7大题，23小题，2hour，100分

题干会有提示/答案，注意加粗字

线程、进程

T1~6, 27’

15：基础概念解释（结合Linux输出回答）

4：部分syscall的执行过程，系统的角色（信号绑定，syscall）

8：性能分析

- 线程过多?
- swap的开销?

*基础概念：线程、进程、协程、IPC、中断、syscall的ABI

优先级反转

T2, 7~8，6'

（只考虑一个线程只有一把锁的情况）

2：基础概念解释（结合例子）

4：如何避免优先级反转

内存管理

T3, 14'

5：基础概念解释（结合程序输出）

5：物理内存管理

4：KPTI、Meltdown

基本概念:

页表上的地址是什么?

页表基地址是什么？物理/虚地址

malloc了一个地址并打印出来，物理/虚拟地址?

MMU:

MAKE,SATP改基地址

物理内存管理:

内核能否感觉到物理地址的存在？e.g.buddy伙伴算法到底放在什么地方

内核拿到的是虚拟地址

如何实现高效页管理?

KPTI、Meltdown?

主要讲KPTI，把一张页表拆成两张。

文件系统

T5, 21'

默认linux上的ext处理

6：基础概念解释（ext系列的inode、superblock、journal（日志））

4：容错(主要和superblock有关，为什么要写多个superblock。e.g.makefs)

6：类FAT文件系统的分析/设计

5：Page Cache的设计

backup的作用是什么?

*软链接和硬链接的区别是什么？两者删文件为什么不一致?

为什么文件page要4k对齐？（与内存页表匹配？与硬盘匹配？）

a syscall: mmap

读写一个文件：利用mmap后，用char*数组读写，是怎么从文件系统中读出来的?

mmap为什么快?

mmap不会主动把文件全部读下来

类fat文件系统:

1. 有什么缺点
2. 如何支持可变长度读（写）（只考察read）

虚拟化与安全

T4, 12'

7：基础概念解释与运用

5：watchdog

3个涉及ept：扩展页表

Problem1：EPT or NestPT的作用?

加快虚拟机的内存访问速度。从硬件层面上解决同一个机器上跑多个虚拟机时的页表加速问题。加速shadow page的updating问题。

shadow page：guest os为自己的每个process准备一个页表

Problem 2：支持EPT的场景下，执行过程和没有ept有什么区别？性能差距有多大?

Problem 3：一种病毒有能力在真实机上修改CR3（x86下的页表基地址）（改映射表），使系统调用转换到它那里。有了硬件扩展（ept）后，在虚拟机内它能否发挥作用?

尝试病毒。为了保证他的宿主机不会受到影响，他选择将病毒放在虚拟机中执行。在虚拟机中，病毒对关键寄存器进行修改，如页表计时器等（CR3），并不会直接修改宿主机的对应寄存器。简要说明虚拟机如何达成这一权限控制。

虚拟机运行时，CR3不会再被干预。物理CR3=虚拟机CR3(拷贝)

逻辑上，每个虚拟机都有一个独立的CR3

Problem 4：由于时钟中断存在，同一操作系统下的不同进程不会相互干扰。CPU有个特殊的core，可以向所有的core发送NMI（不可屏蔽中断）。某病毒关闭一个核的所有中断使其死亡，怎么处理?

让那个特殊的core挂一个NMI给dead core，由于NMI不可屏蔽，NMI通过中断矢量表硬件，自动找到dead core上的os的handler，handler打开所有中断，从而让这个dead core被激活，进而自己解决其他问题。

其他hint

多个虚拟机，一个虚拟机死亡，所有中毒那都会被关掉；hypervision强制打开所有中断

fork分析

T2, 21~22，10'

10：fork分析

其他hint：

父子进程除了pid，全部一摸一样

fork后，两个进程一个读一个写，枚举所有可能?

四个基本调用?

综合设计

T1, 10'

A.虚拟化加速：VMEXIT、中断

B.NUMA场景下的同步原语：cache冲突如何避免

C.异构计算和可信执行环境：跨CPU和GPU的执行环境

评分标准：4+3+3

4.基本知识概念正确：简要复述题目内容

3.方案设计合理：合理即可，不一定实际实现

3.要求想办法测试是否解决了问题：逻辑问题/性能问题