

编程作业 4--Shell 管道和重定向功能的实现

一. 实验内容

使用 `fork()`, `exec()`, `dup2()`, `pipe()`, `open()` 系统调用完成与下列 shell 命令等价的功能。(提示: 为简化编程, 不需要用 `strtok` 断词, 直接实现能达到下述 shell 命令相同功能的程序即可)

命令如下:

```
grep -v usr</etc/passwd|wc -l>r.txt; cat r.txt
```

二、实验目的

1. 熟悉 Linux 系统中的各类命令, 以及能够通过编程使用系统调用实现相应的功能。
2. 加强对 Shell 管道和重定向功能的理解。

三、实验步骤

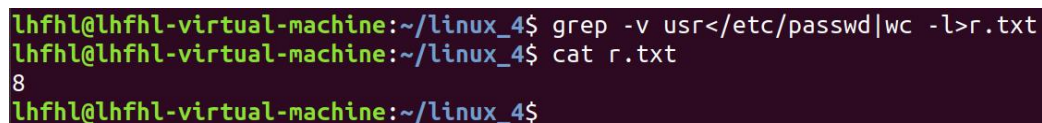
1. 所给命令分析:

`grep -v` 表示该命令的输入为 `/etc/passwd` 文件 (输入重定向), `-v usr` 选项表示从该文件筛选出不包含 `usr` 的行, 并将它们输出到管道。

`wc` 表示处理 `grep` 命令输出到管道的结果, 通过 `wc -l` 命令统计行数, 结果输出到 `r.txt` 文件 (输出重定向)。

`cat` 表示打印 `r.txt` 的内容到标准输出设备上。

2. 查看所给命令的输出结果:



```
lhfh1@lhfh1-virtual-machine:~/linux_4$ grep -v usr</etc/passwd|wc -l>r.txt
lhfh1@lhfh1-virtual-machine:~/linux_4$ cat r.txt
8
lhfh1@lhfh1-virtual-machine:~/linux_4$
```

从上图可以看到所给命令的结果为 8

3. 编写代码

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/wait.h>

int main(void)
{
    int t, fd[2];          //t是wait函数的参数, fd[2]用于作为pipe函数的参数
    pipe(fd);             //建立管道
    if (fork() == 0)       //fork()函数创建子进程, 开始执行grep
    {
        dup2(open("/etc/passwd", O_RDONLY), 0);
        dup2(fd[1], 1);
        close(fd[0]);
        execlp("grep", "grep", "-v", "usr", NULL);
    }
    if (fork() == 0)       //fork()函数创建子进程, 开始执行wc
    {
        int out = open("r.txt", O_RDWR | O_CREAT, S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH);
        dup2(fd[0], 0);
        dup2(out, 1);
        close(fd[1]);
        execlp("wc", "wc", "-l", NULL);
    }
    close(fd[1]);          //关闭管道
    close(fd[0]);          //关闭管道
    wait(&t);              //等待进程销毁
    wait(&t);
    if (fork() == 0)       //fork()函数创建子进程, 执行cat命令
    {
        execlp("cat", "cat", "r.txt", NULL);
    }
    wait(&t);
    return 0;
}
```

4. 编译执行

```
lhfh1@lhfh1-virtual-machine:~/linux_4$ gcc 4.c -o 4
lhfh1@lhfh1-virtual-machine:~/linux_4$ ./4
8
```

结果与原命令执行一致。

四、实验总结

通过本次实验, 提高了我的 shell 脚本的编写能力, 也加强了我对于 shell 管道和重定向的理解, 收获很多。

五、实验代码

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/wait.h>
int main(void)
{
    int t, fd[2]; //t 是 wait 函数的参数, fd[2]用于作为 pipe 函数的参数
    pipe(fd); //建立管道
    if (fork() == 0) //fork()函数创建子进程, 开始执行 grep
    {
        dup2(open("/etc/passwd", O_RDONLY), 0);
        dup2(fd[1], 1);
        close(fd[0]);
        execlp("grep", "grep", "-v", "usr", NULL);
    }
    if (fork() == 0) //fork()函数创建子进程, 开始执行 wc
    {
        int out = open("r.txt", O_RDWR | O_CREAT, S_IRUSR | S_IWUSR | S_IRGRP
| S_IROTH);
        dup2(fd[0], 0);
        dup2(out, 1);
        close(fd[1]);
        execlp("wc", "wc", "-l", NULL);
    }
    close(fd[1]); //关闭管道
    close(fd[0]); //关闭管道
    wait(&t); //等待进程销毁
    wait(&t);
    if (fork() == 0) //fork()函数创建子进程, 执行 cat 命令
    {
        execlp("cat", "cat", "r.txt", NULL);
    }
    wait(&t);
    return 0;
}
```