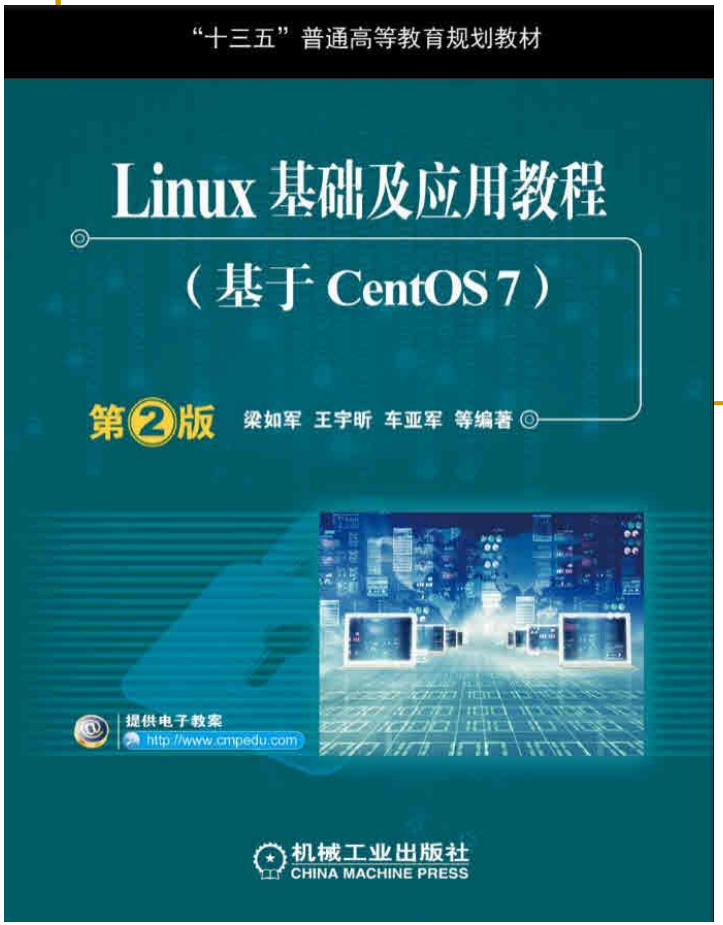


第2章 Linux操作基础

主讲人： 梁如军

2015-05-05

本章内容要点

- Shell及其功能
- 命令操作基础
- **Linux的文件和目录**
- 使用Linux的相关帮助
- 文件和目录操作命令
- 信息显示命令
- 正则表达式和**文本文件操作命令**
- Vim文本编辑器
- 命令补全、命令别名、命令历史
- 重定向、管道、命令替换、命令聚合
- **Shell变量** 和 **Shell环境**

梁如军（linuxbooks@126.com）

本章学习目标

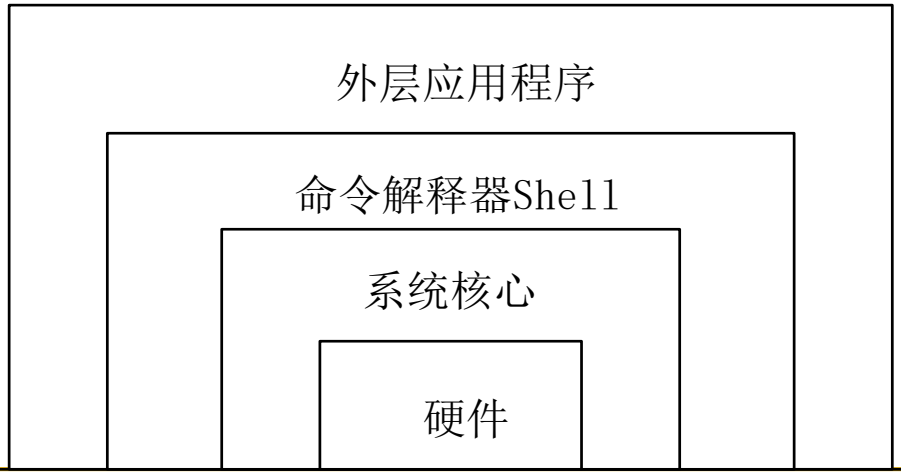
- 理解Shell功能和地位
- 掌握命令格式、文件和**通配符**
- 学会使用命令帮助
- 掌握各种常用命令的使用
- 学会使用**正则表达式**
- 学会使用VIM文本编辑器
- 学会使用命令补全、命令别名、命令历史
- 掌握重定向、管道、命令替换、命令组合
- 掌握Shell变量的定义、作用域和使用
- 学会配置用户工作环境

梁如军（linuxbooks@126.com）

SHELL及其功能

Shell

- Shell是系统的**用户界面**，提供了用户与内核进行交互操作的一种接口(命令解释器)。它接收用户输入的命令并把它送入内核去执行。起着协调用户与系统的一致性和在用户与系统之间进行交互的作用。
- Shell在Linux系统上具有**极其重要**的地位



梁如军（linuxbooks@126.com）

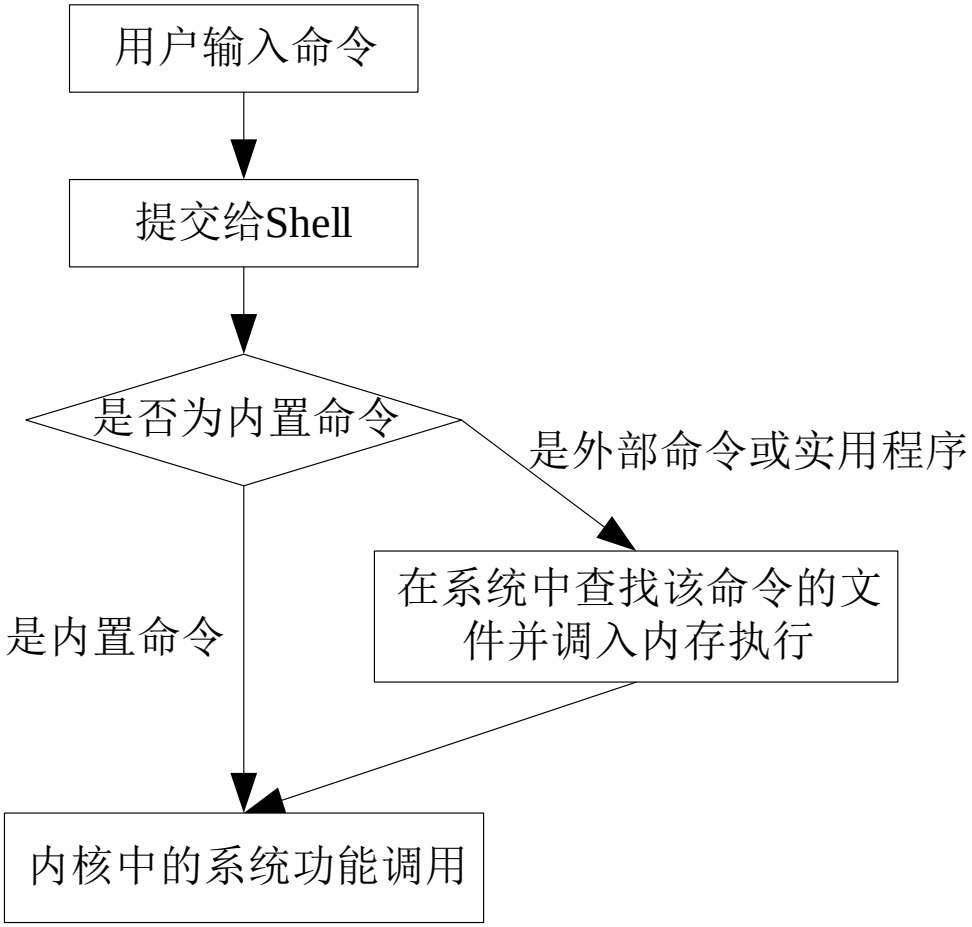
Shell的重要功能

- 命令行解释
- 命令序的多种执行顺
- 通配符（ wild-card characters ）
- 命令补全、别名机制、命令历史
- I/O重定向（ Input/output redirection ）
- 管道（ pipes ）
- **命令替换（``或$( )）**
- Shell编程语言（ Shell Script ）

梁如军（linuxbooks@126.com）

命令解释过程

- Shell可以执行
  - 内部命令
  - 应用程序
  - shell脚本



梁如军（linuxbooks@126.com）

Shell的主要版本

Bash（Bourne Again Shell）	**bash是大多数Linux系统的默认Shell。 bash与bsh完全向后兼容，并且在bsh的基础上增加和增强了很多特性。 bash也包含了很多C Shell和Korn Shell中的优点。 bash有很灵活和强大的编程接口，同时又有很友好的用户界面**
Ksh（Korn Shell）	Korn Shell（ksh）由Dave Korn所写。它是**UNIX系统上的标准Shell**。 在Linux环境下有一个专门为Linux系统编写的Korn Shell的扩展版本，即Public Domain Korn Shell（pdksh）。
tcsh （csh 的扩展）	tcsh是C Shell的扩展。tcsh与csh完全向后兼容，但它包含了更多的使用户感觉方便的新特性，其最大的提高是在命令行编辑和历史浏览方面

梁如军（linuxbooks@126.com）

Linux的元字符

- 在 Shell 中有一些具有特殊的意义字符，称为 **Shell 元字符**（shell metacharacters）。
- 若不以特殊方式（使用转义字符）指明，Shell并不会把它们当做普通文字符使用。

字符	含义	字符	含义
‘	强引用	*、?、！	通配符
“	弱引用	<、>、>>	重定向
\	转义字符	-	**选项标志**
$	变量引用	#	注释符
;	命令分离符	空格、换行符	命令分隔符

梁如军（linuxbooks@126.com）

LINUX命令格式

梁如军（linuxbooks@126.com）

命令基本格式

- 一般格式：
  - cmd [options] [arguments]
- 说明：
  - 最简单的Shell命令只有命令名，复杂的Shell命令可以有多个选项和参数。
  - 选项和参数都作为Shell命令执行时的输入，它们之间用空格分隔开。

注：Linux 区分大小写！

梁如军（linuxbooks@126.com）

Linux系统中可执行文件的分类

- 内置命令：出于效率的考虑，将一些常用命令的解释程序构造在Shell内部
- 外置命令：存放在/bin、/sbin目录下的命令
- 实用程序：存放在/usr/bin、/usr/sbin、/usr/share、/usr/local/bin等目录下的实用程序
- 用户程序：用户程序经过编译生成可执行文件后，可作为Shell命令运行
- Shell脚本：由Shell语言编写的批处理文件，可作为Shell命令运行

命令基本格式（续）

- 说明：
  - 单字符参数前使用一个减号（-）
  - 单词参数前使用两个减号（--）。
  - 多个单字符参数前可以只使用一个减号。
  - 操作对象（arguments）可以是文件也可以是目录，有些命令必须使用多个操作对象， 如cp命令必须指定源操作对象和目标操作对象。
  - 并非所有命令的格式都遵从以上规则，例如dd、find等

命令在正常执行结果后返回一个 0 值，如果命令出错，则返回一个非零值 (在shell中可用变量 $? 查看)。

命令基本格式举例

- $ ls
- $ ls -lRa /home
- $ cat abc xyz
- $ ls --help
- $ su -
- $ passwd
- $ date
- $ cal 2011

梁如军（linuxbooks@126.com）

Linux 常用命令

cat	查看文件内容	more/less	查看文件内容
cd	切换工作目录	touch	改变文件的时间属性
chown	改变文件属权	mv	改名或移动文件
chmod	改变文件权限	pwd	显示当前所在的目录
clear	清除屏幕	rm	删除文件或目录
cp	拷贝文件	find	查找文件
ln	创建文件链接	which	寻找命令
ls	显示目录内容	tar	文件打包
mkdir rmdir	创建/删除目录	[g]zip/unzip 7za	文件压缩和解压

梁如军（linuxbooks@126.com）

目录和文件名的命名规则

- 在Linux下可以使用长文件或目录名
  - 可以**长达255个字符**
- 可以给目录和文件取任何名字，但必须遵循下列的规则：
  - 除了/之外，所有的字符都合法
  - 有些字符最好不用，如空格符、制表符、退格符和字符：？，@ # $ & () \ | ; ‘ ’ “ ” < >等。
  - 避免使用+、-或.来作为普通文件名的第一个字符
  - 大小写敏感
  - 以“.”开头的文件或目录是隐含的

梁如军（linuxbooks@126.com）

通配符

- *：匹配任何字符和任何数目的字符
- ?：匹配单一数目的任何字符
- [ ]：匹配[ ]之内的任意一个字符
- [!]：匹配除了[!]之外的任意一个字符，!表示非的意思

“*”能匹配文件或目录名中的“.”。
“*”不能匹配首字符是“.”的文件或目录名。

梁如军（linuxbooks@126.com）

通配符使用举例

- ls *.c
  - 列出当前目录下的所有C语言源文件。
- ls /home/*/*.c
  - 列出/home目录下所有子目录中的所有C语言源文件。
- ls n*.conf
  - 列出当前目录下的所有以字母n开始的conf文件。
- ls test?.dat
  - 列出当前目录下的以test开始的，随后一个字符是任意的.dat文件。
- ls [abc]*
  - 列出当前目录下的首字符是a或b或c的所有文件。
- ls [!abc]*
  - 列出当前目录下的首字符不是a或b或c的所有文件。
- ls [a-zA-Z]*
  - 列出当前目录下的首字符是字母的所有文件

梁如军（linuxbooks@126.com）

LINUX下的文件与目录

梁如军（linuxbooks@126.com）

文件的类型

- 普通文件 (-)
- 目录 (d)
- 符号链接 (l)
- 字符设备文件 (c)
- 块设备文件 (b)
- 套接字 (s)
- 命名管道 (p)

梁如军（linuxbooks@126.com）

普通文件

- 普通文件仅仅就是字节序列，Linux 并没有对其内容规定任何的结构。
- 普通文件可以是程序源代码（c、c++、python、perl等）、可执行文件（文件编辑器、数据库系统、出版工具、绘图工具等）、图片、声音、图像等。
- Linux 不会区别对待这些普通文件，只有处理这些文件的应用程序才会根据文件的内容赋予相应的含义。
- 在Linux环境下，只要是可执行的文件并具有可执行属性它就能执行，不管其文件名后缀是什么。但是对一些数据文件一般也遵循一些文件名后缀规则。

梁如军（linuxbooks@126.com）

目录

- 目录文件是由一组目录项组成，目录项可以是对其他文件的指向也可以是其下的子目录指向。
- 一个文件的名称是存储在他的父目录中的，而并非同文件内容本身存储在一起。
- 硬连接文件实际上就是在某目录中创建目录项，从而使不止一个目录可以引用到同一个文件。这种链接关系由 **ln** 命令行来建立。
- 硬链接并不是一种特殊类型的文件，只是因为在文件系统中允许不止一个目录项指向同一个文件。

梁如军（linuxbooks@126.com）

用户主目录

- 用户登录后，将会进入一个系统指定的专属目录，即用户的主目录，该目录名通常为用户的登录账号。如
  - 用户osmond的主目录为：/home/osmond
- 在创建用户时，系统管理员会给每个用户建立一个主目录，通常在 /home/ 目录下。
- 用户对自己主目录的文件拥有所有权，可以在自己的主目录下进行相关操作。
- 每个用户名对应一个用户 ID 号（一个数字）；每个用户都被分配到一个指定的组 (group) 中。
- 默认情况下 RHEL/CentOS 在创建用户的同时会创建一个和用户同名的私有组。

符号链接

- 符号链接又称**软链接**，是指将一个文件指向另外一个文件的文件名。
- 这种符号链接的关系由 **ln -s** 命令行来建立。

梁如军（linuxbooks@126.com）

硬链接和软链接的比较

- 硬链接
  - 链接文件和被链接文件必须位于同一个文件系统内
  - 不能建立指向目录的硬链接
- 软链接
  - 链接文件和被链接文件可以位于不同文件系统
  - 可以建立指向目录的软链接

梁如军（linuxbooks@126.com）

设备文件

- 设备是指计算机中的外围硬件装置，即除了CPU和内存以外的所有设备。通常，设备中含有数据寄存器或数据缓存器、设备控制器，它们用于完成设备同CPU或内存的数据交换。
- 在 Linux 下，为了屏蔽用户对设备访问的复杂性，采用了设备文件，即可以通过象访问普通文件一样的方式来对设备进行访问读写。
- 设备文件用来访问硬件设备，包括硬盘、光驱、打印机等。每个硬件设备至少与一个设备文件相关联。
- 设备文件分为：**字符设备**（如：键盘）和**块设备**（如：磁盘）。

Linux下设备的使用

- 设备的使用方法
  - 用户可以用设备名来使用设备
  - 用户可以用访问文件的方法来使用设备
- 设备名以文件系统中的设备文件的形式存在
- 所有的设备文件存放在/dev目录下
- 几个特殊的设备
  - /dev/null　　—空设备
  - /dev/zero　　—零设备

梁如军（linuxbooks@126.com）

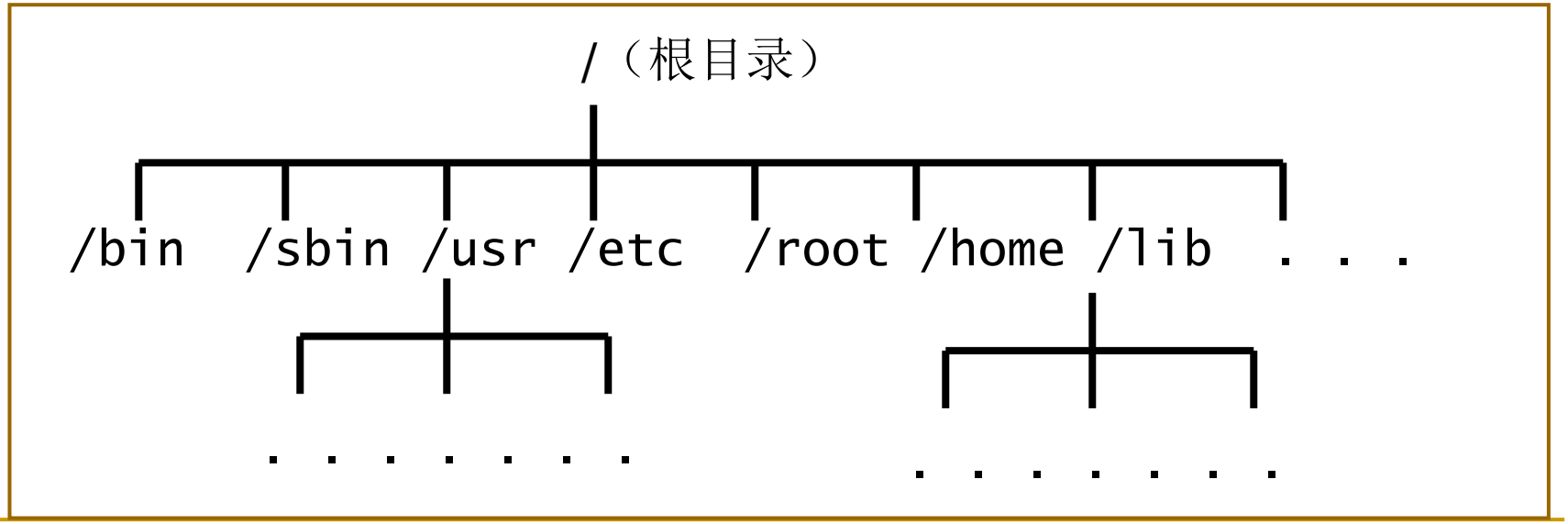
套接字和命名管道

CentOS
The Community ENTerprise Operating System

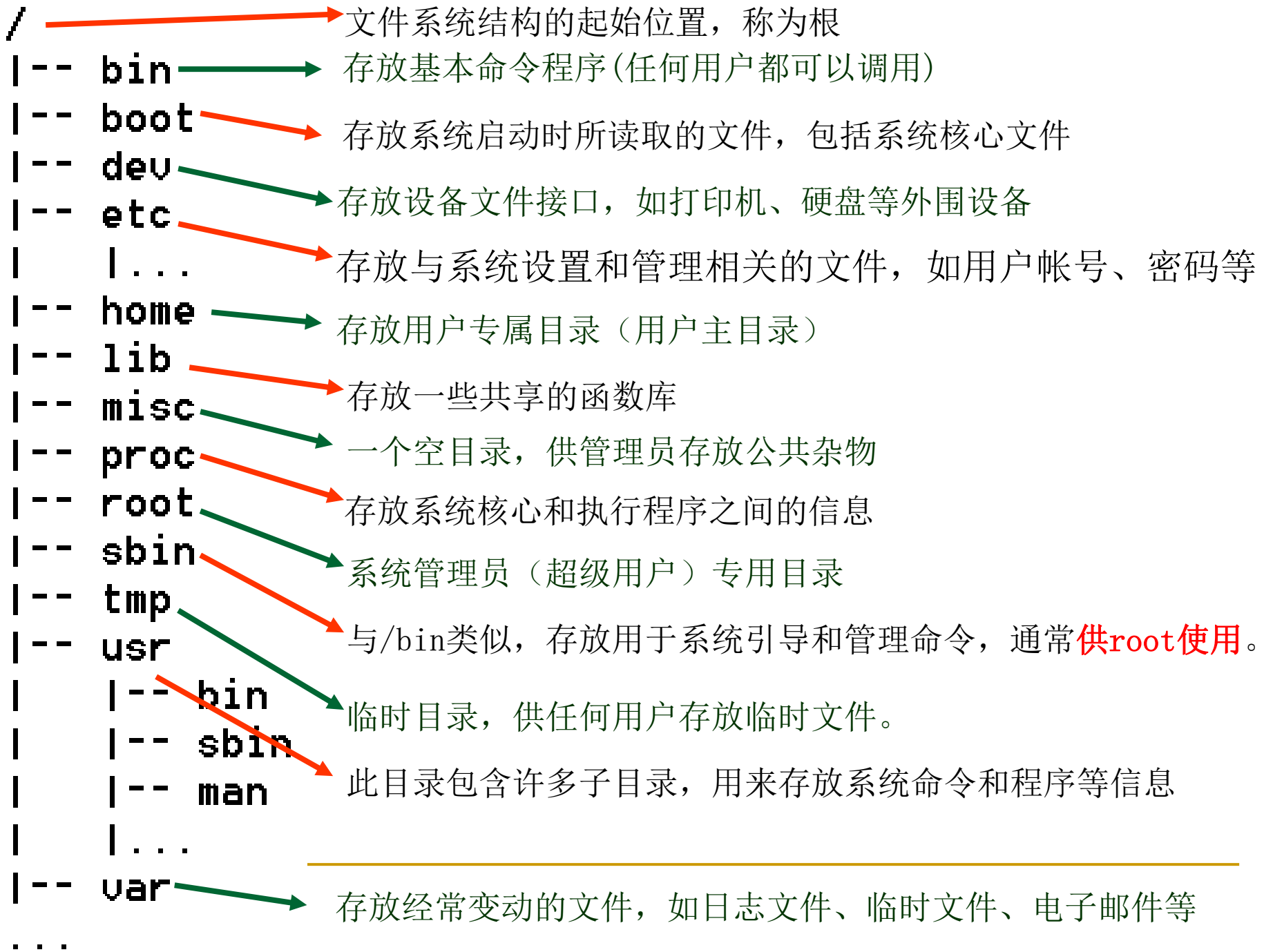
- 套接字和命名管道是 Linux 环境下实现进程间通信（IPC）的机制。
- 命名管道（FIFO）文件允许运行在同一台计算机上的两个进程之间进行通信。
- 套接字（socket）允许运行在不同计算机上的进程之间相互通信。
- 套接字和命名管道通常是在进程运行时创建或删除的，一般无需系统管理员干预。

Linux 的目录结构

- Linux 文件系统是一个目录树的结构，文件系统结构从一个根目录开始，根目录下可以有任意多个文件和子目录，子目录中又可以有任意多个文件和子目录。
- Linux 的这种文件系统结构使得一个目录和它包含的文件/子目录之间形成一种层次关系。



梁如军（linuxbooks@126.com）


/
文件系统结构的起始位置，称为根
|-- bin
存放基本命令程序(任何用户都可以调用)
|-- boot
存放系统启动时所读取的文件，包括系统核心文件
|-- dev
存放设备文件接口，如打印机、硬盘等外围设备
|-- etc
| |...
存放与系统设置和管理相关的文件，如用户帐号、密码等
|-- home
存放用户专属目录（用户主目录）
|-- lib
存放一些共享的函数库
|-- misc
一个空目录，供管理员存放公共杂物
|-- proc
存放系统核心和执行程序之间的信息
|-- root
系统管理员（超级用户）专用目录
|-- sbin
与/bin类似，存放用于系统引导和管理命令，通常供root使用。
|-- tmp
临时目录，供任何用户存放临时文件。
|-- usr
| |-- bin
| |-- sbin
| |-- man
此目录包含许多子目录，用来存放系统命令和程序等信息
| |...
|-- var
存放经常变动的文件，如日志文件、临时文件、电子邮件等
...

CentOS
The Community ENTerprise Operating System

文件和目录操作命令

梁如军（linuxbooks@126.com）

常用的目录操作命令

CentOS
The Community ENTerprise Operating System

命令	功能
ls	显示文件和目录列表
cd	切换目录
pwd	显示当前工作目录
mkdir	创建目录
rmdir	删除空目录
tree	显示目录树

梁如军（linuxbooks@126.com）

当前工作目录

- 用户目前所处的目录
- 用户登录后进入的目录通常是自己的主目录
- 可用 pwd 命令查看用户的当前目录
- 可用 cd 命令来切换目录
- 一些特殊字符的特殊含义：
  - “ .” 表示当前目录
  - “..” 表示当前目录的上一级目录（父目录）
  - “-” 表示用 cd 命令切换目录前所在的目录
  - “~” 表示用户主目录的绝对路径名

梁如军（linuxbooks@126.com）

路径 (path)

- 路径是指文件或目录在文件系统中所处的位置
- 绝对路径
  - 以斜线（/）开头
  - 描述到文件位置的**完整说明**
  - 任何时候你想指定文件名的时候都可以使用
- 相对路径
  - 不以斜线（/）开头
  - 指定**相对于你的当前工作目录而言**的位置
  - 可以被用作指定文件名的简捷方式

梁如军（linuxbooks@126.com）

ls命令

CentOS
The Community ENTerprise Operating System

- 功能：显示文件或目录信息
- 格式：　ls [选项]　[目录或是文件]
- 说明：
  - 对于目录，该命令将列出其中的所有子目录与文件。
  - 对于文件，ls 将输出其文件名以及所要求的其他信息。
  - 默认情况下，输出条目按字母顺序排序。
  - 当未给出目录名或文件名时，就显示当前目录的信息。

梁如军（linuxbooks@126.com）

ls命令选项

选项	说明
-a	列出目录下的所有文件，包括以 . 开头的隐含文件。
-l	列出文件的详细信息，通常称为“长格式”。
-d	输入参数是目录时，只显示该目录本身。
-A	显示除 “.” 和 “..” 外的所有文件。
-R	递归地列出所有子目录下的文件。
-h	以人类易读的单位显示文件大小。
-S	以文件大小排序输出。
-t	以时间排序输出。

梁如军（linuxbooks@126.com）

ls命令举例

ls	列表显示当前目录下的文件和目录
ls -a	列表显示当前目录下的文件和目录（包括隐含文件和目录）
ls -l	以长格式列表显示结果
ls -R	递归地显示当前目录及其子目录下的文件和目录
ls -dl /usr/share/	仅显示/usr/share/目录本身，而非/usr/share/ 目录中的内容
ls -lt	按最后修改时间顺序，以长格式列出当前目录下的文件

梁如军（linuxbooks@126.com）

mkdir和tree命令举例

- mkdir
  - mkdir /home/lrj/mybin　　#创建一个空目录
  - mkdir -p mydoc/FAQ　　#创建一个空目录树
  - mkdir -p /srv/www/{abc,bcd}/htdocs　　#创建/srv/www/abc/htdocs和/srv/www/bcd/htdocs目录
- tree
  - tree /srv/www　　# 显示/srv/www 目录树
  - tree -L 3 /srv　　#显示 /srv/ 的 3级目录树

梁如军（linuxbooks@126.com）

pwd和cd命令举例


```
$ pwd
$ cd
$ cd /some/dir/
$ cd ~
$ cd ..
$ cd ../..
$ cd -
```


梁如军（linuxbooks@126.com）

常用的文件操作命令

命令	功能
touch	生成一个空文件或更改文件的时间
cp	复制文件或目录
mv	移动文件或目录、文件或目录改名
rm	删除文件或目录
ln	建立链接文件
find	查找文件
file/stat	查看文件类型或文件属性信息

梁如军（linuxbooks@126.com）

touch命令

- 功能：生成新的空文件或更改现有文件的时间
  - 若文件不存在，系统会建立一个文件
  - 默认情况下将文件的时间记录改为当前时间
- 格式：touch [参数] <文件> …
- 参数
  - -a：只更改访问时间。
  - -m：只更改修改时间。
  - -t <STAMP>：使用[[CC]YY]MMDDhhmm[.ss]格式的时间而非当前时间。
  - -r <参考文件或目录>：使用指定文件的时间属性而非当前时间。

梁如军（linuxbooks@126.com）

touch命令举例

CentOS
The Community ENTerprise Operating System

```
$ touch newfile
$ touch  file
$ touch  -a file
$ touch  -m file
$ touch  -t 200701311200  file
$ touch  -r oldfile  newfile
```


梁如军（linuxbooks@126.com）

文件的时间戳

- GNU/Linux 文件的3种类型的时间戳：
  - mtime: 最后修改时间 (ls -lt)
  - ctime: 状态改变时间 (ls -lc)
  - atime: 最后访问时间 (ls -lu)
- 说明
  - ctime并非文件创建时间。
  - 覆盖一个文件会改变mtime、ctime和atime三类时间。
  - 改变文件的访问权限或拥有者会改变文件的ctime和atime。
  - 读文件会改变文件的atime。

梁如军（linuxbooks@126.com）

判断文件类型

CentOS
The Community ENTerprise Operating System

- 文件可以包含许多类型的数据
- 在打开前**检查文件的类型**来决定要使用的恰当命令或程序
- 命令
  - file [选项] <文件名>…
  - stat [选项] <文件名>…
- 举例
  - file /etc/passwd /bin/bash /dev/console
  - stat /etc/passwd

梁如军（linuxbooks@126.com）

cp命令

- 功能：复制文件或目录。
- 格式：cp [参数] <源> <目标>
- 说明
  - 若复制的目标文件已存在，则被覆盖。
  - 可以将多个源文件复制到目标目录中。
  - 可以将源目录复制为指定的目标目录（目标目录不存在）。
  - 可以将源目录复制到指定的目标目录中。

梁如军（linuxbooks@126.com）

cp命令的常用选项

参数	说明
-a	等价于 –dpR
-R，-r	递归地复制目录及目录内的所有项目
-p	在复制文件过程中保留文件属性，包括属主、组、权限与时间戳
-d	当复制符号链接的源文件时，目标文件也将创建符号链接且指向源文件所链接的原始文件
-f	强制复制，不管目标是否存在
-i	交互式复制，覆盖文件前需要确认
-u	只有当源文件的状态改变时间（ctime）比目标文件更新时或目标尚不存在时才进行复制

cp命令举例

cp file1 file2	将当前目录下的file1复制为file2
cp some/dir/file1 someother/dir/	将某目录下的文件复制到另一个目录
cp /bin/cpio ~/bin	复制文件 /bin/cpio 到 ~/bin 目录
cp abc bcd mydoc	将两个指定的文件复制到mydoc目录下
cp some/dir/f* someother/dir/	将某目录下的以f开头的文件复制到另一个目录
cp /bin/?sh .	使用“?”通配符复制多个文件到当前目录下
cp file{1,2,3} someother/dir/	将当前目录下的file1、file2、file3复制到另外一个目录
cp /etc/httpd/conf/httpd.conf{,.orig}	将/etc/httpd/conf/目录下的httpd.conf拷贝成httpd.conf.orig
cp -r some/dir/ someother/dir/	将某目录下的所有文件（包括子目录）复制到另一个目录

梁如军（linuxbooks@126.com）

ln命令

- 功能：创建链接文件。
- 格式：ln [参数] <被链接的文件> <链接文件名>
- 参数：
  - -s：创建符号链接，而非硬链接。
  - -f：强行创建链接，不论其是否存在。
  - -i：覆盖原有文件之前先询问用户。
- 举例：
  - $ ln somefile hardlinkfile
  - $ ln -s somefile softlinkfile
  - $ ln -s somedir softlinkfile

mv和rm命令举例

- mv
  - mv file file.bak
  - mv file1 mydata/
  - mv -i file1 mydata/file2
  - mv datadir1 /home/zhang/data
- rm
  - rm ash
  - rm .*
  - rm -f file{1,3,5}
  - rm -r myusr/
  - rm -rf myusr/

梁如军（linuxbooks@126.com）

find命令

- 功能：在文件系统中**查找匹配的文件**
- 格式

  find [<起始目录> ...] [<选项表达式>] [<条件匹配表达式>] [<动作表达式>]
- 说明
  - <起始目录>：对每个指定的 <起始目录> 递归搜索目录树
    - **省略<起始目录>时表示当前目录**
  - <选项表达式>：控制 find 命令的行为
  - <条件匹配表达式>：根据匹配条件查找文件
  - <动作表达式>：指定对查找结果的操作，默认为显示在标准输出（-print）
  - 不带任何参数的 find 命令将在屏幕上递归显示当前目录下的文件列表。

梁如军（linuxbooks@126.com）

find 命令——选项表达式

选项	说明
-L	如果遇到符号链接文件，就跟踪链接所指的文件
-regextype TYPE	指定 -regex 和 -iregex 使用的正则表达式类型，默认为 emacs
-depth/-d	查找目录自身之前先处理目录中的文件（即深度优先）
-mount/-xdev	查找文件时不跨越文件系统
-maxdepth LEVELS	设置最大的查找深度
--help	显示 find 命令帮助信息
--version	显示 find 的版本

梁如军（linuxbooks@126.com）

find命令举例（1）

- 递归显示当前目录的文件列表

  $ find
- 递归显示 / 目录的文件列表

  $ find /
- 递归显示 / 目录的文件列表（仅限于3层目录）

  $ find / -maxdepth 3
- 递归显示 / 目录的文件列表（仅限于 / 文件系统）

  $ find / -xdev
- 递归显示 /home、/www、/srv 目录的文件列表

  $ find /home /www /srv

梁如军（linuxbooks@126.com）

find 命令
——条件表达式（1）文件名

条件	说明
-name PATTERN	匹配文件名
-iname PATTERN	匹配文件名（忽略大小写）
-lname PATTERN	匹配符号链接文件名
-ilname PATTERN	匹配符号链接文件名（忽略大小写）
-wholename PATTERN	匹配文件的完整路径（不把 ‘/’ 和 ‘.’ 作为特殊字符）
-iwholename PATTERN	匹配文件的完整路径（忽略大小写）

PATTERN 使用 Shell 的匹配模式
- 可以使用 Shell 的通配符（*、？ []）
- 要用“”或‘’括起来

梁如军（linuxbooks@126.com）

find命令举例（2）

CentOS
The Community ENTerprise Operating System

```
$ find -name myfile
$ find -maxdepth 2 -name symfony
$ find -name 'd*'
$ find -name '???'
$ find -name 'ch[0-2][0-9].txt*'
$ find -iname '[a-z]*'
$ find ~   -name  "*.txt"
$ find .   -name  "[A-Z]*"
$ find /etc -name  "host*"
```


梁如军（linuxbooks@126.com）

find 命令
——条件表达式（2）文件名续

条件	说明
-regex REPATTERN	以正则表达式匹配文件名
-iregex REPATTERN	以正则表达式匹配文件名（忽略大小写）

REPATTERN 使用正则表达式匹配模式
- 使用 -regextype TYPE 指定的正则表达式类型
- 默认为 emacs 类型

梁如军（linuxbooks@126.com）

find命令举例（3）

- $ find ~ -regex '.*'
- $ find ~/book -regex '.*ch0.*'
- $ find ~/book -regex '.*ch[0-9]+'
- $ find ~/book -iregex '.*ch[0-9]+\.txt'

梁如军（linuxbooks@126.com）

find 命令
——条件表达式（3）时间属性

条件	说明
-amin N	查找 N 分钟以前被访问过的所有文件
-atime N	查找 N 天以前被访问过的所有文件
-cmin N	查找 N 分钟以前文件状态被修改过的所有文件（比如权限修改）
-ctime N	查找 N 天以前文件状态被修改过的所有文件 （比如权限修改）
-mmin N	查找 N 分钟以前文件内容被修改过的所有文件
-mtime N	查找 N 天以前文件内容被修改过的所有文件
-used N	查找被改动过之后，在N天内被存取过的所有文件

注：以上选项中的 N 可以有三种输入方式：N，+N，-N

- 20 表示等于 20
- +20 表示大于 20 （21，22，23 等）
- -20 表示小于 20 （19，18，17 等）

梁如军（linuxbooks@126.com）

find命令举例（4）

- 在/home下查找最近2天之内改动过的文件

  $ find /home -mtime -2
- 在/home下查找最近2天之内被访问过的文件

  $ find /home -atime -2
- 在/home下查找60分钟之前改动过的文件

  $ find /home -mmin +60
- 在/home下查找最近5分钟之内修改过的文件

  $ find /home -mmin -5
- 在/home下查找30分钟之前被访问过的文件

  $ find /home -amin +30
- 在/home下查找最近在2日内被改动之后被访问过的文件

  $ find /home -used -2

梁如军（linuxbooks@126.com）

find 命令
——条件表达式（4）时间属性续

条件	说明
-newer FILE	查找所有比 FILE 的内容修改时间新的文件
-cnewer FILE	查找所有比 FILE 的状态修改时间新的文件（比如权限修改）
-anewer FILE	查找所有比 FILE 的访问时间新的文件

FILE 为要比对的样板文件或目录

梁如军（linuxbooks@126.com）

find命令举例（5）

- 在/home下查找更新时间比tmp.txt新的文件或目录

  $ find /home -newer tmp.txt

- 在/home下查找存取时间比tmp.txt新的文件或目录

  $ find /home -anewer tmp.txt

梁如军（linuxbooks@126.com）

find 命令
——条件表达式（5）用户和组

条件	说明
-uid N	用户ID为N的所有文件
-gid N	组ID为N的所有文件
-user USERNAME	用户名为 USERNAME 的所有文件
-group GROUPNAME	组名为 GROUPNAME 的所有文件
-nouser	文件属于不在 /etc/passwd 文件中的用户
-nogroup	文件属于不在 /etc/group 文件中的组

N 可以使用

N，+N，-N

梁如军（linuxbooks@126.com）

find 命令举例（6）

- 查找在系统中已作废用户（在/etc/passwd中无记录）的文件

  $ find / -nouser
- 查找在系统中已作废组（在/etc/group中无记录）的文件

  $ find / -nogroup
- 查找属于 osmond 用户的文件

  $ find / -user osmond
- 查找属于 students 组的文件

  $ find / -group students
- 查找UID为502的所有文件

  $ find / -uid 502
- 查找UID大于600的所有文件

  $ find / -uid +600
- 查找GID小于60的所有文件

  $ find / -gid -60

梁如军（linuxbooks@126.com）

find 命令
——条件表达式（6）文件权限

条件	说明
-perm MODE	精确匹配权限模式为 MODE 的文件
-perm -MODE	匹配权限模式至少为 MODE 的文件（用户、组和其他人这三种权限必须都匹配）
-perm /MODE 或 -perm +MODE	匹配权限模式至少为 MODE 的文件（用户、组和其他人这三种权限中有一种匹配即可）

MODE 与 chown 命令的书写方式一致，既可以使用字符模式也可以使用八进制的数值模式，通常使用八进制的数值模式

梁如军（linuxbooks@126.com）

find 命令举例（7）

- 在/home目录下查找权限为700的所有文件或目录

  $ find /home -perm 700

- 在/home目录下查找权限至少为740的所有文件或目录

  $ find /home -perm -740

- 在~/bin目录下查找权限至少为111的所有文件或目录（或者属主可执行，或者组可执行，或者其他人可执行）

  $ find ~/bin -perm /111

梁如军（linuxbooks@126.com）

find 命令
——条件表达式（7）类型和大小

条件	说明
-type TYPE	查找类型为 TYPE 的文件，可用的类型： b-块设备文件; c-字符设备文件; d-目录文件; p-命名管道; f-普通文件; l-符号链接; s-socket 文件
-links N	查找有N个链接的文件
-inum N	查找文件inode为N的文件
-samefile NAME	查找与NAME文件具有相同inode的文件
-size N[bcwkMG]	查找文件大小为N的文件，单位可以为： b-512 字节的块（省略单位的默认值）; c-字节; w-双字节; k-Kilobytes; M-Megabytes; G-Gigabytes
-empty	查找空文件（大小为0）

N 可以使用： N, +N, -N

梁如军（linuxbooks@126.com）

find 命令举例（8）

- 在当前目录下查找目录

  $find . -type d
- 在/home目录下查找硬连接数大于2的文件

  $find /home -links +2
- 在当前目录下查找长度大于10kB的文件

  $find . -size +10k
- 在/tmp目录下查找长度等于1GB的文件

  $find /tmp -size 1G
- 在当前目录下查找长度小于10MB的文件

  $find . -size -10M
- 在/var/log目录下查找所有的空文件或目录

  $ find /var/log -empty

梁如军（linuxbooks@126.com）

find 命令——组合条件表达式

- 可以使用逻辑运算符与、或、非组成的复合条件，并可以用()改变默认的操作符优先级。
- **若以空格作为各个表达式的间隔符，则各个表示式之间是与关系。**

(EXPR)	改变操作符优先次序，一些 UNIX 版的 find 命令要使用 \(EXPR \) 形式
! EXPR	表示对表达式取反
EXPR1 EXPR2	与逻辑，若 EXPR1 为假，将不再评估 EXPR2
EXPR1 -a EXPR2	与 EXPR1 EXPR2 功能一致
EXPR1 -o EXPR2	逻辑或，若 EXPR1 为真，将不再评估 EXPR2
EXPR1 , EXPR2	若 EXPR1 为假，继续评估 EXPR2

梁如军（linuxbooks@126.com）

find命令举例（9）

- 查找 /tmp 目录下21天之前访问过的大于 10G 的文件

  $ find /tmp -size +10M -a -atime +21
- 查找 /home 目录下属主为 jjheng 或 osmond 的大于 10M 的文件

  $ find /home \(-user jjheng -o -user osmond \) -size +10M
- 查找 /tmp 目录下的属主不是 osmond 的文件

  $ find /tmp ! -user osmond
- 在 /mnt 下查找 *.txt 且文件系统类型不为 vfat 的文件

  $ find /mnt -name '*.txt' ! -fstype vfat
- 在 /tmp 下查找名为 l 开头且类型为符号链接的文件

  $ find /tmp -name 'l*' -type l
- 找出 /var/log 目录下所有的5天前修改过的.log 文件

  $ find /var/log -name '*.log' -mtime +5
- 查找所有比 FILE1 的内容修改时间新的且比 FILE2 旧的文件

  $ find -newer FILE1 ! -newer FILE2

梁如军（linuxbooks@126.com）

find 命令——动作表达式

- -print
  - 在标准输出上列出查找结果（每行一个文件）
- -ls
  - 使用 ‘ls -dils’ 在标准输出上列出查找结果
- -prune
  - **忽略对某个目录的查找**
- -exec COMMAND {} \;
  - 对符合查找条件的文件执行 Linux 命令
- -ok COMMAND {} \;
  - 对符合查找条件的文件执行 Linux 命令；与 -exec 不同的是，它会询问用户是否需要执行

- { } 两个大括号之间不能有空格，表示查找到的对象
- \; 表示命令结束，反要留空斜杠与前面的大括号之间必须格

梁如军（linuxbooks@126.com）

find命令举例（10）

- 查找并列出当前目录下不安全的文件（世界可读写执行）

  $ find . -perm -007 -ls
- 查找 logs 目录下的所有的 .log 文件并查看它的详细信息

  $ find logs -name "*.log" -type f -exec ls -l {} \;
- 查找当天修改过的普通文件

  $ find . -type f -mtime -1 -exec ls -l {} \;
- 查找当前目录下的.php文件、并用grep过滤出包含include的行

  $ find . -name "*.php" -exec grep "include" {} \; -print

梁如军（linuxbooks@126.com）

find命令举例（10）续

- 查找并删除当前目录及其子目录下所有扩展名为.tmp 的文件

  $ find . -name '*.tmp' -exec rm {} \;

- 在logs目录中查找7天之内未修改过的文件并在删除前询问

  $ find logs -type f -mtime +7 -exec -ok rm {} \;

- 从当前目录下查找所有以.repo为后缀的文件，并为其改名（添加.bak后缀）

  $ find . -name '*.repo' -type f -exec mv {} {}.bak \;

- 查询并删除一周以来从未访问过的以 .o 结尾，或名为 a.out 且不存在于 vfat 文件系统中的所有文件

  $ find / (-name a.out -o -name ‘*.o’) -atime +7 ! -fstype vfat -exec rm {} \;

梁如军（linuxbooks@126.com）

find命令举例（10）续2

- 显示当前目录下除 book 目录之外的所有文件
  $ find . -name book -prune -o -print
- 查找当前目录下（除了 book 目录）之外的所有 .sh 文件
  $ find . -name book -prune -o -name '*.sh' -print
- 显示当前目录下（除了 book/server 目录）之外的所有文件
  $ find . -path ./book/server -prune -o -print
- 查找当前目录下（除了 book/server 目录）之外的所有 .sh 文件
  $ find . -path ./book/server -prune -o -name '*.sh' -print
- 显示当前目录下除 book/server 和 book/server-utf8 目录的所有文件
  $ find . -path './book/server*' -prune -o -print
- 查找当前目录下（除了 book/server 和 book/server-utf8 目录）的所有 .sh 文件
  $ find . -path './book/server*' -prune -o -name '*.sh' -print
- 查找当前目录下（除了 book/server 和 book/basic 目录）的所有 .sh 文件
  $ find . \(-path ./book/server -o -path ./book/basic \) -prune -o -name '*.sh' -print

find命令举例（11）

- 下面 find 命令的书写形式均等价

```
$ find -name \*.sh -exec cp {} /tmp \;
$ find -name '*.sh' -exec cp {} /tmp ';'
$ find -name "*.sh" -exec cp {} /tmp ";"
$ find -name \*.sh -exec cp \{\} /tmp \;
$ find -name '*.sh' -exec cp '{}' /tmp ';'
$ find -name “*.sh” -exec cp “{}” /tmp ";”
```


梁如军（linuxbooks@126.com）

文件打包和压缩命令

梁如军（linuxbooks@126.com）

常用的文件打包和压缩命令

命令	功能
xz	使用LZMA 算法的高性能压缩/解压工具
gzip	流行的 GNU gzip 数据压缩/解压程序
bzip2	免费的，无专利的高性能数据压缩工具
zip/unzip	与WinZIP兼容的压缩/解压工具
rar	与WinRAR兼容的压缩/解压工具
7za	使用LZMA 算法的高性能压缩/解压工具
tar	文件打包、归档工具

梁如军（linuxbooks@126.com）

打包和压缩文件的文件后缀

文件后缀	说明
.bz2	用 bzip2 压缩的文件
.gz	用 gzip 压缩的文件
.xz	用 xz 压缩的文件
.tar	用 tar 打包的文件，也称 tar 文件
.tbz	tar 打包时用 bzip2 压缩的文件
.tgz	tar 打包时用 gzip 压缩的文件
.zip	用 zip/winzip 压缩的文件
.rar	用 rar 压缩的文件
.7z	用 7za 压缩的文件

梁如军（linuxbooks@126.com）

gzip

- Linux下常用的压缩和解压缩命令。
- 由官方仓库的gzip软件包提供。
- 压缩后 gzip 会在每个文件的后面添加扩展名 .gz。
- 压缩后原文件会被自动删除。
- 在 windows 下可以用 winzip 或 winrar或7-zip 解压。

梁如军（linuxbooks@126.com）

gizp命令

- 用法：gzip　[选项]　文件列表
- 选项：
  - -d: 解开压缩文件。
  - -f: 强行压缩文件，不理会文件名称或硬链接是否存在以及该文件是否为符号链接。
  - -l: 列出压缩文件的相关信息（压缩文件的大小；未压缩文件的大小；压缩比；未压缩文件的名字）。
  - -n: 压缩文件时，不保存原来的文件名称及时间戳（默认为保存，即-N）。
  - -r : 递归处理，将指定目录下的所有文件及子目录一同处理。
  - -t : 测试压缩文件是否正确无误。
  - -v : 显示指令执行过程。
  - -<压缩率> : 压缩率是一个介于1～9的数值，默认值为“6”，数值越大压缩率越高。
  - --best 参数等价于-9；--fast参数等价于-1。

梁如军（linuxbooks@126.com）

gizp命令举例

- 压缩文件filename

  $ gzip filename

- 压缩文件 file1和file2并显示执行过程

  $ gzip -v file1 file2

- 递归地高度压缩mydir目录下的所有文件（逐个文件进行）

  $ gzip -9r mydir

- 显示当前目录下所有压缩过的gz文件信息

  $ gzip -l *.gz

- 解压filename.gz文件

  $ gzip -d filename.gz

  $ gunzip filename.gz

梁如军（linuxbooks@126.com）

bzip2

- Linux下常用的压缩和解压缩命令。
- 由官方仓库的bzip2软件包提供。
- 比gzip的压缩比更高。
- 压缩后 bzip2 会在每个文件的后面添加扩展名 .bz2。
- 压缩后原文件会被自动删除。
- 在 windows 下可以用 winrar或7-zip 解压。
- bzip2命令的格式和参数与gzip类似。

bzip2命令举例

- 压缩文件filename

  $ bzip2 filename

- 高度压缩文件 file1和file2并显示执行过程

  $ bzip2 -9v file1 file2

- 解压filename.bz2文件

  $ bzip2 -d filename.bz2

  $ bunzip2 filename.bz2

梁如军（linuxbooks@126.com）

zip/unzip

CentOS
The Community ENTerprise Operating System

- 与windows下的 winzip兼容
- 由官方仓库的zip/unzip软件包提供
- 例如：
  - 压缩文件 file1为 fiel1.zip，原文件保留

```
$ zip file1.zip  file1
```

  - 将子目录 data1/ 下的所有文件压缩到文件 data1.zip

```
$ zip –r data1.zip  data1
```

  - 解压释放压缩文件 data1.zip 中的所有文件

```
$ unzip data1.zip
```


梁如军（linuxbooks@126.com）

rar和7za

- rar
  - 由RPMForge仓库的rar软件包提供
- 7za
  - http://www.7-zip.org
  - http://p7zip.sourceforge.net/
  - 由EPEL仓库的p7zip软件包提供

tar

- 基本功能：打包和解包
- 格式： tar [选项] 文件或者目录
- 常用选项
  - -c：创建新的打包文件。
  - -t：列出打包文件的内容，查看已经打包了哪些文件。
  - -x：从打包文件中释放文件。
  - -f：指定打包文件名。
  - -v：详细列出 tar 处理的文件信息。
  - -z：用 gzip 来压缩/解压缩打包文件。
  - -j：用 bzip2 来压缩/解压缩打包文件。
  - -J：用 xz 来压缩/解压缩打包文件。

tar命令举例

CentOS
The Community ENTerprise Operating System

```
$ tar -cvf myball.tar somedirname
$ tar -tf myball.tar
$ tar -xvf myball.tar
$ tar -zcvf myball.tar.gz somedirname
$ tar -ztf myball.tar.gz
$ tar -zxvf myball.tar.gz
$ tar -jcvf myball.tar.bz2 somedirname
$ tar -jtf myball.tar.bz2
$ tar -jxvf myball.tar.bz2
```

注意： “-f 文件名|设备名”是一个整体， 所以 -cvf myball.tar
不能写成：-cfv myball.tar 或 -fcv myball.tar

梁如军（linuxbooks@126.com）

在BASH中提高工作效率

梁如军（linuxbooks@126.com）

命令补全

CentOS
The Community ENTerprise Operating System

- 通常用户在 bash 下输入命令时不必把命令输全， shell 就能判断出你所要输入的命令。
- 该功能的核心思想是：bash 根据用户已输入的信息来查找以这些信息开头的命令，从而试图完成当前命令的输入工作。用来执行这项功能的键是 **Tab 键**，按下一次 Tab 键后，bash 就试图完成整个命令的输入，如果不成功，可以再按一次 Tab 键，这时 bash 将列出所有能够与当前输入字符相匹配的命令列表。

梁如军（linuxbooks@126.com）

命令补全举例

- 执行system-config-network-tui

  system<Tab>-config-n<Tab>etwork-t<Tab>ui
- 进入/etc/sysconfig/network-scripts/目录

  cd /e<Tab>sys<Tab>c<Tab>ne<Tab>-<Tab>
- 显示$BASH变量的值

  echo $B<Tab>ASH

命令历史

- bash可以记录一定数目的以前在Shell中输入的命令。
  - 记录历史命令的文本文件由环境变量 HISTFILE 来指定，默认的记录文件是.bash_history，这是一个隐含文件，位于用户自己的目录中。
  - 可以记录历史命令的数目由环境变量 HISTSIZE 的值指定，默认为1000。
- 查看命令历史
  - **history**
  - **history 30** # 查看最近 30 个历史命令
  - **fc -l 30 50** # 列出命令历史中第30到第50之间的命令

命令历史（续）

- 键盘快捷键
  - 最简单的方法是用上下方向键、<PgUp>和<PgDn>键来查看历史命令
  - 如果需要的话，可以使用键盘上的编辑功能键对显示在命令行上的命令进行编辑
- 感叹号的用法
  - 用 !! 执行最近执行过的命令
  - 用 !<命令事件号> 执行已经运行过的命令
  - 用 !<已经使用过的命令前面的部分> 执行已经运行过的以该字符串开头的最近的命令

命令别名

- 允许用户按照自己喜欢的方式对命令进行自定义
- 格式
  - alias [alias_name='original_command']
- 说明
  - alias_name是用户给命令取的别名。
  - original_command是原来的命令和参数。若命令中包含空格或其他的特殊字符串必须使用引号。
  - 在定义别名时，**等号两边不允许有空格**。
  - 不带任何参数的alias命令显示当前已定义的所有别名。
  - 可以使用 **unalias** alias_name 命令取消某个别名的定义。
  - 如果用户需要别名的定义在每次登录时均有效，应该将其写入用户自家目录下的.bashrc文件中。

命令别名（续）

- 定义别名举例

  alias lh='ls -lh'

  alias grep='grep --color=auto'

  alias gitcam='git commit -a -m '

- 注意
  - 若系统中有一个命令，同时又定义了一个与之同名的别名（例如，系统中有grep命令，且又定义了grep的别名），则别名将优先于系统中原有的命令的执行。
  - 要想临时使用系统中的命令而非别名，应该在命令前添加“\”字符，例如，$ \grep命令将运行系统中原来的grep命令而不是grep别名，它不在输出中显示颜色。

梁如军（linuxbooks@126.com）

正则表达式

梁如军（linuxbooks@126.com）

正则表达式

- 正则表达式是使用某种模式（pattern）去匹配（matching）一类字符串的一个公式。
- 通常使用正则表达式进行查找、替换等操作。
- 在适当的情况下使用正则表达式可以极大地提高工作效率。
- 有两种风格的正则表达式：
  - POSIX 风格的正则表达式
  - Perl 风格的正则表达式（Perl-compatible regular expression）

支持 RE 的文本处理工具

- 基本的正则表达式 Basic regular expression (BRE)
  - grep 按**模式匹配**文本
  - ed 一个原始的行编辑器
  - sed 一个**流编辑器**
  - vim 一个屏幕编辑器
  - emacs 一个屏幕编辑器
- 扩展的正则表达式 Extended regular expression (ERE)
  - egrep 按模式匹配文本
  - **awk** 进行简单的文本处理

梁如军（linuxbooks@126.com）

正则表达式的组成

CentOS
The Community ENTerprise Operating System

- 正则表达式由一些普通字符和一些元字符（metacharacters）组成。
  - 普通字符包括大小写的字母、数字（即所有非元字符）
  - 元字符则具有特殊的含义

梁如军（linuxbooks@126.com）

正则表达式的元字符

元字符	含义	类型	举例	说明
^	匹配首字符	BRE	^x	以字符x开始的字符串
$	匹配尾字符	BRE	x$	以x字符结尾的字符串
.	匹配任意一个字符	BRE	l..e	love, life, live …
?	匹配任意一个可选字符	ERE	xy?	x, xy
*	匹配零次或多次重复	BRE	xy*	x, xy, xyy, xyyy …
+	匹配一次或多次重复	ERE	xy+	xy, xyy, xyyy …
[…]	匹配任意一个字符	BRE	[xyz]	x, y, z
()	对正则表达式分组	ERE	(xy)+	xy, xyxy, xyxyxy, …

梁如军（linuxbooks@126.com）

正则表达式的元字符（续）

元字符	含义	类型	举例	说明
\{n\}	匹配n次	BRE	go\{2\}gle	google
\{n,\}	匹配最少n次	BRE	go\{2,\}gle	google, gooogle, goooogle …
\{n,m\}	匹配n到m次	BRE	go\{2,4\}gle	google, gooogle, goooogle
{n}	匹配n次	ERE	go{2}gle	google
{n,}	匹配最少n次	ERE	go{2,}gle	google, gooogle, goooogle …
{n,m}	匹配n到m次	ERE	go{2,4}gle	google, gooogle, goooogle
|	以或逻辑连接多个匹配	ERE	good|bon	匹配 good 或 bon
\	转义字符	BRE	*	*

梁如军（linuxbooks@126.com）

正则表达式的元字符（续2）

- POSIX RE 用于**方括号之内**的元字符

元字符	含义	类型	举例	说明
^	非（仅用于启始字符）	BRE	[^xyz]	匹配xyz之外的任意一个字符
-	用于指明字符范围（不能是首字符和尾字符）	BRE	[a-zA-Z]	匹配任意一个字母
\	转义字符	BRE	[\.]	.

梁如军（linuxbooks@126.com）

正则表达式举例 1

- ...x..x..x
- ^d
- ^the
- sh$
- ^....$
- ^$
- \.
- ^.2
- *\.pas
- t.*\.sh$
- t*\.sh$
- .a.*

- []中都是单个字符匹配
  - [0123456789]
  - [0-9]
  - [a-zA-Z0-9\-]
  - [^0-9]
  - [^abc]
  - ^[^1]
  - [Gg]reen
  - [a-z][a-z]*
  - ^\.[0-9][0-9]

正则表达式举例 2

- gr(a|e)y
- (^To:|^From:) (Seaman|Ramsay)
- [0-9]\{2\}-[0-9]\{2\}-[0-9]\{4\}
- (yellow|red) flower(s)?

- /etc/rc\.d/init\.d/httpd
- /usr/sbin/httpd(\.worker)?
- /var/www(/.*)?/logs(/.*)?
- /var/log/apache(2)?(/.*)?
- /var/www/[^/]*/cgi-bin(/.*)?
- /srv/([^/]*/)?www(/.*)?
- /usr/lib(64)?/httpd(/.*)?

梁如军（linuxbooks@126.com）

CentOS
The Community ENTerprise Operating System

文本文件操作命令

梁如军（linuxbooks@126.com）

常用的文本文件提取命令

命令	功能
cat、tac	滚屏显示文本文件内容
more、less	分屏显示文本文件内容
head、tail	显示文本文件的前或后若干行 （**横向截取**文本文件内容）
cut	**纵向切割**出文本指定的部分 （纵向截取文本文件内容）
grep	在文本文件中**查找指定的字符串** （按关键字提取文本文件中匹配的行）

梁如军（linuxbooks@126.com）

文本显示命令举例

命令	举例
cat /etc/passwd	滚屏显示文件/etc/passwd的内容
cat -n /etc/passwd	滚屏显示文件/etc/passwd的内容，并显示行号
more /etc/passwd	分屏显示文件/etc/passwd的内容（注意空格键、回车键和q的使用）
more +10 /etc/passwd	从第10行分屏显示文件/etc/passwd的内容
less /etc/passwd	分屏显示文件/etc/passwd的内容（注意空格键、回车键、PgDn键、PgUp键和q的使用）
head -4 myalllist	显示文件myalllist前4行的内容
tail -4 myalllist	显示文件myalllist后4行的内容
tail +10 myalllist	显示文件myalllist从10行开始到文件尾的内容
tail -f /var/log/messages	跟踪显示不断增长的文件结尾内容（通常用于显示日志文件）

grep 简介

- grep（global search regular expression）是一个强大的文本搜索工具。grep 使用正则表达式搜索文本，并把匹配的行打印出来。
- UNIX 的 grep 家族包括 grep、egrep 和 fgrep：
  - grep 使用 Basic regular expression (BRE) 书写匹配模式，等效于 grep -G
  - egrep 使用 Extended regular expression (ERE) 书写匹配模式，等效于 grep -E
  - fgrep 不使用任何正则表达式书写匹配模式（以固定字符串对待），执行快速搜索，等效于 grep -F

grep 命令

- 格式
  - grep [options] PATTERN [FILE...]
- 说明
  - PATTERN 是查找条件
    - 可以是**普通字符串**
    - 可以是**正则表达式**，通常用单引号将RE**括起来**。
  - FILE 是要查找的文件，可以是用空格间隔的多个文件，也可是使用Shell的通配符在多个文件中查找PATTERN，省略时表示在标准输入中查找。
  - grep命令不会对输入文件进行任何修改或影响，可以使用**输出重定向**将结果存为文件。

grep命令选项

选项	说明
-c	只显示匹配行的次数
-i	搜索时不区分大小写
-n	输出匹配行的行号
-v	输出不匹配的行（反向选择）
-r	对目录（子目录）的所有文件递归地进行
-l	列出匹配PATTERN的文件名
--color=auto	对匹配内容高亮显示
-A NUM	同时输出匹配行的后 NUM 行
-B NUM	同时输出匹配行的前 NUM 行
-C NUM	同时输出匹配行的前、后各 NUM 行

梁如军（linuxbooks@126.com）

grep命令举例

- 在文件 myfile 中查找包含字符串 mystr的行

  $ grep mystr myfile

- 显示 myfile 中第一个字符为字母的所有行

  $ grep '^[a-zA-Z]' myfile

- 在文件 myfile 中查找首字符不是 # 的行（即过滤掉注释行）

  $ grep -v '^#' myfile

- 过滤掉/etc/samba/smb.conf的注释行和空行

  $ egrep -v '^#|^$|^;' /etc/samba/smb.conf

grep命令举例续

- 列出/etc目录（包括子目录）下所有文件内容中包含字符串“root”的文件名

  # grep -lr root /etc/*

- 在文件 myfile 中查找包含字符 $（在RE中具有特殊含义） 的行

  $ grep \\$ myfile

  $ grep '\$' myfile

  $ fgrep '$' myfile

  $ fgrep $ myfile

梁如军（linuxbooks@126.com）

常用的文本文件分析命令

CentOS
The Community ENTerprise Operating System

命令	功能
wc	统计文本
sort	以行为单位对文本文件排序
uniq	删除文本文件中连续的重复的行
diff	比较两个文本文件的差异
diff3	比较三个文本文件的差异
patch	为文本文件打补丁
aspell	为文本文件做拼写检查（西文）

梁如军（linuxbooks@126.com）

wc命令

- 功能：统计文本文件的行数、字数、字符数
- 格式：wc [选项] [<文件> …]
- 举例

```
$ wc file
$ wc -l file          # 统计行数
$ wc -w file          # 统计字数
$ wc -c file          # 统计字符数
$ wc -L file          # 统计最长一行的长度
```

sort命令

- 功能：以行为单位对文件进行排序
- 格式：sort [选项] [<文件> …]
- 选项

-r	逆向排序
-f	忽略字母的大小写
-n	根据字符串的数值进行排序
-u	对相同的行只输出一行
-t c	选项使用c做为列的间隔符
-b	忽略前导的空格
-i	只考虑可打印字符
-k N	以第N列进行排序（默认以空格或制表符作为列的间隔符）

梁如军（linuxbooks@126.com）

sort命令举例

CentOS
The Community ENTerprise Operating System

```
$ sort file
$ sort -n file
$ sort -fr file
$ sort -u file
$ sort file1 file2
$ sort -br file1 file2
$ sort -n -k 3 -t ':'  /etc/passwd
```


梁如军（linuxbooks@126.com）

常用的文本文件处理命令

命令	功能
tr	字符替换
sed	流编辑器，常用于**字符串替换**
paste	纵向合并多个文本
expand	将文件中的制表符转换为空格
unexpand	将文件中的空格转换为制表符
dos2unix	将DOS格式的文本转换成UNIX格式
unix2dos	将UNIX格式的文本转换成DOS格式
iconv	将文本从一种编码转换成另一种编码

梁如军（linuxbooks@126.com）

sed 简介

- sed 是一个**流编辑器**（stream editor）。sed 是一个非交互式的行编辑器，它在命令行中输入编辑命令、指定被处理的输入文件，然后在屏幕上查看输出。输入文件可以是指定的文件名，也可以来自一个管道的 输出。
- 与 vi 不同的是 sed 能够过滤来自管道的输入。在 sed 编辑器运行的时候不必人工干涉，所以 sed 常常被称作**批编辑器** 。
- sed 默认不改变输入文件的内容，且总是将处理结果输出到标准输出，可以使用输出重定向将 sed 的输出保存到文件中。

梁如军（linuxbooks@126.com）

sed 的工作方式

- sed 以**按顺序逐行的方式工作**，过程为：
  - 从输入读取一行数据存入临时缓冲区，此缓冲区称为**模式空间**（pattern space）
  - 按指定的 sed 编辑命令处理缓冲区中的内容
  - 把模式空间的内容送往屏幕并将这行内容从模式空间中删除
  - 读取下面一行。重复上面的过程直到全部处理结束。

梁如军（linuxbooks@126.com）

sed命令

- 格式

  sed [选项] [-e] cmd1 [[-e cmd2] ... [-e cmdn]] [input-file]...

- 说明
  - 在命令行上执行sed编辑命令。可以指定**多个编辑命令**，每个编辑命令前都要**使用 -e 参数**，sed 将对这些编辑命令依次进行处理。若只有一个编辑命令时，-e 可以省略。
  - 每个sed的编辑命令cmdX均应使用单引号括起来。
  - input-file：sed 处理的文件列表，若省略，sed 将从标准输入中读取输入，也可以从输入重定向或管道获得输入。
- 选项
  - -r：使用**扩展正则表达式**进行模式匹配
  - -i：直接对输入文件进行sed的命令操作

梁如军（linuxbooks@126.com）

sed命令举例

sed -i 's/Windows/Linux/g‘ myfile	将myfile中的所有Windows替换成Linux
sed 's/Windows/Linux/g' myfile	功能同上，但不对输入文件本身进行替换，仅在屏幕输出结果
sed 's/cc*/c/g' myfile	将myfile中所有连续出现的c都压缩成单个的c，仅在屏幕输出结果
sed 's/^[\t]*//' myfile	删除myfile中每一行前导的连续“空白字符”（空格，制表符），仅在屏幕输出
sed 's/ *$//' myfile	删除myfile中每行结尾的所有空格，仅在屏幕输出
sed 's/^$//' myfile	删除所有空白行，仅在屏幕输出
sed 's/^/> /' myfile	在每一行开头加上一个尖括号和空格（引用信息），仅在屏幕输出
sed 's/.*\///' myfile	删除路径前缀，仅在屏幕输出

梁如军（linuxbooks@126.com）

iconv命令

- 功能：将文件**从一种编码转换成另一种编码**
- 格式：iconv [选项] <输入文件>
- 选项
  - -f <encoding>：指定原始文本编码。
  - -t <encoding>：指定要转换的编码。
  - -o <output file>：指定输出文件，而不是在标准输出上显示。
  - -l：列出所有已知编码字符集。

梁如军（linuxbooks@126.com）

iconv命令举例

- 将编码为GB2312的inputfile文件转化为UTF-8编码的outputfile

  $ iconv -f GB2312 -t UTF-8 -o outputfile inputfile

- 又如

  $ iconv -l

  $ iconv -f ISO-8859-1 -t UTF-8 -o outputfile inputfile

  $ iconv -f GBK -t UTF-8 -o outputfile inputfile

  $ iconv -f BIG5 -t UTF-8 -o outputfile inputfile

  $ iconv -f UTF-8 -t GB2312 -o outputfile inputfile

梁如军（linuxbooks@126.com）

VIM文本编辑器

梁如军（linuxbooks@126.com）

常用的文本编辑器

- vim
- nano
- emacs
- gedit

梁如军（linuxbooks@126.com）

vi简介

- vi 是 “**Visual interface**” 的简称，它可以执行输出、删除、查找、替换、块操作等众多文本操作，而且用户可以根据自己的需要对其进行定制，这是其他编辑程序所没有的。
- vi 不是一个排版程序，它不像 M$ Word 或 WPS 那样可以对字体、格式、段落等其他属性进行编排，它只是一个**文本编辑程序**。
- vi 是全屏幕文本编辑器，它没有菜单，只有命令。
- **vim** 即 **Vi IMproved**，**vi** 克隆版本之一。

梁如军（linuxbooks@126.com）

vi 的进入

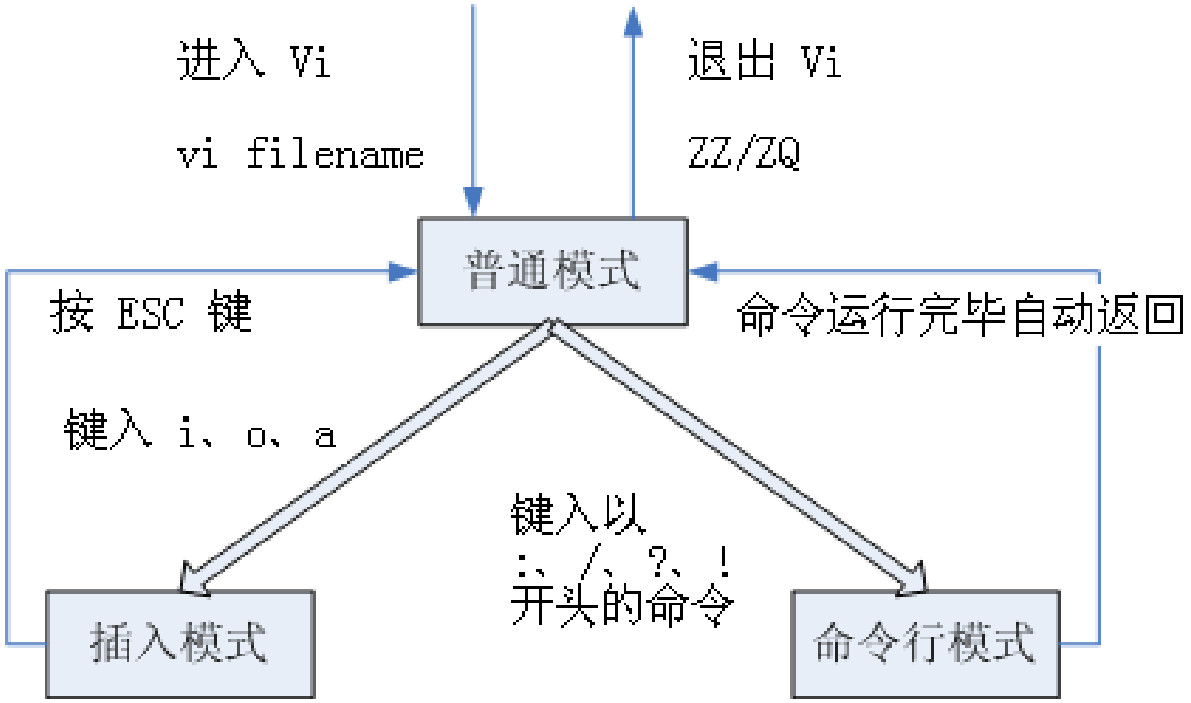
CentOS
The Community ENTerprise Operating System

命令	说明
vi	直接进入
vi filename	打开或新建文件filename，并将光标置于第一行首
vi +n filename	打开文件filename，并将光标置于第n行首
vi + filename	打开文件filename，并将光标置于最后一行首
vi +/pattern filename	打开文件filename，并将光标置于第一个与pattern匹配的串处
vi -r filename	打开上次用vi编辑时发生系统崩溃，恢复filename

梁如军（linuxbooks@126.com）

Vi 的3种运行模式

- 普通(normal)模式
- 插入(insert)模式
- 命令行(cmdline)模式



梁如军（linuxbooks@126.com）

Vi 的 Normal 模式

- 在shell中输入 vim 启动编辑器时，即进入该模式。
- 无论什么时候，不管用户处于何种模式，只要按一下 Esc Esc 键，即可使 vim 进入 Normal 模式。
- 在该模式下，用户可以输入各种合法的 vim 命令，用于管理自己的文档。此时从键盘上输入的任何字符都被当做编辑命令来解释。
- 若输入的字符是合法的 vim 命令，则 vim 在接受用户命令之后完成相应的动作。但需注意的是，所输入的命令并不在屏幕上显示出来。若输入的字符不是 vim 的合法命令，vim 会响铃报警。

Vi 的 Insert 模式

- 在 **Normal** 模式下输入插入命令 **i**、附加命令 **a** 、打开命令 **o**、修改命令 **c**、取代命令 **r** 或替换命令 **s** 等都可以进入 Insert 模式。
- 在该模式下，用户输入的任何字符都被vim当做文件内容保存起来，并将其显示在屏幕上。在文本输入过程中，若想回到Normal模式下，按 **Esc** 键即可。

Vi的 Command 模式

- Normal 模式下，用户按冒号 “:” 即可进入 Command 模式，此时 vim 会在显示窗口的最后一行 (屏幕的最后一行) 显示一个 “:” 作为 Command 模式的提示符，等待输入命令。
- 多数文件管理都是在此模式下执行的 (如保存文件等)
- Command 模式中所有的命令都必须按 <回车>后执行，命令执行完后，vim 自动回到 Normal 模式。
- 若在 Command 模式下输入命令过程中改变了主意，可按 Esc键，或用退格键将输入的命令全部删除之后，再按一下退格键，即可使 vi 回到 Normal 模式下。

Normal模式下的基本操作

- G 用于直接跳转到文件尾
- x 删除光标所在的字符
- r 替换光标所在的字符
- ~ 切换光标所在字母的大小写
- /和？ 用于查找字符串
- dd、YY、p分别用于剪切、复制和粘贴一行文本
- u 取消上一次编辑操作（undo）
- . 重复上一次编辑操作（redo）
- ZZ 用于存盘退出Vi
- ZQ用于不存盘退出Vi

Command 模式下的基本操作

- :n1,n2 co n3 用于块复制
- :n1,n2 m n3 用于块移动
- :n1,n2 d 用于块删除
- :w 保存当前编辑文件，但并不退出
- :w newfile 存为另外一个名为“newfile”的文件
- :wq 用于存盘退出Vi
- :q! 用于不存盘退出Vi
- :q 用于直接退出Vi （未做修改）

梁如军（linuxbooks@126.com）

Command 模式
——设置 Vi 环境

- :set autoindent 缩进,常用于程序的编写
- :set noautoindent 取消缩进
- :set number 在编辑文件时显示行号
- :set nonumber 不显示行号
- :set tabstop=value 设置显示制表符的空格字符个数
- :set 显示设置的所有选项
- :set all 设置的选显示所有可以项

梁如军（linuxbooks@126.com）

信息显示命令

梁如军（linuxbooks@126.com）

常用的系统信息显示命令

CentOS
The Community ENTerprise Operating System

命令	功能
hostname	显示主机名称
uname	显示操作系统信息
dmesg	显示系统启动信息
lsmod	显示系统加载的内核模块
date	显示系统时间（cal 可以显示系统时间的日历）
env	显示系统环境变量
locale	显示当前语言环境（cat /etc/sysconfig/i18n）
cat /etc/redhat-release	显示操作系统版本（head -1 /etc/issue）
cat /proc/cpuinfo	显示CPU信息
lspci/lsusb	显示PCI/USB接口信息
rpm -qa	显示系统已安装的所有软件包

梁如军（linuxbooks@126.com）

常用的资源显示命令

命令	功能
top	显示当前系统中耗费资源最多的进程
free	显示当前内存的使用情况（cat /proc/meminfo）
du -h	显示指定的文件（目录）已使用的磁盘空间的总量
df -h	显示文件系统磁盘空间的使用情况
uptime	显示系统运行时间、用户数、负载
fdisk -l	查看所有分区
mount	查看已经挂装的分区
swapon -s	查看所有交换分区
ps -ef	查看所有进程
pstree	显示进程树
chkconfig --list	列出所有系统服务

梁如军（linuxbooks@126.com）

常用的用户相关显示命令

命令	功能
who、w	显示在线登录用户
whoami	显示用户自己的身份
tty	显示用户当前使用的终端
id	显示当前用户的id信息
groups	显示当前用户属于哪些组
last	查看用户登录日志
crontab -l	查看当前用户的计划任务

梁如军（linuxbooks@126.com）

常用的网络信息显示命令

命令	功能
ifconfig	显示网络接口信息
route	显示系统路由表
iptables -nL	显示包过滤防火墙的规则设置
netstat	显示网络状态信息
cat /etc/resolv.conf	显示DNS配置
cat /etc/hosts	显示静态主机解析表

梁如军（linuxbooks@126.com）

进一步使用 SHELL

梁如军（linuxbooks@126.com）

标准输入/输出设备

- Linux命令在执行时常常期望接收输入数据，命令执行后又期望将产生的数据结果输出。
- Linux的大部分命令都具有标准的输入/输出设备端口。

名称	文件描述符	含义	设备	说明
STDIN	0	标准输入	键盘	命令在执行时所要的输入通过它来取得
STDOUT	1	标准输出	显示器	命令执行后的输出结果从该端口送出
STDERR	2	标准错误	显示器	命令执行时的错误信息通过该端口送出

标准输入/输出举例

- 标准输入和标准输出

```
[root@soho ~]# cat
hello
hello
^D
```

命令等待标准输入

标准输入的屏幕回显

标准输出

- 标准错误输出

```
[root@soho ~]# cat x
cat: x: 没有那个文件或目录
```

标准错误输出

梁如军（linuxbooks@126.com）

重定向（Redirection）

- 所谓重定向，就是不使用系统的标准输入端口、标准输出端口或标准错误端口，而**进行重新的指定**，所以重定向分为输出重定向、输入重定向和错误重定向。通常情况下重定向到一个文件。
- 在Shell中，要实现重定向主要**依靠重定向符实现**，即Shell是检查命令行中有无重定向符来决定是否需要实施重定向。

梁如军（linuxbooks@126.com）

重定向符

重定向符	说明
<	输入重定向
<<! …… !	输入重定向的特例，即HERE文件，通常用于Shell脚本中。其中“!”可以使用任何字符或字符串替换，只要其没在……中出现过即可。
>	覆盖式的输出重定向
>>	追加式的输出重定向
2>	覆盖式的错误输出重定向
2>>	追加式的错误输出重定向
&>	同时实现输出重定向和错误重定向（覆盖式）

梁如军（linuxbooks@126.com）

重定向举例

CentOS
The Community ENTerprise Operating System

```
$ ls -l /tmp >mydir
$ ls -l /etc >>mydir

$ myprogram 2> err_file
$ myprogram &> output_and_err_file

$ find  ~  -name *.mp3 > ~/cd.play.list
$ echo “Please call me : 68800000”>message

$ cat <<! >mytext
> This text forms the content of the heredocument ,
> which continues until the end of text delimiter
> !
```


梁如军（linuxbooks@126.com）

输出重定向与空设备

- 空设备（/dev/null）
  - 空设备是个黑洞，发往它的任何内容都将不复存在
  - 经常用于屏蔽命令的输出或错误输出，尤其用于Shell脚本中
- 空设备使用举例
  - 屏蔽命令的输出和错误输出

    $ myprogram &> /dev/null

    $ myprogram >/dev/null 2>&1
  - 清空文件内容

    $ cp /dev/null myfile

    $ > myfile

管道的引入

- UNIX 系统的一个**基本哲学**是：**一连串的小命令能够解决大问题**。其中每个小命令都能够很好地完成一项单一的工作。现在需要有一些东西能够将这些简单的命令**连接起来**，这样**管道**就应运而生。
- 许多Linux命令具有过滤特性，即一条命令通过标准输入端口接受一个文件中的数据，命令执行后产生的结果数据又通过标准输出端口送给后一条命令，作为该命令的输入数据。后一条命令也是通过标准输入端口而接受输入数据。

梁如军（linuxbooks@126.com）

管道（Pipe）

- 管道（使用符号“|”表示）用来连接命令
  - 命令1 | 命令2
  - 将命令1的STDOUT发送给命令2的STDIN
  - STDERR不能通过管道转发
- 用来组合多种工具的功能
  - **命令1 | 命令2 | 命令3 |……**
    - ls -C | tr 'a-z' 'A-Z‘ | wc
  - 管道线中的每一条命令都作为一个单独的进程运行，每一条命令的输出作为下一条命令的输入。
  - 由于管道线中的命令总是从左到右顺序执行的，因此管道线是单向的。

梁如军（linuxbooks@126.com）

管道应用举例（1）

CentOS
The Community ENTerprise Operating System

```
$ ls -lR /etc | less
$ tail +15 myfile | head -3
$ man bash | col -b > bash.txt
# echo "p4ssW0rd" | passwd --stdin user1

$ ls -l | grep "^d"
$ cat /etc/passwd | grep username
$ dmesg | grep eth0
$ rpm –qa | grep httpd

$ echo "test email" | mail -s "test" user@example.com
$ echo "test print" | lpr
```


梁如军（linuxbooks@126.com）

管道应用举例（2）

- 统计磁盘占用情况
  - 统计当前目录下磁盘占用最多的10个一级子目录

$ du . --max-depth=1 | sort -rn | head -11

  - 以降序方式显示使用磁盘空间最多的普通用户的前十名

$ du * -cks | sort -rn | head -11

  - 以排序方式查看当前目录（不包含子目录）的磁盘占据情况。

$ du -S | sort -rn | head -11

梁如军（linuxbooks@126.com）

管道应用举例（3）

- 统计进程
  - 按内存使用从大到小排列输出进程。

```
# ps -e -o "%C : %p : %z : %a"|sort -k5 -nr
```

  - 按CPU使用从大到小排列输出进程。

```
# ps -e -o "%C : %p : %z : %a"|sort -nr
```


梁如军（linuxbooks@126.com）

管道应用举例（4）

- 列出**YUM仓库**中所有可用的 Apache 模块并按升序输出

```
# yum list | grep ^mod_ | cut -d'.' -f 1 | sort
# yum list | grep ^mod_ | awk -F\. '{print $1}' | sort
```

- 以排序方式列出YUM仓库中在 /etc/httpd/conf.d/ 目录下生成配置文件的所有 Web 应用软件包（不包含 Apache 模块）

```
# repoquery --queryformat="%{NAME}\n"  \
  --whatprovides “/etc/httpd/conf.d/*” |  \
 egrep -v  "(^$|^mod)" | sort | uniq
```


梁如军（linuxbooks@126.com）

管道应用举例（5）

- 从ifconfig 命令的输出过滤出 eth0 网络接口当前的IPv4地址

```
# ifconfig eth0 | awk -F\: '/inet / {print $2}'|awk '{print $1} '

# ifconfig eth0 | grep 'inet ' | awk -F '[ :]+' ' {print $4}'

# ifconfig eth0 | grep -i 'inet[^6]' | sed 's/[a-zA-Z:]//g' | awk '{print $1}‘
```

- 从ip命令的输出过滤出 eno16777736网络接口当前的IPv4地址

```
# ip a s eno16777736|grep 'inet '| awk -F '[ /]+' '{print $3}'
```


梁如军（linuxbooks@126.com）

T型管道（tee）

- 格式
  - 命令1 | tee　文件名 | 命令2
- 功能
  - 将命令1的STDOUT保存在文件名中，然后管道输入给命令2
- 用于
  - 保存不同阶段的输出
  - 复杂管道的故障排除
  - 同时查看和记录输出

梁如军（linuxbooks@126.com）

命令替换（Command Substitution）

- 使用**命令的输出**，常用于
  - 在文本中嵌入命令的执行结果
  - 命令参数是另一个命令执行的结果
- 使用方法

  $(command)　　或　　`command`

  cmd1 $(cmd2)　　或　　cmd1 `cmd2`
- 使用举例

```
$ echo The present time is `date`
$ rpm -qi $(rpm -qf $(which date))      # 嵌套
```


梁如军（linuxbooks@126.com）

命令组合

命令行形式	说明	举例
CMD1 ; CMD2	顺序执行若干命令	pwd;date;ls
CMD1 && CMD2	当CMD1运行成功时才运行CMD2	gzip mylargefile && echo “OK.”
CMD1 || CMD2	当CMD1运行失败时才运行CMD2	write osmond || mail -s test osmond < my.log
(CMDLIST)	在子Shell中执行命令序列	(date; who | wc -l) > ~/login-users.log
{CMDLIST}	在当前Shell中执行命令序列	{ cd /home/jjh; chown jjh:bin s* ;}

SHELL变量 和 SHELL环境

梁如军（linuxbooks@126.com）

Shell 变量

- Shell 变量大致可以分为三类
  - **内部变量**：由系统提供，用户只能使用不能修改。
  - **用户变量**：由用户建立和修改，在 shell 脚本编写中会经常用到。
  - **环境变量**：这些变量决定了用户工作的环境，它们不需要用户去定义，可以直接在 shell 中使用，其中某些变量用户可以修改。

梁如军（linuxbooks@126.com）

用户自定义变量

- 变量赋值（定义变量）
  - varName=Value
  - export varName=Value
- 引用变量 $varName

  - 一般地，所有的Shell变量都是字符串。
  - 当变量的值仅仅包含数字时才允许进行数值计算。
  - 在较新的 bash 中，可是使用 declare 或 typeset 命令声明变量及其属性，但一般不需要声明。而且为了使脚本兼容于不同的 shell，在没有必要的情况下尽量不使用变量声明。

引用

- 在 bash 中，有些字符具有特殊含义，如果需要忽略这些字符的特殊含义，就必须使用引用技术。
- 引用可以通过下面三种方式实现
  - 使用转义字符：\
  - 使用单引号：‘ ’
  - 使用双引号：“ ”
- 转义字符的引用方法就是直接在字符前加反斜杠。例：\$，\‘，\“，\\，\ ，\!

强引用和弱引用

- 强引用
  - 单引号对是强引用
  - 单引号对中的字符都将作为普通字符，但不允许出现另外的单引号。
- 弱引用
  - 双引号对是弱引用
  - 双引号对中的部分字符仍保留特殊含义
    - $（美元符号）— 变量扩展
    - `（反引号） — 命令替换
    - \（反斜线） — 禁止单个字符扩展
    - !（叹号） — 历史命令替换

梁如军（linuxbooks@126.com）

命令行执行过程

CentOS
The Community ENTerprise Operating System

1. 将命令行分成单个命令词
2. 展开别名
3. 展开大括号中的声明（{}）
4. 展开颚化声明（~）
5. 命令替换（$() 或 ``）
6. 再次把命令行分成命令词
7. 展开文件通配（*、?、[abc]等等）
8. 准备I/O重定向（<、>）
9. 运行命令！

梁如军（linuxbooks@126.com）

Shell 变量的作用域

CentOS
The Community ENTerprise Operating System

- 局部变量的作用范围仅仅限制在其命令行所在的Shell或Shell脚本文件中；
- 全局变量的作用范围则包括本Shell进程及其所有子进程。
- 可以使用 export 内置命令将局部变量设置为全局变量。
- 可以使用 export 内置命令将全局变量设置为局部变量。

梁如军（linuxbooks@126.com）

export 命令

- 显示当前Shell可见的全局变量
  - export [-p]
- 定义变量值的同时声明为全局变量
  - export <变量名1=值1> [<变量名2=值2> ...]
- 声明已经赋值的某个（些）局部变量为全局变量
  - export <变量名1> [<变量名2> ...]
- 声明已经赋值的某个（些）全局变量为局部变量
  - export -n <变量名1> [<变量名2> ...]

梁如军（linuxbooks@126.com）

Shell环境变量

- 环境变量定义 Shell 的运行环境，保证 Shell 命令的正确执行。
- Shell用环境变量来确定查找路径、注册目录、终端类型、终端名称、用户名等。
- 所有环境变量都是**全局变量**（即可以传递给 Shell 的子进程），并可以由用户重新设置。

梁如军（linuxbooks@126.com）

常见的 Shell 环境变量

变量名	含义
HOME	用户主目录
LOGNAME	登录名
USER	用户名，与登录名相同
PWD	当前目录/工作目录名
MAIL	用户的邮箱路径名
HOSTNAME	计算机的主机名
INPUTRC	默认的键盘映像
SHELL	用户所使用的 shell 的路径名
LANG	默认语言
HISTSIZE	history 所能记住的命令的最多个数
PATH	shell 查找用户输入命令的路径 (目录列表)
PS1、PS2	shell 一级、二级命令提示符

梁如军（linuxbooks@126.com）

Shell变量的查询、显示和取消

- 显示当前已经定义的所有变量
  - 所有环境变量：env
  - 所有变量和函数（包括环境变量）：set
- 显示某（些）个变量的值
  - echo $NAME1 [$NAME2 ……]
- 取消变量的声明或赋值
  - unset <NAME>

用户工作环境

- 用户登录系统时，Shell为用户自动定义唯一的工作环境并对该环境进行维护直至用户注销。
  - 该环境将定义如身份、工作场所和正在运行的进程等特性。这些特性由指定的环境变量值定义。
- 用户工作环境有**登录环境**和**非登录环境**之分。
  - **登录环境**是指用户登录系统时的工作环境，此时的Shell对登录用户而言是主Shell。
  - **非登录环境**是指用户再调用子Shell时所使用的用户环境。

设置用户工作环境

- 对所有用户进行设置
  - /etc/profile
  - /etc/bashrc
- 只对当前用户进行设置
  - ~/.bash_profile
  - ~/.bashrc

通常，个人bash 环境设置都定义在 ~/.bashrc 文件里

梁如军（linuxbooks@126.com）

登录 shell 和非登录 shell 的启动过程

- Login shell

  /etc/profile → /etc/profile.d/*.sh

  $HOME/.bash_profile

  $HOME/.bashrc → /etc/bashrc

- Non-Login shell

  $HOME/.bashrc → /etc/bashrc

梁如军（linuxbooks@126.com）

本章思考题

- 什么是Shell？它具有什么功能？Linux默认使用什么Shell？
- 简述文件的类型。硬链接和软链接有何区别？
- 在Linux下如何使用设备？常用的设备名有哪些？
- 简述Linux的标准目录结构及其存放内容？
- Linux的基本命令格式如何？Linux下经常使用的通配符有哪些？
- 如何获得命令帮助？help命令和--help命令选项的作用分别是什么？

梁如军（linuxbooks@126.com）

本章思考题

- 常用的文件和目录操作命令有哪些？各自的功能是什么？
- 常用的信息显示命令有哪些？各自的功能是什么？
- 打包和压缩有何不同？常用的打包和压缩命令有哪些？
- 简述在Shell中可以使用哪几种方法提高工作效率。
- Linux下的隐含文件如何标识？如何显示？
- Linux下经常使用-f和-r参数，它们的含义是什么？
- Vi的3种运行模式是什么？如何切换？
- 什么是重定向？什么是管道？什么是**命令替换**？
- Shell变量有哪几种？如何定义和引用Shell变量？
- 登录Shell和非登录Shell的启动过程？
- 如何**设置**用户自己的工作环境？

本章实验

- 浏览并熟悉Linux目录结构。
- 学会使用命令帮助。
- 熟悉各种常用命令的使用。
- 熟悉文本编辑器Vi的使用。
- 熟悉并使用Shell的各种功能。
- 学会定义和输出Shell变量。
- 学会设置用户自己的工作环境。

梁如军（linuxbooks@126.com）

进一步学习

- 学习使用Vundle管理vim插件。
  - https://github.com/VundleVim/Vundle.vim
- 学习使用 tmux 和/或 screen。