## 第1届GeekGame参赛WriteUp

昵称：wwzzj（汪汪吱吱叽）

## 签到

**文件夹：signin**

这道题打开之后是一个pdf文档，下面的乱码应该就是flag的某种形式了，直接复制发现复制不全。

于是用Acrobat的编辑模式将内容复制了出来。

fa{aeAGetTm@ekaev!

lgHv\_\_ra\_ieGeGm\_1}

发现是竖着排列的flag，写了几行python调整格式，得到答案！

## 小北问答 Remake

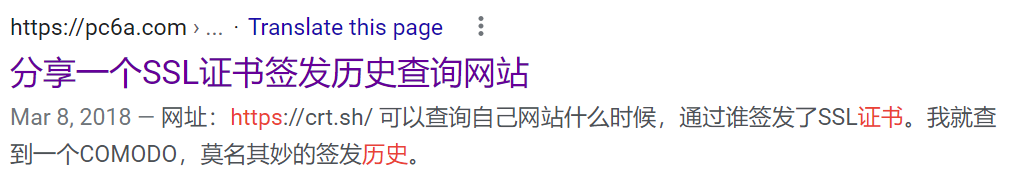
**文件夹：QA**

**第一题**：共有几号理科楼，凭借生活经验回答，5

**第二题**，上届比赛注册人数，从linux俱乐部的公众号上找了，407

**第三题**：geekgame网站上一次https证书过期，这题我找了好久，不停变换关键词搜索，希望找到一个有https证书记录的网站，结果最后还是在google上通过**中文**搜到的：

关键词：**https 证书 历史**



可能是我的英文搜索技巧不够地道，或者是外国人没有这种需求，怎么用英文怎么搜都搜不到。2021-07-11T08:49:53+08:00

**第四题**：DEFCON签到题的flag，在网站<https://scoreboard2020.oooverflow.io/#/>上，第一道题的flag竟然是点击就送。OOO{this\_is\_the\_welcome\_flag}

**第五题**：有**嘉心糖**在引流！，在google上一搜全是asoul的内容！最后没有找到现成的答案。

使用恰好有几对皇后会互相攻击推了一下**容斥**的式子，发现计算恰好有两对时化简比较麻烦，感觉写了个O(边长)的程序，结果怎么都不对。然后写了个暴力对拍，终于调对了。

第二阶段时被提醒可以用<https://www.wolframalpha.com/>直接算，感觉白白浪费了很多时间，亏豹！2933523260166137923998409309647057493882806525577536

**第六题**：看github上去年比赛的源码。Submits

**第七题**：这题我一直以为是AS24349 CERNET2 IX at Peking University，而且深信不疑，最后交了好多次才定位原来是这道题错了，重新搜，才发现原来还有AS59201 Peking University in China。

**第八题**：信息官网上2021年6月3日有个招生简章，上面列了所有的实验室。找到最长的就行。区域光纤通信网与新型光通信系统国家重点实验室

## 共享的机器

这道题我也花了很多时间，竟然放得这么靠前，做出来的人这么多，看了一晚上连猜带蒙才做出来。

之前没有接触过智能合约完全不知道要做什么，查了发现智能合约上可以放代码，一开始以为要交互，按照说明配置**Remix**和**MetaMask**，但一直跑不起来。

然后去看反编译的代码，**Etherscan**和**Ethervm.io**上的竟然还不一样，不过看起来Ethervm.io要专业一些，看来代码大概知道是位运算操作，看了看交易记录发现只有一条**Create**，还以为管理员没有设置参数，所以两个参数都是默认的**0**，直接逆着位运算算了一下，结果怎么都不对。

尝试了好久在网上查到一个Writeup，竟然还能在Etherscan上通过**parity trace**查看交易的输入和输出，然后才发现还有一条Internal交易，感觉输入应该是设置两个参数的，但是位数不对，猜他的编码方式是前面几位是函数名，后面两个256位的整数，尝试之后竟然真的解出了flag。

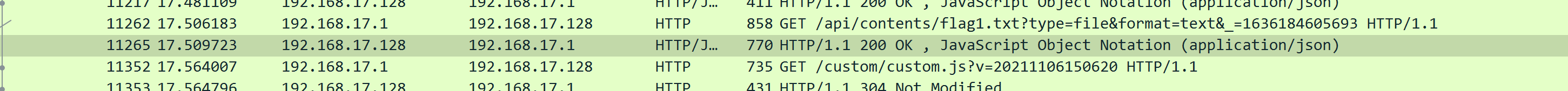
## 翻车的谜语人

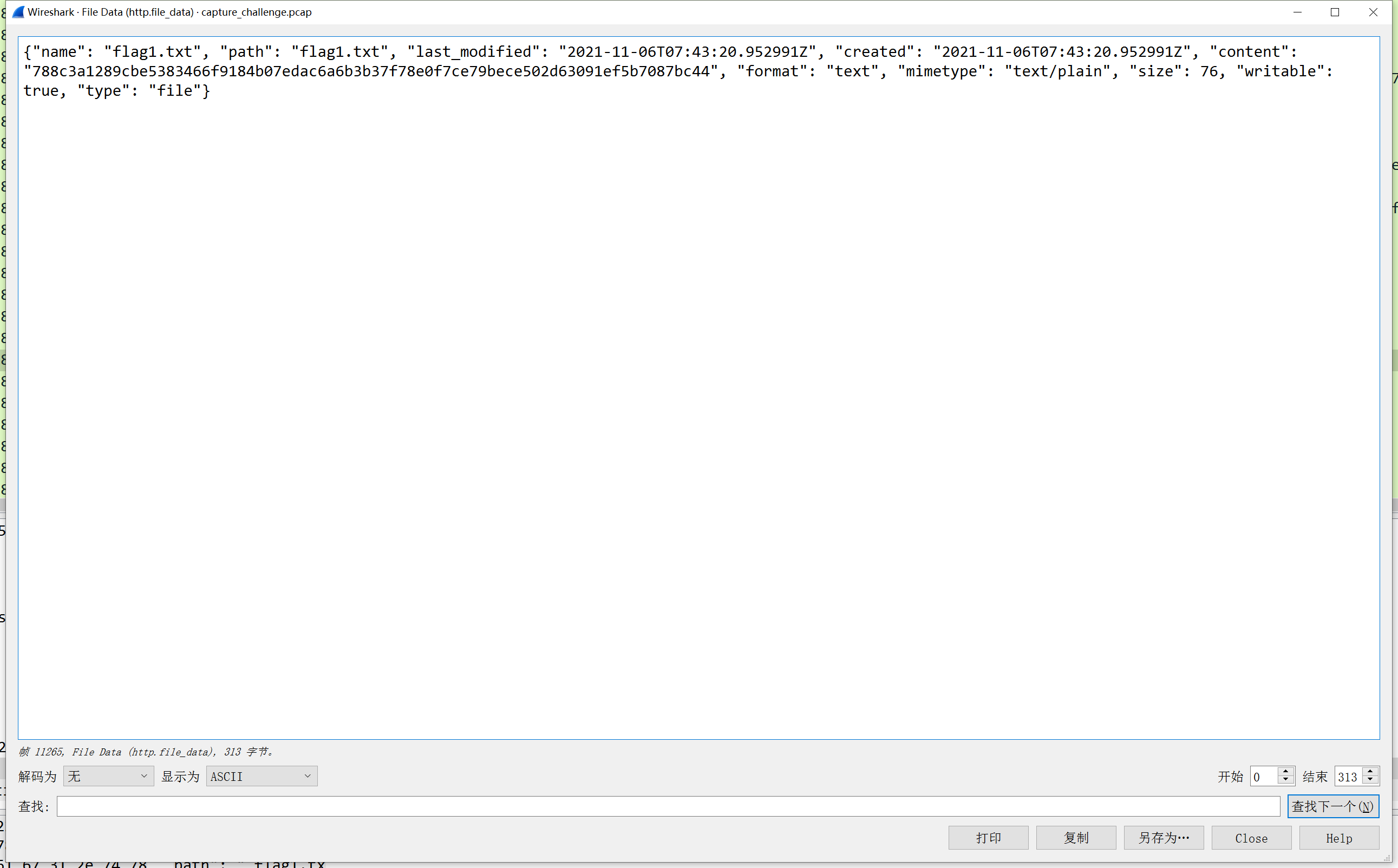
**文件夹：car**

做这道题也是我第一次用WireShark，一开始看里面的流量记录出来HTTP协议的其他都看不懂。

先看HTTP协议的流量，发现应该是和Jupiter Notebook交互的记录，从中可以提取出一段代码，在car/run.ipynb里，大概就是怎么对flag进行异或运算加密的。

然后发现了17s的时候GET了flag1的文件，随后该文件以json返回，进行解密运算之后得到了flag1.





之后想得到flag2，发现flag2最后被以一个7z的方式从HTTP中返回了，我下载了这个7z，发现要密码。

在21s~70s中是没有HTTP流量的，所以应该是别的流量。

我发现大部分协议都是加密的，只有很少的一些web socket有一些Line-based text data，这是我唯一看得懂的了，仔细查看，发现像是命令行的指令和回应，于是从头到尾筛查来看，发现这些指令做了一些事：

1. 安装了一个python lsp隐写库。
2. 使用这个隐写库将flag2.txt写到一个wav里，flag长76bytes
3. 将这个wav打包，密码为：Wakarimasu! `date` `uname -nom` `nproc`

所以要恢复这个密码，uname可以从之前的“mroot@you-kali-vm”找到，系统应该就是x86\_64 GNU/Linux吧，nproc看后面有cpu信息，但不知道这个虚拟机给了几核，可以枚举，date也是可以枚举的，但date格式有多种，我尝试了：

**Sat Nov 06 15:44:{} CST 2021**

**Nov 6, 2021 15:44:{} 中国标准时间**

枚举了很久，最后还是没试出来，我一度以为nproc或是uname是其他情况。

结果，第二阶段公布提示后，才知道竟然是**12小时制**的，最后枚举得出答案为：

“**Wakarimasu! Sat 06 Nov 2021 03:44:15 PM CST you-kali-vm x86\_64 GNU/Linux 8**”

## 叶子的新歌

**文件夹：music**

这道题高度好评，花样非常多，而且还有剧情！

由于受到CTF-wiki的启发，以为三个flag都是音频隐写，感觉不太熟悉，太难了，之前就没做这题。第二阶段才知道metadata中就有线索，才后悔莫及。

Metadata中有aHR0cDovL2xhYi5tYXh4c29mdC5uZXQvY3RmL2xlZ2FjeS50Ynoy，base64解码得到网站：<http://lab.maxxsoft.net/ctf/legacy.tbz2>

下载得到一个tbz2压缩文件，一个txt提示另一个img是软盘镜像，我直接按照网上的方法改名为bin打开了

然后是一个MEMORY.zip的压缩文件，和一个密码提示“宾驭令诠怀驭榕喆艺艺宾庚艺怀喆晾令喆晾怀”，这个密码从网上可以找到解码规则，大概就是在数字和汉字之间做了一个映射。程序在music/run.ipynb。密码为：72364209117514983984

打开之后是一个left.bin和一个right.bin，还有提示：“找不同”，“NES”，用010 Editor打开两个bin发现只有一点点不同，写了个程序将不同的字符提出(run.ipynb)。发现文件开头是NES，我下载了一个超级玛丽的镜像，开头也是NES，感觉这就是NES镜像了。

但放到Fceux模拟器里打不开，应该是找不同处理错了，一开始匹配时我只想前看来一位，结果出现了很多长度为2的不同字符，之后我改为多看几位，每次不同的长度就是1了，现在就能打开了。

没想到竟然也是一个超级玛丽的游戏，**一开始我还跟随题目沉浸在怀旧的氛围中，决定手动通关，只开一个无线生命的外挂，**但我实在是太菜了，恼羞成怒之后开了一大堆秘籍，跳关到8完成了游戏。

然后屏幕上出现了一个网址，这个网站需要填入一个密码，我试了之前的汉字密码发现不对，试了metadata也不对。

最后我觉得试着按照提示的“虚拟化工具”正式地打开一下软盘，用VirtualBox加载之后果然出现了密码，得到了flag3。同时软盘直接给出了flag2。

其实很很想多试试解出flag1，感觉flag1应该是真正和音频有关的，但是研究生狗事情太多，没有时间了。

## 在线解压网站

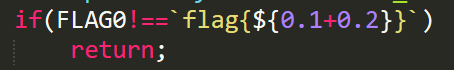
**文件夹：unzip**

发现服务器代码直接做的解压，没有什么安全措施。我从网上看了怎么把一个软连接压缩起来，好像用—symblink选项就行，然后压了一个指向根目录的软链接，看到了flag。

然后又压了指向flag的软连接，得到了flag。

## Flag即服务

想通过../看看能不能拿到别的文件，发现能拿到package.json。在<https://prob11-flrde9t3.geekgame.pku.edu.cn/api/..%2fpackage.json>拿到了package.json，里面有后端源码的链接，然后index.js里有关于flag1的信息：



直接在浏览器console里对`flag{${0.1+0.2}}`进行求值，得到flag1。

## 最强大脑

看了题学习了一下brainfuck，先试了一个网上的hello world，运行正常。

我又试了一下走一步输出一下当前位置，竟然直接得到了一个flag！

## 密码学实践

**文件夹：passwd**

看了源码，发现第一个flag和一句话都是用的PublicKey做的加密，查看加密操作只在4x8个bits内进行，主要是异或，模拟了一下，发现最后等价于每个数异或上一堆Key，数的主体是c, d, a^c, b^d。

这样的话，完全可以通过明文和密文推断出异或那的一团东西，然后进行逆运算，就可以解密了，顺利解出flag1。

第二问主要是通过模数的长度不够，让我们可以构造与目标内容同余的数，然后利用长度相减得到负数，利用python的负下标特性，使得最终dec得到的name为Alice。

同时可以让info和P同余，这样最终用来加密的NewKey还是PublicKey。