操作系统 · Lab4

计01　容逸朗　2020010869

功能简述

- 本次实验主要完成了目录项和文件的硬链接功能，以及获取文件状态的系统调用。
- 对于 `linkat`，需要从 `ROOT_INODE` 中找到旧文件名对应的 inode，并取得其 id，然后在 `ROOT_INODE` 对应的 disk inode 项增加一个 `DirEntry` 记录便可；
- 对于 `unlinkat`，则需要先找到文件名对应的 inode（及其 id），然后在 root inode 项中寻找同名的 `DirEntry` 并删除之，若删除后 inode_id 对应的目录数为 0 则还需要清除所有为此文件分配的空间；
- 对于 `fstat`，根据 `get_disk_inode_pos` 倒推 `ino`，遍历 root inode 的 `DirEntry` 统计 `nlink`，利用 `is_dir` 判断文件类型即可。

简答题

ch6

1. 在我们的 easy-fs 中，root inode 起着什么作用？如果 root inode 中的内容损坏了，会发生什么？

   Root inode 代表了文件系统中的根目录，是用户查找文件系统内容的起始点。若 root inode 中的内容损坏了，则用户无法通过文件 / 目录的名字定位到对应的文件，由于在我们的 easy-fs 中仅有一个根目录，这使得用户无法使用文件系统。

ch7

1. 举出使用 pipe 的一个实际应用的例子。

   例如要查看伺服器 8080 端口的情况，我们可以执行 `netstat -tunlp | grep 8080` 命令。

   其中 `netstat -tunlp` 显示了所有的 TCP 和 UDP 端口和进程的情况，而 `grep 8080` 则是在文本中查找含有 8080 的行，两者结合即可得到 8080 端口的情况。

2. 如果需要在多个进程间互相通信，则需要为每一对进程建立一个管道，非常繁琐，请设计一个更易用的多进程通信机制。

   可以设计一个菊花状的通信机制，即有一个中心进程只负责数据交换和通信。其余进程只需要和这个进程通信即可，这样便完成了一个简单的多进程通信机制。

Honor Code

1. 在完成本次实验的过程（含此前学习的过程）中，我曾分别与 **以下各位** 就（与本次实验相关的）以下方面做过交流，还在代码中对应的位置以注释形式记录了具体的交流对象及内容：

   无

2. 此外，我也参考了 **以下资料** ，还在代码中对应的位置以注释形式记录了具体的参考来源及内容：

   rCore-Tutorial-Guide 2023 春季学期 的第六节。

   https://stackoverflow.com/q/18429021

3. 我独立完成了本次实验除以上方面之外的所有工作，包括代码与文档。 我清楚地知道，从以上方面获得的信息在一定程度上降低了实验难度，可能会影响起评分。

4. 我从未使用过他人的代码，不管是原封不动地复制，还是经过了某些等价转换。我未曾也不会向他人（含此后各届同学）复制或公开我的实验代码，我有义务妥善保管好它们。 我提交至本实验的评测系统的代码，均无意于破坏或妨碍任何计算机系统的正常运转。 我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按 “-100” 分计。