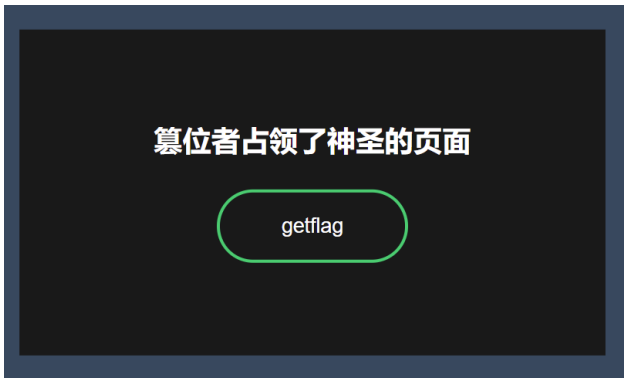
Web 部分

Web1 LazyDogR4U

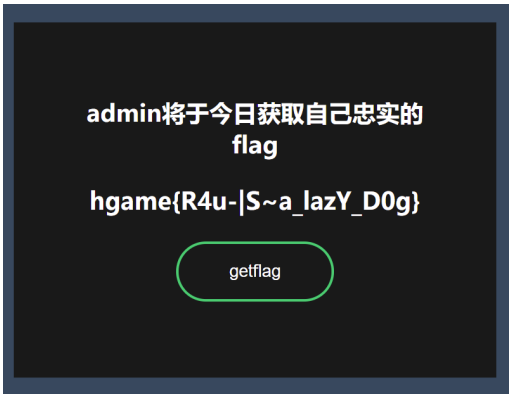
打开，没什么头绪，跑了个扫描，发现有 www.zip，果断下载。~~（当时线程调太高了，IP 被封了，吓了我一跳，换了个 IP 继续。后来，放了 hint，感觉亏大了……）~~

毕竟不是搞 web 的，看了几天没什么想法，直到有了 hint，得知是变量覆盖，赶紧去学。发现首先要登录 `testuser` 账户。去查了下 PHP 的 md5，发现 0e 开头的都会被认为相同，于是找了个 `240610708`，成功登录。



然后开始实施变量覆盖。

后面直接接`?_SESSION[username]=admin`，发现并不能成功。重新查看代码后发现了 `filter`，`SESSION` 变量会被替换，那就试试接`?_SESSESSIONSION[username]=admin`（里面插入了个 `SESSION`），完美：



Reverse 部分

Reverse1 ezApk

Android 逆向。

用 JEB 打开 apk，发现被混淆了。找到 com.ryen.ezapk 包，定位到了关键函数。

花了很久去看逻辑，终于搞明白了就是一个 AES。

结合 JEB 的注释中自动标出的字符串值，写出解密代码

```
from Crypto.Cipher import AES
from base64 import b64decode
import hashlib

key = hashlib.sha256('A_HIDDEN_KEY'.encode('ascii')).digest()
iv = hashlib.md5('A_HIDDEN_KEY'.encode('ascii')).digest()
data = b64decode(b'EEB23sI1Wd9Gvhvk1sgWyQZhjilnYwCi5au1guzOaIg5dMAj
                   9qPA7lnIyVoPSdRY')

cipher = AES.new(key, AES.MODE_CBC, iv)
print(cipher.decrypt(data).decode('ascii'))
```

运行得到 flag：hgame{jUst_A_3z4pp_write_in_k07l1n}

Reverse3 fake_debugger beta

~~*我保证我会被出题人打死*~~

根据直觉，试了试 hgame{，长度不够，于是 hgame{aaaa：

发现每次单步执行有 eax、ebx、ecx、zf 四个寄存器。推测每个字母有两次判断，eax 与自己输入的有关；ebx 的第一次值似乎没啥用，第二次与 eax 匹配就继续；ecx 表示目前进度；zf 表示两次判断中的第几次。

然后，琢磨了几次后发现 eax 同一字母每一位的值不一样，于是，决定采用最笨的办法：人肉暴力尝试……

对，就是凑，差了 20-30，多半是大小写错了，怎么都不对，可能是下划线。

最后得到 flag：hgame{You_Kn0w_debuGg3r}

~~*（可把我累坏了，没有功劳也有苦劳，求出题人放过……）*~~

Crypto 部分

Crypto1 signin

首先看到 `c = a ** p * m % p`，感觉很头疼。突然想到了费马小定理，直接化简得到：

`c = a * m % p`

这下好多了。

然后看出这就是个同余方程：m: $am = bc + p, b \in \mathbb{Z}$，于是使用 gmpy2 库求模反，直接得答案：

```python
from gmpy2 import mpz, invert
from libnum import n2s
a = mpz("……")   # 省略
p = mpz("……")   # 省略
c = mpz("……")   # 省略
print(n2s(int(invert(a, p) * c % p)).decode("ascii"))
```

运行得到 flag：`hgame{M0du1@r_m4th+1s^th3~ba5is-Of=cRypt0!!}`

~~（顺便一提，每次下载的 a、p、c 居然是不一样的……）~~

Crypto2 gcd or more?

一看就是 RSA，然后发现 e=2，这么奇特，去查了查，发现了 Rabin 算法。

直接解密：

```
import gmpy2
import libnum

c = ……   # 省略
p = ……   # 省略
q = ……   # 省略
n = p * q
u = pow(c, (p + 1) // 4, p)
v = pow(c, (q + 1) // 4, q)
s = gmpy2.invert(p, q)
t = gmpy2.invert(q, p)
x = (t * q * u + s * p * v) % n
y = (t * q * u - s * p * v) % n

print(libnum.n2s(int((-x) % n)))
print(libnum.n2s(int(x % n)))     # 经测试，这个是乱码
print(libnum.n2s(int(y % n)))     # 经测试，这个是乱码
print(libnum.n2s(int((-y) % n)))  # 经测试，这个是乱码
```

得到 flag：hgame{3xgCd~i5_re4l1y+e@sy^r1ght?}

Crypto3 WhitegiveRSA

真・白给

使用 RsaCtfTool，直接：

```
python3 RsaCtfTool.py -n 882564595536224140639625987659416029426239230804614613279163 -e 65537 --uncipher 747831491353896780365654517748216624798517769637260742155527
```

得到：

```
Unciphered data :
HEX : 0x006867616d657b7730777e794f555f6b4e6f572b523540217d
INT (big endian) : 25599744719368613322506956018967498313805867172277298220 77
INT (little endian) : 78545302450082053438307133470272813832511552122445573543 7312
STR : b'\x00hgame{w0w~yOU_kNoW+R5@!}'
```

MISC 部分

MISC1 Tools

首先得到了一个压缩包，解压，一个加密的 7z 文件，一个图片。看 7z 文件名，应该是 F5

隐写。在图片的备注里找到了密码，解密：

```
java Extract /path/to/Matryoshka.jpg -p "!LyJJ9bi&M7E72*JyD"
```

在 output.txt 中找到密码：`e@317S*p1A4bIYIs1M`，解密，又得到两个文件。同样的逻

辑，上 Steghide 隐写，密码同样找：

```
steghide extract -sf ~/01.jpg -p "A7SL9nHRJXLh@\$EbE8"
```

得到 pwn.txt，密码：`u0!FO4JUhl5!L55%$&`

解压，outguess 隐写：

```
outguess -r 02.jpg -k "z0GFieYAee%gdf0%lF" out.txt
```

得到 out.txt，密码：`@UjXL93044V5zl2ZKI`

解压，JPHS 隐写。这个用了 jphswin 工具，在 GUI 页面，点击 Seek，输入密码~~（不知道

为什么要输入两次）~~，然后保存导出文件。打开，得到密码：`xSRejK1^Z1Cp9M!z@H`

至此，得到了 01.jpg、02.jpg、03.jpg、04.jpg 四个二维码碎片。丢进 Photoshop 拼合

扫描，得到 flag：`hgame{Taowa_is_N0T_g00d_but_T001s_is_Useful}`

MISC2 Telegraph

~~*（题目后面的那一串数字是什么？）*~~

听了一遍，发现从 1:10-1:33 有摩尔电码。故用 GlodWave 打开，截取，把除 1000Hz 频率

的全部关闭，得到清晰的频谱，开始摘录，得到：

```
-.--/---/..-/.-./..-./.-../.-/--./../.../---.../....-/--./-----/-
----/-../.../-----/-./--./-.../..-/-/-./-----/-/....-/--./-----/-
----/-../--/.-/-./-----/...--/----./...--/.----/-----/-.-/..
```

转码：`YOURFLAGIS:4G00DS0NGBUTN0T4G00DMAN039310KI`

故 flag：`hgame{4G00DS0NGBUTN0T4G00DMAN039310KI}`

MISC3 Hallucigenia

先用 Stegsolve 打开，逐一翻找，发现一个二维码：

截出来，反色，得到一个正常的二维码：

扫描，得到一串 Base64。（具体见下方 Python 脚本）

网页解码发现末尾 GNP 字样，同时结合 “我们不仅弄错了他的上下，还颠倒了它的左右。”，

意识到是 PNG。故写 Python 脚本：

```python
import base64
coded_string = '''gmBCrkRORUkAAAAA+jrgsWajaq0BeC3IQhCEIQhCKZw1MxTzS
lNKnmJpivW9IHVPrTjvkkuI3sP7bWAEdIHWCbDsGsRkZ9IUJC9AhfZFbpqrmZBtI+Z
vptWC/KCPrL0gFeRPOcI2WyqjndfUWlNj+dgWpe1qSTEcdurXzMRAc5EihsEflmIN8
RzuguWq61JWRQpSI51/KHHT/6/ztPZJ33SSKbieTa1C5koONbLcf9aYmsVh7RW6p3S
pASnUSb3JuSvpUBKxscbyBjiOpOTq8jcdRsx5/IndXw3VgJV6iO1+6jl4gjVpWouVi
O6ih9ZmybSPkhaqyNUxVXpV5cYU+Xx5sQTfKystDLipmqaMhxIcgvplLqF/LWZzIS5
PvwbqOvrSlNHVEYchCEIQISICSZJijwu50rRQHDyUpaF0y///p6FEDCCDFsuW7YFoV
EFEST0BAACLgLOrAAAAAggUAAAAtAAAAFJESEkNAAAAChoKDUdOUIk='''
with open('out.png', 'wb') as f:
    f.write(base64.b64decode(coded_string)[::-1])
```

得到 out.png：

那句话还真的对……果然是颠倒的……

翻过来就是结果：hgame{tenchi_souzou_dezain_bu}

MISC4 DNS

用 Wireshark 打开，导出 HTTP 对象，找到 `flag.hgame2021.cf`

发了个 post，得到了提示：

```
<html>

<head>
</head>

<body>
    <script>
        while(true){
            alert("Flag is here but not here")
        }
    </script>
    <b>Do you know SPF?</b>
</body>

</html>
```

SPF？好办。

```
~ nslookup -q=txt flag.hgame2021.cf
服务器:  UnKnown
Address:  192.168.0.1

非权威应答:
flag.hgame2021.cf       text =

        "hgame{D0main_N4me_5ystem}"
```

针不戳。