

# Table of Contents

[Introduction](#Top_of_index_html)

[First Chapter](#Top_of_chapter1_html)

# Table of Contents

1. [Introduction](#Top_of_index_html)
2. [First Chapter](#Top_of_chapter1_html)

# 南京大学 操作系统（2015秋季 教师： 华景煜）实验报告

姓名：吴斌斌 学号131220073

本文档使用Gitbook写作，Gitbook网站对中文pdf支持不好，因此pdf文件的汉字字形有问题。可访问[Gitbook](https://www.gitbook.com/book/binismwu/oslab2015/details)在线查看。

* First Chapter———第二次实验:Linux常用命令及Makefile实验

# Linux 常用命令及 Makefile 实验

姓名：吴斌斌 学号：131220073

# 一、使用 Linux Shell 命令完成以下操作：

### 1. 查看当前登录在系统中的用户列表、系统中的用户总数和系统启动时间。

查看系统用户列表

binism@binism-GE60-2OC-2OD-2OE:~$ who -q

查看系统用户列表

查看系统用户总数

binism@binism-GE60-2OC-2OD-2OE:~$ who -q

查看系统用户总数

查看系统启动时间

binism@binism-GE60-2OC-2OD-2OE:~$ uptime

查看系统启动时间

### 2.将系统文件/etc/profile 复制到主用户目录，并改名为 profile.txt，查看此文件的内容，并对非空行进行编号；重新打开此文件，从 profile 的第 5 行开始显示，每屏幕仅显示 5 行。

将系统文件/etc/profile 复制到主用户目录，并改名为 profile.txt

binism@binism-GE60-2OC-2OD-2OE:~$ cp /etc/profile ~/profile.txt

将系统文件/etc/profile 复制到主用户目录，并改名为 profile.txt

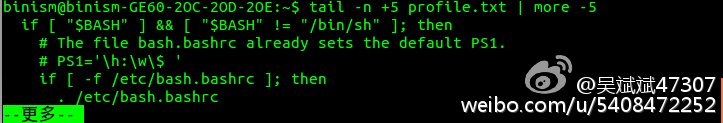
查看此文件的内容，并对非空行进行编号

binism@binism-GE60-2OC-2OD-2OE:~$ cat -b profile.txt



重新打开此文件，从 profile 的第 5 行开始显示，每屏幕仅显示 5 行。

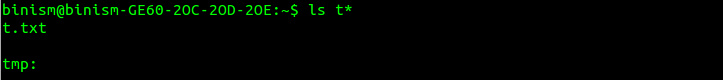
tail -n +5 profile.txt | more -5



### 3. 在主用户目录创建临时目录 tmp，在此目下录，将/etc 目录压缩成 etc.zip 文件，然后解压缩。

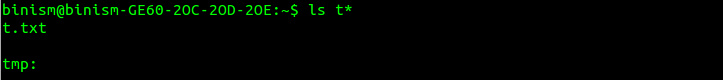
在主用户目录创建临时目录 tmp

binism@binism-GE60-2OC-2OD-2OE:~$ mkdir tmp



在此目下录，将/etc 目录压缩成 etc.zip 文件,然后解压缩。

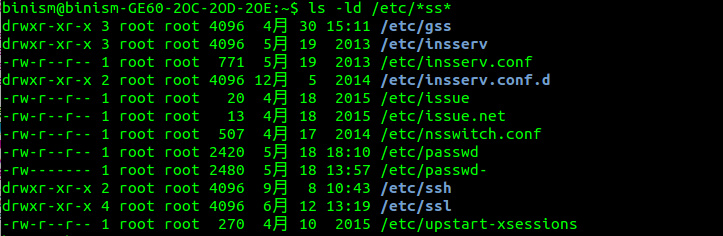
binism@binism-GE60-2OC-2OD-2OE:~$ cd ./tmp
binism@binism-GE60-2OC-2OD-2OE:~/tmp$ tar -jcvf etc.tar.bz2 /etc
binism@binism-GE60-2OC-2OD-2OE:~/tmp$ tar -jxvf etc.tar.bz2



### 4. 查找/etc目录下包含字符串“ss”的文件；复制/etc/passwd 文件到用户的主目录下，搜索这个文件中包含字符串“root”的行，并显示行号。

查找/etc 目录下包含字符串“ss”的文件

binism@binism-GE60-2OC-2OD-2OE:~$ ls -ld /etc/\*ss\*



复制/etc/passwd文件到用户的主目录下，搜索这个文件中包含字符串“root”的行，并显示行号。

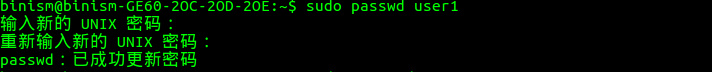
binism@binism-GE60-2OC-2OD-2OE:~$ cp /etc/passwd ~/
binism@binism-GE60-2OC-2OD-2OE:~$ grep -n "root" passwd

复制/etc/passwd 文件到用户的主目录下，搜索这个文件中包含字符串“root”的行，并显示行号。

### 5.创建一个新用户 user1，给该用户设置密码为 LoveLinux，将用户名更改为 user2。创建user3,将 user3 的有效组切换为 admin。切换到 user3，在/home 目录下创建 dir 目录。切换到 user2，查看 user2 是否可以在 dir 目录下创建、删除文件。如果不可以修改这个目录的权限，或者修改这个目录的所有者、所属组，使得用户user2 可以在这个目录下创建、删除文件。

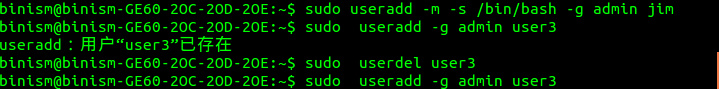
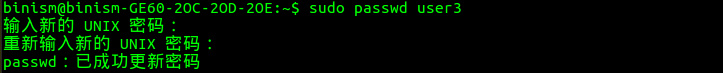
创建一个新用户 user1，给该用户设置密码为 LoveLinux，将用户名更改为 user2。

binism@binism-GE60-2OC-2OD-2OE:~$ sudo useradd user1
binism@binism-GE60-2OC-2OD-2OE:~$ sudo passwd user1
binism@binism-GE60-2OC-2OD-2OE:~$ sudo usermod -l user2 user1

创建一个新用户 user1  更名user1为user2

创建user3,将 user3 的有效组切换为 admin。

binism@binism-GE60-2OC-2OD-2OE:~$ sudo useradd -g admin user3
binism@binism-GE60-2OC-2OD-2OE:~$ sudo passwd user3

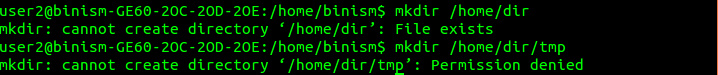
切换到 user3，在/home 目录下创建 dir 目录

binism@binism-GE60-2OC-2OD-2OE:~$ su user3
user3@binism-GE60-2OC-2OD-2OE:/home/binism$ sudo mkdir /home/dir

切换到 user3，在/home 目录下创建 dir 目录

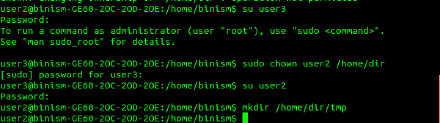
切换到 user2，查看 user2 是否可以在 dir 目录下创建、删除文件。

user3@binism-GE60-2OC-2OD-2OE:/home/binism$ su user2`
user2@binism-GE60-2OC-2OD-2OE:/home/binism$ mkdir /home/dir/tmp



修改这个目录的权限, 或者修改这个目录的所有者、所属组, 使得用户 user2 可以在这个目录下创建、删除文件

user2@binism-GE60-2OC-2OD-2OE:/home/binism$ su user3
user3@binism-GE60-2OC-2OD-2OE:/home/binism$ sudo chown user2 /home/dir
user3@binism-GE60-2OC-2OD-2OE:/home/binism$ su user2
user2@binism-GE60-2OC-2OD-2OE:/home/binism$ mkdir /home/dir/tmp



### 6.完全使用命令下载、安装、运行并卸载 Linux 版本的 QQ。

binism@binism-GE60-2OC-2OD-2OE:~$ wget http://im.qq.com/qq/linux/download.shtml
binism@binism-GE60-2OC-2OD-2OE:~$ sudo rpm –U qq.rpm

### 7. 查看网络适配器的网络设置，将dhcp动态IP的设置方式改为static 静态IP的设置方式；查看当前系统服务端口的监听状态。

查看网络适配器的网络设置

binism@binism-GE60-2OC-2OD-2OE:~$ ifconfig

将dhcp动态IP的设置方式改为static 静态IP的设置方式

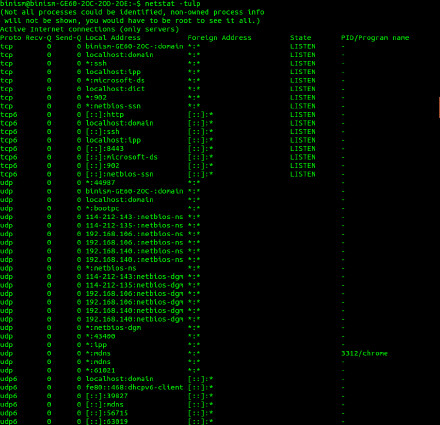
binism@binism-GE60-2OC-2OD-2OE:~$ sudo vim /etc/network/interfaces

/etc/network/interfaces

auto eth0
iface eth0 inet static
address 192.168.1.100
netmask 255.255.255.0
gateway 192.168.1.1
dns-nameservers 8.8.8.8 8.8.4.4

查看当前系统服务端口的监听状态。

binism@binism-GE60-2OC-2OD-2OE:~$ netstat -tulp
binism@binism-GE60-2OC-2OD-2OE:~$ netstat -ap



### 8. 插入 u 盘，在/mnt 下建立一个名叫 USB 的文件夹,然后将 u 盘挂载到/mnt/USB 下，在此目录下创建一个 temp.txt 文件，然后卸载 u 盘。

在/mnt 下建立一个名叫 USB 的文件夹

binism@binism-GE60-2OC-2OD-2OE:~$ sudo mkdir /mnt/USB

将 u 盘挂载到/mnt/USB 下

binism@binism-GE60-2OC-2OD-2OE:~$ sudo mount /dev/sdb1 /mnt/USB

在此目录下创建一个 temp.txt 文件，然后卸载 u 盘

binism@binism-GE60-2OC-2OD-2OE:~$ touch /mnt/USB/temp.txt
binism@binism-GE60-2OC-2OD-2OE:~$ sudo umount /dev/sdb1

# 二、实现从 Windows 到所用 Linux 系统的远程连接

### 1.linux(ubuntu)下的操作：

安装ssh：

binism@binism-GE60-2OC-2OD-2OE:~$ sudo apt-get install ssh
binism@binism-GE60-2OC-2OD-2OE:~$ sudo apt-get install openssh-server

ssh设置：

binism@binism-GE60-2OC-2OD-2OE:~$ sudo /etc/init.d/ssh start
binism@binism-GE60-2OC-2OD-2OE:~$ sudo ufw allow 22

查看linux机器ip地址：

binism@binism-GE60-2OC-2OD-2OE:~$ ifconfig

### 2.windows下的操作：

下载putty，运行，填入目的机器的ip地址，点击“open”。

键入登录用户的用户名和口令。

# 85045cb0.jpg

# 三、Makefile 实验

### 1.请自行查找资料，阅读 Makefile 相关资料，了解 Makefile 的基本概念和基本结构，初步掌握编写简单 Makefile 的方法，了解递归 Make 的编译过程，初步掌握利用 GNU Make编译应用程序的方法。

略

### 2.

(1) 利用文本编辑器（vi）编写一种排序算法sort.c，对一个数组中的整数进行排序;

binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ vim sort.c

sort.c

#include "stdio.h"
#include <stdlib.h>
#define ArrLen 10
void SelectSort(int \*pData,int size)
{
for(int i = 0;i<size-1;++i){
int index = i;
for(int j = i+1;j<size;++j) {
if(pData[j]<pData[index])
index = j;
}
if(index != i){
int temp = pData[i];
pData[i] = pData[index];
pData[index] = temp;
}
}
}
int Array[ArrLen] = {6,5,7,1,3,22,45,2,4,9};
int main(){
printf("Before Sort:\n");
for(int i = 0; i < ArrLen; i++){
printf("%u ",Array[i]);
}
printf("\n");
SelectSort(Array,ArrLen);
printf("After Sort:\n");
for(int j = 0; j < ArrLen; j++)
printf("%u ",Array[j]);
printf("\n");
return 0;
}

(2) 利用 gcc 手动编译、运行该程序;

binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ gcc sort.c -o sort
binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ ./sort
Before Sort:
6 5 7 1 3 22 45 2 4 9
After Sort:
1 2 3 4 5 6 7 9 22 45
binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$

(3) 利用 gdb手动加入断点进行调试，在屏幕上打印断点信息，以及任何一个变量的值。

binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ gdb sort

(gdb) b main
Breakpoint 1 at 0x4006af
(gdb) b SelectSort
Breakpoint 2 at 0x4005ca
(gdb) info b
Num Type Disp Enb Address What
1 breakpoint keep y 0x00000000004006af <main+4>
2 breakpoint keep y 0x00000000004005ca <SelectSort+4>
(gdb) p Array
$1 = 6
(gdb) p Array@10
$5 = {6, 5, 7, 1, 3, 22, 45, 2, 4, 9}
(gdb) r
Starting program: /home/binism/workspace/OS2015/Lab2/sort
Breakpoint 1, 0x00000000004006af in main ()
(gdb) c
Continuing.
Before Sort:
6 5 7 1 3 22 45 2 4 9
Breakpoint 2, 0x00000000004005ca in SelectSort ()
(gdb) watch Array
Hardware watchpoint 3: Array
(gdb) c
Continuing.
Hardware watchpoint 3: Array
Old value = 6
New value = 1
0x000000000040067d in SelectSort ()
(gdb) c
Continuing.
After Sort:
1 2 3 4 5 6 7 9 22 45
[Inferior 1 (process 1525) exited normally]

### 3. 针对 sort.c 利用文本编辑器创建一个 makefile 文件，通过 make 编译次程序，并运行。

Makefile

#set compiler and compile options
CC = gcc
Ld = ld
CFLAGS =-std=c99
#target to compile
CFILES = $(shell find ./ -name "\*.c")
OBJS = $(CFILES:.c=.o)
sort: $(OBJS)
$(CC) -o sort $(OBJS) $(CFLAGS)
clean:
rm -f sort
rm -f $(OBJS) $(OBJS:.o=.d)

### 4.

(见文件夹sort0)

(1) 修改 sort.c，在排序完成后创建一个进程；

(2) 创建完成后父进程打印有序队列的首地址，然后休眠 5 秒钟；

sort.c修改如下：（队列相关定义和实现在Queue.h和queue.c中

#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include "queue.h"
#define ArrLen 10
QueueDef\* MyQueue;
void SelectSort(int \*pData,int size)
{
int i;
for( i = 0;i<size-1;++i){
int index = i;
int j;
for(j = i+1;j<size;++j) {
if(pData[j]<pData[index])
index = j;
}
if(index != i){
int temp = pData[i];
pData[i] = pData[index];
pData[index] = temp;
}
}
pid\_t pid;
int num = 0;
pid = fork();
if(pid == 0) {//子进程
InsertQueue(MyQueue,10);
unsigned int Addr = MyQueue->Head;
printf("In Child Process MyQueue Addr is: %x.\n",Addr);
}
else{//父进程
unsigned int Addr = MyQueue->Head;
printf("In Parent Process MyQueue Addr is： %x.\n Sleep!\n",Addr);
sleep(5);
printf(" Wake!\n");
}
}
int Array[ArrLen] = {6,5,7,1,3,22,45,2,4,9};
int main(){
MyQueue = NewQueue();
int ele = 0;
for(; ele < 10; ele++)
InsertQueue(MyQueue,ele);
printf("Before Sort:\n");
int i;
for(i = 0; i < ArrLen; i++){
printf("%u ",Array[i]);
}
printf("\n");
SelectSort(Array,ArrLen);
printf("After Sort:\n");
int j;
for(j = 0; j < ArrLen; j++)
printf("%u ",Array[j]);
printf("\n");
printf("MyQueue (length %u)is: \n",MyQueue->length);
Node \*ptr = MyQueue->Head;
printf("%u ",ptr->content);
while(ptr->link != NULL){
ptr = ptr->link;
printf("%u ",ptr->content);
}
printf("\n");
return 0;
}

4) 针对 sort.c 和 insert.c 利用文本编辑器创建一个 makefile 文件，通过 make 编译此程 序，并运行。 (Makefile不做变化；)

(5) 分析运行结果，写出你的发现。结果如下：

binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ make
gcc -std=c99 -c -o sort.o sort.c
sort.c: In function ‘SelectSort’:
sort.c:30:29: warning: initialization makes integer from pointer without a cast
unsigned int Addr = MyQueue->Head;
^
sort.c:34:29: warning: initialization makes integer from pointer without a cast
unsigned int Addr = MyQueue->Head;
^
gcc -o sort ./sort.o ./queue.o -std=c99
binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ ./sort
Before Sort:
6 5 7 1 3 22 45 2 4 9
In Parent Process MyQueue Addr is： 1c70030.
Sleep!
In Child Process MyQueue Addr is: 1c70030.
After Sort:
1 2 3 4 5 6 7 9 22 45
MyQueue (length 11)is:
0 1 2 3 4 5 6 7 8 9 10
Wake!
After Sort:
1 2 3 4 5 6 7 9 22 45
MyQueue (length 10)is:
0 1 2 3 4 5 6 7 8 9
binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$

分析：1.父进程打印了队列首地址后，进入睡眠，然后去调用子进程。子进程在父进程睡眠的5秒内，完成了打印队列首地址、输出排序后的数组、输出插入元素的队列工作。父进程等待5秒后重新被唤醒，继续完成睡眠前未完成的工作。

2.子进程、父进程输出的队列首地址相同，这说明：在fork之后，两个进程用的是相同的物理空间（内存区），子进程的代码段、数据段、堆栈都是指向父进程的物理空间，也就是说，两者的虚拟空间不同，但其对应的物理空间是同一个。(错误的结论）。

(见文件夹sort1)为证明分析2是错误的，我修改了sort.c文件，修改后内容如下：

#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include "queue.h"
#define ArrLen 10
QueueDef\* MyQueue;
void SelectSort(int \*pData,int size)
{
int i;
for( i = 0;i<size-1;++i){
int index = i;
int j;
for(j = i+1;j<size;++j) {
if(pData[j]<pData[index])
index = j;
}
if(index != i){
int temp = pData[i];
pData[i] = pData[index];
pData[index] = temp;
}
}
pid\_t pid;
int num = 0;
pid = fork();
if(pid == 0) {//子进程
InsertQueue(MyQueue,10);
unsigned int Addr = MyQueue->Head;
printf("In Child Process MyQueue Addr is: %x.\n",Addr);
printf("C Sleep!\n");
sleep(6);//睡眠时间比父进程长
printf("C Wake!\n");
printf("MyQueue (length %d)in child is: \n",MyQueue->length);
MyQueue->Head->content = -111;//在子进程改变第一个元素的值。
Node \*ptr = MyQueue->Head;
printf("%d ",ptr->content);
while(ptr->link != NULL){
ptr = ptr->link;
printf("%d ",ptr->content);
}
printf("\n");
}
else{//父进程
unsigned int Addr = MyQueue->Head;
printf("In Parent Process MyQueue Addr is： %x.\n Sleep!\n",Addr);
sleep(5);
printf(" Wake!\n");
printf("MyQueue (length %d) in parent is: \n",MyQueue->length);
Node \*ptr = MyQueue->Head;
printf("%d ",ptr->content);
while(ptr->link != NULL){
ptr = ptr->link;
printf("%d ",ptr->content);
}
printf("\n");
}
}
int Array[ArrLen] = {6,5,7,1,3,22,45,2,4,9};
int main(){
MyQueue = NewQueue();
int ele = 0;
for(; ele < 10; ele++)
InsertQueue(MyQueue,ele);
printf("Before Sort:\n");
int i;
for(i = 0; i < ArrLen; i++){
printf("%d ",Array[i]);
}
printf("\n");
SelectSort(Array,ArrLen);
printf("After Sort:\n");
int j;
for(j = 0; j < ArrLen; j++)
printf("%d ",Array[j]);
printf("\n");
return 0;
}

主要修改： 1.设置子进程睡眠时间6秒，超过父进程睡眠时间； 2.在子进程修改了队列第一个元素的值； 3.在父、子进程中输出队列元素的值；

运行结果如下：

binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ ./sort
Before Sort:
6 5 7 1 3 22 45 2 4 9
In Parent Process MyQueue Addr is： 12ac030.
Sleep!
In Child Process MyQueue Addr is: 12ac030.
C Sleep!
Wake!
MyQueue (length 10) in parent is:
0 1 2 3 4 5 6 7 8 9
After Sort:
1 2 3 4 5 6 7 9 22 45
binism@binism-GE60-2OC-2OD-2OE:~/workspace/OS2015/Lab2$ C Wake!
MyQueue (length 11)in child is:
-111 1 2 3 4 5 6 7 8 9 10
After Sort:
1 2 3 4 5 6 7 9 22 45

分析：程序现进入父进程，输出队列首地址后，睡眠5秒，同时调用子进程，在子进程中改变队列第一个元素的值，子进程输出队列首地址后，睡眠6秒，期间，父进程睡眠结束，重新执行，输出了队列元素，可以看到，队列首个元素并未改变。所以，父、子进程的队列内存不是同一块。只是虚拟内存地址相同。

### 5. 阅读 Linux源码中的/Documentation/kbuild/makefiles.txt 文件（网上有中文版），并根据此文档分析并注释/kernel 目录下的 Makefile 文件。

这里注释的Makefile是linux1.0版本kernel下的Makefile

注释见文件夹内文本文件“linux 1.0 kernel Makefile 的注释”