**西安邮电大学**

**网络空间安全学院**

**实验报告**

**（2020/ 2021学年第1学期）**

实验名称： 安全操作系统课内实验

学生姓名： 王璐 c

专 业： 网安2101班

学 号： 26213035（35）

指导教师： 闵祥参

**实验一 Linux基本命令以及命令接口的使用**

**一、实验目的**

1、熟练linux的基本命令

2、熟练运用Linux的在线求助系统（man info help）

3、 使用vi编辑器

4、熟练Shell程序的设计

**二、实验内容**

1，文件与磁盘操作

man/info：man命令是Linux下的帮助指令，通过man指令可以查看Linux中的指令帮助、配置文件帮助和编程帮助等信息。



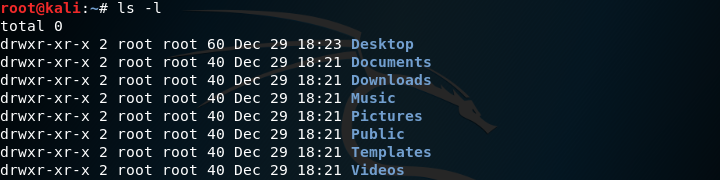
ls(-a -l -h -t)：

**ls命令**用来显示目标列表，在Linux中是使用率较高的命令。ls命令的输出信息可以进行彩色加亮显示，以分区不同类型的文件。

-a:表示查看当前目录下的隐藏文件；



-l:表示查看当前文件的具体信息（包括文件的所有人/所有组，权限，创建时间等)，后面不加具体的文件名表示查看当前目录下的搜友文件的信息；



-h: 与-l参数合用，以人可读取的方式显示文件大小，如1K,2M,3G



-t：用文件和目录的更改时间排序；



cp(-R -f -l)：cp命令用来将一个或多个源文件或者目录复制到指定的目的文件或目录。它可以将单个源文件复制成一个指定文件名的具体的文件或一个已经存在的目录下。cp命令还支持同时复制多个文件，当一次复制多个文件时，目标文件参数必须是一个已经存在的目录，否则将出现错误。

-R:递归处理，将指定目录下的所有文件与子目录一并处理；

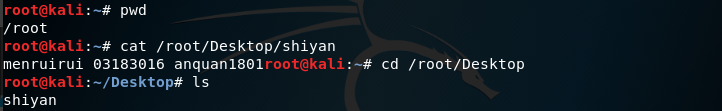
-f:强行复制文件或目录，不论目标文件或目录是否已存在；

-l：对源文件建立硬连接，而非复制文件；

cat ：cat命令连接文件并打印到标准输出设备上，cat经常用来显示文件的内容，类似于下的type命令。



Pwd：pwd命令以绝对路径的方式显示用户当前工作目录。命令将当前目录的全路径名称（从根目录）写入标准输出。全部目录使用/分隔。第一个/表示根目录，最后一个目录是当前目录。执行pwd命令可立刻得知您目前所在的工作目录的绝对路径名称。



rm(-r -f)：rm命令可以删除一个目录中的一个或多个文件或目录，也可以将某个目录及其下属的所有文件及其子目录均删除掉。对于链接文件，只是删除整个链接文件，而原有文件保持不变。

注意：使用rm命令要格外小心。因为一旦删除了一个文件，就无法再恢复它。所以，在删除文件之前，最好再看一下文件的内容，确定是否真要删除。

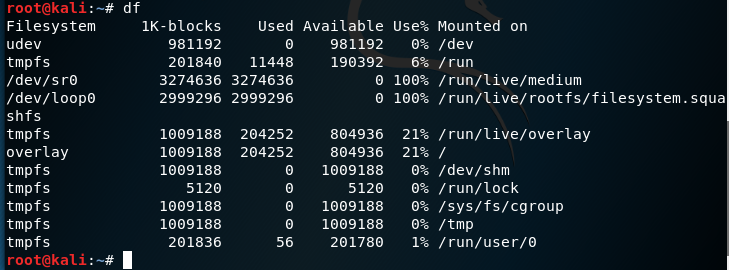


G)Q@[GHPZ5WLTQ)UDF9`}7I

Grep：grep（global search regular expression(RE) and print out the line，全面搜索正则表达式并把行打印出来）是一种强大的文本搜索工具，它能使用正则表达式搜索文本，并把匹配的行打印出来。

J]4P6A5)8PQ6UT1H8[TS$XW

df（-h) ：df命令用于显示磁盘分区上的可使用的磁盘空间。默认显示单位为KB。可以利用该命令来获取硬盘被占用了多少空间，目前还剩下多少空间等信息。



mount /umount clear time ： mount命令用于加载文件系统到指定的加载点。此命令的最常用于挂载cdrom，使我们可以访问cdrom中的数据，因为你将光盘插入cdrom中，Linux并不会自动挂载，必须使用Linux mount命令来手动完成挂载。注意：挂载和卸载都不要在所挂在的目录下面进行，在要挂在的目录下是无法进行挂载的。

2、用户管理

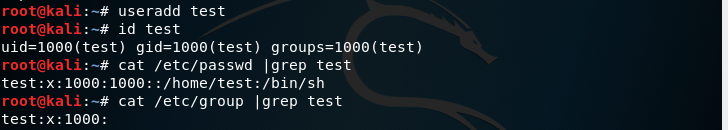
useradd (-g -d -s -G)：useradd命令用于Linux中创建的新的系统用户。useradd可用来建立用户帐号。使用useradd指令所建立的帐号，实际上是保存在/etc/passwd文本文件中。

-g：指定用户所属的群组；

-d：指定用户登入时的启始目录；

-s：指定用户登入后所使用的shell；

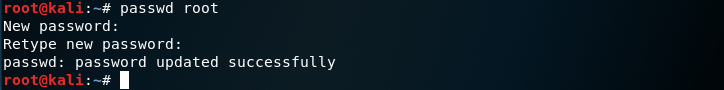
-G：指定用户所属的附加群组；



userdel (-r)：userdel命令用于删除给定的用户，以及与用户相关的文件。若不加选项，则仅删除用户帐号，而不删除相关文件（当你在重新创建这个用户的时候，系统就会报错，显示此用户可能已经存在或者无法创建）。

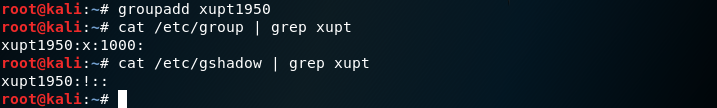
-r：删除用户的同时，删除与用户相关的所有文件。

passwd ： passwd命令用于设置用户的认证信息，包括用户密码、密码过期时间等。系统管理者则能用它管理系统用户的密码。只有管理者可以指定用户名称，一般用户只能变更自己的密码。



groupadd/groupdel ：groupadd命令用于创建/删除一个新的工作组，新工作组的信息将被添加到系统文件中。

gpasswd(-a -d) ：gpasswd命令是Linux下工作组文件/etc/group和/etc/gshadow管理工具。（可能是批量的为某一组或者是一群均有某种特征的用户添加一个密码）



su ：su命令用于切换当前用户身份到其他用户身份，变更时须输入所要变更的用户帐号与密码（切换到root用户的时候可以使用SU 或者是 su - )。由root用户切换到普通用户是不需要输入密码的，由普通用户切换到root用户需要输入root用户的密码。



kernelversion hostname ：修改当前用户所在的主机名，在本系统中可以在配置文件中永久的修改（/etc/hostname)，也可以改变当前的主机名使用shell命令hostname + 主机名。

3、软件安装

rpm（ -i -e -ivh -U -q -qa -qd -ql -qi -qR）tar

rpm命令是RPM软件包的管理工具。rpm原本是Red Hat Linux发行版专门用来管理Linux各项套件的程序，由于它遵循GPL规则且功能强大方便，因而广受欢迎。逐渐受到其他发行版的采用。RPM套件管理方式的出现，让Linux易于安装，升级，间接提升了Linux的适用度。

-i：显示套件的相关信息；

-e<套件档>或--erase<套件档>：删除指定的套件；

-v：显示指令执行过程；

-h或--hash：套件安装时列出标记；

-U<套件档>或--upgrade<套件档>：升级指定的套件档；

q：使用询问模式，当遇到任何问题时，rpm指令会先询问用户；

-a：查询所有套件；

-d：只列出文本文件，本参数需配合"-l"参数使用；

-R：显示套件的关联性信息；

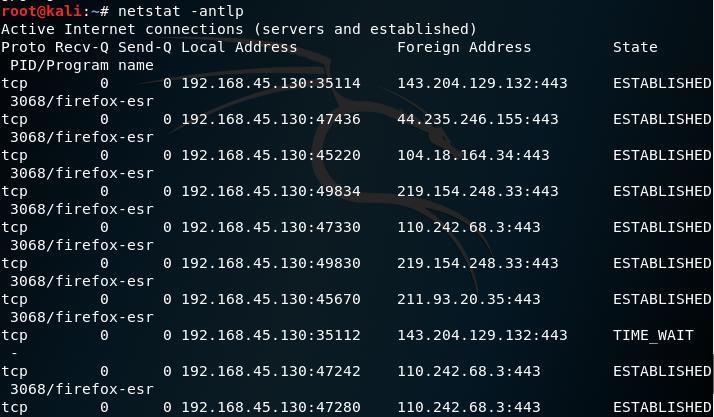
——其实rpm的安装和yum install 的作用是一样的，不过rpm更加简洁好看，可以指定输出的参数，rpm的安装的基础是你的仓库是搭建好的，否则其中需要的依赖包可能找不见。

4，网络命令

ifconfig（up/down）：ifconfig命令被用于配置和显示Linux内核中网络接口的网络参数。用ifconfig命令配置的网卡信息，在网卡重启后机器重启后，配置就不存在。要想将上述的配置信息永远的存的电脑里，那就要修改网卡的配置文件了。



netstat（-t -p -u）：netstat命令用来打印Linux中网络系统的状态信息，可让你得知整个Linux系统的网络情况。

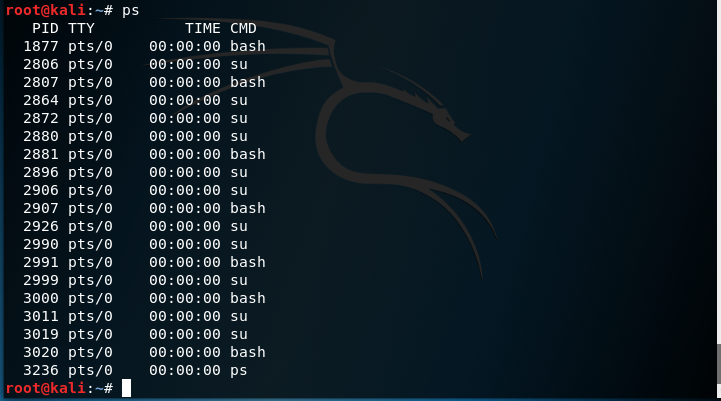


5，进程管理

ps （-a -f -x) ; 查看目前系统中运行的进程。

kill -9 pid ; 利用进程号强行停止某一项进程。

Killall ： 强行禁止与某一项服务有关的进程

****

****

**三、实验心得体会**

通过本次实验，我了解学习了linux系统中一些基本命令，学会了基本的对文件的查看，和对进程的控制，实际操作时有很多问题，多练习理解这些基本命令操作也能进一步了解linux系统。

**实验二 linux的C语言开发**

**一、实验目的**

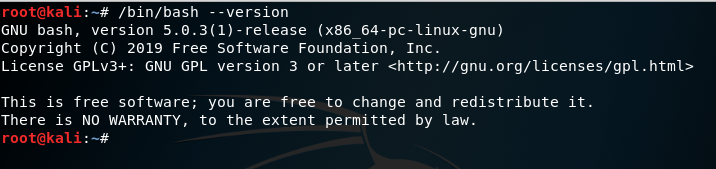
1、熟练掌握Vim的基本使用

2、熟练掌握gcc的使用

**二、实验内容**

（一）shell程序设计

shell编程 属于脚本编程，脚本文件就是指令的集合，GCC是GNU编译系统驱动程序。Linux中的库分两种：静态库和共享库。静态库以.a结尾，也叫归档文件（archive），类似于windows中的.lib文件，他的缺点是同时运行的多个程序使用同个函数库函数时，内存中会有多个该函数及该程序文件的副本，浪费了内存。共享库以.so结尾，类似于windows中的.Dll文件。shell是对linux内核的一种封装，提供了一些函数的接口，shell程序设计就是利用这些函数接口进行编程。一般linux系统的shell装的是bash（Bourne Angain Shell），安装为/bin/sh，可以用$/bin/bash --version命令查看bash版本号：



Shell是一种脚本语言，可以使逻辑判断、循环等语法；可以自定义函数，shell是系统命令的集合，shell脚本可以实习自动化运维，能得得增加运维效率。

A、shell script的结构和运行、调试方法。

1. shell结构

开头需要加#!/bin/bash

以#开头的行作为解释说明

脚本的名字以.sh结尾，用于区分这是一个shell脚本

执行方法有两种：

Chmod +x 1.sh; ./1.sh

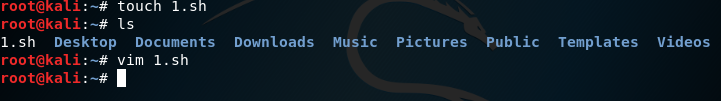
Bash 1.sh

查看脚本执行过程 bash -x 1.sh

查看脚本是否语法错误 bash -n 1.sh

（1）文件头

以#！/bin/bash作为第一行，如果第二行也是一样的语句或者以#开头，就表示解释说明了，这个文件头的作用：指定是由哪一个解释器来操作下面的命令的，通常都是/bin/bash这个解释器。这行代码的目的：如果此脚本在本机上执行，则可以不用写这个第一行，因为本机是可以识别脚本后面一条一条的命令的，但是如果不写这一行，换一台机器后就不一定能识别这个脚本，我们来写如下一个脚本1.sh



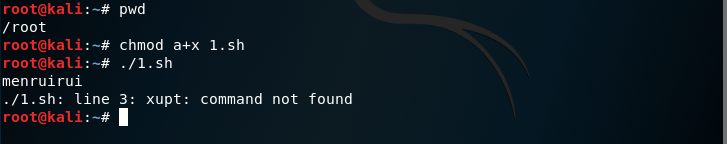


1. shell的执行方法

第一种  # chmod a+x 01.sh

//授予脚本01.sh执行权限，当前所在目录为此脚本所在的目录

# ./01.sh //执行脚本



第二种   # sh 01.sh   或者    # bash 01.sh



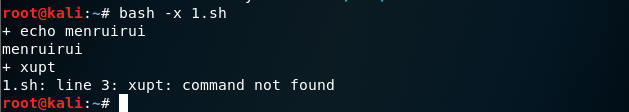
实际上sh 命令和bash命令是同一个文件，如下图，sh软链接到bash



1. 查看脚本执行过程

# bash -x 01.sh

// -x表示查看脚本执行过程，使用-x的时候不能用./的方式来运行



w命令用于显示目前登入系统的用户信息

图中黄色部分的行表示执行的命令，所在行下面的内容表示命令执行的结果。

查看脚本是否语法错误

# bash -n 01.sh

//-n表示查看语法是否有误，如果无误，则没有任何输出，如下图



如果有语法错误，则会提示错误，但是有些错误，比如我定义了一个不存在的命令，此时是不会报错的，但是shell脚本的一些语法规则的错误是可以被检测出来的。

B、shell script的变量声明方法及其输入输出

（1）变量的定义

shell变量中的变量名可以字母、数字、下划线组成，但数字不能作为变量名的第一个字符。通过赋值符等于“=”来定义一个变量的值

myname = 'galley' #字符串类型，不解析任何字符

course = “asdfgh” #双引号内部会解析$和反斜杠特殊字符

sum =100; price = 123

now\_date = `date` #反引号执行系统命令

shell的变量类型只有字符串类型，所以当不用引号引起来的时候，要确保变量的值不包特殊的符号，如空格符、美元、&符号。

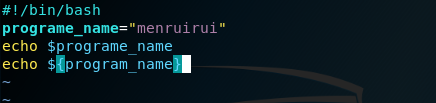
（2）变量的引用

变量的引用，使用美元符号$来加变量名，来引用一个变量的值。

定义变量：programe\_name= "xiuhaiyan"（变量名与等号之间不要有空格）

引用变量：echo $ programe\_name

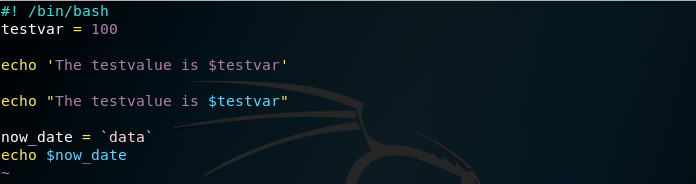
echo $ {programe\_name}

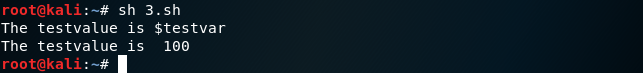






（3）变量值的特殊符号引用





等号后不要有空格

2、shell环境变量、局部变量以及预定义变量

（1）环境变量

环境变量一般指在操作系统中用来指定操作系统运行环境的一些参数，如PATH、PWD、USER、当前使用的SHELL名称等。

（b）常见的环境变量：

PATH：系统路径

HOME：当前用户home目录

HISTSIZE：保存历史命令记录的参数

LOGNAME：当前用户登录名

HOSTNAME：主机名称，若应用程序要用到主机名的话，一般是从这个环境变量中取得

SHELL：当前用户用的是哪一种shell

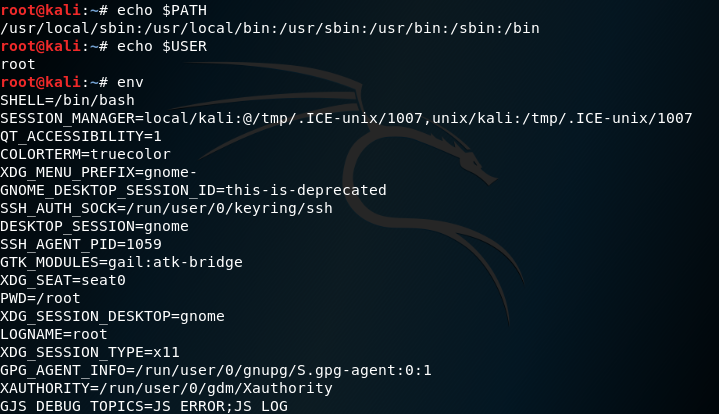
LANG/LANGUAGE：和语言相关的环境变量，使用多种语言的用户可以修改此环境变量

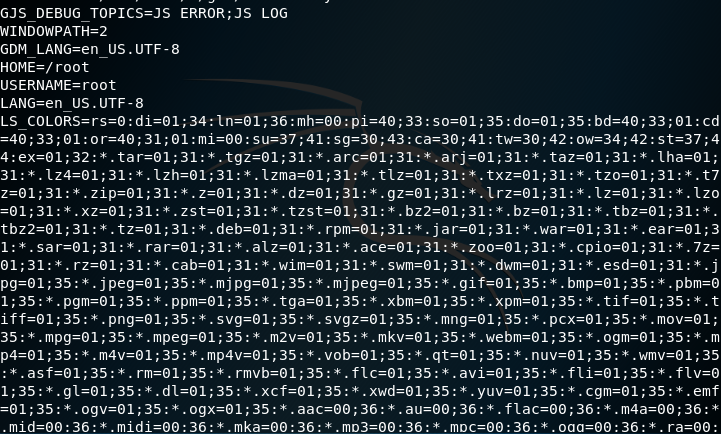
MAIL：当前用户的邮件存放目录

（a）查看环境变量

env：显示所有的环境变量

echo $PATH：显示某个环境变量的值







（2）局部变量

（a）在shell脚本中由用户自定义的变量。

shell脚本中定义的变量作用域是从被定义的地方开始，到shell结束为止，起作用域为本脚本，离开本脚本，变量无效。

（b）在用户登录的shell中定义的变量，重新登录之后该变量消失。

（3）预定义变量

（a）预定义变量和环境变量类似，也是在shell一开始是就定义了的，所不同的是，用户只能根据shell的定义来使用这些变量，不能改变它。所有的预定义变量都是由$和另一个符号组成的，常用的shell预定义变量有：

$# 位置参数的数量 （/bin/sh test.sh 1 2 3）

$\* 所有位置参数的内容

$? 命令执行后返回的状态 #0表示成功 非0表示失败

$$ 当前进程的进程号

$! 后台运行的最后一个进程号

$0 当前执行的进程名



3、脚本的输入和输出

（1）echo 输出

（a）echo命令发送数据到标准的输出设备，数据采用的是字符串的方式，echo命令可以输出一个变量。

（b）echo的两个重要参数：

-e 识别输出内容里面的转义序列

-n 忽略结尾的换行



（2）printf输出

（a）和echo的功能类似，printf也可以输出变量，printf还可以按照格式输出变量，但是需要注意：

a）、printf的默认输出没有换行，换行需要自己加“\n”

b）、printf支持格式化输出

（3）read输入

read命令行读取标准输入设备的下一行，标准输入中的新一行到换行符前的所有字符会被读取，并赋值给对应的变量。







（4）<<按行输入

（a）“<<”符用作每行的命令输入来操作数据库

（b）自动连接ftp服务器来下载数据

C、熟悉shell script的分支结构。编写shell script文件test1.sh，判断用户根目录下是否有名为test的文件或者目录，如果有的话，将其删除。

分支结构：if -then结构

if -then语句的基本格式如下：

if [ 条件判断式 ];then 程序 fi 或者：

if [ 条件判断式 ] then 程序 fi

注意事项： Shell中的if语句使用fi结尾，不同于一般语句的大括号结尾； [ 条件判断式 ]就是使用test命令判断，所以[、]和条件判断式之间必须有空格； then后面跟符合条件执行的程序，可以放在[]之后，用;分割，也可以换行写入，不需要;分割

if -then -else结构

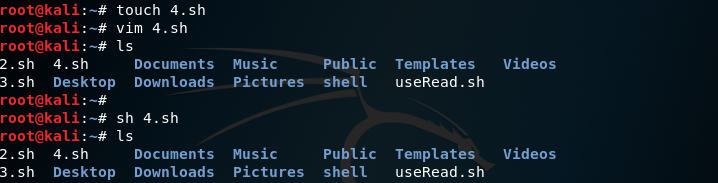
if -then -else语句的基本格式如下：

if [ 条件判断式 ] then

条件成立时，执行的程序

else 条件不成立时，执行的程序 fi





文件表达式  
-e filename 如果 filename存在，则为真  
-d filename 如果 filename为目录，则为真

D、熟悉shell script的循环结构。编写shell script文件test2.sh，实现1到100所有奇数的和并输出。

While循环

while循环是最简单的一种循环，如果条件满足则执行循环里的语句，如果条件不满足则退出循环：

while 条件

do

命令

done

for循环

for ((初始化表达式; 判断表达式; 修改变量表达式))

do

命令

done





**（二）gcc的使用**

**gcc命令**使用GNU推出的基于C/C++的编译器，是开放源代码领域应用最广泛的编译器，具有功能强大，编译代码支持性能优化等特点。现在很多程序员都应用GCC，怎样才能更好的应用GCC。目前，GCC可以用来编译C/C++、FORTRAN、JAVA、OBJC、ADA等语言的程序，可根据需要选择安装支持的语言。

-o：指定生成的输出文件；

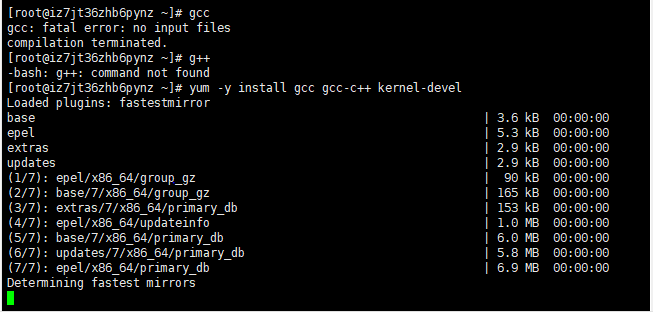
-E：仅执行编译预处理；

-S：将C代码转换为汇编代码；

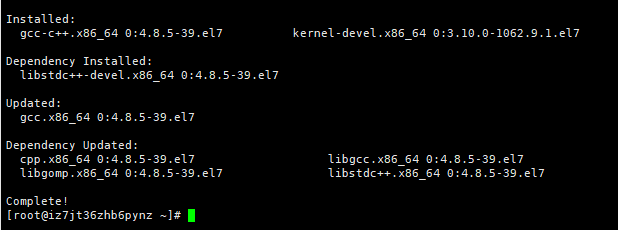
-c：仅执行编译操作，不进行连接操作。

先安装gcc

输入命令# yum -y install gcc gcc-c++ kernerl -devel



安装完成



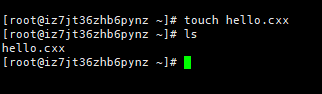
输入命令# gcc -v查看版本号



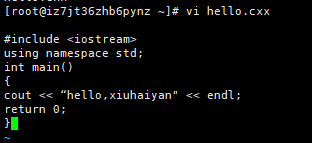
gcc安装成功

1. 利用touch命令创建一个c++文件

输入命令touch hello.cxx

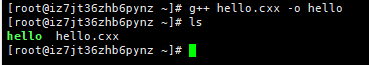


2、编写hello.cxx程序：



3、编译命令:

g++ hello.cxx -o hello



执行结果

输入命令#./hello



3、静态库的建立和使用

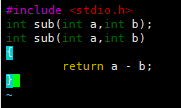
Linux 下使用 ar 工具（Windows 下vs使用lib.exe，将目标文件压缩到一起，并且对其进行编号和索引，以便于查找和检索）

编写程序：main函数所在的文件名字为test3.c，程序功能为从键盘输入两个数求其和（调用add.c中的函数）和差（调用sub.c中的函数）。

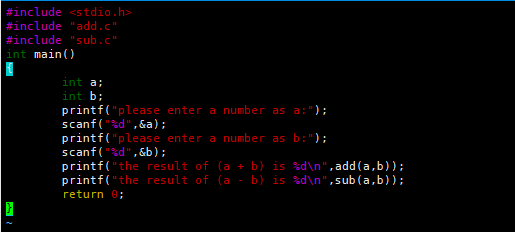
1. add.c函数



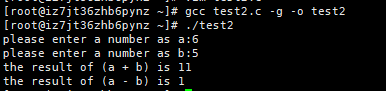
1. sub.c函数



(3) test2.c

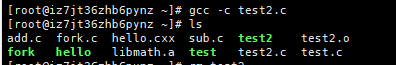


(4)编译、执行



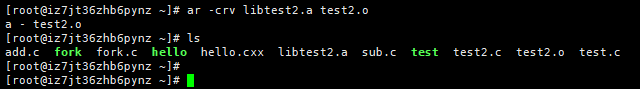
（5）将源文件编译成目标文件.o

输入命令 # gcc -c test2.c生成test2.o



(6) 然后用 ar 工具将目标文件打包成 .a 静态库文件

ar -crv libtest2.a test2　　执行此命令后，便会生成 libmath.a 静态库



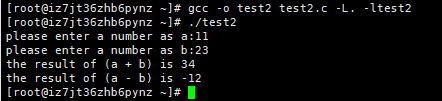
(7) 使用该静态库

如何使用它内部的函数呢？只需要在使用到这些公用函数的源程序中包含这些公用函数的原型声明，然后在用gcc命令生成目标文件时指明静态库名，gcc将会从静态库中将公用函数连接到目标文件中。注意，gcc会在静态库名前加上前缀lib，然后追加扩展名.a得到的静态库文件名来查找静态库文件。

输入命令 # gcc -o test2 test2.c -L. -ltest2

-L：指定该静态库的路径

-l：指定链接时需要的库（编译器在查找链接库时，有隐含的命名规则，即在给出的名字前面加上 lib ，在后面加上 .a 或 .so 来确定库的名称，所以，我们只需要在-l后面加上 该静态库的名字就行）



1. 共享库的建立和使用

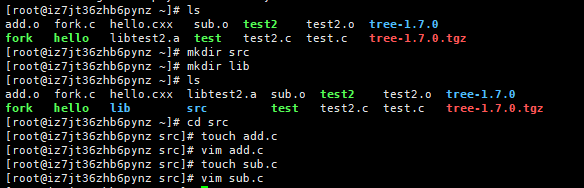
共享库与静态库的区别在于，共享库只是在调用模块中嵌入调用代码的在库的相对位置的地址，当执行程序时，共享库会和程序一起加载到内存中，当执行到调用共享库中代码的指令时跳转到共享库中执行，执行完毕后在跳转回来。

相比于静态库占用空间小，方便更新（共享库发生变化后，程序不需要再次编译），但相比于静态库执行效率略低。

共享库以.so结尾，Linux下需要以lib开头



（1）先创建一个src和lib文件夹，src下存放源代码,，lib下存放共享库



(2)编译产生.o文件

输入命令：

#gcc -c -fPIC add.c

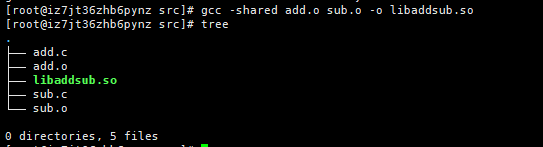
#gcc -c -fPIC sub.c



1. 创建共享库

输入命令：

# gcc -shared -Wl,-soname,libmycal.so.1,-o libmycal.so.1.10 add.o sub.o



将生成的共享库copy到上级的lib下，并生成linkname：

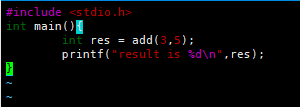
输入命令：

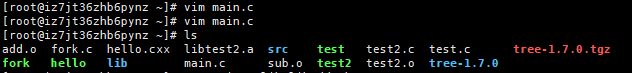
#cp libaddsub.so ../lib #copy

#ln -s libaddsub.so libaddsub1.so #创建link，linkname



(4)回到父目录，创建main.c作为测试程序



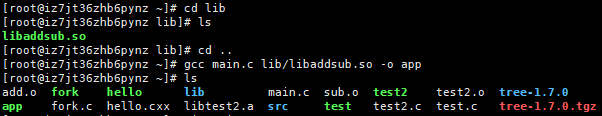


1. 编译生成可执行程序

输入命令：

# gcc main.c lib/libaddsub.so -o app

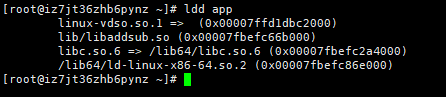
此时已经编译成功，生成app



运行



使用ldd查看依赖，发现全部满足



**三、实验心得体会**

通过本次实验我学会了在Linux中进行shell程序设计的，vim的使用，变量的使用，shell的执行。用gcc进行编程，编译执行、文件。对网络安全来说，Linux的熟练使用是尤为重要的，今后我会更多地使用Linux系统进行学习。

**实验三 linux多进程设计**

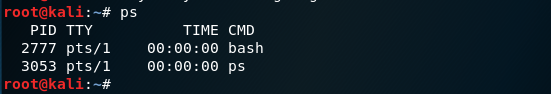
**一、实验目的**

1.了解进程与程序的区别，加深对进程概念的理解；

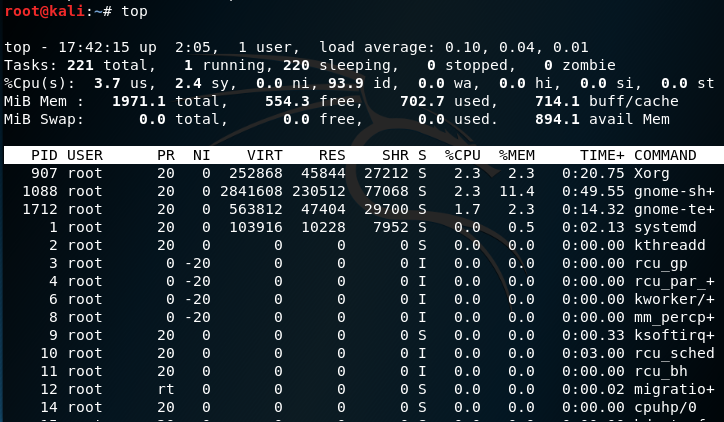
2.了解fork( )系统调用的返回值，掌握用fork()创建进程的方法；

**二、实验内容**

（1）利用ps命令查看当前启动的进程，其中PID代表进程号



（2）top命令：查看系统当前进程的情况。#top



显示信息中第一行依次是当前时间、系统启动时间、当前登录系统的用户数和系统平均负载信息，第二行依次是进程总数、休眠进程数、运行进程数、僵死的进程数和停止的进程数。第三行的CPU状态依次为用户进程、系统进程、优先进程、输入/出等待进程和空闲进程占用的百分比数。第四到第五行的内存使用情况依次为平均可用内存、已用内存、空闲内存、共享内存和缓冲内存等。第六行交换空间使用情况依次为平均可用交换容量、已使用量、空闲数量和高速缓存数量。第七行以后为各个进程的部分状态字段信息列表。

（3）进程的启动与中止

手工启动：

前台启动top进程：top

后台启动top进程：top &，在转为后台执行时会返回一个进程ID，利用这个进程ID可以查看进程的执行情况。



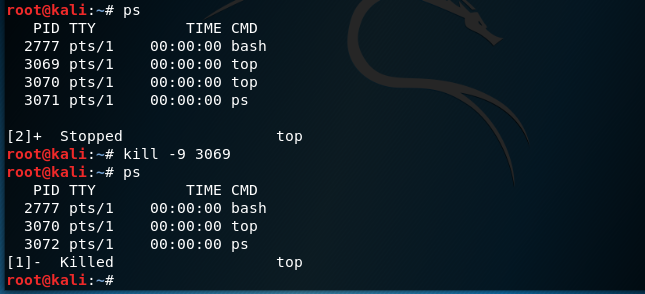
现在我在后台启动了两个top进程，PID号分别是14471和14473，再ps查看一下发现这两个都添加上了

（4）kill命令

功能：终止进程，超级用户可终止所有的进程，普通用户只能终止自己启动的进程。

格式：kill -9 PID

实例：首先启动一个top进程，产生进程ID号为14471的进程，然后利用ps命令查看启动项的进程有哪些，然后使用kill命令杀死进程ID号为14471的进程。操作如下图所示：



Kill 14471和14473进程后ps查看一下发现没有了。

1、用vi输入下面的程序，编译、调试、输出结果

#include <sys/types.h>

#include <unistd.h>

int main()

{

pid\_t child\_pid;

/\*fork函数

\*作用：系统调用，产生一个子进程,调用一次返回两次

\*返回值：pid\_t类型，在头文件中定义，通常为整形

\* 0 子进程返回

\* ID 父进程返回的子进程ID

\*/

child\_pid=fork();

if(child\_pid==0)

{

int i=0;

for(;i<5;i++)

{

printf("child\n");

sleep(1);//wait one second

}

}

else

{

printf("child's pid is %d",child\_pid);

}

int j=0;

for(;j<5;j++)

{

printf("child\n");

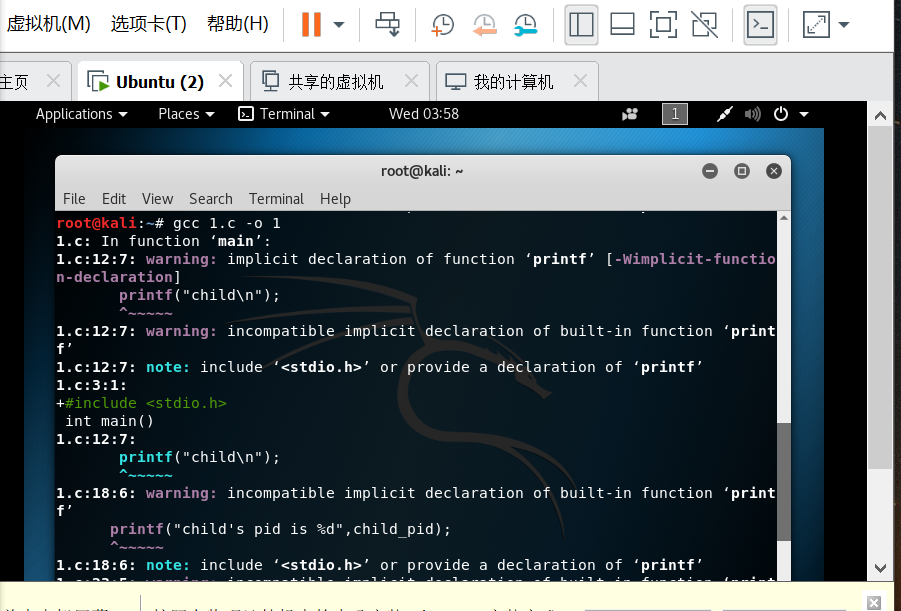
sleep(1);

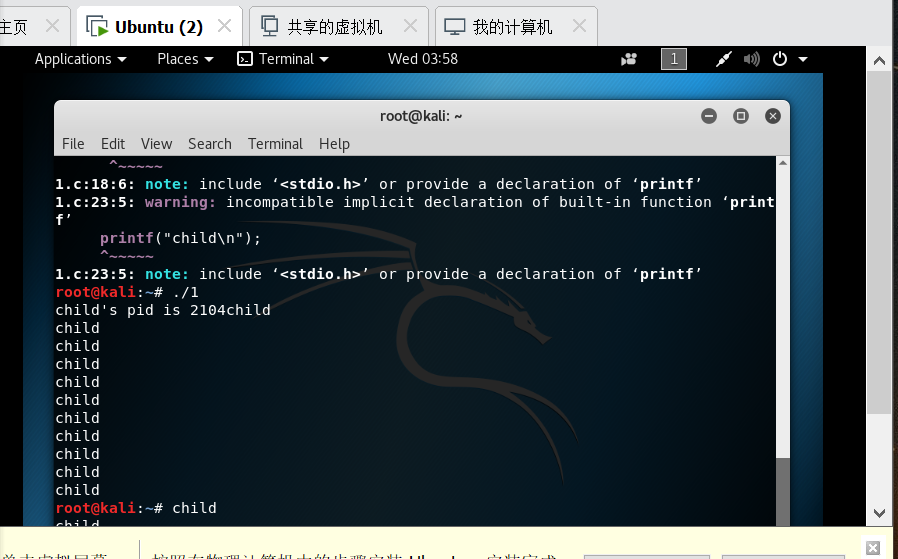
}

return 0;

}

实验结果：





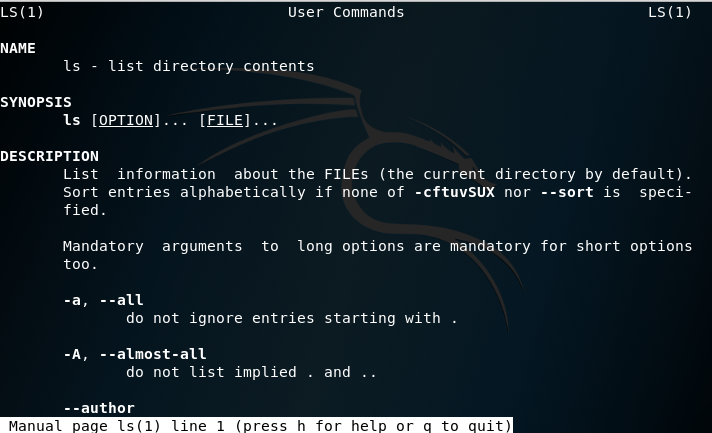
2、用man察看fork、sleep的用法

man 命令简介  
– 在linux下使用man命令相当于阅读window或者Android的文档  
• man 系统  
– man本身就是linux命令，可以使用man命令查看man的说明文档

每个命令行实用程序、每个系统调用、编程库中每个可调用的函数和  
每种文件格式都包含一个手册页  
– 手册的分页是有一定的规律的  
– 分页为man1、man2……man8

1. man1  
   – 一般命令。常见的linux命令，例如ls，cd，cat等等

#man ls



1. man 2

用来放内核提供的系统调用或者函数。例如man 2 fork等

1. man 3  
   – C库函数。

（4）man 4

– 特殊文件，例如设备和驱动程序

（5） man 5

– 文件格式。包括完全使用文本配置文件定制系统的操作，大量的配置文

件，网络服务列表，可用的shell列表等等

（6）man 6

– 游戏和屏幕保护程序。

（7）man 7

– 杂类文件。

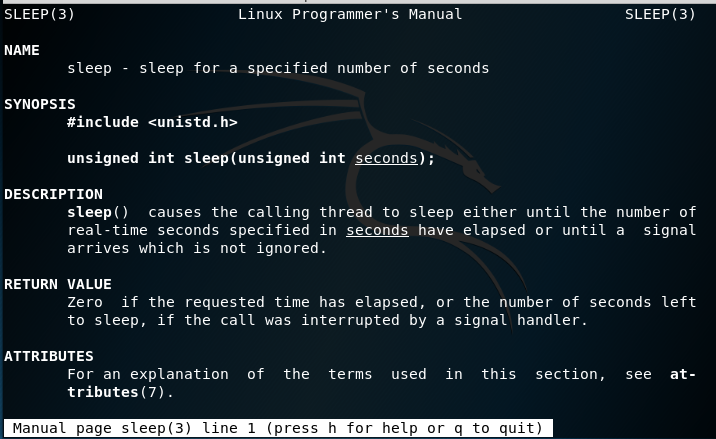
（8）man 8

– 系统管理命令，超级用户可能需要用到它们。

这时再输入命令#man 2 fork



输入命令# man 3 sleep



手册页的框架

• 各种命令选项差异很大，但是手册页的内容是有规律可以掌握的

• 最简单的手册包含5个部分：名称、大纲、描述、一些事例和其它相关资料的引用

– name（名称）给出命令、函数或文件格式的名称，以及对软件作用的单行的准确描述

– synopsis（大纲）简要描述如何使用这个软件

– description（描述）讨论特性、使用方法和命令行上可以使用的所有选项

– examples（示例）部分给出实用程序的一般用途、常见的特殊情况和解释

– See Also（参见）提供相关资料的引用，比如其他相关命令、重要的系统文件、行业标准规范等等。

– 其它特殊部分

**实验任务：**

　　编写一C语言程序，实现在程序运行时通过系统调用fork( )创建两个子进程，使父、子三进程并发执行，父亲进程执行时屏幕显示“I am father”，儿子进程执行时屏幕显示“I am son”，女儿进程执行时屏幕显示“I am daughter”。

先新建一个.c文件



通过vim编写代码

#include <stdio.h>

#include <sys/types.h>

#include <unistd.h>

int main()

{

printf("\nI am fathter !\n");

pid\_t son\_pid;

pid\_t daughter\_pid;

son\_pid=fork();

if(son\_pid != 0)

{

printf("I am son !\n");

printf("my PID is %d\n",son\_pid);

}

daughter\_pid=fork();

if(daughter\_pid != 0)

{

printf("I am daughter !\n");

printf("my PID is %d\n",daughter\_pid);

}

}

通过gcc编译



执行实验运行结果



**三、实验心得体会**

通过本次实验我了解了进程与程序的区别，学会了创建和停止进程，学会了fork（）的用法，了解了父、子进程的创建方式和运行关系，通过学习fork函数调用, 了解了linux下如何创建进程, 和进程之间的关系, 初步认识到进程之间如何进行同步。进一步熟悉了Linux环境下的编程。

**实验四 Linux文件操作**

**一、实验目的**

1、熟练掌握文件的基本操作

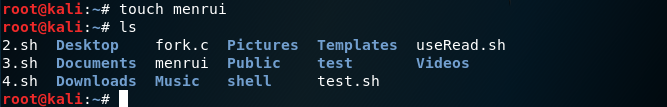
2、熟练掌握文件的权限操作

3、熟练掌握进程的文件操作

**二、实验内容**

**1、常见的文件操作命令**

**Touch：touch命令**有两个功能：一是用于把已存在文件的时间标签更新为系统当前的时间（默认方式），它们的数据将原封不动地保留下来；二是用来创建新的空文件。



chgrp命令：**chgrp命令**用来改变文件或目录所属的用户组。该命令用来改变指定文件所属的用户组。其中，组名可以是用户组的[id](http://man.linuxde.net/id" \t "http://man.linuxde.net/_blank" \o "id命令)，也可以是用户组的组名。文件名可以 是由空格分开的要改变属组的文件列表，也可以是由通配符描述的文件集合。如果用户不是该文件的文件主或超级用户(root)，则不能改变该文件的组。



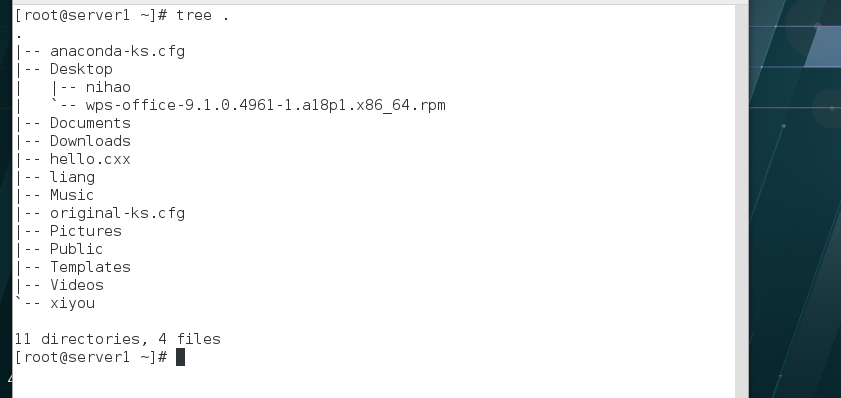
**2、常见的目录操作命令**

install命令：

**install命令**的作用是安装或升级软件或备份数据，它的使用权限是所有用户。install命令和[cp](http://man.linuxde.net/cp" \t "http://man.linuxde.net/_blank" \o "cp命令)命令类似，都可以将文件/目录拷贝到指定的地点。但是，install允许你控制目标文件的属性。install通常用于程序的makefile，使用它来将程序拷贝到目标（安装）目录。

tree命令：

**tree命令**以树状图列出目录的内容



popd命令：

**popd命令**用于删除目录栈中的记录；如果popd命令不加任何参数，则会先删除目录栈最上面的记录，然后切换到删除过后的目录栈中的最上面的目录。

pushd命令：

**pushd命令**是将目录加入命令堆叠中。如果指令没有指定目录名称，则会将当前的工作目录置入目录堆叠的最顶端。置入目录如果没有指定堆叠的位置，也会置入目录堆叠的最顶端，同时工作目录会自动切换到目录堆叠最顶端的目录去。

dirs命令：

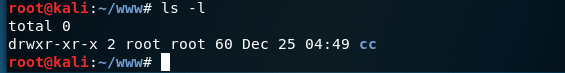
**dirs命令**显示当前目录栈中的所有记录（不带参数的dirs命令显示当前目录栈中的记录）。dirs始终显示当然目录, 再是堆栈中的内容；即使目录堆栈为空, dirs命令仍然只显示当然目录。

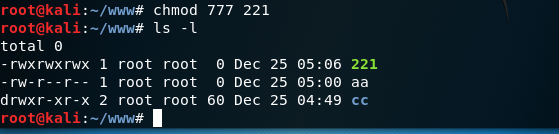
mv：**mv命令**用来对文件或目录重新命名，或者将文件从一个目录移到另一个目录中。source表示源文件或目录，target表示目标文件或目录。如果将一个文件移到一个已经存在的目标文件中，则目标文件的内容将被覆盖。

****

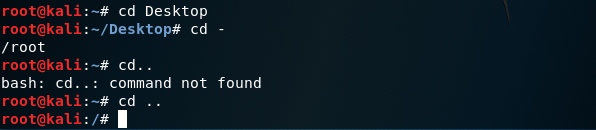
chmod [-R] user[:group] file

**chmod命令**用来变更文件或目录的权限。在UNIX系统家族里，文件或目录权限的控制分别以读取、写入、执行3种一般权限来区分，另有3种特殊权限可供运用。用户可以使用chmod指令去变更文件与目录的权限，设置方式采用文字或数字代号皆可。符号连接的权限无法变更，如果用户对符号连接修改权限，其改变会作用在被连接的原始文件。

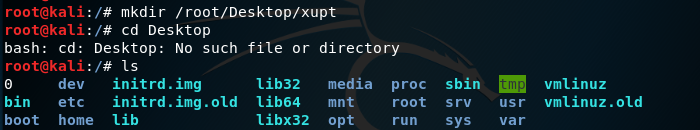




cd ：**cd命令**用来切换工作目录至[dirname](http://man.linuxde.net/dirname" \t "http://man.linuxde.net/_blank" \o "dirname命令)。 其中dirName表示法可为绝对路径或相对路径。若目录名称省略，则变换至使用者的home directory(也就是刚[login](http://man.linuxde.net/login" \t "http://man.linuxde.net/_blank" \o "login命令)时所在的目录)。另外，~也表示为home directory的意思，.则是表示目前所在的目录，..则表示目前目录位置的上一层目录。



Mkdir：**mkdir命令**用来创建目录。该命令创建由[dirname](http://man.linuxde.net/dirname" \t "http://man.linuxde.net/_blank" \o "dirname命令)命名的目录。如果在目录名的前面没有加任何路径名，则在当前目录下创建由dirname指定的目录；如果给出了一个已经存在的路径，将会在该目录下创建一个指定的目录。在创建目录时，应保证新建的目录与它所在目录下的文件没有重名。



**三、实验心得体会**

通过本次实验，我学习了Linux系统中关于文件的基本命令，了解了Linux中文件目录的创建，文件移动复制以及文件权限操作等等，这些都是最基本的操作命令，但是要熟练掌握还是要多加运用练习，另外，实际操作过程中一定要细心。