Hgame 2019 Writeup Week1

Author: Rainbow

I WEB

1. 谁吃了我的flag

Question

Description

呜呜呜，Mki一起床发现写好的题目变成这样了，是因为昨天没有好好关机吗T_T hint: 据当事人回忆，那个夜晚他正在用vim编写题目页面，似乎没有保存就关机睡觉去了,现在就是后悔，十分的后悔。

URL

http://118.25.111.31:10086/index.html

Base Score

50

Answer

由于Description可知，是vim没有保存。

所以拿到http://118.25.111.31:10086/.index.html.swp文件，

然后用vim打开，Recover即得：

```
<!DOCTYPE HTML>
<html>
        <head>
                <title>谁吃了我的flag???</title>
        </head>
        <body>
                <p>damn...hgame2019 is coming soon, but
the stupid Mki haven't finished his web-challenge...</p>
                </br>
                <p>fine, nothing serious, just give you
flag this time...</p>
                </br>
                <p>the flag is
hgame{3eek_diSclOSure_fRom+wEbsit@}
        </body>
</html>
```

所以flag就是`hgame{3eek_diSclOSure_fRom+wEbsit@}`

2. 换头大作战

Question

Description

想要flag嘛 工具: burpsuite postman hackbar 怎么用去百度，相信你可以的

URL

http://120.78.184.111:8080/week1/how/index.php

Base Score

100

Answer

进入网页后点了一下submit之后，出现了

> request method is error.I think POST is better

所以就用Postman进行POST咯。（然后还要添加一个字段want。。。

然后最后多了两行

> https://www.wikiwand.com/en/X-Forwarded-For only localhost can get flag

那么显然这个就和`X-Forwarded-For`有关了，又看到`only localhost can get flag`.

所以在header请求里加上`X-Forwarded-For: 127.0.0.1`

最后两行又变成了

> https://www.wikiwand.com/en/User_agent please use Waterfox/50.0

所以就是加上`User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; rv:57.0) Gecko/20100101 Waterfox/50.0`

又变成了

> https://www.wikiwand.com/en/HTTP_referer the requests should referer from www.bilibili.com

那就再加上`Referer: www.bilibili.com`

下一个。。。

> https://www.wikiwand.com/en/HTTP_cookie you are not admin

看来是cookie了呀。

怎么弄啊。。

然后就加了个`Cookie: not admin`

（我也不知道为什么这脑洞这么。。。

结果出来了`hgame{hTTp_HeaDeR_iS_Ez}`

3. very easy web

Question

Description

代码审计初♂体验

URL

http://120.78.184.111:8080/week1/very_ez/index.php

Base Score

100

Answer

打开拿到了php源码:

```
<?php
error_reporting(0);
include("flag.php");
```

```php
if(strpos("vidar",$_GET['id'])!==FALSE)
  die("<p>干巴爹</p>");

$_GET['id'] = urldecode($_GET['id']);
if($_GET['id'] === "vidar")
{
  echo $flag;
}
highlight_file(__FILE__);
?>
```

看来想要拿到flag，只要执行到`echo $flag;`，问题就解决了。

所以只要输入一个构造好的id就可以了。

问题是这个id他既不是`"vidar"`,又要进行`urldecode`后表示是`"vidar"`。

对于`"vidar"`而言，进行`urlencode`，其实就是十六进制的ASCII码，所以就是`%76%69%64%61%72`。

可是在输入到php时会自动进行一次`urldecode`,这样的话id拿到的其实还是`vidar`，

所以对`%76%69%64%61%72`在进行一次`urlencode`，得到

`%25%37%36%25%36%39%25%36%34%25%36%31%25%37%32`

然后访问http://120.78.184.111:8080/week1/very_ez/index.php?id=%25%37%36%25%36%39%25%36%34%25%36%31%25%37%32，得到flag为`hgame{urlDecode_Is_GoOd}`

4. can u find me?

Question

Description

为什么不问问神奇的十二姑娘和她的小伙伴呢 学习资料: https://www.cnblogs.com/yaoyaojing/p/9530728.html https://www.cnblogs.com/logsharing/p/8448446.html https://blog.csdn.net/z929118967/article/details/50384529

URL

http://47.107.252.171:8080/

Base Score

100

Answer

打开网页，看到

> the gate has been hidden
>
> can you find it? xixixi

(等于没说。。。

然后一把打开F12，看到body里面藏着个`<a href="f12.php"></a>`.

遂打开http://47.107.252.171:8080/f12.php，结果看到

> yeah!you find the gate
>
> but can you find the password?
>
> please post password to me! I will open the gate for you!

找了一下，发现了`Response Headers`里面的东东

> HTTP/1.1 200 OK Server: nginx/1.15.8 Date: Sun, 27 Jan 2019 14:01:33 GMT Content-Type: text/html; charset=UTF-8 Transfer-Encoding: chunked Connection: keep-alive X-Powered-By: PHP/7.2.14 **password: woyaoflag**

所以`password`就是`woyaoflag`

然后POST这个密码过去，网页变成了

```
<!DOCTYPE html>
<html>
    <head>
        <title>can u find me?</title>
    </head>
    <body>
        <p>yeah!you find the gate</p>
        <p>but can you find the password?</p>
        <p>please post password to me! I will open the
gate for you!</p>
        <p>right!</p>
        <a href='iamflag.php'> click me to get flag</a>
    </body>
</html>
```

然后打开`iamflag.php`，结果这是个302，然后跳转到`toofast.php`,显示了

> aoh,your speed is sososo fast,the flag must have been left in somewhere

显然真正的flag应该就藏在那个被跳过的`iamflag.php`里面。

然后用Burp Suite抓包，再发到Repeater里，重发一遍，拿到iamflag.php的Response

```
HTTP/1.1 302 Found
Server: nginx/1.15.8
Date: Mon, 28 Jan 2019 04:55:02 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
X-Powered-By: PHP/7.2.14
location: toofast.php
Content-Length: 132

<html>
    <head>
        <title>can you find me?</title>
    </head>
    <body>
        <p>flag:hgame{f12_1s_aMazIng111}</p>
    </body>
</html>
```

那么flag就是`hgame{f12_1s_aMazIng111}`

II RE

1. brainfxxker

Question

Description

Ouch! What is this? I don't think that I am pretty good at C++, what a brain fxxker it is! 学习资料: https://zh.wikipedia.org/wiki/Brainfuck https://zh.wikipedia.org/zh/ASCII 读懂我的代码逻辑答案就出来了 补充说明: 判定答案是否正确的是 Notice 2，即“不执行 [+.] 这个部分”，不要单纯看有没有输出 orz

URL

http://plir4axuz.bkt.clouddn.com/hgame2019/brainfucker.cpp

Base Score

100

Answer

得到源码如下

```
#include <iostream>
#include <cctype>

// Orz... I haven't learnt C++ before.
// It seems like my brain was fxxked by these codes...

// Notice:
// 1. the answer is your input when nothing strange was
printed
// 2. that is, wrong inputs will encounter with the part
"[+.]"
// 3. [!!!] REMEMBER TO WRAP YOUR ANSWER WITH "hgame{"
AND "}"
//    [!!!] BEFORE YOU SUBMITTED IT

// oyiadin, Jan 18, 2019
// enjoy it! ;)

namespace bf {

class Parser {
 public:
  Parser() = default;
```

```
  ~Parser() = default;
  void execute(const std::string &buf);

 protected:
  uint8_t data[100] = {0};
  int ptr = 0;
};


void Parser::execute(const std::string &buf) {
  for (auto i = buf.cbegin(); i != buf.cend(); ++i) {
    switch (*i) {
      case '>':
        ++ptr;
        break;
      case '<':
        --ptr;
        break;
      case '+':
        ++data[ptr];
        break;
      case '-':
        --data[ptr];
        break;
      case '.':
        putchar(data[ptr]);
        break;
      case ',':
        while ((data[ptr] = getchar()) == '\n') ;
        break;
      case '[':
        if (!data[ptr]) {
          while (*i++ != ']') continue;
          --i;
```

```
          }
          break;
        case ']':
          if (data[ptr]) {
            while (*(i-1) != '[') --i;
            --i;
          }
          break;
        default:
          break;
      }
    }
  }

  }


  int main() {
    bf::Parser parser;
    parser.execute(",>++++++++++[<---------->-]<++
  [+.],>+++++++++[<--------->-]<-[+.],>+++++++[<------->-]
  <---[+.],>++++++[<------>-]<+++[+.],>++++++++[<---------
  ->-]<++[+.],>++++++++++[<---------->-]<--
  [+.],>++++++++++[<-------->-]<-----[+.],>++++++++++[<---
  ------->-]<+[+.],>+++++++++[<-------->-]<---[+.]");
  }
```

最后的这一串实际上就是一堆指令，其中`,`是输入，`.`是输出，`<`和`>`是左移和右移指针`ptr`，`+`和`-`就是数组里的`ptr`指着的那个值自增和自减。

因为`,`是输入的意思，所以对于最后的字符串，其实应该分组成

`,>++++++++++[<---------->-]<++[+.]`

`,>+++++++++[<-------->-]<-[+.]`

`,>+++++++[<------>-]<---[+.]`

`,>++++++[<-----><]<+++[+.]`

`,>++++++++[<---------><]<++[+.]`

`,>++++++++++[<---------><]<--[+.]`

`,>++++++++++[<-------><]<----[+.]`

`,>++++++++++[<---------><]<+[+.]`

`,>+++++++++[<-------><]<---[+.]`

要是最后不输出什么奇怪的值，就要让每一行执行到最后一个`[`的时候，`ptr`指着的值应该是0，因为只有这样才能跳过最后的`[+.]`。

前面的一块操作，比如`>++++++++++[<---------><]`，其实就是一个循环，把输入的值减去`+`乘上后面连着的`-`的值。

然后在后面还有一块有若干个`+`或`-`，就是在前面循环的基础上，进行加减。

整理一下格式就是`,>{a个+}[<{b个-}><]<{c个+}[+.]`.

上面的`z个+`，如果是`-`号就等于`-z`。

所以设输入的值的ASCII码值为X，则$X-ab+z=0$，所以$X=ab-z$

依次算出之后就是

```
//    ",>++++++++++[<---------->-]<++[+.]" //98
//    ",>+++++++++[<---------->-]<-[+.]"    //82
//    ",>+++++++[<------->-]<---[+.]"      //52
//    ",>++++++[<------>-]<+++[+.]"        //33
//    ",>++++++++[<---------->-]<++[+.]"   //78
//    ",>++++++++++[<---------->-]<--[+.]" //102
//    ",>++++++++++[<--------->-]<-----[+.]"//85
//    ",>++++++++++[<---------->-]<+[+.]"  //99
//    ",>+++++++++[<--------->-]<---[+.]"); //75
```

得到的数字，查ASCII表拼起来得到 bR4!NfUcK

所以最后的flag就是`hgame{bR4!NfUcK}`

2. HelloRe

Question

Description

Welcooooooome!

URL

http://plps4kyke.bkt.clouddn.com/HelloRe

Base Score

50

Answer

这个东西就是直接UE看就是了，flag就明文藏在里面。

> 000007E0: 01 00 02 00 50 6C 65 61 73 65 20 69 6E 70 75 74
> ....Please.input

> 000007F0: 20 79 6F 75 72 20 6B 65 79 3A 00 68 67 61 6D 65 .your.key:.hgame
>
> 00000800: 7B 57 65 6C 63 30 6D 33 5F 74 30 5F 52 33 5F 57 {Welc0m3_t0_R3_W
>
> 00000810: 6F 72 6C 64 21 7D 00 73 75 63 63 65 73 73 00 66 orld!}.success.f
>
> 00000820: 61 69 6C 65 64 2E 2E 00 01 1B 03 3B 34 00 00 00 ailed.....;4...

所以flag就是`hgame{welc0m3_t0_R3_World!}`

3. わかります

Question

Description

POSITION ZERO!

URL

http://plps4kyke.bkt.clouddn.com/wakarimasu

Base Score

100

Answer

一把进入IDA，然后找到main，再F5反编译成C语言

```
__int64 __fastcall main(__int64 a1, char **a2, char
**a3)
```

```
{
  char s; // [rsp+0h] [rbp-40h]
  unsigned __int64 v5; // [rsp+38h] [rbp-8h]

  v5 = __readfsqword(0x28u);
  memset(&s, 0, 0x30uLL);
  puts("You are a good Reverser!");
  puts(off_602250);
  puts("wakalimasu.Give me your starlight!");
  fgets(&s, 47, stdin);
  if ( (unsigned __int8)sub_40094C(&s) )
    puts("you are top star!");
  else
    puts("non-non dayo~");
  return 0LL;
}
```

最后我们应该要让if里面的`(unsigned __int8)sub_40094C(&s)`成立，所以看下`sub_40094C`

```
__int64 __fastcall sub_40094C(const char *a1)
{
  unsigned __int8 v2; // [rsp+13h] [rbp-2Dh]
  signed int i; // [rsp+14h] [rbp-2Ch]
  signed int j; // [rsp+18h] [rbp-28h]
  signed int v5; // [rsp+1Ch] [rbp-24h]
  _DWORD *ptr; // [rsp+20h] [rbp-20h]
  _DWORD *v7; // [rsp+28h] [rbp-18h]
  _DWORD *v8; // [rsp+30h] [rbp-10h]
  _DWORD *v9; // [rsp+38h] [rbp-8h]

  v2 = 1;
  v5 = strlen(a1);
  if ( v5 > 37 )
```

```
    return 0LL;
  ptr = sub_400736(36);
  v7 = sub_400736(36);
  for ( i = 0; i < v5; ++i )
  {
    ptr[i] = (char)(a1[i] >> 4);
    v7[i] = a1[i] & 0xF;
  }
  v8 = sub_40078E((__int64)ptr, (__int64)&unk_602080,
6);
  v9 = sub_400892((__int64)v7, (__int64)&unk_602080, 6);
  for ( j = 0; j <= 35; ++j )
  {
    if ( v8[j] != dword_602120[j] || v9[j] !=
dword_6021C0[j] )
      v2 = 0;
  }
  free(ptr);
  free(v7);
  free(v8);
  free(v9);
  return v2;
}
```

那么a1其实就是我们的输入，并且应该长度是36。

然后的

```
for ( i = 0; i < v5; ++i )
  {
    ptr[i] = (char)(a1[i] >> 4);
    v7[i] = a1[i] & 0xF;
  }
```

就相当于把我们的输入的每个字符的前四位和后四位分别存入`ptr`和`v7`

再接着的

```
v8 = sub_40078E((__int64)ptr, (__int64)&unk_602080, 6);
v9 = sub_400892((__int64)v7, (__int64)&unk_602080, 6);
```

相当于把`ptr`和`v7`分别做了处理(这个接下来再说)，然后再赋值给v8和v9。

最后的

```
for ( j = 0; j <= 35; ++j )
  {
    if ( v8[j] != dword_602120[j] || v9[j] !=
dword_6021C0[j] )
      v2 = 0;
  }
return v2;
```

为了要让v2最后是1，所以也就是说：

对于所有的`j`,都要`v8[j] == dword_602120[j] && v9[j] == dword_6021C0[j]`成立

这里的`dword_602120`和`dword_6021C0`（包括上面的`unk_602080`）都是在ida里可以找到数据的。

好的，看回到上面的两个处理`ptr`和`v7`的函数：

第一个

```
_DWORD *__fastcall sub_40078E(__int64 a1, __int64 a2,
int a3)
{
  int v4; // [rsp+Ch] [rbp-34h]
  int i; // [rsp+2Ch] [rbp-14h]
  int j; // [rsp+30h] [rbp-10h]
```

```
  int k; // [rsp+34h] [rbp-Ch]
  _DWORD *v8; // [rsp+38h] [rbp-8h]

  v4 = a3;
  v8 = sub_400736(a3);
  for ( i = 0; i < v4; ++i )
  {
    for ( j = 0; j < v4; ++j )
    {
      for ( k = 0; k < v4; ++k )
        v8[v4 * i + j] += *(_DWORD *)(4LL * (v4 * i + k)
+ a1) * *(_DWORD *)(4LL * (v4 * k + j) + a2);
    }
  }
  return v8;
}
```

第二个

```
_DWORD *__fastcall sub_400892(__int64 a1, __int64 a2,
int a3)
{
  int v4; // [rsp+Ch] [rbp-24h]
  int i; // [rsp+20h] [rbp-10h]
  int j; // [rsp+24h] [rbp-Ch]
  _DWORD *v7; // [rsp+28h] [rbp-8h]

  v4 = a3;
  v7 = sub_400736(a3);
  for ( i = 0; i < v4; ++i )
  {
    for ( j = 0; j < v4; ++j )
      v7[v4 * i + j] = *(_DWORD *)(4LL * (v4 * i + j) +
a1) + *(_DWORD *)(4LL * (v4 * i + j) + a2);
```

```
    }
    return v7;
}
```

这个函数里面的`a1,2,3`用之前的变量代入其实就是：

```
for (i = 0; i < 6; ++i) {
    for (j = 0; j < 6; ++j) {
        for (k = 0; k < 6; ++k){
            v8[6 * i + j] += v6[6 * i + k] *
&unk_602080[6 * k + j];
        }
    }
}
```

```
for (i = 0; i < 6; ++i)
    for (j = 0; j < 6; ++j)
        v9[6 * i + j] = v7[6 * i + j]+ & unk_602080[6 *
i + j]
```

因为`v8`，`v9`应该等于`dword_602120`和`dword_6021C0`，

所以这个时候，我们只需要求出v6和v7就可以还原出我们的输入，也就是flag.

从ida中提取数据之后，分别如下

```
v8 = ['7A', 'CF', '8C', '95', '8E', 'A8',
      '5F', 'C9', '7A', '91', '88', 'A7',
      '70', 'C0', '7F', '89', '86', '93',
      '5F', 'CF', '6E', '86', '85', 'AD',
      '88', 'D4', 'A0', 'A2', '98', 'B3',
      '79', 'C1', '7E', '7E', '77', '93']
```

```
v9 = ['10', '08', '08', '0E', '06', '0B',
      '05', '17', '05', '0A', '0C', '17',
      '0E', '17', '13', '07', '08', '0A',
      '04', '0D', '16', '11', '0B', '16',
      '06', '0E', '02', '0B', '12', '09',
      '05', '08', '08', '0A', '10', '0D']
```

```
unk = [8, 1, 7, 1, 1, 0,
       4, 8, 1, 2, 3, 9,
       3, 8, 6, 6, 4, 8,
       3, 5, 7, 8, 8, 7,
       0, 9, 0, 2, 3, 4,
       2, 3, 2, 5, 4, 0]
```

显然代码中的运算都可以看做是6x6的矩阵运算。

所以可以得到

$V_8 = V_6 \cdot Unk$

$V_9 = V_7 + Unk$

所以

$V_6 = V_8 \cdot Unk^{-1}$

$V_7 = V_9 - Unk$

对于V7而言，只要执行

```
v7 = [(int(x, 16) - y) for (x, y) in zip(v9, unk)]
```

对于V8而言，则需要求逆，然后就在网上求了一下。

$$\begin{pmatrix} 8 & 1 & 7 & 1 & 1 & 0 \\ 4 & 8 & 1 & 2 & 3 & 9 \\ 3 & 8 & 6 & 6 & 4 & 8 \\ 3 & 5 & 7 & 8 & 8 & 7 \\ 0 & 9 & 0 & 2 & 3 & 4 \\ 2 & 3 & 2 & 5 & 4 & 0 \end{pmatrix}^{(-1)} = \begin{pmatrix} \frac{482}{8477} & \frac{1324}{8477} & \frac{-461}{8477} & \frac{-660}{8477} & \frac{-902}{8477} & \frac{192}{1211} \\ \frac{575}{16954} & \frac{-1871}{33908} & \frac{1503}{33908} & \frac{-2243}{33908} & \frac{5129}{33908} & \frac{9}{1211} \\ \frac{1399}{16954} & \frac{-5997}{33908} & \frac{2389}{33908} & \frac{2887}{33908} & \frac{3663}{33908} & \frac{-254}{1211} \\ \frac{-1947}{16954} & \frac{1087}{33908} & \frac{8297}{33908} & \frac{-6381}{33908} & \frac{-7873}{33908} & \frac{376}{1211} \\ \frac{821}{16954} & \frac{395}{33908} & \frac{-11771}{33908} & \frac{9535}{33908} & \frac{5967}{33908} & \frac{-143}{1211} \\ \frac{-468}{8477} & \frac{1685}{16954} & \frac{649}{16954} & \frac{543}{16954} & \frac{-1801}{16954} & \frac{-101}{1211} \end{pmatrix}$$

然后一乘

$$\begin{pmatrix} 122 & 207 & 140 & 149 & 142 & 168 \\ 95 & 201 & 122 & 145 & 136 & 167 \\ 112 & 192 & 127 & 137 & 134 & 147 \\ 95 & 207 & 110 & 134 & 133 & 173 \\ 136 & 212 & 160 & 162 & 152 & 179 \\ 121 & 193 & 126 & 126 & 119 & 147 \end{pmatrix} \times \begin{pmatrix} \frac{482}{8477} & \frac{1324}{8477} & \frac{-461}{8477} & \frac{-660}{8477} & \frac{-902}{8477} & \frac{192}{1211} \\ \frac{575}{16954} & \frac{-1871}{33908} & \frac{1503}{33908} & \frac{-2243}{33908} & \frac{5129}{33908} & \frac{9}{1211} \\ \frac{1399}{16954} & \frac{-5997}{33908} & \frac{2389}{33908} & \frac{2887}{33908} & \frac{3663}{33908} & \frac{-254}{1211} \\ \frac{-1947}{16954} & \frac{1087}{33908} & \frac{8297}{33908} & \frac{-6381}{33908} & \frac{-7873}{33908} & \frac{376}{1211} \\ \frac{821}{16954} & \frac{395}{33908} & \frac{-11771}{33908} & \frac{9535}{33908} & \frac{5967}{33908} & \frac{-143}{1211} \\ \frac{-468}{8477} & \frac{1685}{16954} & \frac{649}{16954} & \frac{543}{16954} & \frac{-1801}{16954} & \frac{-101}{1211} \end{pmatrix} = \begin{pmatrix} 6 & 6 & 6 & 6 & 6 & 7 \\ 3 & 5 & 7 & 6 & 6 & 6 \\ 6 & 5 & 4 & 6 & 7 & 7 \\ 3 & 7 & 5 & 6 & 7 & 5 \\ 7 & 6 & 7 & 7 & 5 & 7 \\ 7 & 6 & 6 & 3 & 6 & 7 \end{pmatrix}$$

最后我们可以得到这两个数组

```
v6 = [
    6, 6, 6, 6, 6, 7,
    3, 5, 7, 6, 6, 6,
    6, 5, 4, 6, 7, 7,
    3, 7, 5, 6, 7, 5,
    7, 6, 7, 7, 5, 7,
    7, 6, 6, 3, 6, 7,]
v7 = [8, 7, 1, 13, 5, 11,
      1, 15, 4, 8, 9, 14,
      11, 15, 13, 1, 4, 2,
      1, 8, 15, 9, 3, 15,
      6, 5, 2, 9, 15, 5,
      3, 5, 6, 5, 12, 13]
```

然后一拼

```
for c in [x * 16 + y for (x, y) in zip(v6, v7)]:
    print(chr(c), end="")
```

得到最后结果`hgame{1_think_Matr1x_is_very_usef5l}`

4. r & xor

Question

Description

论r 与 xor 的重要性 ida里奇怪的大数字?不如按r试一试

URL

http://plir4axuz.bkt.clouddn.com/hgame2019/xor

Base Score

100

Answer

显然先拉到IDA里去，看一下main：

```
int __cdecl main(int argc, const char **argv, const char
**envp)
{
  int result; // eax
  signed int i; // [rsp+8h] [rbp-138h]
  int v5[6]; // [rsp+10h] [rbp-130h]
  int v6; // [rsp+28h] [rbp-118h]
  ...
  int v29; // [rsp+94h] [rbp-ACh]
```

```
__int64 v30; // [rsp+A0h] [rbp-A0h]
__int64 v31; // [rsp+A8h] [rbp-98h]
__int64 v32; // [rsp+B0h] [rbp-90h]
__int64 v33; // [rsp+B8h] [rbp-88h]
int v34; // [rsp+C0h] [rbp-80h]
char s[104]; // [rsp+D0h] [rbp-70h]
unsigned __int64 v36; // [rsp+138h] [rbp-8h]

v36 = __readfsqword(0x28u);
v30 = 34839514623048026 64LL;
v31 = 685993493088052 0053LL;
v32 = 356022345849145 8926LL;
v33 = 238722599700715 0963LL;
v34 = 8200481;
memset(v5, 0, 0x90uLL);
v6 = 1;
...
v29 = 30;
puts("Input the flag:");
__isoc99_scanf("%s", s);
if ( strlen(s) == 35 )
{
  for ( i = 0; i < 35; ++i )
  {
    if ( s[i] != (v5[i] ^ *((char *)&v30 + i)) )
    {
      puts("Wrong flag , try again later!");
      return 0;
    }
  }
  puts("You are right! Congratulations!!");
  result = 0;
}
else
```

```
    {
      puts("Wrong flag , try again later!");
      result = 0;
    }
    return result;
  }
```

看下来，其实flag就是`s[i] == (v5[i] ^ *((char *)&v30 + i))`成立

i从0-34，其实对应的就是v5一直到v29的数，但是看这些的地址，中间是有空隙的，所以要补0，最后拿到这前面的35个数分别是

```
[0, 0, 0, 0, 0, 0, 1, 0, 7, 0, 92, 18, 38, 11, 93, 43,
11, 23, 0, 23, 43, 69, 6, 86, 44, 54, 67, 0, 66, 85,
126,
      72, 85, 30, 0]
```

后面的就对应着

```
v30 = 34839514623048026 64LL;
v31 = 68599349308805200 53LL;
v32 = 35602234584914589 26LL;
v33 = 23872259970071509 63LL;
v34 = 8200481;
```

一开始我就直接拼起来，化成Hex，然后按照char取出来进行异或运算，结果出来一堆奇怪的东西。

然后看到提示，说要按 R?

然后就变成了

```
v30 = 'OY{emagh';
v31 = '_3byam_u';
v32 = '1ht_deen';
v33 = '!!!enO_s';
v34 = '}!!';
```

嗯嗯嗯？？？

（应该是地址是减小的，所以要反过来拼）

显然要的就是`hgame{YOu_mayb3_need_th1s_One!!!!!}`

我直接提交了一下，发现并不行。

然后再拿着前面那个35个数字异或一下，变成了

`hgame{XOr_1s_interest1ng_isn't_it?}`

没错就是这个。

5. Pro的Python教室(一)

Question

Description

Easiest Python Challenge!

URL

http://plqbnxx54.bkt.clouddn.com/first.py

Base Score

100

Answer

点开网页，得到源码如下

```
import base64
import hashlib

enc1 = 'hgame{'
enc2 = 'SGVyZV8xc18zYXN5Xw=='
enc3 = 'PythOn}'

print 'Welcome to Processor\'s Python Classroom!\n'
print 'Here is Problem One.'
print 'There\'re three parts of the flag.'

print '-----------------------------------------------'

print 'Plz input the first part:'
first = raw_input()
if first == enc1:
    pass
else:
    print 'Sorry , You\'re so vegatable!'
    exit()

print 'Plz input the secend part:'
secend = raw_input()
secend = base64.b64encode(secend)
if secend == enc2:
    pass
else:
    print 'Sorry , You\'re so vegatable!'
    exit()

print 'Plz input the third part:'
third = raw_input()
```

```
third = base64.b32decode(third)
if third == enc3:
    pass
else:
    print 'Sorry , You\'re so vegatable!'
    exit()

print 'Oh, You got it !'
```

从`secend = base64.b64encode(secend)`看出来，里面的`enc2`明显就是base64编码了，然后base64解码之后，把`enc1,2,3`拼起来就是了。

`SGVyZV8xc18zYXN5Xw==`解码得到`Here_1s_3asy_`

最后flag就是`hgame{Here_1s_3asy_PythOn}`

（enc3这个不是应该也要加密一下么。。但是我忘了，但但是就过了。。。）

III PWN

2. aaaaaaaaaa

Question

Description

pwn很简单的，a上去就完事了 nc 118.24.3.214 9999

URL

http://plps4kyke.bkt.clouddn.com/aaaaaaaaaa

Base Score

50

Answer

下载这个文件，一把拉到IDA里，汇编看不来，看看C语言。

```
int __cdecl main(int argc, const char **argv, const char
**envp)
{
  signed int v3; // eax
  signed int v5; // [rsp+Ch] [rbp-4h]

  setbuf(_bss_start, 0LL);
  signal(14, (__sighandler_t)handle);
  alarm(0xAu);
  puts("Welcome to PWN'world!let us aaaaaaaaaa!!!");
  v5 = 0;
  while ( 1 )
  {
    v3 = v5++;
    if ( v3 > 99 )
      break;
    if ( getchar() != 97 )
      exit(0);
  }
  system("/bin/sh");
  return 0;
}
```

显然要执行到`system("/bin/sh");`才能搞事情，所以就是要让`v3`大于99，那么看下来只要输入好多好多的`aaaaaaaaaaaaaaaa...`就行了。（具体来讲大于99个a）

> ubuntu@VM-0-10-ubuntu:~/bin$ nc 118.24.3.214 9999 Welcome to PWN'world!let us aaaaaaaaaa!!! aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

然后这里就多了一个光标，等待执行命令。

那就来个`ls`，看看有啥吧。

> ls aaaaaaaaaa bin dev flag lib lib64 run.sh

哎哟，flag！那么打开flag就行了。

所以`vi flag`，结果：

> /bin/sh: 2: vi: not found

没有？那我看看你有啥命令。。。

执行`cd bin`和`ls`

> ls cat ls sh timeout

好吧好吧...那就`cat flag`

> cat flag hgame{Aa4_4aA_4a4aAAA}

那么flag就是`hgame{Aa4_4aA_4a4aAAA}`

(这里好像回不去上级菜单，所以我就重新nc一遍了）

IV MISC

---

1. Hidden Image in LSB

Question

Description

Here are some magic codes which can hide information in an ordinary picture, can you extract the hidden image in the provided picture? 其实本来想让大家写写代码，后来干脆就送分了 有个神器叫 stegsolve，利用它可以直接提取本题 flag

URL

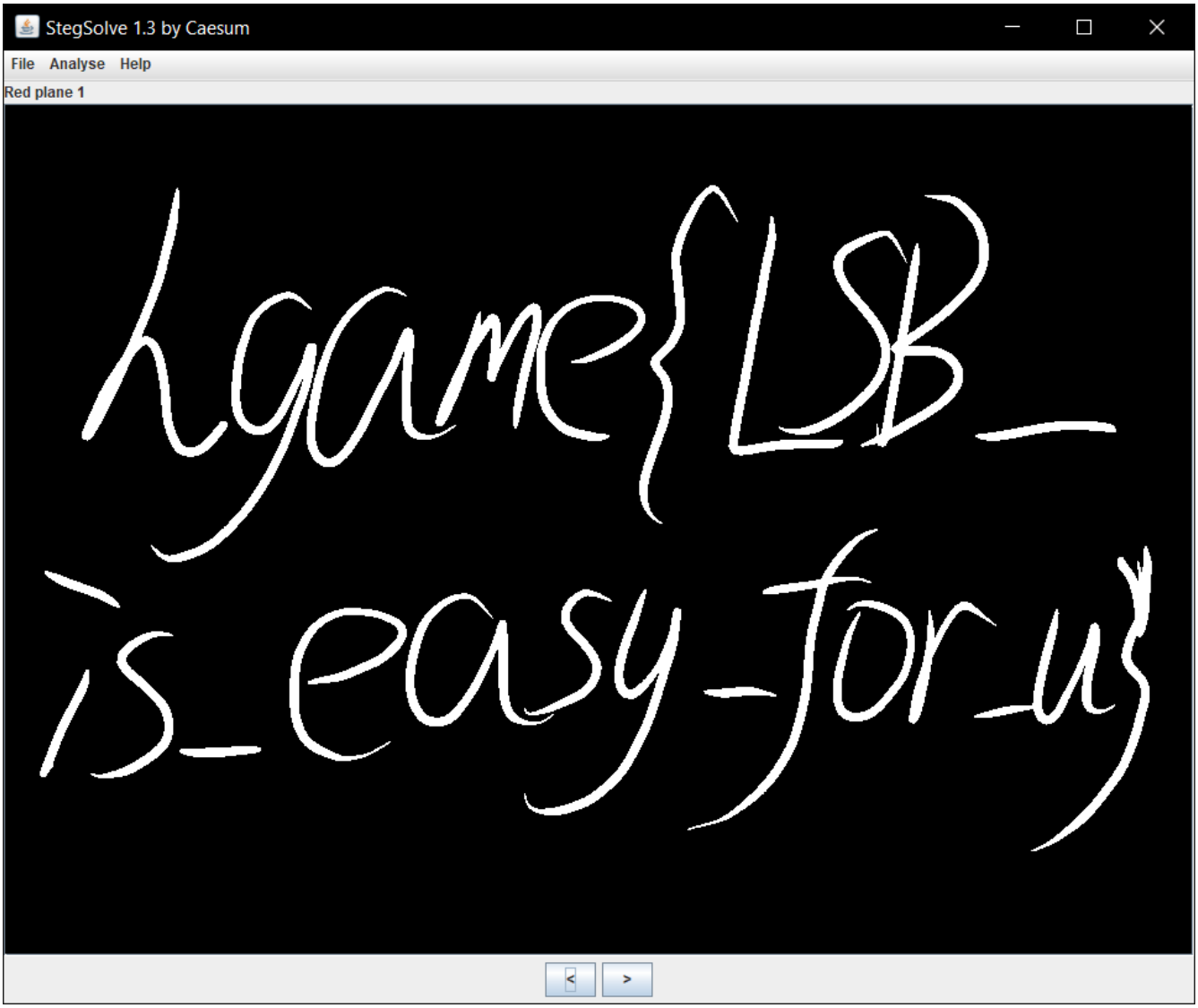
http://plir4axuz.bkt.clouddn.com/hgame2019/lsb.zip

Base Score

50

Answer

这个就是看每个像素的最后一位，然后放进stegsolve，真的就出来了。



没错，flag就是`hgame{LSB_is_easy_for_u}`

2. 打字机

Description

Aris(划掉)牌打字机，时尚时尚最时尚~ hint:谷歌有个以图搜图功能很不错，百度识图好垃圾的。。。

URL

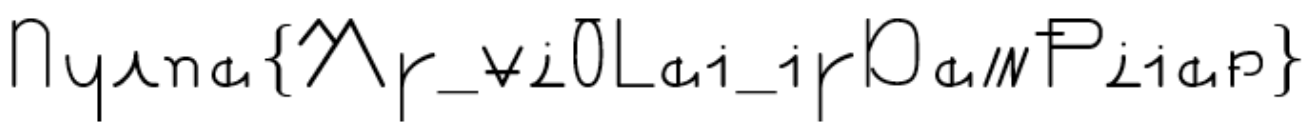
http://plps4kyke.bkt.clouddn.com/打字机.zip

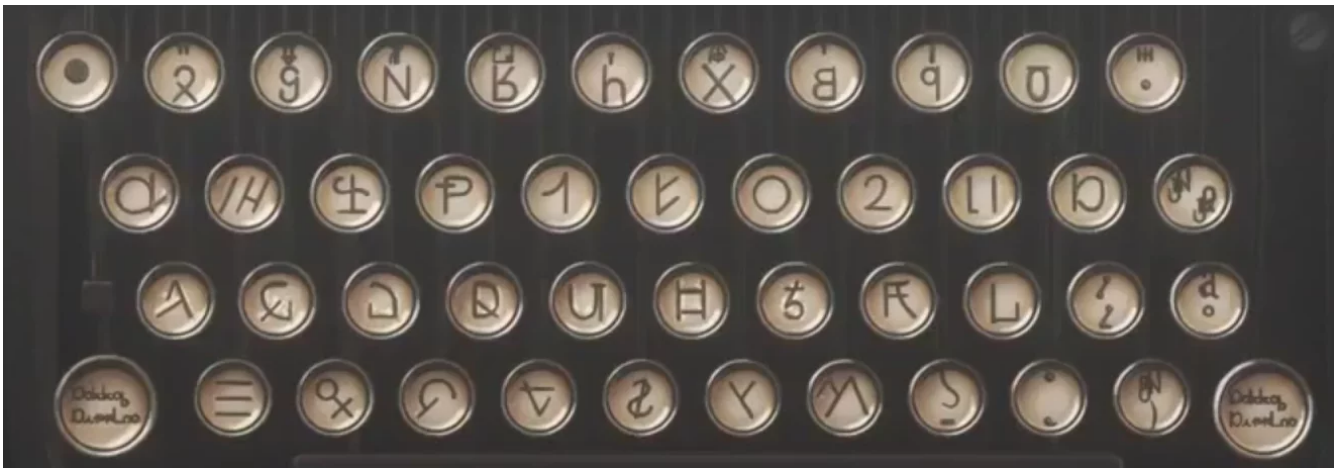
Base Score

50

Answer

拿到zip后，发现里面有两张图片。





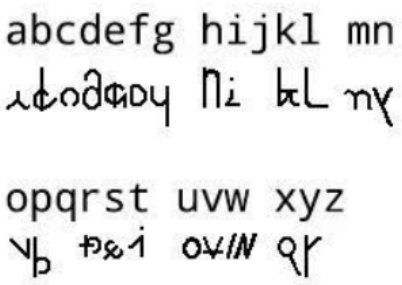
第一个看上去就像是hgame{...}的感觉，然后下面的这个键盘第一感觉就是对于QWERTY键盘。所以只要翻译一下就行了。

但是问题是flag里面有好多字母，下面都没有。

看来打字机上是大写，那么我们还需要小写版的对应表。

Google搜图之后发现了，早有牛人对出了小写版的。

至此小写字母终于全部解读完成，得到了小写字母对照表（J Q Z的小写字符并没有出现）：



翻译之后，就拿到了`hgame{My_viOlet_tyPewRiter}`

3.Broken Chest

Question

Description

这个箱子坏掉了！快用你无敌的[疯狂钻石]想想办法啊！ 更新一波学习资料 https://ctf-wiki.github.io/ctf-wiki/misc/archive/zip/

URL

http://plqfgjy5a.bkt.clouddn.com/Broken-Chest.zip

Base Score

50

Answer

拿到压缩包，结果坏了。

看十六进制，开头的50 4B 03 04被改了，改回来。

打开压缩包，结果flag.txt被锁住了。

~~但是可以看到大小是20B，CRC是b0b37cce~~

~~所以爆破这20B的内容，找到一样的CRC。~~

~~爆个头，爆完电脑都没了~~

突然发现压缩包里有个注释`SOmETh1ng_U5efuL`.

嗯？这就是密码？没错，用这个打开得到`hgame{Cra2y_D1aMOnd}`

4. Try [UNSOLVED]

Question

Description

无字天书

URL

http://plqfgjy5a.bkt.clouddn.com/try-it.pcapng

Base Score

100

V CRYPTO

1. Mix

Question

Description

--.../....-/....-/-.../--.../...--/...../..-./-..../-../-..../..-./--.../----./....-/....-/--.../.---/-..../-.../--.../-.../-..../..---/...../.----/-..../-..../...--/....-/...--/-----/-..../...../--.../-.. So easy

URL

http://example.com

Base Score

50

Answer

这玩意一看就是个摩斯电码，翻译一波。

结果拿到了`744B735F6D6F7944716B7B6251663430657D`

感觉是个十六进制编码，然后按照ASCII码翻译一下，拿到了`tKs_moyDqk{bQf40e}`

这个看上去好像就要成功了，但是还是莫名其妙的。

和我们想要的hgame{...}看上去，大小括号都有了，但是左大括号的位置不对。

两个一组进行栅栏重排，得到了`tsmyq{Q4eK_oDkbf0}`

这个看上去就像是`hgame{..........}`的偏移版。

然后发现规律就是这个

> ABCDEFGHIJKLMNOPQRSTUVWXYZ
> OPQRSTUVWXYZABCDEFGHIJKLMN

全部对应下来就得到了`hgame{E4sY_cRypt0}`

2. perfect_secrecy!

Question

Description

Mom told me OTP is perfect secrecy!（结果加上hgame{}）

URL

http://plqbnxx54.bkt.clouddn.com/easy_otp.py

Base Score

100

Answer

拿到源码，看下

```
import binascii
import string
import random

def strxor(a, b):
    return "".join(hex(x ^ y)[2:].zfill(2) for (x, y) in
zip(a, b))


fp = open('poem.txt', 'rb')
flag = "**********************************"
strings = fp.readlines()
key = hex(random.randint(2**511, 2**512))[2:]
strs = [strxor(i[:-3], binascii.unhexlify(key)) for i in
strings]
```

```
result = strxor(flag.encode('utf-8'),
binascii.unhexlify(key))
print(strs)
print(result)

'''
output:
['daaa4b4e8c996dc786889cd63bc4df4d1e7dc6f3f0b7a0b61ad488
11f6f7c9bfabd7083c53ba54',
'c5a342468c8c7a88999a9dd623c0cc4b0f7c829acaf8f3ac13c7830
0b3b1c7a3ef8e193840bb',
'dda342458c897a8285df879e3285ce511e7c8d9afff9b7ff15de8a1
6b394c7bdab920e7946a05e9941d8308e',
'd9b05b4cd5ce7c8f938bd39e24d0df191d7694dfeaf8bfbb56e2890
0e1b8dff1bb985c2d5aa154',
'd9aa4b00c88b7fc79d99d38223c08d54146b88d3f0f0f38c03df8d5
2f0bfc1bda3d7133712a55e9948c32c8a',
'c4b60e46c9827cc79e9698936bd1c55c5b6e87c8f0febdb856fe805
2e4bfc9a5efbe5c3f57ad4b9944de34',
'd9aa5700da817f94d29e81936bc4c1555b7b94d5f5f2bdff37df825
2ffbecfb9bbd7152a12bc4fc00ad7229090',
'c4e24645cd9c28939a86d3982ac8c819086989d1fbf9f39e18d5c60
1fbb6dab4ef9e12795bbc549959d9229090',
'd9aa4b598c80698a97df879e2ec08d5b1e7f89c8fbb7beba56f0c61
9fdb2c4bdef8313795fa149dc0ad4228f',
'cce25d48d98a6c8280df909926c0de19143983c8befab6ff21d99f5
2e4b2daa5ef83143647e854d60ad5269c87',
'd9aa4b598c85668885df9d993f85e419107783cdbee3bbba1391b11
afcf7c3bfaa805c2d5aad42995ede2cdd82977244',
'e1ad40478c82678995df809e2ac9c119323994cffbb7a7b713d4c62
6fcb888b5aa920c354be853d60ac5269199',
'c4ac0e53c98d7a8286df84936bc8c84d5b50889aedfebfba18d2835
2daf7cfa3a6920a3c',
```

```
'd9aa4f548c9a609ed297969739d18d5a146c8adebef1bcad11d4925
2c7bfd1f1bc87152b5bbc07dd4fd226948397',
'c4a40e698c9d6088879397d626c0c84d5b6d8edffbb792b902d4945
2ffbec6b6ef8e193840',
'c5ad5900df8667929e9bd3bf6bc2df5c1e6dc6cef6f2b6ff21d8921
ab3a4c1bdaa991f3c12a949dd0ac5269c']
'c2967e7fc59d57899d8bac852ac3c866127fb9d7f1e5b68002d9871
cccb8c6b2aa'
'''
```

可以看到，这个代码随机生成了一个512位的密码，然后对所有的字符串进行异或运算。

由于这个加密是按照一个字节一个字节算的，所以一开始我拿出了所有字符串的前两个。

`['da','c5','dd','d9','d9','c4','d9','c4','d9','cc','d9','e1','c4','d9','c4','c5']`

又因为异或两次会不变，所以我对上面的这些分别从0-255进行异或，然后去寻找那些恰好可以让所有的字符都是字母的值。

代码如下：

```
import binascii

s = ['da', 'c5', 'dd', 'd9', 'd9', 'c4', 'd9', 'c4',
'd9', 'cc', 'd9', 'e1', 'c4', 'd9', 'c4', 'c5']

for key in range(0, 256):
    answers = [chr(binascii.unhexlify(t)[0] ^ key) for t
in s]
    f = True
    for answer in answers:
        f = f & (('a' <= answer <= 'z') | ('A' <= answer
<= 'Z') | (answer == ' '))
    if f:
        print(key)
```

但是一运行，结果却让我有点懵逼：

> 136 137 138 139 141 142 143 148 149 150 168 169 170 171 173 174 175 180 181 182

只是第一个就有这么多种？那后面会有多少种啊。。。

但是呢，我感觉我的思路应该是对的，所以干脆把后面的都先弄出来。

结果后面的值都是唯一的，所以我就直接输出flag的值了

代码如下：

```
import binascii

s =
['daaa4b4e8c996dc786889cd63bc4df4d1e7dc6f3f0b7a0b61ad488
11f6f7c9bfabd7083c53ba54',
```

```
'c5a342468c8c7a88999a9dd623c0cc4b0f7c829acaf8f3ac13c78300b3b1c7a3ef8e193840bb',

'dda342458c897a8285df879e3285ce511e7c8d9afff9b7ff15de8a16b394c7bdab920e7946a05e9941d8308e',

'd9b05b4cd5ce7c8f938bd39e24d0df191d7694dfeaf8bfbb56e28900e1b8dff1bb985c2d5aa154',

'd9aa4b00c88b7fc79d99d38223c08d54146b88d3f0f0f38c03df8d52f0bfc1bda3d7133712a55e9948c32c8a',

'c4b60e46c9827cc79e9698936bd1c55c5b6e87c8f0febdb856fe8052e4bfc9a5efbe5c3f57ad4b9944de34',

'd9aa5700da817f94d29e81936bc4c1555b7b94d5f5f2bdff37df8252ffbecfb9bbd7152a12bc4fc00ad7229090',

'c4e24645cd9c28939a86d3982ac8c819086989d1fbf9f39e18d5c601fbb6dab4ef9e12795bbc549959d9229090',

'd9aa4b598c80698a97df879e2ec08d5b1e7f89c8fbb7beba56f0c619fdb2c4bdef8313795fa149dc0ad4228f',

'cce25d48d98a6c8280df909926c0de19143983c8befab6ff21d99f52e4b2daa5ef83143647e854d60ad5269c87',

'd9aa4b598c85668885df9d993f85e419107783cdbee3bbba1391b11afcf7c3bfaa805c2d5aad42995ede2cdd82977244',

'e1ad40478c82678995df809e2ac9c119323994cffbb7a7b713d4c626fcb888b5aa920c354be853d60ac5269199',
```

```
'c4ac0e53c98d7a8286df84936bc8c84d5b50889aedfebfba18d28352daf7cfa3a6920a3c',

'd9aa4f548c9a609ed297969739d18d5a146c8adebef1bcad11d49252c7bfd1f1bc87152b5bbc07dd4fd226948397',

'c4a40e698c9d6088879397d626c0c84d5b6d8edffbb792b902d49452ffbec6b6ef8e193840',

'c5ad5900df8667929e9bd3bf6bc2df5c1e6dc6cef6f2b6ff21d8921ab3a4c1bdaa991f3c12a949dd0ac5269c']

flag = 'c2967e7fc59d57899d8bac852ac3c866127fb9d7f1e5b68002d9871cccb8c6b2aa'

for i in range(0, 33):
    for key in range(0, 256):
        answers = [chr(binascii.unhexlify(t[i * 2:i * 2 + 2])[0] ^ key) for t in s]
        f = True
        for answer in answers:
            f = f & (('a' <= answer <= 'z') | ('A' <= answer <= 'Z') | (answer == ' '))
        if f:
            print(chr(binascii.unhexlify(flag[i * 2:i * 2 + 2])[0] ^ key), end="")
```

最后运行得到

> JKHIOLMVWTjkhiolmvwtTP_is_not_safe_if_more_than_once

前面的`JKHIOLMVWTjkhiolmvwt`就是来自第一个的，后面的`TP_is_not_safe_if_more_than_once`很明显就是flag的一部分了。

那么很显然，第一个字母就是O，然后得到flag：

`hgame{OTP_is_not_safe_if_more_than_once}`

3. Base全家

Question

Description

全家老小

URL

http://plir4axuz.bkt.clouddn.com/hgame2019/enc.txt

Base Score

50

Answer

既然说了是base全家，那么就是base16,base32,base64全上呗。

用Python写了个

```
import base64
import binascii


def b64(t): return base64.b64decode(t)
def b32(t): return base64.b32decode(t)
def b16(t): return base64.b16decode(t)
```

```
file = open("enc.txt", "rb")
text = file.read()

while 1:
    try:
        text = b16(text)
        print(text)
    except binascii.Error:
        try:
            text = b32(text)
            print(text)
        except binascii.Error:
            text = b64(text)
            print(text)
```

运行一波，最后虽然报错了，但是在输出里面发现了一行

`b'base58 : 2BAja2VqXoHi9Lo5kfQZBPjq1EmZHGEudM5JyDPREPmS3CxrpB8BnC'`

base58解码之后就是`hgame{40ca78cde14458da697066eb4cc7daf6}`