1. **Linux 简介**
   1. **简介**

免费使用和自由传播的类Unix操作系统，基于POSIX和UNIX的多用户、多任务、支持多线程和多CPU的操作系统。

能运行主要的UNIX工具软件、应用程序和网络协议。支持32位和64位硬件。

Linux继承了Unix以网络为核心的设计思想，是性能稳定的多用户网络操作系统。

* 1. **发行版**

将Linux内核与应用软件做一个打包。

目前市面上较知名有：

|  |
| --- |
| Ubuntu、RedHat、CentOS、Debian、Fedora、SuSE、OpenSUSE、TurboLinux、BluePoint、RedFlag、Xterm、SlackWare等。 |

* 1. **应用领域**

从嵌入式设备到超级计算机，且在服务器领域确定了地位

服务器：

|  |
| --- |
| LAMP（Linux + Apache + MySQL + PHP）  或LNMP（Linux + Nginx+ MySQL + PHP）组合。 |

目前Linux不仅在家庭与企业中使用，在政府中也很受欢迎。

|  |
| --- |
| 巴西联邦政府由于支持Linux而世界闻名。  有新闻报道俄罗斯军队自己制造的Linux发布版的，做为G.H.ost项目已经取得成果.  印度的Kerala联邦计划在向全联邦的高中推广使用Linux。  中华人民共和国为取得技术独立，在龙芯过程中排他性地使用Linux。  在西班牙的一些地区开发了自己的Linux发布版，并且在政府与教育领域广泛使用，如Extremadura地区的gnuLinEx和Andalusia地区的Guadalinex。  葡萄牙同样使用自己的Linux发布版Caixa Mágica，用于Magalh?es笔记本电脑和e-escola政府软件。  法国和德国同样开始逐步采用Linux。 |

* 1. **Linux vs Windows**

|  |  |  |
| --- | --- | --- |
| **比较** | **Windows** | **Linux** |
| 界面 | 界面统一，外壳程序固定所有Windows程序菜单几乎一致，快捷键也几乎相同 | 图形界面风格依发布版不同而不同，可能互不兼容。  GNU/Linux的终端机是从UNIX传承下来，基本命令和操作方法也几乎一致。 |
| 驱动程序 | 驱动程序丰富，版本更新频繁。  默认安装程序里面一般包含有该版本发布时流行的硬件驱动程序，之后所出的新硬件驱动依赖于硬件厂商提供。  对于一些老硬件，如果没有了原配的驱动有时很难支持。  另外，有时硬件厂商未提供所需版本的Windows下的驱动，也会比较头痛。 | 由志愿者开发，由Linux核心开发小组发布，很多硬件厂商基于版权考虑并未提供驱动程序，尽管多数无需手动安装，但是涉及安装则相对复杂，使得 新用户面对驱动程序问题（是否存在和安装方法）会一筹莫展。  但在开源开发模式下，许多老硬件尽管在Windows下很难支持的也容易找到驱动。HP、 Intel、AMD等硬件厂商逐步不同程度支持开源驱动，问题正在得到缓解。 |
| 使用 | 使用较简单，易入门。  图形化界面对没有计算机背景知识的用户使用十分有利。 | 图形界面使用简单，容易入门。  文字界面，需学习才能掌握。 |
| 学习 | 系统构造复杂、变化频繁，且知识、技能淘汰快，深入学习困难。 | 系统构造简单、稳定，且知识、技能传承性好，深入学习相对容易。 |
| 软件 | 每一种特定功能可能都需要商业软件的支持，需要购买相应的授权。 | 大部分软件都可自由获取，同样功能的软件选择较少。 |

1. **Linux 安装**

http://www.runoob.com/linux/linux-install.html

1. **Linux 系统启动过程**

过程可分为5个阶段：

|  |
| --- |
| 内核引导  运行 init  系统初始化  建立终端  用户登录系统 |

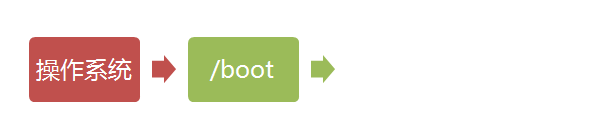
**init程序的类型：**

|  |
| --- |
| SysV: init, CentOS 5之前, 配置文件： /etc/inittab。  Upstart: init,CentOS 6, 配置文件： /etc/inittab, /etc/init/\*.conf。  Systemd： systemd, CentOS 7,配置文件： /usr/lib/systemd/system、 /etc/systemd/system。 |

* 1. **内核引导**

当计算机打开电源后，首先是BIOS(基本输入输出系统)开机自检，照BIOS中设置的启动设备（通常是硬盘）来启动。

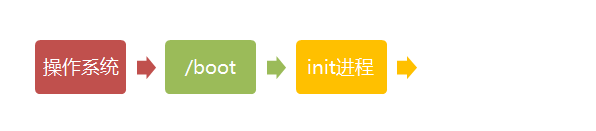
操作系统接管硬件后，首先读入 /boot 目录下的内核文件。



* 1. **运行init**

init 进程是系统所有进程的起点，可把它比拟成系统所有进程的老祖宗，没这个进程，系统中任何进程都不会启动。

init 程序首先需读取配置文件 /etc/inittab。



* 1. **运行级别**

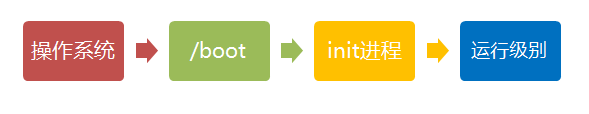
许多程序需开机启动。在Windows叫"服务"（service），在Linux就叫做"守护进程"（daemon）。

init进程的一大任务，就是去运行这些开机启动的程序。

但不同场合需启动不同程序，比如用作服务器时，需要启动Apache，用作桌面就不需要。

Linux允许为不同场合，分配不同开机启动程序，叫做"运行级别"（runlevel）。

就是说，启动时根据"运行级别"，确定要运行哪些程序。



Linux系统有7个运行级别(runlevel)：

|  |
| --- |
| 运行级别0：系统停机状态，系统默认运行级别不能设为0，否则不能正常启动  运行级别1：单用户工作状态，root权限，用于系统维护，禁止远程登陆  运行级别2：多用户状态(没有NFS)  运行级别3：完全的多用户状态(有NFS)，登陆后进入控制台命令行模式  运行级别4：系统未使用，保留  运行级别5：X11控制台，登陆后进入图形GUI模式  运行级别6：系统正常关闭并重启，默认运行级别不能设为6，否则不能正常启动 |

* 1. **系统初始化**

init配置文件中有一行： si::sysinit:/etc/rc.d/rc.sysinit

调用执行了/etc/rc.d/rc.sysinit，而rc.sysinit是一个bash shell的脚本，主要完成一些系统初始化的工作，rc.sysinit是每一个运行级别都要首先运行的重要脚本。

主要完成工作有：激活交换分区，检查磁盘，加载硬件模块及其它一些需优先执行的任务。

|  |
| --- |
| l5:5:wait:/etc/rc.d/rc 5 |

这一行表示以5为参数运行/etc/rc.d/rc

/etc/rc.d/rc是Shell脚本，接受5作为参数，去执行/etc/rc.d/rc5.d/目录下的所有的rc启动脚本，/etc/rc.d/rc5.d/目录中的这些启动脚本实际上都是一些连接文件，不是真正的rc启动脚本，真正的rc启动脚本实际上都是放在/etc/rc.d/init.d/目录下。

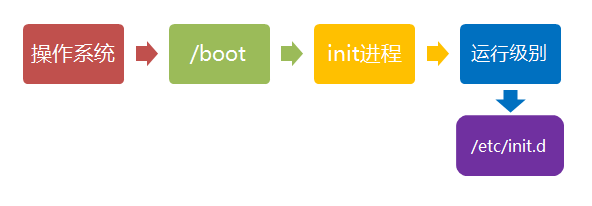
这些rc启动脚本有类似的用法，一般能接受start、stop、restart、status等参数。

/etc/rc.d/rc5.d/中的rc启动脚本通常是K或S开头的连接文件，对以S开头的启动脚本，将以start参数来运行。

而如发现存在相应脚本也存在K打头的连接，且已处于运行态了(以/var/lock/subsys/下的文件作为标志)，则将首先以stop为参数停止这些已经启动了的守护进程，再重新运行。

是为了保证当init改变运行级别时，相关守护进程都将重启。

至于在每个运行级中将运行哪些守护进程，用户可以通过chkconfig或setup中的"System Services"来自行设定。



* 1. **建立终端**

rc执行完毕后，返回init。基本系统环境已设置好了，各种守护进程也已启动了。

init接下来会打开6个终端，以便用户登录系统。在inittab中以下6行就是定义了6个终端：

|  |
| --- |
| 1:2345:respawn:/sbin/mingetty tty1  2:2345:respawn:/sbin/mingetty tty2  3:2345:respawn:/sbin/mingetty tty3  4:2345:respawn:/sbin/mingetty tty4  5:2345:respawn:/sbin/mingetty tty5  6:2345:respawn:/sbin/mingetty tty6 |

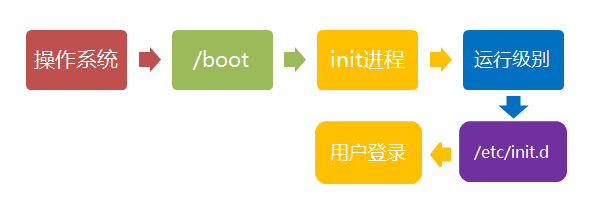
从上面可看出在2、3、4、5的运行级别中都将以respawn方式运行mingetty程序，mingetty程序能打开终端、设置模式。

同时会显示一个文本登录界面，就是经常看到的登录界面，在这个登录界面中会提示用户输入用户名，而用户输入的用户将作为参数传给login程序来验证用户的身份。

* 1. **用户登录系统**

用户登录方式有三种：

|  |
| --- |
| * 命令行登录 * ssh登录 * 图形界面登录 |

对运行级别为5的图形方式用户来说，登录是通过一个图形化的登录界面。

登录成功后可直接进入KDE、Gnome等窗口管理器。

本文主要讲的还是文本方式登录的情况：

当看到mingetty的登录界面时，就可输入用户名和密码来登录系统了。

Linux的账号验证程序是login，login会接收mingetty传来的用户名作为用户名参数。

然后login会对用户名进行分析：

* 如用户名不是root，且存在/etc/nologin文件，login将输出nologin文件内容，然后退出。

通常用来系统维护时防止非root用户登录。只有/etc/securetty中登记了的终端才允许root用户登录，如不存在这个文件，则root可以在任何终端上登录。

/etc/usertty用于对用户作出附加访问限制，如不存在这个文件，则没有其他限制。

* 1. **图形模式与文字模式的切换方式**

Linux预设提供了六个命令窗口终端机让我们来登录。

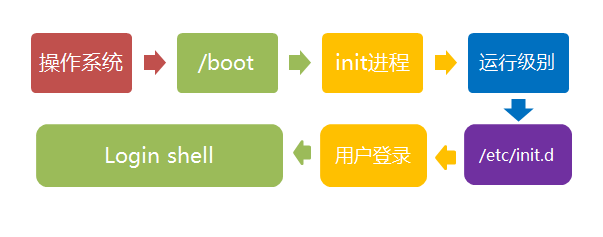
默认登录的是第一个窗口，就是tty1，这六个窗口分别为tty1,tty2 … tty6，可按下Ctrl + Alt + F1 ~ F6 来切换。

如安装了图形界面，默认进入图形界面，就可按Ctrl + Alt + F1 ~ F6来进入其中一个命令窗口界面。

当进入命令窗口界面后再返回图形界面只要按下Ctrl + Alt + F7 就回来了。

如用的vmware 虚拟机，命令窗口切换的快捷键为 Alt + Space + F1~F6.

如在图形界面下请按Alt + Shift + Ctrl + F1~F6 切换至命令窗口。



* 1. **Linux 关机**

linux领域内大多用在服务器上，很少遇到关机操作。毕竟服务器上跑一个服务是永无止境的，除非特殊情况下，不得已才会关机。

正确的关机流程：sync > shutdown > reboot > halt

例如你可运行如下命令关机：

|  |
| --- |
| sync 将数据由内存同步到硬盘中。  shutdown 关机指令，可man shutdown 来看帮助文档。  shutdown –h 10 ‘This server will shutdown after 10 mins’ //10分钟后关机，且会显示在登陆用户的当前屏幕中。  Shutdown –h now 立马关机  Shutdown –h 20:25 系统会在今天20:25关机  Shutdown –h +10 十分钟后关机  Shutdown –r now 系统立马重启  Shutdown –r +10 系统十分钟后重启  reboot : 重启，等同于 shutdown –r now  halt:关闭系统，等同于shutdown –h now 和 poweroff |

不管重启还是关闭系统，首先要运行 sync 命令，把内存中的数据写到磁盘中。

关机的命令有 shutdown –h now halt poweroff 和 init 0 ,

重启系统的命令有 shutdown –r now reboot init 6

1. **Linux 系统目录结构**

* 登录系统后，在当前命令窗口下输入命令：

|  |
| --- |
| ls / |

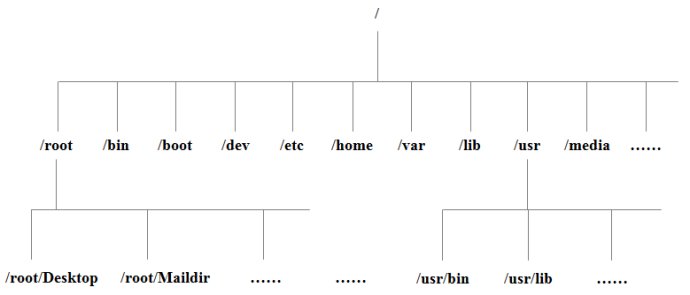
会看到如下图所示:

4_20

注:我的



* 树状目录结构：



* 对目录的解释：

|  |  |
| --- | --- |
| /bin：bin是Binary的缩写, 这个目录存放着最经常使用的命令。  /boot：存放的是启动Linux时使用的一些核心文件，包括一些连接文件及镜像文件。  /dev ：是Device(设备)的缩写, 存放的是Linux的外部设备，在Linux中访问设备和文件的方式相同。  /etc：存放所有系统管理所需的配置文件和子目录。  /home：用户的主目录，Linux中，每个用户都有自己的目录，一般以用户的账号命名的。  /lib：存放系统最基本的动态连接共享库，作用类似于Windows里的DLL文件。几乎所有应用程序都需用到这些共享库。  /lost+found：一般空的，当系统非法关机后，这里就存放一些文件。  /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：为了让用户临时挂载别的文件系统，可将光驱挂载在/mnt/上，然后进入该目录就可查看光驱里的内容。  /opt：给主机额外安装软件所摆放的目录。比如你安装一个ORACLE数据库则就可放到这个目录下。默认空的。  /proc：虚拟目录，是系统内存的映射，可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在内存里，也可直接修改里面的某些文件，  比如可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：  echo 1 > /proc/sys/net/ipv4/icmp\_echo\_ignore\_all  /root：系统管理员，也称作超级权限者的用户主目录。  /sbin：s是Super User的意思，存放的是系统管理员使用的系统管理程序。  /selinux：是Redhat/CentOS特有的目录，Selinux是一个安全机制，类似于windows的防火墙，但这套机制比较复杂，这个目录就是存放selinux相关的文件的。  /srv：存放服务启动后需提取的数据。  /sys：是linux2.6内核的一个很大变化。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时，对应的文件和目录也在内核对象子系统中被创建。  /tmp：存放临时文件。  /usr：重要目录，用户的很多应用程序和文件都放在该目录下，类似与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：存放着在不断扩充着的东西，习惯将常被修改的目录放在这个目录下。包括各种日志文件。 |

linux系统中，有几个目录较重要，平时需注意不要误删除或随意更改内部文件。

* /etc：系统中的配置文件，如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: 系统预设的执行文件的放置目录，如 ls 就是在/bin/ls 目录下的。

注：/bin, /usr/bin 是给系统用户使用的指令（除root外的普通用户）

而/sbin, /usr/sbin 则是给root使用的指令。

* /var： 非常重要的目录，程序日志被记录到该目录下，具体在/var/log 目录下，另外mail的预设放置也是在这里。

**笔记列表**

在 Linux 或 Unix 操作系统中，所有文件和目录都被组织成以一个根节点开始的倒置的树状结构。

文件系统的最顶层是由根目录(/)开始。根目录之下的既可是目录，也可是文件，而每一个目录中又可包含子目录文件。如此反复就可以构成一个庞大的文件系统。

在Linux文件系统中有两个特殊的目录:

一个用户所在的工作目录，也叫当前目录，可使用一个点 . 来表示；

另一个是当前目录的上一级目录，也叫父目录，可以使用两个点 .. 来表示。

|  |
| --- |
| . ：代表当前目录，也可以使用 ./ 来表示；  .. ：代表上一层目录，也可以 ../ 来代表。 |

如一个目录或文件名以一个点 . 开始，表这个目录或文件是一个隐藏目录或文件(如：.bashrc)。即以默认方式查找时，不显示该目录或文件。

1. **Linux 忘记密码解决方法**

进入单用户模式更改一下root密码即可。

http://www.runoob.com/linux/linux-forget-password.html

1. **Linux 远程登录(Putty没看)**

需远程登录到Linux服务器来管理维护系统。

Linux中是通过ssh服务实现的远程登录功能，默认ssh服务端口号为 22。

Window系统上 Linux 远程登录客户端有SecureCRT, Putty, SSH Secure Shell等，本文以Putty为例来登录远程服务器。

**6.1 使用密钥认证机制远程登录linux**

SSH 为 Secure Shell 的缩写，由 IETF 的网络工作小组（Network Working Group）所制定。

SSH 为建立在应用层和传输层基础上的安全协议。

首先使用工具 PUTTYGEN.EXE 生成密钥对。打开工具PUTTYGEN.EXE后如下图所示：

1. **Linux 文件基本属性**

**7.1 基本属性**

多用户系统，不同用户处于不同地位，拥有不同权限。

为了保护系统安全性，Linux系统对不同用户访问同一文件（包括目录文件）的权限做了不同规定。

在Linux中可使用ll或者ls –l来显示文件属性及文件所属的用户和组，如：

|  |
| --- |
| [root@www /]# ls -l  total 64  dr-xr-xr-x 2 root root 4096 Dec 14 2012 bin  dr-xr-xr-x 4 root root 4096 Apr 19 2012 boot  …… |

实例中，bin文件的第一个属性用"d"表示。

"d"代表该文件是目录文件。

* Linux中第一个字符代表这个文件是目录、文件或链接文件等等。

|  |
| --- |
| 当为[ d ]则是目录  当为[ - ]则是文件；  若是[ l ]则表示为链接文档(link file)；  若是[ b ]则表示为装置文件里的可供储存的接口设备(可随机存取装置)；  若是[ c ]则表示为装置文件里面的串行端口设备，例如键盘、鼠标(一次性读取装置)。 |

* 接下来的字符中，以三个为一组，且均为『rwx』 的三个参数的组合。

其中，[ r ]代表可读(read)、[ w ]代表可写(write)、[ x ]代表可执行(execute)。

要注意这三个权限的位置不会改变，如没有权限，就会出现减号[ - ]而已。

* 每个文件的属性由左边第一部分的10个字符来确定（如下图）。



从左至右用0-9这些数字来表示。

|  |
| --- |
| 第0位确定文件类型  第1-3位确定属主（该文件的所有者）拥有该文件的权限。  第4-6位确定属组（所有者的同组用户）拥有该文件的权限，  第7-9位确定其他用户拥有该文件的权限。  第1、4、7位表读权限，如用"r"字符表示，则有读权限，如用"-"字符表示，则没有读权限；  第2、5、8位表写权限，如果用"w"字符表示，则有写权限，如果用"-"字符表示没有写权限；  第3、6、9位表可执行权限，如用"x"字符表示，则有执行权限，如用"-"字符表示，则没执行权限。 |

**7.2 Linux文件属主和属组**

|  |
| --- |
| [root@www /]# ls -l  total 64  drwxr-xr-x 2 root root 4096 Feb 15 14:46 cron  drwxr-xr-x 3 mysql mysql 4096 Apr 21 2014 mysql  …… |

对于文件来说，都有一个特定的所有者，就是对该文件具有所有权的用户。

同时，在Linux系统中，用户是按组分类的，一个用户属于一个或多个组。

文件所有者以外的用户又以分为文件所有者的同组用户和其他用户。

，Linux系统按文件所有者、文件所有者同组用户和其他用户来规定了不同的文件访问权限。

以上实例中，mysql 文件是目录文件，属主和属组都为 mysql，属主有可读、可写、可执行的权限；

与属主同组的其他用户有可读和可执行的权限；

其他用户也有可读和可执行的权限。

对于 root 用户来说，一般文件权限对其不起作用。

**7.3 更改文件属性**

1、chgrp：更改文件属组(CHange GRouP)

* 语法：

|  |
| --- |
| chgrp [-R] 属组名文件名 |

* 参数选项

-R：递归更改文件属组，就是在更改某个目录文件的属组时，如果加上-R的参数，那么该目录下的所有文件的属组都会更改

2、chown：更改文件属主，也可同时更改文件属组(CHange OWNer)

语法：

|  |
| --- |
| chown [–R] 属主名 文件名  chown [-R] 属主名：属组名 文件名 |

进入 /root 目录（~）将install.log的拥有者改为bin这个账号：

|  |
| --- |
| [root@www ~] cd ~  [root@www ~]# chown bin install.log  [root@www ~]# ls -l  -rw-r--r-- 1 bin users 68495 Jun 25 08:53 install.log |

将install.log的拥有者与群组改回为root：

|  |
| --- |
| [root@www ~]# chown root:root install.log  [root@www ~]# ls -l  -rw-r--r-- 1 root root 68495 Jun 25 08:53 install.log |

3、chmod：更改文件9个属性(CHange MODe)

Linux文件属性有两种设置方法，一种是数字，一种是符号。

Linux文件的基本权限就有九个，分别是owner/group/others三种身份各有自己的read/write/execute权限。

可使用数字代表各个权限：r:4 w:2 x:1

每种身份(owner/group/others)各自的三个权限(r/w/x)分数是需要累加的，例如当权限为： [-rwxrwx---] 分数则是：

|  |
| --- |
| owner = rwx = 4+2+1 = 7  group = rwx = 4+2+1 = 7  others= --- = 0+0+0 = 0 |

所以等一下设定权限的变更时，该文件的权限数字就是770啦！

变更权限的指令chmod的语法是这样的：

|  |
| --- |
| chmod [-R] xyz 文件或目录 |

选项与参数：

|  |
| --- |
| xyz : 就是刚刚提到的数字类型的权限属性，为 rwx 属性数值的相加。  -R : 进行递归(recursive)的持续变更，亦即连同次目录下的所有文件都会变更 |

举例来说，如果要将.bashrc这个文件所有的权限都设定启用，那么命令如下：

|  |
| --- |
| [root@www ~]# ls -al .bashrc  -rw-r--r-- 1 root root 395 Jul 4 11:45 .bashrc  [root@www ~]# chmod 777 .bashrc  [root@www ~]# ls -al .bashrc  -rwxrwxrwx 1 root root 395 Jul 4 11:45 .bashrc |

那如要将权限变成 -rwxr-xr-- 呢？

那么权限的分数就成为 [4+2+1][4+0+1][4+0+0]=754。

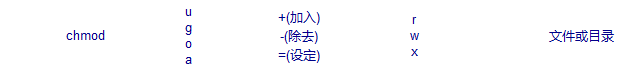
**符号类型改变文件权限**

1. user (2)group (3)others三种身份

u, g, o来代表三种身份权限！

此外，a 代表 all 亦即全部身份！

读写的权限就可写成r, w, x！也就是可使用底下的方式来看：



如需将文件权限设置为 -rwxr-xr-- ，可使用 chmod u=rwx,g=rx,o=r 文件名 来设定:

|  |
| --- |
| [root@www ~]# ls -al .bashrc  -rwxr-xr-x 1 root root 395 Jul 4 11:45 .bashrc  [root@www ~]# chmod a+w .bashrc  [root@www ~]# ls -al .bashrc  -rwxrwxrwx 1 root root 395 Jul 4 11:45 .bashrc |

而如是要将权限去掉而不改变其他已存在的权限呢？例如要拿掉全部人的可执行权限，则：

|  |
| --- |
| [root@www ~]# chmod a-x .bashrc  [root@www ~]# ls -al .bashrc  -rw-rw-rw- 1 root root 395 Jul 4 11:45 .bashrc |

1. **Linux 文件与目录管理**

Linux的目录为树状结构，最顶级的目录为根目录 /。

其他目录通过挂载可将它们添加到树中，通过解除挂载可移除它们。

在开始本教程前需先知道什么是绝对路径与相对路径。

|  |
| --- |
| * 绝对路径：由根目录 / 写起，例如： /usr/share/doc 这个目录。 * 相对路径：不由 / 写起，例如由 /usr/share/doc 要到 * /usr/share/man 底下时，可以写成： cd ../man 这就是相对路径的写法啦！   **./**  当前目录  **../** 上级目录 |

**8.1 处理目录的常用命令**

|  |
| --- |
| ls: 列出目录  cd：切换目录  pwd：显示目前的目录  mkdir：创建一个新的目录  rmdir：删除一个空的目录  cp: 复制文件或目录  rm: 移除文件或目录 |

你可以使用 man [命令] 来查看各个命令的使用文档，如 ：man cp。

* **ls (列出目录)**

最常被运行。

* 语法：

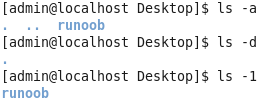
|  |
| --- |
| [root@www ~]# ls [-aAdfFhilnrRSt] 目录名称  [root@www ~]# ls [--color={never,auto,always}] 目录名称  [root@www ~]# ls [--full-time] 目录名称 |

* 选项与参数

|  |
| --- |
| -a ：全部文件，连同隐藏档( 开头为 . 的文件) 一起列出来(常用)  -d ：仅列出目录本身，而不是列出目录内的文件数据(常用)  -l ：长数据串列出，包含文件属性与权限等数据；(常用) |

将家目录下的所有文件列出来(含属性与隐藏档)

|  |
| --- |
| [root@www ~]# ls -al ~ |



* **cd (切换目录)**

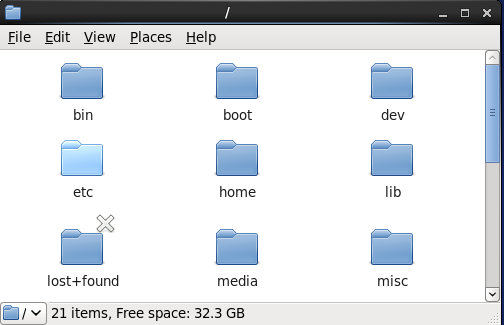
Change Directory的缩写，变换工作目录。

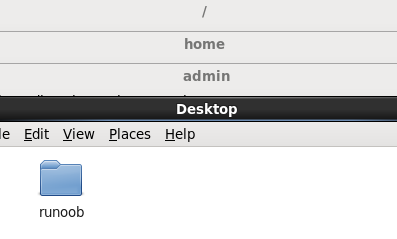
语法：

|  |
| --- |
| cd [相对路径或绝对路径] |

|  |
| --- |
| #使用 mkdir 命令在桌面创建 runoob 目录  [admin@localhost Desktop]# mkdir runoob  #使用绝对路径切换到 runoob 目录    #使用相对路径切换到 runoob 目录    # 表示回到自己的家目录，亦即是 /home/admin这个目录  [admin@localhost runoob]# cd ~  # 表示去到目前的上一级目录，亦即是 /home/admin的上一级目录的意思； |

* /：指的是computer-FileSystem





**pwd (显示目前所在目录)**

Print Working Directory缩写

|  |
| --- |
| [admin@localhost ~]$ pwd [-P]  选项与参数：  -P ：显示出确实的路径，而非使用连结 (link) 路径。  范例：单纯显示当前目录：  [admin@localhost ~]$ pwd  /home/admin <== 显示当前目录  范例：显示实际工作目录，而非连结档本身的目录名而已  [admin@localhost ~]$ cd /var/mail <==注意，/var/mail是一个连结档  [admin@localhost mail] $ pwd  /var/mail <==列出当前工作目录  [admin@localhost mail]# pwd -P  /var/spool/mail <==加上 pwd –P，显示正确的完整路径  [admin@localhost mail]# ls -ld /var/mail  lrwxrwxrwx 1 root root 10 Sep 4 17:54 /var/mail -> spool/mail  #因 /var/mail 是连结档，连结到 /var/spool/mail  #加上 pwd -P 选项后，不以连结档的数据显示，而是显示正确的完整路径！ |

* **mkdir (创建新目录)**

make directory

* 语法：

|  |
| --- |
| mkdir [-mp] 目录名称 |

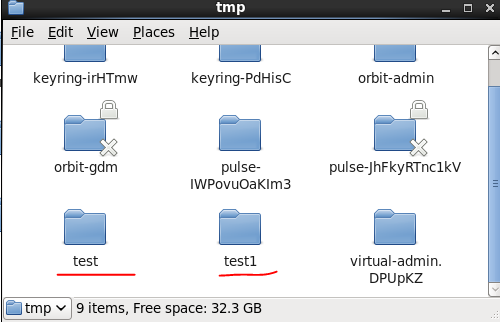
* 选项与参数：

|  |
| --- |
| -m ：配置文件权限！直接配置，不需看默认权限 (umask) 的脸色～  -p ：将所需要的目录(包含上一级目录)递归创建起来！ |

* 范例：到/tmp下尝试创建数个新目录：

|  |
| --- |
| [admin@localhost ~]# cd /tmp  [admin@localhost tmp]# mkdir test <==创建一名为 test 的新目录  [admin@localhost tmp]# mkdir test1/test2/test3/test4  mkdir: cannot create directory `test1/test2/test3/test4':  No such file or directory <== 没办法直接创建此目录啊！  [admin@localhost tmp]# mkdir -p test1/test2/test3/test4 |

加了 -p 选项，可自行帮你在当前目录下创建多层目录！



* 范例：创建权限为rwx--x--x的目录

|  |
| --- |
| [admin@localhost tmp]# mkdir -m 711 test2 |

上面权限部分，如没有加上 -m 来强制配置属性，系统会使用默认属性。

如使用 -m ，如上例我们给予 -m 711 来给予新的目录 drwx--x--x 的权限。

* **rmdir (删除空目录)**
* 语法：

|  |
| --- |
| rmdir [-p] 目录名称 |

* 选项与参数：

|  |
| --- |
| -p ：连同上一级『空的』目录也删除 |

* 范例：将於mkdir范例中创建的目录(/tmp底下)删除掉！

|  |
| --- |
| [admin@localhost tmp]# mkdir test <==创建一名为 test 的新目录  [admin@localhost tmp]# mkdir -p test1/test2/test3/test4  [admin@localhost tmp]# ls -l <==看有多少目录存在  drwxr-xr-x 3 root root 4096 Jul 18 12:50 test  drwxr-xr-x 3 root root 4096 Jul 18 12:53 test1  [admin@localhost tmp]# rmdir test <==可直接删除掉，没问题  [admin@localhost tmp]# rmdir test1 <==因为尚有内容，所以无法删除！  rmdir: `test1': Directory not empty  [admin@localhost tmp]# rmdir –p test1/test2/test3/test4  [admin@localhost tmp]# ls -l <==底下的输出中test与test1不见了！ |

利用 -p，立刻就可将 test1/test2/test3/test4 一次删除。

rmdir 仅能删除空目录，可用 rm 来删除非空目录。

* **cp (复制文件或目录)(** **CoPy)**
* 语法:

|  |
| --- |
| [root@www ~]# cp [-adfilprsu] 来源档(source) 目标档(destination)  [root@www ~]# cp [options] source1 source2 source3 .... directory |

* 选项与参数：

|  |
| --- |
| -a：相当於 -pdr 的意思，至於 pdr 请参考下列说明；(常用)  -d：若来源档为连结档的属性(link file)，则复制连结档属性而非文件本身；  -f：为强制(force)的意思，若目标文件已存在且无法开启，则移除后再尝试一次；  -i：若目标档(destination)已存在，在覆盖时会先询问动作的进行(常用)  -l：进行硬式连结(hard link)的连结档创建，而非复制文件本身；  -p：连同文件属性一起复制过去，非使用默认属性(备份常用)；  -r：递归持续复制，用於目录的复制行为；(常用)  -s：复制成为符号连结档 (symbolic link)，亦即『捷径』文件；  -u：若 destination 比 source 旧才升级 destination ！ |

将 /home/admin目录下的 public 复制到 /tmp 下，并命名为 bashrc



* **rm (移除文件或目录)**
* 语法：

rm [-fir] 文件或目录

* 选项与参数：

|  |
| --- |
| -f ：force ，忽略不存在的文件，不会出现警告信息；  -i ：互动模式，在删除前会询问使用者是否动作  -r ：递归删除！最常用在目录的删除了！这是非常危险的选项！！！ |

* 将刚刚在 cp 的范例中创建的 bashrc 删除掉！



//强制删除

如果加上 -i 的选项就会主动询问喔，避免你删除到错误的档名！

* **mv (移动文件与目录，或修改名称)**
* 语法：

|  |
| --- |
| [admin@localhost ~]# mv [-fiu] source destination  [r admin@localhost ~]# mv [options] source1 source2 source3 .... directory |

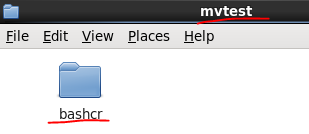
* 选项与参数：

|  |
| --- |
| -f ：force的意思，如目标文件已存在，不会询问而直接覆盖；  -i ：若目标文件 (destination) 已存在，就会询问是否覆盖！  -u ：若目标文件已存在，且 source 较新，才会升级 (update) |

* 复制一文件，创建一目录，将文件移到目录中

|  |
| --- |
| [admin@localhost ~]# cd /tmp  ~~[root@www tmp]# cp ~/.bashrc bashrc~~  [r admin@localhost tmp]# mkdir mvtest  [admin@localhost tmp]# mv bashrc mvtest |

将某个文件移动到某个目录去！



将刚刚的目录名称**更名**为 mvtest2

|  |
| --- |
| [admin@localhost tmp]# mv mvtest mvtest2 |

**8.2 Linux 文件内容查看**

|  |
| --- |
| cat 由第一行开始显示文件内容  tac 从最后一行开始显示，可以看出 tac 是 cat 的倒著写！  nl 显示的时候，顺道输出行号！  more 一页一页的显示文件内容  less 与 more 类似，但是比 more 更好的是，他可以往前翻页！  head 只看头几行  tail 只看尾巴几行 |

可使用 man [命令]来查看各个命令的使用文档，如 ：man cp。

* **cat(CATenate-连接)**

由第一行开始显示文件内容

* 语法：cat [-AbEnTv]
* 选项与参数：

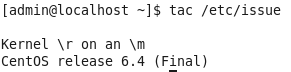
|  |
| --- |
| -A ：相当於 -vET 的整合选项，可列出一些特殊字符而不是空白而已；  -b ：列出行号，仅针对非空白行做行号显示，空白行不标行号！  -E ：将结尾的断行字节 $ 显示出来；  -n ：列印出行号，连同空白行也会有行号，与 -b 的选项不同；  -T ：将 [tab] 按键以 ^I 显示出来；  -v ：列出一些看不出来的特殊字符 |

* 检看 /etc/issue 这个文件的内容：



* **tac**

tac与cat命令刚好相反，文件内容从最后一行开始显示，可以看出 tac 是 cat 的倒着写！如：



* **nl**

显示行号

* 语法：

|  |
| --- |
| nl [-bnw] 文件 |

* 选项与参数：

|  |
| --- |
| -b ：指定行号指定的方式，主要有两种：  -b a ：表示不论是否为空行，也同样列出行号(类似 cat -n)；  -b t ：如果有空行，空的那一行不要列出行号(默认值)；  -n ：列出行号表示的方法，主要有三种：  -n ln ：行号在萤幕的最左方显示；  -n rn ：行号在自己栏位的最右方显示，且不加 0 ；  -n rz ：行号在自己栏位的最右方显示，且加 0 ；  -w ：行号栏位的占用的位数。 |

* 范例一：用 nl 列出 /etc/issue 的内容

|  |
| --- |
| [admin@localhost ~]# nl /etc/issue  1 CentOS release 6.4 (Final)  2 Kernel \r on an \m |

* **more**

一页一页翻动

|  |
| --- |
| [admin@localhost ~]# more /etc/man.config  #  # Generated automatically from man.conf.in by the  # configure script.  #  # man.conf from man-1.6d  ....(中间省略)....  --More--(28%) <== 重点在这一行喔！你的光标也会在这里等待你的命令 |

more 这个程序运行过程中，有几个按键可按：

|  |
| --- |
| 空白键 (space)：代表向下翻一页；  Enter ：代表向下翻『一行』；  /字串 ：代表在这个显示的内容当中，向下搜寻『字串』这个关键字；  :f ：立刻显示出档名以及目前显示的行数；  q ：代表立刻离开 more ，不再显示该文件内容。  b 或 [ctrl]-b ：代表往回翻页，不过这动作只对文件有用，对管线无用。 |

* **less**

一页一页翻动，以下实例输出/etc/man.config文件的内容：

|  |
| --- |
| [[admin@localhost ~]# less /etc/man.config  #  # Generated automatically from man.conf.in by the  # configure script.  #  # man.conf from man-1.6d  ....(中间省略)....  : <== 这里可以等待你输入命令！ |

|  |
| --- |
| less运行时可输入的命令有：  空白键 ：向下翻一页；  [pagedown]：向下翻一页；  [pageup] ：向上翻一页；  /字串 ：向下搜寻『字串』的功能；  ?字串 ：向上搜寻『字串』的功能；  n ：重复前一个搜寻 (与 / 或 ? 有关！)  N ：反向的重复前一个搜寻 (与 / 或 ? 有关！)  q ：离开 less 这个程序； |

* **head**

取出文件前几行

* 语法：

|  |
| --- |
| head [-n number] 文件 |

* 选项与参数：

-n ：后接数字，代表显示几行

|  |
| --- |
| [[admin@localhost ~]# head /etc/man.config |

默认情况中，显示前面 10 行！若要显示前 20 行，就得这样：

|  |
| --- |
| [admin@localhost ~]# head -n 20 /etc/man.config |

* **tail**

取出文件后面几行

* 语法：

|  |
| --- |
| tail [-n number] 文件 |

* 选项与参数：

|  |
| --- |
| -n ：后面接数字，代表显示几行的意思  -f ：表示持续侦测后面所接的档名，要等到按下[ctrl]-c才会结束tail的侦测 |

|  |
| --- |
| [admin@localhost ~]# tail /etc/man.config  # 默认情况，显示最后十行！若要显示最后 20 行，就得要：  [admin@localhost ~]# tail -n 20 /etc/man.config |

1. **Linux 用户和用户组管理**

多用户多任务的分时操作系统，任何一个要使用系统资源的用户，都必须先向系统管理员申请账号，然后以这个账号的身份进入系统。

用户的账号一方面可帮助系统管理员对使用系统的用户进行跟踪，并控制他们对系统资源的访问；

另一方面也可帮助用户组织文件，并为用户提供安全性保护。

每个用户账号都拥有一个惟一的用户名和各自的口令。

用户在登录时键入正确的用户名和口令后，就能够进入系统和自己的主目录。

实现用户账号的管理，要完成的工作：

|  |
| --- |
| 用户账号的添加、删除与修改。  用户口令的管理。  用户组的管理。 |

**一、Linux系统用户账号的管理**

用户账号的添加、修改和删除。

就是在系统中创建一个新账号，为新账号分配用户号、用户组、主目录和登录Shell等资源。刚添加的账号是被锁定的，无法使用。

* **添加新的用户账号(useradd)**
* useradd 选项 用户名
* 参数说明：

|  |
| --- |
| **选项:**  -c<备注>：加上备注文字。备注文字会保存在passwd的备注栏位中；  -d 指定用户主目录，如此目录不存在，则同时使用-m选项，可创建主目录。  -D：变更预设值；  -e<有效期限>：指定帐号的有效期限；  -f<缓冲天数>：指定在密码过期后多少天即关闭该帐号；  -g<用户组>：指定用户所属的用户组。  -G<用户组>：指定用户所属的附加组。  -m：自动建立用户的登入目录；  -M：不要自动建立用户的登入目录；  -n：取消建立以用户名称为名的群组；  -r：建立系统帐号；  -s<shell>：指定用户登入后所使用的shell；  -u<uid>：指定用户id。 |

* 实例1

|  |
| --- |
| # useradd –d /usr/sam -m sam |

创建了一个用户sam，其中-d和-m选项用来为登录名sam产生一个主目录/usr/sam（/usr为默认的用户主目录所在的父目录）。

* 实例2

|  |
| --- |
| # useradd –s /bin/sh -g group –G adm, root gem |

新建一个用户gem，该用户的登录Shell是 /bin/sh，属于group用户组(主组)，同时又属于adm和root用户组。

增加用户账号就是在/etc/passwd文件中为新用户增加一条记录，同时更新其他系统文件如/etc/shadow, /etc/group等。

Linux提供了集成的系统管理工具userconf，可用来对用户账号进行统一管理。

* **删除帐号(userdel)**

如一个用户的账号不再使用，可从系统中删除。

删除用户账号就是要将/etc/passwd等系统文件中的该用户记录删除，必要时还删除用户的主目录。

删除一个已有的用户账号使用userdel命令，其格式如下：

|  |
| --- |
| userdel 选项 用户名 |

常用的选项是-r，作用是把用户的主目录一起删除。

例如：

|  |
| --- |
| # userdel sam |

此命令删除用户sam在系统文件中（主要是/etc/passwd, /etc/shadow, /etc/group等）的记录，同时删除用户的主目录。

* **修改帐号(usermod)**

就是根据实际情况更改用户的有关属性，如用户号、主目录、用户组、登录Shell等。

格式：

|  |
| --- |
| usermod 选项 用户名 |

常用选项包括-c, -d, -m, -g, -G, -s, -u及-o等，选项意义与useradd命令中的选项一样，可为用户指定新的资源值。

另外，有些系统可使用选项：

-l 新用户名

这个选项指定一个新账号，即将原来的用户名改为新用户名。

例如：

|  |
| --- |
| # usermod -s /bin/ksh -d /home/z –g developer sam |

此命令将用户sam的登录Shell修改为ksh，主目录改为/home/z，用户组改为developer。

* **用户口令的管理(passwd)**

用户账号刚创建时没口令，被系统锁定，无法使用，须为其指定口令后才可使用，即使是指定空口令。

指定和修改用户口令的Shell命令是passwd。

超级用户可为自己和其他用户指定口令，普通用户只能用它修改自己的口令。

命令的格式为：

|  |
| --- |
| passwd 选项 用户名 |

可使用的选项：

|  |
| --- |
| -l 锁定口令，即禁用账号。  -u 口令解锁。  -d 使账号无口令。  -f 强迫用户下次登录时修改口令。 |

如默认用户名，则修改当前用户的口令。

例如，假设当前用户是sam，则下面的命令修改该用户自己的口令：

|  |
| --- |
| $ passwd  Old password:\*\*\*\*\*\*  New password:\*\*\*\*\*\*\*  Re-enter new password:\*\*\*\*\*\*\* |

如是超级用户，可以用下列形式指定任何用户的口令：

|  |
| --- |
| # passwd sam  New password:\*\*\*\*\*\*\*  Re-enter new password:\*\*\*\*\*\*\* |

普通用户修改自己口令时，passwd命令会先询问原口令，验证后再要求用户输入两遍新口令，如果两次输入的口令一致，则将这个口令指定给用户；

而超级用户为用户指定口令时，就不需知道原口令。

为了系统安全起见，用户应该选择较复杂的口令，例如最好使用8位长的口令，口令中包含有大写、小写字母和数字，并且应该与姓名、生日等不相同。

为用户指定空口令：

|  |
| --- |
| # passwd -d sam |

此命令将用户sam的口令删除，sam下一次登录时，系统就不再询问口令。

passwd命令还可以用-l(lock)选项锁定某一用户，使其不能登录，例如：

# passwd -l sam

**二、Linux系统用户组的管理**

每个用户都有一个用户组，系统可对一个用户组中的所有用户进行集中管理。

不同Linux 系统对用户组的规定有所不同，如Linux下的用户属于与它同名的用户组，这个用户组在创建用户时同时创建。

用户组的添加、删除和修改。实际上就是对/etc/group文件的更新。

1. **增加一个新的用户组(groupadd)**

格式：

|  |
| --- |
| groupadd 选项 用户组 |

**选项**：

|  |
| --- |
| -g GID 指定新用户组的组标识号（GID）。  -o 一般与-g选项同时使用，表新用户组的GID可与系统已有用户组的GID相同。 |

实例1：

|  |
| --- |
| # groupadd group1 |

向系统中增加了一个新组group1，新组组标识号是在当前已有最大组标识号的基础上加1。

实例2：

|  |
| --- |
| # groupadd -g 101 group2 |

向系统中增加了一个新组group2，同时指定新组的组标识号是101。

1. **删除一个已有用户组(groupdel)**

格式：

|  |
| --- |
| groupdel 用户组 |

例如：

|  |
| --- |
| # groupdel group1 |

1. **修改用户组的属性(groupmod)**

语法：

|  |
| --- |
| groupmod 选项 用户组 |

**选项**：

|  |
| --- |
| -g GID 为用户组指定新的组标识号。  -o 与-g选项同时使用，用户组的新GID可与系统已有用户组的GID相同。  -n 新用户组 将用户组的名字改为新名字 |

* 实例1：

|  |
| --- |
| # groupmod -g 102 group2 |

将组group2的组标识号修改为102。

* 实例2：

|  |
| --- |
| # groupmod –g 10000 -n group3 group2 |

将组group2的标识号改为10000，组名修改为group3。

1. **用户在用户组间切换(newgrp)**

如一个用户同时属于多个用户组，用户可在用户组间切换，以便具有其他用户组的权限。

例如：

|  |
| --- |
| $ newgrp root // root:目的用户组 |

将当前用户切换到root用户组，前提条件是root用户组确实是该用户的主组或附加组。

类似于用户账号的管理，用户组的管理也可通过集成的系统管理工具来完成。

**三、与用户账号有关的系统文件**

用户管理有多种方法，实际上都是对有关的系统文件进行修改。

与用户和用户组相关的信息都存放在一些系统文件中，包括/etc/passwd, /etc/shadow, /etc/group等。

下面分别介绍这些文件的内容。

1. **/etc/passwd:最重要**

每个用户都在/etc/passwd文件中有一个对应的记录行，记录了用户的基本属性。

这个文件对所有用户可读。

内容类似下面的例子：

|  |
| --- |
| ＃ cat /etc/passwd  root:x:0:0:Superuser:/:  daemon:x:1:1:System daemons:/etc:  bin:x:2:2:Owner of system commands:/bin:  sys:x:3:3:Owner of system files:/usr/sys:  adm:x:4:4:System accounting:/usr/adm:  uucp:x:5:5:UUCP administrator:/usr/lib/uucp:  auth:x:7:21:Authentication administrator:/tcb/files/auth:  cron:x:9:16:Cron daemon:/usr/spool/cron:  listen:x:37:4:Network daemon:/usr/net/nls:  lp:x:71:18:Printer administrator:/usr/spool/lp:  sam:x:200:50:Sam san:/usr/sam:/bin/sh |

从上面例子可看到，/etc/passwd中一行记录对应着一个用户

每行记录又被冒号(:)分隔为7个字段，其格式和具体含义如下：

|  |
| --- |
| 用户名:口令:用户标识号:组标识号:注释性描述:主目录:登录Shell |

1）"用户名"是代表用户账号的字符串。

通常长度不超过8个字符，且由大小写字母和/或数字组成。

登录名中不能有冒号(:)，因冒号在这里是分隔符。

为了兼容起见，登录名中最好不要包含点字符(.)，并且不使用连字符(-)和加号(+)打头。

2）“口令”: 加密后的用户口令字。

虽然这个字段存放的只是用户口令的加密串，不是明文，但由于/etc/passwd文件对所有用户可读，所以这仍是一个安全隐患。

因此，现在许多Linux 系统（如SVR4）都使用了shadow技术，把真正加密后的用户口令字存放到/etc/shadow文件中，而在/etc/passwd文件的口令字段中只存放一个特殊字符，例如“x”或“\*”。

3）“用户标识号”: 整数，系统内部用它来标识用户。

与用户名一一对应。

如几个用户名对应的用户标识号一样，系统内部将把它们视为同一个用户，但它们可有不同的口令、主目录及登录Shell等。

通常用户标识号的取值范围是0～65535。

0是超级用户root的标识号，1～99由系统保留，作为管理账号，普通用户的标识号从100开始。在Linux系统中，这个界限是500。

4）“组标识号”: 记录用户所属的用户组。

对应着/etc/group文件中的一条记录。

5)“注释性描述”: 记录着用户的一些个人情况。

例如用户的真实姓名、电话、地址等，并没有什么实际用途。

在不同Linux 系统中，这个字段的格式没有统一。

在许多Linux系统中，该字段存放一段任意的注释性描述文字，用做finger命令的输出。

6)“主目录”： 用户的起始工作目录。

用户登录到系统后所处的目录。

大多数系统中，各用户的主目录都被组织在同一个特定的目录下，而用户主目录的名称就是该用户的登录名。

各用户对自己的主目录有读、写、执行（搜索）权限，其他用户对此目录的访问权限则根据具体情况设置。

7)用户登录后，要启动一个进程，将用户的操作传给内核，这个进程是用户登录到系统后运行的命令解释器或某个特定的程序，即Shell。

Shell是用户与Linux系统间的接口。

Linux的Shell有多种，每种都有不同特点。

常用的有sh(Bourne Shell), csh(C Shell), ksh(Korn Shell), tcsh(TENEX/TOPS-20 type C Shell), bash(Bourne Again Shell)等。

系统管理员可根据系统情况和用户习惯为用户指定某个Shell。

如不指定Shell，系统使用sh为默认的登录Shell，即这个字段的值为/bin/sh。

用户的登录Shell也可指定为某个特定的程序（此程序不是一个命令解释器）。

利用这一特点，可限制用户只能运行指定的应用程序，在该应用程序运行结束后，用户就自动退出了系统。

有些Linux 系统要求只有那些在系统中登记了的程序才能出现在这个字段中。

8)伪用户（psuedo users）。

在/etc/passwd文件中也占有一条记录，但不能登录，因为它们的登录Shell为空。

存在主要是方便系统管理，满足相应的系统进程对文件属主的要求。

常见的**伪用户**：

|  |
| --- |
| 伪 用 户 含 义  bin 拥有可执行的用户命令文件  sys 拥有系统文件  adm 拥有帐户文件  uucp UUCP使用  lp lp或lpd子系统使用  nobody NFS使用 |

1. **拥有帐户文件**

1、还有许多标准的伪用户，例如：audit, cron, mail, usenet等，也都各自为相关的进程和文件所需要。

由于/etc/passwd文件是所有用户可读，如用户密码太简单或规律较明显，一台普通的计算机就能够很容易地将它破解，因此对安全性要求较高的Linux系统都把加密后的口令字分离出来，单独存放在一个文件中，这个文件是/etc/shadow文件。

有超级用户才拥有该文件读权限，这就保证了用户密码的安全性。

2、/etc/shadow中的记录行与/etc/passwd中的一一对应，它由pwconv命令根据/etc/passwd中的数据自动产生

文件格式与/etc/passwd类似，由若干个字段组成，字段之间用":"隔开。字段是：

|  |
| --- |
| 登录名:加密口令:最后一次修改时间:最小时间间隔:最大时间间隔:警告时间:不活动时间:失效时间:标志 |

|  |
| --- |
| "登录名"是与/etc/passwd文件中的登录名相一致的用户账号  "口令"字段存放的是加密后的用户口令字，长度为13个字符。如果为空，则对应用户没有口令，登录时不需要口令；如果含有不属于集合 { ./0-9A-Za-z }中的字符，则对应的用户不能登录。  "最后一次修改时间"表示的是从某个时刻起，到用户最后一次修改口令时的天数。时间起点对不同的系统可能不一样。例如在SCO Linux 中，这个时间起点是1970年1月1日。  "最小时间间隔"指的是两次修改口令之间所需的最小天数。  "最大时间间隔"指的是口令保持有效的最大天数。  "警告时间"字段表示的是从系统开始警告用户到用户密码正式失效之间的天数。  "不活动时间"表示的是用户没有登录活动但账号仍能保持有效的最大天数。  "失效时间"字段给出的是一个绝对的天数，如果使用了这个字段，那么就给出相应账号的生存期。期满后，该账号就不再是一个合法的账号，也就不能再用来登录了。 |

例子：

|  |
| --- |
| ＃ cat /etc/shadow  root:Dnakfw28zf38w:8764:0:168:7:::  daemon:\*::0:0::::  bin:\*::0:0::::  sys:\*::0:0::::  adm:\*::0:0::::  uucp:\*::0:0::::  nuucp:\*::0:0::::  auth:\*::0:0::::  cron:\*::0:0::::  listen:\*::0:0::::  lp:\*::0:0::::  sam:EkdiSECLWPdSa:9740:0:0:::: |

3、用户组的所有信息都存放在/etc/group文件中。

将用户分组是Linux 系统中对用户进行管理及控制访问权限的一种手段。

每个用户都属于某个用户组；一个组中可有多个用户，一个用户也可属于不同的组。

当一个用户同时是多个组中的成员时，在/etc/passwd文件中记录的是用户所属的主组，登录时所属的默认组，而其他组称为附加组。

用户要访问属于附加组的文件，须首先使用newgrp命令使自己成为所要访问的组中成员。

用户组的所有信息都放在/etc/group文件中。

此文件格式也类似于/etc/passwd文件，由冒号(:)隔开若干个字段，字段有：

|  |
| --- |
| 组名:口令:组标识号:组内用户列表  组名":用户组的名称，由字母或数字构成。与/etc/passwd中的登录名一样，组名不应重复。  "口令"字段存放的是用户组加密后的口令字。一般Linux 系统的用户组都没有口令，即这个  字段一般为空，或者是\*。  "组标识号"与用户标识号类似，也是一个整数，被系统内部用来标识组。  "组内用户列表"是属于这个组的所有用户的列表，不同用户之间用逗号(,)分隔。这个用户  组可能是用户的主组，也可能是附加组。 |

/etc/group文件的一个例子如下：

|  |
| --- |
| root::0:root  bin::2:root,bin  sys::3:root,uucp  adm::4:root,adm  daemon::5:root,daemon  lp::7:root,lp  users::20:root,sam |

**四、添加批量用户**

Linux系统提供了创建大量用户的工具，可立即创建大量用户，方法如下：

（1）先编辑一个文本用户文件。

每一列按照/etc/passwd密码文件的格式书写，要注意每个用户的用户名、UID、宿主目录都不可以相同，其中密码栏可以留做空白或输入x号。一个范例文件user.txt内容如下：

user001::600:100:user:/home/user001:/bin/bash

user002::601:100:user:/home/user002:/bin/bash

user003::602:100:user:/home/user003:/bin/bash

user004::603:100:user:/home/user004:/bin/bash

user005::604:100:user:/home/user005:/bin/bash

user006::605:100:user:/home/user006:/bin/bash

（2）以root身份执行命令 /usr/sbin/newusers，从刚创建的用户文件user.txt中导入数据，创建用户：

# newusers < user.txt

然后可执行命令 vipw 或 vi /etc/passwd 检查 /etc/passwd 文件是否已经出现这些用户的数据，且用户的宿主目录是否已创建。

（3）执行命令/usr/sbin/pwunconv。

将 /etc/shadow 产生的 shadow 密码解码，回写到 /etc/passwd 中，并将/etc/shadow的shadow密码栏删掉。是为了方便下一步的密码转换工作，即先取消 shadow password 功能。

# pwunconv

（4）编辑每个用户的密码对照文件。

范例文件 passwd.txt 内容如下：

user001:密码

user002:密码

user003:密码

user004:密码

user005:密码

user006:密码

（5）以root身份执行命令 /usr/sbin/chpasswd。

创建用户密码，chpasswd 会将经过 /usr/bin/passwd 命令编码过的密码写入 /etc/passwd 的密码栏。

|  |
| --- |
| # chpasswd < passwd.txt |

（6）确定密码经编码写入/etc/passwd的密码栏后。

执行命令 /usr/sbin/pwconv 将密码编码为 shadow password，并将结果写入 /etc/shadow。

# pwconv

就完成大量用户的创建了，之后可到/home下检查这些用户宿主目录的权限设置是否都正确，并登录验证用户密码是否正确。

1. **Linux 磁盘管理**

Linux磁盘管理好坏管理直接关系到整个系统的性能问题。

常用命令:

|  |
| --- |
| **df**：列出文件系统的整体磁盘使用量  **du**：检查磁盘空间使用量  **fdisk**：用于磁盘分区 |

* **df(Disk Free)**

功能：检查文件系统的磁盘空间占用情况。可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。

* 语法：

|  |
| --- |
| df [-ahikHTm] [目录或文件名] |

* 选项与参数：

|  |
| --- |
| -a ：列出所有文件系统，包括系统特有的 /proc 等文件系统；  -k ：以 KBytes 的容量显示各文件系统；  -m ：以 MBytes 的容量显示各文件系统；  -h ：以较易阅读的 GBytes, MBytes, KBytes 等格式自行显示；  -H ：以 M=1000K 取代 M=1024K 的进位方式；  -T ：显示文件系统类型, 连同该 partition 的 filesystem 名称 (例如 ext3) 也列出；  -i ：不用硬盘容量，而以 inode 的数量来显示 |

* 实例 1

将系统内所有的文件系统列出来！

|  |
| --- |
| [root@www ~]# df |

如df 没加任何选项，默认会将系统内所有的(不含特殊内存内的文件系统与 swap) 都以 1 Kbytes 的容量来列出来！

* 实例 2

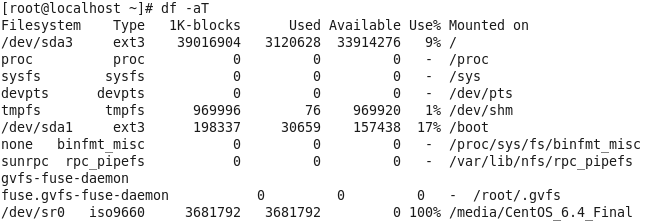
将容量结果以易读的容量格式显示出来

|  |
| --- |
| [root@www ~]# df -h |

* 实例 3

将系统内的所有特殊文件格式及名称都列出来

[root@www ~]# df –aT



* 实例 4

将 /etc 底下的可用磁盘容量以易读的容量格式显示



* **du(Disk Usage)**

查看使用空间，与df不同的是Linux du命令是对文件和目录磁盘使用的空间的查看

* 语法：

|  |
| --- |
| du [-ahskm] 文件或目录名称 |

* 选项与参数：

|  |
| --- |
| -a ：列出所有文件与目录容量，因为默认仅统计目录底下的文件量而已。  -h ：以较易读的容量格式 (G/M) 显示；  -s ：列出总量而已，而不列出每个各别的目录占用容量；  -S ：不包括子目录下的总计，与 -s 有点差别。  -k ：以 KBytes 列出容量显示；  -m ：以 MBytes 列出容量显示； |

* 实例 1

列出目前目录下所有文件容量

|  |
| --- |
| [root@www ~]# du  8 ./test4 <==每个目录都会列出来  8 ./test2  ....中间省略....  12 ./.gconfd <==包括隐藏文件的目录  220 . <==这个目录(.)所占用的总量 |

直接输入 du时，du 会分析当前所在目录的文件与目录所占用的硬盘空间。

* 实例 2

将文件容量也列出来

|  |
| --- |
| [root@localhost ~]# du -a  12 ./install.log.syslog <==有文件的列表了  8 ./.bash\_logout  8 ./test4  8 ./test2  ....中间省略....  12 ./.gconfd  220 . |

* 实例 3

检查根目录下每个目录所占用的容量

|  |
| --- |
| [root@ localhost ~]# du –sm /\*  7 /bin  6 /boot  .....中间省略....  0 /proc  .....中间省略....  1 /tmp  3859 /usr <==系统初期最大就是他了啦！  77 /var |

通配符 \* 来代表每个目录。

与 df 不一样的是，du 其实会直接到文件系统内去搜寻所有的文件数据。

* **fdisk((partition a) fixed disk 磁盘分区)**

磁盘分区表操作工具。

* 语法：

|  |
| --- |
| fdisk [-l] 装置名称 |

* 选项与参数：

|  |
| --- |
| -l ：输出后接装置所有的分区内容。  若仅有 fdisk -l 时， 则系统将会把整个系统内能够搜寻到的装置的分区均列出来。 |

* 实例 1

列出所有分区信息

|  |
| --- |
|  |

* 实例 2

找出系统中的根目录所在磁盘，并查阅该硬盘内的相关信息

|  |
| --- |
| [root@localhost ~]# df / <==注意：重点在找出磁盘文件名而已  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/sda3 39016904 3120625 33914252 9% / |

|  |
| --- |
| [root@www ~]# fdisk /dev/ sda <==仔细看，不要加上数字喔！  The number of cylinders for this disk is set to 5005.  There is nothing wrong with that, but this is larger than 1024,  and could in certain setups cause problems with:  1) software that runs at boot time (e.g., old versions of LILO)  2) booting and partitioning software from other OSs  (e.g., DOS FDISK, OS/2 FDISK) |

|  |
| --- |
| Command (m for help): <==等待你的输入！ |

输入 m 后，就会看到底下这些命令介绍

|  |
| --- |
| Command (m for help): m <== 输入 m 后，就会看到底下这些命令介绍  Command action  a toggle a bootable flag  b edit bsd disklabel  c toggle the dos compatibility flag  d delete a partition <==删除一个partition  l list known partition types  m print this menu  n add a new partition <==新增一个partition  o create a new empty DOS partition table  p print the partition table <==在屏幕上显示分割表  q quit without saving changes <==不储存离开fdisk程序  s create a new empty Sun disklabel  t change a partition's system id  u change display/entry units  v verify the partition table  w write table to disk and exit <==将刚刚的动作写入分割表  x extra functionality (experts only) |

离开 fdisk 时按下q，所有动作都不会生效！

相反的, 按下w动作生效。

|  |
| --- |
| Command (m for help): p <== 这里可输出目前磁盘的状态  Disk /dev/hdc: 41.1 GB, 41174138880 bytes <==这个磁盘的文件名与容量  255 heads, 63 sectors/track, 5005 cylinders <==磁头、扇区与磁柱大小  Units = cylinders of 16065 \* 512 = 8225280 bytes <==每个磁柱的大小  Device Boot Start End Blocks Id System  /dev/hdc1 \* 1 13 104391 83 Linux  /dev/hdc2 14 1288 10241437+ 83 Linux  /dev/hdc3 1289 1925 5116702+ 83 Linux  /dev/hdc4 1926 5005 24740100 5 Extended  /dev/hdc5 1926 2052 1020096 82 Linux swap / Solaris  # 装置文件名 启动区否 开始磁柱 结束磁柱 1K大小容量 磁盘分区槽内的系统 |

用 p 可列出目前这个磁盘的分割表信息，信息的上半部显示整体磁盘的状态。

* **磁盘格式化(mkfs- make filesystem)**
* 语法：

|  |
| --- |
| mkfs [-t 文件系统格式] 装置文件名 |

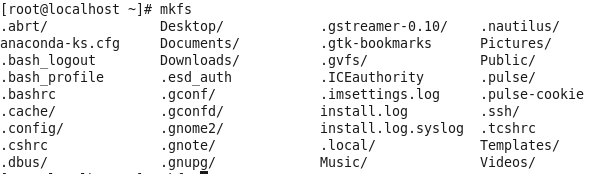
* 选项与参数：

-t ：可接文件系统格式，例如 ext3, ext2, vfat 等(系统有支持才会生效)

* 实例 1:查看 mkfs 支持的文件格式

|  |
| --- |
| [root@www ~]# mkfs [tab] [tab]  mkfs mkfs.cramfs mkfs.ext2 mkfs.ext3 mkfs.msdos mkfs.vfat |

按下两个[tab]，会发现 mkfs 支持的文件格式如上所示。



* 实例 2

将分区 /dev/hdc6（可指定你自己的分区） 格式化为 ext3 文件系统：

|  |
| --- |
| [root@www ~]# mkfs -t ext3 /dev/hdc6  mke2fs 1.39 (29-May-2006)  Filesystem label= <==这里指的是分割槽的名称(label)  OS type: Linux  Block size=4096 (log=2) <==block 的大小配置为 4K  Fragment size=4096 (log=2)  251392 inodes, 502023 blocks <==由此配置决定的inode/block数量  25101 blocks (5.00%) reserved for the super user  First data block=0  Maximum filesystem blocks=515899392  16 block groups  32768 blocks per group, 32768 fragments per group  15712 inodes per group  Superblock backups stored on blocks:  32768, 98304, 163840, 229376, 294912  Writing inode tables: done  Creating journal (8192 blocks): done <==有日志记录  Writing superblocks and filesystem accounting information: done  This filesystem will be automatically checked every 34 mounts or  180 days, whichever comes first. Use tune2fs -c or -i to override.  # 这样就创建起来所需的 Ext3 文件系统了！ |

* **磁盘检验(fsck-file system check)**

检查和维护不一致的文件系统。

若系统掉电或磁盘发生问题，可利用fsck对文件系统进行检查。

* 语法：

|  |
| --- |
| fsck [-t 文件系统] [-ACay] 装置名称 |

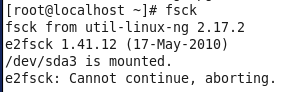
* 选项与参数：

|  |
| --- |
| -t : 给定档案系统型式，若在 /etc/fstab 中已定义或 kernel 本身已支援的则不需加上此参数  -s : 依序一个一个地执行 fsck 的指令来检查  -A : 对/etc/fstab 中所有列出来的 分区（partition）做检查  -C : 显示完整的检查进度  -d : 打印出 e2fsck 的 debug 结果  -p : 同时有 -A 条件时，同时有多个 fsck 的检查一起执行  -R : 同时有 -A 条件时，省略 / 不检查  -V : 详细显示模式  -a : 如果检查有错则自动修复  -r : 如果检查有错则由使用者回答是否修复  -y : 选项指定检测每个文件是自动输入yes，在不确定那些是不正常的时候，可以执行 # fsck -y 全部检查修复。 |

* 实例 1

查看系统有多少文件系统支持的 fsck 命令：

|  |
| --- |
| [root@www ~]# fsck [tab][tab]  fsck fsck.cramfs fsck.ext2 fsck.ext3 fsck.msdos fsck.vfat |



* 实例 2

强制检测 /dev/hdc6 分区:

|  |
| --- |
| [root@www ~]# fsck -C -f -t ext3 /dev/hdc6  fsck 1.39 (29-May-2006)  e2fsck 1.39 (29-May-2006)  Pass 1: Checking inodes, blocks, and sizes  Pass 2: Checking directory structure  Pass 3: Checking directory connectivity  Pass 4: Checking reference counts  Pass 5: Checking group summary information  vbird\_logical: 11/251968 files (9.1% non-contiguous), 36926/1004046 blocks |

如没有加上 -f 的选项，则由于这个文件系统不曾出现问题，检查的经过非常快速！

若加上 -f 强制检查，才会一项一项的显示过程。

* **磁盘挂载(mount)与卸除(umount)**
* 磁盘挂载语法：

|  |
| --- |
| mount [-t 文件系统] [-L Label名] [-o 额外选项] [-n] 装置文件名 挂载点 |

* 实例 1

用默认的方式，将刚刚创建的 /dev/hdc6 挂载到 /mnt/hdc6 上面！

|  |
| --- |
| [root@www ~]# mkdir /mnt/hdc6  [root@www ~]# mount /dev/hdc6 /mnt/hdc6  [root@www ~]# df  Filesystem 1K-blocks Used Available Use% Mounted on  .....中间省略.....  /dev/hdc6 1976312 42072 1833836 3% /mnt/hdc6 |

* 磁盘卸载：

|  |
| --- |
| umount [-fn] 装置文件名或挂载点 |

* 选项与参数：

|  |
| --- |
| -f ：强制卸除！可用在类似网络文件系统 (NFS) 无法读取情况下；  -n ：不升级 /etc/mtab 情况下卸除。 |

|  |
| --- |
| 卸载/dev/hdc6  [root@www ~]# umount /dev/hdc6 |

1. **Linux vi/vim**

所有 Unix Like 系统都会内建 vi 文书编辑器。

但目前使用较多的是 vim 编辑器。

vim 有程序编辑能力，可主动的以字体颜色辨别语法的正确性，方便程序设计。

* **什么是 vim？**

Vim是从 vi 发展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。

简单来说， vi 是老式的字处理器，不过功能已很齐全，但还是有可进步的地方。

vim 则可以说是程序开发者的一项很好用的工具。

vim 的官方网站 (http://www.vim.org) 也说 vim 是一个程序开发工具而不是文字处理软件。

vim 键盘图：



* **vi/vim 的使用**

vi/vim 共分为三种模式:

|  |
| --- |
| 命令模式（Command mode）  插入模式（Insert mode）  底线命令模式（Last line mode）。 |

作用分别是：

* **命令模式：**

用户刚启动 vi/vim，便进入了命令模式。

此状态下敲击键盘动作会被Vim识别为命令，而非输入字符。

比如此时按下i，并不会输入一个字符，i被当作了一个命令。

常用命令：

|  |
| --- |
| i 切换到输入模式，以输入字符。  x 删除当前光标所在处的字符。  : 切换到底线命令模式，以在最底一行输入命令。 |

若想要编辑文本：启动Vim，进入了命令模式，按下i，切换到输入模式。

命令模式只有一些最基本命令，仍要依靠底线命令模式输入更多命令。

* **输入模式**

在命令模式下按下i就进入输入模式。

输入模式可使用按键：

|  |
| --- |
| 字符按键以及Shift组合，输入字符  ENTER，回车键，换行  BACK SPACE，退格键，删除光标前一个字符  DEL，删除键，删除光标后一个字符  方向键，在文本中移动光标  HOME/END，移动光标到行首/行尾  Page Up/Page Down，上/下翻页  Insert，切换光标为输入/替换模式，光标将变成竖线/下划线  ESC，退出输入模式，切换到命令模式 |

* **底线命令模式**

在命令模式下按下:（英文冒号）就进入了底线命令模式。

底线命令模式可输入单个或多个字符的命令，可用的命令非常多。

在底线命令模式中，基本的命令有（已经省略了冒号）：

|  |
| --- |
| q 退出程序  w 保存文件 |

按ESC键可随时退出底线命令模式。

简单的说，可将这三个模式想成底下的图标来表示：



* **vi/vim 使用实例**

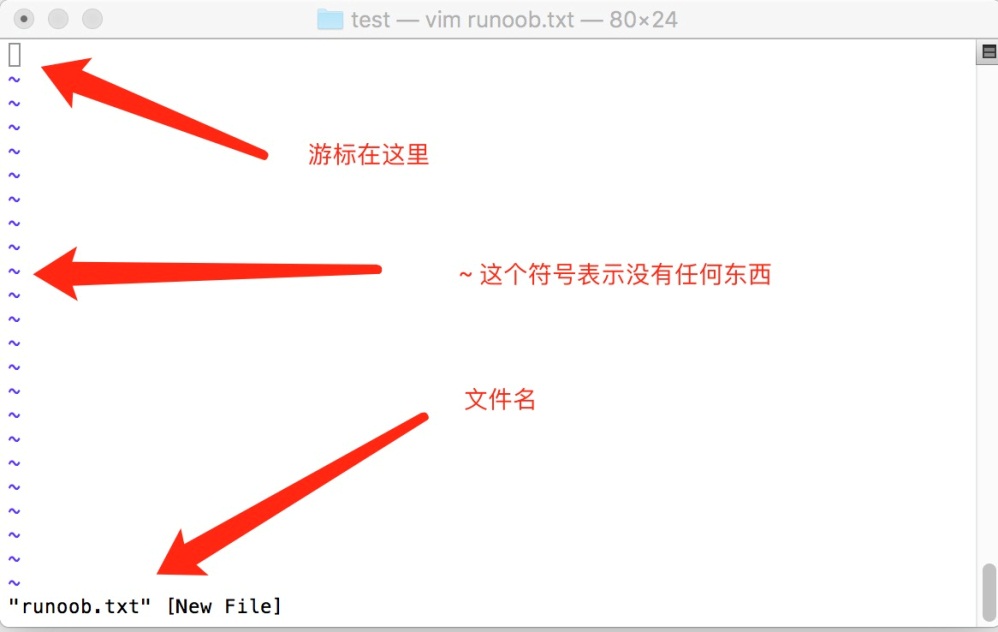
使用 vi/vim 进入一般模式

如果你想要使用 vi 来建立一个名为 test.txt 的文件时，可这样做：

|  |
| --- |
| $ vi runoob.tx |

直接输入 vi 文件名 就能进入 vi 的一般模式了。

请注意，记得 vi 后面一定要加文件名，不管该文件存在与否！



按下 i 进入输入模式(也称为编辑模式)，开始编辑文字

一般模式中，只要按下 i, o, a 等字符就可进入输入模式了！

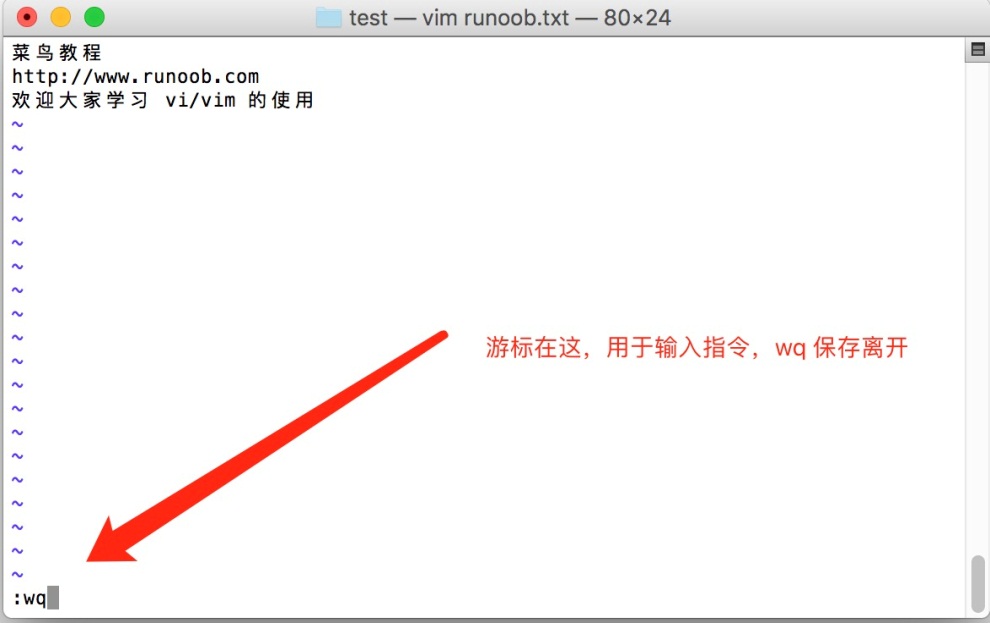
在编辑模式当中，可发现在左下角状态栏中会出现 –INSERT- 字样，就是可输入任意字符的提示。

这时，键盘上除了 Esc 这个按键之外，其他的按键都可以视作为一般的输入按钮了，所以可进行任何的编辑。



按下 ESC 按钮回到一般模式

在一般模式中按下 :wq 存盘并离开 vi！



就成功创建了一个 runoob.txt 的文件。

* **vi/vim 按键说明**

除上面简易范例的 i, Esc, :wq 之外，其实 vim 还有非常多的按键可使用。

**第一部份：一般模式可用的光标移动、复制粘贴、搜索替换等**







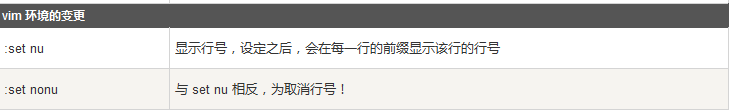


**第二部份：一般模式切换到编辑模式的可用的按钮说明**



**第三部份：一般模式切换到指令行模式的可用的按钮说明**





特别注意，在 vi/vim 中，数字很有意义！通常代表重复做几次的意思！

也可能是代表去到第几个什么什么的意思。

举例来说，要删除 50 行，则是用 『50dd』 对吧！ 数字加在动作之前，如我要向下移动 20 行呢？那就是『20j』或者是『20↓』即可。

1. **linux yum 命令（没仔细看）**

yum（ Yellow dog Updater, Modified）是一个在Fedora和RedHat以及SUSE中的Shell前端软件包管理器。

基於RPM包管理，能从指定的服务器自动下载RPM包且安装，可自动处理依赖性关系，且一次安装所有依赖的软体包，无须繁琐地一次次下载、安装。

yum提供了查找、安装、删除某一个、一组甚至全部软件包的命令，且命令简洁而又好记。

* **yum 语法**

|  |
| --- |
| yum [options] [command] [package ...] |

|  |
| --- |
| options：可选，包括-h（帮助），-y（当安装过程提示选择全部为"yes"），-q（不显示安装的过程）等等。  command：要进行的操作。  package操作的对象。 |

* **yum常用命令**

|  |
| --- |
| 1.列出所有可更新的软件清单命令：yum check-update  2.更新所有软件命令：yum update  3.仅安装指定软件命令：yum install <package\_name>  4.仅更新指定软件命令：yum update <package\_name>  5.列出所有可安裝的软件清单命令：yum list  6.删除软件包命令：yum remove <package\_name>  7.查找软件包 命令：yum search <keyword>  8.清除缓存命令:  yum clean packages: 清除缓存目录下的软件包  yum clean headers: 清除缓存目录下的 headers  yum clean oldheaders: 清除缓存目录下旧的 headers  yum clean, yum clean all (= yum clean packages; yum clean oldheaders) :清除缓存目录下的软件包及旧的headers |

* 实例 1

|  |
| --- |
| [root@www ~]# yum install pam-devel //安装 pam-devel  Setting up Install Process  Parsing package install arguments  Resolving Dependencies <==先检查软件的属性相依问题  --> Running transaction check  ---> Package pam-devel.i386 0:0.99.6.2-4.el5 set to be updated  --> Processing Dependency: pam = 0.99.6.2-4.el5 for package: pam-devel  --> Running transaction check  ---> Package pam.i386 0:0.99.6.2-4.el5 set to be updated  filelists.xml.gz 100% |=========================| 1.6 MB 00:05  filelists.xml.gz 100% |=========================| 138 kB 00:00  -> Finished Dependency Resolution  ……(省略) |

* 实例 2

|  |
| --- |
| [root@www ~]# yum remove pam-devel //移除 pam-deve l  Setting up Remove Process  Resolving Dependencies <==同样的，先解决属性相依的问题  --> Running transaction check  ---> Package pam-devel.i386 0:0.99.6.2-4.el5 set to be erased  --> Finished Dependency Resolution  Dependencies Resolved  =============================================================================  Package Arch Version Repository Size  =============================================================================  Removing:  pam-devel i386 0.99.6.2-4.el5 installed 495 k  Transaction Summary  =============================================================================  Install 0 Package(s)  Update 0 Package(s)  Remove 1 Package(s) <==还好，并没有属性相依的问题，单纯移除一个软件  Is this ok [y/N]: y  Downloading Packages:  Running rpm\_check\_debug  Running Transaction Test  Finished Transaction Test  Transaction Test Succeeded  Running Transaction  Erasing : pam-devel ######################### [1/1]  Removed: pam-devel.i386 0:0.99.6.2-4.el5  Complete! |

* 实例 3

|  |
| --- |
| [root@www ~]# yum list pam\* //找出以 pam 为开头的软件名称  Installed Packages  pam.i386 0.99.6.2-3.27.el5 installed  pam\_ccreds.i386 3-5 installed  pam\_krb5.i386 2.2.14-1 installed  pam\_passwdqc.i386 1.0.2-1.2.2 installed  pam\_pkcs11.i386 0.5.3-23 installed  pam\_smb.i386 1.1.7-7.2.1 installed  Available Packages <==底下则是『可升级』的或『未安装』的  pam.i386 0.99.6.2-4.el5 base  pam-devel.i386 0.99.6.2-4.el5 base  pam\_krb5.i386 2.2.14-10 base |

* **国内 yum 源**

网易（163）yum源是国内最好的yum源之一 ，速度还是软件版本，都非常不错。

将yum源设置为163 yum，可提升软件包安装和更新速度，同时避免一些常见软件版本无法找到。

**安装步骤**

首先备份/etc/yum.repos.d/CentOS-Base.repo

mv /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo.backup

下载对应版本repo文件, 放入/etc/yum.repos.d/(操作前请做好相应备份)

CentOS5 ：http://mirrors.163.com/.help/CentOS5-Base-163.repo

CentOS6 ：http://mirrors.163.com/.help/CentOS6-Base-163.repo

运行以下命令生成缓存

yum clean all

yum makecache

除网易之外，国内还有其他不错的yum源，比如中科大和搜狐。

中科大的yum源，安装方法查看：https://lug.ustc.edu.cn/wiki/mirrors/help/centos

sohu的yum源安装方法查看: http://mirrors.sohu.com/help/centos.html

**配置本地Yum仓库**

实现此案例需要按照如下步骤进行。

步骤一：搭建一个本地Yum，将RHEL6光盘手动挂载到/media

命令操作如下所示：

|  |
| --- |
| [root@localhost ~]# mount /dev/cdrom /media/  mount: block device /dev/sr0 is write-protected, mounting read-only  [root@localhost ~]# mount | tail -1  /dev/sr0 on /media type iso9660 (ro) |

步骤二：将本地设置为客户端，进行Yum验证

Yum客户端需编辑配置文件，命令操作如下所示：

|  |
| --- |
| [root@localhost ~]# cd /etc/yum.repos.d/ //必须在这个路径下  [root@localhost yum.repos.d]# ls //此路径下事先有配置文件的模板  rhel-source.repo  [root@localhost yum.repos.d]# cp rhel-source.repo rhel6.repo //配置文件必须以.repo结尾  [root@localhost yum.repos.d]# vim rhel6.repo  [rhel-6] //中括号里内容要求唯一，但不要出现特殊字符  name=Red Hat Enterprise Linux 6 //此为描述信息，可以看情况填写  baseurl=file:///media/ //此项为yum软件仓库位置，指向光盘挂载点  enabled=1 //此项为是否开启，1为开启0为不开启  gpgcheck=1 //此项为是否检查签名，1为监测0为不检测  gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release //签名认证信息的路径  [root@localhost /]# yum repolist  Loaded plugins: product-id, refresh-packagekit, security, subscription-manager  This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.  rhel-6 | 3.9 kB 00:00 ...  rhel-6/primary\_db | 3.1 MB 00:00 ...  repo id repo name status  rhel-6 Red Hat Enterprise Linux 6 3,690 |