现代操作系统 shell gui 内核态 完全硬件访问权，所有指令 用户态

500万行 windows unix linux

远程过程调用rpc 本地过程调用lpc 服务 子系统.net

控制芯片 来使cpu对于各个硬件端口进行访问 这些硬件操作都被抽象为文件读写函数

多程序多硬件同时运行，互相竞争，操作系统分配资源 cpu指令

复用计算机资源 轮流使用 分割内存，可以同时使用，多个程序都载入内存

分时系统 win98 16位 unix x window

CPU指令集 寄存器 程序计数器cs:ip，保存下一条指令内存地址 ss:sp堆栈指针（可以设任意内存为堆栈） 标志寄存 现代可以同时取多条指令，流水线， 可能有多个单元分别执行不同指令 cpu内核态 应用使用系统调用 使cpu进入内核

多线程

存储器 寄存器-高速缓存-内存-磁盘， 文件会被缓存在缓存中或者内存中 cmos保存时间等

磁盘 多个盘片高速旋转 多个磁头读取 磁道 柱面 在磁道间移动

程序间切换时要把内存内容写回磁盘 上下文切换

IO设备 设备控制器和设备，接受cpu命令

用户-系统调用-控制io设备 忙等待/ 中断 ，中断后当前程序地址 和寄存数据 入栈，完成另外活动后返回原命令 数据出栈回复/dma芯片，设备直接访问内存

总线 桥 pci总线 通用串行总线 usb ide磁盘 scsi ieee火线 中断级别 冲突

Bios基本输入输出系统 底层io 读键盘 写屏幕 磁盘io等 按cmos设备清单启动设备

系统 分类 服务器 嵌入式 智能卡 实时 有cpu

进程process 地址空间 存有可执行程序，数据，堆栈 以及ip ss 等信息 相关进程，文件 分时系统中进程不断启动、挂起 分配不同时间片 进程表（数组或链表 ） 保存进程地址和进程挂起前的数据

子进程 的子进程 构成进程树

系统调用：进程启动 挂起 请求内存 释放内存 等待子进程结束

进程间通信 发送 应答 重发 系统发送信号给进程 UID 用户ID

地址 内存多个进程 防止干扰 虚拟内存技术:分配抽象的内存空间，一部分存在磁盘上，需要的时候才加载

虚拟内存和实际内存不是一一对应，而是由系统来分配和映射，相连的虚拟内存地址在实际内存中可能是断开的

文件 隐藏磁盘和io设备硬件细节 抽象成文件 目录结构 增删查改 root根目录 path

读写文件 先要检查权限 会返回一个文件描述符(资源，句柄) unix特殊文件

块特殊 磁盘 字符特殊文件 打印机字符流 管道 进程间交流数据

访问权限 shell 标准输出 > 输出重定向 程序输出作为另一个程序输入 |管道

单cpu一次一条指令，用户程序使用系统服务，系统调用 read(file,buffer,bytes)

参数压栈 值调用 buffer引用传递（传的是地址） trap 陷入内核

posix有一百个左右系统函数 进程fork派生 创建进程副本子进程 waitpid exit execve(command,param)执行命令 文件读写 read write lseek文件读写指针 目录和文件增删 mkdir rmdir link创建文件连接（可改名） 杂项 chdir 工作目录 chmod（）改变文件权限 kill杀进程

winapi 有几千个 不全是内核级调用 管理gui

进程 创建 等待 终止 文件打开关闭读取 删除

2、 系统类别 单体 一个可执行文件 分层系统 内核层应用层 微内核 c-s模式 服务器 客户端 两个进程之间交互 虚拟机 模拟裸机，上面运行软件 jvm

C语言 指针 没有垃圾回收 没有类型安全 线程 内建字符串 头文件 宏 条件编译

目标文件 o或exe

make makefile 只重编译修改项 再连接成out

二、进程和线程

并行，每个进程运行几十或几百毫秒就迅速切到另一个

顺序进程 模型 一个进程就包括 eip 寄存器 变量当前值（内存），这些挂起时都在内存中，当运行时加载到真正cpu中

有系统进程 和用户进程

后台进程 又称为守护进程daemon ps指令 新进程都是由存在的进程执行了createProcess产生的 fork是复制进程和execve 修改进程存储映像共同使用

不同进程地址空间不同，互不干扰。

进程终止 正常 错误 严重错误 被杀死

进程层级 父子 unix和win不同 win父子同级，但可以用句柄控制子，也可以委托其他进程控制

进程状态 运行 就绪 阻塞 发生中断时本进程会阻塞

进程表，每个进程占一项 包括各种信息 寄存器 cs:ip ss:sp 优先级 包括进程管理，存储管理，文件管理信息 进程id 用户id 文件句柄 工作目录 各内存段的指针

中断向量存储中断服务（如磁盘io）的地址，中断时 进程入栈 跳转到中断程序 IO耗时很长

线程 共享一个地址空间 共享数据 容易创建和撤销 大量io的话会很快

一般io因为中断程序会卡住，可以给鼠标键盘加入中断机制，中断掉io，之后再还原，这样很麻烦。。。。。。还是多线程

并发请求，线程挂起和恢复

分派线程 和工作线程 thread 有寄存器 有堆栈 纳秒级切换线程 cpu在线程间快速切换 载入寄存器数据

线程可以访问 进程空间中任何位置，线程之间无保护 ，共享打开的文件，全局变量，子进程

线程堆栈给函数使用 thread\_create thread\_exit thread\_join等待线程退出 thread\_yield 强制让给下个线程 父线程应该共享给子进程吗？

线程 访问同一个文件的问题 锁 私有全局

线程表 只记录各线程 cs ss堆栈 寄存器 挂起时存好数据到内存

另一个阻塞机制:提前通知 然后先执行其他函数，在安全的时候再执行会阻塞的函数 （回调机制）

内核中实现线程 内核级线程 速度慢 混合线程

3、进程间通信ipc 竞争条件 读取共享数据（内存或者文件） 互斥 其他进程不能用

忙等待互斥 1、屏蔽中断 2、锁变量 （实际上是个全局变量） 为0则可读写 否则不行 3、严格轮换 4、tsl xchg

4、睡眠和唤醒 生产者消费者 一个写入一个取出 有界缓冲区 当满的时候还想放入数据，就必须让写入沉睡，等到取出空间

5、信号量 sleep wakeup 6、互斥量 解锁加锁 mutex\_lock

条件变量 7、管程 死锁 两个进程都阻塞 管程只有一个活跃进程

8、消息传递 send receive 9.屏障

4、调度 调度程序 管理cpu交替 io活动使用cpu吗？从磁盘到内存? 网络io,只有阻塞时才是io

2、调度时机 创建 退出 阻塞 中断完成（就绪恢复） 时钟发送周期性中断50h

3、调度算法选择原则 ，优先

IPC问题 哲学家就餐 读者写者

三、存储管理（内存

1、无抽象 实地址模式 相当于大数组 过去用ram rom分层 利用磁盘 一个时候只有一个程序

2.抽象 地址空间 进程地址受保护 重定位 每个程序都有自己的空间从0到2G

处理进程过多， 内存超载

1、交换技术 运行一段时间，然后存回磁盘，空闲的时候在磁盘上 进程切换时在磁盘和内存间来回 运行时申请内存

空闲的内存管理 位图或列表

2、虚拟内存，只有一部分程序被调入内存 仍可以执行 （软件超过内存） 把磁盘当内存用，内存修改的话磁盘也会同步修改，因为这块磁盘相当于内存

进程地址空间 分成多块 （页） 页被映射到物理内存 执行到时才进行映射

虚拟地址 被送到内存管理单元映射成 物理地址 页面大小固定 用页表进行对应，页号

虚拟地址=页号:偏移量

页表项 访问位 保护位-可读/可写 在不在-缺页中断 已修改则存回硬盘 未修改直接丢弃

加速分页 转换检测缓冲区tlb 快表 多级页表倒排页表

缺页中断 置换页面 算法lru

负载 页面大小 可以把指令空间和数据空间分隔 共享页面

共享库 动态连接库dll so

内存映射文件 文件映射到进程虚拟内存空间 动态按页写入写回 （这里的文件是指用于当虚拟内存的文件，所以会自动映射修改 ）

何时清除？

多进程共享内存 共享页面

2缺页中断 时 进程或线程停止 找到虚拟页面 写回 或者读出 页表随后正常

4、页面锁定 文件映射 和io的相互作用

5、后备存储 页面换出时存在哪 特殊交换区 windows有个虚拟内存文件

进程的交换区 磁盘地址 记录在进程表

7、分段 编译过程的很多表 源程序 符号表 变量名 常量表 语法分析树 堆栈

段 segment 段可能被装满 段有保护

8、分段分页结合 multics 和 段表 每个段相当于一个虚拟内存

奔腾 线性地址 基址 偏移量

四．文件系统 长期存储 用于进程共享数据和持久化

构造 命名 存取 使用 保护 实现 管理

扩展名

结构 字节序列

类型 目录 字符特殊文件

Ascii文件和二进制文件 文本文件分行 每行都有回车 换行

文件：文件头 魔数 正文 数据 重定位 符号表

2、存取 顺序，必须从头到尾 随机，可以读取任意部分 read seek

3、文件属性 元数据 大小 创建日期 创建者 只读标志

4、操作 创删 打开（装入内存） 读写 获取修改属性 重命名

5、目录 多级 绝对路径名，相对路径

6、文件系统 主引导记录 分区表 活动分区 引导块中的操作系统 空闲管理

文件分配 1、连续 2 链表分配 每个文件建造链表 文件块连接，这样不会浪费空间

文件分配表FAT 3、i节点

目录实现 共享文件link 有向无环图

日志结构的文件系统 ntfs ext3

虚拟文件系统 vfs

7．空间管理 文件块大小 空闲块记录

8、备份文件

9、文件系统一致性，如果崩溃

五、输入输出 IO

CPU控制 IO设备 中断 设备错误 提供程序员使用操作硬件的接口（函数）

块设备 字符设备

设备控制器 具有自己的寄存器 写入命令可以让设备运行 读取可以知道状态

可能还有设备缓存

cpu如何通信 1、使用端口号 in/out port 2、控制寄存器映射到内存空间 分配内存地址 可以直接当成普通内存读写 []

交换 数据 利用cpu命令每次交换 一个字节 很浪费 mov [] []

dma方案 直接 独立于cpu访问总线 在接受了CPU传输的信息后（读/写，需求内存地址，字节数，每次传输字还是字节）

可以自发进行传输到内存，无需cpu

2、中断 io设备完成工作时 产生中断 使阻塞的进程恢复 中断-总线上的中断控制芯片-处理中断 cpu停止目前的行为 中断向量

精确中断 不精确

Io软件 应于设备无关 比如读取文件 错误处理 缓冲buffer 共享设备 死锁

Io是一个持续的过程，由于读取，打印速度，网络连接的差异会导致需要等待完成才能继续下一个io ，这之间需要个机制让cpu放开

1. 程序控制io cpu做所有工作 当设备工作时，cpu等待返回 会占用cpu全部时间 忙等待
2. 中断io 发出命令 切到其他程序，本程序的语句阻塞 打印机完成-发送中断-cpu停止当前进程-执行中断处理程序恢复阻塞
3. Dma 最简单的形式 只在缓冲区满或者任务结束的时候 对cpu下达中断，要求清空白缓存

IO层次 ：用户io 系统 设备驱动 中断处理程序 中断非常复杂，需要相当多cpu指令

设备 cd 时钟 键盘鼠标 监视器

输出软件 xwindow gui 位图

六、死锁

资源

七、多媒体

八、多处理器

九、安全

1. Liunx
2. Windows vista

Windos下程序如何运行 内核态到用户态

1、层次 NT服务 gui user32.dll gdi.dll 子系统api kernel.dll 动态库rpc.dll 子系统进程crss 本地NT API c++运行时 ntdll.dll

内核层:设备驱动 gui驱动 ntoskrenl.exe 硬件抽象层hal

系统调用由微软写 ，用户态接口在子系统中 子系统.net win32 winfx .net运行库 clr

一个应用 使用子系统库(win32或者.net)，createProcess，通过子系统进程smss.exe启动crss.exe来深入到内核层 win32pai-调用内核api lpc调用，跨进程

NtcreateProcess 提供串表 共享字符串

Nt系统调用 对内核对象进行操作，文件，管道，信号量，线程 io 互斥量 ipc端口 i/o完成队列 设备 驱动 （这里的对象和面向对象无关）

创建和打开对象都返回句柄 可以把句柄复制到其他进程句柄表 共享访问对象

安全描述信息 访问限制

Api：nt创建进程 线程 读写虚拟内存 创建文件 NtDuplicateObject 复制句柄

参数中需要句柄，可以读取任意进程的空间

对象管理器 可以给对象命名，用名字打开句柄 甚至永久命名

操作系统 本身就在创建使用对象 驱动 设备

名字空间 在内核的虚拟内存中

Win32api 来自kernel.dll 里面会调用原始api 用win 函数的参数来传入到native函数参数 ，需要通知csrss.exe 监督 创造线程 读取文件 挂起线程 读取删除文件 转移句柄 申请内存

关于ASCII编码 和Unicode编码 有两套程序 A W

特殊环境 wow32 wow64 允许32位在64位运行

内存映射文件 1、进程自己管理 虚拟空间 2、文件映射 称为section 3、文件映射试图映射到地址空间 句柄 可以复制到其他进程

文件io 字节流 文件文件目录 文件属性 nfts的文件流可加密 异步

卷 raid io可取消 可支持同步io

安全 线程有访问权限列表，各用户权限

支持gui 使用win32.sys驱动 创建管理窗口 各种工具条

2、注册表

保存有配置 hive文件 启动时加载system 存了大量变量和驱动信息 其他的块保存了软件信息 如com组件对象模型 用户信息 键值对

3、内核 包括执行体(大部分服务)和内核（线程，异常，陷阱，中断，cpu）

执行体 lpc进程线程 虚拟内存和高速缓存 对象和io管理 配置和安全

设备驱动器（文件，杀毒，网络协议 ）

Hal抽象 管理中断等

调度程序 同步机制

control对象 延迟过程调用 DPC减少isr中断处理程序时间 优先级

APC 异步过程调用

dispathcher对象 调度对象 调度头:对象头，同步标记，

通知对象信号

执行体 数据不能在用户态访问 对象管理器 命名 分配内存 句柄访问

驱动 内核错误导致蓝屏 进程管理 内存管理 分页虚拟内存

安全 lpc/alpc,进程间通信（基于rpc）

1. 系统启动 cpu-bios-引导程序-windows bootmgr-休眠唤醒或者启动 winresume winload-加载内核执行体 hal.dll win32.sys 和各种驱动-内核加载完毕smss.exe运行用户态
2. 内核对象仅仅只是数据结构而已 保存了某些信息 对象头-对象数据 curl方法 引用计数，没使用会被释放
3. 句柄 进程句柄表 内核句柄表保存在系统进程

创建新对象 打开对象 OpenSemphore 7、名字空间Parse winobj.exe可以看到 名字空间 目录：所有io设备 用户创建的对象 win32gui的对象

1. 打开文件返回句柄 NtCreateFile 传入路径名 靠目录和符号链接表找到设备对象-调用Parse-创建irp(io请求包)-分配文件对象-发送请求到io设备堆栈
2. 对象类型 进程 线程 互斥 锁 定时器 队列（异步io通知） 打开的文件句柄

文件映射 设备io 注册表 信号量 事件 端口lpc

用户态组件 环境子系统 dll 服务进程

Windows中几乎所有库都是dll 每个进程都有ntdll.dll ITA导入地址表

Com应用

服务进程 lsass.exe 用户身份令牌token 加密密钥 很多服务运行在svchost.exe中 服务经常有安全漏洞

9、进程 虚拟地址空间 线程句柄 令牌 线程优先级 PEB进程环境块（包括加载的模块列表,exe dll等各种数据） 线程id 进程id

线程有两个堆栈 应用态和内核态 teb 是cpu调度基本单位

用户共享数据 只能内核写 用户读

段对象创建进程 映射段 分配虚拟内存 写参数 环境变量 文件描述符到句柄表

作业 进程 线程 纤程

线程执行系统调用时 进入内核态 用户态寄存器 存入内核堆栈 context形式

线程创建时 Cpu上下文信息（栈指针和cs:ip）

1. 进程间通信 管道 命名管道 邮件槽 套接字 远程过程调用rpc 共享文件

字节管道 消息管道

Rpc实现 用tcp/ip套接字 命名管道 alpc实现

进程间共享对象 如段对象

1. 同步 信号量 互斥量 临界区 事件

CreateSemaphore 创建信号量 可以复制其句柄传递给其他进程 相同的信号量实现同步 可命名 可用acl集合保护 up down调用 ReleaseSemaphore(up) Waitforsingleobject(down)

互斥是锁 waitforsingleobject上锁 解锁release mutex 可以复制 传递

临界区 不能传递 上锁 解锁entercritica section() leave……

事件 通知和同步事件 状态 收到信号没收到信号 等待事件 Waitforsingleobject

设置事件 setevent resetevent

11、调度 发生阻塞时信号量 互斥 事件 io 线程向对象发信号时 时钟切换

设置线程优先级 setPriorityClass 设置相对优先级setThreadPriority

饥饿

12、内存管理

每个程序有4GB空间 虚拟地址 内核进程2G 用户2G

13、存储管理 虚拟地址描述符VAD 给进程 exe dll映像 写时复制 超级预读

页表项 失效页表 物理页号，虚拟页号

未完