→签到←

将文件下载下来，然后用浏览器打开（Firefox、Edge均可），全选复制，然后粘贴到记事本：

```
fa{aeAGetTm@ekaev!
lgHv__ra_ieGeGm_1}
```

每行有18个字符，用Python运行`''.join('fa{aeAGetTm@ekaev!'[i]+'lgHv__ra_ieGeGm_1}'[i] for i in range(18))`，即可得到结果`'flag{Have_A_Great_Time@GeekGame_v1!}'`

小北问答 Remake

1. 直接看北大地图即可。
2. https://news.pku.edu.cn/xwzh/203d197d93c245a1aec23626bb43d464.htm
3. https://crt.sh/?id=4362000550 （关于此网站，可搜索“证书透明度日志”）
   注意：如果把证书下载到本地打开，里面给出的过期时间是“<U+200E>2021<U+200E>年<U+200E>7<U+200E>月<U+200E>11<U+200E>日 8:49:53”，含有不可见字符“<U+200E>”。如果修改的时候没删干净是无法通过的。可以去 https://regex101.com/ 检查是否存在不可见字符。
4. https://scoreboard2020.oooverflow.io/challenge_files/8a40c3a2ec3ef13216c5b3fd45b6631f1ad5b42497eda1ba618518f80f46712d/welcome.txt
5. 先找到 https://oeis.org/A047659 ，然后可以看到大小为 n * m 的棋盘上放 3 枚皇后的方法数是`n^3/6*(m^3 - 3*m^2 + 2*m) - n^2/2*(3*m^3 - 9*m^2 + 6*m) + n/6*(2*m^4 + 20*m^3 - 77*m^2 + 58*m) - 1/24*(39*m^4 - 82*m^3 - 36*m^2 + 88*m) + 1/16*(2*m - 4*n + 1)*(1 + (-1)^(m+1)) + 1/2*(1 + abs(n - 2*m + 3) - abs(n - 2*m + 4))*(1/24*((n - 2*m + 11)^4 - 42*(n - 2*m + 11)^3 + 656*(n - 2*m + 11)^2 - 4518*(n - 2*m + 11) + 11583) - 1/16*(4*m - 2*n - 1)*(1 + (-1)^(n+1)))`。将该公式复制到Mathematica中（abs()需改成Abs[]）即可算出结果。
6. https://github.com/PKU-GeekGame/geekgame-0th/blob/main/src/choice/game/db.py
7. https://bgp.he.net/search?search%5Bsearch%5D=Peking&commit=Search
8. https://ele.pku.edu.cn/

翻车的谜语人

用Wireshark的“导出对象”功能可以导出flag1.txt、flag2.7z、Untitled.ipynb文件。flag1可以直接从导出的文件算出。

flag2.7z是加密的，通过查找分组字节流可以看到7za的输出是通过WebSocket传输。然后运行`tshark -r capture_challenge.pcap -Y "websocket and ip.src==192.168.17.128 and ip.dst==192.168.17.1" -T fields -e text`，可以得到加密的密钥为“Wakarimasu! `date` `uname -nom` `nproc`”。根据7-Zip的输出可以知道nproc为8，uname -m为x86_64；根据提示符可以知道uname -n为you-kali-vm。然后在分组字节流搜索“2021”，可以知道70.247582s时时间为Sat, 06 Nov 2021 07:44:18，从而可以反推出运行7za的时间（可能有1-2秒的误差，需要枚举）。第二阶段的提示给了日期格式（给出提示前连时区都不知道，只能猜），但是我在WSL的ubuntu、kali虚拟机和“早期人类的聊天室”一题的机器上运行的结果日期格式不同，其中kali虚拟机上格式是`Sat Nov  6 11:45:14 PM CST 2021`，WSL的ubuntu和“早期人类的聊天室”一题的机器上运行的格式是`Sat Nov 6 23:45:14 CST 2021`（ubuntu和kali虚拟机可以用faketime修改时间，结果Nov和6之间有2个空格。）

叶子的新歌

第一部分

先用http://exif.regex.info/exif.cgi 查询mp3的元信息，可以发现里面有一个Picture（binwalk也能看到），用十六进制编辑器提取出来，再用stegsolve打开，可以发现lsb隐写了另一个png。这个PNG可以用https://www.onlinebarcodereader.com/ 识别，里面的文本为“Gur frperg va uvfgbtenz.”，ROT13后为“The secret in histogram.”
接着用numpy+PIL（或Mathematica）统计各个像素值出现次数，生成另一张图片（我生成的尺寸为256x60），其中某个像素值出现了该列涂黑，用https://online-barcode-reader.inliteresearch.com/ 识别（onlinebarcodereader和zxing都无法识别），可以识别出xmcp.ltd/KCwBa
该链接里面有一些Ook代码，用https://www.dcode.fr/ook-language 解码，可以得到flag{y0u_h4ve_f0rgott3n_7oo_much}。

第二部分

上文mp3的元信息中TRACKTOTAL用base64解码，得到http://lab.maxxsoft.net/ctf/legacy.tbz2
该文件可以解压出To_the_past.img，挂载到Linux上，得到文件MEMORY.ZIP和NOTE.TXT，其中NOTE.TXT有“宾驭令诠怀驭榕喆艺艺宾庚艺怀喆晾令喆晾怀”这些文字，可以直接用百度搜索。
MEMORY.ZIP里面有left.bin、readme.txt、right.bin三个文件，（一开始找不到任何软件打开两个bin文件），用Python把left.bin和right.bin重复的字节删除（注意lookahead两个字节，只lookahead一个字节会出错），将不重复的字节保存到另一个文件，可以得到一个nes文件。
这个nes文件可以用模拟器打开，但是我不知道怎么提取里面的flag（试过各类编辑工具和反编译器均无效）。

第三部分

不知道第三个flag也在nes文件里还是隐写在mp3的内容里（也不知道用什么软件处理）。

在线解压网站

先任意创建一个往/flag的符号链接，然后用--symlinks参数打包（打包后文件的Base64：UEsDBAoAAAAAAKO5bVNOewN1BQAAAAUAAAAHABwAc3ltbGlua1VUCQAD0oyQYeOMkGF1eAsAAQToAwAABOgDAAAvZmxhZ1BLAQIeAwoAAAAAAKO5bVNOewN1BQAAAAUAAAAHABgAAAAAAAAAAAD/oQAAAABzeW1saW5rVVQFAAPSjJBhdXgLAAEE6AMAAAToAwAAUEsFBgAAAAABAAEATQAAAEYAAAAAAA==）。上传到服务器即可拿到flag。

早期人类的聊天室

查看src可以发现chatlog接口可以访问磁盘的任何文件（例如请求../../../../etc/hosts可以访问/etc/hosts）。但是不能访问/flag（即使通过/proc/PID/root/flag也不能访问）。
接着通过访问/proc/PID/cmdline（PID需要枚举）可以找到所有运行的程序，从命令行参数可以得到这些程序的配置文件。可以得到supervisord和uwsgi的日志和uwsgi的端口（3031）。
通过搜索可以得到uWSGI运行任意代码的exp（和提示中相同），然后（需要先将exp改成base64输出）尝试运行一些代码，通过uwsgi的日志和将结果重定向到文件可以发现这些代码确实被执行了。但是仍然不能输出flag（例如，把flag复制到其他目录或者建立往flag的符号链接再访问均失败，同时crontab也不能写）。运行whoami可以发现当前用户为nobody，通过ls可以发现flag权限为600因此不能访问。
然后看看有什么文件可写，可以发现uwsgi的配置可写，用sed将里面uid和gid（指key，非value）替换成任意字符串，接着用kill -9杀掉所有uwsgi进程（supervisord会自动重启uwsgi；另外必须带-9否则无法完全杀掉uwsgi进程），可以发现当前用户变成root并且可以访问到flag了。

Flag即服务

使用Python的socket库就可以看到package.json。里面有源代码的地址。可以把`flag{${0.1+0.2}}`复制到浏览器控制台查看flag。

```
import socket, ssl

context = ssl.SSLContext(ssl.PROTOCOL_TLSv1_2)
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s_sock = context.wrap_socket(s, server_hostname="prob11-
********.geekgame.pku.edu.cn")
s_sock.connect(("115.27.243.170", 443))
s_sock.send("GET /api/../package.json HTTP/1.1\r\nHost: prob11-
********.geekgame.pku.edu.cn\r\n\r\n".encode())

print(s_sock.recv(2048))
```

PS：看到第二阶段提示前我还以为要在靶机上找源代码。

第二问不会做。第三问因为flag文件从未关闭，从/proc/PID/fd应该能拿到flag文件。因为只有in_path支持eval，需要逐个bit爆破flag的内容。另外有a-zA-Z"',;的字符串不会被eval，但是把代码转成JSFuck就不会有这些字符了。

诡异的网关

静态反编译工具是没用的。正确的方法是运行程序后用Cheat Engine等工具搜索程序内存。

最强大脑

Flag 1保存在内存里，所以用+[-.>+]（2B 5B 2D 2E 3E 2B 5D）遍历内存空间就能拿到flag。

电子游戏概论

曾经尝试了大量错误的做法：

- 以为是单机游戏，尝试用Cheat Engine改内存
- 以为是pyinstaller（用文本编辑器打开exe能看到Python代码），用解包工具
- 用Fiddler抓包（实际上程序没有用HTTPS，而是自己实现了加密的socket）

正确的提取代码的方法：

1. 用可以查看PE文件结构的工具打开，可以发现里面有名为PYTHONSCRIPT的resource
2. 将resource导出，然后删除前17个字节（注意不是前16个），得到的结果用marshal加载
3. 可以看到一些code对象，取最后一个，参考这里的代码，生成pyc文件
4. 用decompyle3反编译该pyc文件
   代码用到了一个securesocket库，把exe文件扩展名改成zip解压就能找到securesocket.pyc文件，同样用decompyle3反编译。
   两个文件反编译结果有部分的缩进是错的，需要修正。
   然后可以修改源代码，其中dirt和evildirt的图片是相同的，可以将evildirt的图片换成空白。同时在命令行输出发送和接收的内容；修改Canvas尺寸使整个游戏画面都显示到窗口（此时可以注释掉canvas.xview_moveto和canvas.yview_moveto）。附件的py3.png给出了修改后的游戏界面。

以上部分均在第二阶段开始前已完成。
这样游戏可以手玩了，有几个同学（应该是通过第二阶段给的pyw文件）做出了Flag 1。但是Flag 2还需要做一个自动的游戏AI，我已经写了一部分但是仍然有bug没有达到可用的程度。最后没做出来。

密码学实践

Flag 1

根据给的代码可以发现MESenc的结果按照8个byte分段，每段都是原文和key的某个xor组合。简单尝试可以发现：（这里的key、key1、key2均为任意的）

1. MESenc(MESenc(MESenc(x,key),key),key)=x
2. MESenc(x,key1) xor MESenc(x,key2)=MESenc(y,key1) xor MESenc(y,key2)
   因此用以下Python代码可以获得Flag 1：（xxx、yyy为Talk to Richard的输出结果）
   ```
   MESenc(MESenc(bytes([MESenc(pad(("Sorry, I forget to verify your identity. Please give me your certificate.").encode("utf-8")),b'\0'\*256)[i] ^ bytes.fromhex("xxx")[i] ^ bytes.fromhex("yyy")[i] for i in range(96)]),b'\0'\*256),b'\0'\*256)
   ```

Flag 2

这里给定了一个工具，可以构造出<A><len(A)><B><len(B)>的RSA加密结果（A和B可以任选，len长度为2字节），要求构造一个加密结果，使A为Alice，而使用该工具时A不能为Alice或Richard。
可以先假设B为空，那么要构造0x416C6963650005的d次方（d未知），而0x416C6963650005=0X82D8D2C6CA000A/2，只要计算出0X82D8D2C6CA000A和2的d次方即可。流程如下（需使用Python 3.8+，旧版本不支持exp为负数）：

1. 选择Talk to God，name=000000000082d8d2c6ca，key为空，得到的certificate设为A。
2. 选择Talk to God，name为空，key=0000，得到的certificate设为B。
3. 计算`(pow(B,-1,N)*A)%N`；选择Talk to Richard，输入计算的结果。

最后会给出另一段密文MESenc加密结果，将该密文分成3个32字节的部分，和上面类似（第一部分的明文已知）可以解密。

扫雷

（请先阅读这篇Writeup再阅读此段）
通过扫雷的界面内容每次可以得出MT19937的8个块（以下8个块称为一段），而每个块的值只和之前第624、623、227个块有关。因此只要78次gen_board就能把MT19937的所有块确定下来。
简单模式下每段有1/2概率被删除，因此第i-624、i-623、i-227个块（此处使用Python表述，第-x个块就是倒数第x个块；0<=i<8）若未被删除平均会被移动到第i-312、i-311、i-115个块。后面的预测流程类似，但是每次预测不一定成功，需要多试几次。
ms1.py提供了完整的程序（需安装pwntools）。

龙珠模拟器

（请先阅读这篇Writeup再阅读此段）
以下不考虑生成的bits恰好在某范围导致要重新生成的情况，也不考虑多种珠子重叠或者和已知的air重叠，因为出现的概率很小。

Flag 1

这里可以设chunkOffsetX和chunkOffsetY为0，此时7颗龙珠的位置分别为($a\%n_i$,$b\%n_i$)，这里n_i分别为24、48、192、1536、24576、393216、25165824，而a和b都是固定的随机数（因为随机数种子相同）。这些n_i每个都是前一个倍数，因此可以逐个搜索，例如知道$a\%393216$后$a\%25165824$只有64种可能，只要搜索64*64次就能确定$a\%25165824$和$b\%25165824$。注意请求次数可能很多，需要异步网

络请求。
lz1.py提供了部分函数。

Flag 2

因为0x2e01和0x76b5互质，而chunkOffsetX和chunkOffsetY可以自行调整，如此可以对任意的n获得种子为n+0x15beccc9*baseSeed的随机数模29的值（其中n=0x2e01 * chunkOffsetX + 0x76b5 * chunkOffsetY + 0x7296ea13）。
设N=n+0x15beccc9*baseSeed，查看随机数发生器的源代码可以发现获取的是((((N ^ 0x5deece66d) & ((1 << 48) - 1) * 0x5deece66d + 0xb) & ((1 << 48) - 1)) >> 17) % 29。由于0xb远小于131072，可以忽略0xb这一项，因此获取的数变成(((((N ^ 25214903917) & 281474976710655) * 25214903917) & 281474976710655) >> 17) % 29。显然baseSeed只有后48位有意义。
考虑把两个数N1、N2，其中N1二进制第i位为0，N2二进制第i位为1，其余位相同，那么(N2 ^ 0x5deece66d)-(N1 ^ 0x5deece66d)为1 << i或-1 << i，且固定i取不同的N1和N2该数是相等的。((N ^ 25214903917) & 281474976710655) * 25214903917的差也和N1和N2的选取无关。(((N ^ 25214903917) & 281474976710655) * 25214903917) & 281474976710655的差根据乘25214903917的进位数量有2种不同可能（两个数的差固定为a，之间的b的倍数可能有int(a/b)个或者int(a/b)+1个，同理除以b的余数的差也有2种不同可能），而>> 17之后有4种不同可能。
当N为奇数时，N+2*k永远是奇数，从而N+2*k和N-1+2*k在二进制中只有一位不同，以它们为种子的随机数差模29只有4种可能。而N为偶数时这两个数不同通常多于一位，随机数差模29多于4种可能。从而可以得到N的二进制的最低位。
已知N是偶数，如果不是4的倍数，N和N-1二进制后两位分别为10和01，和0x5deece66d异或后差是固定的。取k=N+4*i或k=N+4*i+2（i取0-99），考虑以k和k-1为种子的随机数差模29，如果N+4*i+2不是4的倍数只有4种不同的值，而N+4*i是4的倍数时通常有多于4种不同的值。从而可以调整n使N为4的倍数。
依此类推，可以使N二进制每一位为0，但是因为API请求的x和y必须是32位有符号整数，只能将N的后32位变成0。但是这样已经知道baseSeed的后32位，前16位只要枚举即可。以上必须用异步请求，同步请求很可能在时限内跑不完。
lz2.py提供了完整的程序（需先pip install java-random）。