**《操作系统原理》**

**实 验 报 告 书**

**学 生 姓 名**

**学 号**

**班 级**

**2021 — 2022 学年 第 2 学期**

|  |  |  |  |
| --- | --- | --- | --- |
| **《操作系统原理》实验报告** | | | |
| **实验名称** | **系统调用基础** | **实验序号** | **1** |
| **实验日期** | **2022/6** | **实验人** |  |
| **一、实验题目** | | | |
| 1. **相关原理与知识** 2. **linux基本命令：**   **vim:编辑器：**  **下载vim:** yum -y install vim\*  **vim+文件名创建文件并进入一般模式；**  **进入一般模式，按下i之后才会进入编辑模式。而如果要回到一般模式时， 则必须要按下【Esc】即可退出编辑。输入冒号 ：进入底线命令模式，输入wq保存并退出；**  **：q 直接退出**  **：!q强制退出**  **编译c程序：gcc –o name name.c**  **运行c程序:./name**  **2.系统调用：**  **Linux内核中设置了一组用于实现各种系统功能的子程序，称为系统调用。用户可以通过系统调用命令在自己的应用程序中调用它们。从某种角度来看，系统调用和普通的函数调用非常相似。区别仅仅在令在自己的应用程序中调用它们。从某种角度来看，系统调用和普通的函数调用非常相似。区别仅仅在于，系统调用由操作系统核心提供，运行于核心态；而普通的函数调用由函数库或用户自己提供，运行于用户态。二者在使用方式上也有相似之处。Linux系统的核心部分即是Linux内核，是一系列设备的驱动程序。系统调用是Linux内核提供的功能十分强大的一系列的函数。这些函数是在内核中实现的，它们是应用程序和内核交互的接口，系统调用在Linux系统中发挥着巨大的作用，如果没有系统调用，那么应用程序就失去了内核的支持。**  **3.程序编写**   * + **拷贝文件功能**   + **模仿cp命令的执行效果**     - **错误检查（参数检查，程序逻辑，函数返回值判断等）**     - **只考虑文件，不考虑目录** * **strace**   + **查看所编写程序执行过程中调用的系统调用**   **strace用来判断一个程序执行出错的原因** | | | |
| **三、实验过程**  **1.在centos7 里安装vim**  **2.在终端命令行输入vim.file1.txt，创建file1.txt文件**  **3.进入vim编辑器插入模式将‘hello world hello linux’写入文件中。**  **4.在终端命令行输入vim.file2.txt，创建file1.txt文件**  **5.直接退出，file2.txt为空文件**  **6.用编辑器vim编写c程序，命名为 task.c**  **5.用gcc命令编译c程序，命名为task**  **7.用命令./task来运行c程序task.c**  **8.按照提示输入源文件和目标文件名字。**  **9.用cat file2.txt, cat file1.txt查看目标文件内容和源文件内容，发现源文件file1.txt内容已经写入目标文件file2.txt2中。说明实验成功。**  **10.将编译好的源程序在终端通过strace命令运行，输入的命令如下：**  **Strace -tt -f ./task** | | | |
| 1. **实验结果与分析**   **结果如下：**  **PLG6KDTOBC3L9XX$FW{{O]K**  **GN(JG482~JIDTMPCHO)3V)5**  **8}IAPLRH46GSAK@38QO@(}U**  **分析：**  **本次实验要在linux的虚拟机中完成如下操作，编写程序实现将一个文件的内容复制到另一个目标文件。**  **首先要创建两个文件，其次程序要提示用户输入源文件和目标文件的名称，并且包含了所有必要的错误检查以及保证源文件的存在。在linux里，使用gcc编译一个c程序代码，可以通过gcc -o name name.c来给编译文件命名，然后在命令行输入./name即可运行该c程序。**  **通过strace命令得到在本次实验出用了以下系统调用**  **Execve():执行文件**  **Brk():改变程序间断点的位置**  **Access():检查调用进程是否可对文件执行某种操作**  **Openat():作用类似于open()，用于打开文件**  **Fstat():返回文件信息**  **Mmap():将文件内容映射进内存**  **Close():关闭文件**  **Read(),write():从文件读/向文件写**  **Mprotect():设置内存的访问权限**  **Munmap():解除映射**  **Lseek():改变文件的偏移量（偏移量或指针）**  **Exit\_group():程序的正常结束** | | | |
| 1. **问题总结**   **在一开始使用VIM的时候，不懂得VIM创建文件的格式，没有输入有效的命令。进入操作界面以后要按i进入文本编辑模式.如果进入VIM界面时没有创建文件，退出时会报错，要！q强行退出。程序的第一版本因为BUFFER不够大，没能运行成功，将BUFFER从20改为30后成功。**  **通过这次实验，是我更加了解Linux -些常用指令的操作以及其作用，对于-一个刚开始接触Lniux操作系统的初学者来说非常有用，助于以后能够更进一步学习Linux 操作系统。而且原本对liunx下的编译和调试环境不是很熟悉，但通过这次的实验，让我熟悉了linux下的编译器和调试器的使用。** | | | |
| **六、源代码**  **（源程序）**  **# include <stdio.h>**  **# include <sys/types.h>**  **# include <sys/stat.h>**  **# include <fcntl.h>**  **# include <unistd.h>**  **# define MAX 100**  **int main()**  **{**  **char BUFFER[30];**  **char source[100];**  **char target[100];**  **int n = 0;**  **printf(" input SOURCE filename\n");**  **scanf("%s",source);**  **int fsource = open(source,O\_RDONLY);**  **if(fsource == -1)**  **{**  **printf("unable to open the file\n");**  **return 1;**  **}**  **printf("input TARGET filename\n");**  **scanf("%s",target);**  **int ftarget = open(target,O\_WRONLY);**  **if(ftarget == -1)**  **{**  **printf("error");**  **return 1;**  **}**  **n = read(fsource,BUFFER,MAX);**  **while(n > 0)**  **{**  **printf("%d\n",n);**  **write(ftarget,BUFFER,n);**  **n = read(fsource,BUFFER,MAX);**  **}**  **close(fsource);**  **close(ftarget);**  **return 0;**  **}** | | | |