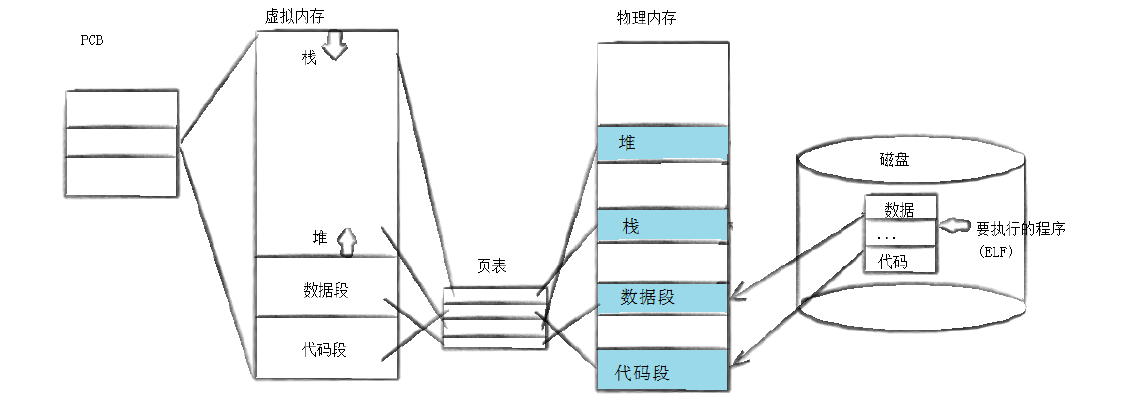
进程程序替换

1. 替换原理

用fork创建子进程后执行的是和父进程相同的程序(但有可能执行不同的代码分支),子进程往往要调用一种exec函数以执行另一个程序。当进程调用一种exec函数时,该进程的用户空间代码和数据完全被新程序替换,从新程序的启动例程开始执行。调用exec并不创建新进程,所以调用exec前后该进程的id并未改变。



2. 替换函数

其实有六种以exec开头的函数,统称exec函数:

#include <unistd.h>`

int execl(const char \*path, const char \*arg, ...);

int execlp(const char \*file, const char \*arg, ...);

int execle(const char \*path, const char \*arg, ...,char \*const envp[]);

int execv(const char \*path, char \*const argv[]);

int execvp(const char \*file, char \*const argv[]);

int execve(const char \*path, char \*const argv[], char \*const envp[]);

3.函数解释

①这些函数如果调用成功则加载新的程序从启动代码开始执行,不再返回。

②如果调用出错则返回-1

③所以exec函数只有出错的返回值而没有成功的返回值。

4.命名理解

这些函数原型看起来很容易混,但只要掌握了规律就很好记。

l(list) : 表示参数采用列表

v(vector) : 参数用数组

p(path) : 有p自动搜索环境变量PATH

e(env) : 表示自己维护环境变量

