名词解释题

1.内核（kernel）：

内核是一种软件，它控制计算机硬件资源，提供程序运行环境。内核通常包括中断处理、时钟管理、进程控制、进程通信和调度原语，以及资源管理中的基本操作等。

2.程序（program）：

程序是一个存储在磁盘上某个目录中的可执行文件。

3.进程（process）：

程序的执行实例被称为进程。

4.线程（thread）

线程是某一个时刻执行的一组机器指令，通常，一个进程只有一个控制线程，即一个进程在某一个时刻只能做一件事情。有了多个控制线程后，在程序设计时可以把进程设计成某一个时刻能够做不止一件事，每个线程处理各自独立的任务。

5.系统调用（system call）：

内核的接口被称为系统调用，公用函数库构建在系统调用接口之上，应用程序既可使用公用函数库，也可使用系统调用。

6.文件系统（file system）：

文件系统是目录和文件的一种层次结构，用于管理存储设备上的文件。

7.POSIX 标准：

POSIX 是一个最初由 IEEE 制定的标准族，指的是可移植操作系统接口（Portable Operating System Interface），后来则扩展成包括很多标记为 1003 的标准及标准草案。

8.原子操作（Atomic operation）：

原子操作指的是由多步组成的一个操作。如果该操作原子地执行，则要么执行完所有步骤，要么一步也不执行，不可能只执行所有步骤的一个子集。

9.符号链接（Symbolic Links）：

符号链接是对一个文件的间接指针，一般用于将一个文件或整个目录结构移到系统中另一个位置。符号链接与硬链接（hard links）的不同在于，硬链接直接指向文件的 i 节点。引入符号链接的原因是为了避开硬链接的一些限制，例如：硬链接通常要求链接与文件位于同一文件系统中；只有超级用户才能创建指向目录的硬链接。

10.控制终端（control terminal）：

由于在 Linux 中，每一个系统与用户进行交流的界面称为终端，每一个从此终端开始运行的进程都会依附于这个终端，这个终端就称为这些进程的控制终端，当控制终端被关闭时，相应的进程都会自动关闭。但是守护进程却能够突破这种限制，它从被执行开始运转，直到整个系统关闭时才退出。如果想让某个进程不因为用户或终端或其他地变化而受到影响，那么就必须把这个进程变成一个守护进程。

11.冲洗（flush）：

在标准 I/O 库中，冲洗指将缓冲区中的内容写到磁盘上。

12.缓冲（buffer）：

缓冲是指用于临时保存 I/O 数据的预留内存空间，目的是尽可能减少使用 read 和 write 调用的次数。

13.全缓冲 I/O（Fully Buffered I/O）：

在填满标准 I/O 缓冲区后才进行实际 I/O 操作。对于驻留在磁盘上的文件通常是由标准 I/O 库实施全缓冲。一个流上执行第一次 I/O 操作时，相关标准 I/O 函数通常调用 malloc 获得需使用的缓冲区。

14.会话（Session）：

会话是一个或多个进程组的集合。

15.shell:

shell 是一个命令行解释器，它读取用户输入，然后执行命令。

16.终端（terminal）：

交互式 shell

17.哑终端（dumb terminal）：

用硬连接连到主机

18.远程终端（remote terminal）：

通过调制解调器连接到主机

19.孤儿进程/进程组（orphan process/process group）：

其父进程已经终止的进程称为孤儿进程，孤儿进程由 init 进程收养。

POSIX.1 将孤立进程组定义为：该组中每个成员的父进程要么是该组的一个成员，要么不是该组所属会话的成员。

20.可靠信号（reliable signal）与不可靠信号（unreliable signal）：

可靠信号是指多个信号发送到进程，进程没来得及处理的信号会排入进程的队列，保证信号不丢失。

信号不可靠指的是，信号可能会丢失；一个信号发生了，但进程却可能一直不知道，同时进程对信号的控制能力也很差，它能捕捉信号或忽略它。早期版本里的另一个问题是，在进程不希望某种信号发生时，它不能关闭该信号，进程能做的一切就是忽略该信号。

21.未决信号：

信号已经被产生于进程表，但是还没被递送至进程。这段时间里信号称为未决的。

22.可重入函数（reentrant function）：

重入一般可以理解为一个函数同时被多个程序调用。可重入函数在任何时候都可以被中断，而一段时间之后又可恢复运行，而相应的数据不会破坏或者丢失。

一般可重入函数有三个条件：1）不使用静态数据结构；2）不调用 malloc 或 free；3）不是标准 IO 函数。

23.读写锁（reader-writer lock）：

读写锁与互斥量类似，从本质上说是一把锁，在访问共享资源前加锁，在访问完成后解锁。读写锁有读模式下加锁状态、写模式下加锁状态、不加锁状态 3 种状态，相比于互斥量，读写锁允许更高的并行性。非常适合对数据结构读的次数远大于写的情况。

24.条件变量（Condition variable）：

条件变量允许线程由于一些未达到的条件而阻塞，此处的“条件”可以由用户来定义，在访问该条件时需要加锁（这个锁是加给该条件的），如果条件没达到，线程将阻塞在该条件上。（为什么条件变量更好呢？因为如果没有条件变量的话，线程要一直尝试获得互斥锁来检查条件是否发生，这样一直占着 CPU 会造成浪费）。线程在因条件未满足并等待前，需要访问“条件”，而“条件”是允许其他线程修改的，因此，访问“条件”时需要加锁，访问结束后释放锁。

25.存储映射 I/O（Memory-Mapped I/O）：

存储映射 I/O 能将一个磁盘文件映射到存储空间中的一个缓冲区上，当从缓冲区中取数据时，就相当于读文件中的相应字节。与此类似，将数据存入缓冲区时，相应字节就自动写入文件。这样就可以在不使用 read 和 write 的情况下执行 I/O。

26.非阻塞式 I/O（Non-blocking I/O）：

使我们可以调用 open、read 和 write 这样的操作，并使这些操作不会永远阻塞。如果这种操作不能完成，则调用立即出错返回，表示该操作如继续执行将阻塞。

27.异步 I/O（Asynchronous I/O）：

由 BSD 和系统 V 派生的所有系统提供了使用一个信号的异步 I/O 方法，该信号通知进程某个描述符已经发生了所关心的某个事件。异步 I/O 的一个限制是：每个进程只有一个信号。如果要对几个描述符进行异步 I/O，那么在进程接收到该信号时并不知道这一信号对应于哪一个描述符。

28.消息队列（Message queue）：

消息队列是消息的链接表，存储在内核中，由消息队列标识符标识。

29.管道(Pipeline)：

管道是 UNIX 系统 IPC 的最古老形式，所有 UNIX 系统都提供此种通信机制。通常有两种局限性：1）半双工；2）只能在具有公共祖先的进程之间使用，比如父子进程之间通信。

30.FIFO：

又称为命名管道，是一种文件类型。它与管道不同，管道只能由相关进程使用，而通过 FIFO，不相关的进程也能交换数据。

31.信号量（Semaphore）：

信号量（Semaphore）是一个计数器，用于为多个进程提供对共享数据对象的访问。在进入一个关键代码段之前，线程必须获取一个信号量；一旦该关键代码段完成了，那么该线

程必须释放信号量。当信号量小于零时，其它想进入该关键代码段的线程必须等待直到第一个线程释放信号量。

32.POSIX 信号量：

POSIX 信号量有两种：有名信号量和无名信号量，无名信号量也被称作基于内存的信号量。有名信号量通过 IPC 名字进行进程间的同步，而无名信号量如果不是放在进程间的共享内存区中，是不能用来进行进程间同步的，只能用来进行线程同步。

33.XSI IPC 好处、不足：

优点：它们是可靠的、流控制的以及面向记录的；它们可以用非 FIFO 次序处理。

不足：IPC 结构是在系统范围内起作用的，没有引用计数；这些 IPC 结构在文件系统中没有名字。

34.IPC：

IPC（interprocess communication，进程间通信）是各种进程通信方式的统称。IPC 包含消息队列、信号量、共享存储等。

35.伪终端（pseudo terminal）：

终端登录是经过由自动提供终端语义的终端设备进行的。在终端和运行程序之间有一个终端行规程，通过该规程我们能设置终端的特殊字符（如退格、行删除、中断等）。但是，当一个登录请求到达网络连接时，终端行规程并不是自动被加载到网络连接和登录 shell 之间的，伪终端则用于提供终端语义。

伪终端这个术语是指，对于一个应用程序而言，它看上去像一个终端，但事实上它并不是一个真正的终端。伪终端仿真串行终端的运行行为，并将终端操作映射为网络操作，反之亦然。

36.临界区（critical regions）：

临界区是一个代码段，在该代码段里进程会可能改变共享数据。

37.权限管理（authorization management）：

权限管理指的是，用户能且只能访问已被系统安全策略授权的资源。

38.I-node 节点：

操作系统中储存文件元信息的一种数据结构，被称为索引节点，包括：文件类型，文件访问权限位、文件所有者，文件大小，指向文件数据块在磁盘上的位置的指针等元信息。

39.shared libraries(共享库)：

共享库使得可执行文件中不再需要包含公用的库函数，而只需在所有进程都可引用的存储区中保存这个库例程的一个副本，这减少了每个执行文件的长度，另一个优点是更新库函数时无需对使用该库函数的程序重新连接编辑。

40.可移植性和可兼容性（Portability&compatibility）：

可兼容性：指一个操作系统执行为其他操作系统或为同一系统的早期版本所编的程序的能力。

可移植性：使整个操作系统以尽可能少的改动移植到一个具有不同处理器或不同配置的计算机上。

41.子进程(child process)

由另外一个进程（父进程）创建的进程。

42.工作目录、当前目录、根目录(working/curent/root directory)

工作目录：一般来说，每个进程都有一个与之相关联的分级文件系统(hierarchical file system)下的目录，称之为该进程的当前工作目录。

当前目录：就是工作目录

根目录：根目录指逻辑驱动器的最上一级目录，它是相对子目录来说的。

43.客户端/服务器模式

它是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到 Client 端和 Server 端来实现，降低了系统的通讯开销。

44.线程私有数据(Thread Private Data)

存储和查询与某个线程相关数据的一种机制。在线程内部，私有数据可以被各个函数访问，但对其他线程是屏蔽的。

45.单实例守护进程(Single Instance Daemon)

为了正常运作，某些守护进程会实现为，在任一时刻只运行该守护进程的一个副本。

46.相对路径与绝对路径(Relative/Absolute Path)

相对路径名：不以斜线开头的路径名称，表示相对于当前文件的路径。

绝对路径名：以斜线为开头的路径名称。

47.计算机的启动过程(Computer startup process)：

BIOS 被通电，BIOS 启动。BIOS 寻找主引导区。主引导区启动系统

48.中断向量、中断例程(Interrupt vector, Interrupt routine)：

每个中断源都有对应的处理程序，这个处理程序称为中断服务程序，其入口地址称为中断向量。

49.dup()函数与 fflush()函数

dup()复制一个现存的文件描述符。

fflush()强制冲洗一个流，使该流所有未写的数据都被传送至内核。

简答题

1.文件的几种类型，分别表述：

普通文件

目录文件：包含了其他文件的名字以及指向与这些文件有关信息的指针

块特殊文件：提供对设备带缓冲的访问

字符特殊文件：提供对设备不带缓冲的访问

FIFO：用于进程间通信

套接字：用于进程间网络通信，也可用于一台宿主机上进程间的非网络通信

符号链接：指向另一个文件

2.解释一下实际 ID、有效 ID（e）、保存设置 ID：

实际 ID（唯一）：取自口令文件/etc/passwd，标识我们究竟是谁

有效 ID：决定了我们的访问权限，属于进程的性质

保存设置 ID：在执行一个程序时包含了有效用户 ID 和有效组 ID 的副本，由 exec 函数保存。

3.进程每次打开、创建或删除一个文件时，内核就进行文件访问权限测试。简述内核进行的测试过程：

若进程的有效用户 ID 是 0（超级用户），则允许访问。这给予超级用户对整个文件系统进行处理的最充分的自由。

若进程的有效用户 ID 等于文件所有者 ID（也就是进程拥有此文件），那么如果所有者适当的访问权限位被设置，则允许访问；否则拒绝访问。适当的访问权限位指的是，若进程为读而打开该文件，则用户读位应为 1；若进程为写而打开该文件，则用户写位应为 1；若进程将执行该文件，则用户执行位应为 1。

若进程的有效组 ID 或进程的附属组 ID 之一等于文件的组 ID，那么如果组适当的访问权限位被设置，则允许访问；否则拒绝访问。

若其他用户适当的访问权限位被设置，则允许访问；否则拒绝访问。

4.如果对一个目录设置了黏着位（sticky bit），只有对该目录具有权限的用户并且满足什么条件，才能删除或重命名该目录下的文件：

拥有此文件；拥有此目录；是超级用户。

目录/tmp 和/var/tmp 是设置粘着位的典型候选者——任何用户都可以在这两个目录中创建文件，任一用户（用户、组和其他）对这两个目录的权限通常都是读、写和执行，但是用户不应能删除或重命名属于其他人的文件，为此在这两个目录的文件模式中都设置了粘着位。

5.环境变量和命令行参数的区别，怎么使用环境变量：

当执行一个程序时，调用 exec 的进程可将命令行参数（Command-Line Arguments）传递给该新程序。

环境表（Environment List）是一个字符指针数组，其中每个指针包含一个以 null 结束的 C 字符串的地址。

环境变量（Environment Variable）通常指那些在操作系统中指明操作环境的参数。

6.解释一下预编译、编译、汇编、链接的过程：

预编译过程：负责头文件展开，宏替换，条件编译的选择，删除注释等工作。

编译过程：负载将预处理生成的文件，经过词法分析，语法分析，语义分析及优化后生成汇编文件。

汇编：将汇编代码转换为机器可执行指令的过程。

链接：负载根据目标文件及所需的库文件产生最终的可执行文件。链接主要解决了模块间的相互引用的问题，分为地址和空间分配，符号解析和重定位几个步骤。

7.进程的三种状态：

进程有就绪状态、运行状态、阻塞状态 3 种状态。就绪状态(ready)：进程已获得除处理机以外的所需资源，等待分配处理机资源；运行状态(running)：占用处理机资源运行，处于此状态的进程数小于等于 CPU 数；阻塞状态(stuck)： 进程等待某种条件，在条件满足之前无法执行。

8.僵尸进程（zombie process）是什么？如何防止僵尸进程？

在 UNIX 术语中，一个已经终止、但是其父进程尚未对其进行善后处理（获取终止子进程的有关信息、释放它仍占用的资源）的进程被称为僵尸进程。

避免僵尸进程的产生:

在 fork 后调用 wait/waitpid 函数取得子进程退出状态。

调用 fork 两次（第一次调用产生一个子进程，第二次调用 fork 是在第一个子进程中调用，同时将父进程退出（第一个子进程退出），此时的第二个子进程的父进程 id 为 init 进程 id（注意：新版本 Ubuntu 并不是 init 的进程 id））。

在程序中显示忽略 SIGCHLD 信号（子进程退出时会产生一个 SIGCHLD 信号，我们显示忽略此信号即可）。

捕获 SIGCHLD 信号并在捕获程序中调用 wait/waitpid 函数。

9.什么是守护进程？如何写一个守护进程：

守护进程也称为精灵进程，是生存周期较长的一种进程。它们常常在系统自举时启动，在系统关闭时才终止。因为没有控制终端，所以说它们是在后台运行的。

调用 umask 将文件模式创建屏蔽字设置为 0.

调用 fork 生成子进程，使父进程退出。

调用 setsid 生成一个会话。

将当前工作目录改成根目录。

关闭不需要的文件描述符。

某些守护进程打开/dev/null 使其具有文件描述符 0,1 和 2，这样，任何一个试图读标准输入，写标准输出或标准出错的库例程都不会产生任何效果。

10.有人说，管道（pipe）不仅是两个进程间用来传递信息的方法，而且是两个进程间同步或互斥的一种方法。你认为这种说法对不对？为什么？

对。

向管道（共享文件）提供输入的发送进程（即写进程），以字符流形式将大量的数据送入管道；而接收管道输出的接收进程（即读进程），可从管道接收数据，由于发送和接收都是利用管道进行通信的，故称为管道通信。

为了协调双方的通信，管道通信机制必须提供以下三方面的协调能力：① 互斥。一个进程正在对 pipe 进行读/写操作时，另一进程必须等待。② 同步。当写（输入）进程把一定数量的数据写入 pipe 后，便去睡眠等待，直到读（输出）进程取走数据将其唤醒；当读进程读一空 pipe，也应睡眠等待，直至写进程将数据写入管道，才将其唤醒。③ 对方是否存在。只有确定对方已存在时，才能进行管道通信，否则会造成因对方不存在而无限期等待。

11.在网络通信时，服务器端进程对下列函数的执行顺序：

（1）connect；(2)close；（3）reads/writes；（4）socket；（5）bind；（6）accept；（7）listen。

答：Socket()—bind()—listen()—accept()—reads/writes----close()

12.加密口令：（Encrypted Password）

加密口令是经单向加密算法处理过的用户口令副本。为了让他人难以通过试探的方法猜测口令，现在某些系统将加密口令存放在一个称为阴影口令（Shadow password）的文件中，该文件至少包含用户名和加密口令，且该文件不应是一般用户可以读取的。

13.产生信号的条件：

当用户按某些终端键时，引发终端产生信号。

硬件异常产生信号：除数为 0、无效的内存引用等。

进程调用 kill(2)函数可将任意信号发送给另一个进程或进程组。

用户可用 kill(1)命令将信号发送给其他进程。

当检测到某种软件条件已经发生，并应将其通知给有关进程时也产生信号。

在某个信号出现时，可以告诉内核按以下 3 种方式之一进行处理：

忽略此信号。捕捉信号。执行系统默认动作

14.解释 UNIX 系统是如何支持在不同进程间共享打开文件

多个进程打开同一个文件时，每个进程都获得各自的一个文件表项，这样每个进程都有它自己的对该文件的当前偏移量。这里，当不同进程对同一文件进行写操作时，通过一些原子操作和文件记录锁来避免产生冲突的。

15.当程序执行时，其 main 函数是如何被调用的？命令行函数是如何传递给新程序的？

当内核执行一个 C 程序时（使用 exec 函数），在调用 main 前先调用一个特殊的启动例程，可执行文件将此启动例程指定为程序的起始地址。启动例程从内核取得命令行参数和环境变量值，然后为按上述方式调用 main 函数做好安排。当执行一个程序时，调用 exec 的进程可将命令行参数传递给新程序。

16.在 UNIX 系统中，特权以及访问控制，是基于用户 ID 和组 ID 的，当程序需要增加特权，或需要访问当前并不允许访问的资源时，如何更换自己的用户 ID 或组 ID，使得新 ID 具有合适的特权或访问权限？

一般而言，我们总是试图使用最小特权模型，我们的程序应当只具有为完成给定任务所需的最小特权。当需要提升特权的时候，我们通过设置程序文件的设置用户 ID 而得到的额外权限，其他时间进程在运行时只具有普通的权限。

如果一个进程正以特殊权限运行，它又想生成另一个进程执行另一个程序，则它应当直接使用 fork 和 exec,而且在 fork 之后，exec 之前要更改回普通权限。设置用户 ID 和设置组 ID 程序绝不应该调用 system 函数

17.解释存储和查询某个特定线程相关数据的一种机制

这种机制称为线程特定数据，每个线程访问它自己独立的数据拷贝，而不用担心和其它线程的访问的同步。系统为每个进程维护一个称之为 Key 的结构数组这个数组中的每个结构称之为一个线程特定数据元素。key 结构中存储的是索引，pthread 结构中存储的是指针，指向线程中的私有数据，通常是 malloc 函数返回的指针。

当一个线程调用 pthread_key_create 创建一个新的线程特定数据元素时，系统搜索其所在进程的 Key 结构数组，找出其中第一个未使用的元素，并通过 keyptr 返回该元素的键，即就是我们前面说的索引。pthread_key_create 函数的第二个参数 destructor 是一个函数指针，指向一个析构函数，用于线程结束以后的一些后期处理工作，析构函数的参数就是线程特定数据的指针。

18.UNIX 进程间的通信方式：

管道（Pipe）及命名管道（FIFO）：管道可用于具有亲缘关系进程间的通信; 命名管道克服了管道没有名字的限制,除具有管道所具有的功能外，它还允许无亲缘关系进程间的通信。

信号（Signal）：信号是比较复杂的通信方式，用于通知接受进程有某种事件发生，除了用于进程间通信外，进程还可以发送信号给进程本身。

消息队列：消息队列是消息的链接表，包括 Posix 消息队列 system V 消息队列。有足够权限的进程可以向队列中添加消息，被赋予读权限的进程则可以读走队列中的消息。消息队列克服了信号承载信息量少，管道只能承载无格式字节流以及缓冲区大小受限等缺点。

共享存储：使得多个进程可以访问同一块内存空间，是最快的可用 IPC 形式。是针对其他通信机制运行效率较低而设计的。往往与其它通信机制，如信号量结合使用，来达到进程间的同步及互斥。

内存映射（mapped memory）：内存映射允许任何多个进程间通信，每一个使用该机制的进程通过把一个共享的文件映射到自己的进程地址空间来实现它。

信号量（semaphore）：主要作为进程间以及同一进程不同线程之间的同步手段。

套接口（Socket）：更为一般的进程间通信机制，可用于不同机器之间的进程间通信。起初是由 Unix 系统的 BSD 分支开发出来的，但现在一般可以移植到其它类 Unix 系统上：Linux 和 System V 的变种都支持套接字。