Linux 技术栈

一、 计算机系统

1.1 五大单元

输入单元，内存，CPU，输出单元，外部存储设备

1.2 存储单位

1Byte（字节）=8bit

八进制：
1KB=1024Byte
1MB=1024KB
1GB=1024MB
1TB=1024GB
1PB=1024TB

十进制：
1KB=1000Byte
1MB=1000KB
1GB=1000MB
1TB=1000GB
1PB=1000TB

网络上常用的单位是 Mbit/s，1MB/s=8Mbit/s
硬盘上的存储单位是采用十进制 Byte，所以 500G=500,000,000,000Byte（十进制）≈460GB（八进制）

1.3 CPU

主板最重要的是芯片组，芯片组有两个桥接器，北桥负责连接速度较快的 CPU，内存和显卡，南桥负责连接速度比较慢的周边接口，如硬盘，USB，网卡等；

北桥 VS 南桥
AMD 的 CPU 为了加快内存和 CPU 的通讯，直接让内存控制组件集成到 CPU

中，而不通过北桥让内存和 CPU 通讯，理论上加速了 CPU 和内存的传输速度；北桥的总线称为系统总线，因为是内存传输的主要通道，南桥就是输入输出（I/O）总线，主要用于连接硬盘、USB、网卡等设备。
北桥所支持的频率称为前端总线速度（Front Side Bus，FSB）。而每次传送的位数则为总线宽度，而总线频宽= FSB*总线宽度。
与总线宽度相似的，CPU 每次能处理的数据量称为字组大小（word size），字组大小依据 CPU 的涉及有 32 位和 64 位之分，也就是现在我们常说的电脑 32 位和 64 位的由来。

外频 VS 倍频
外频指的是 CPU 与外部组件进行数据传输/运算时的速度，倍频是 CPU 内部用来加速工作的一个倍数，两者相乘才是 CPU 的频率。

1.4 内存 Memory

DRAM，Dynamic Random Access Memory，动态随机访问内存，由于断电即消失，又称这种 RAM 为挥发性内存；

SRAM，Static Random Access Memory，动态随机访问内存，晶体管比较多，价格比较贵，不易做成大容量，不过由于速度比较快，因此集成到 CPU 内称为高速缓存以加快数据的访问。

ROM 只读存储器，非挥发性内存，Read Only Memory，BIOS（Basic Input Output System）就是写在 ROM 里面；

1.5 操作系统

内核：Kernel
其实是一组程序，重点在于管理计算机的所有活动以及驱动系统中的所有硬件。即硬件的所有操作都必须通过操作系统来完成，负责整个计算机系统相关的资源分配和管理，内核的功能至少也要有：

- 系统调用的接口
- 程序管理
- 内存管理
- 文件系统管理
- 设备驱动

系统调用：System Call
操作系统提供一整组的开发接口给工程师来开发软件。

在定义上讲，只要能够让计算机硬件正确无误的运行，那就是操作系统了，所以

也就说操作系统就是内核与其提供的接口工具。

驱动程序
操作系统通常会提供给一个开发接口给硬件开发商，让他们可以根据这个接口设计可以驱动他们硬件的驱动程序。

二、 Linux 是什么

Linux 的内核原型是 1991 年由托瓦兹（Linux Torvalds）写出来的，

1 Linux 是一套操作系统

早期是针对 386 写的，Torvalds 先生再写 Linux 的时候，其实该内核仅能“驱动 386 的所有硬件”而已，即所谓的“让 386 计算机开始运行，并且等待用户指令输入”。
Linux 是 Open Source 的，具备“可移植性”，提供了一个完整操作系统当中最底层的硬件控制与资源管理的完整架构。
Linux 之前 UNIX 的历史，UNIX 一开始是由 Ken Thompson 用汇编写出来的，后来用高级语言的 B 语言写，发现性能不好，后来他们把 B 语言改进成 C 语言，再用 C 语言去写，最后放出 UNIX 的发行版（1973 年）。

三、 Linux 如何学习

1. Linux 有两种界面

分别是图形界面 X Window 和命令行界面 Common Line，X Window System 也只是运行在 Linux 上的一套软件。
基础知识：

1) 计算机概论与硬件相关知识；
2) 先从 Linux 的安装和命令学起；
3) Linux 操作系统的基础技能；
4) 务必学会 vi 文本编辑器；
5) Shell 和 Shell 脚本的学习；
6) 一定要会软件管理员；
7) 网络基础的建立；

2. CentOS7.2 设置 GRUB2 引导界面分辨率
   https://www.cnblogs.com/nidey/p/6374860.html

最近在学习 OS 引导启动，GRUB2 的学习材料也不少，主要还看官方手册清晰些。

公司里办公机的多启动用的 ubuntu 的界面，还挺炫酷的。之前看其他博客网文里看到可以设置 grub2 的分辨率，我拿 CentOS7.2 试了下，发现不行。
网上都是说设置 GRUB_GFXMODE=1440x900，再 update-grub 更新下 grub.cfg，但是没生效（我又拿 ubuntu14.04 server 试了，可以的）
后来还是看 GRUB 官方手册解决了。

进入系统后，更改分辨率，设置如下

[root@min-base ~]# vim /etc/default/grub

将 GRUB_TERMINAL_OUTPUT 值由默认的"console"改为"gfxterm"，并添加 GRUB_GFXMODE
复制代码：

```
GRUB_TIMEOUT=2
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="gfxterm"
GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet"
GRUB_DISABLE_RECOVERY="true"
GRUB_GFXMODE=1440x900,1024x768,640x480
```

3. 修改命令行颜色

https://www.cnblogs.com/Q--T/p/5394993.html

PS1 应用之——修改 linux 终端命令行各字体颜色

最近在学习 linux 操作系统（CentOS 6 & CentOS 7）。觉得 linux 终端命令行全部为白色,会经常导致命令与输出内容难以分辨。于是上网找到修改 linux 终端命令行颜色的方法，发现通过定义 PS1 环境变量即可实现,下面我以 root 用户身份进行操作。

1.了解 PS1

PS1 是 Linux 终端用户的一个环境变量，用来定义命令行提示符的参数

在终端输入命令：

```
# echo $PS1
```

可得到当前 PS1 的定义值：

```
PS1='[\u@\h \W]\$ '
```

PS1 的常用参数以及含义:

\d ：代表日期，格式为 weekday month date，例如："Mon Aug 1"

\H ：完整的主机名称

\h ：仅取主机名中的第一个名字

\t ：显示时间为 24 小时格式，如：HH：MM：SS

\T ： 显示时间为 12 小时格式
\A ： 显示时间为 24 小时格式：HH：MM
\u ： 当前用户的账号名称
\v ： BASH 的版本信息
\w ： 完整的工作目录名称
\W ： 利用 basename 取得工作目录名称，只显示最后一个目录名
\# ： 下达的第几个命令
\$ ： 提示字符，如果是 root 用户，提示符为 # ，普通用户则为 $
所以 linux 默认的命令行提示信息的格式

PS1='[\u@\h \W]\$ ' 的意思就是：[当前用户的账号名称@主机名的第一个名字 工作目录的最后一层目录名]#

2.颜色设置参数

在 PS1 中设置字符颜色的格式为：\[\e[F;Bm\].....\[\e[0m\]，其中“F “为字体颜色，编号为 30-37，“B”为背景颜色，编号为 40-47,\[\e[0m\]作为颜色设定的结束。

颜色对照表：

F	B	
30	40	黑色
31	41	红色
32	42	绿色
33	43	黄色
34	44	蓝色
35	45	紫红色
36	46	青蓝色
37	47	白色

只需将对应数字套入设置格式中即可。

比如要设置命令行的格式为绿字黑底(\[\e[32;40m\])，显示当前用户的账号名称(\u)、主机的第一个名字(\h)、完整的当前工作目录名称(\w)、24 小时格式时间(\t)，可以直接在命令行键入如下命令：

```
# PS1='[\[\e[32;40m\]\u@\h \w \t]$ \[\e[0m\]'
```

经过多次测试后,最终确定了一个适合我自己的格式:

```
# PS1="\[\e[37;40m\][\[\e[32;40m\]\u\[\e[37;40m\]@\h \[\e[36;40m\]\w\[\e[0m\]]\\$ "
```

但注意这样的设置只是临时性的.

3.修改.bashrc 文件,永久保存命令行样式.

上面的设置的作用域只有当前终端的登陆有效，关闭终端或退出登录即刻失效。要想永久性的保存设置，需要修改.bashrc 配置文件。

键入命令：

```
1 # cd
```

2 # ls -la

现在可以看到.bashrc 这个文件。编辑.bashrc:

vim .bashrc

加入这一行:

PS1="\[\e[37;40m\][\[\e[32;40m\]\u\[\e[37;40m\]@\h \[\e[36;40m\]\w\[\e[0m\]]\\$ "

保存退出。

重新加载 bash 配置文件:

source .bashrc

即可立即并永久生效了。

四、 主机规划与磁盘分区

1 各种硬件设备在 Linux 中的文件名

在 Linux 这个系统中，几乎所有额硬件设备文件都在/dev 这个目录内：

设备	设备在 Linux 内的文件名
IDE 硬盘	/dev/hd[a-d]
SCSI/SATA/USB	/dev/sd[a-p]
软驱	/dev/fd[a-p]
打印机	25 针：/dev/lp[0-2] USB：/dev/usb/lp[0-15]
鼠标	USB：/dev/usb/mouse[0-15] PS2：/dev/psaux
当前的 CD ROM/DVD ROM	/dev/cdrom
当前鼠标	/dev/mouse
磁带机	IDE：/dev/ht0 SCSI：/dev/st0

2 磁盘分区

1) IDE 硬盘

通常主机会提供两个 IDE 接口，分别叫 IDE1(Primary)和 IDE2(Secondary)，每个 IDE 接口可以连接一条 IDE 扁平电缆，每条 IDE 扁平电缆可以连接两个 IDE Master（主设备）和 Slave（从设备）

IDE/Jumping	Master	Slave
IDE1(Primary)	/dev/hda	/dev/hdb
IDE2(Secondary)	/dev/hdc	/dev/hdd

2) SCSI/SATA/USB
磁盘没有一定的顺序，要看 Linux 内核具体检测到的磁盘顺序；

3 磁盘的组成

磁盘由盘片，机械手臂，磁头和主轴马达组成，数据在盘片上，盘片又可以细分为扇区（Sector）和柱面（Cylinder），其中扇区每个为 512Bytes。
其实这个扇盘的第一扇区特别重要，因为它主要记录了两个重要的信息：

1) 主引导分区（Master Boot Record, MBR）可以安装引导加载程序的地方，有 446Bytes
2) 分区表（partition table）记录整块硬盘分区的状态，有 64Bytes。分区的最小单位是煮柱面，总共分为 4 组记录区 P1~P4，每组记录区记录了该区段的起始和结束的柱面号码。分区分为主分区 primary 和拓展分区 extended
   - P1：/dev/hda1
   - P2：/dev/hda2
   - P3：/dev/hda3
   - P4：/dev/hda4

3) 简单的定义
   - 主分区和拓展分区最多可以有 4 个；
   - 拓展分区最多有一个；
   - 逻辑分区是由拓展分区持续切割出来的分区；
   - 能够被格式化后作为数据访问的分区为主分区和逻辑分区，拓展分区无法格式化；
   - 逻辑分区的数量依操作系统的不同而有所区别，IDE 硬盘最多有 59 个逻辑分区（5 号~63 号），SATA 硬盘最多有 11 个（5 号~15 号）
   - 主分区和逻辑分区不能够被整合，若强制整合则会破坏拓展分区，拓展分区遭到破坏，其他逻辑分区也会收到牵连。

   拓展分区的目的是使用额外的扇区来记录分区信息，其本身并不能拿来格式化，但可以继续切为逻辑分区（Logical partition）而逻辑分区的命名则从 /dev/hda5 开始。给新硬盘上建立分区时都要遵循以下的顺序：建立主分区→建立扩展分区→建立逻辑分区→激活主分区→格式化所有分区。

4 开机流程与主引导分区（MBR）

定义：韧体，写入到硬件上的一个软件程序，BIOS 是一个韧体。
流程：开机 → BIOS → MBR → 引导加载程序→ 内核文件
位于 MBR 里面的引导加载程序叫 Boot roader：

- 提供选择不同系统的选项；
- 载入内核文件；

- 转交其他 loader

位于分区里面启动扇区（boot sector）的引导加载程序：

- 每个分区都拥有自己的启动扇区（boot sector）；
- 实际可开机的内核文件放置到各个分区里面；
- loader 只认自己分区的内核文件，以及其他 loader；
- loader 可将管理权交给另外的管理程序；

此特色造就了“多重引导”的功能。

五、 首次登陆和在线求助 man page

1 首次登陆系统

1) 窗口模式和命令行模式的切换

Ctrl+Alt+F1 窗口模式

Ctrl+Alt+F2~F6 命令行模式

命令行模式登陆输入密码的时候“不会显示任何字样。”

退出账号的时候使用命令：exit

模式登陆后所求的程序被称为 shell，由于这个程序是内核最外层跟用户进行通讯工作的，所以戏称为“shell”；

2) 命令格式

[lvhongbin@loccalhost ~]$ command [-option] parameter1 parameter2 …

账号名　　命令　选项　参数 1　　参数二

一行命令中第一个输入的绝对是“命令”或者是“可执行文件”；

中括号并不实际存在，选项前添加一个短横线-；

命令太长的时候，可以使用反斜杠“\”来转义[Enter]符号，使命令连续到下一行

3) Linux 是区分大小写的；

如 ls 这个命令列出“自己主文件夹（~）”，显示日期的 date，显示日历 cal，显示好用的计算器：bc，离开计算器回到命令提示符时用 quit。

4) 重要的热键

【Tab】【Tab】 具有“命令补全”和“文件补齐”的功能；

【Ctrl】+ c　　强制当前程序“停止”；

【Ctrl】+ d　　关闭文字界面；

man 命令和 info 命令

进入 man 命令后，可以按下空格键进行往下翻页，按下“q”离开 man 环境；

进入 info 命令后，可以按下 N、P、U 分别进行下一个，上一个和上一个节点的选择，光标选中开头具有“*”命令的一行并按下【ENTER】便可进入该节点，？号显示命令一览表

2 超简单的文本编辑器：nano

打开或者新建一个新的文件：nano text.txt
其中，指数符号^ 表示【Ctrl】M 表示【Alt】

3 正确的开关机

Linux 是一个多用户工作的系统，不能随便关掉电源来关机。在关机时需要注意：

- 查看系统的使用状态；
  如果看网络的联机状态，可以执行“netstat -a”的命令，如果要看后台执行的程序，可以执行“ps -aux”的命令;
- 通知在线用户关机的时刻；
  对于 Linux 系统来讲，重启和关机：reboot，halt 和 poweroff 是非常重要的操作，一般需要 root 用户才能执行

1) 数据同步写入磁盘：sync
   一般来讲，数据都要经过内存才能被 CPU 执行，而且数据又经常需要通过内存读入或者写进硬盘。但是硬盘的传输和读写效率很低，为了提高效率，某部分数据进入内存后先不写入硬盘，继续保留在内存中，方便后续的修改。如果这时候突然死机后者关掉电源，由于内存是挥发性存储器，数据会丢失。所以在关机之前需要做好数据从内存写入硬盘的操作。用命令 sync 完成。
   事实上一般账户也能使用 sync 命令，只是只能保留自己的数据罢了。
2) 常用的关机命令：shutdown
   shutdown 执行它的工作是送信号〔signal〕给 init 程序，要求它改变 runlevel。Runlevel 0 被用来停机〔halt〕，runlevel 6 是用来重新激活〔reboot〕系统，而 runlevel 1 则是被用来让系统进入管理工作可以进行的状态；这是预设的，假定没有-h 也没有-r 参数给 shutdown。要想了解在停机〔halt〕或者重新开机〔reboot〕过程中做了哪些动作，你可以在这个文件/etc/inittab 里看到这些 runlevels 相关的资料。
   参数说明:
   [-t] 在改变到其它 runlevel 之前，告诉 init 多久以后关机。
   [-r] 重启计算器。
   [-k] 并不真正关机，只是送警告信号给每位登录者〔login〕。
   [-h] 关机后关闭电源〔halt〕。
   [-n] 不用 init，而是自己来关机。不鼓励使用这个选项，而且该选项所产生的后果往往不总是你所预期得到的。
   [-c] cancel current process 取消目前正在执行的关机程序。所以这个选项当然没有时间参数，但是可以输入一个用来解释的讯息，而这信息将会送到每位使用者。
   [-f] 在重启计算器〔reboot〕时忽略 fsck。

[-F] 在重启计算器〔reboot〕时强迫 fsck。

[-time] 设定关机〔shutdown〕前的时间。

例如：

／sbin／shutdown －h now－－－－－立即关机；

／sbin／shutdown －h 20：49－－－－20：49 分关机

／sbin／shutdown －h ＋10－－－－－10 分钟后关机

／sbin／shutdown －r now －－－－－立即重启

／sbin／shutdown －r ＋10 ‘The system will reboot'－－－－10 分钟后系统重启并给每个登录用户发通知

／sbin／shutdown －k now ‘The system will reboot'－－－－仅给每个登录用户发通知并不真关机

3) halt 最简单的关机命令

其实 halt 就是调用 shutdown -h。halt 执行时，杀死应用进程，执行 sync 系统调用，文件系统写操作完成后就会停止内核。

参数说明:

[-n] 防止 sync 系统调用，它用在用 fsck 修补根分区之后，以阻止内核用老版本的超级块〔superblock〕覆盖修补过的超级块。

[-w] 并不是真正的重启或关机，只是写 wtmp〔/var/log/wtmp〕纪录。

[-d] 不写 wtmp 纪录〔已包含在选项[-n]中〕。

[-f] 没有调用 shutdown 而强制关机或重启。

[-i] 关机〔或重启〕前，关掉所有的网络接口。

[-p] 该选项为缺省选项。就是关机时调用 poweroff。

4) reboot

reboot 的工作过程差不多跟 halt 一样，不过它是引发主机重启，而 halt 是关机。它的参数与 halt 相差不多。

5) 切换执行等级：init

init 是所有进程的祖先，它的进程号始终为 1，所以发送 TERM 信号给 init 会终止所有的用户进程、守护进程等。shutdown 就是使用这种机制。init 定义了 8 个运行级别(runlevel)，init 0 为关机，init1 为重启。关于 init 可以长篇大论，这里就不再叙述。另外还有 telinit 命令可以改变 init 的运行级别，比如，telinit -iS 可使系统进入单用户模式，并且得不到使用 shutdown 时的信息和等待时间。

6) Poweroff

关机，在多用户方式下(Runlevel 3)不建议使用。Poweroff 基本和 halt 一致，不过使用 poweroff 会关机后并且切断电源。poweroff 在关闭计算机操作系统之后，最后还会发送 ACPI 指令，通知电源，最后切断电源供应，当然路由器等嵌入系统不支持 ACPI 的，所以这个无效。

六、 文件权限和目录配置

1 文件权限的概念

“ls”命令表示 list，“-al”表示 all，文件名前面的“.”表示包含隐藏文件，下面是一般格式：

-rw-rw-r--. 1 lvhongbin lvhongbin 17 Dec 31 17:20 hello.txt
文件权限 连接数 所有者 用户组 文件大小 最后修改的日期 文件名

1) 第一列代表这个文件的类型和权限（permission）；
   第一个文件权限共 10 个字符，共分 4 组，第一个字符自己一组，后面每三个字符一组，分别记为 group1~group4。r 的权利最容易获得，其次是 x，最难是 w。
   - group1:目录，文件或者链接文件
     - [d]表示目录
     - [-]表示文件
     - []表示连接文件 linkfile
     - [b]表示设备文件里面的可供存储的接口设备
     - [c]表示设备文件里面的串行端口设备，如鼠标和键盘
   - group2:文件所有者的权限
     - [r]表示可读
     - [w]表示可写
     - [x]表示可执行
     - [-]表示啥权利都没有>《<
   - group3:同用户组的权限
     - [r]表示可读
     - [w]表示可写
     - [x]表示可执行
     - [-]表示啥权利都没有>《<
   - group4:其他非本用户组的权限
     - [r]表示可读
     - [w]表示可写
     - [x]表示可执行
     - [-]表示啥权利都没有>《<

2) 第二列表示有多少文件名连接到此节点；
3) 第三列表示这个文件（或目录）的“所有者账号”；
4) 第四列表示所属的用户组；
   你的账号会依附于一个或者多个用户组。
5) 第五列为文件大小；
   文件大小的单位默认为 Byte。

6) 第六列为最近修改的日期；
7) 第七列为文件名；

2 改变文件的属性和权限

1) chgrp 改变文件所属用户组
   格式：chgrp [-r] 用户组 文件名/文件目录
   [-r]表示递归，即连同子目录下的所有文件和目录；
2) chown 改变文件所有者
   格式：chown [-r] 所有者 文件名/文件目录
   chown 也能改变用户组名称，在所有者后面加上“.”和所要修改的用户组名称，当然了，也能单纯的修改用户组名称，所有者换成“.”和所要修改的用户组名称；
3) chmod 改变文件的权限
   法一：数字类型改变
   格式：chmod [-r] xyz 文件名/文件目录
   rwx 分别代表 4、2、1，然后 xyz 分别代表 owner、group 和 others 三种身份的权限累加；
   法二：符号类型改变

chmod	u g o a	+(加入) -(除去) =(设置)	r w x	文件或者目录

   例子：chmod u=rwx,go=rx hello.txt
   注意，以上 u=rwx,go=rx 的逗号旁边是没有空格隔开的；
4) cp 复制
   格式：cp 源文件 目标文件

3 目录与文件的权限意义

1) 权限对文件的重要性
   三种权限均主要针对文件的内容而言的，与文件的存在与否没有关系
   - r：可读取此文件的实际内容；
   - w：可以编辑，新增或者修改该文件的内容；
   - x：该文件具有被系统执行的权限；
2) 权限对目录的重要性；
   - r：可读取此目录结构列表的权限，既可以查询该文件目录下的文件名数据；
   - w：可以更改该目录结构列表的权限，即新建新的文件与目录，删除已经存在的文件或者目录，将已存在的文件或者目录进行重命名，转移改目录内的文件、目录位置；

- x：该文件具有被系统执行的权限，即代表用户能否进入该目录称为工作目录的用途，所谓工作目录（work directory）就是你目前所在的目录，拥有 x 权限可以查阅文件的内容，所以在开放目录给别人浏览的时候，应该至少给予 r 及 x 的权限，但是 w 的权限不能随便给；

3) 重要的命令：
   - cd：切换目录
   - mkdir：新建目录
   - touch：新建空的文件
   - su：切换用户
   - cd ..：返回上一级目录
   - cd /：返回根目录
   - rm：删除文件
   - rm -r ：删除文件目录

4 Linux 的目录配置

Filesystem hierarchy standard, FHS 目录配置标准

	可分享的 shareable	不可分享的 unshareable
不变的 static	/usr 软件放置处	/etc 配置文件
	/opt 第三方软件	/boot 开机与内核文件
可变动的 variable	/var/mail 用户邮件信箱	/var/run 程序相关
	/var/spool/news 新闻组	/var/lock 程序相关

可分享的 shareable：可以分享给网络上其他主机挂载用的目录；
不可分享的 unshareable：不可以分享给网络上其他主机挂载用的目录；
不变的：这些数据不经常变动，例如一些函数库和文件说明文件等；
root：与开机系统有关；
usr：UNIX software resource：与软件的安装/执行有关；
var：variable，与系统运作有关；
prop：跟内存数据相关的文件夹，由于数据都在内存中，所以不占据硬盘空间；

目录	内容
/bin	系统执行文件
/boot	开机会使用到的文件
/dev	任何设备和接口设备的文件
/etc	系统的主要配置文件，如人员的账号密码
/home	系统默认用户的主文件夹
/lib	系统的函数库
/media	可删除的设备，包括 DVD，软盘等
/mnt	暂时挂载某些设备
/opt	第三方软件放置的目录，如 KDE 这个桌面管理系统
/root	系统管理员的主文件夹
/sbin	包括开机，修复，还原系统所需要的重要系统执行命令

/srv	跟网络服务有关
/tmp	程序运行时数据暂时放置的地方

绝对路径 VS 相对路径

- 绝对路径：由根目录（/）开始写起的文件名或者目录名称；
- 相对目录：不是由根目录（/）开始写起的文件名或者目录名称；
- uname：查看系统信息命令

七、 Linux 文件与目录管理

1 目录的相关操作

- .：代表当前目录，也可以用./来表示；
- ..：代表上一层目录，也可以用../来表示
- -：代表前一个工作目录
- ~：代表当前用户身份所在的主文件夹
- ~account：代表 account 这个用户的主文件夹，account 是一个账号名称，即 /home/account
- pwd：print working directory 显示当前目录；
- mkdir：可以创建目录，但是一次只能生成一层，不过可以使用-p 的参数进行多层创建；
- rmdir：删除空的目录，也可以使用-p 的参数
- mv：文件移动，即剪切然后到别的目录粘贴 mv [-option] 原文件路径+名字 新文件路径+名字
- cp：复制 cp [-option] 原文件路径+名字 新文件路径+名字
- -i：该参数表示在执行操作的时候会询问用户要不要进行覆盖等信息
- -a：全部
- echo：显示出，打印的意思
- basename：取得文件的名字
- dirname：取得文件的目录

2 关于执行文件路径的变量：$PATH

命令 ls 的完整路径是/bin/ls(绝对路径)
那为什么只用输入 ls 就可以了呢？在于$PATH；
可以尝试用 echo $PATH 看看都有哪些变量，每个变量都用冒号：隔开
/usr/local/bin:
/usr/local/sbin:
/usr/bin:
/usr/sbin:

/bin:
/sbin:
/home/lvhongbin/.local/bin:
/home/lvhongbin/bin
在 PATH 中加入新的目录可执行如下命令：
PATH=”$PATH”:/root

3 文件内容查阅

- cat：concatenate 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 的倒写形式;
- nl：显示的时候顺便输出行号；
- more：一页一页地显示文件内容；
- head：只看前面几行；
- tail：只看后面几行；
- od：以二进制的形式读取文件内容；

4 修改文件时间或创建新文件：touch

- modification time (mtime)：这个是文件更新的时间；
- status time(ctime)：状态或者权限被改变的时候的时间；
- access time(atime)：文件被使用的时候的时间，如被打开或者复制；

那怎么查看呢？
ls -l --time=atime 文件全路径
若想查看所有的时间：
ll test.txt; ll --time=ctime test.txt; ll --time=atime test.txt
“ll”两个小写的 l 表示“ls -l”的命令，命令与命令之间用分号；隔开。默认查询修改时间 mtime，此时间不需要添加--time 命令，否则会出错；
修改时间命令：
touch -d "2 days ago" test.txt
使用该命令修改的时间只会修改 mtime 和 atime，而 ctime 会更改为目前的时间；

5 文件的默认权限与隐藏属性与特殊权限

1) 文件默认权限
   文件的默认权限为-rw- rw- rw-，简称 666；
   目录的默认权限是 drwxrwxrwx，简称 777；
   使用 umask 时后面跟随 3 位数字，用默认的减去 umask 后面跟随的数字便可得新得到的默认权限；
   如：umask 002

2) 文件的隐藏属性：
   - chattr 修改文件隐藏属性
     chattr [+-=][ASacdistu] 文件或者目录名称
     如：chattr +i 文件全路径
     表示添加文件不被删除，改名，设置连接，也无法添加或者写入数据的属性
     chattr -i 文件全路径
     表示删除该属性
     另外还有其他属性
   - lsattr 显示文件的隐藏属性

3) 文件特殊权限
   - SUID(SetUserID)
     符号：s，数字：4
     功能：可以暂时获得 root 的权限，执行者暂时具备程序所有者 owner 的权限；
     前提：二进制程序（binary program），执行者对于改程序具有 x 的可执行权限；
     例子：chmod u=rwxs,go=x text.txt 或者 chmod 4755 test.txt
   - SGID(SetGroupID)
     符号：s，数字：2
     功能：执行者暂时具备程序用户组 Group 的权限；
     前提：二进制程序（binary program），执行者对于改程序具有 x 的可执行权限；
     例子：chmod, g+s text.txt
   - SBIT
     符号：t，数字：1
     功能：当用户拥有 SBIT 权限时。仅有自己和 root 才有权限对自己创建的文件和目录进行删除，重命名和移动等操作，而无法删除别人的文件；
     前提：目录，执行者对于改程序具有 x 的可执行权限；；
     例子：chmod, o+t text.txt

6 命令和文件的查询

1) 脚本文件名的查询 which
   Which [-a] command
   在$PATH 中查询命令的路径，加上-a 表示，查找出所有的路径；
2) 文件名的查找
   - whereis 查找特定文件

   whereis [-bmus] 文件或者目录名
   - locate 查找含有特定前缀的文件或者目录，和 whereis 一样特点是查找已创建的数据库，对于不同发行版的 Linux，每次更新数据库的时间都不相

同，比如 CentOS 5.x 是每天一更新。更新后才能找最新的文件，手动更新的命令为“updatedb”

- find 直接在硬盘中查找，比较花时间

  find [PATH] [option] [action]

  跟时间有关的参数：mtime,ctime,atime

  - -mtime n：n 为数字，意义是 n 天之前的“一天之内”被更改的文件，即 n<mtime<n+1；
  - -mtime +n：列出 n 天之前（不包含 n 天本身）被更改的文件，即≥n+1 天的文件；
  - -mtime -n：列出 n 天之内（含 n 天本身）被更改的文件，即≤n；
  - -newer file：file 为一个存在的文件，列出比 file 还要更新的文件名；
    如 find /etc -newer /etc/passwd
    即在/etc 目录下寻找比/etc/passwd 还要新的文件；

  与用户名有关的参数

  - -user name：name 为用户名
  - -group name：name 为用户组名
  - -nouser：寻找的用户名不存在
  - -nogroup：寻找的用户组不存在
    如： find /home -user lvhongbin 把某用户的所有文件都找出来

  与文件权限及名称相关的参数

  - -name filename：查找文件名为 filename 的文件
  - -size [+-]SIZE：查找比 SIZE 大（+）或者小（-）的文件，单位 c 代表 Byte，k 代表 KB
  - -perm mode：mode 是数字属性，属性刚好等于 mode 的文件，如 4755.
  - -perm -mode：mode 是数字属性，属性要包含 mode 的文件
  - -perm +mode：mode 是数字属性，属性有任一 mode 的文件

  当然了，find 还有许多其他的功能；

八、 Linux 磁盘与文件系统的管理

1. 认识 EXT2 文件系统

1) Linux 最传统的磁盘文件系统：EXT2（Linux second extended file system，Ext2fs）

扇区 sector 为最小的物理存储单位，每个扇区为 512Bytes；
将扇区围成一个圆，就成为一个柱面，柱面是最小的分区（partition）单位
文件系统分为三个部分；

- Super block：记录此文件系统的整体信息，每个；
- Inode：记录文件的权限和属性，一个文件占用一个 inode，同时记录此文

件的数据所在的 block 号码；

- Block：实际记录文件的内容，若文件太大时，会占用多个 block。

这种通过 Inode 查找 Block 的访问方法称为“索引式文件系统”（indexed allocation）
另外还有另外一种方式，如 U 盘，他没有 Inode，只有 Block，每一个 Block 除了记录信息和数据以外，还记录了下一个 Block 的编号；

2. EXT2 文件系统

由启动扇区和多个 blockgroup 组成，每个 blockgroup 由 superblock，文件系统描述，块对应表，inode 对应表，inode table，data block。
EXT2 文件系统中 block 大小只有 1KB，2KB 和 4KB 三种而已，而且格式化的时候 block 的大小就已经固定了，每个 block 都有编号，以便 indode 的记录。

3. 注意：

- Super block
  每个 super block 都可以用 dumpe2fs 这个命令来查询；
- Block
  Block 的大小和数量在格式化完了就不能在改变了（除非重新格式化）；
  每个 block 只能放置一个文件的数据；
  如果一个文件大于一个 block 的大小，则会占用多个 block；
  若文件小于 block，则该 block 的剩余空间就不能再被利用了（磁盘空间就会被浪费）
- Inodetable
  分为 12 个直接记录+1 个间接记录+1 个双间接+1 个三间接，间接事宜 block 拓展作为号码记录；
- Filesystem Description
  这个区段描述的是每个 blockgroup 的开始和结束的 block 号码；
- Block bitmap
  用来记录哪些 block 是空的；
- Inode bitmap
  用来记录哪些 inode 是空的；

4. 相关命令

- dumpe2fs：查询 super block 的信息；
  dumpe2fs [-hb]，如 dumpe2fs /dev/hdc2
- df：调出目前挂载的设备；

- ls -i：列出文件的 inode 值

5. 目录树与其他文件系统

1) 文件的读取流程：
   16777283 drwxr-xr-x. 2 lvhongbin lvhongbin 1024 Jan 1 17:42 text.text
   其实目录就是一个挂载点的信息，通过目录一层一层地揭开 inode 的信息，最终找到文件数据的 block 位置。从而找到文件。

2) 文件的写入流程：
   - 先确定用户对于添加文件的目录是否具有 w 和 x 的权限，若有才添加；
   - 跟据 Inode bitmap 找到没有使用的 inode 号码，并将新文件的权限/属性写入；
   - 跟据 Block bitmap 找到没有使用的 block 号码，并将实际的数据写入 block，且更新 inode 的 block 的指向数据；
   - 将刚写入的 inode 与 block 数据同步更新 Inode bitmap 和 Block bitmap，并更新 superblock 的内容；

3) 数据存放区域：inode table，data block；
   中间数据 metadata：superblock，block bitmap，inode bitmap，数据经常变动，每次添加，删除，编辑的时候都可能会影响这三部分的数据；

4) 数据不一致的状态（Inconsistent）
   若在写入数据的时候发生断电，最后一步的同步中间数据的步骤没有完成，就会出现数据不一致的情况，早些时候的系统在重新启动后都会通过 superblock 当中记录 valid bit(是否有挂载)与文件系统 state（clean 与否）等状态来判断是否进行强制数据一致性检查。若需要检查时则以 e2fack 这支程序来进行，不过检查这的很费时间，后来就衍生了日志文件系统；
   日志文件系统

5) 日志文件系统
   为了克服上面的困难，我们的前辈想出了一个好的办法，在文件系统中规划出一个块，专门用来记录写入和修订文件的一些操作步骤，依次简化一致性检查的步骤。
   - 预备：当系统要写入一个文件的时候，先在日志记录中记录某个文件要写入的信息；
   - 实际写入：检查用户的权限，检查 inode bitmap 和 block bitmap，开始写入文件的权限和属性和数据，然后更新 inode bitmap 和 block bitmap 和 superblock 的内容。
   - 结束：完成数据和 meta data 的更新，再在记录块中完成该文件的记录。

6) Linux 文件系统的操作

异步处理 asynchronous

当一个文件被加载到内存后，如果他没有被改动过，他就被打上 clean 的标签，如果他被改动了，就会被打上 dirty 的标签。系统会不定时使用 sync 的命令。把 dirty 的文件数据写入硬盘，以保证内存和硬盘数据的一致性。若正常关机的时候，关机命令会主动调用 sync 来将内存的数据回写入磁盘内。

7) 挂载点（mount point）的意义与其他文件系统

每个 block group 都有自己的独立的 inode，block 和 superblock，这个文件树要连接到目录树中才能被我们使用。将文件系统和目录树相结合的操作我们称为挂载。重点是，挂载点一定是目录，该目录为进入该文件的入口。由于每个 inode 只容纳一个文件，所以可以判断相同的 inode 号码的文件是同一个文件。

命令：

- # ls -l /lib/modules/$(uname -r)/kernel/fs
  查看本 linux 系统支持的文件系统有哪些
- cat /proc/filesystems
  查看加载到内存中的文件系统有哪些

Linux VFS

VFS Virtual Filesystem Switch, 虚拟文件系统，他是 linux kernel 内核的一个部分，负责调用不同的模块去读写硬件中的文件系统

6. 文件系统的简单操作

1) 列出文件系统的整体磁盘使用量：df [-ahikHTm] [目录或者文件名]

-a 显示所有

-k 以 KB 为单位

-m 以 MB 为单位

-h 以 KB，MB，GB 等单位自行显示；

-i 不用硬盘容量，而以 inode 的数量来显示；

例子：

[root@localhost /]# df -a

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
rootfs	-	-	-	-	/
sysfs	0	0	0	-	/sys
proc	0	0	0	-	/proc
devtmpfs	1916980	0	1916980	0%	/dev

1K-blocks 下面数字的单位是 1KB；

Mounted on 表示磁盘挂载的目录所在（挂载点）df 读取的是 superblock 里面的信息，所以显示结果的速度非常快速。

2) 评估文件系统的磁盘使用量：du [-ahskm] 文件或者目录名称
会把目录里面所有的子目录和文件都列出来

3) 连接文件 ln
Ln [-sf] 源文件 目标文件
如果不加任何参数，那就是硬连接，如果加了-s 那就是符号连接，而-f 则是如果目标文件中已存在相同名字的文件，则会将之删除再重新建立
- 硬连接 hard link

通过文件系统的 inode 链接产生新的文件名，而不是产生新的文件，称为“硬连接”（hard link）。而硬连接则是在某个目录 block 下新建一条【文件名连接到某 inode 号码】的关联记录而已，既不会增加 inode 也不会耗用 block 的数量。后面不管你用哪个文件进行修改，最终都会写入相同的 inode 和 block 中。而且连接数会加 1。但是他也是有限制的，如不能跨文件系统，也不能连接到目录（不能连接到目录的意思是一旦建立硬链接，目录底下的所有的文件都一次性建立了连接，如果后面你需要增加文件时，需要重新建立硬连接，所以不是那种单纯的仅仅是目录的连接）
- 符号连接，也叫快捷方式 symbolic link

其实就是新建一个文件，该文件可以让数据的读取指向他连接的那个文件的文件名，当原文件删除后，快捷方式就打不开了，使用 ls -l 命令的时候看到文件名后面带→的就表示是符号连接。但是连接数不会加 1。

7. 磁盘的分区，格式化，检验和挂载

1) 新买的硬盘如何使用：
- 建立分区
- 格式化 format，已建立系统可用的文件系统；
- 对新建好的文件系统进行检验；
- 在 Linux 系统中需要建立挂载点（也就是目录），并将它挂载起来；

2) 磁盘分区
- fdisk 磁盘分区操作
- fdisk -l 查看系统所有磁盘信息

[root@localhost Desktop]# fdisk -l

Disk /dev/sda: 85.9 GB, 85899345920 bytes, 167772160 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000cee2e

Device Boot		Start	End	Blocks	Id	System
/dev/sda1	*	2048	2099199	1048576	83	Linux
/dev/sda2		2099200	167772159	82836480	8e	Linux LVM

Disk /dev/mapper/centos-root: 53.7 GB, 53687091200 bytes, 104857600 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/mapper/centos-swap: 4160 MB, 4160749568 bytes, 8126464 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/mapper/centos-home: 27.0 GB, 26969374720 bytes, 52674560 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

3) 磁盘格式化 mkfs(make file system)
mkfs [-t 文件系统格式] 系统有支持才会生效
文件系统参数有 ext3, ext2, vfat 等
事实上，当我们去使用命令 mkfs -t ext3 时，调用的是 mkfs.ext3 的命令
maks[Tab] [Tab]可以查看 mkfs 支持的文件系统格式
[root@localhost Desktop]# mkfs
mkfs mkfs.btrfs mkfs.cramfs mkfs.ext2 mkfs.ext3
mkfs.ext4 mkfs.fat mkfs.minix mkfs.msdos mkfs.vfat
mkfs.xfs

4) 磁盘检验：fsck(file system check)，badblocks（坏道检查）
fsck 命令不要随便用，通常只有你身为 root 且你的文件系统有问题，而且已经卸载（不在挂载状态）的时候采用，因为该命令可能会对你的系统造成损伤
5) 磁盘挂载与卸载
挂载前需要注意：
单一文件系统和单一挂载点（目录）应该是一一对应的关系；
作为挂载点的目录理论上应该是空目录才对，否则，原来的文件会被暂时隐藏，当该文件系统被卸载时才会被显现出来；
mount 和 umount

6) 内存交换空间 swap 的构建

功能：应付物理内存不足的时候所造成的内存扩展记录

由于 swap 是用硬盘来暂时放置内存信息的，所以用到 swap 的时候，主机硬盘的灯会响个不停；

过程：

- 分区
- 格式化 mkswap [设备文件名]
- 使用 swapon [设备文件名]
- 查看 通过 free 命令查看内存的使用情况

  Swap 对于桌面计算机而言意义不太大，因为一般内存都超过 512M，但是对于常年在线的服务器而言，还是有设置一下的必要。

九、 文件与文件系统的压缩与打包

公共类是代码重用的一种形式，它将各个公用功能模块经常调用的方法提取到公用的 Java 类中，例如数据库的 Dao 类容纳了所有访问数据库的方法，并同时管理这数据库的连接和关闭，这样不但实现了代码的重用，还提高了程序的性能和代码的可读性。

1 常见的压缩命令

*.Z compress 程序压缩的文件
*.gz gzip 程序压缩的文件
*.bz2 bzip2 程序压缩的文件
*.tar tar 程序打包的程序，并没有压缩过；
*.tar.gz tar 程序打包的数据，其中经过 gzip 的压缩
*. tar.bz2 tar 程序打包的数据，其中经过 bzip2 的压缩

1) compress

   非常老旧的一款压缩程序，CentOS 默认没有安装，需要安装 ncompress 软件

   compress [-rcv] 文件或者目录

   uncompress 文件.Z

2) gzip

   gzip [-cdtv#] 文件名 压缩 -#是数字，-1 最快，-9 最慢，压缩比默认是-6 最好，-v 会显示详细属性

   gzip -d 文件名 解压

   zcat 文件名 查看压缩文件的数据

3) bzip2

   bzip2 [-cdkzv#] 文件名 压缩 -#是数字，-1 最快，-9 最慢，压缩比默认是-6

最好，-v 会显示详细属性
bzip2 -d 文件名 解压
bzcat 文件名 查看压缩文件的数据

4) tar
压缩：tar -jcv -f filename.txt.tar.bz2 要被压缩的文件或者目录名称
查询：tar -jtv -f filename.txt.tar.bz2
解压缩：tar -jxv -f filename.txt.tar.bz2 -C 与解压的目录
-j：表示利用 bzip2 程序的支持；
-z：表示利用 gzip 程序的支持；
-c：表示压缩；
-t：表示查看；
-x：表示解压缩；

2 完整的备份工具 dump 和恢复工具 restore

能够备份实时的文件系统，设立 9 等级，0 等级表示完整的实时备份，level 1 表示实时文件系统与 level 0 比较后差异部分的备份，level 2~9 以此类推。
如果找不到 dump 命令，需要安装，命令为 yum -y install dump
Dump [-Suvj] [-level] [-f 备份文件] 待备份数据
-S：仅列出后面带备份数据需要多少磁盘空间才能够备份文件
-u；将此次 dump 的时间记录到/etc/dumpdates 文件中
-v：将备份的过程显示出来
-j：加入 bzip2 的压缩程序支持
1) 当备份的是单一文件系统时；
可以使用完整的 dump 功能，包括利用 0-9 的数个 level 来备份
2) 当备份的只是目录，而非单一文件系统时；
所有的备份必须放在该目录下，而且只能使用 level0，仅支持完整备份，不支持-u 参数（无法创建/etc/dumpdates 这个 level 备份的时间记录文件）
注意，在备份之前先用 df -h 的命令看看那个属于文件系统

十、 vim 程序编辑工具

1 vi 与 vim

每一个 linux distribution 都会有一套文本编辑器 vi，vim 是 vi 的高级版，vim 有程序编辑的功能，可以主动以字体颜色辨别语法的正确性

vi 的使用
1) 一般模式

“vi 文件名”，在此模式下，你可以左右移动光标，也可以删除字符或者删除正行，也可以复制粘贴文字数据。

2) 编辑模式
除了拥有一般模式的功能，还可以进行编辑，不过你要先按下“i,l,o,O,a,A,r,R”等其中任意一个字符。

3) 命令行模式
在一般模式当中，输入“：”或者“？”或者“/”三个字符的其中一个的时候，便可进入该模式，可以进行数据的查找，读取，保存，离开 vi，保存，大量替换字符等操作。

注意

- 一般模式和编辑模式和命令行模式可以相互切换，按一下 Esc 便可还原为一般模式，但是编辑模式和命令行模式不可相互切换；一般模式下输入“：wq”便可保存离开。
- 当我们在编辑的时候，vim 会在被编辑文件的目录下再建一个.filename.swp 的暂存文件，你做的所有操作都会被保存在改文件中。
- Alias 命令：可以发现 alias vi='vim'，说明我们使用的 vi 命令，其实指的就是 vim，

```
[lvhongbin@MiWiFi-R3G-srv test]$ alias
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias perlll='eval `perl -Mlocal::lib`'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

2 一些常用的命令

1) [Ctrl]+f：向下移动一页；
2) [Ctrl]+b：向上移动一页；
3) [Ctrl]+d：向下移动半页；
4) [Ctrl]+u：向上移动半页；
5) n+<space>：在同一行中向右移动 20 个字符距离；
6) 0 或功能键[Home]：光标移动到行首；
7) $或功能键[End]：光标移动到行末；
8) G：移动到文件的最后一行；
9) nG：n 是数字，表示光标移动到文件的第 n 行；
10) gg：表示光标移动到文件的第 1 行；

11) N+[Enter]：n 是数字，光标向下移动 N 行；

12) /字符串：向下寻找字符串；
13) ？字符串：向上寻找字符串；
14) n：重复上一次寻找，方向向下；
15) N：重复上一次寻找，方向向上；

16) :n1,n2s/string1/string2/g：从第 n1 行到第 n2 行寻找 string1，并将其替换成 string2；
17) :1,$s/string1/string2/g:从第 1 行到最后 1 行寻找 string1，并将其替换成 string2；
18) :1,$s/string1/string2/gc：从第 n1 行到第 n2 行寻找 string1，并在将其替换成 string2 时向用户确认；

19) x：相当于向后删除一个字符；
20) X：相当于向前删除一个字符；
21) Nx：相当于向后删除 N 行；
22) dd：删除整行；
23) Ndd：向下删除 N 行；
24) d1G：删除光标所在到第 1 行的数据；
25) dG：删除光标所在到最后 1 行的数据；
26) d0：删除同一行中光标所在到第一个字符的数据；
27) d$：删除同一行中光标所在到最后一个字符的数据；
28) Nyy：向下复制 N 行；
29) y1G：复制光标所在到第 1 行的数据；
30) yG：复制光标所在到最后 1 行的数据；
31) y0：复制同一行中光标所在到第一个字符的数据；
32) y$：复制同一行中光标所在到最后一个字符的数据；
33) p：向后粘贴；
34) P：向前粘贴；

35) u：撤销上一步操作；
36) [ctrl]+r：还原上一步操作；
37) :set nu：显示行号；
38) :set nonu：取消行号；
39) . ：重复前一个操作；

3 Vim 的文件保存，恢复与打开时的警告信息

1) [Ctrl]+z：将 vim 丢到后台，此时在同一目录下会产生一个同名+.swp 的暂存文件；
2) Kill -9 %1：模拟断线停止 vim 的工作，算是不正常关掉 vim 吧；

3) 用 vim 再次打开之前的文件的时候，会出现提示：

```
E325: ATTENTION
Found a swap file by the name "../tem/.test1.txt.swp"
          owned by: lvhongbin   dated: Wed Jan  3 13:45:40 2018
        file name: ~lvhongbin/Desktop/tem/test1.txt
         modified: YES
        user name: lvhongbin   host name: MiWiFi-R3G-srv
      process ID: 76718
While opening file "../tem/test1.txt"
              dated: Wed Jan  3 11:54:59 2018

(1) Another program may be editing the same file.  If this is the case,
    be careful not to end up with two different instances of the same
    file when making changes.  Quit, or continue with caution.
(2) An edit session for this file crashed.
    If this is the case, use ":recover" or "vim -r ../tem/test1.txt"
    to recover the changes (see ":help recovery").
    If you did this already, delete the swap file "../tem/.test1.txt.swp"
    to avoid this message.

Swap file "../tem/.test1.txt.swp" already exists!
[O]pen Read-Only, (E)dit anyway, (R)ecover, (D)elete it, (Q)uit, (A)bort:
```

4) 当你恢复后需要使用(D)elete it 把同名+.swp 暂存文件删掉，要不然每次打开都会有警告提示：

4 其他操作与语系转换

1) 块选择（Visual Block）

在一般模式下按下[Ctrl]+ v，然后移动光标，就可以进行矩形选择，然后按下 y 或者 d 可以进行复制或者删除。

- [Ctrl]+ v：块选择；
- V：行选择；
- v：字符选择

2) 多文件操作

用 vi 命令同时打开多个文件，文件与文件之间用空格打开即可，显示的时单窗口界面，打开的是第一个文件；

选择文件可以使用以下命令

- :n ：选择下一个文件；
- :N ：选择上一个文件；
- :files ：列出所有文件

3) 多窗口功能

主要针对同一个文件或者已有文件跟空白文件之间的编辑

命令：

:sp [filename]：若不加文件名，就是分割同一个文件，你对比同一个文件不同位置的文字内容；若加了名字，则是打开一个新文件

: [Ctrl]+ w+ ↑：选择上一个窗口；

: [Ctrl]+ w+ ↓：选择下一个窗口；

4) 语系转换

命令：

- iconv --list：列出所有的编码语法；
- iconv -f 原本编码 -t 新编码 filename [-o newfile]：
  -o newfile：如果要保留原来的文件，可以使用新文件名另存为新的文件；

十一、 bash

1 硬件，内核，shell 和 bash 的关系

管理整个计算机硬件的是操作系统的内核 kernel，kernel 需要被保护起来，用户只能通过 shell 与 kernel 进行通讯。只要能操作应用程序的接口都叫做 shell，狭义的 shell 指的是命令行方面的软件，广义的 shell 包括图形界面的软件。其中 bash 就是 shell 中的一种。

1) Shell 的特点：
   - 几乎所有的 distribution 发行版的 shell 命令相同；
   - 命令行界面比较快；
   - Linux 下默认的 shell 是 bash（Bourne Again Shell, Bourne Shell 的增强版本）基于 GUN 架构开发出来的。
2) bash 的特点：
   - 命令记忆功能，按住↑或者↓可以查询最近使用的命令；
   - 命令与文件补全功能，[Tab]键的好处；
   - 命令别名设置功能，如 alias lm = 'ls -al'
   - 作业控制，前台和后台控制（job control, foreground, background）
   - 程序脚本（shell script）
   - 通配符（Wildcard）
3) Bash 的内部命令：type
   Type [-tpa] name：查询命令是来自内部命令 builtin 还是 alias 抑或是外部命令 file，类似于 which 的功能；
4) 如果一行太长写不完，可以用\[Enter]将[Enter]转义使之不具备执行命令的功能；

2 shell 的变量功能

为了区别自定义变量，环境变量通常以大写字母来表示；
变量的显示和设置：

- echo $变量名：显示变量的内容；
- 变量名=变量的内容：设置变量；
- unset 变量名：取消变量的设置

设置规则：

- 等号两边的变量名和变量的内容不允许出现空格；
- 变量名必须时英文字母或者数字，且首位不能是数字；
- 变量内容中用双引号包含特殊字符，该特殊字符可以保持他原来的特性，如“$name”中 name 变成变量，显示的是变量 name 的内容，或者双引号里面有空格也可以；
- 单引号内的特殊字符则仅为一般字符（纯文本）；
- 若该变量为了增加变量的内容，则可以使用“$变量名”累加内容；
- export 使变量成为环境变量；

1) 返单引号（`）的作用
   程序先执行单引号里面的内容，再把执行的结果返回给单引号外面的命令，其实这个功能可以使用 后执行的命令 $(先执行的命令)来替代。
2) 环境变量的功能
   env：environment 查看环境变量；
   export：功能跟 env 类似；
   set：查看多有变量；
   RANDOM：随机数环境变量，显示 0~32767 的随机变量；
3) $的功能
   表示这个 Shell 的线程代号，即所谓的 PID，Process ID 控制
   ```
   [lvhongbin@MiWiFi-R3G-srv tem]$ $$
   bash: 25672: command not found...
   ```
   出现的数字就是线程号
4) ？的功能
   上一个命令的回传码，一般来讲，如果上一条命令运行成功，则会回传 0，否则回传一个非零的错误码；
5) OSTYPE，HOSTYPE，MACHTYPE
   ```
   [lvhongbin@MiWiFi-R3G-srv tem]$ echo $OSTYPE
   linux-gnu
   [lvhongbin@MiWiFi-R3G-srv tem]$ echo $HOSTTYPE
   x86_64
   [lvhongbin@MiWiFi-R3G-srv tem]$ echo $MACHTYPE
   x86_64-redhat-linux-gnu
   ```
6) locale 语系变量
7) 变量的有效范围
   环境变量=全局变量

自定义变量=局部变量;

8) 变量键盘的读取，数组和声明
   read [-pt] variable
   -p：双引号里面添加提示的内容；
   -t：后面接等待的秒数
9) declare/typeset
   用于声明变量的类型
   -a：声明为数组类型 array
   -i：声明为 integer
   -x：声明为环境变量
   -r：声明为 readonly 类型，相当于 final 的功能
   默认是字符串的类型
10) 与文件系统的及程序的限制关系：ulimit;
    包括限制用户的打开文件的数量，可以使用 CPU 的时间，可以使用的内存总量等
11) 变量内容的删除
    ${变量名#要删除的内容}，可以使用通配符，
    #表示从最左边往右开始删除，删除最短的那个，##表示从最左边往右开始删除，删除最长的那个；
    %表示从最右边往左开始删除，删除最短的那个，%%表示从最左边往右开始删除，删除最长的那个；
    如：　截取路径名中的文件名 ${path 名字##/*/}
    　　截取路径名中的目录 ${path 名字%/*}
12) 变量内容的替换
    ${变量名/旧内容/新内容}，可以使用通配符，
    第一个/表示从最左边往右开始替换，只替换第一个，//表示从最左边往右开始替换，替换全部；
13) 变量内容的存在性检测-
    newVar=${oldVar-content}　检测 oldVar 有值的话，让 newVar 等于$ oldVar，若没有值则让 newVar=content；
    newVar=${oldVar+content}　检测 oldVar 有值的话，让 newVar 等于 content，若没有值则让 newVar=$ oldVar；
    “”表示空字符串，有值；
14) 命名别名和去除别名
    alias 和 unalias
15) 历史命令：history

3 Bash shell 的操作环境

命令运行的顺序：

1) 以相对或绝对路径执行命令，如“/bin/ls”；
2) 由 alias 找到该命令；

3) 由 bash 内置的 bulitin 命令来执行；
4) 通过$PATH 这个变量来执行；

bash 的登陆和欢迎信息
1) 切换终端机接口时的欢迎界面，修改/etc/issue 文件；
2) 登陆时显示欢迎界面，修改/etc/motd 文件

4 Bash shell 的环境配置文件（太复杂了，不看）

logic shell VS non-logic shell
1) logic shell，取得 bash 时需要完整的登陆流程，包括输入用户名和密码；
2) non-logic shell，不需要输入用户名和密码即可使用 bash；

5 数据的重定向

1) 输出>和>>
   我们在执行一个命令时，流程一般是先读取一个文件 standard input，然后执行相应的操作，最后输出，那输出也分为标准输出 standard output 和标准错误输出 standard error output
   数据的重定向是指利用符号“>”将本应显示在命令行窗口的信息输出到文件中，文件中已有数据则会覆写里面的数据；若使用“>>”则不会覆写，而是加在文末。
   当输出符号前面有 1 时则将正确的数据输出，2 则将错误的数据输出。
   同时将正确和错误的信息输出，有两种方法：
   - > 文件名 2>&1
   - &> 文件名
2) 输入<和<<
   一般用在 cat 之上的文件创建和文字的输入
   - cat > 文件名 表示创建一个新的文件和使用键盘输入
     如：
     [lvhongbin@MiWiFi-R3G-srv Desktop]$ cat > test2.txt
     Hello
     I'm writing!
     [Ctrl]+d 表示结束输入
   - cat > 新的文件 < 源文件 从源文件中把文字流传入新的文件
   - cat > 文件名 <<“结束字符串”
     如：[lvhongbin@MiWiFi-R3G-srv Desktop]$ cat > test2.txt << "eof"
     > Input
     > success!
     > eof

3) 什么场合会用到数据的重定向
   - 屏幕的输出很重要；
   - 后台执行的程序，不希望干扰到屏幕正常的输出信息的时候；
   - 想丢掉一些不想看到的信息；
   - 将正确和错误的信息分别输出；

6 命令与管道命令

1) 命令的合并
   - cmd1 ; cmd2
     不考虑命令的相关性，从左到右按顺序执行；
   - cmd1 && cmd2
     考虑命令的正确性，若 cmd1 错误，则 cmd2 不执行；
     &&命令会传递前一个命令的$?值，如果$?==0，则执行第二个命令，若$?<>0 则跳过第二命令，并把$?<>0 传递到第三个命令；
   - cmd1 || cmd2
     考虑命令的正确性，若 cmd1 正确，则 cmd2 不执行；
     如创建文件目录，如果不存在则创建，如果存在就不创建
     ls -l 文件名 || touch 文件名
     &&命令会传递前一个命令的$?值，如果$?<>0，则执行第二个命令，若$?==0 则跳过第二命令，并把$?==0 传递到第三个命令
   - 管道命令 pipe cmd1 | cmd2
     管道命令仅会处理 standard output，对于 standard error output 会予以忽略；
     管道命令必须能接收第一个命令的数据成为 standard input 继续处理才行；
     常用的管道命令如 less(将大量的输出做成可翻页的形式)，more，head，tail
2) 选取命令：cut, grep
   - cut [-dfc] 截取信息
     -d：后面接分割字符，与-f 一起使用，如-d ':' -f 5, 表示以冒号为分割，取第 5 段；
     -c n-：取每行从第 n 个起后面的字符，包括第 n 个的字符；
     -c -n：取每行从第 n 个起前面的字符，包括第 n 个的字符；
     [lvhongbin@MiWiFi-R3G-srv Desktop]$ ls -al .. | cut -c 16- | cut -c -9 实现多重截取；
   - grep 选取我们需要的信息所在的那一行
     grep [-acinv] [--color=auto] '查找字符串' filename
     [lvhongbin@MiWiFi-R3G-srv Desktop]$ ls -al .. | cut -c 16- | cut -c -9 |grep --color=auto "lvhongbin"
3) 排序命令：sort，wc, uniq
   - sort [-fbMnrtuk] [file or stdin]
     -f：忽略大小字母的差异；

-b：忽略最前面空格的影响；
-M：以月份的名字来排序；
-n：以纯数字进行排序；
-r：反向排序；
-u：uniq，相同的数据，仅出现一行；
-t：分隔符，默认是[Tab]键来分割；
-k：以分割的区间的第 n 个来排序
-t 和-k 一般一起使用：
[lvhongbin@MiWiFi-R3G-srv Desktop]$ ls -al .. | sort -t ' ' -k 2

- uniq [-ic]
  -i：忽略大小字母的差异；
  -c：进行计数；
- wc [-lwm]
  -l：仅列出行数；
  -w：仅列出字数（英文单字）；
  -m：多少字符；
- last 列出多少的登陆的情况
  [lvhongbin@MiWiFi-R3G-srv Desktop]$ last
  lvhongbi pts/0 :0 Wed Jan 3 03:01 - 03:02 (00:00)
  lvhongbi :0 :0 Wed Jan 3 03:00 - 03:11 (00:10)
  reboot system boot 3.10.0-693.el7.x Wed Jan 3 02:58 - 20:01 (-6:-56)

  wtmp begins Wed Jan 3 02:58:11 2018

  但是由于最后两行分别时空白行和含有“wtmp”字符串，所以如果需要计数到底有多少次登陆的话，需要去掉最后两行，然后在进行计数。
  所以可以使用命令：last | grep [a-zA-Z] | grep -r “wtmp” | wc -l

4) 双向重定向：tee
   tee [-a] file
   -a 表示累加到文件中，tee 将输出流保存一份到文件中，同时输出屏幕
5) 字符转换命令：tr，col，join，paste，expand
   tr -d 要删除的字符串 ‘旧字符串’ ‘新字符串’
   如 tr ‘[a-z]’ ‘[A-z]’
   可以利用 tr 命令将断行符号^M 去除，^M 可以用‘\r’替代出现在 tr 中被删除；
   col：[tab]换成空格
   join：将两个文件的同一行的相同部分只显示一次，然后两行合并，每一行都做这样的操作；
   paste：直接把两个文件的同一行放在一起，然后中间用[tab]键隔开；

expand：[tab]换成任意固定个数的空格；

6) 切割命令：split
   文件太大用于分割成任意大小的子文件
   [lvhongbin@MiWiFi-R3G-srv Desktop]$ split -b 1k 要分割的文件 前导文件名
   最后文件名将按照前导文件名 aa，前导文件名 ab，前导文件名 ac 等命名；、
7) 关于减号：-
   将前一个命令的 stdout 作为后一个命令的 stdin
   如 tar -cvf - /home | tar -xvf –

十二、 正则表达式和文件格式化处理

正则表达式依照严谨度的不同又分为基础正则表达式和拓展正则表达式
语系正则表达式的影响；
比如 LANG=C 时：01234…ABC…Zabc…z；
比如 LANG=zh_CN 时：01234…AaBbCc…Zz；

1) Grep 的高级用法
   Grep [-A] [-B]
   -A 后面可加数字 N，为 after 的意思，除了搜索的行被显示外，其前面 N 行也会被显示；
   -B 后面可加数字 N，为 after 的意思，除了搜索的行被显示外，其后面 N 行也会被显示；
   行尾用$表示；

2) 基础正则表达式字符，用在‘’内部
   - ^word：待查的字符串 word 在行首；
   - word$：待查的字符串 word 在行尾；
   - .：代表一定有一个任意字符
   - *：重复零个到无穷多个的前一字符；
   - \{n,m\}：连续 n 到 m 个前一个字符；
   - [list]：从字符集中选取；
   - [^list]：从字符集中反选；

3) 拓展正则表达式字符，用在‘’内部
   - ｜：或；
   - ＋：重复一个至多个前一个字符；
   - ？：0 个或者 1 个前一个字符；
   - （）：作为一个字符串组
   - （）+：重复多个组

4) sed 工具

sed [-nefr] ‘动作’

[动作] = n1, n2, function

Function 可选：a（新增）, c（替换）, d（删除）, i（插入）, p（打印）, s（替换），如‘2，5d’删除第二行到第五行 ‘2a thefirstline\ thesecondline’不同的行用“\”隔开 ‘2,5c thenewline’替换 2~5 行中间的行、

5) 格式化打印：printf

‘打印格式’ 实际内容

特殊样式：\f ： 清除屏幕；\n ： 输出新的一行；\r ： 亦即[Enter]按键；\t ： 水平[Tab]按键；\v ： 垂直[Tab]按键；\a ： 警告声音输出；\b ： 退格键；

在 C 语言中，%ns：那个 n 是数字，s 代表 string，即多少个字符；

%ni：那个 n 是数字，i 代表 integer，即多少个整数数字；

%N.nf：那个 n 和 N 是数字，N 代表包括小数点在内的所有位数之和，n 表示小数点后有多少位数；

如 printf '%15s\t %5.2f\t %5.2f\t %5.2f\t %5.2f\t \n' $(cat text1.txt)

```
      lvhongbin      123.00  44.10   22.00     2.00
     lvhongchao       3.40    2.33   31.11     3.20
```

6) awk：好用的数据处理工具

相比于 sed 处理一整行的数据，awk 倾向于将一行分成好几个“字段”进行处理；

变量：

- NF 每一行（$0）拥有的字段总数
- NR 目前 awk 所处理的是“第几行”的数据
- FS 目前的分隔符，默认是空格键
- $1，$2 等变量名称，表示每一行的每个字段，字段与字段之间通常用空格隔开，$0 表示一整行数据的意思；

整个 awk 的处理流程：

- 读入第一行，并将第一行的数据填入$0，$1，$2 等变量中；
- 依据条件类型的限制，判断是否需要执行后面的动作；
- 做完所有的条件类型和动作；
- 若有后续“行”的数据，则重复以上操作；

标准写法：文件 stdin | awk ‘条件{printf “正则表达式”, $1, $2,```$n, “自己的内容”}’

```
cat text1.txt | awk 'NR<3{printf "%10s %5.2f %5.2f %5.2f %5.2f %10s %10s\n",$1, $2, $3, $4, $5, " total:" NF, " line:" NR}'
```

7) 文件比较工具

Diff：比较两个文件的不同，通过比较行的差异，然后输出所有的差异；

Cmp：比较两个文件的不同，通过比较每个字符的差异，然后输出第一个的差异；

十三、 Shell Script

1)