UC Day07

预习课

预习内容

进程的概念

相关命令

父子孤尸

进程的概念

进程的概念

知识讲解

- 进程：是程序执行时的一个实例
  - 程序是被存储在磁盘上，包含机器指令和数据的文件
  - 当这些指令和数据被装载到内存并被CPU所执行，即形成了进程。
  - 一个程序可以被同时运行为多个进程
  - 在Linux源码中通常将进程称为任务（task）
  - 从内核观点看，进程的目的就是担当分配系统资源（CPU时间，内存等）的实体

相关命令

相关命令

知识讲解

- pstree 以树状结构显示当前所有进程关系

- ps 以简略方式显示当前用户拥有控制终端的进程信息，也可以配合以下选项
  - a - 显示所有用户拥有控制终端的进程信息
  - x - 也包括没有控制终端的进程
  - u - 以详尽方式显示
  - w - 以更大列宽显示

相关命令

- 进程信息列表
  - user ： 进程的用户ID
  - PID ： 进程ID
  - %CPU ： CPU使用率
  - %MEM ： 内存使用率
  - VSZ ： 占用虚拟内存的大小（KB）
  - RSS ： 占用物理内存的大小（KB）
  - TTY ： 终端次设备号

知识讲解

相关命令

知识讲解

- 进程信息列表
  - STAT ：进程状态
    - R - 运行，即正在被处理器执行
    - S - 可唤醒睡眠，系统中断、获得资源、收到信号，都可唤醒
    - D - 不可唤醒睡眠，只能被wake_up系统调用唤醒
    - T - 收到SIGSTOP(19)信号进入暂停状态，收到SIGCONT(18)信号后继续运行
    - Z - 僵尸，已终止但其终止状态未被回收
    - < - 高优先级
    - N - 低优先级
    - L - 存在被锁定的内存分页
    - s - 会话首进程
    - l - 多线程化进程
    - + - 在前台进程组中

相关命令

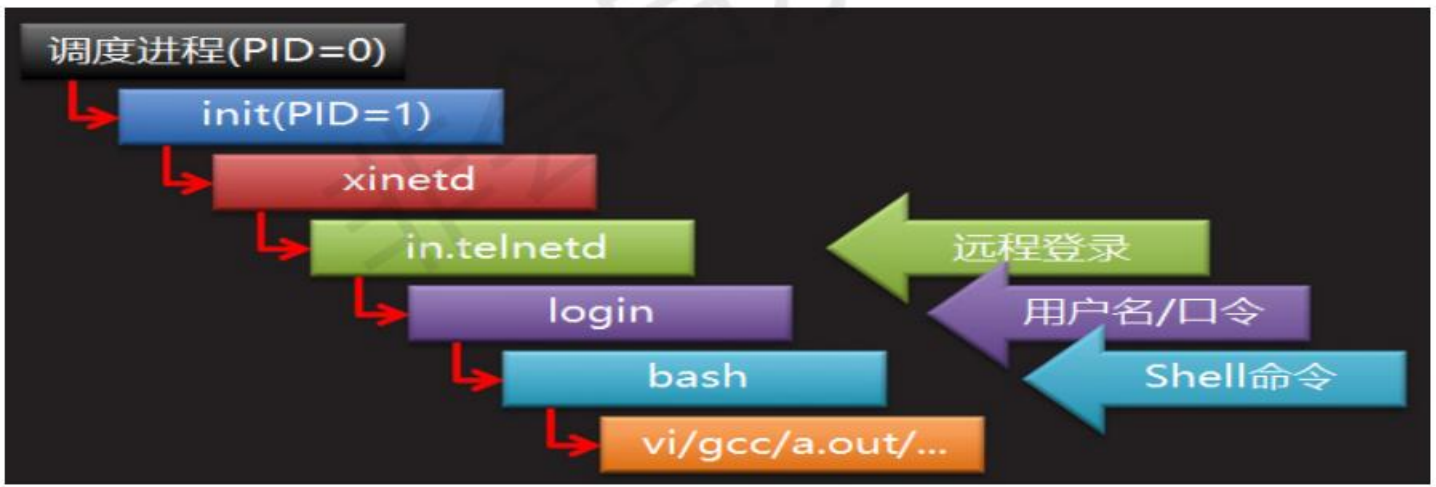
- 进程信息列表
  - START ：进程开始时间
  - TIME ：进程运行时间
  - COMMAMD ：进程启动命令

知识讲解

父子孤尸

父子孤尸

- Unix系统中的进程存在父子关系。一个父进程可以创建一到多个子进程，但每个子进程有且仅有一个父进程。整个系统中只有一个根进程，即PID为0的调度进程



父子孤尸

- 父进程创建子进程以后，子进程在操作系统的调度下与其父进程同时运行如果父进程先于子进程终止，子进程即成为孤儿进程，同时被某个专门的进程收养，即成为该进程的子进程，因此该进程又被称为孤儿院进程

知识讲解

父子孤尸

知识讲解

- 父进程创建子进程以后，子进程在操作系统的调度下与其父进程同时运行。如果子进程先于父进程终止，但由于某种原因，父进程并没有回收该子进程的终止状态，这时子进程即处于僵尸状态，被称为僵尸进程
- 僵尸进程虽然已不再活动，即不会继续消耗处理机资源，但其所携带的进程终止状态会消耗内存资源。因此，作为程序的设计者，无论对子进程的终止状态是否感兴趣，都应该尽可能及时地回收子进程的僵尸。

直播课见

UC

C/C++教学体系

目录

进程标识

进程标识

知识讲解

- 每个进程都有一个非负整数形式的唯一编号，即PID(Process Identification，进程标识)
- PID在任何时刻都是唯一的，但是可以重用，当进程终止并被回收以后，其PID就可以为其它进程所用
- 进程的PID由系统内核根据延迟重用算法生成，以确保新进程的PID不同于最近终止进程的PID

进程标识

- 系统中有些PID是专用的，比如
  - 0号进程，调度进程，亦称交换进程(swapper)，系统内核的一部分，所有进程的根进程，磁盘上没有它的可执行程序文件
  - 1号进程，init进程，在系统自举过程结束时由调度进程创建，读写与系统有关的初始化文件，引导系统至一个特定状态，以超级用户特权运行的普通进程，永不终止
  - 除调度进程以外，系统中的每个进程都有唯一的父进程，对任何一个子进程而言，其父进程的PID即是它的PPID

知识讲解

进程标识

知识讲解

- 相关函数
  - #include <unistd.h>
  - pid_t getpid(void); // 返回调用进程的PID
  - pid_t getppid(void); // 返回调用进程的父进程的PID
  - uid_t getuid(void); // 返回调用进程的实际用户ID
  - gid_t getgid(void); // 返回调用进程的实际组ID
  - uid_t geteuid(void); // 返回调用进程的有效用户ID
  - gid_t getegid(void); // 返回调用进程的有效组ID

创建子进程

创建子进程

知识讲解

- #include <unistd.h>
- pid_t fork(void);
  - 功能：创建调用进程的子进程
  - 返回值：成功分别在父子进程中返回子进程的PID和0，失败返回-1
  - 注意！！！该函数调用一次返回两次，在父进程中返回所创建子进程的PID，而在子进程中返回0，函数的调用者可以根据返回值的不同，分别为父子进程编写不同的处理分支
  - 系统中总的线程数达到了上线，或者用户的总进程数达到了上线，fork函数会返回失败

创建子进程

- 为父子进程设计相同的执行过程

```
int main(void) {
    ......

    pid_t pid = fork();

    printf(“父子进程都执行的代码\n”);
    return 0;
}
```

知识讲解

创建子进程

- 为父子进程设计先不同而后相同的执行过程

```
int main(void) {
        ......
        pid_t pid = fork();
        if(pid == 0){
                printf(“子进程代码\n”);
        }else{
                printf(“父进程代码\n”);
        }
         printf(“父子进程都执行的代码\n”);
         return 0;
}
```

知识讲解

创建子进程

- 为父子进程设计不同的执行过程

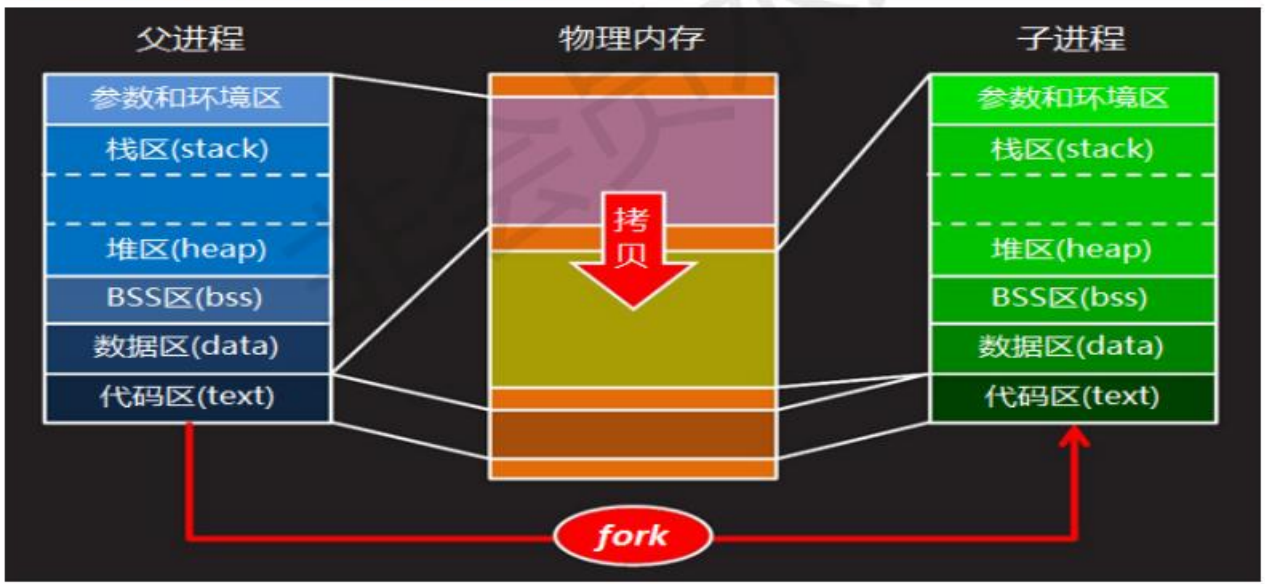
```
int main(void) {
        ......
        pid_t pid = fork();
        if(pid == 0){
                printf(“子进程代码\n”);
                return 0;
        }

        printf(“父进程代码\n”);
        return 0;
}
```

知识讲解

父子进程间的关系

父子进程间的关系

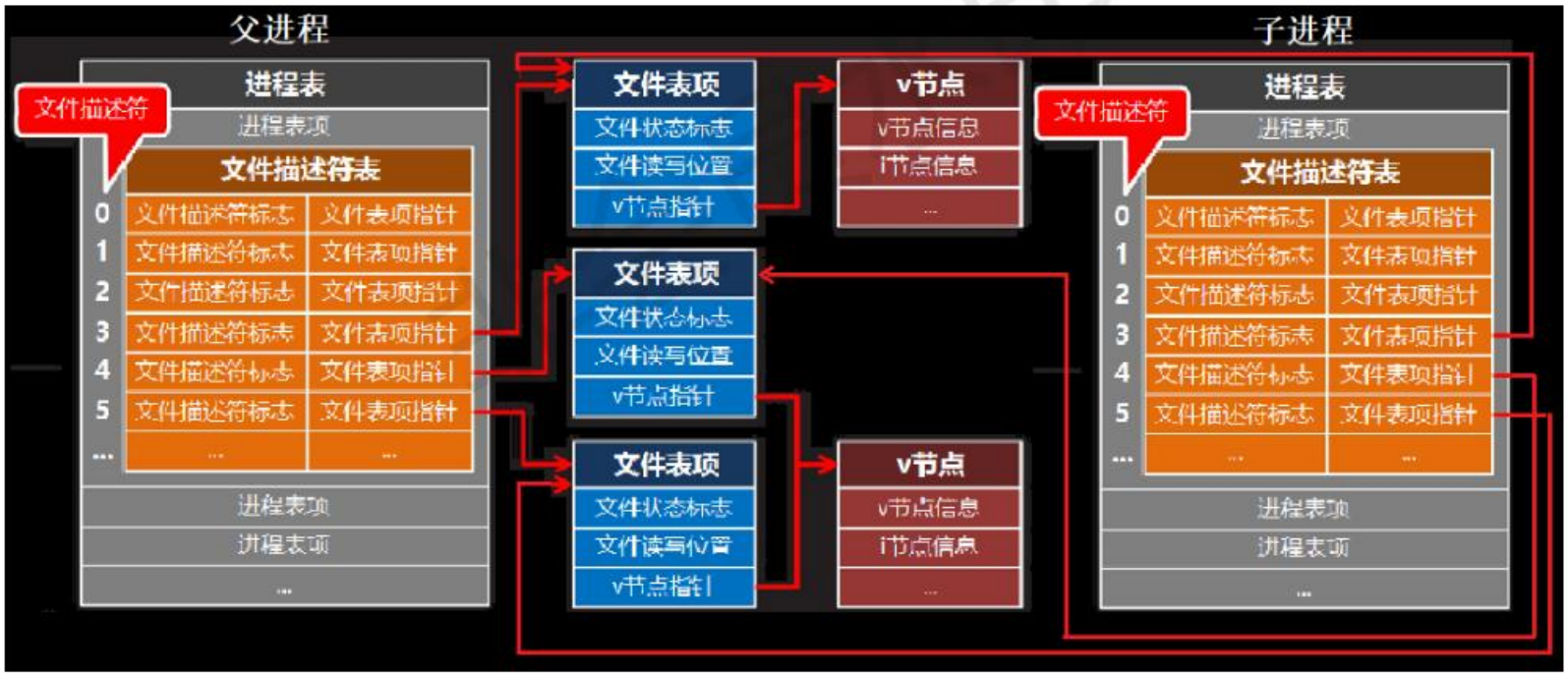
- 由fork产生的子进程是其父进程的不完全副本，子进程在内存中的映像除了代码区与父进程共享同一块物理内存，其它各区映射到独立的物理内存，但其内容从父进程拷贝。



知识讲解

父子进程间的关系

- fork函数返回后，系统内核会将父进程维护的文件描述符表也复制到子进程的进程表项中，但并不复制文件表象



谢谢

UC Day07

复习课

僵尸进程

僵尸进程

知识讲解

- 父进程创建子进程以后，子进程在操作系统的调度下与其父进程同时运行。如果子进程先于父进程终止，但由于某种原因，父进程并没有回收该子进程的终止状态，这时子进程即处于僵尸状态，被称为僵尸进程
- 接下来我们通过代码，来实际演示下僵尸进程

僵尸进程

- 代码思路
  - int main(void){
    //父进程创建子进程
    //子进程代码中，快速结束
    //父进程代码中，睡眠10秒
  - }

下节课见