高性能集群计算机

使用说明书

版本 1.0.8

高性能计算研究组编

2008 年 3 月 12 日

高性能计算集群使用说明

1. 集群系统概述

本集群系统由 14 台 IBM HS21 刀片服务器和 2 台 x3650 服务器组成计算和管理节点，网络连接采用千兆以太网和 infiniband 2.5G 网连接各个节点。每个节点配置 4GB 内存，cpu 为 intel xeon 5150 2.66GMhz 主频，每个 14 个刀片服务器计算节点配置 73GB 硬盘，2 台 x3650 分别配置 4*300GB 硬盘作为主存储和管理节点，两台图形工作站做为前端机，可以进行科学数据可视化，系统计算速度为 4500 亿次每秒以上。

集群系统设置一个登录节点，其外网 ip 地址为 w245.shu.edu.cn ，内网的 ip 地址为 192.168.10.16，内网机器名为 h16.

机器系统每个节点都有两个内部 ip 地址分别对于千兆以太网 ip 地址（192.168.10.*）和 infiniband 网 ip 地址（192.168.9.*），ip 地址分配方式为：

192.168.10.1----192.168.10.16 地址依次对应主机名为 h1----h15

192.168.9.1----192.168.9.16 地址依次对应主机名为 i1----i15

IP 地址	对应主机名	IP 地址	对应主机名
192.168.10.1	h1	192.168.9.1	i1
192.168.10.2	h2	192.168.9.2	i2
192.168.10.3	h3	192.168.9.3	i3
192.168.10.4	h4	192.168.9.4	i4
192.168.10.5	h5	192.168.9.5	i5
192.168.10.6	h6	192.168.9.6	i6
192.168.10.7	h7	192.168.9.7	i7
192.168.10.8	h8	192.168.9.8	i8
192.168.10.9	h9	192.168.9.9	i9
192.168.10.10	h10	192.168.9.10	i10
192.168.10.11	h11	192.168.9.11	i11
192.168.10.12	h12	192.168.9.12	i12
192.168.10.13	h13	192.168.9.13	i13
192.168.10.14	h14	192.168.9.14	i14
192.168.10.15	h15	192.168.9.15	i15
192.168.10.16	h16	192.168.9.16	i16

采用 TCP/IP 协议进行通信时，以 h 开头的主机名对应 ip 地址通信时，通信采用千兆以太网络，采用以 i 字母开头的主机名对应 ip 地址进行通信时，通信采用 infiniband 网络。MPI 利用网络方式与 MPI 具体实现有关。

图 1 主机

图 1 图形工作站

2. 使用方法

1.登录方法

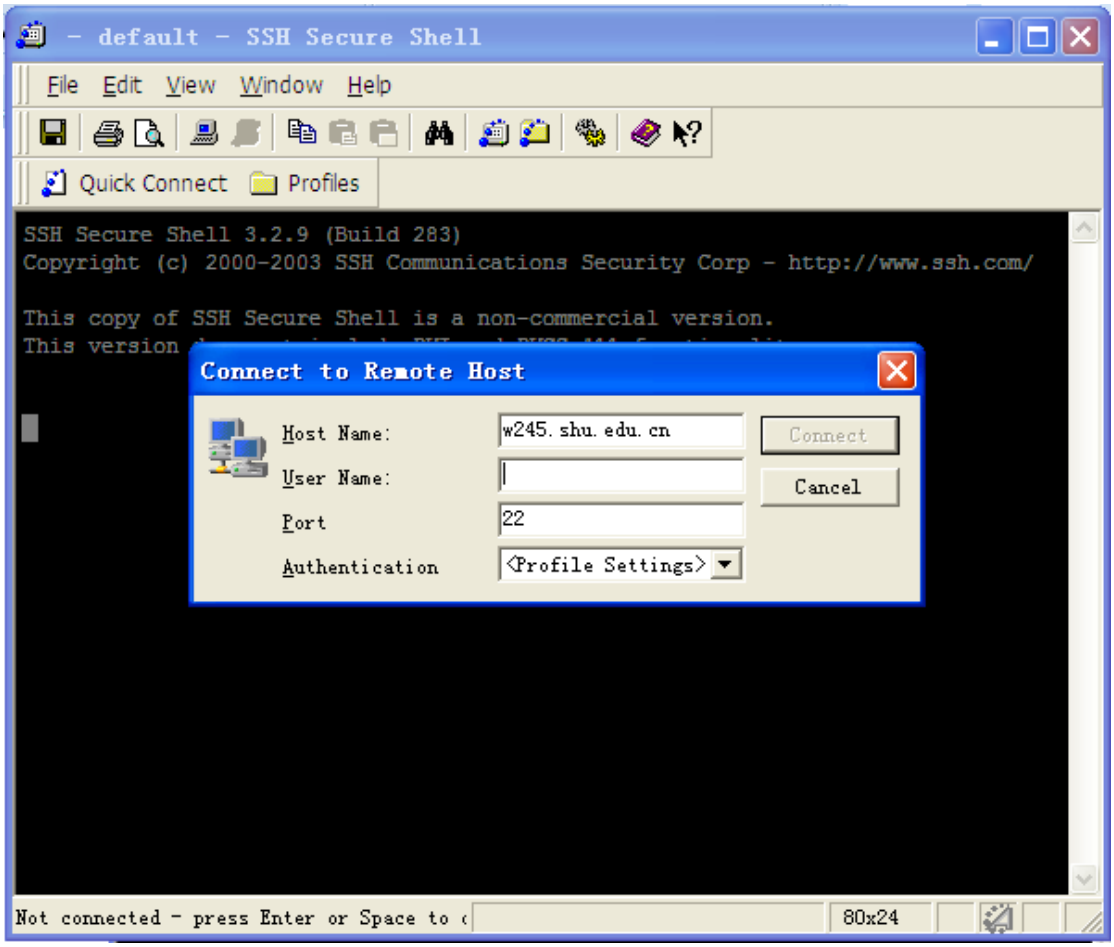
本集群系统的登录方式采用 ssh 登录，登录节点的 **IP 地址：w245.shu.edu.cn ，端口 22**。用户在 **linux** 操作系统下在终端中直接输入 **ssh w245.shu.edu.cn –l username**，出现密码提示以后输入密码即可以登录系统。

在 **windows** 操作系统下需要下载相应的 ssh 客户端软件才可以用 ssh 方式登录，建议选择 SSHSecureShellClient 软件做为客户端软件。可到相关网站下载，也可到本机软件下载地址 http://w245.shu.edu.cn/downloads/SSHSecureShellClient-3.2.9.exe 下载。 注意本机上传文件是用 sftp 协议。

本系统建议采用 linux 客户端登录，这种使用方式较为稳定方便。

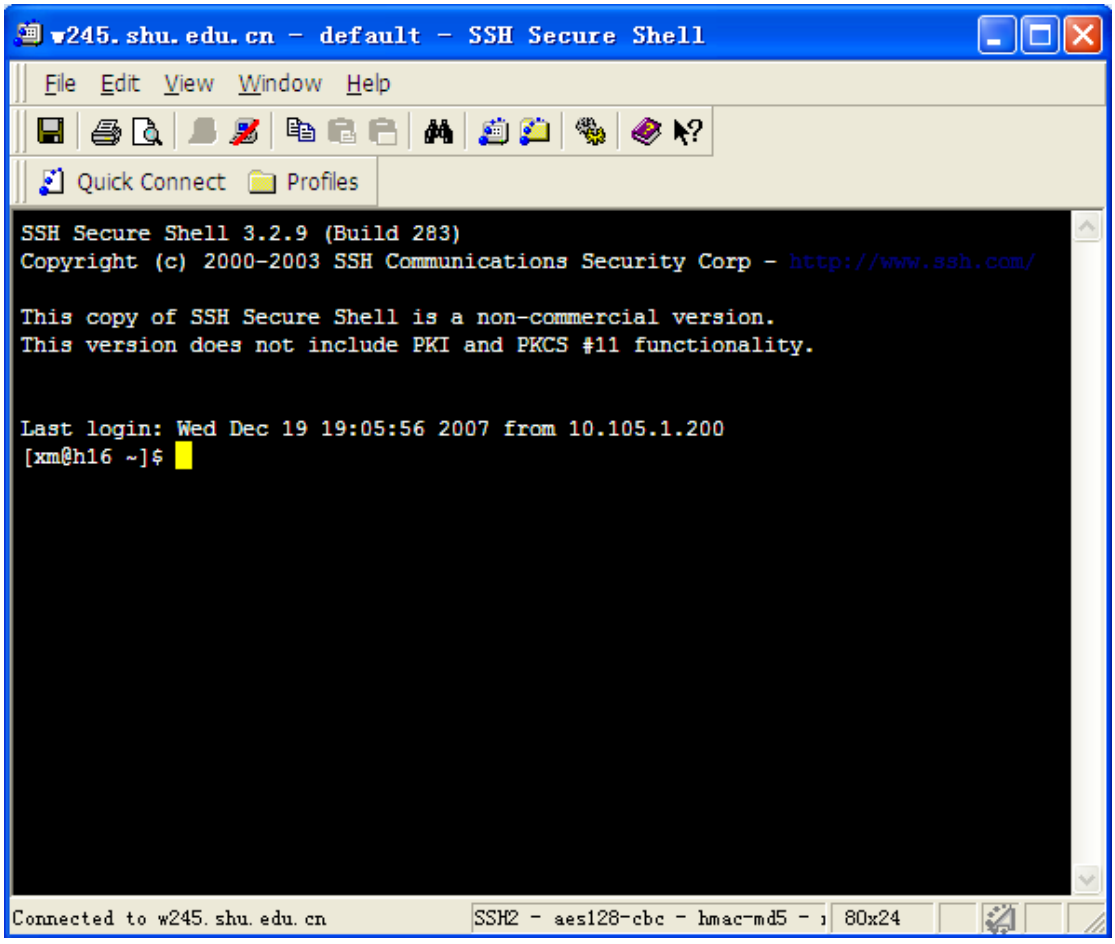
示例如下：

在 windows 下启动 SSHSecureShellClient 软件以后点击左上角的 Quick Connect 出现下面界面。



输入 Host name 地址(**w245.shu.edu.cn**)，用户名，端口号后点击 Connect，就会看到

输入密码以后就会登录到集群系统的登录节点。出现以下界面。



在此界面中即可输入 linux 系统相关命令进行操作。

2.MPI 使用方法

登录系统以后将出现 shell 窗口界面，上传源程序后，即可在此窗口里输入 MPI 相关命令即可进行编译，执行操作。

2.1.上传程序

上传程序可以用 ftp 方式传送,登录用户名和密码与用 ssh 登录相同，建议用 sftp 方式上传程序，这样较为安全，在 windows 下可以用 sftp 工具进行上传，SSHSecureShellClient 软件具有此功能非常易于使用。上传文件应该放在个人用户目录下，通常为/home/**username,** 可在此目录下建立子目录便于分类管理。

2.2. 编译 MPI 程序

编译 C 语言 mpi 程序命令如下

mpicc -o 可执行文件名 源文件路径/源文件名

假设在当前目录下有一个 cpi.c 源文件，编译成文件名 cpi 的可执行文件，执行命令如下

mpicc -o cpi cpi.c

如果源文件不在当前目录下必须在文件名前加上路径。
编译 c++语言 mpi 程序命令如下
mpiCC -o 可执行文件名 源文件路径/源文件名

假设在当前目录下有一个 cpi.c 源文件，编译成文件名 cpi 的可执行文件，执行命令如下

mpiCC -o cpi cpi.cpp

如果源文件不在当前目录下必须在文件名前加上路径。
编译 f77 fortran 语言 mpi 程序命令如下
mpif77 -o 可执行文件名 源文件路径/源文件名

假设在当前目录下有一个 cpi.c 源文件，编译成文件名 cpi 的可执行文件，执行命令如下

mpif77 -o cpi cpi.f

如果源文件不在当前目录下必须在文件名前加上路径。

2.3. 运行 MPI 程序

运行 MPI 程序命令如下
mpirun_ssh -np 处理器数目 -hostfile 机器文件 程序文件路径/程序文件名

假设在/home/student/目录下有一个 hfile 主机文件，并且程序文件在/home/student/examples/下文件名为 cpi，如果运行 2 个进程处理问题，示例如下
mpirun_ssh -np 2 -hostfile /home/student/hfile /home/student/examples/cpi

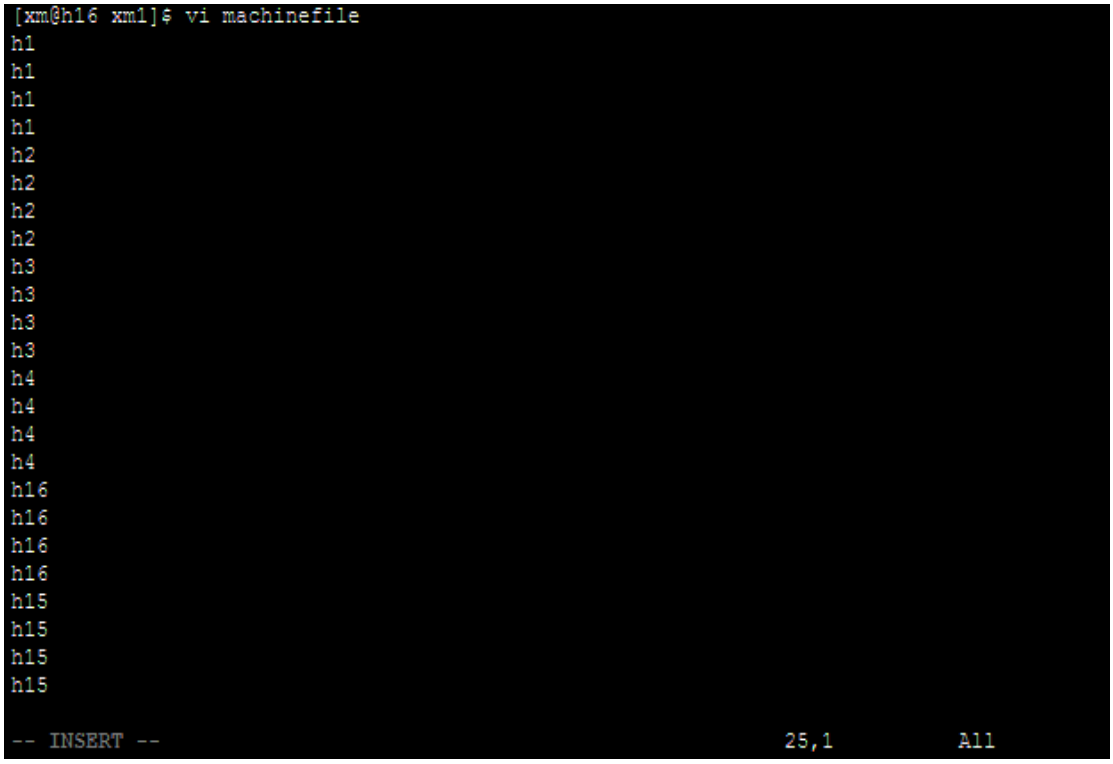
2.4. 以 C 语言举例

例 1：
并行程序为：hello.c
源程序为：

```
#include "mpi.h"
#include <stdio.h>
#include <math.h>
int main(int argc,char** argv)
{
int myid, numprocs;
int namelen;
```

```
char processor_name[MPI_MAX_PROCESSOR_NAME];
MPI_Init(&argc,&argv);
MPI_Comm_rank(MPI_COMM_WORLD,&myid);
MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
MPI_Get_processor_name(processor_name,&namelen);
printf("Hello World! Process %d of %d on %s\n",
myid, numprocs, processor_name);
MPI_Finalize();
}
```

由于需要主机的数目很多，不必没运行一次程序都把主机一个个写出来，就建立一个 machinefile 文件，建立这个文件命令为：vi machinefile　按 i 就可以在 machinefile 中输入你想要的主机名了,如下图示：



输完之后按 **Esc 键** 再按 **:wq** 保存退出。

然后就编译、运行 hello.c，命令如下图示：

```
[xm@h16 xm1]$ mpiCC -o hello hello.c
[xm@h16 xm1]$ mpirun_ssh -np 24 -hostfile ./machinefile ./hello
Hello World! Process 0 of 24 on h1
Hello World! Process 1 of 24 on h1
Hello World! Process 2 of 24 on h1
Hello World! Process 3 of 24 on h1
Hello World! Process 4 of 24 on h2
Hello World! Process 5 of 24 on h2
Hello World! Process 6 of 24 on h2
Hello World! Process 7 of 24 on h2
Hello World! Process 8 of 24 on h3
Hello World! Process 9 of 24 on h3
Hello World! Process 10 of 24 on h3
Hello World! Process 11 of 24 on h3
Hello World! Process 13 of 24 on h4
Hello World! Process 12 of 24 on h4
Hello World! Process 14 of 24 on h4
Hello World! Process 15 of 24 on h4
Hello World! Process 16 of 24 on h16
Hello World! Process 18 of 24 on h16
Hello World! Process 17 of 24 on h16
Hello World! Process 19 of 24 on h16
Hello World! Process 20 of 24 on h15
Hello World! Process 22 of 24 on h15
Hello World! Process 23 of 24 on h15
Hello World! Process 21 of 24 on h15
[xm@h16 xm1]$
```

例 2：

cpi.c 的程序如下：

```
#include "mpi.h"
#include <stdio.h>
#include <math.h>

double f(double);

double f(double a)
{
    return (4.0 / (1.0 + a*a));
}

int main(int argc,char *argv[])
{
    int done = 0, n, myid, numprocs, i;
    double PI25DT = 3.14159265358979323846264 3;
    double mypi, pi, h, sum, x;
    double startwtime = 0.0, endwtime;
    int    namelen;
```

```
    char processor_name[MPI_MAX_PROCESSOR_NAME];

    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid);
    MPI_Get_processor_name(processor_name,&namelen);

    fprintf(stdout,"Process %d of %d on %s\n",
        myid, numprocs, processor_name);

    n = 0;
    while (!done)
    {
        if (myid == 0)
        {
/*
            printf("Enter the number of intervals: (0 quits) ");
            scanf("%d",&n);
*/
        if (n==0) n=10000; else n=0;

        startwtime = MPI_Wtime();
        }
        MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD);
        if (n == 0)
            done = 1;
        else
        {
            h   = 1.0 / (double) n;
            sum = 0.0;
        /* A slightly better approach starts from large i and works back */
            for (i = myid + 1; i <= n; i += numprocs)
            {
                x = h * ((double)i - 0.5);
                sum += f(x);
            }
            mypi = h * sum;

            MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0,
MPI_COMM_WORLD);

            if (myid == 0)
        {
                printf("pi is approximately %.16f, Error is %.16f\n",
```

```
                                pi, fabs(pi - PI25DT));
            endwtime = MPI_Wtime();
            printf("wall clock time = %f\n", endwtime-startwtime);
            fflush( stdout );
            }
            }
        }
        MPI_Finalize();
        return 0;
}
```

先建立一个 imachine 文件，命令为：vi imachine

按 i 就可以输入你想要的主机名，记住主机数目

```
[xm@h16 xm1]$ vi imachine
i1
i1
i1
i1
i2
i2
i2
i2
i3
i3
i3
i3
i4
i4
i4
i4
i15
i15
i15
i15
i16
i16
i16
i16
~
~
-- INSERT --                                          24,4          All
```

按 esc 之后按 :wq 退出保存

编译，运行，结果如下图示：

```
[xm@h16 xm1]$ mpiCC -o cpi cpi.c
[xm@h16 xm1]$ mpirun_ssh -np 24 -hostfile ./imachine ./cpi
Process 0 of 24 on h1
Process 1 of 24 on h1
Process 2 of 24 on h1
Process 3 of 24 on h1
Process 4 of 24 on h2
Process 5 of 24 on h2
Process 6 of 24 on h2
Process 7 of 24 on h2
Process 8 of 24 on h3
Process 9 of 24 on h3
Process 10 of 24 on h3
Process 11 of 24 on h3
Process 12 of 24 on h4
Process 21 of 24 on h16
Process 20 of 24 on h16
Process 23 of 24 on h16
Process 14 of 24 on h4
Process 15 of 24 on h4
Process 13 of 24 on h4
Process 16 of 24 on h15
Process 17 of 24 on h15
Process 19 of 24 on h15
Process 18 of 24 on h15
Process 22 of 24 on h16
pi is approximately 3.1415926544231279, Error is 0.0000000008333347
wall clock time = 0.665082
[xm@h16 xm1]$
```

2.5. Screen 命令使用

以前为了让程序在脱离终端的情况下运行，要么让它在后台运行，要么使用 nohup 运行，但是如果需要交互的程序就麻烦了。例如，你需要使用 scp 拷贝，需要输入密码，而且数据量很大，需要很长时间。遇到过的人就知道痛苦了。有了 screen，一切都简单了。

1. screen 是什么？

根据其 man 介绍，screen 是个多元化多功能的全屏窗口管理器，每个虚拟终端都可以为你提供 DEC VT100 terminal 的功能，也许你会问:DEC VT100 terminal 又是什么?如果你登陆过某些字符界面的 BBS,或许你会记得在注册时，其要求你输入你的终端机型别，而一般预设就是我们刚刚提到的 DEC VT100 termina 了.另外 screen 还附加提供了比如 SO 6429 (ECMA 48, ANSI X3.64) and ISO 2022 standards 的操作功能.

2. screen 可以做些什么？

如果在以前或许 screen 是你登陆 bbs 站的好伴侣,但是相信现在大家都是直接登陆图形界面的也就是 WEB 界面的 BBS.当你正在登陆多个 BBS 而又不想在多个窗口之间切换.那么 screen 就可以帮你的忙了。

当然 screen 可不是专为 BBS 服务，它可以让你只需要打开一个终端窗口就可以地处理很多的(进程)事情,举个例子:你正在 shell 上编写某个程序,碰巧你又需要重新启动某个服务,同时还要 FTP 上传个大文件,这个时候就可以使用调用 screen,只需要按下 3 个键就可以无须用鼠标在 3 个窗口间切换.又或者你使用 PUTTY 等工具登陆到服务器,不想在退出时关闭当前的进程,比如你正在复制文件等.这个时候就可以利用 screen 让你复制文件这个前台进程享受后台进程的"待遇"。

正是因为 screen 的种种实用功能 ,已经成为不少*unix 玩家的必备利器,让*unix 的日常操作管理更加方便。

3. screen 使用：

使用 screen 非常简易.只需在 SHELL 键入 screen,便可打开一个 screen session。而在每个 screen session 下，所有命令都以 ctrl+a(C-a) 开始。

4. 现在让我来简单介绍基本的命令：

C-a c Create，开启新的 window

C-a n Next，切换到下个 window

C-a p Previous，前一个 window

C-a Other，在两个 window 间切换

C-a w Windows，列出已开启的 windows 有那些

C-a 0 切换到第 0 个 window

C-a 1..9 切换到第 1..9 个 window

C-a a 发出 C-a，在 emacs, ve, bash, tcsh 下可移到行首

C-a t Time，显示当前时间，和系统的 load

C-a K(大写) kill window，强行关闭当前的 window

C-a [进入 copy mode，在 copy mode 下可以回滚、搜索、复制就像用使用 vi 一样

C-b Backward，PageUp

C-f Forward，PageDown

H(大写) High，将光标移至左上角

L Low，将光标移至左下角

0 移到行首

$ 行末

w forward one word，以字为单位往前移

b backward one word，以字为单位往后移

Space 第一次按为标记区起点，第二次按为终点

Esc 结束 copy mode

C-a] Paste，把刚刚在 copy mode 选定的内容贴上

C-a ? Help，显示简单说明

C-a d detach，将目前的 screen session (可能含有多个 windows)丢到后台执行 当按了 C-a d 把 screen session detach 掉后，会回到还没进 screen 时的状态，此时在 screen session 里每个 window 内运行的 process (无论是前台/后台)都在继续执行，即使 logout 也不影响。

下次 login 进来时:

screen -ls 显示所有的 screen sessions

screen -r [keyword] 选择一个 screen session 恢复对话

若 screen -ls 里有 Attached sessions:

screen -d [keyword] 强制 detach，以便「接手」过来

5. 实例

说明看了那么多,让我们用一个实际例子来结束我们今天的学习。

在我们开启一个 screen 后,然后使用 joe 编辑一个文件,之后因为临时需要离开这时就可以运行 Ctrl+a d,显示如下：

[becks@ec-base becks]$ screen

[detached]

这个时候当我们运行 ps -e 可以看到 pts/2 这个我刚刚运行的 screen 正在运行 joe

6264 pts/2 00:00:00 bash

6354 pts/2 00:00:00 joe

而当我们回来后想恢复这个 session,只需要键入 screen -r,而当你有多个 session 时候,系统将提示你选择一个,如下：

[becks@ec-base becks]$ screen -r

There are several suitable screens on:

6263.pts-1.ec-base (Detached)

6382.pts-1.ec-base (Detached)

Type "screen [-d] -r [pid.]tty.host" to resume one of them.

输入该 session 的 pid 进行恢复

[becks@becks becks]$ screen -r 6263

想退出 screen 的 session,和退出 shell 一样,只需要键入 exit 命令,成功退出后将有以下提示

[screen is terminating]

screen 的简单用法就介绍到这里,更多的功能和应有请读者参考 MAN 自行研究.

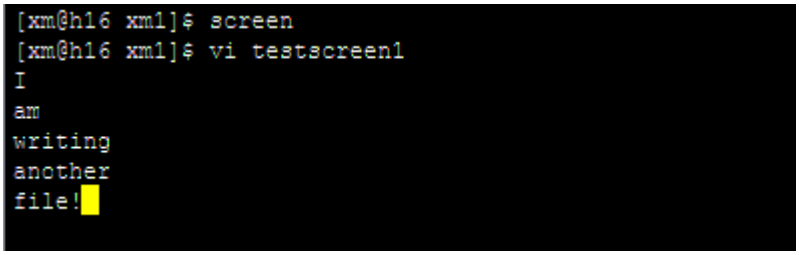
举例：

假设你现在正在编写一个文档，如下图所示：

如果你突然有别的重要的事情要做，你就可以把这个任务放到后台。按 ctrl+a d，就会看到如下内容（如图示）：

如果你还想新建一个 testscreen1 文件，

这时在光标处输入 screen 回车，然后建立新文件 testscreen1 如：



这时想中途退出这个任务去做别的事情，按 ctrl+a d 就会看到如下内容（如图示）：

如果想返回以前执行的程序，键入 screen –r，如果只有一个任务在后台就会直接返回到这个后台任务，如果有多个执行的程序，就会有如下提示（如下图所示）：

```
[xm@h16 xm1]$ screen -r
There are several suitable screens on:
        11914.pts-1.h16 (Detached)
        11946.pts-1.h16 (Detached)
Type "screen [-d] -r [pid.]tty.host" to resume one of them.
[xm@h16 xm1]$
```

键入 screen –r pid　pid 为上图的‘11914’、‘11946’就可以切换到你想要的任务

输入命令和得到的结果如下图所示：

```
[xm@h16 xm1]$ screen -r 11946
I
am
writing
another
file!

~
~
~
~
```

如果你做完了你的任务（如：完成文件编辑后保存退出按 esc 之后按 :wq）输入 exit 退出命令和显示的提示如下图示：

```
"testscreen1" [New] 6L, 28C written
[xm@h16 xm1]$ exit
exit

[screen is terminating]
[xm@h16 xm1]$
```

你输入几个后台任务，完成以后就要输入多少个 exit 退出。

3.常用命令

Linux 系统常用命令格式

command　[option]　[argument1]　[argument2]　...

其中 option 以“-”开始，多个 option 可用一个“-”连起来，如“ls -l -a” 与“ls -la”的效果是一样的。根据命令的不同，参数分为可选的或必须的；所有的命令从标准输入接受输入，输出结果显示在标准输出，而错误信息则显示在标准错误输出设备。可使用重定向功能对这些设备进行重定向。

Linux 系统常用命令

帮助命令:

man　获取相关命令的帮助信息

例如：man dir 可以获取关于 dir 的使用信息。

info　获取相关命令的详细使用方法
　　例如：info info 可以获取如何使用 info 的详细信息。

文件操作

cat　显示文件内容和合并多个文件
clear　清屏
chattr　改变文件属性
chgrp　改变文件组权
chmod　改变文件或目录的权限
chown　改变文件的属权
comm　比较两个已排过序的文件
cp　将文件拷贝至另一文件
dd　从指定文件读取数据写到指定文件
df　报告磁盘空间使用情况
diff　比较两个文本文件，列出行不同之处
du　统计目录／文件所占磁盘空间的大小
file　辨识文件类型
emacs　功能强大的编辑环境
find　搜索文件并执行指定操作(find2)
grep　按给定模式搜索文件内容
head　显示指定文件的前若干行
less　按页显示文件
ln　创建文件链接
locate　查找符合条件的文件
more　在终端屏幕按帧显示文本文件
mv　文件或目录的移动或更名
rm/rmdir　删除文件／目录
sed　利用 script 来处理文本文件
sort　对指定文件按行进行排序
tail　显示指定文件的最后部分
touch　创建文件
tr　转换字符
vi　全屏编辑器
wc　显示指定文件中的行数，词数或字符数
which　在环境变量 $PATH 设置的目录里查找符合条件的文件

压缩与备份:
bzip2/bunzip2　.bz2 文件的压缩/解压缩程序
cpio　备份文件

dump　备份文件系统
gzip/gunzip　.gz 文件的压缩/解压缩程序
gzexe　压缩可执行文件
restore 还原由倾倒(Dump)操作所备份下来的文件或整个文件系统(一个分区)
tar　将若干文件存档或读取存档文件
unarj　解压缩.arj 文件
zip/unzip　压缩/解压缩 zip 文件
zipinfo　列出 zip 压缩文件的详细信息

磁盘操作:

cd/pwd　切换目录/显示当前工作目录
df　显示磁盘的相关信息
du　显示目录或文件的大小
ls　列出目录内容
mkdir　创建目录

网络通信:

ftp　文件传输
lftp　文件传输
ping　向网络上的主机发送 icmp echo request 包
ssh　安全模式下的远程登录

Linux 文件的复制、删除和移动命令

cp 命令

该命令的功能是将给出的文件或目录拷贝到另一文件或目录中，同 MSDOS 下的 copy 命令一样，功能十分强大。

语法： cp [选项] 源文件或目录 目标文件或目录

说明：该命令把指定的源文件复制到目标文件或把多个源文件复制到目标目录中。

该命令的各选项含义如下：

- a 该选项通常在拷贝目录时使用。它保留链接、文件属性，并递归地拷贝目录，其作用等于 dpR 选项的组合。

- d 拷贝时保留链接。

- f 删除已经存在的目标文件而不提示。

- i 和 f 选项相反，在覆盖目标文件之前将给出提示要求用户确认。回答 y 时目标文件将被覆盖，是交互式拷贝。

- p 此时 cp 除复制源文件的内容外，还将把其修改时间和访问权限也复制到新文件中。

- r 若给出的源文件是一目录文件，此时 cp 将递归复制该目录下所有的子目录和文件。此时目标文件必须为一个目录名。

- l 不作拷贝，只是链接文件。

需要说明的是，为防止用户在不经意的情况下用 cp 命令破坏另一个文件，如用户指定的目标文件名已存在，用 cp 命令拷贝文件后，这个文件就会被新源文件覆盖，因此，建议用户在使用 cp 命令拷贝文件时，最好使用 i 选项。

例如： cp filename test //将 filename 文件复制到 test 目录中

mv 命令

用户可以使用 mv 命令来为文件或目录改名或将文件由一个目录移入另一个目录中。该命令如同 MSDOS 下的 ren 和 move 的组合。

语法：mv［选项］源文件或目录 目标文件或目录

说明：视 mv 命令中第二个参数类型的不同（是目标文件还是目标目录），mv 命令将文件重命名或将其移至一个新的目录中。当第二个参数类型是文件时，mv 命令完成文件重命名，此时，源文件只能有一个（也可以是源目录名），它将所给的源文件或目录重命名为给定的目标文件名。当第二个参数是已存在的目录名称时，源文件或目录参数可以有多个，mv 命令将各参数指定的源文件均移至目标目录中。在跨文件系统移动文件时，mv 先拷贝，再将原有文件删除，而链至该文件的链接也将丢失。

命令中各选项的含义为：

- I 交互方式操作。如果 mv 操作将导致对已存在的目标文件的覆盖，此时系统询问是否重写，要求用户回答 y 或 n，这样可以避免误覆盖文件。

- f 禁止交互操作。在 mv 操作要覆盖某已有的目标文件时不给任何指示，指定此选项后，i 选项将不再起作用。

如果所给目标文件（不是目录）已存在，此时该文件的内容将被新文件覆盖。为防止用户用 mv 命令破坏另一个文件，使用 mv 命令移动文件时，最好使用 i 选项。

例如：mv hello.c test/　　//将 hello.c 移至 test 目录下

mv -i hello.c test/

mv:overwrite ‘test/hello.c’？ //询问用户是否覆盖已知文件

rm 命令

用户可以用 rm 命令删除不需要的文件。该命令的功能为删除一个目录中的一个或多个文件或目录，它也可以将某个目录及其下的所有文件及子目录均删除。对于链接文件，只是断开了链接，原文件保持不变。

rm 命令的一般形式为：

rm［选项］文件…

如果没有使用- r 选项，则 rm 不会删除目录。

该命令的各选项含义如下：

- f 忽略不存在的文件，从不给出提示。

- r 指示 rm 将参数中列出的全部目录和子目录均递归地删除。

- i 进行交互式删除。

使用 rm 命令要小心。因为一旦文件被删除，它是不能被恢复的。了防止这种情况的发生，可以使用 i 选项来逐个确认要删除的文件。如果用户输入 y，文件将被删除。如果输入任何其他东西，文件则不会删除。

例如：rm -rf /xm/test

//直接删除/xm/test 目录 连同该目录下的子目录一并删除

Linux 目录的创建与删除命令

mkdir 命令

功能：创建一个目录（类似 MSDOS 下的 md 命令）。

语法：mkdir [选项] dir-name

说明：该命令创建由 dir-name 命名的目录。要求创建目录的用户在当前目录中（dir-name 的父目录中）具有写权限，并且 dirname 不能是当前目录中已有的目录或 文件名称。

命令中各选项的含义为：

- m 对新建目录设置存取权限。也可以用 chmod 命令设置。

- p 可以是一个路径名称。此时若路径中的某些目录尚不存在， 加上此选项后， 系统将自动建立好那些尚不存在的目录，即一次可以建立多个目录。

例如 ： mkdir test //创建一个 test 目录

rmdir 命令

功能：删除空目录。

语法：rmdir [选项] dir-name

说明：dir-name 表示目录名。该命令从一个目录中删除一个或多个子目录项。需要 特别注意的是，一个目录被删除之前必须是空的。rm - r dir 命令可代替 rmdir，但是有危险性。删除某目录时也必须具有对父目录的写权限。

命令中各选项的含义为：

- p 递归删除目录 dirname，当子目录删除后其父目录为空时，也一同被删除。如果整个路径被删除或者由于某种原因保留部分路径，则系统在标准输出上显示相应 的信息。

例如：[xm@h16 ~]$ rmdir -p /home/xm/test

//如果 test 目录为空，删除 test ，然后，如果 xm 为空目录也会删除 xm 目录

cd 命令

功能：改变工作目录。

语法：cd [directory]

说明：该命令将当前目录改变至 directory 所指定的目录。若没有指定 directory， 则回到用户的主目录。为了改变到指定目录，用户必须拥有对指定目录的执行和读 权限。

该命令可以使用通配符。

例如：[xm@h16 ~]$ cd /home/

[xm@h16 home]$ //进入到 home 工作目录

[xm@h16 home]$ cd

[xm@h16 ~]$ //返回到起始路径

pwd 命令

在 Linux 层次目录结构中，用户可以在被授权的任意目录下利用 mkdir 命令创建新目录，也可以利用 cd 命令从一个目录转换到另一个目录。然而，没有提示符来告知用 户目前处于哪一个目录中。要想知道当前所处的目录，可以使用 pwd 命令，该命令显示整个路径名。

语法：pwd

说明：此命令显示出当前工作目录的绝对路径。

例如：[xm@h16 ~]$ pwd

/home/xm

ls 命令

ls 是英文单词 list 的简写，其功能为列出目录的内容。这是用户最常用的一个命令之一，因为用户需要不时地查看某个目录的内容。该命令类似于 DOS 下的 dir 命令。

语法：ls [选项] [目录或是文件]

对于每个目录，该命令将列出其中的所有子目录与文件。对于每个文件，ls 将输出 其文件名以及所要求的其他信息。默认情况下，输出条目按字母顺序排序。当未给出目录名或是文件名时，就显示当前目录的信息。

命令中各选项的含义如下：

- a 显示指定目录下所有子目录与文件，包括隐藏文件。

- A 显示指定目录下所有子目录与文件，包括隐藏文件。但不列出“.”和 “..”。

- b 对文件名中的不可显示字符用八进制逃逸字符显示。

- c 按文件的修改时间排序。

- C 分成多列显示各项。

- d 如果参数是目录，只显示其名称而不显示其下的各文件。往往与 l 选项一起使 用，以得

到目录的详细信息。

- f 不排序。该选项将使 lts 选项失效，并使 aU 选项有效。

- F 在目录名后面标记“/”，可执行文件后面标记“*”，符号链接后面标记 “@”，管道（或 FIFO）后面标记“|”，socket 文件后面标记“=”。

- i 在输出的第一列显示文件的 i 节点号。

- l 以长格式来显示文件的详细信息。这个选项最常用。

每行列出的信息依次是：文件类型与权限 链接数 文件属主 文件属组 文件大小 建立或最近修改的时间 名字

对于符号链接文件，显示的文件名之后有“—〉”和引用文件路径名。

对于设备文件，其“文件大小”字段显示主、次设备号，而不是文件大小。

目录中的总块数显示在长格式列表的开头，其中包含间接块。

- L 若指定的名称为一个符号链接文件，则显示链接所指向的文件。

- m 输出按字符流格式，文件跨页显示，以逗号分开。

- n 输出格式与 l 选项相同，只不过在输出中文件属主和属组是用相应的 UID 号和 GID 号来表示，而不是实际的名称。

- o 与 l 选项相同，只是不显示拥有者信息。

- p 在目录后面加一个“/”。

- q 将文件名中的不可显示字符用“?”代替。

- r 按字母逆序或最早优先的顺序显示输出结果。

- R 递归式地显示指定目录的各个子目录中的文件。

- s 给出每个目录项所用的块数，包括间接块。

- t 显示时按修改时间（最近优先）而不是按名字排序。若文件修改时间相同，则 按字典顺序。修改时间取决于是否使用了 c 或 u 选项。缺省的时间标记是最后一次修 改时间。

- u 显示时按文件上次存取的时间（最近优先）而不是按名字排序。即将-t 的时间 标记修改为最后一次访问的时间。

- x 按行显示出各排序项的信息。

用 ls -l 命令显示的信息中，开头是由 10 个字符构成的字符串，其中第一个字符表示文件类型，它可以是下述类型之一：

- 普通文件

d 目录

l 符号链接

b 块设备文件

c 字符设备文件

后面的 9 个字符表示文件的访问权限，分为 3 组，每组 3 位。

第一组表示文件属主的权限，第二组表示同组用户的权限，第三组表示其他用户的权限。每一组的三个字 符分别表示对文件的读、写和执行权限。

各权限如下所示：

r 读

w 写

x 执行。对于目录，表示进入权限。

s 当文件被执行时，把该文件的 UID 或 GID 赋予执行进程的 UID（用户 ID）或 GID（组 ID）。

t 设置标志位（留在内存，不被换出）。如果该文件是目录，在该目录中的文件只能被超级用户、目录拥有者或文件属主删除。如果它是可执行文件，在该文件执行 后，指向其正文段的指针仍留在内存。这样再次执行它时，系统就能更快地装入该文件。

例如：ls /home/xm　　//显示/home/xm 目录下的内容

Linux 备份与压缩命令

tar 命令

tar 可以为文件和目录创建档案。利用 tar，用户可以为某一特定文件创建档案（备份文件），也可以在档案中改变文件，或者向档案中加入新的文件。tar 最初被用来在磁带上创建档案，现在，用户可以在任何设备上创建档案，如软盘。利用 tar 命令，可以把一大堆的文件和目录全部打包成一个文件，这对于备份文件或将几个文件组合成为一个文件以便于网络传输是非常有用的。Linux 上的 tar 是 GNU 版本的。

语法：tar [主选项+辅选项] 文件或者目录

使用该命令时，主选项是必须要有的，它告诉 tar 要做什么事情，辅选项是辅助使用的，可以选用。

主选项：

c 创建新的档案文件。如果用户想备份一个目录或是一些文件，就要选择这个选项。

r 把要存档的文件追加到档案文件的末尾。例如用户已经作好备份文件，又发现还有一个目录或是一些文件忘记备份了，这时可以使用该选项，将忘记的目录或文件追加到备份文件中。

t 列出档案文件的内容，查看已经备份了哪些文件。

u 更新文件。就是说，用新增的文件取代原备份文件，如果在备份文件中找不到要更新的文件，则把它追加到备份文件的最后。

x 从档案文件中释放文件。

辅助选项：

b 该选项是为磁带机设定的。其后跟一数字，用来说明区块的大小，系统预设值为 20（20*512 bytes）。

f 使用档案文件或设备，这个选项通常是必选的。

k 保存已经存在的文件。例如我们把某个文件还原，在还原的过程中，遇到相同的文件，不会进行覆盖。

m 在还原文件时，把所有文件的修改时间设定为现在。

M 创建多卷的档案文件，以便在几个磁盘中存放。

v 详细报告 tar 处理的文件信息。如无此选项，tar 不报告文件信息。

w 每一步都要求确认。

z 用 gzip 来压缩/解压缩文件，加上该选项后可以将档案文件进行压缩，但还原时也一定要使用该选项进行解压缩。

例如：[xm@h16 ~]$ tar -cf test.bar test

//将 test 目录备份到 test.tar 存档文件中

gzip 命令

减少文件大小有两个明显的好处，一是可以减少存储空间，二是通过网络传输文件时，可以减少传输的时间。gzip 是在 Linux 系统中经常使用的一个对文件进行压缩和解压缩的命令，既方便又好用。

语法：gzip [选项] 压缩（解压缩）的文件名

各选项的含义：

-c 将输出写到标准输出上，并保留原有文件。

-d 将压缩文件解压。

-l 对每个压缩文件，显示下列字段：

压缩文件的大小

未压缩文件的大小

压缩比

未压缩文件的名字

-r 递归式地查找指定目录并压缩其中的所有文件或者是解压缩。

-t 测试，检查压缩文件是否完整。

-v 对每一个压缩和解压的文件，显示文件名和压缩比。

-num 用指定的数字 num 调整压缩的速度，-1 或--fast 表示最快压缩方法（低压缩比），-9 或--best 表示最慢压缩方法（高压缩比）。系统缺省值为 6。

例如：[xm@h16 ~]$ gzip 12.txt　　//对 12.txt 进行压缩

[xm@h16 ~]$ gzip -l 12.txt.gz　　//测试压缩文件

compressed　　uncompressed　ratio uncompressed_name

64　　45　31.1% 12.txt

[xm@h16 ~]$ gzip -d 12.txt.gz　　//解压 12.txt 文件

unzip 命令

用 MS Windows 下的压缩软件 winzip 压缩的文件如何在 Linux 系统下展开呢？可以用 unzip 命令，该命令用于解扩展名为.zip 的压缩文件。

语法：unzip [选项] 压缩文件名.zip

各选项的含义分别为：

-x 文件列表 解压缩文件，但不包括指定的 file 文件。

-v 查看压缩文件目录，但不解压。

-t 测试文件有无损坏，但不解压。

-d 目录 把压缩文件解到指定目录下。

-z 只显示压缩文件的注解。

-n 不覆盖已经存在的文件。

-o 覆盖已存在的文件且不要求用户确认。

-j 不重建文档的目录结构，把所有文件解压到同一目录下。

例如：unzip –l test.zip //显示压缩文件的内容；

unzip –t test.zip //测试压缩文件看是否有错

unzip –d xm test.zip //解压文件到 xm 目录下

Linux 改变文件或目录的访问权限命令

Linux 系统中的每个文件和目录都有访问许可权限，用它来确定谁可以通过何种方式对文件和目录进行访问和操作。

文件或目录的访问权限分为只读，只写和可执行三种。以文件为例，只读权限表示只允许读其内容，而禁止对其做任何的更改操作。可执行权限表示允许将该文件作为一个程序执行。文件被创建时，文件所有者自动拥有对该文件的读、写和可执行权限，以便于对文件的阅读和修改。用户也可根据需要把访问权限设置为需要的任何组合。

有三种不同类型的用户可对文件或目录进行访问：文件所有者，同组用户、其他用户。所有者一般是文件的创建者。所有者可以允许同组用户有权访问文件，还可以将文件的访问权限赋予系统中的其他用户。在这种情况下，系统中每一位用户都能访问该用户拥有的文件或目

录。

每一文件或目录的访问权限都有三组，每组用三位表示，分别为文件属主的读、写和执行权限；与属主同组的用户的读、写和执行权限；系统中其他用户的读、写和执行权限。当用 ls -l 命令显示文件或目录的详细信息时，最左边的一列为文件的访问权限。例如：

$ ls -l sobsrc. tgz

-rw-r--r-- 1 root root 483997 Ju1 l5 17:3l sobsrc. tgz

横线代表空许可。r 代表只读，w 代表写，x 代表可执行。注意这里共有 10 个位置。第一个字符指定了文件类型。在通常意义上，一个目录也是一个文件。如果第一个字符是横线，表示是一个非目录的文件。如果是 d，表示是一个目录。

例如：

- rw- r-- r--

普通文件 文件主 组用户 其他用户

是文件 sobsrc.tgz 的访问权限，表示 sobsrc.tgz 是一个普通文件；sobsrc.tgz 的属主有读写权限；与 sobsrc.tgz 属主同组的用户只有读权限；其他用户也只有读权限。

确定了一个文件的访问权限后，用户可以利用 Linux 系统提供的 chmod 命令来重新设定不同的访问权限。也可以利用 chown 命令来更改某个文件或目录的所有者。利用 chgrp 命令来更改某个文件或目录的用户组。

下面分别对这些命令加以介绍。

chmod 命令

chmod 命令是非常重要的，用于改变文件或目录的访问权限。用户用它控制文件或目录的访问权限。
该命令有两种用法。一种是包含字母和操作符表达式的文字设定法；另一种是包含数字的数字设定法。
1. 文字设定法
chmod [who] [+ | - | =] [mode] 文件名?
命令中各选项的含义为：
操作对象 who 可是下述字母中的任一个或者它们的组合：
u 表示“用户（user）”，即文件或目录的所有者。
g 表示“同组（group）用户”，即与文件属主有相同组 ID 的所有用户。

o 表示“其他（others）用户”。

a 表示“所有（all）用户”。它是系统默认值。

操作符号可以是：

+ 添加某个权限。

- 取消某个权限。

= 赋予给定权限并取消其他所有权限（如果有的话）。

设置 mode 所表示的权限可用下述字母的任意组合：

r 可读。

w 可写。

x 可执行。

X 只有目标文件对某些用户是可执行的或该目标文件是目录时才追加 x 属性。

s 在文件执行时把进程的属主或组 ID 置为该文件的文件属主。方式“u＋s”设置文件的用户 ID 位，“g＋s”设置组 ID 位。

t 保存程序的文本到交换设备上。

u 与文件属主拥有一样的权限。

g 与和文件属主同组的用户拥有一样的权限。

o 与其他用户拥有一样的权限。

文件名：以空格分开的要改变权限的文件列表，支持通配符。

在一个命令行中可给出多个权限方式，其间用逗号隔开。例如：chmod g+r，o+r example

使同组和其他用户对文件 example 有读权限。

2. 数字设定法

我们必须首先了解用数字表示的属性的含义：0 表示没有权限，1 表示可执行权限，2 表示可写权限，4 表示可读权限，然后将其相加。所以数字属性的格式应为 3 个从 0 到 7 的八进制数，其顺序是（u）（g）（o）。

例如，如果想让某个文件的属主有“读/写”二种权限，需要把 4（可读）+2（可写）＝6（读/写）。

数字设定法的一般形式为：

chmod [mode] 文件名
例如： ls -l filename
-rw-r--r-- 1 xm xm 452 Dec 14 18:00 filename
chmod u=rwx,g=rx,o=r filename
ls -l filename
-rwxr-xr-- 1 xm xm 452 Dec 14 18:00 filename
// filename 的属性-rw-r--r—改为-rwxr-xr--

chgrp 命令

功能：改变文件或目录所属的组。

语法：chgrp [选项] group filename?

该命令改变指定指定文件所属的用户组。其中 group 可以是用户组 ID，也可以是/etc/group 文件中用户组的组名。文件名是以空格分开的要改变属组的文件列表，支持通配符。如果用户不是该文件的属主或超级用户，则不能改变该文件的组。

该命令的各选项含义为：

- R 递归式地改变指定目录及其下的所有子目录和文件的属组。
例如：chgrp –R test hello.c //将 hello.c 改变在 test 目录下

chown 命令

功能：更改某个文件或目录的属主和属组。这个命令也很常用。例如 root 用户把自己的一个文件拷贝给用户 xu，为了让用户 xu 能够存取这个文件，root 用户应该把这个文件的属主设为 xu，否则，用户 xu 无法存取这个文件。

语法：chown [选项] 用户或组 文件

说明：chown 将指定文件的拥有者改为指定的用户或组。用户可以是用户名或用户 ID。组可以是组名或组 ID。文件是以空格分开的要改变权限的文件列表，支持通配符。

该命令的各选项含义如下：

- R 递归式地改变指定目录及其下的所有子目录和文件的拥有者。

- v 显示 chown 命令所做的工作。
例如：chown -h xmm 11.txt//将 11.txt 的所有这改为 xmm