Linux Basics for Biostack® Server

1.准备工作

Biostack® 生物信息一体机剖析

- 第一层： 硬件层，服务器硬件架构，内存、处理器、硬盘、电源
- 第二层：操作系统， CentOS
- 第三层：开发和运行环境， Perl、 Python、 GCC、 R开发环境
- 第四层：行业应用层，软件、数据库、生物信息数据分析流程

推荐安装软件

- **MobaXterm**: Enhanced terminal for Windows with X11 server, tabbed SSH client.
- **Sublime Text 3**: A sophisticated text editor for code, markup and prose
- **EmEditor**: fast, lightweight, yet extensible, easy-to-use text editor for Windows.

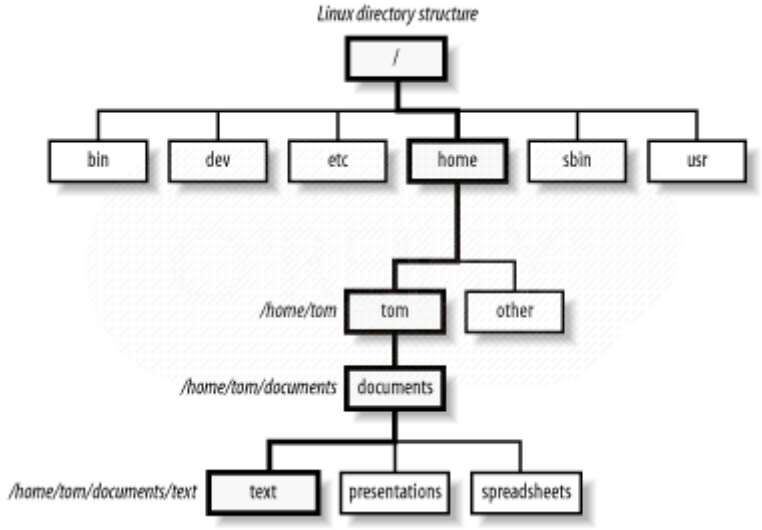
如何使用SSH登录Linux服务器： MobaXterm

1.设置新会话
2.选择SSH登录
3.设置远程服务器IP地址：192.168.0.16
4.设置远程服务器用户名：biostack
5.设置远程服务器IP端口：22

2.linux基础

约定: linux终端符号为 `$` (普通用户) 或者 `#` (root用户)， linux终端符后的程序一般可以直接运行, bash 代码域 `#`后面为注释文件, 不能拷贝至命令行。

linux系统目录结构



以下是对这些目录的解释：

- **/bin**：
  bin是Binary的缩写, 这个目录存放着最经常使用的命令。
- **/boot**：
  这里存放的是启动Linux时使用的一些核心文件，包括一些连接文件以及镜像文件。
- **/dev**：
  dev是Device(设备)的缩写, 该目录下存放的是Linux的外部设备, 在Linux中访问设备的方式和访问文件的方式是相同的。
- **/etc**：
  这个目录用来存放所有的系统管理所需要的配置文件和子目录。
- **/home**：
  用户的主目录，在Linux中，每个用户都有一个自己的目录，一般该目录名是以用户的账号命名的。
- **/lib**：
  这个目录里存放着系统最基本的动态连接共享库，其作用类似于Windows里的DLL文件。几乎所有的应用程序都需要用到这些共享库。
- **/lost+found**：
  这个目录一般情况下是空的，当系统非法关机后，这里就存放了一些文件。
- **/media**：
  linux 系统会自动识别一些设备，例如U盘、光驱等等，当识别后，linux会把识别的设备挂载到这个目录下。
- **/mnt**：
  系统提供该目录是为了让用户临时挂载别的文件系统的，我们可以将光驱挂载在/mnt/上，然后进入该目录就可以查看光驱里的内容了。
- **/opt**：
  这是给主机额外安装软件所摆放的目录。比如你安装一个ORACLE数据库则就可以放到这个目录下。默认是空的。

- **/proc**：
  这个目录是一个虚拟的目录，它是系统内存的映射，我们可以通过直接访问这个目录来获取系统信息。
  这个目录的内容不在硬盘上而是在内存里，我们也可以直接修改里面的某些文件，比如可以通过下面的命令来屏蔽主机的ping命令，使别人无法ping你的机器：
- **/root**：
  该目录为系统管理员，也称作超级权限者的用户主目录。
- **/sbin**：
  s就是Super User的意思，这里存放的是系统管理员使用的系统管理程序。
- **/srv**：
  该目录存放一些服务启动之后需要提取的数据。
- **/sys**：

  这是linux2.6内核的一个很大的变化。该目录下安装了2.6内核中新出现的一个文件系统 sysfs 。

  sysfs文件系统集成了下面3种文件系统的信息：针对进程信息的proc文件系统、针对设备的devfs文件系统以及针对伪终端的devpts文件系统。

  该文件系统是内核设备树的一个直观反映。

  当一个内核对象被创建的时候，对应的文件和目录也在内核对象子系统中被创建。
- **/tmp**：
  这个目录是用来存放一些临时文件的。
- **/usr**：
  这是一个非常重要的目录，用户的很多应用程序和文件都放在这个目录下，类似于windows下的program files目录。
- **/usr/bin**：
  系统用户使用的应用程序。
- **/usr/sbin**：
  超级用户使用的比较高级的管理程序和系统守护程序。
- **/usr/src**：
  内核源代码默认的放置目录。
- **/var**：
  这个目录中存放着在不断扩充着的东西，我们习惯将那些经常被修改的目录放在这个目录下。包括各种日志文件。
- **/run**：
  是一个临时文件系统，存储系统启动以来的信息。当系统重启时，这个目录下的文件应该被删掉或清除。如果你的系统上有 /var/run 目录，应该让它指向 run。

在 Linux 系统中，有几个目录是比较重要的，平时需要注意不要误删除或者随意更改内部文件。

/etc： 上边也提到了，这个是系统中的配置文件，如果你更改了该目录下的某个文件可能会导致系统不能启动。

/bin, /sbin, /usr/bin, /usr/sbin: 这是系统预设的执行文件的放置目录，比如 ls 就是在/bin/ls 目录下的。

值得提出的是，/bin, /usr/bin 是给系统用户使用的指令（除root外的通用户），而/sbin, /usr/sbin 则是给root使用的指令。

/var： 这是一个非常重要的目录，系统上跑了很多程序，那么每个程序都会有相应的日志产生，而这些日志就被记录到这个目录下，具体在/var/log 目录下，另外mail的预设放置也是在这里。

Linux常用特殊字符

Shell 输入/输出重定向

重定向命令列表如下：

命令	说明
command > file	将输出重定向到 file。
command < file	将输入重定向到 file。
command >> file	将输出以追加的方式重定向到 file。
n > file	将文件描述符为 n 的文件重定向到 file。
n >> file	将文件描述符为 n 的文件以追加的方式重定向到 file。
n >& m	将输出文件 m 和 n 合并。
n <& m	将输入文件 m 和 n 合并。
<< tag	将开始标记 tag 和结束标记 tag 之间的内容作为输入。

> 需要注意的是文件描述符 0 通常是标准输入（STDIN），1 是标准输出（STDOUT），2 是标准错误输出（STDERR）。

常用的文件描述符如下：

文件描述符	名称	常用缩写	默认值
0	标准输入	**stdin**	键盘
1	标准输出	**stdout**	屏幕
2	标准错误输出	**stderr**	屏幕

管道符	|	管道符|可将命令的结果输出给另一个命令作为输入之用
连接符号	；	连续执行多个命令，放在一行执行，中间用"；"分开
后台执行	**&**	用户有时候执行命令要花很长时间，可能会影响做其他事情。最好的方法是将它放在后台执行。后台运行的程序在用户注销后系统还可以继续执行。当要把命令放在后台执行时，在命令的后面加上**"&"**
斜线	**/**	在路径表示中代表目录
反斜线	\	放在指令前，有取消 **aliases**的作用；放在特殊符号前，该特殊符号的作用消失,；放在指令的最末端，表示指令连接下一行。
井号	#	**#**：管理员，**$**：普通用户，脚本中的：**#**号是注释，如果被用在指令中，或者引号双引号括住的话，或者在倒斜线的后面，那他就变成一般符号，不具上述的特殊功能。
点（**dot**）	.	**.**代表当前目录，**..** 代表上层目录，如果在档案名称前有 **.**，需要**ls -a** 才会显示，特殊字符点号用来匹配除换行符之外的任意单个字符。它必须匹配一个字符，如果在点号字符的位置没有字符，那么模式就不成立
逗号	,	在运算中当做区隔的用途
惊叹号	!	代表反逻辑的作用
问号	?	问号表明前面的字符可以出现**0**次或**1**次，不包含**null**字元
星号	*	常用的符号，在文件名扩展上，用来代表任何字元，包含**null**字元
锚字符	^	这个符号在正则表达式中，代表行的"开头"位置，在**[]**中也与"！"（叹号）一样表示"非"
连续分号	;;	专用在**case**的选项，担任**Terminator**的角色
单引号	''	被单引号用括号括住的内容，将被视为单一字串。在引号内的代表变数的**$**符号，没有作用，被视为一般符号处理，防止任何变量替换
双引号	""	被双引号括住的内容，将被视为单一字串，防止通配符扩展，但允许变量扩展，这点与单引号的处理方式不同
倒引号	```	在前面的单双引号，括住的是字串，但如果该字串是一列命令列，要处理这种情况，得用倒单引号来做
次方运算	**	两个星号在运算时代表"次方"的意思

工作目录切换

pwd命令：显示当前工作目录的绝对路径, 用户登入linux系统后, 一般直接进入指定home用户目录。

语法：`pwd`

示例：

```
$ pwd
```

```
/home/biostack
```

cd命令：用来切换工作目录至指定目录。

语法：`cd [指定目录]`

解读：

```
cd /          #进入主目录
cd            #进入用户主目录；
cd ~           #进入用户主目录；
cd -          #返回进入此目录之前所在的目录；
cd ..         #返回上级目录（若当前目录为"/"，则执行完后还在"/"；".."为上级目录的意思）；
cd ../..     #返回上两级目录；
```

ls命令：用来显示指定目录列表。

语法：`ls [-option] [目录]`

示例：

```
$ cd /
$ ls
```

```
bin  biostack  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc
project  root  run  sbin  srv  sys  tmp  usr  var
```

打包压缩和解压文件

tar命令：tar是`linux`中最常用的解压缩命令。tar命令可用于处理后缀名为`tar，tar.gz，tgz，.tar.Z，tar.bz2`的文件.

语法：`tar [-option][文件或目录...]`

解读：

```
tar -czvf  files.tar.gz files          #压缩要被压缩的文件或目录名称
tar -tzvf  files.tar.gz                #查询压缩包内容列表
tar -xzvf  files.tar.gz -C ./files    #解压缩欲解压缩的目录
```

示例：

```
$ echo "hello, word!"  > file.txt
$ tar  czvf  file.tar.gz  file.txt
$ tar  tzvf  file.tar.gz
$ rm file.txt
$ tar xzvf file.tar.gz
```

更多内容： Linux上常用的文件（打包）解压和压缩工具：tar

gzip命令：用来压缩文件, gzip是个使用广泛的压缩程序，文件经它压缩过后，其名称后面会多处".gz"扩展名。

语法：`gzip[-option][文件列表]`

解读：

```
gzip -k ./*       #当前目录下所有文件进行压缩，每个文件一个gz包
gzip -d file.gz   #解压file.gz文件
gunzip file.gz    #相当于gzip -d file.gz
```

示例：

```
$ echo "hello, word!" > file.txt
$ gzip file.txt
$ gzip -d file.txt.gz
```

```
$ echo "hello, word!" > file.txt
$ gzip file.txt
$ gunzip file.txt.gz
```

zip命令：可以用来解压缩文件，或者对文件进行打包操作，主要用于处理`zip`包。

语法：`zip [option] [zip压缩包] [文件列表]`

示例：

打包`files`目录下的文件，并命名为`files.zip`

```
$ mkdir files
$ echo "Hello, biostack" > files/file.txt
$ zip -r files.zip files
```

unzip命令：用于解压缩由zip命令压缩的`.zip`压缩包。

语法：`unzip [option][压缩包]`

示例：

```
$ mkdir dir
$ zip -r file.zip dir
$ rm -r dir
$ unzip -o file.zip -d dir
```

搜索命令

grep命令：（`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。grep 支持不同的匹配模式，比如默认的 BRE 模式，增强型的 ERE模式，还有更强悍的 PRE 模式。普通情况下使用默认的 `BRE(basic regular expression)` 模式就可以了，这种方式的特点是支持的正则表达式语法有限。如果需要更进一步的正则表达式语法支持，可以使用 `ERE(extended regular expression)` 模式。如果要使用复杂的正则表达式语法，可以使用 `PRE` 模式，它支持 `Perl` 语言的正则表达式语法。

语法：`grep [-option] [filename]`

参数：

```
-G, --basic-regexp          #BRE 模式，也是默认的模式
-E, --extended-regexp       #ERE 模式
-P, --perl-regexp           #PRE 模式
-w                          #只显示全字符合的列。
```

解读：

```
grep "word" file.txt        #打印一个包含"word"的文本行
grep -v "word" file.txt     #打印不包含"word"的文本行
```

示例：

```
$ echo  "hello, word!" > file.txt
$ echo  "hello, world" >>file.txt
$ grep  "word" file.txt
$ grep -v "word" file.txt
```

find命令：用来在指定目录下查找文件。任何位于参数之前的字符串都将被视为欲查找的目录名。如果使用该命令时，不设置任何参数，则`find`命令将在当前目录下查找子目录与文件。并且将查找到的子目录和文件全部进行显示。

语法：`find [option] [起始目录]`

解读：

```
find .                      #列出当前目录及子目录下所有文件和文件夹
find /home -name "*.txt"    #在`/home`目录下查找以.txt结尾的文件名
```

示例：

```
$ mkdir biostack
$ echo "hello, word" > biostack/file.txt
$ find .
$ find biostack -name "*.txt"
```

which命令：用于查找并显示给定命令的绝对路径，环境变量PATH中保存了查找命令时需要遍历的目录。which指令会在环境变量$PATH设置的目录里查找符合条件的文件。也就是说，使用which命令，就可以看到某个系统命令是否存在，以及执行的到底是哪一个位置的命令。

语法：`which [option][指令名]`

示例：

显示pwd命令绝对路径

```
$ which pwd
```

文本文件查看命令

cat命令：通常是用于观看某个文件的内容。

语法：`cat [-OPTION][fileName]`

示例：

一次显示整个文件

```
$ echo "hello, word!" > file1.txt
$ cat file1.txt
```

从键盘创建一个文件。只能创建新文件,不能编辑已有文件。

```
$ cat file1.txt >  file2.txt
```

将几个文件合并为一个文件, 可以直接将几个压缩包文件进行合并,结果也是压缩文件格式。

```
$ cat file1.txt file2.txt > file.txt
$ cat file.txt
$ gzip file1.txt file2.txt
$ cat file1.txt.gz file2.txt.gz  >file.txt.gz
```

zcat命令：用于不真正解压缩文件，就能显示压缩包中文件的内容的场合。

语法：`zcat [-OPTION] [file]`

示例：

```
$ echo "hello, word!" > file1.txt
$ gzip file1.txt
$ zcat file1.txt.gz
```

more命令：是一个基于vi编辑器文本过滤器，它以全屏幕的方式按页显示文本文件的内容，支持vi中的关键字定位操作。

语法：`more [-OPTION] [file]`

解读：

```
more  file          #显示文件内容
more -c -10 file    #显示文件file的内容，每10行显示一次，而且在显示之前先清屏。
```

示例：

```
$ echo "hello, word!" > file.txt
$ more  file.txt
$ more -c -10 file
```

less命令的作用与more十分相似，都可以用来浏览文字档案的内容，不同的是less命令允许用户向前或向后浏览文件，而more命令只能向前浏览。用less命令显示文件时，用PageUp键向上翻页，用PageDown键向下翻页。要退出less程序，应按Q键。

语法：`less [-OPTION] [file]`

解读：

```
less file #显示文件内容
```

示例：

```
$ echo "hello, word!" > file.txt
$ less file.txt
```

head命令:用于显示文件的开头的内容。在默认情况下，head命令显示文件的头10行内容。

语法：`head [-OPTION] [文件列表]`

解读：

```
head file         #显示file文件的前10行的内容
head -n 5  file   #显示file文件的前5行的内容
head -c 20 file   #显示file文件的前20字节的内容
```

示例：

```
$ seq 1 20 > file.txt
$ head file.txt
$ head -n 5 file
$ head -c 20 file
```

tail命令:用于输入文件中的尾部内容。tail命令默认在屏幕上显示指定文件的末尾10行。

语法：`tail [-OPTION] [文件]`

解读：

```
tail file         #显示文件file的最后10行
tail +20 file     #显示文件file的内容，从第20行至文件末尾
tail -c 10 file   #显示文件file的最后10个字符
```

示例：

```
$ seq 1 20 > file.txt
$ head file.txt
$ head -n 2  file.txt
$ head -n +2 file.txt
$ head -c 20 file
```

wc命令:用来计算数字。利用wc指令我们可以计算文件的Byte数、字数或是列数，若不指定文件名称，或是所给予的文件名为"-"，则wc指令会从标准输入设备读取数据。

语法：`wc [-OPTION] [file]`

解读：

```
wc file #计算file文件的行数、字数，以及字节数
wc -l file #显示file文件行数
```

示例：

```
$ echo "hello, word!" > file
$ wc file
$ wc -l file
```

stat命令用于显示文件的状态信息。stat命令的输出信息比ls命令的输出信息要更详细。

语法：`stat [-OPTION] [文件]`

解读：

```
stat file 查看file文件的状态信息
```

示例：

```
$ echo "hello, word!" > file.txt
$ stat file.txt
```

cut命令：用来显示行中的指定部分，删除文件中指定字段。

语法：`cut [-OPTION] [file]`

解读：

```
cut -f 1, 3 file     #提取file文件的1,3列内容
cut -f2 -d";" file   #添加参数-d指定列分隔符
```

示例：

```
$ echo "hello world !" > file.txt
$ cut -f 1, 3 file.txt
```

Vim命令:是从 vi 发展出来的一个文本编辑器。代码补完、编译及错误跳转等方便编程的功能特别丰富，在程序员中被广泛使用。

语法：`vim file`

解读：

```
$ vim hello-world.txt
#插入编辑模式： i键
#切换命令行模式： Esc
#退出： shift + : 后输入 "wq" 保存退出， 输入 "x!" 不保存退出
```

Rsync命令：是一个远程数据同步工具，可通过LAN/WAN快速同步多台主机间的文件。Rsync使用所谓的"Rsync算法"来使本地和远程两个主机之间的文件达到同步，这个算法只传送两个文件的不同部分，而不是每次都整份传送，因此速度相当快。

语法：`rsync [-OPTION] SRC [USER@]host:DEST`

rsync六种不同的工作模式：

1. 拷贝本地文件：当SRC和DES路径信息都不包含有单个冒号":"分隔符时就启动这种工作模式。
2. 使用一个远程**shell**程序（如**rsh**、**ssh**）来实现将本地机器的内容拷贝到远程机器：当DST路径地址包含单个冒号":"分隔符时启动该模式。

3. 使用一个远程**shell**程序（如**rsh**、**ssh**）来实现将远程机器的内容拷贝到本地机器：当SRC地址路径包含单个冒号":"分隔符时启动该模式。
4. 从远程**rsync**服务器中拷贝文件到本地机：当SRC路径信息包含"::"分隔符时启动该模式。
5. 从本地机器拷贝文件到远程**rsync**服务器中：当DST路径信息包含"::"分隔符时启动该模式。
6. 列远程机的文件列表：这类似于rsync传输，不过只要在命令中省略掉本地机信息即可。

解读：

```
#将当前目录的所有文件跨服务器拷贝到biostack@10.10.86.71:/project/biostack文件夹下
$ rsync -avP * biostack@10.10.86.71:/project/biostack
#将biostack@10.10.86.71:/project/biostack目录跨服务器拷贝到当前目录下
$ rsync -avP biostack@10.10.86.71:/project/biostack ./#将
```

示例：

同一个服务器内部文件同步.

```
$ mkdir /home/biostack
$ rsync -avP * /home/biostack
```

文本文件管理命令

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。

语法：`touch[-OPTION][文件]`

示例：

创建一个名为"file"的新的空白文件

```
$ touch file.txt
```

mkdir命令：用来创建目录。该命令创建由dirname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。

语法：`mkdir [-OPTION] [filename]`

示例：

在home文件夹下创建`file.txt`目录

```
$ mkdir  /home/file.txt
```

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。

语法：`cp [-OPTION][源文件][目标文件]`

示例：

将home文件夹下的file文件复制到当前目录

```
$ echo "hello, biostack!" >file.txt
$ mkdir biostack
$ cp file.txt  biostack
```

将目录/home/logs下的所有文件及其子目录复制到目录/home/Desktop中

```
$ cp -r biostack service
$ ls
```

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。

语法：`mv [-OPTION] [源文件] [目标文件]`

命令格式	运行结果
mv 文件名 文件名	将源文件名改为目标文件名
mv 文件名 目录名	将文件移动到目标目录
mv 目录名 目录名	目标目录已存在，将源目录移动到目标目录；目标目录不存在则改名
mv 目录名 文件名	出错

解读：

```
mv file1.txt file2.txt    #将file1.txt命名为file2.txt
mv folder/*  .            #将folder文件夹下的全部文件转移到当前目录下
```

示例：

将file1命名为file2

```
$ mkdir /home/file
$ touch /home/file/file1
$ mv /home/file/file1 /home/file/file2
```

将file文件夹下的全部文件转移到当前目录下

```
$ mv /home/file .
```

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。

语法：`rm [-OPTION] [文件]`

解读：

```
$ rm  file #删除file文件
$ rm -rf folder #强制删除folder文件夹
```

示例：

```
$ mkdir /home/file
$ touch /home/file/file1
$ rm  /home/file/file1
$ rm -rf /home/file
```

常用系统工作命令

date命令：是显示或设置系统时间与日期。

语法：`date [-OPTION][<+时间日期格式>]`

解读：

```
$ date +"%Y-%m-%d"      #显示当前年月日
```

示例：

```
$ date +"%Y-%m-%d"
```

wget命令：用来从指定的URL下载文件。

语法：`wget [-OPTION][URL]`

解读：

```
$ wget ftp://ftp.ncbi.nlm.nih.gov/blast/blastftp.txt             # 使用wget下载单个文件
$ wget -O file.zip ftp://ftp.ncbi.nlm.nih.gov/blast/blastftp.txt # 下载并以不同的文件名保存
$ wget -b ftp://ftp.ncbi.nlm.nih.gov/blast/blastftp.txt          # 使用wget后台下载
```

示例：

```
$ wget ftp://ftp.ncbi.nlm.nih.gov/blast/blastftp.txt
$ wget -O file.zip ftp://ftp.ncbi.nlm.nih.gov/blast/blastftp.txt
$ wget -b ftp://ftp.ncbi.nlm.nih.gov/blast/blastftp.txt
```

kill命令：用来删除执行中的程序或工作。

语法：`kill [选项][进程或作业识别号]`

示例：

```
 $ kill -l #列出所有信号名称

$ ps -ef | grep vim #先用ps查找进程，然后用kill杀掉
root      3268  2884  0 16:21 pts/1    00:00:00 vim install.log
root      3370  2822  0 16:21 pts/0    00:00:00 grep vim

$kill 3268 #杀掉root进程
```

top命令：可以实时动态地查看系统的整体运行情况，是一个综合了多方信息监测系统性能和运行信息的实用工具。通过top命令所提供的互动式界面，用热键可以管理。

语法：`top [-OPTION]`

示例：

```
$ top
```

htop命令：是Linux系统中的一个互动的进程查看器，一个文本模式的应用程序(在控制台或者X终端中)，需要ncurses。与Linux传统的top相比，htop更加人性化。它可让用户交互式操作，支持颜色主题，可横向或纵向滚动浏览进程列表，并支持鼠标操作。

htop相比较top的优势：

1. 可以横向或纵向滚动浏览进程列表，以便看到所有的进程和完整的命令行。
2. 在启动上比top 更快。
3. 杀进程时不需要输入进程号。
4. htop 支持鼠标选中操作（反应不太快）。
5. top 已不再维护。

语法：`htop [-OPTION]`

示例：

```
$ htop
```

linux常用快捷命令

- **tab** #命令或路径等的补全键，linux用的最多的一个快捷键
- **ctrl+a** #光标迅速回到行首
- **ctrl+e** #光标迅速回到行尾
- **ctrl+k** #剪切（删除）光标处到行尾的所有字符
- **ctrl+u** #剪切（删除）光标处到行首的所有字符
- **ctrl+y** #粘贴 ctrl+k、ctrl+u、ctrl+w删除的字符
- **ctrl+c** #中断终端正在执行的任务并开启一个新的一行
- **ctrl+d** #退出当前shell命令行，如果是切换过来的用户，则执行这个命令回退到原用户
- **ctrl+r** #搜索命令行使用过的历史命令记录
- **ctrl+l** #清楚屏幕所有的内容，并开启一个新的一行
- **ctrl+z** #暂停在终端运行的任务,使用"fg"命令可以使暂停恢复
- **!!** #执行上一条命令
- **!pw** #这是一个例子，是执行以pw开头的命令，这里的pw可以换成任何已经执行过的字符
- **!pw:p** #这是一个例子，是仅打印以pw开头的命令，但不执行，最后的那个"p"是命令固定字符
- **!num** #执行历史命令列表的第num条命令，num代指任何数字（前提是历史命令里必须存在）
- **!$** #代指上一条命令的最后一个参数，该命令常用于shell脚本中
- **esc+.** #注意那个"." 意思是获取上一条命令的(以空格为分隔符)最后的部分

3.linux管理员常用命令

系统的关机、重启以及登出

reboot命令：用来重新启动正在运行的Linux操作系统。

语法：`reboot [-OPTION]`

解读：

```
$ reboot         #重开机。
$ reboot -w      #做个重开机的模拟（只有纪录并不会真的重开机）。
```

poweroff命令：用来关闭计算机操作系统并且切断系统电源。

语法：`poweroff [-OPTION]`

解读：

```
$ poweroff        #关机
```

shutdown命令：用来系统关机命令。shutdown指令可以关闭所有程序，并依用户的需要，进行重新开机或关机的动作。

语法：`shutdown [-OPTION] [time]`

解读：

```
$ shutdown +5 "System will shutdown after 5 minutes"  #指定5分钟后关机，同时送出警告信息给登入用户
$ shutdown -h now                                      #指定现在立即关机
```

logout命令:用于退出系统。

语法：`logout`

解读：

```
$ logout          #注销
```

系统管理命令

df命令：用于显示目前在Linux系统上的文件系统的磁盘使用情况统计。

语法：`df [-OPTION]`

解读：

```
$ df              #显示文件系统的磁盘使用情况统计
```

free 命令：显示系统内存的使用情况，包括物理内存、交换内存(swap)和内核缓冲区内存。

语法：`free [-OPTION]`

解读：

```
$ free            #显示内存使用信息
$ free -t         #以总和的形式查询内存的使用信息
$ free -s 10      #每10s 执行一次命令
```

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。

语法：`ifconfig[-OPTION]`

解读：

```
$ ifconfig        #显示网络设备信息（激活状态的）
```

uname命令:用于打印当前系统相关信息（内核版本号、硬件架构、主机名称和操作系统类型等）

语法：`uname [-option]`

解读：

```
$ uname -a          #打印当前系统全部信息
```

uptime命令：能够打印系统总共运行了多长时间和系统的平均负载。uptime命令可以显示的信息显示依次为：现在时间、系统已经运行了多长时间、目前有多少登陆用户、系统在过去的1分钟、5分钟和15分钟内的平均负载。

语法：`uptime[-OPTION]`

解读：

```
$ uptime            #打印系统总共运行了多长时间和系统的平均负载
```

who命令:是显示目前登录系统的用户信息。执行who命令可得知目前有那些用户登入系统，单独执行who命令会列出登入帐号，使用的终端机，登入时间以及从何处登入或正在使用哪个X显示器。

语法：`who [-OPTION] [file]`

解读：

```
$ who               #显示目前登录系统的用户信息
```

history命令:用于显示指定数目的指令命令，读取历史命令文件中的目录到历史命令缓冲区和将历史命令缓冲区中的目录写入命令文件。

语法：`history [-OPTION]`

解读：

```
$ history 10        #显示最近使用的10条历史命令
```

文件权限属性设置

- u 表示该文件的拥有者，g 表示与该文件的拥有者属于同一个群体(group)者，o 表示其他以外的人，a 表示这三者皆是。
- + 表示增加权限、- 表示取消权限、= 表示唯一设定权限。
- r 表示可读取，w 表示可写入，x 表示可执行，X 表示只有当该文件是个子目录或者该文件已经被设定过为可执行。
- 其中a,b,c各为一个数字，分别表示User、Group、及Other的权限。
- **r=4，w=2，x=1**
- 若要rwx属性则4+2+1=7
- 若要rw-属性则4+2=6
- 若要r-x属性则4+1=5。

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。

语法：`chmod [-OPTION] [file]`

解读：

chmod a=rwx file 和 **chmod 777 file**效果相同
chmod ug=rwx,o=x file 和 **chmod 771 file**效果相同

```
$ chmod ugo+r file           #将文件 file 设为所有人皆可读取
$ chmod a+r   file           #将文件 file 设为所有人皆可读取
```

实例

```
$ mkdir       /home/file
$ touch       /home/file/file
$ chmod ugo+r /home/file/file
$ chmod a+r   /home/file/file
```

chown命令：改变某个文件或目录的所有者和所属的组，该命令可以向某个用户授权，使该用户变成指定文件的所有者或者改变文件所属的组。用户可以是用户或者是用户D，用户组可以是组名或组id。文件名可以使由空格分开的文件列表，在文件名中可以包含通配符。只有文件主和超级用户才可以使用该命令。

语法：`chown [-OPTION] [file]`

解读：

```
chown -R biostack:biostack *将目前目录下的所有文件与子目录的拥有者皆设为 biostack，群体的使用者 biostack
```

示例：

修改`file.txt`文件用户归属属性

```
$ su training
$ echo "hello world !" > file.txt
$ chown biostack:biostack file.txt
```

用户和工作组管理

root 用户:超级管理员用户，权限最大 。切换至 root 用户：su命令。

biostack用户: 管理员用户， 使用sudo行使管理员权限。切换至biostack用户：sudo命令。

普通用户: 普通用户具有最小的用户权限，只能在自己用户目录完成操作，90%的生物信息数据分析任务是不需要管理员权限的。

groupadd命令：用于创建一个新的工作组，新工作组的信息将被添加到系统文件中。

语法： `groupadd [-OPTION] [组名]`

示例：

添加`training`用户组

```
# groupadd training
```

groupdel命令:用于删除指定的工作组，本命令要修改的系统文件包括`/ect/group`和`/ect/gshadow`。若该群组中仍包括某些用户，则必须先删除这些用户后，方能删除群组。

语法： `groupdel [组名]`

示例：

删除`training`用户组

```
# groupdel training
```

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。帐号建好之后，再用`passwd`设定帐号的密码．而可用`userdel`删除帐号。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。

语法：`useradd [-OPTION] [用户组]`

示例：

添加用户组`training`，指定登入时的启始目录为`/home/training`，指定用户所属的群组为`training`。

```
# useradd -d /home/training -g training training
```

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。

语法：`userdel [-OPTION] [用户名]`

示例：

删除用户training

```
# usredel training
```

4.Linux 文件共享: Samba 共享服务

添加**Samba**用户:

```
$ sudo smbpasswd -a biostack
```

配置：

```
$ sudo mv /etc/samba/smb.conf  /etc/samba/smb.conf.backup
$ sudo vim /etc/samba/smb.conf
```

添加共享目录：

```
[training]
path = /project/training
public = yes
browsable = yes
valid users = @training
writable = yes
guest ok = yes
read only = no
available = yes
```

修改文件属性：

```
$ cd /project/training
$ chmod -R  775 /project/training
$ chcon -R  -t samba_share_t /project/training
```

重启服务：

```
sudo systemctl enable smb.service
sudo systemctl enable nmb.service
sudo systemctl restart smb.service
sudo systemctl restart nmb.service
```