目前的工作

- 尝试阅读理解英伟达的内核态驱动NVIDIA/open-gpu-kernel-modules: NVIDIA Linux open GPU kernel module source
- 大致理清了该内核态驱动的作用与项目中各个文件夹之间的关系
- 大致确定了较为重要的文件，大概确定了和GPU虚拟化相关的文件夹与文件（例如和内存分配相关，模块初始化相关的文件）
- 找到了编译出的nvidia.ko的入口函数与文件操作接口，下一步正在理清nvidia.ko初始化后驱动做的工作

思路

1. 用例如API HOOK的方式，在执行内存分配的时候先判断是否超出人为的限制，在执行任务时判断是否要切换上下文到另一个用户的任务；
2. 在一些用户态虚拟化论文中，会采用API拦截的方式，将执行流跳转到自己设置的一个vGPU manager，进行时间片划分与内存管理等
3. 直接修改NVIDIA的内核驱动源码，实现一个静态的GPU虚拟化（例如静态划分为4个vGPU）

难点与困惑

1. 在虚拟机中找不到GPU，无法编译驱动；又在WSL2中尝试，这次是因为WSL内核与Linux有不同，也无法编译成功；
2. 关于GPU上下文的代码与内容暂时没有头绪，也没有找到具体的代码文件，对于如何进行任务的切换还不清楚；
3. 在尝试理清一个GPU计算任务的执行流时，暂时没能在内核态驱动中发现与计算相关的代码，大多数都是管理硬件的代码，推测认为计算相关的代码实现应该在用户态的GPU库中（如CUDA和OpenGL）
4. 英伟达只开源了内核态源码，用户态的代码库都是闭源的，而这两者之间又有频繁的交互，仅研究内核态驱动也不能保证有效
5. 对于如何在Linux中替换和加载驱动有些疑惑
6. 内核态代码太过复杂，代码量也很多，并且涉及到很多方面，理解起来有些困难