H GAME Week 4 Write Up

CRYPTO

easy_rsa

看到题目，两个e，两个c，一个n，应该是rsa的共模攻击。

然后，嘿嘿嘿，上次碰巧在V爷爷博客里，瞄到一眼，他还无私地给了脚本，那么，谢谢了！

代码如下：

```
from gmpy import invert
n =
0x00b0bee5e3e9e5a7e8d00b493355c618fc8c7d7d03b82e409951c182f398dee3104580e7ba70d383ae5311475656e8
a964d380cb157f48c951adfa65db0b122ca40e42fa709189b719a4f0d746e2f6069baf11cebd650f14b93c977352fd13
b1eea6d6e1da775502abff89d3a8b3615fd0db49b88a976bc20568489284e181f6f11e270891c8ef80017bad238e3630
39a458470f1749101bc29949d3a4f4038d463938851579c7525a69984f15b5667f34209b70eb261136947fa123e549df
ff00601883afd936fe411e006e4e93d1a00b0fea541bbfc8c5186cb6220503a94b2413110d640c77ea54ba3220fc8f4c
c6ce77151e29b3e06578c478bd1bebe04589ef9a197f6f806db8b3ecd826cad24f5324ccdec6e8fead2c2150068602c8
dcdc59402ccac9424b790048ccdd9327068095efa010b7f196c74ba8c37b128f9e1411751633f78b7b9e56f71f77a1b4
daad3fc54b5e7ef935d9a72fb176759765522b4bbc02e314d5c06b64d5054b7b096c601236e6ccf45b5e611c805d335d
bab0c35d226cc208d8ce4736ba39a0354426fae006c7fe52d5267dcfb9c3884f51fddfdf4a9794bcfe0e1557113749e6
c8ef421dba263aff68739ce00ed80fd0022ef92d3488f76deb62bdef7bea6026f22a1d25aa2a92d124414a8021fe0c17
4b9803e6bb5fad75e186a946a17280770f1243f4387446ccceb2222a965cc30b3929
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)

c1 = int(open('flag.enc1', 'rb').read().encode('hex'),16)
c2 = int(open('flag.enc2', 'rb').read().encode('hex'),16)
e1 = 17
e2 = 65537
s = egcd(e1,e2)
s1 = s[1]
s2 = s[2]
if s1<0:
    s1 = - s1
    c1 = invert(c1, n)
elif s2<0:
    s2 = - s2
    c2 = invert(c2, n)
m = pow(c1, s1, n) * pow(c2, s2, n) % n
print hex(m)[2:].decode('hex')
```

这是V爷爷的代码，这里是直接打印出字符串的，而这题我们要的是m的值，所以最后一句就改为： **print m**

（嗷，当然要把这道题目的数值带进去）得到结果：
m=2116552625739668810628237952201796446074121623710 69

嗯?这么长，不是说好m是17位的嘛。突然想起来，V爷爷说过，共模攻击的e要互质，于是去判断一下，先把e转十进制，然后用C语言的代码：

```
#include<stdio.h>
int gongyue(int m,int n)
{
int r;
if(m==n) return m;
else
while((r=m%n)!=0)
{
m=n;
n=r;
}
return n;
}
void main()
{
int a,b,i;
printf("please input two number:\n");
a=15951;
b=209472;
i=gongyue(a,b);
printf("最大公约数是:%d\n",gongyue(a,b));
}
```

得到结果：最大公约数是:3

果然，那么就将两个e都除以三，然后将结果开三次方，得到结果

(mpz(59594981651654789L), True)

数一下，正好17位，那么提交flag：hgame{59594981651654789}

最终脚本为：

```
from gmpy2 import invert
from gmpy2 import iroot
n =
0x9439682bf1b4ab48c43c524778c579cc844b60872275725c1dc893b5bcb358b9f136e4dab2a06318bb0c80e202a14b
c54ea334519bec023934e01e9378abf329893f3870979e9f2f2be8fff4df931216a77007a2509f49f697bf286285e97f
ac5dc6e4a164b5c2cc430887b18136437ba67777bda05aafdeaf918221c812b4c7d1665238f84ab0fab7a77fcae92a05
96e58343be7a8e6e75a5017c63a67eb11964970659cd6110e9ec6502288e9e443d86229ef2364dfecb63e2d90993a753
56854eb874797340eece1b19974e86bee07019610467d44ec595e04af02b574a97fa98bdb2e779871c804219cab715f4
a80fef7f8fb52251d86077560b39c1c2a1
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)

c1 =
0x7c7f315a3ebbe305c1ad8bd2f73b1bb8e300912b6b8ba1b331ac2419d3da5a9a605fd62915c11f8921c450525d2efd
a7d48f1e503041498f4f0676760b43c770ff2968bd942c7ef95e401dd7facbd4e5404a0ed3ad96ae505f87c4e12439a2
da636f047d84b1256c0e363f63373732cbaf24bda22d931d001dcca124f5a19f9e28608ebd90161e728b782eb67deeba
4cc81b6df4e7ee29a156f51a0e5148618c6e81c31a91036c982debd1897e6f3c1e5e248789c933a4bf30d0721a18ab87
08d827858b77c1a020764550a7fe2ebd48b6848d9c4d211fd853b7a02a859fa0c72160675d832c94e0e43355363a2166
b3d41b8137100c18841e34ff52786867d
c2 =
0xf3a8b9b739196ba270c8896bd3806e9907fca2592d28385ef24afadc2a408b7942214dad5b9e14808ab988fb15fbd9
3e725edcc0509ab0dd1656557019ae93c38031d2a7c84895ee3da1150eda04cd2815ee3debaa7c2651b62639f785f6ca
bf83f93bf3cce7778ab369631ea6145438c3cd4d93d6f2759be3cc187651a33b3cc4c3b477604477143c32dfff62461f
dfd9f8aa879257489bbf977417ce0fbe89e3f2464475624aafef57dd9ea60339793c69b53ca71d745d626f45e6a7beb9
fcbd9d1a259433d36139345b7bb4f392e78f1b5be0d2c56ad50767ee851fac670946356b3c05d0605bf243b89c7e683c
c75030b71633632fb95c84075201352d6
e1 = 0x33240
e2 = 0x3e4f
s = egcd(e1,e2)
s1 = s[1]
s2 = s[2]
if s1<0:
    s1 = - s1
    c1 = invert(c1, n)
elif s2<0:
    s2 = - s2
    c2 = invert(c2, n)
mmm = pow(c1, s1, n) * pow(c2, s2, n) % n
m=iroot(mmm,3)
print m
```

（在写wp的时候发现，就算e不除以三，也就是直接把先前得出来的m开三次方，也是这个答案，奇怪。emmm得好好研究一下共模攻击的原理了。。。）

MISC

暗藏玄机

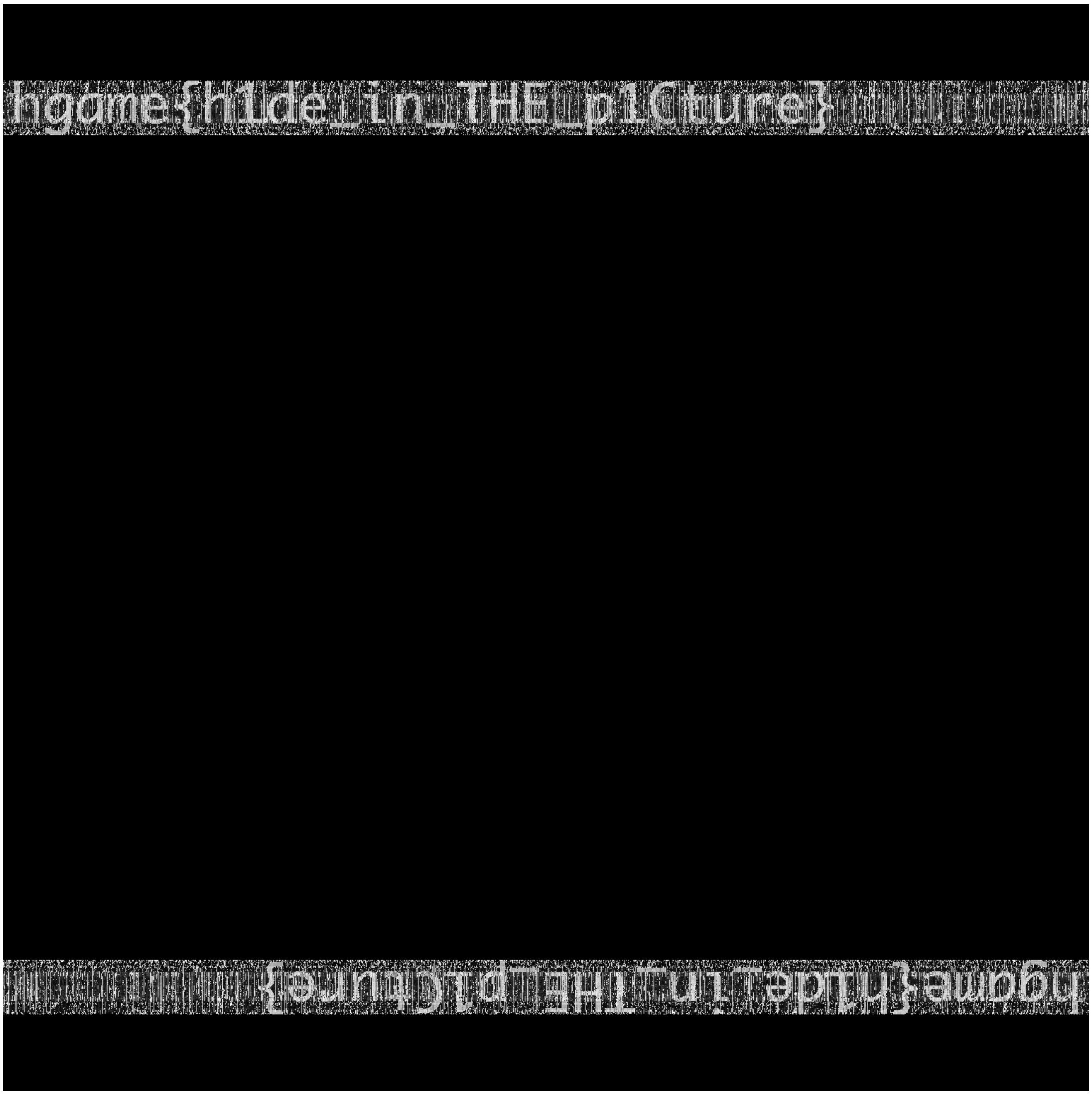
拿到题目，是个zip包，里面是两张一样的图片，所以无疑是盲水印攻击。网上找一个现成脚本bmp.py，

```
#!/usr/bin/env python
# -*- coding: utf8 -*-

import sys
import random

cmd = None
debug = False
seed = 20160930
alpha = 3.0

if __name__ == '__main__':
    if '-h' in sys.argv or '--help' in sys.argv or len(sys.argv) < 2:
        print 'Usage: python bwm.py <cmd> [arg...] [opts...]'
        print '  cmds:'
        print '    encode <image> <watermark> <image(encoded)>'
        print '           image + watermark -> image(encoded)'
        print '    decode <image> <image(encoded)> <watermark>'
        print '           image + image(encoded) -> watermark'
        print '  opts:'
        print '    --debug,          Show debug'
        print '    --seed <int>,     Manual setting random seed (default is 20160930)'
        print '    --alpha <float>,  Manual setting alpha (default is 3.0)'
        sys.exit(1)
    cmd = sys.argv[1]
    if cmd != 'encode' and cmd != 'decode':
        print 'Wrong cmd %s' % cmd
        sys.exit(1)
    if '--debug' in sys.argv:
        debug = True
        del sys.argv[sys.argv.index('--debug')]
    if '--seed' in sys.argv:
        p = sys.argv.index('--seed')
        if len(sys.argv) <= p+1:
            print 'Missing <int> for --seed'
            sys.exit(1)
        seed = int(sys.argv[p+1])
        del sys.argv[p+1]
        del sys.argv[p]
    if '--alpha' in sys.argv:
        p = sys.argv.index('--alpha')
        if len(sys.argv) <= p+1:
            print 'Missing <float> for --alpha'
            sys.exit(1)
        alpha = float(sys.argv[p+1])
        del sys.argv[p+1]
        del sys.argv[p]
    if len(sys.argv) < 5:
        print 'Missing arg...'
        sys.exit(1)
    fn1 = sys.argv[2]
    fn2 = sys.argv[3]
    fn3 = sys.argv[4]

import cv2
import numpy as np
import matplotlib.pyplot as plt

# OpenCV是以(BGR)的顺序存储图像数据的
# 而Matplotlib是以(RGB)的顺序显示图像的
def bgr_to_rgb(img):
    b, g, r = cv2.split(img)
    return cv2.merge([r, g, b])


if cmd == 'encode':
    print 'image<%s> + watermark<%s> -> image(encoded)<%s>' % (fn1, fn2, fn3)
    img = cv2.imread(fn1)
    wm = cv2.imread(fn2)

    if debug:
        plt.subplot(231), plt.imshow(bgr_to_rgb(img)), plt.title('image')
        plt.xticks([]), plt.yticks([])
        plt.subplot(234), plt.imshow(bgr_to_rgb(wm)), plt.title('watermark')
        plt.xticks([]), plt.yticks([])

    # print img.shape # 高, 宽, 通道
    h, w = img.shape[0], img.shape[1]
    hwm = np.zeros((int(h * 0.5), w, img.shape[2]))
    assert hwm.shape[0] > wm.shape[0]
    assert hwm.shape[1] > wm.shape[1]
    hwm2 = np.copy(hwm)
    for i in xrange(wm.shape[0]):
        for j in xrange(wm.shape[1]):
            hwm2[i][j] = wm[i][j]

    random.seed(seed)
    m, n = range(hwm.shape[0]), range(hwm.shape[1])
    random.shuffle(m)
    random.shuffle(n)
    for i in xrange(hwm.shape[0]):
        for j in xrange(hwm.shape[1]):
            hwm[i][j] = hwm2[m[i]][n[j]]

    rwm = np.zeros(img.shape)
    for i in xrange(hwm.shape[0]):
        for j in xrange(hwm.shape[1]):
            rwm[i][j] = hwm[i][j]
            rwm[rwm.shape[0] - i - 1][rwm.shape[1] - j - 1] = hwm[i][j]

    if debug:
        plt.subplot(235), plt.imshow(bgr_to_rgb(rwm)), \
            plt.title('encrypted(watermark)')
        plt.xticks([]), plt.yticks([])

    f1 = np.fft.fft2(img)
    f2 = f1 + alpha * rwm
    _img = np.fft.ifft2(f2)

    if debug:
        plt.subplot(232), plt.imshow(bgr_to_rgb(np.real(f1))), \
            plt.title('fft(image)')
        plt.xticks([]), plt.yticks([])

    img_wm = np.real(_img)

    assert cv2.imwrite(fn3, img_wm, [int(cv2.IMWRITE_JPEG_QUALITY), 100])


    # 这里计算下保存前后的(溢出)误差
    img_wm2 = cv2.imread(fn3)
    sum = 0
    for i in xrange(img_wm.shape[0]):
        for j in xrange(img_wm.shape[1]):
            for k in xrange(img_wm.shape[2]):
                sum += np.power(img_wm[i][j][k] - img_wm2[i][j][k], 2)
    miss = np.sqrt(sum) / (img_wm.shape[0] * img_wm.shape[1] * img_wm.shape[2]) * 100
    print 'Miss %s%% in save' % miss

    if debug:
        plt.subplot(233), plt.imshow(bgr_to_rgb(np.uint8(img_wm))), \
            plt.title('image(encoded)')
        plt.xticks([]), plt.yticks([])

    f2 = np.fft.fft2(img_wm)
    rwm = (f2 - f1) / alpha
    rwm = np.real(rwm)

    wm = np.zeros(rwm.shape)
    for i in xrange(int(rwm.shape[0] * 0.5)):
        for j in xrange(rwm.shape[1]):
            wm[m[i]][n[j]] = np.uint8(rwm[i][j])
    for i in xrange(int(rwm.shape[0] * 0.5)):
        for j in xrange(rwm.shape[1]):
            wm[rwm.shape[0] - i - 1][rwm.shape[1] - j - 1] = wm[i][j]

    if debug:
        assert cv2.imwrite('_bwm.debug.wm.jpg', wm)
        plt.subplot(236), plt.imshow(bgr_to_rgb(wm)), plt.title(u'watermark')
        plt.xticks([]), plt.yticks([])

    if debug:
        plt.show()

elif cmd == 'decode':
    print 'image<%s> + image(encoded)<%s> -> watermark<%s>' % (fn1, fn2, fn3)
    img = cv2.imread(fn1)
    img_wm = cv2.imread(fn2)

    if debug:
        plt.subplot(231), plt.imshow(bgr_to_rgb(img)), plt.title('image')
        plt.xticks([]), plt.yticks([])
        plt.subplot(234), plt.imshow(bgr_to_rgb(img_wm)), plt.title('image(encoded)')
        plt.xticks([]), plt.yticks([])

    random.seed(seed)
    m, n = range(int(img.shape[0] * 0.5)), range(img.shape[1])
    random.shuffle(m)
    random.shuffle(n)

    f1 = np.fft.fft2(img)
    f2 = np.fft.fft2(img_wm)
    if debug:
        plt.subplot(232), plt.imshow(bgr_to_rgb(np.real(f1))), \
            plt.title('fft(image)')
        plt.xticks([]), plt.yticks([])
        plt.subplot(235), plt.imshow(bgr_to_rgb(np.real(f1))), \
            plt.title('fft(image(encoded))')
        plt.xticks([]), plt.yticks([])

    rwm = (f2 - f1) / alpha
    rwm = np.real(rwm)

    if debug:
        plt.subplot(233), plt.imshow(bgr_to_rgb(rwm)), \
            plt.title('encrypted(watermark)')
        plt.xticks([]), plt.yticks([])

    wm = np.zeros(rwm.shape)
    for i in xrange(int(rwm.shape[0] * 0.5)):
        for j in xrange(rwm.shape[1]):
            wm[m[i]][n[j]] = np.uint8(rwm[i][j])
    for i in xrange(int(rwm.shape[0] * 0.5)):
        for j in xrange(rwm.shape[1]):
            wm[rwm.shape[0] - i - 1][rwm.shape[1] - j - 1] = wm[i][j]
    assert cv2.imwrite(fn3, wm)

    if debug:
        plt.subplot(236), plt.imshow(bgr_to_rgb(wm)), plt.title(u'watermark')
        plt.xticks([]), plt.yticks([])

    if debug:
        plt.show()
```

加上一串简单的命令，

```
python bwp.py decode 1.png  2.png flag.png
```

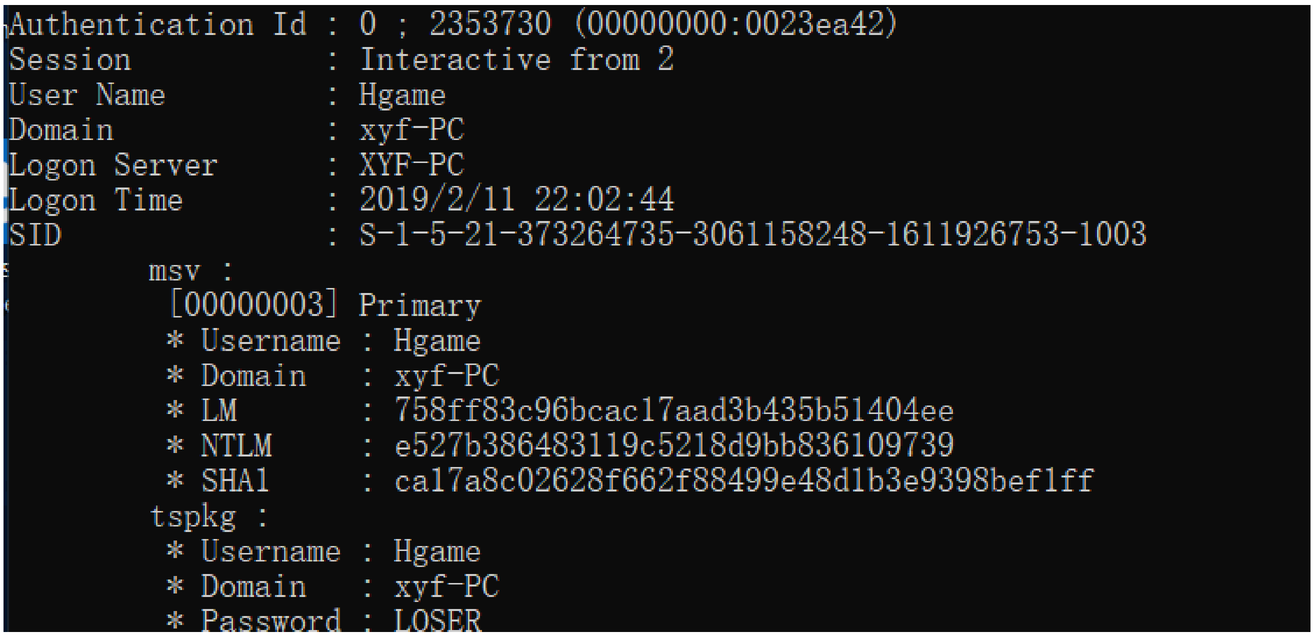
拿到有flag的图片



最终提交flag：hgame{h1de_in_THE_p1Cture}

Warmup

拿到文件，说是一个gif文件，可是却打不开，放进winhex，文件头是4D 44 4D 50。（挂羊头卖狗肉啊）坦白将之前没见过这样的文件头啊，网上搜来是dmp文件，呃，还是不懂怎么操作啊。找了半天分析这种文件方法，最后知道了一个叫mimikatz的软件。然而，无论以何种渠道下过来，电脑都把它当病毒杀了，嗯？！这还怎么做题，工具都用不了。询问了学长之后，给了个方法：在虚拟机里装一个windows系统，再下载。（噗，用着windows操作虚拟机里的windows，奇怪的感觉）那么就在虚拟机里运行mimikatz，打开文件如图



```
Authentication Id : 0 ; 2353730 (00000000:0023ea42)
Session           : Interactive from 2
User Name         : Hgame
Domain            : xyf-PC
Logon Server      : XYF-PC
Logon Time        : 2019/2/11 22:02:44
SID               : S-1-5-21-373264735-3061158248-1611926753-1003
        msv :
         [00000003] Primary
         * Username : Hgame
         * Domain   : xyf-PC
         * LM       : 758ff83c96bcac17aad3b435b51404ee
         * NTLM     : e527b386483119c5218d9bb836109739
         * SHA1     : ca17a8c02628f662f88499e48d1b3e9398bef1ff
        tspkg :
         * Username : Hgame
         * Domain   : xyf-PC
         * Password : LOSER
```

”管理员的密码的sha256“，这里找不到admin，只有Hgame，不过应该是了吧。密码是.....LOSER？！（这么打击人的嘛）然后在线sha256加密，

拿到flag：hgame{dd6dffcd56b77597157ac6c1beb514aa4c59d033098f806d88df89245824d3f5}