

# 《Linux 操作系统设计实践》

## 实验四:文件操作



实验环境:Ubuntu16.04



实验内容:

读取文件内容, 经过内容和格式的修改后保存到另一文件。

### 1. 相关 API 学习



open 函数—用于打开和创建文件:

```
int open(const char *pathname, int oflag, ... );
```

返回值: 成功则返回文件描述符, 否则返回 -1



对于 open 函数来说, 第三个参数 (...) 仅当创建新文件时才使用, 用于指定文件的访问权限位 (accesspermission bits)。pathname 是待打开/创建文件的路径名 (如 C:/cpp/a.cpp); oflag 用于指定文件的打开/创建模式。

O\_RDONLY 只读模式

O\_WRONLY 只写模式

O\_RDWR 读写模式



打开/创建文件时, 至少得使用上述三个常量中的一个。以下常量是选用的:

O\_APPEND 每次写操作都写入文件的末尾

O\_CREAT 如果指定文件不存在, 则创建这个文件

O\_EXCL 如果要创建的文件已存在, 则返回 -1, 并且修改 errno 的值

O\_TRUNC 如果文件存在, 并且以只写/读写方式打开, 则清空文件全部内容

O\_NOCTTY 如果路径名指向终端设备, 不要把这个设备用作控制终端。

O\_NONBLOCK 如果路径名指向 FIFO/块文件/字符文件, 则把文件的打开和后继 I/O 设置为非阻塞模式 (nonblocking mode)




以下三个常量同样是选用的, 它们用于同步输入输出

O\_DSYNC 等待物理 I/O 结束后再 write。在不影响读取新写入的数据的前提下, 不等待文件属性更新。

O\_RSYNC      read 等待所有写入同一区域的写操作完成后再进行  
O\_SYNC        等待物理 I/O 结束后再 write，包括更新文件属性的 I/O

 open 返回的文件描述符一定是最小的未被使用的描述符。

 读写文件

```
ssize_t read(int fd, void *buf, size_t count);  
ssize_t write(int fd, const void *buf, size_t count);
```

size\_t 是 unsigned int 表示只能大于等于 0

ssize\_t 是 signed int 可以为正，可以为负

 关闭文件

```
int close(int fd);
```

## 2. 实验思路

运用 open 函数打开文件，通过上次实验使用的 read 函数读取内容后进行修改，加入“liuhy”、“031602523 ”、“ hahahaha ”等字符串后，再利用 write 函数输出到 2.txt 并用 close 函数关闭文件，完成实验要求。

## 3. 本次实验代码

```
#include<stdio.h>  
#include<stdlib.h>  
#include<sys/types.h>  
#include<fcntl.h>  
#include<string.h>  
#include<unistd.h>  
#include<errno.h>  
#include<sys/stat.h>  
#define BUFFER_SIZE 102400  
int main(int argc,char * argv[])  
{  
    int fd=-1;  
    int fw=-1;  
    struct stat statbuff;  
    int size,write_count;  
    char buffer[BUFFER_SIZE];  
    int read_bytes;  
    if((fd=open(argv[1],O_RDWR))>0)  
    {  
        printf("File1 open success!!\n");  
    }  
    if(( fw=open(argv[2],O_WRONLY|O_CREAT,S_IRUSR|S_IWUSR)>0)  
    {  
        printf("File2 establish sucess!!\n");  
    }  
}
```

```

stat(argv[1],&statbuff);
size=statbuff.st_size;
write(fw,"liuhy",5);
read_bytes=read(fd,buffer,size/2);
printf("The size is:%d\n",size);
write_count=write(fw,buffer,size/2);
write(fw,"031602523",9);
read(fd,buffer,size/2);
write(fw,buffer,size/2);
write(fw,"hahahaha",8);
close(fw);
close(fd);
exit(0);
}

```

#### 4. 实验截图

```

liuhy@liuhy-desktop: ~/Desktop
liuhy@liuhy-desktop:~/Desktop$ gcc test.c
liuhy@liuhy-desktop:~/Desktop$ ls
1.txt 2.txt a.out source routing.p4 test.c untitled.c
liuhy@liuhy-desktop:~/Desktop$ cat 1.txt
linux experiment
liuhy@liuhy-desktop:~/Desktop$ cat 2.txt
liuhy@liuhy-desktop:~/Desktop$ ./a.out 1.txt 2.txt
File1 open success!!
File2 establish sucess!!
The size is:17
liuhy@liuhy-desktop:~/Desktop$ cat 1.txt
linux experiment
liuhy@liuhy-desktop:~/Desktop$ cat 2.txt
liuhylinux ex031602523perimenthahahaha

```

2.txt为空

修改内容，输出到2.txt