守护进程、僵尸进程和孤儿进程

守护进程

指在后台运行的，没有控制终端与之相连的进程。它独立于控制终端，周期性地执行某种任务。
Linux 的大多数服务器就是用守护进程的方式实现的，如 web 服务器进程 http 等

创建守护进程要点：

（1）让程序在后台执行。方法是调用 fork（）产生一个子进程，然后使父进程退出。
（2）调用 setsid（）创建一个新对话期。控制终端、登录会话和进程组通常是从父进程继承下来的，守护进程要摆脱它们，不受它们的影响，方法是调用 setsid（）使进程成为一个会话组长。setsid（）调用成功后，进程成为新的会话组长和进程组长，并与原来的登录会话、进程组和控制终端脱离。
（3）禁止进程重新打开控制终端。经过以上步骤，进程已经成为一个无终端的会话组长，但是它可以重新申请打开一个终端。为了避免这种情况发生，可以通过使进程不再是会话组长来实现。再一次通过 fork（）创建新的子进程，使调用 fork 的进程退出。
（4）关闭不再需要的文件描述符。子进程从父进程继承打开的文件描述符。如不关闭，将会浪费系统资源，造成进程所在的文件系统无法卸下以及引起无法预料的错误。首先获得最高文件描述符值，然后用一个循环程序，关闭 0 到最高文件描述符值的所有文件描述符。
（5）将当前目录更改为根目录。
（6）子进程从父进程继承的文件创建屏蔽字可能会拒绝某些许可权。为防止这一点，使用 unmask（0）将屏蔽字清零。
（7）处理 SIGCHLD 信号。对于服务进程，在请求到来时往往生成子进程处理请求。如果子进程等待父进程捕获状态，则子进程将成为僵尸进程（zombie），从而占用系统资源。如果父进程等待子进程结束，将增加父进程的负担，影响服务器进程的并发性能。在 Linux 下可以简单地将 SIGCHLD 信号的操作设为 SIG_IGN。这样，子进程结束时不会产生僵尸进程。

孤儿进程

如果父进程先退出，子进程还没退出，那么子进程的父进程将变为 init 进程。(注：任何一个进程都必须有父进程)。一个父进程退出，而它的一个或多个子进程还在运行，那么那些子进程将成为孤儿进程。孤儿进程将被 init 进程(进程号为 1)所收养，并由 init 进程对它们完成状态收集工作。

僵尸进程

如果子进程先退出，父进程还没退出，那么子进程必须等到父进程捕获到了子进程的退出状态才真正结束，否则这个时候子进程就成为僵尸进程。
设置僵尸进程的目的是维护子进程的信息，以便父进程在以后某个时候获取。这些信息至少包括进程 ID，进程的终止状态，以及该进程使用的 CPU 时间，所以当终止子进程的父进程调用 wait 或 waitpid 时就可以得到这些信息。如果一个进程终止，而该进程有子进程处于僵尸状态，那么它的所有僵尸子进程的父进程 ID 将被重置为 1（init 进

程)。继承这些子进程的 init 进程将清理它们（也就是说 init 进程将 wait 它们，从而去除它们的僵尸状态)。

如何避免僵尸进程?

- 通过 signal(SIGCHLD, SIG_IGN)通知内核对子子进程的结束不关心，由内核回收。如果不想让父进程挂起，可以在父进程中加入一条语句句：signal(SIGCHLD,SIG_IGN);表示父进程忽略 SIGCHLD 信号，该信号是子进程退出的时候向父进程发送的。
- 父进程调用 wait/waitpid 等函数等待子进程结束，如果尚无子进程退出 wait 会导致父进程阻塞。waitpid 可以通过传递 WNOHANG 使父进程不阻塞立即返回。
- 如果父进程很忙可以用 signal 注册信号处理理函数，在信号处理理函数调用 wait/waitpid 等待子进程退出。
- 通过两次调用 fork。父进程首先调用 fork 创建一个子进程然后 waitpid 等待子进程退出，子进程再 fork 一个孙进程后退出。这样子进程退出后会被父进程等待回收，而对于孙子进程其父进程已经退出所以孙进程成为一个孤儿进程，孤儿进程由 init 进程接管，孙进程结束后，init 会等待回收。

第一种方法忽略 SIGCHLD 信号，这常用于并发服务器器的性能的一个技巧因为并发服务器常常 fork 很多子进程，子进程终结之后需要服务器进程去 wait 清理资源。如果将此信号的处理方式设为忽略，可让内核把僵尸子进程转交给 init 进程去处理，省去了大量僵尸进程占用系统资源。