wechatdb-siebel

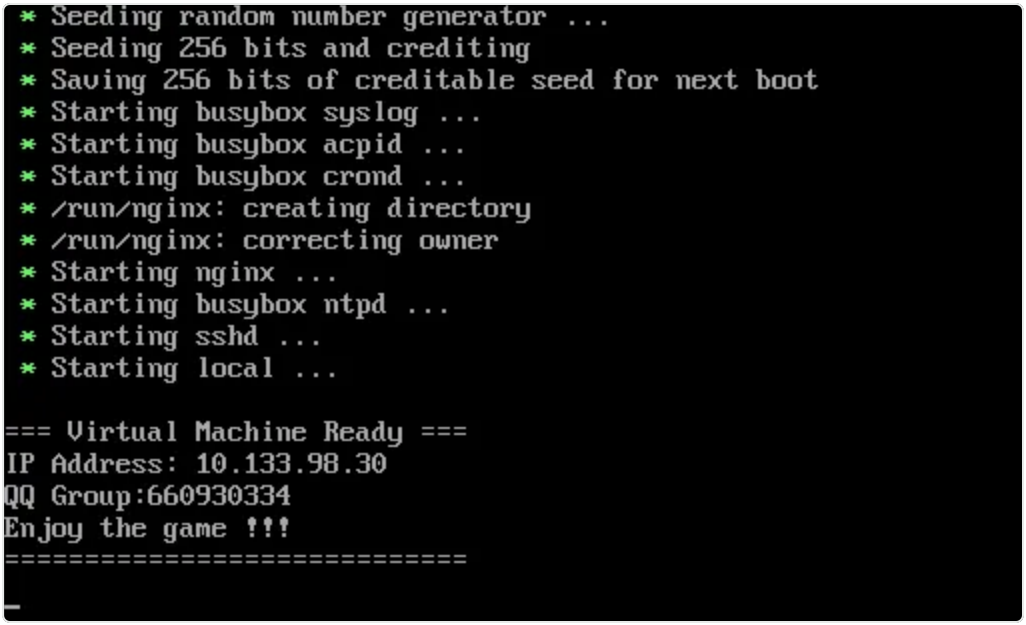
端口扫描

访问 80

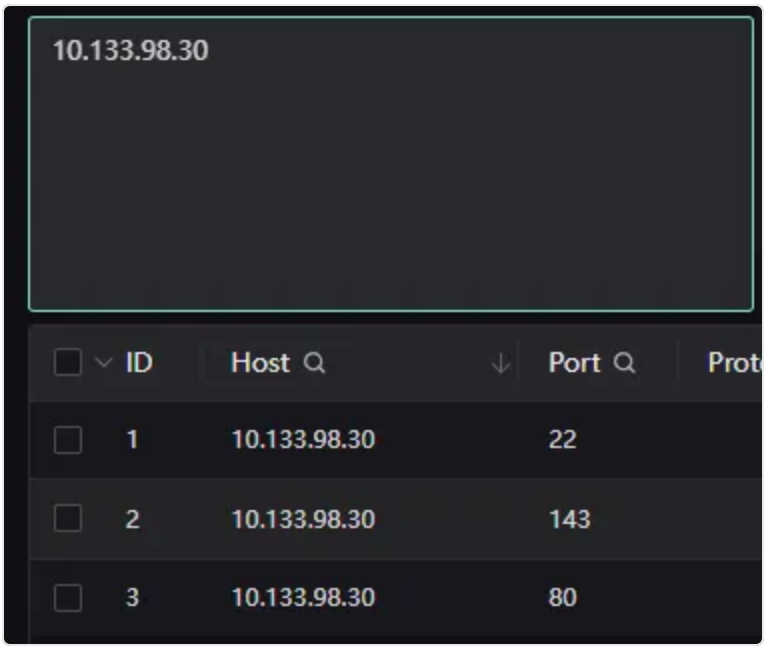
目录扫描

【我的进度到这了，其它找不到了，后来看群友发的 wp 看到是还有一段 是在 图片里，但是页面上并没有=-=...

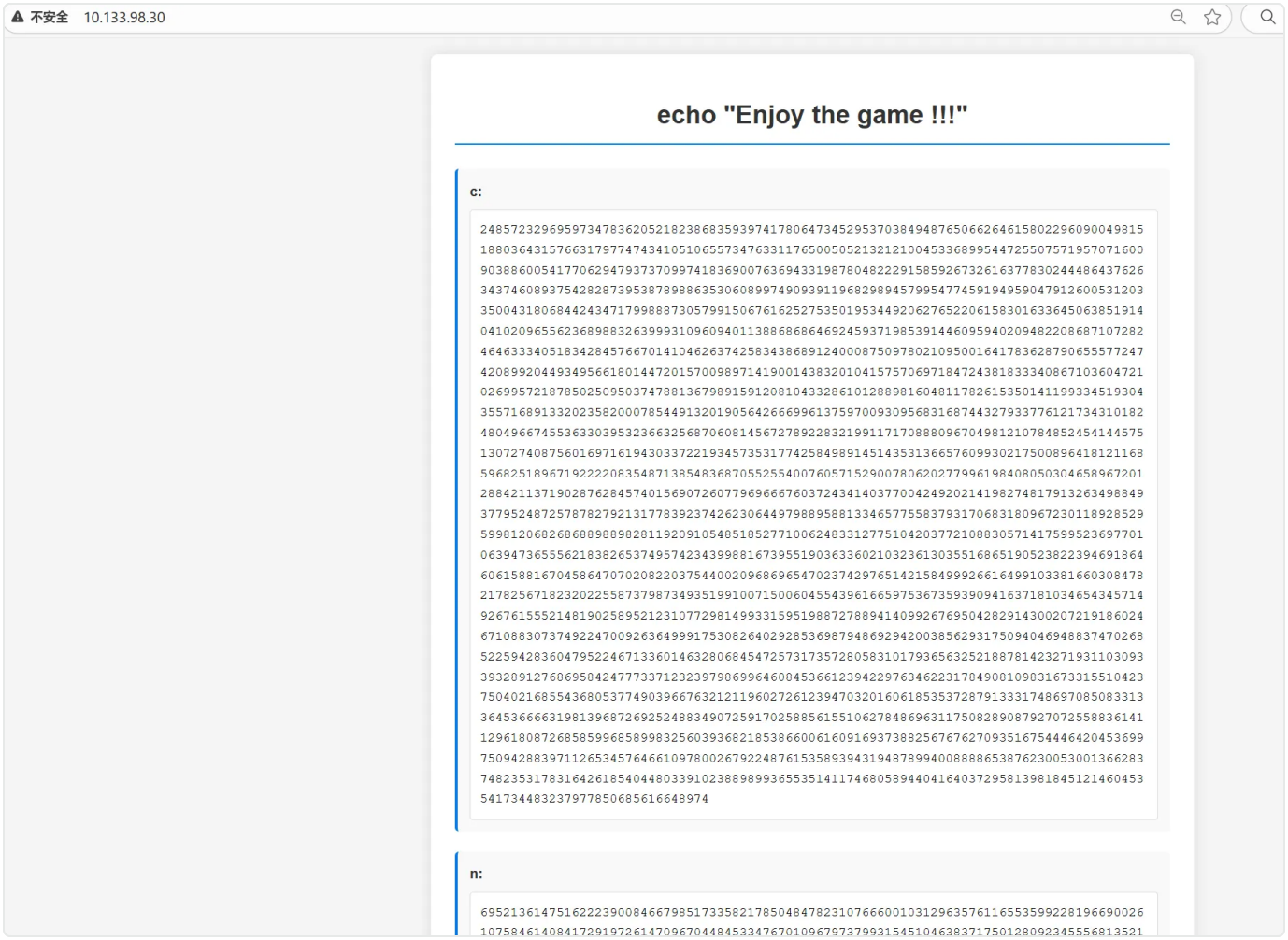
打开靶机直接给了 ip:10.133.98.30



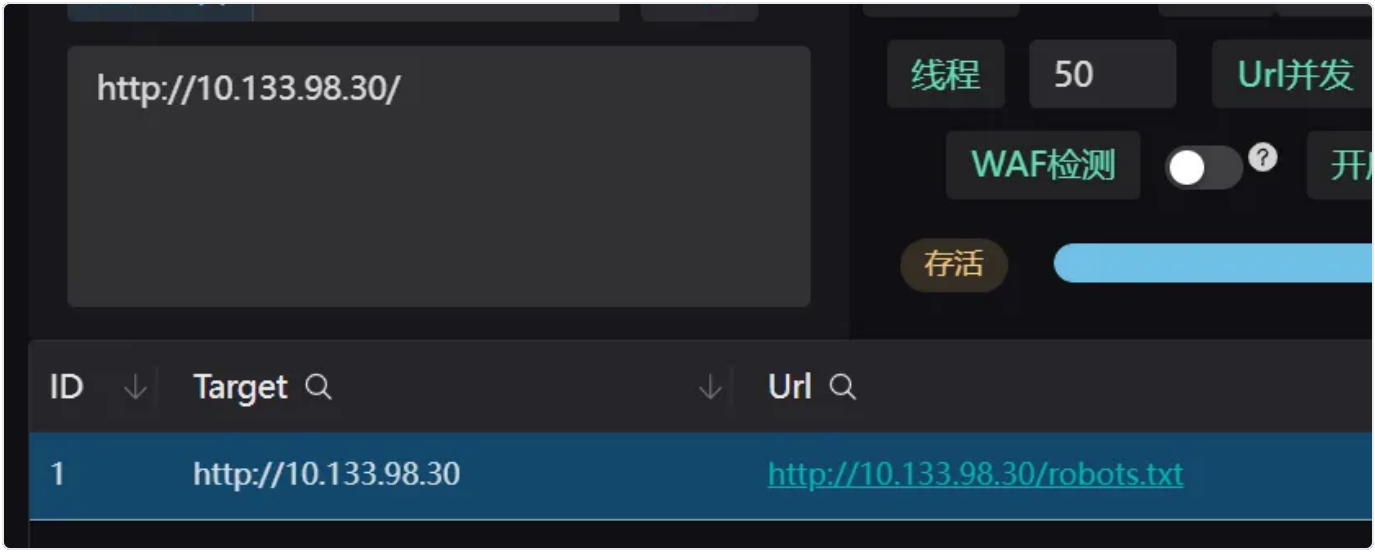
端口扫描


10.133.98.30
ID
Host
Port
Prot
1
10.133.98.30
22
2
10.133.98.30
143
3
10.133.98.30
80

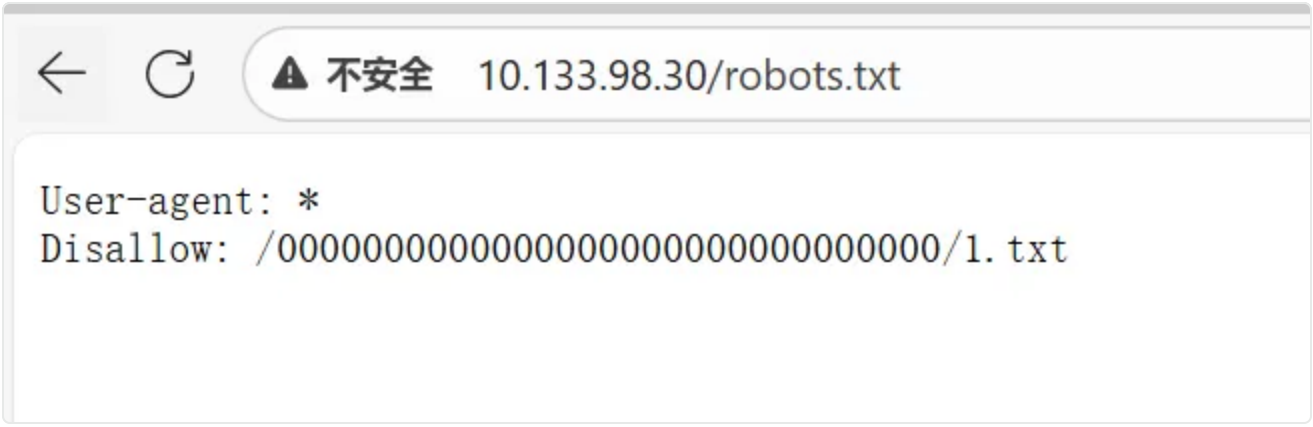
访问 80


不安全 10.133.98.30
echo "Enjoy the game !!!"
c:
n:

目录扫描



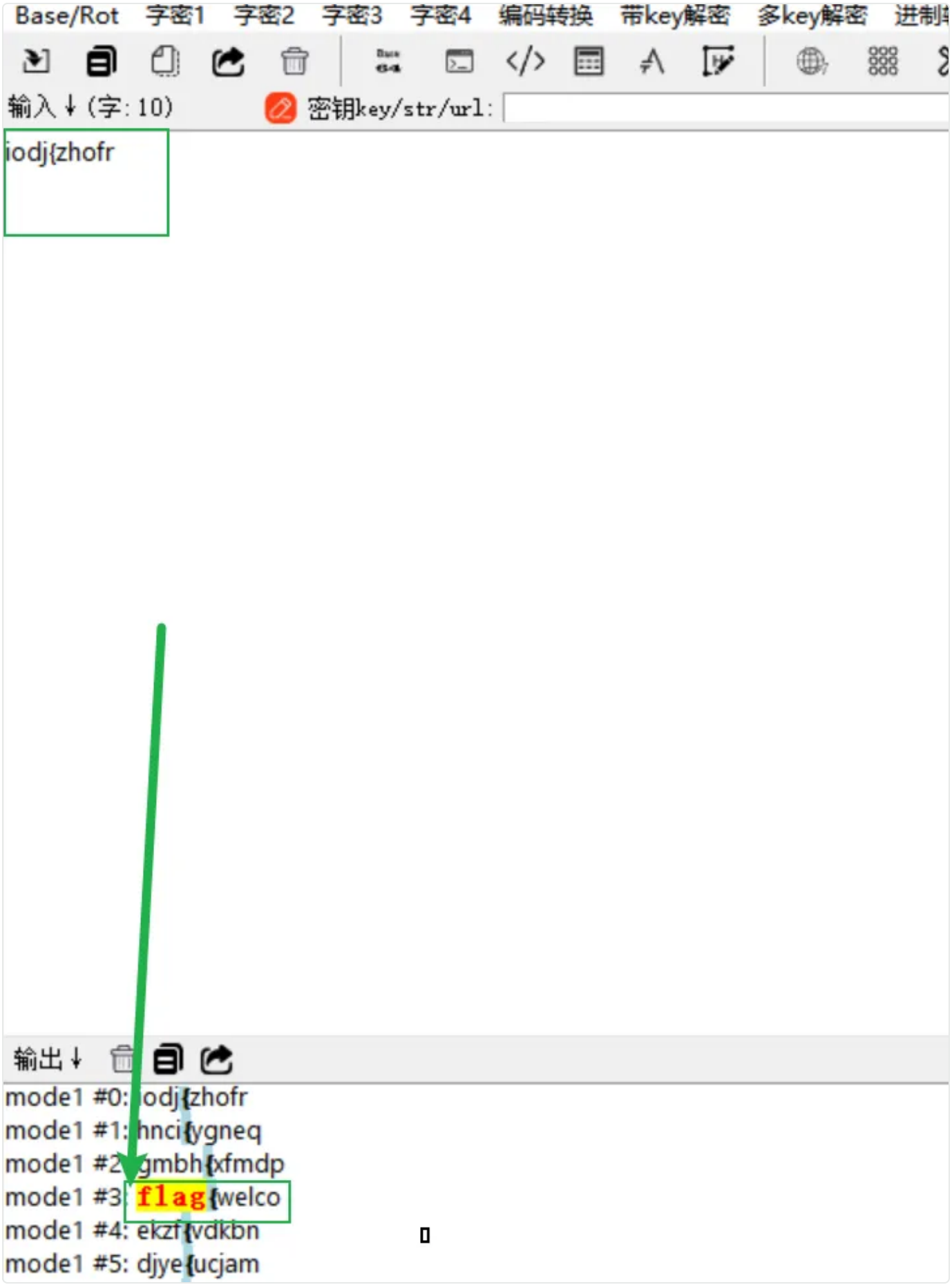
访问 ip/robots.txt



访问 /00000000000000000000000000000000/1.txt

base64 解密

像 flag 的一部分，应该是有偏移，上工具看一下

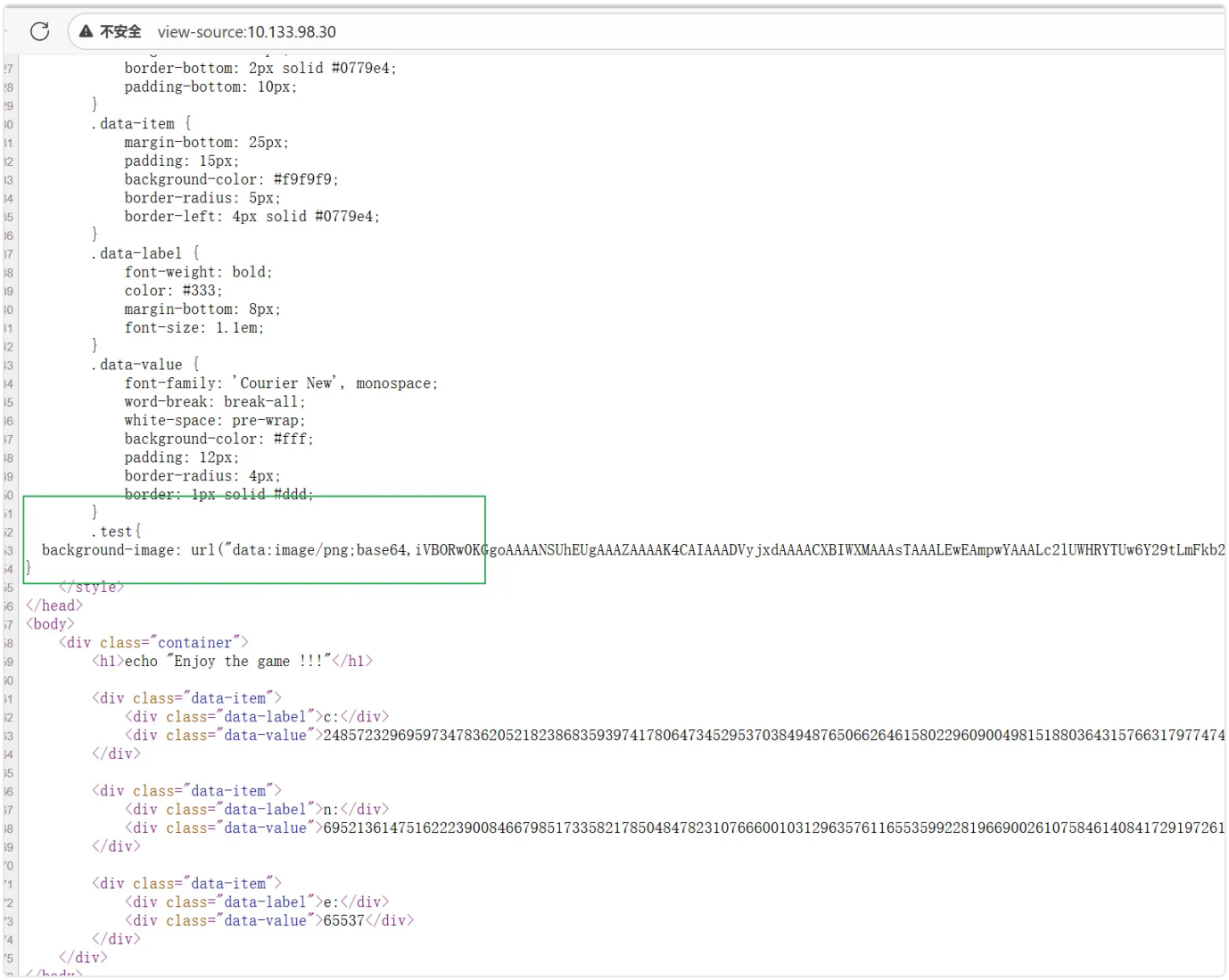

Base/Rot 字密1 字密2 字密3 字密4 编码转换 带key解密 多key解密
输入 (字: 10)
密钥key/str/url:
iodj{zhofr
输出
mode1 #0: iodj{zhofr
mode1 #1: hnci{ygneq
mode1 #2: gmbh{xfmdp
mode1 #3: flag{welco
mode1 #4: ekzf{vdkbn
mode1 #5: djye{ucjam

得到道一个看着正常的： flag{welco

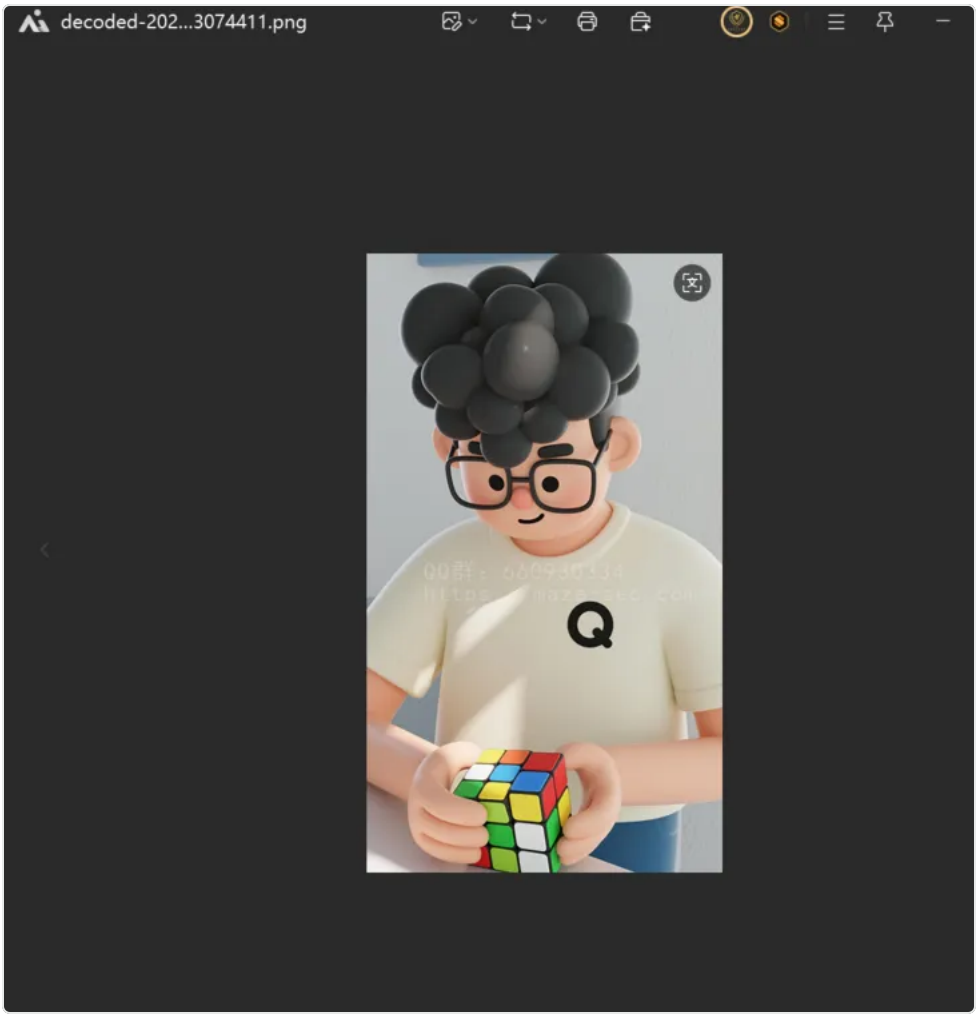
返回 首页，页面给出的是 c,n,e

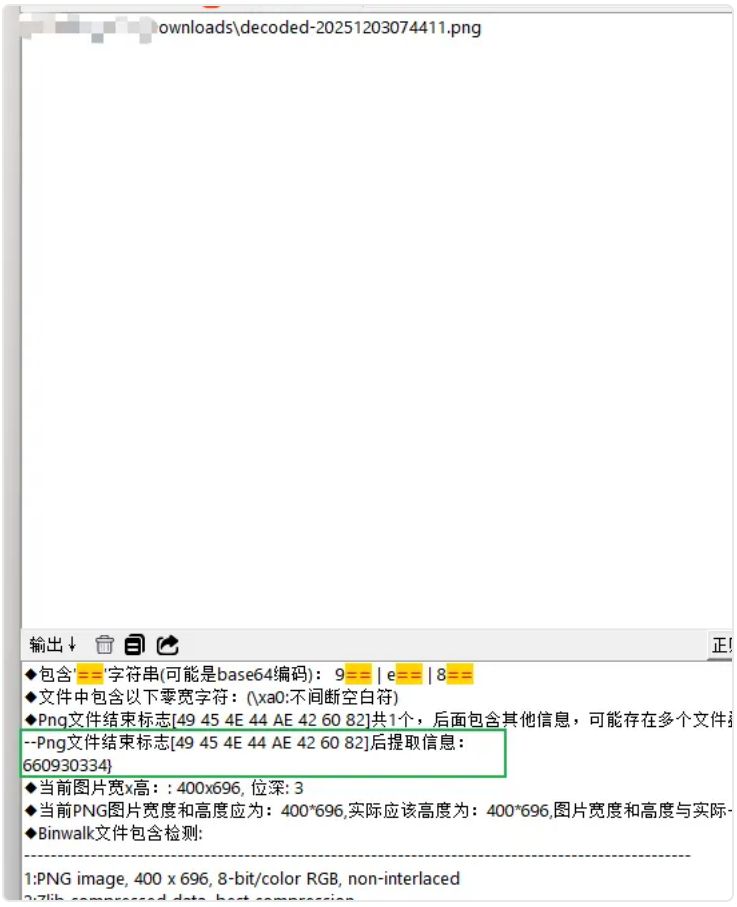
甩 ai 写个脚本或者往上找一个跑一下，得到 me:wlc0mE@

【我的进度到这了，其它找不到了，后来看群友发的 wp 看到是还有一段 是在 图片里，但是页面上并没有=-=】



- 用在线工具（如 Base64 解码）解码 Base64 字符串（去掉 `` 前缀）；
- 解码后下载「二进制文件」，重命名为 `.png` 即可。


decoded-202...3074411.png



`Png文件结束标志[49 45 4E 44 AE 42 60 82]后包含其他信息`，提取到的片段为 `660930334}`。

🆗拼起来 flag{welcome:wlc0mE@660930334}

```bash
ssh welcome@10.133.98.30
```

然后输入密码 `wlc0mE@660930334`

```php
┌──(root㉿kali)-[/home/kali]
└─# ssh welcome@10.133.98.30
welcome@10.133.98.30's password:
==============================
Welcome!!!
QQ Group:660930334
==============================
lingdong:~$ ls
tip.txt       user.txt      wechat_files
lingdong:~$ cat user.txt
flag{user-415621D5297F8F4BE138A5BB03}lingdong:~$
```

将文件打包拿出来

scp welcome@10.133.98.30:~/wechat_files.tar.gz .

用 ai 或者网上找一个微信PC端数据库文件解密脚本

```python
from Crypto.Cipher import AES
import hashlib
import hmac
import ctypes
import sys

# 微信数据库加密固定参数（无需修改）
SQLITE_FILE_HEADER = bytes('SQLite format 3', encoding='ASCII') + b'\x00'  # 修复：补全SQLite文件头
IV_SIZE = 16
HMAC_SHA1_SIZE = 20
KEY_SIZE = 32
DEFAULT_PAGESIZE = 4096
DEFAULT_ITER = 64000


def decrypt_wechat_db():
    """
    微信数据库解密脚本（修复语法错误+优化交互）
    支持输入密钥和文件路径，自动处理解密并覆盖原文件（或可修改为输出新文件）
    """
    try:
        # 1. 交互式输入参数（无需手动改代码）
        input_pass = input('请输入十六进制密钥（不含空格/0x前缀）：').strip()
        input_dir = input('请输入数据库文件路径（如 ./MSG0.db）：').strip()

        # 2. 处理密钥（去除空格，转换为bytes）
        try:
            password = bytes.fromhex(input_pass.replace(' ', ''))
        except ValueError:
            print('❌ 密钥格式错误！请输入纯十六进制字符串（0-9、a-f/A-F）')
            return

        # 3. 读取加密数据库文件
        try:
            with open(input_dir, 'rb') as f:
                blist = f.read()
        except FileNotFoundError:
            print(f'❌ 未找到文件：{input_dir}（请检查路径是否正确）')
            return
        except PermissionError:
            print(f'❌ 无权限读取文件：{input_dir}（请以管理员身份运行）')
            return

        print(f'✅ 成功读取文件，文件大小：{len(blist)} 字节')

        # 4. 提取盐值并生成AES密钥（微信标准PBKDF2算法）
```

```
        salt = blist[:16]
        key = hashlib.pbkdf2_hmac(
            'sha1',
            password,
            salt,
            DEFAULT_ITER,
            KEY_SIZE
        )

        # 5. 校验密码（通过第一页数据的HMAC校验）
        first_page = blist[16:DEFAULT_PAGESIZE]  # 第一页数据（跳过前16字节盐值）
        if len(first_page) < 48:
            print('❌ 文件损坏！第一页数据不完整')
            return

        # 生成MAC校验密钥
        mac_salt = bytes([x ^ 58 for x in salt])
        mac_key = hashlib.pbkdf2_hmac('sha1', key, mac_salt, 2, KEY_SIZE)

        # 计算HMAC并校验
        hash_mac = hmac.new(mac_key, digestmod='sha1')
        hash_mac.update(first_page[:-32])  # 第一页除末尾32字节外的内容
        hash_mac.update(bytes(ctypes.c_int(1)))  # 页数标识（第一页为1）
        if hash_mac.digest() != first_page[-32:-12]:  # 末尾32-12=20字节为HMAC-SHA1值
            print('❌ 密码错误！HMAC校验失败')
            return
        print('✅ 密码校验通过，开始解密...')

        # 6. 分割剩余数据为分页（按SQLite页大小4096分割）
        remaining_pages = [
            blist[i:i + DEFAULT_PAGESIZE]
            for i in range(DEFAULT_PAGESIZE, len(blist), DEFAULT_PAGESIZE)
        ]

        # 7. 解密并写入文件（覆盖原文件，如需保留原文件可修改输出路径）
        with open(input_dir, 'wb') as f:
            # 写入SQLite标准文件头（解密后数据库需识别为SQLite格式）
            f.write(SQLITE_FILE_HEADER)

            # 解密第一页数据
            iv = first_page[-48:-32]  # 第一页末尾48-32=16字节为IV
            cipher = AES.new(key, AES.MODE_CBC, iv)
            decrypted_first = cipher.decrypt(first_page[:-48])  # 第一页除末尾48字节外的内容
            f.write(decrypted_first)
```

```python
                f.write(first_page[-48:])  # 保留IV和HMAC数据（不影响数据库读取）

                # 解密剩余分页
                for page in remaining_pages:
                    if len(page) < 48:
                        print(f'⚠️ 跳过不完整分页（大小：{len(page)} 字节）')
                        continue
                    iv = page[-48:-32]
                    cipher = AES.new(key, AES.MODE_CBC, iv)
                    decrypted_page = cipher.decrypt(page[:-48])
                    f.write(decrypted_page)
                    f.write(page[-48:])

            print(f'✅ 解密成功！文件已保存为：{input_dir}')

        except Exception as e:
            print(f'❌ 解密失败：{str(e)}')
            return


if __name__ == "__main__":
    print('=' * 50)
    print('          微信数据库解密工具（修复版）')
    print('=' * 50)
    decrypt_wechat_db()
```

```
D:\Exclusion_items\密码\脚本\wechat_files>python wechatdb.py
==================================================
          微信数据库解密工具（修复版）
==================================================
请输入十六进制密钥（不含空格/0x前缀）：c22ce550443544439b22d75a1e1e4be286bc480cde0f34583bb490fe686b56061
请输入数据库文件路径（如 ./MSG0.db）：D:\Exclusion_items\密码\脚本\wechat_files\lingdong\msg\MSG0.db
✅ 成功读取文件，文件大小：52428800 字节
✅ 密码校验通过，开始解密...
✅ 解密成功！文件已保存为：D:\Exclusion_items\密码\脚本\wechat_files\lingdong\msg\MSG0.db

D:\Exclusion_items\密码\脚本\wechat_files>python wechatdb.py
```

使用脚本还原微信数据库 `MSG0.db` 的明文数据，然后用数据库工具打开

flag{root–463334051834284576670141O4}