Lab5 实验报告

南京大学 人工智能学院

191300036 刘晔闻 450008668@qq.com

实验进度：因为 lab5 内容较多，临近期末，任务繁重，我只完成了部分内容，包括：系统调用、open、write 的部分内容。

实验过程：

1. 完成系统调用：（lib/syscall.c）

   包括 write、read、lseek、closs、remove；

   以 write 为例，使用 syscall 函数进行系统调用；

```
int write (int fd, uint8_t *buffer, int size) {
    //TODO: Complete the function 'write' just like the function 'open'.
    int index = 0;
    if (fd == SH_MEM)
        index = *(int *)((void *)&size + 4);
    return syscall(SYS_WRITE, (uint32_t)fd, (uint32_t)buffer, (uint32_t)size, (uint32_t)index, 0);
}
```

2. open：

   首先，添加四种权限：

```
#define O_WRITE 0x01
#define O_READ 0x02
#define O_CREATE 0x04
#define O_DIRECTORY 0x08
```

open为Linux提供的系统原语，其用于打开（或创建）由路径path指定的文件，并返回文件描述符fd，fd为该文件对应的内核数据结构FCB的索引，参数flags用于对该文件的类型、访问控制进行设置

   取异或判断 O_DIRECTORY 是否 set，文件是否使用；

```
if((destInode.type == DIRECTORY_TYPE)^(o_directory>0)){
    sf->eax = -1;
```

   当条件满足时，进行进程切换（利用前几次 lab 提供的代码

块）：

```
for(i=0;i<MAX_DEV_NUM;++i){
    if(dev[i].state&&dev[i].inodeOffset==destInodeOffset){
        dev[i].value--;
        if(dev[i].value<0){
            pcb[current].blocked.next = dev[i].pcb.next;
            pcb[current].blocked.prev = &(dev[i].pcb);
            dev[i].pcb.next = &(pcb[current].blocked);
            (pcb[current].blocked.next)->prev = &(pcb[current].blocked);
            pcb[current].state = STATE_BLOCKED;
            pcb[current].sleepTime = 0xFFFFFFF;
            sf->eax = 0;
            asm volatile("int $0x20");
        }
        sf->eax = i;
        return;
```

之后，O_CREATE 和 O_DIRECTORY 为 set 时，返回-1；

O_DIRECTORY 已经 set 时，返回文件名；

3. write：

write为Linux提供的系统原语，其用于向fd索引的FCB中的文件读写偏移量处开始，向文件中写入从buffer开始的内存中的size个字节，并返回成功写入的字节数，若文件支持seek操作，则同时修改该FCB中的文件读写偏移量

文件描述符 fd 在范围内，且处于运行状态，则调用

syscallwritefile 函数写入；

```
int fd = sf->ecx - MAX_DEV_NUM;
if(fd >=0 && fd < MAX_DEV_NUM+MAX_FILE_NUM){
    if(file[fd].state == 1){
        syscallWriteFile(sf);
    }
}
```

思考题：

1. **为什么使用文件描述符？**

直接使用文件名，为 char 类型，不利于参数传递。

2. 不需要进行更新的 exec

exec 作用是将代码进行替换，进程中的文件并没有改变。

3. cd 程序在哪里

cd 程序为内核命令，which 只是可执行命令，无法得知内核命令的信息。