一．

Unix系统包含 用户程序 和 系统内核

内核管理所有的程序资源

系统资源

1. 处理器

执行指令的硬件设备 CPU

1. 输入输出
2. 进程管理
3. 内存
4. 设备
5. 计时器
6. 进程间通信
7. 网络

理解系统编程

1. 分析程序
2. 学习系统调用
3. 编程实现

系统会启动shell程序，由这个程序处理用户请求 每个用户都有属于自己的shell进程

通信 协作 网络访问

客户/服务模型 类似前端/后端一样 前端接收 后端处理

二．

打开文件:open read close

进程与文件的连接 称为 文件描述符fd 打开文件是内核提供的服务

缓冲区是进程内存的一部分

内核缓冲 write先写到缓冲区

四．文件系统

超级块 文件系统本身的结构信息

i-节点表 文件属性

数据区 文件内容保存的区域

五．设备编程

设备和文件的不同

终端文件具有回显、波特率、编辑和换行会话

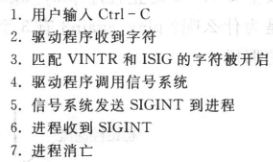
文件 缓冲 自动添加模式

六．终端控制和信号

规范 有缓冲 先缓存后由驱动器处理

非规范 没缓冲 直接处理

信号



信号的来源 1.用户 2.内核 3.进程

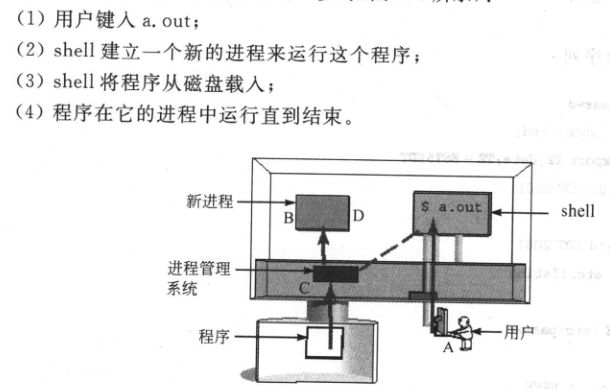
处理多信号的3给方法:

1.递归调用同一个参数

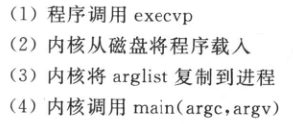
2.忽略第二个信号

3.阻塞信号

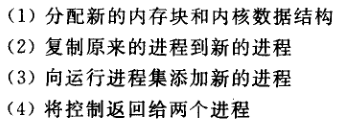
八．进程和程序



一个程序如何运行另一个程序 程序调用execvp

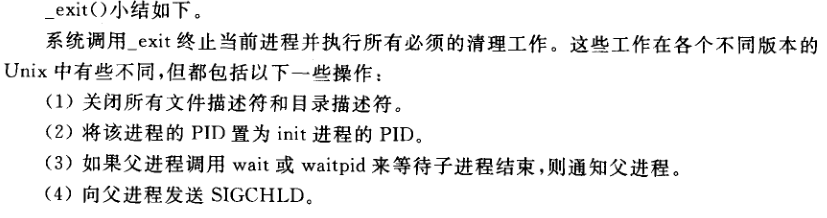


如何建立新进程? fork() 进程复制自己

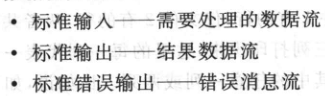


辨别父进程子进程！ 根据fork()返回的pid

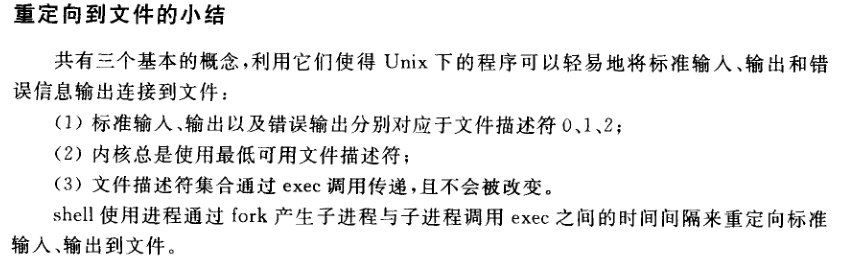
父进程如何等待子进程的退出 进程调用wait()等待子进程结束 子进程exit()退出



十．I/O重定向和管道





实现 who | sort 需要两种技巧

1. 如何创建管道 pipe()
2. 如何将标准输入和输出通过管道连接起来 管道两端连接文件描述符

管道并非文件

1. 从管道中读数据

管道读取阻塞

管道的读取结束标志

多个读者可能会引起麻烦

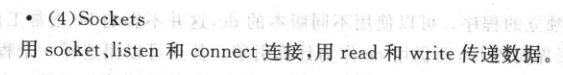
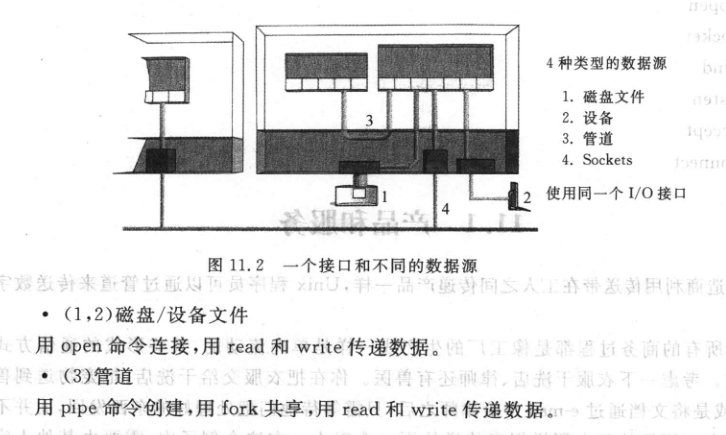
1. 从管道中写数据

写入数据阻塞直到管道有空间去容纳新的数据

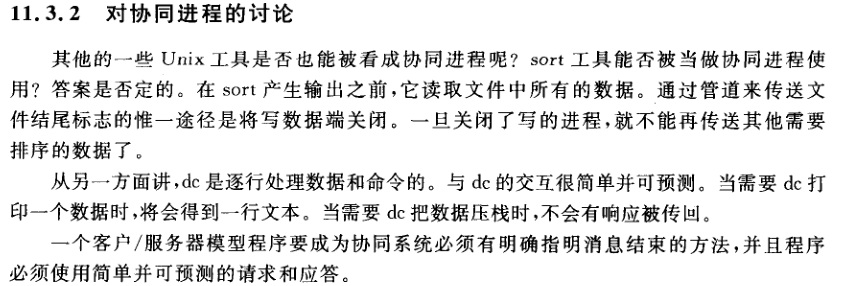
写入必须保证一个最小的块大小

若无读者在读取数据，则写操作执行失败

十一.连接到近端或远端的进程:服务Socket



协同进程 两个程序都持续运行，当其中一个程序完成自己的工作后将控制权传给另一个程序

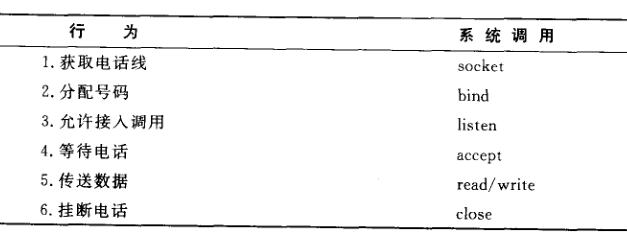


socket与远端进程相连

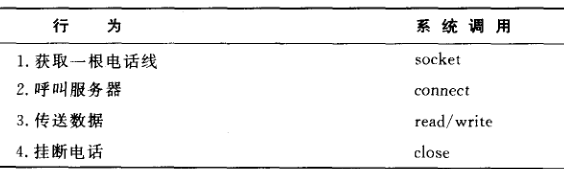
管道被创建 只能fork()实现共享 也只能连同一主机上的进程

文件/etc/services 定义了总所周知服务端口的列表

服务端



客户端



十二.连接和协议

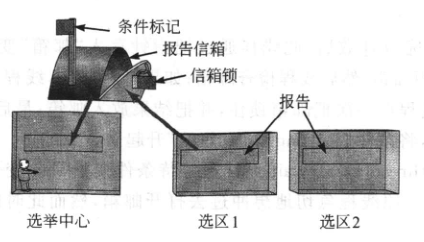
十四.线程机制

创建线程 pthread\_create

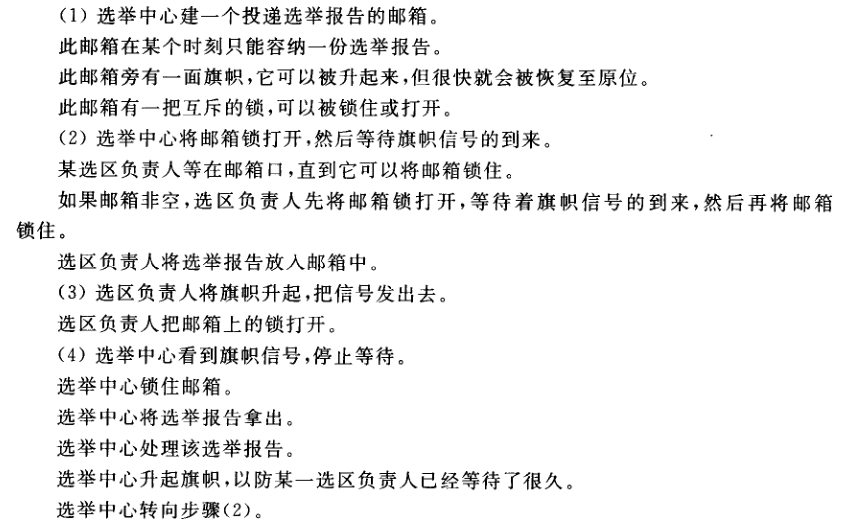
进程:独立的数据空间 文件描述符 进程的ID

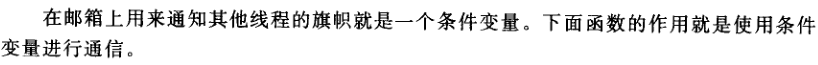
线程:共享数据空间 共享的文件描述符 fork exec exit Signals

线程间互通消息:



容器 旗帜 锁





pthread\_cond\_wait() 使线程挂起直到另一线程通过条件变量发出消息

pthread\_cond\_signal() 通过条件变量 cond发消息 多个线程只唤醒一个

pthread\_join() 等待线程返回 收回资源

pthread\_create() 创建独立线程 自动释放资源

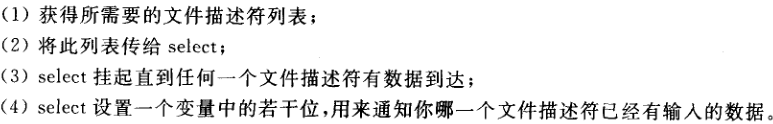
十五.进程间的通信

管道\socket\信号\退出/等待

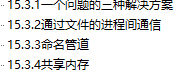
Talk 同时等待从两个文件描述符的输入

Talk 从键盘和socket接收数据

select 系统调用





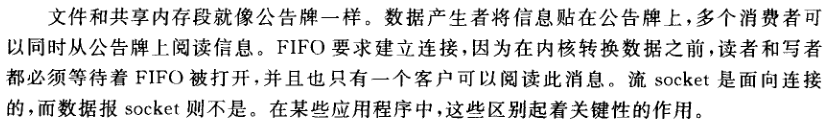


选择标准:

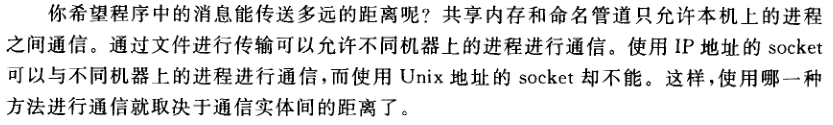
1. 速度

系统内核将数据复制到内核空间中，然后再切换回用户空间

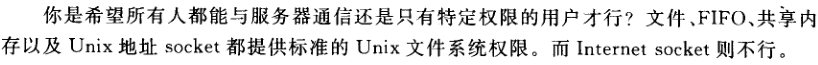
1. 连接和无连接



1. 范围



1. 访问限制



1. 竞态条件

