

Lab5 文件系统

地理与海洋科学学院 191830173 徐嵩

Exercise

1: 想想为什么我们不使用文件名而使用文件描述符作为文件标识。

答: 文件名的长度是不固定的, 在存储这些文件名的时候, 必须为其分配足够大的空间, 对于文件名较短的文件就会造成空间浪费。使用统一长度的文件描述符, 便于文件的存储和管理。

2: 为什么内核在处理 `exec` 的时候, 不需要对进程描述符表和系统文件打开表进行任何修改。

答: 在执行 `exec` 函数时, 默认情况下, 新程序可以使用在父进程中 `fork` 前打开的文件描述符, 即执行 `exec` 函数时, 并不关闭进程原来打开的文件。如果 `exec()` 因某种原因而调用失败可能还需要使描述符保持打开状态。如果这些描述符已经关闭, 将它们重新打开并指向相同文件的难度很大。

3: 我们可以通过 `which` 指令来找到一个程序在哪里, 比如 `which ls`, 就输出 `ls` 程序的绝对路径 (看下面, 绝对路径是 `/usr/bin/ls`)。那我在 `/home/xyz` 这个目录执行 `ls` 的时候, 为什么输出 `/home/xyz/` 路径下的文件列表, 而不是 `/usr/bin/` 路径下的文件列表呢?

答: 在 `/home/xyz` 目录执行 `ls` 时, 会首先 `fork` 出一个子进程, 并继承当前进程的工作目录。`exec` 执行 `ls` 程序时, 新程序将继承原程序的工作目录, 然后输出 `/home/xyz` 路径下的文件列表。

Task

1: 完成 `irqHandle.c` 里面有关文件的系统调用的内容。

2: 在 `app` 里面完善简易的 `ls` 和 `cat` 函数。

答: 运行截图如下:

```
QEMU
ls /boot/
Name: initrd Inode: 4352

ls /dev/
Name: stdin Inode: 4608
Name: stdout Inode: 4736

ls /usr/

create /usr/test and write alphabets to it
ls /usr/
Name: test Inode: 4992

cat /usr/test
ABCDEFGHIJKLMNOPQRSTUVWXYZ

rm /usr/test
ls /usr/

rmdir /usr/
ls /
Name: boot Inode: 4224
Name: dev Inode: 4480
```

Challenge

1: **system** 函数（自行搜索）通过创建一个子进程来执行命令。但一般情况下, **system** 的结果都是输出到屏幕上，有时候我们需要在程序中对这些输出结果进行处理。一种解决方法是定义一个字符数组，并让 **system** 输出到字符数组中。如何使用重定向和管道来实现这样的效果？

Hint: 可以用 **pipe** 函数（自行搜索）、**read** 函数（你们都会）.....

（在 **Linux** 系统下自由实现，不要受约束）

答：没有实现。