苏亦凡　　计算机科学与技术学院　　200111229

进程

进程（英语：process），是指计算机中已运行的程序。进程曾经是分时系统的基本运作单位。在面向进程设计的系统（如早期的UNIX，Linux 2.4及更早的版本）中，进程是程序的基本执行实体；在面向线程设计的系统（如当代多数操作系统、Linux 2.6及更新的版本）中，进程本身不是基本运行单位，而是线程的容器。[1]

进程管理

- fork：原型为 `pid_t fork(void)` ，fork函数将当前进程的所有信息（堆栈空间、变量等）进行复制以产生一个新的子进程，子进程的所有信息（除pid）都和父进程相同。fork在子进程中返回0，在父进程中返回子进程的pid。
- exec：有6种。其中execve是内核级调用，原型为 `int execve(const char *pathname, char *const argv[],char *const envp[])` ，作用为将当前进程替换为其他程序。
- wait：原型为 `pid_t wait(int *wstatus)` ，用于使父进程阻塞，直到一个子进程结束或者该进程接收到了一个指定的信号为止。如果该父进程没有子进程或者它的子进程已经结束，则wait函数就会立即返回。

多进程server

在原始的server程序中，由于检测到客户端连接后，系统在响应时需要进行复杂的处理(用sleep模拟)，所以从客户端发起连接到客户端收到连接耗费了相当多的时间。于是如果其他客户端在服务器处理请求时发起连接，需要等待程序处理完上一请求后才能收到响应。

解决该问题的关键是利用fork产生的子进程处理请求，父进程始终保持对客户端的监听，从而同时处理多个请求。

在实际应用中还需要使用 `waitpid` 等回收子进程，避免出现错误未能正确终止的进程持续占用cpu等资源。

1. https://zh.wikipedia.org/wiki/行程 ↩