→签到←

下载附件打开 pdf 文件，看见奇怪乱码，猜测修改了字体文件，全选复制，按行列依次取字符得到 flag.

小北问答 Remake

1. 常识
2. 模糊记得上一届结束后信科发过一次推送，里面包含这个答案。在 大信科 公众号搜索找到链接。
3. 检索发现存在网站记录这样的信息，结合续期时间~~干则表达式提示~~可以得到忘记续期的那一次。
4. 既然是签到题，那么应该不难？找到对应比赛的网址，看看签到题内容就行。
5. 容斥是不可能容斥的，经过在 OEIS 上的一阵搜索，在 A047659 的 FORMULA 里找到了计算公式，用 python 脚本（见 `exp-chessboard.py` ）计算
6. Github 上开源
7. 网上搜搜找到链接，发现有两个包含 `Peking University` 的记录，猜一个不包含 `CERNET` 的记录。
8. 大概搜出了这些实验室

```
软件工程研究室系统软件与中间件实验室
人机交互与多媒体研究室
计算机系统与理论研究室
信息安全研究室
北京大学贝尔实验室软件技术联合实验室
北大-Intel新技术实验室
北京大学微处理器研究开发中心
北大-Intel软硬件协同设计实验室
数字视频编码与系统技术国家重点工程实验室
电磁场与微波技术研究室电路与系统实验室
区域光纤通信网与新型光通信系统国家重点实验室
卫星通信研究室信号与信息处理实验室
纳米器件物理与化学教育部重点实验室
冷原子物理与精密测量研究室
量子光学与量子信息研究室
原子钟研究室
超快光学研究室
国家集成电路产教融合创新平台
微米纳米加工国家重点实验室
微电子器件与电路教育部重点实验室
集成电路科学与未来技术北京实验室
北京有源显示工程技术研究中心
视感知研究室
听感知研究室
计算智能与知识发现研究室
感知机理研究室
```

共享的机器

etherscan 上反编译出来的代码看不明白，换了一个其他的 网站，整理一下代码可得 flag.

见 `exp-eth.py`

翻车的谜语人

观察 Jupyter 的输出，发现随机生成了 flag1 和 flag2 后异或上了一个常量字符串。

flag1 直接保存在了文件，且调用了 Jupyter 查看文件的 api，可以直接拿到；

flag2 被 stego-lsb 隐写到了 wav 文件里，后用 `Wakarimasu! `date` `uname -nom` `nproc`` 加密生成压缩文件。

日期可根据 flag2.7z 创建日期得到；

执行的 7z 命令显示了 CPU 的基本信息，可知 nproc 为 8；

主机名随处可见，剩下两个直接猜是 `GNU/Linux` 和 `x86_64`；

日期的格式没找到，只能靠提示了。

解压出来后用 stego-lsb 提取，再异或回去得到 flag.

在线解压网站

容易想到把符号链接压缩到 zip 里。

早期人类的聊天室

题面强调 uwsgi，经搜索发现其存在远程代码执行漏洞。

写脚本扫描所有开放端口，发现 uwsgi 监听 3031 端口。

根据脚本 uwsgi_exp.py 生成请求。

利用代码执行漏洞遍历查看所有有价值的文件，发现 /tmp 目录可写，且恰好包含 uwsgi 的配置文件。

修改配置文件里 uwsgi 用户组，并 kill 掉 uwsgi，等待 supervisor 以 root 用户重启 uwsgi，便能拿到 flag.

Q小树洞的一大步

经过大量尝试，发现这个后端不像是有漏洞的迹象，思路转移到~~用过树洞网页版都知道的~~搜索框 setflag 上。

接下来的问题是 alert 会直接让 selenium 抛出异常，结束程序。

搜索后发现可以利用 iframe，如果在沙盒里不允许 allow-modals，则会禁止页面 alert.

因为 flag 写到了 cookie 里，跨域时不能携带，所以只能想办法使之执行注入的 js 代码。

发现存在 eval 语句，从 localStorage["APPSWITCHER_ITEMS"] 读值，而 setflag 正好能修改 localStorage.

不过 eval 语句执行在 setflag 之前，并且会在加载到真正的 APPSWITCHER_ITEMS 后覆盖。

于是可以想到开多个 iframe，利用时间差使得某一个 eval 能够执行到想要的代码。

测试发现 iframe 里无法读取 cookie，修改为将主页面跳转至 qkuhole.

成功率比较低，开个脚本多次提交，即能拿到 flag.

见 `exp-qkuhole.html`

Flag即服务

猜测未检查路径合法性，访问 `https://prob11.geekgame.pku.edu.cn/api/..%2fpackage.json` 拿到源码链接，可得 flag1.

诡异的网关

用任务管理器 dump 出内存信息，搜索 flag.

字符串转义

提供源代码的良心题。

看完题就能想出思路：

1. 让 from 指针跳到 to 指针之前，把位于 ctx 后面的金丝雀值、偏移常数后的栈指针地址以及返回地址往前挪，利用输出的 caseid 得到值
2. 构造 rop 链
3. 利用 to 指针，再把金丝雀值和 rop 链写回到正确的位置

不过构造 1 的字符串花了大量时间，后来索性写代码枚举~~最后构造出的解十分丑陋~~。

见 `exp-escape.py`

最强大脑

发现 flag1 存到了地址偏移 0x1000 的位置，让指针不停增加 1 输出就能得到 flag1.

电子游戏概论

修改反编译出来的代码，去掉所有 tick 请求，同时修改 evildirt 使之显示所有挖掉会出火的方块，手玩拿到 flag1.

密码学实践

在明文密文均已知的情况下，可以直接复原出 MESenc 的加密流程，可得 flag1 。

对于 flag2 ，发现在已知 A 、B 密文的情况下， A * B 的密文就可以确定了。

多次随机 5 位长度的 Alice 证书，将对应明文 C 进行分解，如果能拆成 C = A * B 的形式，
且 A 、 B 均为合法输入，就做完了。

考虑让 A 尽量小，其名称为空，证书为 A 位的 `00`；

然后直接解析 B ，检查是否合法。

见 `exp-crypto.py`

扫雷

接近一半的地雷显然是不能靠单纯的扫雷算法解决的，于是只能预测随机数。

网上搜一份通过连续 624 个 int 重建 mt19937 key 的代码，改改就能通过 hard mode ，拿到 flag1.

对于 easy mode ，程序拿到是随机数的一个子序列。

根据 mt19937 的特性，如果已知第 i 个 int ，第 i + 1 个 int，第 i + 397 个 int，则能预测第 i + 624 个 int ，可以枚举返回块三元组 (i, j, k)，若第 i 个块与第 j 个块能与第 k 个块信息对应，则能知道这些块之间真正的距离（即知道丢失了多少块）。

然后用类似于并查集的思路，把所有信息关联起来，再加上一点优化（如果返回的第 i 个块与第 i + k 个块真正的距离为 k，则可以确定第 i 个块到第 i + k 个块之间没有信息丢失），就~~可能~~可以重建出连续的 624 个 int 。

见 `exp-minesweeper.py`