1.进程&线程（16道）

1.1异步IO和同步IO区别?

答案：如果是同步IO，当一个IO操作执行时，应用程序必须等待，直到此IO执行完。相反，异步IO操作在后台运行，IO操作和应用程序可以同时运行，提高系统性能，提高IO流量。

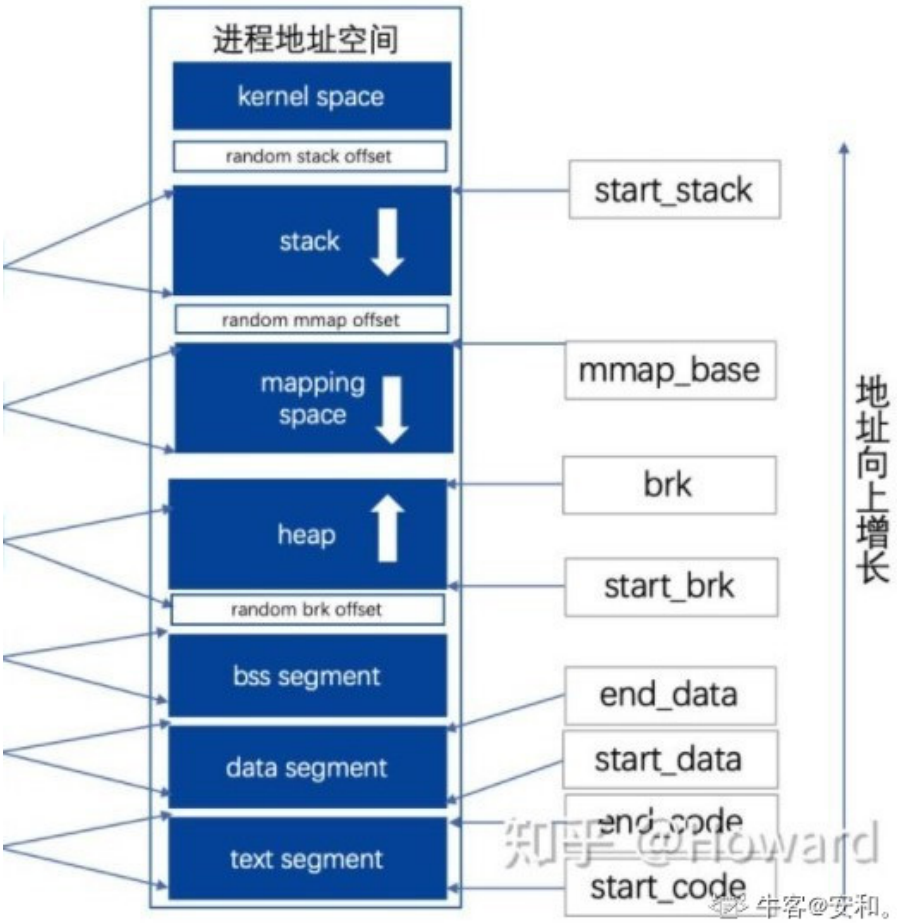
解读：在同步文件IO中，线程启动一个IO操作然后就立即进入等待状态，直到IO操作完成后才醒来继续执行。而异步文件IO中，线程发送一个IO请求到内核，然后继续处理其他事情，内核完成IO请求后，将会通知线程IO操作完成了。

1.2进程间通信方式?

管道 (pipe)	是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程之间使用，通常是父子进程。
有名管道 (named pipe)	也是半双工的通信方式，但是它允许用于无亲缘关系的进程之间的通信。
信号量 (semophore)	是一个计数器，通常作为一种同步机制，用于进程和线程间的同步。
消息队列 (message queue)	是一个消息链表，存放在内核中并且由消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限的缺点。
共享内存 (shared memory)	一段能够被多个进程共同访问的内存，由一个进程创建。共享内存是最快的IPC方式，它是针对其他进程间通信方式运行效率低而设计的，往往与其他通信方式如信号量配合使用，来实现进程间同步与通信。
套接字(socket)	可用于不同主机间的进程通信。

信号(signal)	用于通知接收进程某个事件已经发生，是一种比较复杂的通信方式。

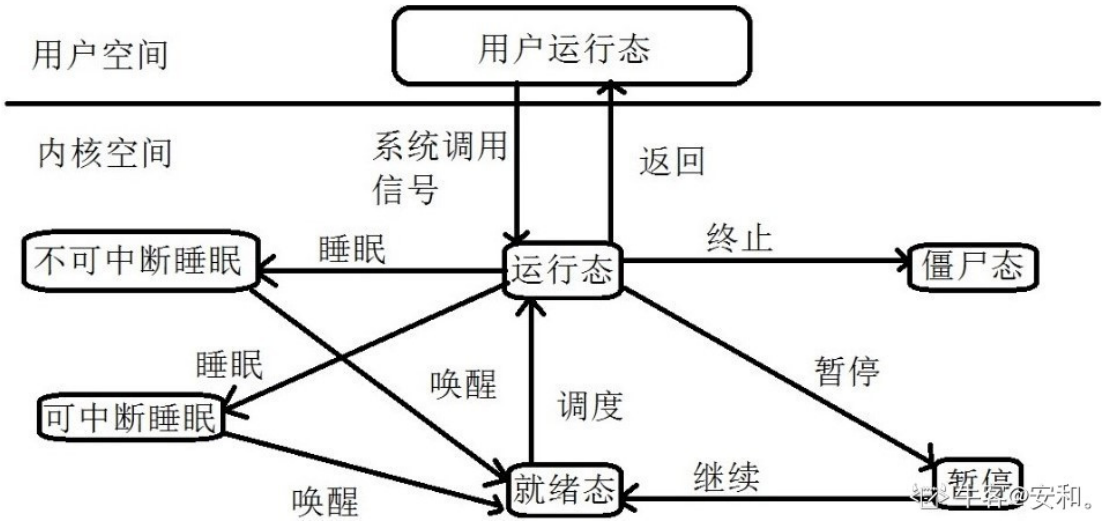
1.3进程的地址空间模型?



text segment	存储代码的区域。
data segment	存储初始化不为0的全局变量和静态变量、const型常量。
bss segment	存储未初始化的、初始化为0的全局变量和静态变量。
heap（堆）	用于动态开辟内存空间。
memory mapping space （内存映射区）	mmap系统调用使用的空间，通常用于文件映射到内存或匿名映射（开辟大块空间），当malloc大于128k时（此处依赖于glibc的配置），也使用该区域。在进程创建时，会将程序用到的平台、动态链接库加载到该区域。
stack（栈）	存储函数参数、局部变量。

kernel space	存储内核代码。

1.4进程的五种状态分别是?



（1）就绪态：所有运行条件已就绪，只要得到了CPU时间就可运行。

（2）运行态：得到CPU时间正在运行。

（3）僵尸态：进程已经结束了但父进程还没来得及回收。

（4）等待态：包括浅度睡眠跟深度睡眠。进程在等待某种条件，条件成熟后即进入就绪态。浅度睡眠时进程可以被信号唤醒，但深度睡眠时必须等到条件成熟后才能结束睡眠状态。

（5）暂停态：暂时停止参与CPU调度（即使条件成熟），可以恢复。

1.5子进程从父进程继承的资源有哪些?

答案：用户号和用户组号、目录（当前目录、根目录）、打开的文件描述符、堆栈、共享内存。

注意：父进程和子进程拥有独立的地址空间和PID参数、自己的文件描述符、不同的父进程号。

1.6什么是进程上下文、中断上下文?

（1）进程上文：是指进程由用户态切换到内核态时需要保存用户态时CPU寄存器中的值，进程状态以及堆栈上的内容。即保存当前进程的状态，以便再次执行该进程

时，能够恢复切换时的状态，继续执行。

（2）进程下文：是指切换到内核态后执行的程序，即进程运行在内核空间的部分。

（3）中断上文：硬件通过中断触发信号，导致内核调用中断处理程序，进入内核空间。这个过程中，硬件的一些变量和参数也要传递给内核，内核通过这些参数进行中断处理。中断上文可以看作硬件传递过来的这些参数和内核需要保存的一些其他环境（主要是当前被中断的进程环境）。

（4）中断下文：执行在内核空间的中断服务程序。

1.7如何防止僵尸进程过多导致系统空间爆满?

答案：每当子进程退出，父进程都会收到SIGCHLD信号，故可在父进程中设置SIGCHLD信号的捕获函数，在捕获函数中回收子进程。

```
void* handler(int sig)
{
    int status;
    if(waitpid(-1, &status, WNOHANG) >= 0)
    {
        printf("child is die\n");
    }
}
int main()
{
    signal(SIGCHLD, handler);
    int pid = fork();
    if(pid > 0)  //父进程循环等待
    {
        while(1)
        {
            sleep(2);
        }
    }
    else if(0 == pid)
    {   //子进程说自己die后就结束生命周期，之后父进程就收到SIGCHLD
        //信号调用handler函数接收结束子进程，打印child is die。
        printf("i am child, i die\n");
        exit(0);
    }
}
```

解读：

（1）僵尸进程是当子进程比父进程先结束，而父进程又没有回收子进程，释放子进程占用的资源，此时子进程将成为一个僵尸进程。

（2）可用top命令查看僵尸进程的数量（zombie），也可用ps -aux | grep Z（僵尸进程的状态显示为Z）来查看僵尸进程的PID等信息。

1.8 /proc目录下，以数字命名的目录表示什么?

以数字命名的目录表示当前一个运行的进程，目录名即为进程PID，其内的目录和文件给出了一些关于该进程的信息。

1.9进程和线程有什么区别?

（1）进程是系统中程序执行和资源分配的基本单位，线程是CPU调度的基本单位。

（2）一个进程个拥有多个线程，线程可以访问其所属进程地址空间和系统资源（数据段、已经打开的文件、I/O设备等），同时也拥有自己的堆栈。

（3）同一进程中的多个线程可以共享同一地址空间，因此它们之间的通信实现也比较简单，而且切换开销小、创建和消亡的开销也小。而进程间的通信则比较麻烦，而且进程切换开销、进程创建和消亡的开销也比较大。

1.10什么时候用多进程？什么时候用多线程?

（1）当程序的安全性、稳定性要求较高时用多进程。

（2）需要频繁通信/切换程序/创建跟销毁程序时用多线程。

1.11线程可以独立运行吗？一个线程崩溃会导致整个进程崩溃吗?

答案：线程不能独立运行，但一个线程崩溃不一定导致整个进程崩溃。

解读：

（1）线程属于进程，线程的运行需要依赖进程的地址空间和系统资源。

（2）线程崩溃的本质就是内存出错，若出错的内存没有被其他线程访问，则不会导致其他线程出错，也就不会导致进程崩溃。

1.12写出下列线程、互斥锁、信号量相关代码。

定义一个线程ID变量	pthread_t tid;
创建线程	pthread_create(&tid, NULL, pthread_func, NULL);
等待子线程结束，并回收资源	pthread_join(tid, NULL);

与当前进程分离	pthread_detach(tid);
退出调用线程	pthread_exit(NULL);
取消线程	pthread_cancel(tid);
创建互斥锁	pthread_mutex mutex = PTHREAD_MUTEX_INITIALIZER;
初始化一个互斥锁	pthread_mutex_init(&mutex, NULL);
对互斥锁上锁	pthread_mutex_lock(&mutex);
对互斥锁解锁	pthread_mutex_unlock(&mutex);
定义一个信号量	sem_t sem;
创建信号量并初始化它的值	sem_init(&sem, 0, 1);
信号量的值减1	sem_wait(&sem);
信号量的值加1	sem_post(&sem);

1.13线程间通信和同步方式有哪些?

信号、信号量、互斥锁、条件变量、自旋锁、读写锁。

1.14如何防止同时产生大量的线程?

（1）方法是使用线程池。

（2）线程池可以提高调度效率和限制资源使用，线程池中的线程达到最大数时，其他线程就会排队等候。

1.15互斥锁与信号量的区别?

（1）信号量用于线程同步，互斥锁用于线程互斥。

（2）信号量可以为非负整数，可以实现多个同类资源的多线程同步；互斥锁只能为0/1，只能用于一个资源的互斥访问。

（3）信号量可以由一个线程释放，另一个线程得到；互斥锁的加锁和解锁必须由同一线程分别对应使用，且多个线程使用多个互斥锁必须注意统一顺序，否则可能造成死锁。

1.16一个进程可以创建多少个线程？和什么有关?

（1）一个进程创建线程的个数由虚拟内存的大小和分配给线程的调用栈大小决定。一个线程的栈大小可通过ulimit -s查看，一般是8~10M。

（2）假设一个线程需要占用10M内存，那么在3G的地址空间上大概可以创建300个线程。

讨论 评论

帖子还没人回复，快来抢沙发!