服务器，从上面可以下载一个名为flag.zip的文件，里面藏有flag。

解题思路

使用WireShark，直接导出HTTP对象可得flag文件。



7.数据恢复1

题目描述

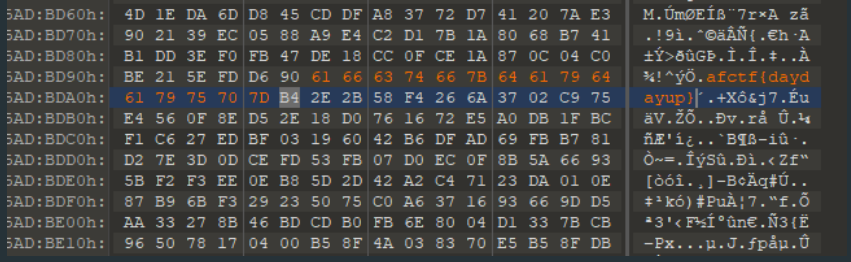
一个损坏的VMware虚拟机镜像，你可能需要修复它才能得到flag?

flag

afctf{daydayup}

出题思路

一开始本来想在ISO文件中创建一个flag.txt文件，但是有点麻烦，所以就直接改了一个Ubuntu镜像的信息。



解题思路

在十六进制编辑器中搜索text：afctf。

解题思路

8.文件解密

题目描述

这是一个被加密了的压缩包，密码格式为“Afctf{xxxxxxxx}”，其中“xxxxxxxx”为0到10000000之间的数字，动手吧！

flag

afctf{Ten_mil1ion}

出题思路

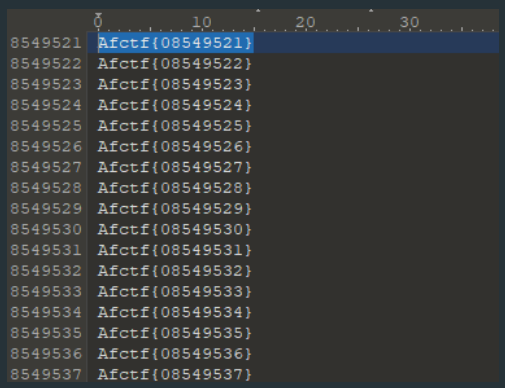
给出一个加密压缩包，定义一定的密码格式使其不能被简单的爆破工具爆破，需要自己编写小程序生成字典再爆破，或者使用掩码攻击。给定的密码为“Afctf{08549521}”。

解题思路

使用Python脚本生成密码字典，再使用例如“Ziperello.exe”这样的爆破工具加载字典进行爆破。

```
with open('dict.txt', 'w') as f:
        for i in range(1,10000000):
                diff = 8 - len(str(i))
                s = '0' * diff + str(i)
                f.write('Afctf{' + s + '}'+ '\n')
        f.write('Afctf{' + str(10000000) + '}')
```

在生成的字典中包含我们需要的密码。



使用工具加载字典并进行爆破。

9.Statistics

题目描述

在计算机取证课堂上，我们学到了一个内部数据泄露案例，有嫌疑人使用将数据伪装成手机中的普通照片，从而使数据成功出去。下面是这张图片，你能从中发现什么吗？在计算机取证的过程中，有时候频次也是一个关键的点。

注意：提交时请自行把flag格式补全为“afctf{}”样式

flag

atctf{MyheArTWil1GoOn!}

出题思路

仿照新鲜出炉的ddctfMisc第三题，先将flag字符串按照从高频到低频的顺序，生成一个杂乱无章的字符串，将字符串写入txt中，再压缩，再与图片拼接。

使用以下代码生成flag.txt

```
#flag = 'afctf{ChinANo1!#}'
from random import choices
flag = 'atctf{MyheArTWil1GoOn!}'

num = {'a': 1000,
       'f': 950,
       'c': 900,
       't': 850,
       '{': 800,
       'M': 750,
       'y': 700,
       'h': 650,
       'e': 600,
       'A': 500,
       'r': 450,
       'T': 400,
       'w': 350,
       'i': 300,
       'l': 250,
       '1': 200,
       'G': 150,
       'o': 100,
       'O': 50,
       'n': 30,
       '!': 20
       }

list_num = list(num.keys())


with open('flag.txt', 'w') as f:
    while True:
        if list_num:
            key = str(choices(list_num)[0])#dict==>key
            #print(key)

            #a item eq 0, pop the key form list
```

```
                    if num[key] == 0:
                        index = list_num.index(key)
                        list_num.pop(index)
                        #continue
                    #get value and write to file, then value -1
                    elif num[key] > 0:
                        #letter = str(num[key])
                        f.write(key)
                        num[key] -= 1
            else:
                    print('jieshu')
                    exit(0)
```

解题思路

在得到flag.txt后，使用以下代码统计各个字符出现频次，按出现次数从低频到高频拼接得到flag。

```
str = '待统计字符串'
str_list = list(str)
char_dict = {}

for char1 in str:
    if char1 in char_dict:
        count = char_dict[char1]
    else:
        count = 0
    count = count + 1
    char_dict[char1] = count
print(char_dict)
```

10.文件系统

题目描述

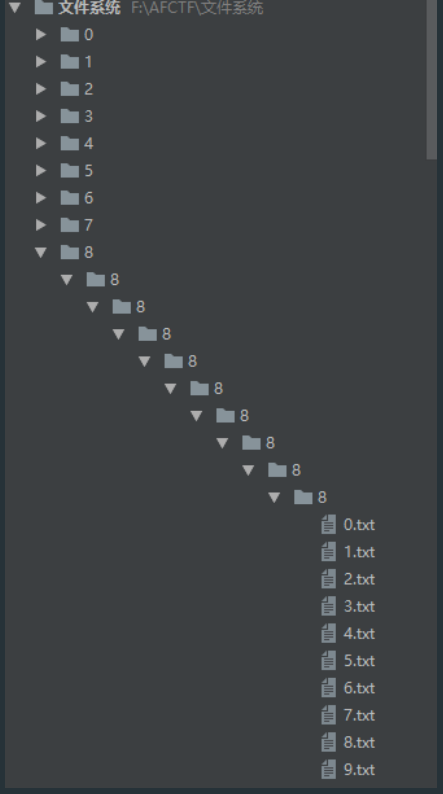
听说你很喜欢手动找flag？这就是为你准备的，从现在开始学习一个程序猿如何编写自动化脚本去解决实际问题。

flag

afctf{Traver31ng_the_d1rect0ry}

出题思路

生成如下格式的文件目录，每个目录下有10层子目录，第10层子目录中有10个txt文件，共计10000个txt文件，其中只有84/.../84/6.txt文件中写有32位flag值，其他txt中均为32位乱码。原本想打包成可挂载的文件系统，但是，没学会。



自动生成目录代码

```
import os
import random

rootdir = os.getcwd()
times = 10
flag = 'afctf{Traver31ng_the_d1rect0ry}'
charset = '1234567890abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!@#$%^&*()_+=-'
print(rootdir)


def random_char():
        sa = []
        for i in range(31):
                sa.append(random.choice(charset))
        salt = ''.join(sa)
        return salt


for i in range(0, 100):
        dir = ''
        for j in range(0, times):
                dir = dir + str(i) + '/'
                print(rootdir + '/' + dir)
                if os.path.isdir(rootdir + '\\' + dir):
                        continue
                else:
                        os.mkdir(dir)
                        # os.chdir(dir)
                        # filename = str(i) + '.txt'
                        # with open(filename, 'w') as f:
                        #         f.write('123')
                        # os.chdir('..')
        os.chdir(dir)
        for m in range(0, times):
                filename = str(m) + '.txt'
                with open(filename, 'w') as f:
                        f.write(random_char())
        os.chdir(rootdir)


        # for parent, dirnames, filenames in os.walk(rootdir):
        #         print(parent)
        #         print(filenames)
```

解题思路

遍历主目录下的所有txt文件，在其中查找flag。

```
import os

rootdir = os.getcwd()

for parent, dirnames, filenames in os.walk(rootdir):
        # print(parent)
        # print(dirnames)
        # print(filenames)
```

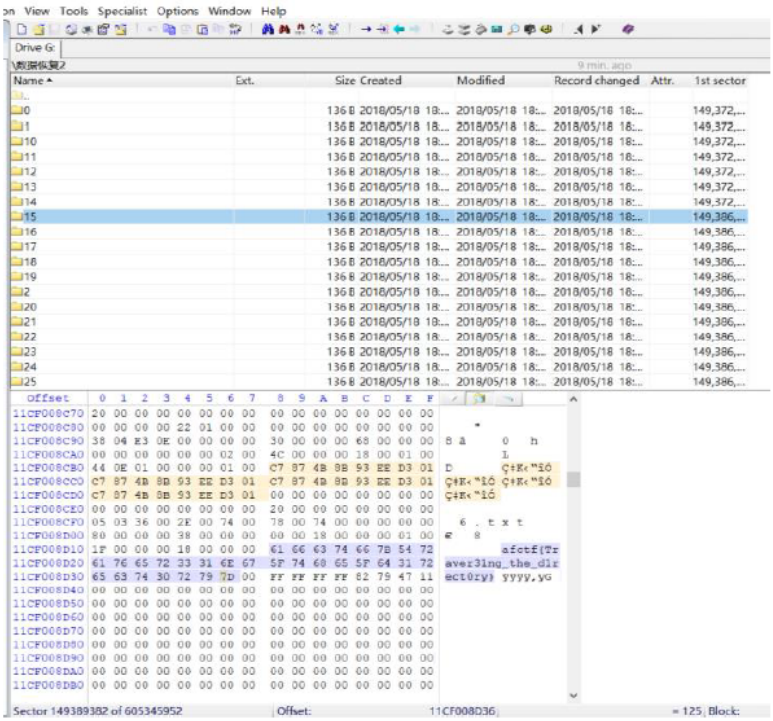
```
#  #  print('*****')
#  #if  os.path.isfile(parent):
#  #  if  filenames:
for  file  in  filenames:
        with  open(parent  +'\\'  +  file,  'r')  as  f:
                text  =  f.read()
                #print(text[0:5])
                if  text[0:5]  ==  'afctf':
                        print(text)
```

解法二

By——CSU-Cans

7. 数据恢复 2

打开题目，推测是大量文件夹中，藏有含有 flag 的 txt，于是不写脚本，用 winhex 暴力搜索得到 flag：



11.找零

题目描述

小明需要支付100元RMB，你能帮助他用两张不同面额的钞票凑足100元吗？ 提示：正确答案样式中不包含“afctf{}”等字样，flag格式为：afctf{data+checkDigit}，其中checkDigit为钞票校验位。

flag

afctf{649B}

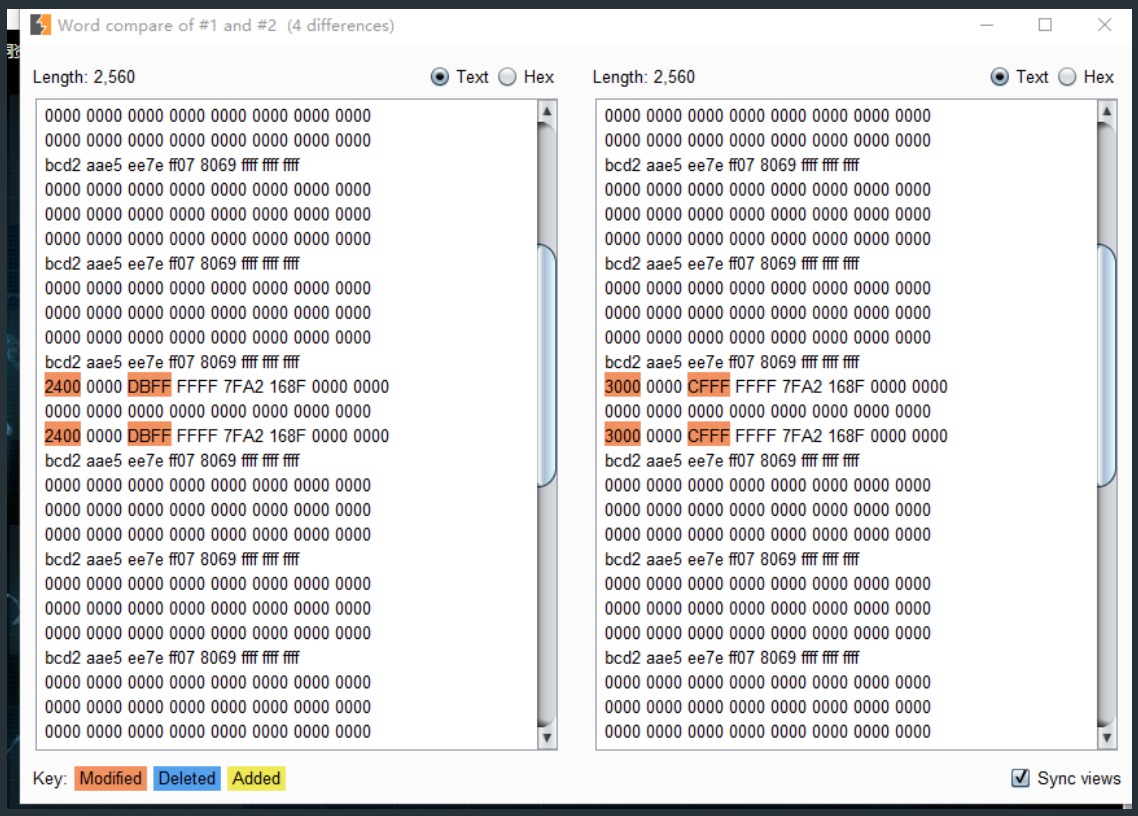
出题思路

此题来自于XDCTF2017，当时是第一次做CTF题目，于第5名提交了此题flag，一度排名全国第2，所以这个题目给我的印象很深，这次就直接搬上来了。

这个题目需要比较36和48两张钞票的不同，从此推断出100面额钞票的样式，并且能找出正确的校验位一同提交。

解题思路

使用文本比对工具找出不同



对比发现第25行数据不同

不同之处换算成十进制后正好相差12，这两位应该就是用来表示金额的，

还有一个不同的地方是DB和CF，

0x24+0xdb=0x30+0xcf=0xff

所以DB和CF估计是校验位,依此100RMB应该是

6400 0000 9BFF FFFF 7FA2 168F 0000 0000

其中，64为data，9B为checkDigit。

赛后点评

这题和文件格式没什么关系，应该说用十六进制编辑器打开文件后，看到两个文件前一部分是一样的，就应该联想到文本比对。

12.Python is best language

题目描述

.pyc文件是由.py文件经过编译后生成的字节码文件，其加载速度相对于之前的.py文件有所提高。说不定这个题目是杂项中的逆向，反汇编一下就出来了？

flag

afctf{songfent}

出题思路

送分题，ctf-wiki看到最后就知道怎么做了，奖励那些看完了的人。唯一的坑就是我乱写了一堆python语句，要是没学过python一时半会还看不懂，看懂了也没用。就是将信息嵌入到pyc字节码中，并不需要反汇编，反汇编也没用。

解题思路

参考ctf-wiki上面的解决方案。

http://www.freebuf.com/sectool/129357.html

```
python  -m  stegosaurus__pycache__/example.cpython-36-stegosaurus.pyc  -x
```

```
PS G:\安全工具\jherron-stegosaurus-cd5c2373c031> python -m stegosaurus __pycache__/example.cpython-36-stegosaurus.pyc -x
Extracted payload: afctf{songfent}
```

13.可爱猫咪

题目描述

这图片的比例似乎不太对?

flag

afctf{Brai11e}

解题过程

下载图片之后用winhex查看可以发现这是png格式，更改文件尾可以看见一张猫咪的照片，这里是一个图片高度缩小的方法，png文件格式中第二行第六列是高度位，改这一位即可，隐藏了图片的下半部分，这一步有点难想，然后就可以得到一张完整的照片，隐去的部分是盲文，查找对应的字母就可以得到flag啦~

14.我们的成员真是太帅了

题目描述

基于这种文件格式的一种常见隐写算法。 提示：隐秘消息长度值要设置得足够大,x>2500

flag

afctf{L_S_13_g00d}

解题过程

识别bmp格式，之后就是常规的LSB算法来提取内容，在网上找其实还挺多的。 这里给出matlab的提取算法：

```
Picture=imread( '需要提取的图片路径');
Picture=double(Picture);
[m,n]=size(Picture);
frr=fopen( '提取出来的秘密消息路径','a');
len=2528;%隐秘消息长度值要设置得足够大（这里我是设的正好的2528）
p=1;
for  f2=1:n
for  f1=1:m
if  bitand(Picture(f1,f2),1)==1
fwrite(frr,1,'ubit1');
result(p,1)=1;
else
fwrite(frr,0,'ubit1');
result(p,1)=0;
end
if  p==len
break;
end
p=p+1;
end
if  p==len
break;
end
end
fclose(frr);
```