



# Linux 操作系统使用

李会民

2005 年 6 月

中国科大 固体微结构研究室



# 内容

- 第一章 概述
- 第二章 系统的运行
- 第三章 文件和目录
- 第四章 shell 基础
- 第五章 vim
- 第六章 进程
- 第七章 Linux 工具
- 第八章 编译器
- 附录 网络文档



# 第一章 概述

- 描述 Linux 的发展和变化
- 介绍 Linux 操作系统的主要组成部分



# Linux 特点

- 多任务、多用户的操作系统
- 功能丰富的可扩展、开放的计算环境
- 可编程 shell



# Linux 的主要组件

- 内核
- 环境
- 文件结构



# shell

- Linux 和用户的界面
- 几种常见的 Shell
  - Korn
  - Bourne
  - C
- 缺省的 Shell —— Bash
- 命令解释器



## 第二章 系统的运行

- 登录及退出系统
- 修改密码
- Linux 的命令结构



# 登录及退出系统

- 本地登录：文本或 X 界面
- 远程登录：ssh 或 telnet
  - ssh team01@HostIP  
team01@HostIP's password: \*\*\*\*\*)
  - telnet HostIP  
Login: team01  
Password: \*\*\*\*\*)
- 退出系统：exit 或 ctrl+d





# 密码

- 创建或者改变密码：  
在系统提示符下输入 `passwd`  
`$passwd`  
team01's old password:  
team01's new password:  
Enter the new password again:



# 命令的格式

\$command options arguments

For Example :

\$mail -f newmail

正确

错误

分离

\$mail -f newmail    \$mail - f newmail

顺序

\$mail -f newmail    \$mail newmail -f

多参数

\$who -m -u

\$who -m-u

\$who -mu

\$who -m u



# 键入命令

- 在 shell 提示符下输入命令，然后按 Enter 键
- shell 识别大小写
- 如找不到输入的命令，会显示反馈信息：  
“Command not Found”
- 如命令太长，要在这行行尾键入 “\” 字符和按下 Enter 键，在下一行的 “>” 后接着输入



# 键盘的快捷方式

- `<ctrl-c>`: 停止命令
- `<ctrl-d>`: 结束传输或者文件
- `<ctrl-s>`: 临时停止输出
- `<ctrl-q>`: 恢复输出
- `<ctrl-u>`: 擦除光标以前的
- `<ctrl-k>`: 擦除光标以后的
- `<backspace>`: 纠正错误
- `<ctrl-r>`: 在以前的命令中搜索

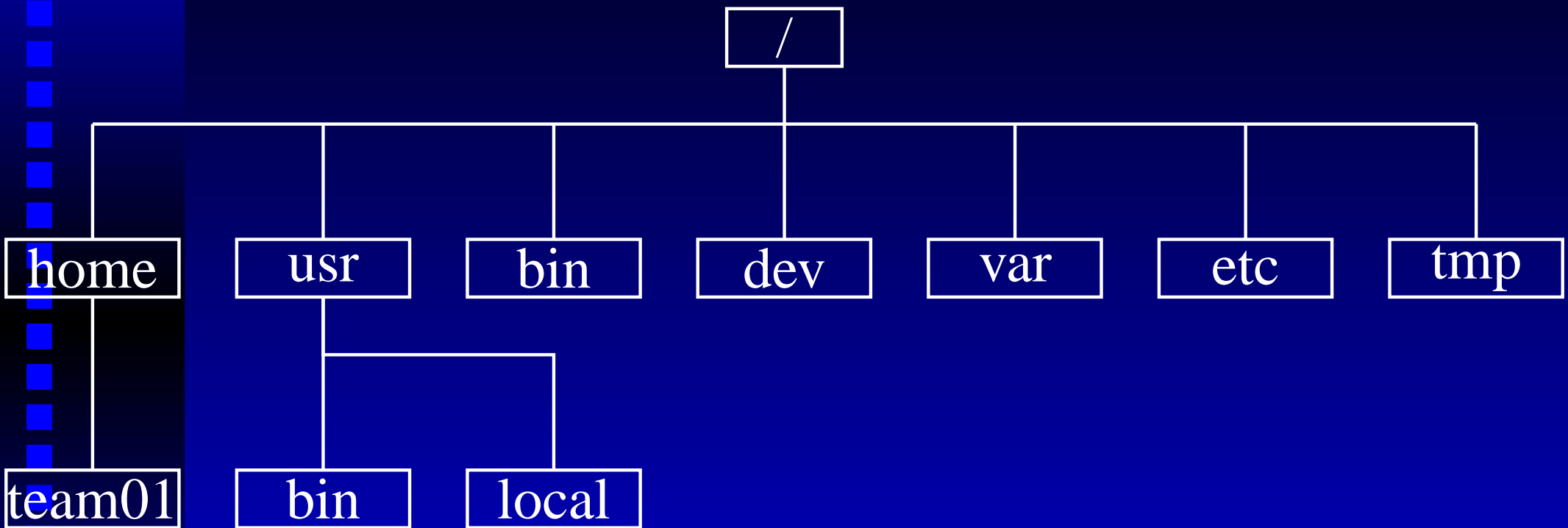


# 第三章 文件和目录

- 描述 Linux 文件系统的结构
- 描述不同的文件格式
- 文件的绝对路径和相对路径
- 创建、删除和列出路径
- 复制、显示、打印、移动、删除和连接文件



# 文件系统的层次结构





# Linux 中的标准系统目录

/	文件系统结构的起始点，称为根目录
/home	包含用户的主目录
/bin	包含了所有的标准指令和工具程序
/usr	包含了系统所使用的文件和指令
/usr/bin	包含了面向用户的命令和工具程序
/usr/lib	包含了编程语言库
/usr/local	包含了后来安装的软件
/usr/doc	包含了Linux文档
/usr/man	包含了在线的联机帮助手册
/usr/sbin	包含了系统管理员的命令
/sbin	包含了系统管理员开启系统的命令
/opt	包含了后来安装的软件
/var	包含了时变的文件，例如邮箱文件
/dev	包含了设备的文件接口
/etc	包含了系统配置文件和所有其它系统文件
/cdroom	光盘



# 文件系统的概念

- 文件系统：磁盘上有特定格式的一片区域。
- 文件：存储数据的一个命名的对象。
- 目录：包含许多文件项目的一类特殊文件。
- 子目录：被包含在另一个目录中的目录，包含子目录的目录称为父目录。
- 文件名：用来标识文件的字符串，保存在一个目录文件项中。
- 路径名：由“/”字符结合在一起的一个或多个文件名的集合。它指定一个文件在文件系统中的位置。





# 文件结构

无论文件是一个程序、文档、数据库、或是目录，操作系统都会赋予它下面的结构：

- - 索引节点（I 节点）：在文件系统结构中，包含有关相应文件信息（文件权限、文件主、文件大小等）的一个记录。
- - 数据：文件的实际内容。



# Linux 文件名称

- 包含大写键、小写键、数字、\$、@、\_
- 一般不包含以下字符：\* ? > < / ; \$ \ ' "
- 一般不能以 “+” 或者 “-” 开头
- 区分大小写
- 最长文件名 255



# 文件的类型

- 普通文件
- 目录文件
- 设备文件: `/dev/tty1`
- 连接文件: 存放文件系统中通向文件的路径
- `file` 文件名: 查看文件类型



# 普通文件

- 也称常规文件，包含各种长度的字符串。例如：信件、报告和脚本。
- 文本文件：由 ASCII 字符构成。
- 数据文件：由来自应用程序的数字型和文本型数据构成。例如：电子表格、数据库等。
- 可执行的二进制程序：由机器指令和数据构成。



# 目录文件

- 由成对的“ $I$ 节点号/文件名”构成的列表。利用目录文件可以构成文件系统的分层树形结构。
- $I$ 节点号是检索 $I$ 节点表的下标， $I$ 节点存放所有文件的状态信息
- 文件名是给一个文件分配的文本形式的字符串，用来标识文件。



# 路径的名称

类型:

- 绝对路径
- 相对路径



# 目录操作命令

- `ls` 显示目录中的内容
- `pwd` 显示当前和工作目录
- `cd` 改变用户工作目录
- `mkdir` 建立用户目录
- `rmdir` 删除目录



# 列出目录内容命令 **ls**

**ls** 命令列出一个子目录中的全部文件和目录名。它有 26 个命令行参数，下面列出来的是它最常用的几个。这些参数可以任意地组合使用。

- **-l** 每列仅显示一个文件或目录名称
- **-a** 显示所有文件或目录，包括以“.”为名称开头字符的文件、现行目录“.”与上层目录“..”
- **-l** 使用详细格式列表。将权限标示、硬件接数目、拥有者与群组名称、文件或目录大小及更改时间一并列出
- **-R** 递归处理，将指定目录下的所有文件及子目录一并处理





# ls

使用长列表方式列出某个子目录中的全部文件，  
使用下面的命令：

```
[ root@glede /root ] $ ls -la
```

```
total 16
drwxr-xr-x    4 root    root    4096 Jan  1 11:28 .
drwxr-xr-x   11 root    root    4096 Jan  1 11:27 ..
drwxr-xr-x    2 root    root    4096 Jan  1 11:27 team011
drwxr-xr-x    2 root    root    4096 Jan  1 11:28 team012
```

列出子目录中以字母 t 打头的全部非隐藏文件，  
使用下面的命令：

```
[ root@glede /root ] $ ls t*
```



# 显示当前工作目录命令 `pwd`

它没有参数，而它唯一的作用就是显示当前工作目录的绝对路径的名称。

```
$pwd
```

```
/home/team01
```



# 改变用户工作目录 `cd`

- `cd` 指令可以让用户在不同的目录间切换，但该用户必须有足够的权限进入目的目录
- `cd [目录名]`
- `cd ~用户名`
- `cd -`: 返回进入这个目录之前所在的目录

```
$pwd
```

```
/home/eagle/tex/linux
```

```
$cd ~/phd
```

```
$pwd
```

```
/home/eagle/phd
```

```
$cd -
```

```
$pwd
```

```
/home/eagle/tex/linux
```



# 建立用户目录命令 **mkdir**

- **mkdir** 可建立目录同时还可给目录设置权限。
- **mkdir [-p] [-m] [文件名]**
- **-p** 若所要建立目录的上层目录目前尚未建立，则会一并建立上层目录
- **-m** 建立目录时，同时设置目录的权限。权限的设置法与 **chmod** 指令相同
- 建立目录 **team02**，并让全部人都有 **rxw** 的权限

```
$ls
```

```
team01
```

```
$mkdir -m 777 team02
```

```
$ls
```

```
team01 team02
```



# 删除目录命令 **rmdir**

- 当有空目录要删除时，可用 **rmdir** 指令。若所给予的目录非空目录，则会出现错误信息。
- **rmdir [-p] [目录名]**
- **-p** 删除指定目录之后，若该目录的上层目录已变成空目录，则将其一并删除



# 文件操作命令

<code>cp</code>	复制文件或目录
<code>mv</code>	移动文件和文件换名
<code>rm</code>	删除文件或目录
<code>ln</code>	在文件间建立连接
<code>find</code>	查找特定的文件
<code>locate</code>	查找特定的文件
<code>which</code>	查看命令的路径
<code>touch</code>	改变文件的时间参数



# 复制文件或目录命令 **cp**

- **cp** 命令用来复制文件。在缺省的情况下，这个命令工作的时候不做任何显示；只有在出现一个错误情况的时候才显示状态信息。
- **cp** [源文件名] ……[目标文件名]
- **cp -r** [源目录名] [目标目录名]



# 移动/重命名文件命令 **mv**

- **mv** 命令用来把文件从一个位置移动到另外一个位置,也可以从一个分区移动到另外一个分区。
- **mv** [源文件列表][目标文件]





# 删除文件或目录 **rm**

- **rm** [选项][文件列表]
- **-r** 删除文件列表中的目录
- **-i** 指定交互模式。在执行删除前提示确认。
- 文件列表：希望删除的用空格分隔的文件列表，可以包括目录名



# 在文件间建立连接 **ln**

- **ln** 命令用来建立硬连接和符号连接。硬连接是一个文件的额外的名字，没有源文件，硬连接便不能存在。而对于符号连接，当原文件被删除后，符号连接仍然存在。
- **ln** [选项] 源文件 目标文件
- **ln** [选项] 源文件列表 目标目录



# ln

- -s 建立一个符号连接而不是硬连接
- -d 建立目录的硬连接
- 现有文件 file1, file2 与目录 team01, 欲在 team02 中建立起符号连接  

```
$ln -s /home/file1 /home/file2 /home/team01 team02
```
- 设 dir3 是一个目录的符号连接, 现在建立其硬连接 dir4  

```
$ln -d dir3 dir4
```



# 查找文件命令 **find**

**find** 命令可以根据各种检索条件查找文件

- **find** [路径...] [表达式]
- 路径...: 准备寻找文件所在的路径以及它的子路径, 也可以是多个路径。
- 表达式: 包含要搜索文件的条件, 可以包含文件名、拥有者、最后修改时间等。



# find 常用参数

- `-atime n` 至少  $n*24$  小时内没有访问过的文件
- `-ctime n` 至少  $n*24$  小时内没有修改过的文件
- `-amin n n`  $n$  分钟之前访问过的文件
- `-cmin n n`  $n$  分钟之前修改过的文件
- `-empty` 文件为空
- `-name name` 指定要寻找的文件或目录的名称
- `-type x`: 以文件的类型作为寻找的条件。若  $x$  为“d”,则表示寻找目录;  $x$  为“f”,表示寻找普通文件;  $x$  为“c”,表示寻找字符特殊设备;  $x$  为“b”,表示寻找特殊块设备;  $x$  为“p”,表示寻找命名管道;  $x$  为“l”,表示寻找符号连接;  $x$  为“s”,表示寻找套接字。



# find 举例

- 如果想查找 /home 子目录中至少 7 天没有被访问过的文件，请使用下面的命令：  
`$find /home -atime 7 -print`
- 如果想找出 /usr/src 子目录中名字为 core 的文件并删除它们，请使用下面的命令：  
`$find /usr/src -name core -exec rm { } \;`
- 如果想找出 /home 中以 .jpg 结尾并且长度超过 100K 的文件，请使用下面的命令：  
`$find /home -name "*.jpg" -size 100k`



# locate/which

- 从数据库中搜索: `locate filename`
- 查看命令的路径: `which ls`



# 改变文件的时间参数 **touch**

- 改变文件访问和修改时间，或用指定时间建立新文件。
- **touch** [选项] MMDDhhmmYY 文件列表
- **-a** 只更改访问时间
- **-c** 若目标文件不存在，不建立空的目标文件

不带参数的 **touch** 命令将文件的时间改为当前时间

```
$ls
```

```
-rw-r--r--  1 root  root          37350 Jan 27  2003 file1
```

```
$touch file1
```

```
-rw-r--r--  1 root  root          37350 Jan  1 16:15 file1
```

使用选项**-t** 直接修改时间

```
$touch -t 01201500 file1
```

```
-rw-r--r--  1 root  root          37350 Jan 20 15:00 file1
```





# 文件显示命令

- cat 显示和合并文件
- more 分屏显示文件
- less 分屏显示文件
- head 显示文件的前几行
- tail 显示文件的最后几行



# 显示和合并文件命令 **cat**

可以结合多个文件，并将它们的内容输出到标准输出设备。

`cat` [选项] [文件列表]

- **-b** 列出文件内容时，在所有非空白列之开头标上编号，从 1 开始累加
- **-E** 在每一列的最后标上“\$”符号
- **-n** 列出文件内容时，在每一列之开头标上编号，从 1 开始累加



# cat 举例

让 cat 指令从标准输入设备（如键盘）读取数据，转而输出至标准输出设备（如显示器）

- `$ cat` 执行指令，不加任何参数  
123 键入任何文字后，按下回车键  
123 系统回应一模一样的文字利用特殊字符“>”将名称为 file1 与 file2 的文件合并成一个文件 file3:
- `$ cat file1 file2 > file3`  
若文件 file3 已经存在，则其内容会被覆盖过去；欲避免这种状况发生，可用“>>”代替“>”，新的内容就会附加在原有内容之后，而不会覆盖它。



# 分屏显示文件命令 **more**

- **more** 可将文件内容显示于屏幕上，每次只显示一页。可以往下浏览，但无法向上浏览，**less** 指令可以上下浏览。
- **more** [选项] [文件名]
- 显示中按 **v**，可调用 **vi** 直接进行编辑



# 分屏显示文件命令 **more**

- -<行数> 指定每次要显示的行数
- +/<字符串> 在文件中查找选项中指定的字符串，然后显示字符串所在该页的内容
- +<行数> 从指定的行数开始显示
- -n 每次只显示 n 行
- -c 不滚屏，在显示下一屏之前先清屏
- 在文件 `file1` 中查找“123”字符串，然后从改页开始显示文件的内容：  
`$more +/123 file1`
- 显示文件 `file1` 的内容，每 10 行显示一次，而且在显示之前先清屏。  
`$more -c -10 file1`



# 分屏显示文件命令 **less**

- 类似 more，但比 more 要强大的多
- 显示中按 v，可调用 vi 直接进行编辑



# 显示文件的前几行命令 **head**

- 在屏幕上显示指定文件的开头若干行。默认值是 10 行。
  - head [选项] 文件名
  - -c N: 显示前 N 个字节
  - -n N: 显示前 N 行
- ```
$head -5 file
```



# 显示文件的最后几行命令 **tail**

- 在屏幕上显示指定文件的末尾若干行。默认值是 10 行。
- **tail** [选项] 文件名
- **-c N**: 显示前 N 个字节
- **-n N**: 显示前 N 行
- **+N**: 从文件开头的第 N 行开始显示





# 比较文件内容命令

`comm` 比较两个已排过序的文件  
`diff` 比较文件的差异



# comm 命令

- 用来对两个已排过序的文件进行逐行比较
- `comm [-123] file1 file2`
- -1 不显示只在第一个文件里出现的行
- -2 不显示只在第二个文件里出现的行
- -3 不显示在第一、二个文件里都出现的行



# comm

file1 的内容如下: file2 的内容如下:

```
main (  
{  
    printf("Hello!\n");  
}
```

```
main (  
{  
    printf("Good!\n");  
}
```

用 comm 命令对这两个文件进行比较只显示它们共有的行。

```
$comm -12 file1 file2  
    main (  
    {  
    }
```



# diff 命令

- 比较两个文本文件，并显示它们的不同
- `diff [选项] file1 file2`
- `-c` 输出格式是带上下文的三行格式
- `-C n` 输出格式是带上下文的 `n` 行格式
- `-r` 两个文件都是目录时，递归比较找到的各子目录



# diff

- 输出的一般形式如下：  
n1 a n3,n4  
n1,n2 d n3  
n1,n2 c n3,n4
- a-附加， d-删除， c-修改
- file1 的内容如下： file2 的内容如下：

```
1 main ()
2 {
3  printf ("Hello !\n");
4 }
5
```

```
1 main ()
2 {
3  int n,m;
4  n=10;
5  printf ("%d\n",m=n*10);
6 }
```



# diff

输入命令

```
$diff file1 file2
```

```
3,5c 3,6
```

```
<3 printf("Hello!\n");
```

```
<4 }
```

```
<5
```

```
-----  
>3 int n,m;
```

```
>4 n=10;
```

```
>5 printf("%d\n",m=n*10);
```

```
>6 }
```



# 文件权限操作

- `chmod` 改变文件或目录的许可权限
- `chown` 改变文件的所有权
- `chgrp` 改变用户分组



# 文件的保护和权限

rwx

rwx

rwx

user

group

others

一个普通文件

- r: 可以查看文件内容
- w: 可以修改文件内容
- x: 可以执行文件

一个路径

- r: 可以查看文件夹下的文件
- w: 可以在文件夹下创建和删除文件
- x: 可以进入文件夹或者访问文件夹下的文件





# 权限

|     | user                  | group                 | others                |
|-----|-----------------------|-----------------------|-----------------------|
| 符号  | <code>rwX</code>      | <code>rw-</code>      | <code>r-</code>       |
| 二进制 | <code>111</code>      | <code>110</code>      | <code>100</code>      |
| 八进制 | <code>7(4+2+1)</code> | <code>6(4+2+0)</code> | <code>4(4+0+0)</code> |

缺省的文件权限:

`file -rw-r--r-- 644`

`directory drwxr-xr-x 755`



# 改变文件属性命令 **chmod**

- 用来改变文件或目录的权限
- `chmod [选项] 模式文件列表`



# 改变文件的权限

- **u** : 文件所属帐户
- **g** : 文件所属组
- **o** : 其余帐户
- **+** : 添加权限
- **-** : 去除权限
- **=** : 使得指定文件只具有这些权限



# chmod

- 使文件 `file` 在各个级别拥有所有权限  
`$chmod 777 file`
- 允许所有人读 `file`，但只有拥有者能改变它  
`$chmod 644 file`
- 给所有人增加写权  
`$chmod a+w file`
- 对组级和其他用户除去写权和读权  
`$chmod o-wr,g-wr file`
- 建立其他用户的只读权  
`$chmod o=r file`



# 改变文件的所有权命令 **chown**

- **chown** 命令可以把一个文件的所有权修改为别人的。只有 **root** 用户能够进行这样的操作。
- **chown** [选项] 用户文件列表
- **-v** 详细说明所有权的变化
- **-r** 递归改变目录及其内容的所有权



# 改变用户分组命令 **chgrp**

- **chgrp** 命令可以改变一个文件的用户分组设置情况
- **chgrp** [选项] 用户文件列表
- **-v** 详细说明文件所属的用户组的变化
- **-r** 改变本目录及其所有子目录中的文件所属的用户组



## 第四章 vi/vim

两种模式:

- 命令模式: 从键盘上输入的任何字符都被作为编辑命令来解释, 按<ESC>进入此模式。
- 输入模式: 从键盘上输入的所有字符都被插入到正在编辑的缓冲区中, 被当作正文, 按a、i、c、o等进入此模式。



# 进入/退出 vi

- 进入 vi  
vi vifile
- 退出 vi
  - :q 退出未被编辑过的文件
  - :q! 强行退出 vi
  - :x 存盘退出 vi
  - :wq 存盘退出 vi



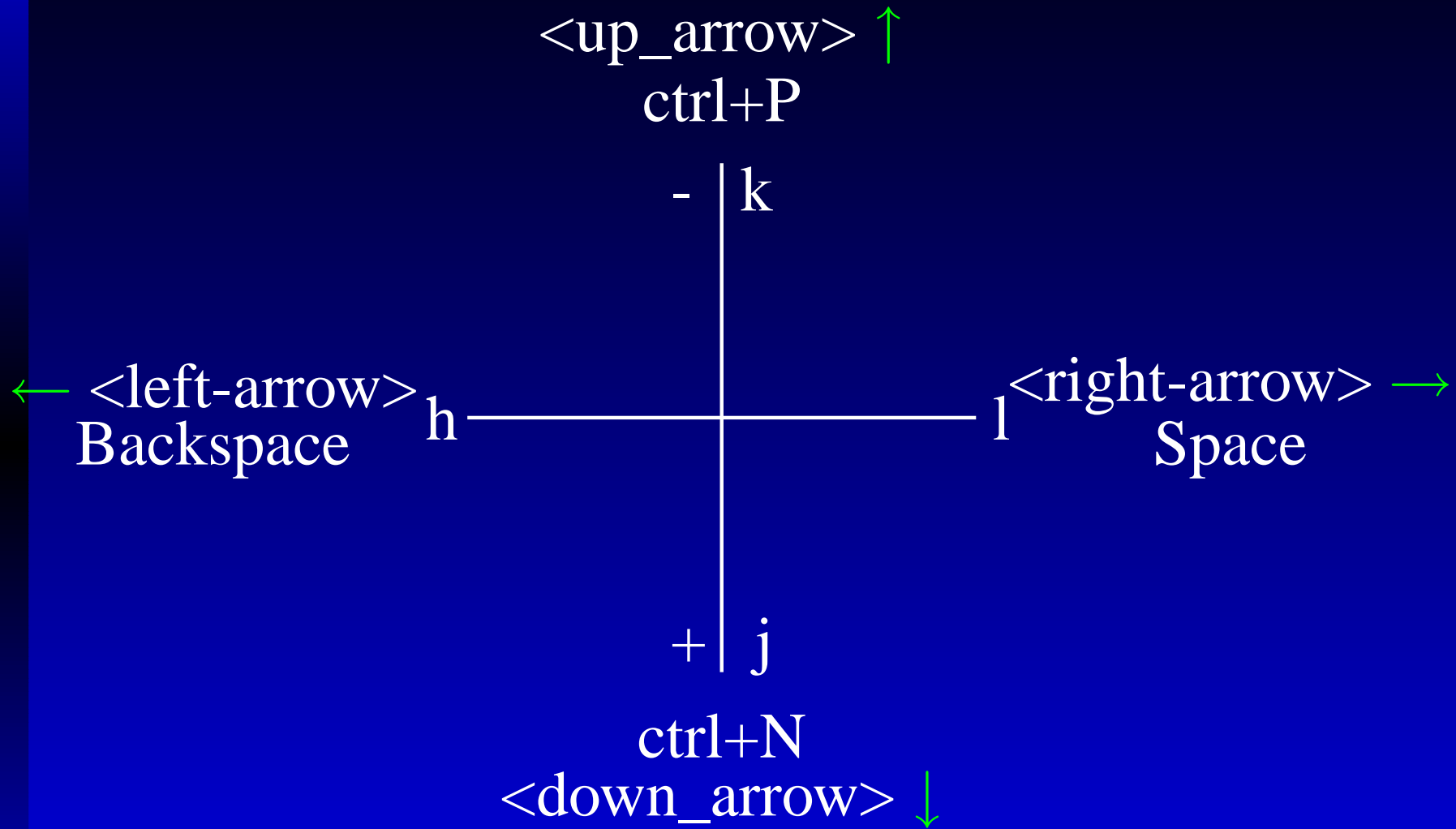


# 文本输入

- 插入命令: i 和 I
- 附加命令: a 和 A
- 新行命令: o 和 O



# 移动光标





# 光标移动

- 移至行首: `^`、`0`
- 移至行尾: `$`
- 移至指定行: `[行号] G` 或: `[行号] [Enter]`
- 移至指定列: `[列号] |`



# 屏幕命令

- 滚屏命令: `ctrl+u` 和 `ctrl+d`, 分别向上和向下滚动半个窗口
- 分页命令: `ctrl+f` 和 `ctrl+b`, 分别向前和向后分页



# 文本删除

- 删除字符
  - **x** 或 **nx**: 从光标所在的位置删除一个或 **n** 个字符
  - **X** 或 **nX**: 删除光标前的一个或 **n** 个字符
- 删除文本对象
  - **dd**: 删除光标所在的行
  - **D**: 删除从光标位置开始至行尾
  - **dw**: 删除从光标位置至该词末的所有字符
  - **d0**: 删除从光标位置开始至行首
  - **d5G**: 将光标所在行至第 5 行删除
  - **dG**: 将光标所在行到文件末尾删除



# 修改命令

- 替换一个字符：命令模式下按 **r**
- 替换本行光标出字符以后的：命令模式下按 **R**
- 替换掉光标处字符并添加：命令模式下按 **c**
- 修改光标后的到行尾：命令模式下按 **C**
- 大小写转换：命令模式下按 **~**



# 搜索和替换命令

- 搜索: /所要搜索的内容
- 替换: %s/替换前/替换后/g



# 复原和重复命令

- **u**: 若插入后用此命令，就删除刚插入的正文；若删除后用它，则插入刚删除的正文。
- **U**: 把当前行恢复成它被编辑之前的状态
- **::** 重复实现刚才的插入命令或删除命令





# 要遵守的步骤提要

|          |                 |
|----------|-----------------|
| 进入 vi    | 键入 vi 并按回车      |
| 到输入模式    | 按<a>或者<i>       |
| 输入文本     | 将文本键入缓冲区        |
| 到命令模式    | 按<Esc>          |
| 保存缓冲区到文件 | 键入:w file ,并按回车 |
| 退出 vi    | 键入:q ,并按回车      |

vim 中文帮助: <http://vcd.gro.clinux.org/>



# 第五章 shell 基础

- 通配符
- 输入输出重定向
- 管道
- 命令组



# Shell 简介

- 作为操作系统的交互式命令解释程序，它在用户和操作系统之间提供了一个面向行的可交互接口。
- 作为一种命令级的程序设计语言，具有变量设置、结构控制、子程序调用、参数传递、中断处理等



# 文件名中的字符代用字

- 单字符代用字: ?
- 多字符代用字: \*
- 包含代用字: [ ] ! [ - ]



# 标准文件

- 标准输入 (0)
- 标准输出 (1)
- 标准错误输出 (2)



# 输入输出重定向

- 输入重定向: `command < filename`
- 输出重定向: `command > filename`
- 错误重定向: `command 2> filename`



# 管道

- 一个命令的标准输出成为另一个命令的标准输入  
`cmd1 | cmd2`
- 将 `ls` 命令输出的文件名列表输送到 `grep` 命令  
`$ls | grep t`



# 自动补全命令行

- 自动补全命令行就是在输入命令时不必把命令输全，shell 能判断出所要输入的命令。
- 输入命令的一部分后按<Tab>键
- `$pass<Tab>` 系统会补全 `passwd` 命令





# Shell 变量

变量是可赋值的名字。它的值可以是字符串、数字等。

- 用户变量：由用户创建和赋值的变量
- 环境变量：由 shell 维护，用于配置系统工作环境的一组变量，可以由用户改变
- 特殊变量：由 shell 设置的，不能改变。例如参数个数，进程号退出状态。



# 用户变量

变量名可以由字母开头的任意字母、数字组成的序列。

- 申报和管理用户变量：  
`var=string` 或 `set var=string`
- 取消变量的定义：  
`unset var`
- 显示变量的值：  
`echo`



# 变量

- 要生成一个值为整数的变量

```
int=5
```

- `set var1=abcd`

```
set var2=var1$efgh
```

执行上面两条语句，变量 `var2` 的内容为：`abcdefgh`

- 显示上面 `var2` 的值

```
echo $var2
```

则输出：`abcdefgh`



# 第六章 进程

- 定义进程
- 进程监视
- 调用后台进程
- 中断进程
- nohup
- 控制 jobs
- 定义系统进程



# 进程概念

- 一个进程就是一个运行的程序。是动态的
- **Linux** 为每一个进程分配一个进程标识号 (PID) 指定和跟踪进程



# 进程和程序的关系

- 进程是程序的执行过程
- 程序是一个静态的指令集，进程是动态的
- 进程之间是并发执行的，而程序本身没有并发
- 进程是分配资源的单位，在运行过程中使用系统资源



# 父进程和子进程

| PID | PPID |
|-----|------|
| 201 | 1    |
| 206 | 201  |
| 207 | 206  |



# ps 命令

查看当前系统中运行的进程的信息

ps [选项]

- -a 显示系统中与 tty 相关的所有进程的信息
- -f 显示所有进程的信息
- -r 只显示正在运行的进程
- -u 显示面向用户的格式
- -x 显示所有终端上的进程信息





# 中断进程

- 前台进程  
ctrl-c  
kill  
killall
- 后台进程kill  
killall



# 杀掉进程

```
$ ps -f
```

```
UID      PID      PPID     $ \cdots$TTY   $ \cdots$COMMAND
john     206      201     $ \cdots$1     $ \cdots$bash
john     209      206     $ \cdots$1     $ \cdots$vim x.txt
```

```
$ kill 206
```

```
$ kill -9 206
```

```
$ killall vim
```



# 运行耗时长进程

- `ls -R 1 > out &`
- `nohup`  
`$nohup ls -R 1 > out &`  
`$nohup ls -R 1 &`



# 查看 CPU 利用率 top

- 用法: `topr [选项]`
- 退出: `q`



# 第七章 Linux 工具



# date 和 cal 命令

- 在屏幕上显示或设置系统的日期和时间

```
$date
```

```
Thu Jan 27 05:34:40 CST 2000
```

- 显示日历

```
$cal 6 2005
```

| Sun | Mon | Tue | Wed | Thu | Fri | Sat |
|-----|-----|-----|-----|-----|-----|-----|
|     |     |     | 1   | 2   | 3   | 4   |
| 5   | 6   | 7   | 8   | 9   | 10  | 11  |
| 12  | 13  | 14  | 15  | 16  | 17  | 18  |
| 19  | 20  | 21  | 22  | 23  | 24  | 25  |
| 26  | 27  | 28  | 29  | 30  |     |     |



# clear 命令

清除屏幕上的信息。清除后，提示符移到屏幕的左上角

```
$clear
```



# echo 命令

- 将命令行中的参数回显到标准输出（即屏幕）上。
- `echo [-n] STRING`
- `-n` 表示输出字符串后，光标不换行
- 几个例子：
  - `$echo 'This is a command.'`  
This is a command.
  - `$echo This is a command`  
This is a command.
  - `$echo -n 'Enter data->'`  
Enter data->\$\_





# grep 命令

- 在文本文件中查找指定模式的词或短语。
- `grep [选项] [查找模式][文件名 1, 文件名 2, ……]`
- 如果在搜索模式中包含空格，应用单引号把模式字符串括起来
- 在文件列表中可以使用通配符
- 对现行目录中，所有扩展名为“.txt”的文件之内容，查找包含“hello”字符串的文件  
`$grep hello *.txt`
- 在文件file1中查找字符串“ramble.\*b”  
`$grep 'ramble*b' file1`



# wc 命令

- 计算字数

```
$wc [-c] [-l] [-w] filename
```

-c 只显示计算字节数

-l 只显示计算行

-w 只显示计算字

- \$ wc myfile

```
17      126      1085      myfile
```

```
lines  words  字节数
```



# ping 命令

用来检测一个系统是否已连接上并在运行。

```
$ping 10.99.19.44
```

```
PING 10.99.19.44 (10.99.19.44) from 10.99.19.44 : 56(84) bytes  
of data.
```

```
64 bytes from 10.99.19.44: icmp_seq=1 ttl=255 time=0.092 ms
```

```
64 bytes from 10.99.19.44: icmp_seq=2 ttl=255 time=0.022 ms
```

```
64 bytes from 10.99.19.44: icmp_seq=3 ttl=255 time=0.020 ms
```

```
64 bytes from 10.99.19.44: icmp_seq=4 ttl=255 time=0.019 ms
```

```
— 10.99.19.44 ping statistics —
```

```
4 packets transmitted, 4 received, 0% loss, time 2997ms
```

```
rtt min/avg/max/mdev = 0.019/0.038/0.092/0.031 ms
```



# finger 命令

获得网络中其他用户的信息。可以查看一个用户最后登陆的时间、他所使用的shell类型、他的主目录的路径等。

```
$finger root
```

```
Login name: root
```

```
Directory: / Shell: /bin/bash
```

```
On since Jan 27 05:34:17 on pts/1, 14  
seconds Idle Time from 128.0.0.71
```



# 联机帮助命令 **man**

- **man** 命令可以格式化并显示某一命令的联机帮助手册
- **man** [选项] 命令名
- **\$man ls**
- **man** 命令输出的指南页主要包括以下几个部分：
  - **NAME** 命令的名称和用法
  - **SYNOPSIS** 显示命令的语法格式，列出其所有可用的选用的选项及参数。
  - **DESCRIPTION** 描述命令的详细用法及每个选项的功能。
  - **OPTION** 对命令的每一个选项进行详细的说明



# 阅读 **info** 帮助命令 **info/pinfo**

阅读 info 文档

- **info** [选项] 命令名
- **pinfo** [选项] 命令名



# 命令 `-help/-h`

- 命令 `-help` 或 `-h` 可以显示这个命令的帮助并且退出
- `$ls -help`
- `$man -h`



# 查看目录大小命令 **du**

- **du**: 总结每个<文件>的磁盘用量，目录则取总用量。
- 用法: **du** [选项] [文件]

```
$du -h /home/eagle/data/  
590M    /home/eagle/data/surface/old  
926M    /home/eagle/data/surface  
31M     /home/eagle/data/1  
34M     /home/eagle/data/ asem  
2.8M    /home/eagle/data/line/old  
105M    /home/eagle/data/line  
1.1G    /home/eagle/data/
```





# 查看分区大小命令 **df**

- **df**: 显示每个<文件>所在的文件系统的信息，默认是显示所有文件系统。
- 用法: **df** [选项] [文件]

```
$df -h
```

| Filesystem | 容量   | 已用   | 可用   | 已用%  | 挂载点      |
|------------|------|------|------|------|----------|
| /dev/hda1  | 958M | 728M | 181M | 81%  | /        |
| tmpfs      | 253M | 4.0K | 253M | 1%   | /dev/shm |
| /dev/hda6  | 9.2G | 4.3G | 4.5G | 49%  | /usr     |
| /dev/hda7  | 65G  | 61G  | 212M | 100% | /home    |
| /dev/hda8  | 36G  | 30G  | 4.1G | 89%  | /backup  |
| tmpfs      | 10M  | 2.8M | 7.3M | 28%  | /dev     |



# 常用软件/命令

- 网络浏览: mozilla、firefox
- 网络下载: ftp、lftp、wget、curl、gftp
- 看图程序: gqview、gthumb



# ftp

- 连接服务器：
  - ftp Host
  - ftp User@Host
  - ftp User:Passwd@Host
- 查看文件列表：ls
- 单个/多个文件下载：get/mput
- 单个/多个文件上载：put/mget
- 查看帮助：help, help CMD
- 二进制传输：bin
- ASCII 传输：asc



# lftp

类似 ftp，但要强大的多，一般系统未必安装  
主要特殊命令：

- 目录上载：`mirror`
- 目录上载：`mirror -R`
- 查看文件：`cat`、`zcat`
- 运行本地命令：`!CMD`，比如`!ls`
- 下载已知地址文件：`lftp ftp://url`



# wget/curl 下载

文件下载，并可以批量下载以及镜像网站

- 文件续传: `wget -c http://url`
- 下载 `http://vcd.gro.clinux.org/doc/usr_01.html - http://vcd.gro.clinux.org/doc/usr_15.html:`  
`curl http://vcd.gro.clinux.org/doc/usr_[01-15].html`



# 第八章 编译器

- Fortran:
  - Intel: ifort、ifc
  - PGI: pgf90、pgf77
  - GNU Fortran: g77、gfortran
- C/C++
  - Intel: icc
  - PGI: pgcc
  - GNU C: gcc、g++



# 附录 网络文档

- 常用论坛:

<http://www.linuxsir.org>

- 常用软件推荐集合汇总:

<http://www.linuxsir.org/bbs/showthread.php?t=199479>

- 指令大全:

<http://www.linuxsir.org/bbs/showthread.php?t=204304>



谢谢!