RHCE 部分视频笔记摘要

RHCE033

whatis：查询一个命令的功能简介
man -k keyword:-k 参数就是查看含有关键字的一些帮助信息
/usr/share/doc:一些帮助文档存放位置

所有的文件名不超过 255 个字符，Linux 不存在扩展名的概念，所有的东西都只是命名，只所以加上扩展名，是为了便于人们识别

系统中.表示当前目录，..表示上一级目录，这两种表示方法为相对路径
绝对路径是从/开始的

alias 默认 cp 的别名为 cp -i

chggrp 组名 文件名 == chown :组名 文件名 ——》修改文件的属组
cp -d 复制链接文件本身

系统用户主要是管理服务的，一般是不能被登录的

每个用户只有一个 uid，但可以有多个 gid

Shell：人机交互的中间层，就像一个翻译官，在中间起到一个翻译的作用
Bash 是 shell 中的一种，比如说英语、汉语等不同的语言一种，shell 比较流行相当于世界中的英语

```
[root@station17 ~]# vim /etc/shells
[root@station17 ~]# sh
sh-3.1# ll
sh: ll: command not found
```

比喻说 ll 在 sh 这种 shell 中翻译不了，而在 bash 下就可以翻译出来

---

*：匹配 0 个或多个字符
?：仅仅匹配一个
[0-9]：匹配所有的数字
[a-z]：匹配所有的小写字母
[^abc]：匹配非[]中的内容
rm -rf ab[cd]：删除 abc 或者删除 abd，只有取[]中的单一字符

```
[root@station17 ~]# vncviewer --shared --viewonly 192.168.0.17:0
```

/etc/login.defs：文件是当创建用户时的一些规划，比如创建用户时，是否需要家目录，UID 和 GID 的范围；用户的期限等等，这个文件是可以通过 root 来定义的

**$()和``** → ``不支持嵌套，所以推荐使用$()

```
[root@localhost etc]# hostname
localhost.localdomain
[root@localhost etc]# echo "This system's name $(hostname)"
This system's name localhost.localdomain
[root@localhost etc]# echo "This system's name `hostname`"
This system's name localhost.localdomain
[root@localhost etc]# _
```

调用命令执行的结果

{}列出其中还的所有内容：

```
$ echo file{1,3,5}
file1 file3 file5
$ rm -f file{1,3,5}
```

hi=hello　//当前 shell 下生效的变量
echo $hi
hello

标准输入、输出(I/O)以及管道

- Linux provides three I/O channels to Programs
  - Standard input (STDIN) - keyboard by default
  - Standard output (STDOUT) - terminal window by default
  - Standard error (STDERR) - terminal window by default

- Supported operators include:
  - > Redirect STDOUT to file
  - 2> Redirect STDERR to file
  - &> Redirect all output to file

>输出重定向
<输入重定向
>>追加

- **mail**: Send input via email:

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

-s 表示主题 管道之前的 echo 的内容就是邮件的内容

默认情况下错误信息不会传递给管道

使用这种方式可以看出只有 1（即是正确的通过了管道）：

```
[root@localhost ~]# tail -f /etc/passwd | grep root

[1]+  Stopped                 tail -f /etc/passwd | grep root
[root@localhost ~]# ps
  PID TTY          TIME CMD
 3479 tty1     00:00:01 bash
20587 tty1     00:00:00 tail
20588 tty1     00:00:00 grep
20589 tty1     00:00:00 ps
[root@localhost ~]# ll /proc/20587/fd/
total 0
lrwx------ 1 root root 64 Feb  9 20:20 0 -> /dev/tty1
l-wx------ 1 root root 64 Feb  9 20:20 1 -> pipe:[72505]
lrwx------ 1 root root 64 Feb  9 20:20 2 -> /dev/tty1
lr-x------ 1 root root 64 Feb  9 20:20 3 -> /etc/passwd
[root@localhost ~]# _
```

- **2>&1**: Redirects STDERR to STDOUT
  - Useful for sending all output through a pipe

```
$ find /etc -name passwd 2>&1 | less
```

```
[root@station17 ~]# (find /etc/ -name passwd ; ls)  > /tmp/test1.txt
```

()把()内的输出作为一个组，最后合起来传给后面

tee 每个管道都输出一个结果

```
[root@station17 tmp]# ls -lR /etc/ |tee stage1.out |sort |tee stage2.out|uniq -
c |tee stage3.out |sort -r |tee stage4.out |less
[root@station17 tmp]# ls
stage1.out  stage2.out  stage3.out  stage4.out
```

- Redirect multiple lines from keyboard to STDIN with ***<<WORD***
  - All text until ***WORD*** is sent to STDIN
  - Sometimes called a *heretext*

```
$ mail -s "Please Call" jane@example.com <<END
> Hi Jane,
>
> Please give me a call when you get in. We may need
> to do some maintenance on server1.
>
> Details when you're on-site,
> Boris
> END
```

循环语句：

- Performs actions on each member of a set of values
- Example:

```
for NAME in joe jane julie
do
     ADDRESS="$NAME@example.com"
     MESSAGE='Projects are due today!'
     echo $MESSAGE | mail -s Reminder $ADDRESS
done
```

- Can also use command-output and file lists:
  - **for num in $(seq 1 10)**
    - Assigns 1-10 to $num
    - **seq X Y** prints the numbers X through Y
  - **for file in *.txt**
    - Assigns names of text files to $file

```
[root@station17 ~]# for USER in user2 admin2 redhat2
> do
> useradd $USER
> echo "password" |passwd --stdin $USER
> done
Changing password for user user2.
passwd: all authentication tokens updated successfully.
Changing password for user admin2.
passwd: all authentication tokens updated successfully.
useradd: user redhat2 exists
Changing password for user redhat2.
passwd: all authentication tokens updated successfully.
```

```
for n in {1..20}; do
   host=192.168.0.$n
    ping -c2 $host &> /dev/null
    if [ $? = 0 ]; then
         echo "$host is UP"
   else
        echo "$host is DOWN"
   fi
done
```

```
[root@station17 tmp]# grep -A 5 --color=auto 'alice' passwd
alice:x:509:509::/home/alice:/bin/bash
bob:x:510:510::/home/bob:/bin/bash
named:x:25:25:Named:/var/named:/sbin/nologin
user2:x:511:511::/home/user2:/bin/bash
admin2:x:512:512::/home/admin2:/bin/bash
redhat8:x:513:513::/home/redhat8:/bin/bash
```

-A 表示找到关键字的后多少行

-B 表示找到关键字的前多少行

```
[root@station17 tmp]# grep -B 5 --color=auto 'alice' passwd
dovecot:x:97:97:dovecot:/usr/libexec/dovecot:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
redhat5:x:507:507::/home/redhat5:/bin/bash
it2:x:508:508::/home/it2:/bin/bash
redhat3:x:505:508::/home/redhat3:/bin/bash
alice:x:509:509::/home/alice:/bin/bash
```

```
[root@localhost ~]# grep 'bash$' /etc/passwd --color=auto
root:x:0:0:root:/root:/bin/bash
wl:x:501:501::/home/wl:/bin/bash
redhat:x:502:502::/home/redhat:/bin/bash
wu:x:503:503::/home/wu:/bin/bash
[root@localhost ~]# grep '^root' /etc/passwd --color=auto
root:x:0:0:root:/root:/bin/bash
[root@localhost ~]# _
```

```
[root@station17 tmp]# ifconfig eth0 |grep 'inet addr' |cut -d: -f2 |cut -d' ' -f1
192.168.0.17
[root@station17 tmp]#
```

- **sed** addresses
  - **sed 's/dog/cat/g' pets**
  - **sed '1,50s/dog/cat/g' pets**
  - **sed '/digby/,/duncan/s/dog/cat/g' pets**
- Multiple **sed** instructions
  - **sed -e 's/dog/cat/' -e 's/hi/lo/' pets**
  - **sed -f myedits pets**

-e 选项是为了实现执行两次操作

```
[root@station17 tmp]# grep --color=auto :root: /etc/passwd
root:x:0:0:root:/root:/bin/bash
[root@station17 tmp]#
```

Using multiple "windows"

- Multiple documents can be viewed in a single **vim** screen.
  - **Ctrl-w**, **s** splits the screen horizontally
  - **Ctrl-w**, **v** splits the screen vertically
  - **Ctrl-w**, ***Arrow*** moves between windows
- Ex-mode instructions always affect the current window
- **:help windows** displays more window commands

VIM 分屏显示，先按 Ctrl-w 再按 s 或者 v 或者方向键，按 Ctrl-w 后按 q 为退出

分屏启动 Vim

1. 使用大写的 O 参数来垂直分屏。

vim -On file1 file2 ...

2. 使用小写的 o 参数来水平分屏。

vim -on file1 file2 ...

注释: n 是数字，表示分成几个屏。

关闭分屏

1. 关闭当前窗口。

Ctrl+W c

2. 关闭当前窗口，如果只剩最后一个了，则退出 Vim。

Ctrl+W q

分屏

1. 上下分割当前打开的文件。

Ctrl+W s

2. 上下分割，并打开一个新的文件。

:sp filename

3. 左右分割当前打开的文件。

Ctrl+W v

4. 左右分割，并打开一个新的文件。

:vsp filename

移动光标

Vi 中的光标键是 h, j, k, l，要在各个屏间切换，只需要先按一下 Ctrl+W

1. 把光标移到**右边**的屏。

Ctrl+W l

2. 把光标移到**左边**的屏中。

Ctrl+W h

3. 把光标移到**上边**的屏中。

Ctrl+W k

4. 把光标移到**下边**的屏中。

Ctrl+W j

5. 把光标移到**下一个**的屏中。.

Ctrl+W w

移动分屏

这个功能还是使用了 Vim 的光标键，只不过都是大写。当然了，如果你的分屏很乱很复杂的话，这个功能可能会出现一些非常奇怪的症状。

1. 向右移动。

Ctrl+W L

2. 向左移动

Ctrl+W H

3. 向上移动

Ctrl+W K

4. 向下移动

Ctrl+W J

屏幕尺寸

下面是改变尺寸的一些操作，主要是高度，对于宽度你可以使用 Ctrl+W <或是>，但这可能需要最新的版本才支持。

1. 让所有的屏都有一样的高度。

Ctrl+W =

2. 增加高度。

Ctrl+W +

3. 减少高度。

Ctrl+W -

- Configuring on the fly
  - `:set` or `:set all`
- Configuring permanently
  - `~/.vimrc` or `~/.exrc`
- A few common configuration items
  - `:set number`
  - `:set autoindent`
  - `:set textwidth=65 (vim only)`
  - `:set wrapmargin=15`
  - `:set ignorecase`
- Run `:help option-list` for a complete list

Dynamic Configuration	Static Configuration
DEVICE=eth*X* HWADDR=0:02:8A:A6:30:45 BOOTPROTO=dhcp ONBOOT=yes Type=Ethernet	DEVICE=eth*X* HWADDR=0:02:8A:A6:30:45 IPADDR=192.168.0.123 NETMASK=255.255.255.0 GATEWAY=192.168.0.254 ONBOOT=yes Type=Ethernet

nameserver 最多给 3 条，多了就没有意义了

```
echo "The program name is $0"
printf "The first argument is %s and the second is %s\n" $1 $2
echo -e "\nAll command line paramenters are $*\n"

[root@station17 tmp]# ./positionalterster Red Hat Enterprise Linux
The program name is ./positionalterster
The first argument is Red and the second is Hat

All command line paramenters are Red Hat Enterprise Linux
```

```
[root@station17 tmp]# read -p "Enter a filename:"  FILE
Enter a filename:testfile
[root@station17 tmp]# echo $FILE
testfile
[root@station17 tmp]#
```

- View Process information with **ps**
  - Shows processes from the current terminal by default
  - **a** includes processes on all terminals
  - **x** includes processes not attached to terminals
  - **u** prints process owner information
  - **f** prints process parentage
  - **o PROPERTY1,PROPERTY2,...** prints custom information:
    - **pid**, **comm**, **%cpu**, **%mem**, **state**, **tty**, **euser**, **ruser**, etc.
- Example:
  - `ps axo pid,%cpu,comm`

?号表示不属于任何终端，f 选项表示打印出进程的父子关系

```
root      3318  0.0  0.0   1820   612 ?        S    09:02   0:00 /sbin/pam_times
root      3327  0.0  1.0  41640 22596 ?        S    09:02   0:00 /usr/bin/python
root      3338  0.0  0.0   2452   836 ?        S    09:02   0:00 /usr/libexec/ma
```

- Most flexible: **ps** ***options*** | ***other commands***

```
ps axo comm,tty | grep ttyS0
```

- By predefined patterns: **pgrep**

```
$ pgrep -U root

$ pgrep -G student
```

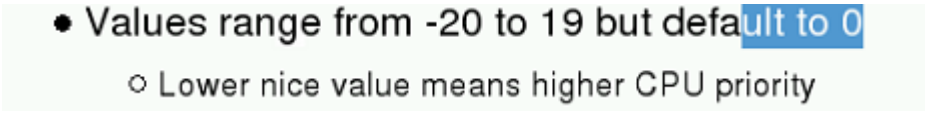
- By exact program name: **pidof**

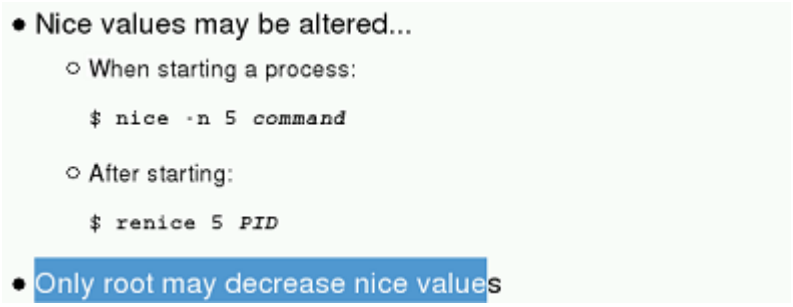
```
$ pidof bash
```

```
[root@station17 ~]# pidof httpd
3512 3511 3510 3509 3508 3507 3506 3505 3503
```

pidof 查看某个程序调用的进程

kill、pkill、killall

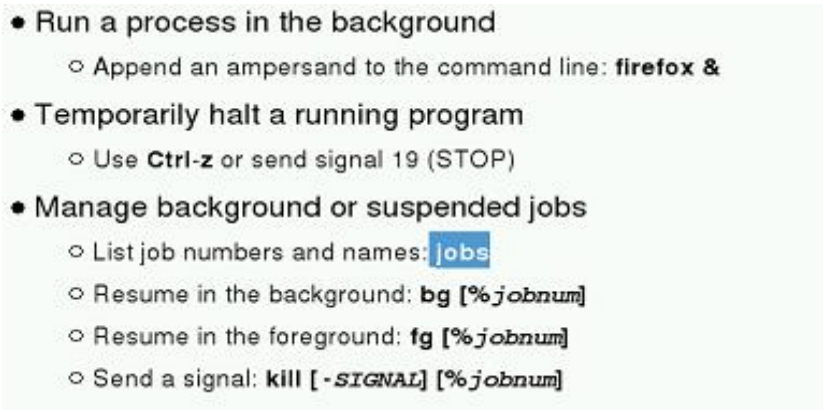




优先级调整，root 用户能调低优先级数，优先级数越小，则优先级越高

```
[root@station17 tmp]# cp -a /usr/ . &
[1] 4507
[root@station17 tmp]# jobs
[1]+  Running                 cp -i -a /usr/ . &
[root@station17 tmp]# fg
cp -i -a /usr/ .

[1]+  Stopped                 cp -i -a /usr/ .
[root@station17 tmp]# jobs
[1]+  Stopped                 cp -i -a /usr/ .
[root@station17 tmp]#
```




```
[root@station17 tmp]# jobs
[1]-  Stopped                 cp -i -a /usr/ .
[2]+  Stopped                 cp -i -a /etc/ .
[root@station17 tmp]# bg %1
[1]- cp -i -a /usr/ . &
[root@station17 tmp]# jobs
[1]-  Running                 cp -i -a /usr/ . &
[2]+  Stopped                 cp -i -a /etc/ .
[root@station17 tmp]# fg
cp -i -a /etc/ .
[root@station17 tmp]#
```

bg 是继续工作在后台，fg 则为前台
man 7 signal 查看相关信号的帮助信息

计划任务：

- One-time jobs use **at**, recurring jobs use **crontab**

Create	**at** *time*	**crontab -e**
List	**at -l**	**crontab -l**
Details	**at -c** *jobnum*	N/A
Remove	**at -d** *jobnum*	**crontab -r**
Edit	N/A	**crontab -e**

- Non-redirected output is mailed to the user
- *root* can modify jobs for other users

at 是执行一次，crontab 周期性执行

```
[root@station17 tmp]# at 23:50
at> sync
at> /sbin/shutdown -h now
at> <EOT>
job 1 at 2009-03-31 23:50
[root@station17 tmp]# jobs
[root@station17 tmp]# at -l
1       2009-03-31 23:50 a root
[root@station17 tmp]#
```

使用 crtl+d 结束 at 命令

```
[root@localhost ~]# cat /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
[root@localhost ~]#
```

/etc/crontab crontab 全局配置文件，run-parts 执行某个目录下的所有脚本

Grouping Commands

- Two ways to group commands:
  - Compound: **date; who | wc -l**
    - Commands run back-to-back
  - Subshell: **(date; who | wc -l) >> /tmp/trace**
    - All output is sent to a single STDOUT and STDERR

- Commands can be run conditionally based on exit status
  - **&&** represents conditional AND THEN
  - || represents conditional OR ELSE
- Examples:

```
$ grep -q no_such_user /etc/passwd || echo 'No such user'
No such user

$ ping -c1 -W2 station1 &> /dev/null  \
>    && echo "station1 is up"         \
>    || (echo 'station1 is unreachable'; exit 1)
station1 is up
```

&&前面的命令执行成功才执行后面的命令，||前面的命令执行失败才执行后面的命令

test 测试命令：

- Evaluates boolean statements for use in conditional execution
  - Returns 0 for true
  - Returns 1 for false
- Examples in long form:

```
$ test "$A"  =  "$B" && echo "Strings are equal"
$ test "$A" -eq "$B" && echo "Integers are equal"
```

- Examples in shorthand notation:

```
$ [ "$A"  =  "$B" ] && echo "Strings are equal"
$ [ "$A" -eq "$B" ] && echo "Integers are equal"
```

- File tests:
  - **-f** tests to see if a file exists and is a regular file
  - **-d** tests to see if a file exists and is a directory
  - **-x** tests to see if a file exists and is executable

```
[ -f ~/lib/functions ] && source ~/lib/functions
```

if 条件语句：

Scripting: if Statements

- Execute instructions based on the exit status of a command

```
if ping -c1 -w2 station1 &> /dev/null; then
    echo 'Station1 is UP'
elif grep "station1" ~/maintenance.txt &> /dev/null; then
    echo 'Station1 is undergoing maintenance'
else
    echo 'Station1 is unexpectedly DOWN!'
    exit 1
fi
```

单引号所有都不转义（是什么就是什么），双引号有些特殊的会转义

- Backslash (\ ) makes the next character literal

```
$ echo Your cost: \$5.00
Your cost: $5.00
```

- Quoting prevents expansion
  - Single quotes (') inhibit all expansion
  - Double quotes (") inhibit all expansion, except:
    - $ (dollar sign) - variable expansion
    - ` (backquotes) - command substitution
    - \ (backslash) - single character inhibition
    - ! (exclamation point) - history substitution

使某个文件立即生效：.或者 **source**

```
[root@station17 ~]# . /etc/profile
[root@station17 ~]# source /etc/profile
```

```
[root@station28 ~]# locate -n 5 passwd
/etc/passwd
/etc/passwd-
/etc/pam.d/passwd
/etc/security/opasswd
/lib/security/pam_passwdqc.so
```

-n 5 只搜索显示前 5 个

```
[root@station28 tmp]# ll
total 12
-rw-r--r-- 1 root root 0 Apr  1 09:43 test1
-rw-rw-rw- 1 root root 0 Apr  1 09:43 test2
-rw-rw-r-- 1 root root 0 Apr  1 09:43 test3
[root@station28 tmp]# find -perm +222
.
./test3
./test2
./test1
[root@station28 tmp]# find -perm -222
.
./test2
[root@station28 tmp]#
```

+222 表示只要一个条件满足就匹配(任何一个可写即可)，-222 则需要所有条件都匹配（所有都需有可写的权限）

- `find -perm -002`
  - matches if other can write

00 则忽略，表示任意

- **find** can match by inode timestamps
  - **-atime** when file was last read
  - **-mtime** when file data last changed
  - **-ctime** when file data or metadata last changed

ctime 数据或状态改变了就会改变

- `find -name "*.conf" -exec cp {} {}.orig \;`
  - Back up configuration files from the current directory, adding a `.orig` extension
- `find /home -type d -ls`
  - Do an **ls -l** style listing of all directories in /home/

```
[root@localhost ~]# find /home/ -type d -ls
553249    8 drwxr-xr-x   5 root     root         4096 Jan 21 20:12 /home/
879625    8 drwx------   3 wu       wu           4096 Jan 21 20:13 /home/wu
```

- **links** a non-GUI web browser
  - Provided by the `elinks` rpm
  - Full support for frames and ssl
  - Examples
    - **links http://www.redhat.com**
    - **links -dump http://www.redhat.com**
    - **links -source http://www.redhat.com**

links:非图形界面浏览器 -dump 是只看字符 -source 只看源码

- Secure replacement for rcp
- Layered on top of ssh
  - **scp *source destination***
  - Remote files can be specified using:
    - **[user@]host:/path/to/file**
  - Use **-r** to enable recursion
  - Use **-p** to preserve times and permissions
  - Use **-C** to compress datastream

rsync: Efficient File Sync

- Efficiently copies files to or from remote systems
- Uses secure **ssh** connections for transport
  - **rsync *.conf barney:/home/joe/configs/**
- Faster than **scp** - copies differences in like files

rsync：异地容灾

私钥加密，公钥解密

```
[root@station28 .ssh]# ssh-copy-id -i  id_rsa.pub 192.168.0.8
10
root@192.168.0.8's password:
Now try logging into the machine, with "ssh '192.168.0.8'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

[root@station28 .ssh]#
```

无需修改指定

认证代理：

- An *authentication agent* stores decrypted private keys
  - Thus, passphrase only needs to be entered once
  - An agent is provided automatically in GNOME
  - Otherwise, run **ssh-agent bash**
- Keys are added to the agent with **ssh-add**

SSH 代理认证过程：

ssh 允许用户把密钥存储在内存中，这就是 ssh 认证代理。认证代理为用户提供了使用 RSA 密钥而不必随时键入口令字的能力。这对于不必在所有登录、X 会话或运行脚本时都要键入口令字提供便利是很有效的。

```
[root@localhost ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
97:c2:32:1d:71:d8:84:4f:32:01:ed:c1:09:48:a9:1e root@localhost.localdomain
[root@localhost ~]# ssh-copy-id -i .ssh/id_rsa.pub 127.0.0.1
15
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
RSA key fingerprint is c7:1a:ae:f1:88:6b:8c:36:c1:c6:4d:b9:f8:8e:69:bc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '127.0.0.1' (RSA) to the list of known hosts.
root@127.0.0.1's password:
Now try logging into the machine, with "ssh '127.0.0.1'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

[root@localhost ~]# _
```

其中使用 ssh-kegen 的时候给生成密钥加密，输入密码
然后使用 ssh-copy-id 传递给目标主机

```
[root@localhost ~]# ssh 127.0.0.1
Enter passphrase for key '/root/.ssh/id_rsa':
Last login: Wed Jan 25 12:19:00 2012 from localhost.localdomain
[root@localhost ~]# _
```

此时连接提醒需要密钥的密码，输入即可登录

```
Connection to 127.0.0.1 closed.
[root@localhost ~]# ssh-agent bash
[root@localhost ~]# ssh-add
Enter passphrase for /root/.ssh/id_rsa:
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
[root@localhost ~]# ssh 127.0.0.1
Last login: Wed Jan 25 12:27:41 2012 from localhost.localdomain
[root@localhost ~]# _
```

执行 ssh-agent 启用代理，使用 ssh-add 加入代理密码，完成之后即可成功登录

```
Connection to 127.0.0.1 closed.
[root@localhost ~]# ssh-add -l
2048 97:c2:32:1d:71:d8:84:4f:32:01:ed:c1:09:48:a9:1e /root/.ssh/id_rsa (RSA)
[root@localhost ~]# _
```

ssh-add -l 显示密钥信息

Xwindow:桌面环境
ssh -X 目标主机 ——》远程桌面环境

```
[root@station28 ~]# traceroute 192.168.1.254
traceroute to 192.168.1.254 (192.168.1.254), 30 hops max, 40 byte packets
 1  server1.cracker.org (192.168.1.254)  0.266 ms  0.210 ms  0.182 ms
[root@station28 ~]#
```

30 hops max：表示最多经过 30 跳下一跳地址

```
[root@localhost ~]# echo "test" > /dev/tty1
test
[root@localhost ~]# _
```

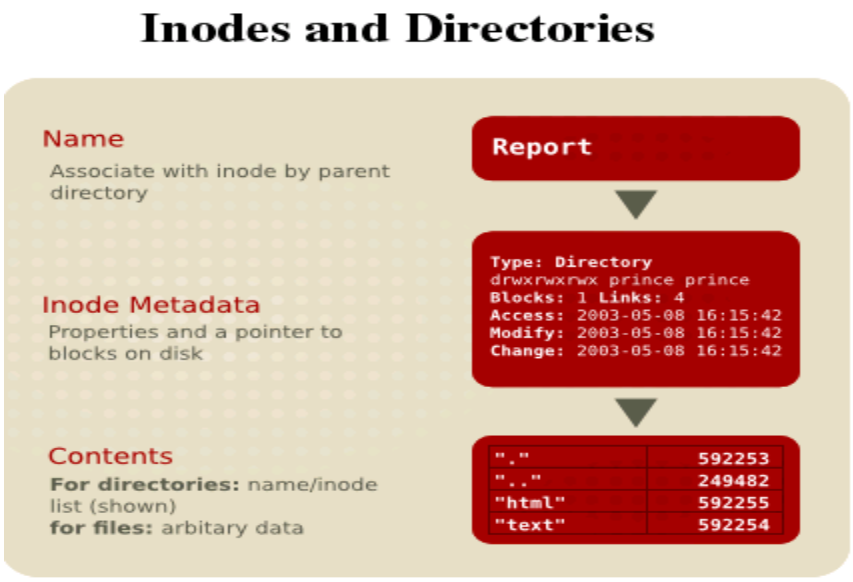
使用以上命令会在指定的某个终端显示指定的字符

w：显示当前登录的用户
lastb 显示哪些用户尝试登录，但是没有登录成功
last
lastlog

inode 表记录所有文件的属性

- An *inode table* contains a list of all files in an ext2 or ext3 filesystem
- An *inode* (index node) is an entry in the table, containing information about a file (the *metadata*), including:
  - file type, permissions, UID, GID
  - the link count (count of path names pointing to this file)
  - the file's size and various time stamps
  - pointers to the file's data blocks on disk
  - other data about the file

显示文件系统的一些详细信息：tune2fs -l /dev/sda1



7 种文件类型：

The Seven Fundamental Filetypes

ls -l symbol	File Type
-	regular file
d	directory
l	symbolic link
b	block special file
c	character special file
p	named pipe
s	socket

```
[root@station28 ~]# umount /mnt/
umount: /mnt: device is busy
umount: /mnt: device is busy
[root@station28 ~]# fuser -v /mnt/

                     USER        PID ACCESS COMMAND
/mnt/:               root       3977 ..c.. bash

[root@station28 ~]# kill -9 3977
[root@station28 ~]# umount /mnt/
```

eject 弹出光盘

RHCE033_台湾

Unit1-4

tty 查看所在终端

在文字模式下可以输入 ASCII code 作为密码 （alt+数字）

passwd –S username 查看用户账号的状态

man 5 文件 查看文件的说明

man –k 关键字 查看包含关键字的 man page

--help 、man、info、/usr/share/doc 帮助文件和命令

The bin directories:

/bin、/usr/bin、/usr/local/bin

/sbin、/usr/sbin、/usr/local/sbin

- /etc 系统配置文件（System config files）
- /tmp 临时文件（Temporary files）
- /dev 设备
- /usr Programs

➢ /var and /srv Sever data

cp -p 不改变文件的属性复制文件
cat -b 使用行号
cat -s 去除多余的空行，只保留一行

UNIX-- |
---BSD:csh->ksh、tchsh、zsh
bash 由 sh 和 tchsh 改良而来的
``倒引号里面的命令会执行 → $()

echo $[$a**$b]表示 a 的 b 次方
backslash （\）跳脱字符
quotes 引号（single quotes、doubel quotes、back quotes）

<ESC> 、alt 这两个快捷键很有作用，和点连用，把上一个命令的最后一个数自动补进来

set -o 切换指令的编辑模式:
如果想用 vi 则可以使用 set -o vi
如果不想用则 set +o vi

gnome terminal 快捷键：
ctrl shift t: 打开一个新的 terminal 标签
ctrl PgUp（PgDown）：切换标签
alt “标签号 n”： 切换到某个标签
ctrl shift c（v）：复制、粘贴
ctrl shift w：关闭标签

unit5-7

标准输入 0：Standard input
标准输出 1：Standard output
标准错误 2:Standard erro

```
[root@localhost ~]# ll /dev/std*
lrwxrwxrwx 1 root root 15 07-30 20:02 /dev/stderr -> /proc/self/fd/2
lrwxrwxrwx 1 root root 15 07-30 20:02 /dev/stdin -> /proc/self/fd/0
lrwxrwxrwx 1 root root 15 07-30 20:02 /dev/stdout -> /proc/self/fd/1
```

```
[root@localhost ~]# find /etc -name passwd >alloutput 2>&1
[root@localhost ~]# find /etc -name passwd &>alloutput
```

上面两条命令都可以把错误和正确的都输入到 alloutput，但是不建议用第二种方法，因为第二种把所有的消息都输出到 alloutput 里面去了

[root@localhost ~]# tr "A-Z" "a-z" <.bash_profile
把.bash_profile 文件中的大写改成小写后输出，不改变原文件

cut 命令：
cut -f3 filename 显示 file 的第三列
cut -f2 -d， filenbame 显示以逗号为分隔的文件 file 的第三列
cut -c4-8 filename 显示文件 file 的第四个到第 8 个字

sort 排序
sort -r 反向排序
sort -f 不区分大小写
sort -u 移除重复的行
sort -t 间隔符号
sort -n 以数字作排列
sort -k 按照哪一栏作排列，第一栏重复，按第二栏

[root@localhost ~]# cut -f1 -d: /etc/passwd | sort -r | less
xfs
wulei
webalizer
vcsa
vboxadd
uucp
user3

mail -s “hello” username
给一个用户发送一个邮件

mail -s “hello” username < hello
把 hello 中的内容交给 mail 命令

mail 读取邮件，按 x 保存，按 q 阅读完后放到 mbox 邮件垃圾桶中

[root@localhost ~]# cat ~root/mbox 存放阅读过的邮件

xargs
[root@localhost ~]# ls *.txt
del_file.txt f1.txt f2.txt f3.txt
[root@localhost ~]# cat del_file.txt
f1.txt
f2.txt
f3.txt
[root@localhost ~]# cat del_file.txt |xargs rm -rf
[root@localhost ~]# ls *.txt

```
del_file.txt
[root@localhost ~]#
```

tee 显示程序的输出并将其复制到一个文件中

语法

tee [-a] [-i] [File ...]

描述

tee 命令读取标准输入，然后将程序的输出写到标准输出，并同时将其复制到指定的一个或多个文件。

标志

-a 将输出添加到 File 的末尾而不是覆盖写入。

-i 忽略中断。

退出状态

命令返回以下退出值：

0 标准输入被成功地复制到所有输出文件中。

>0 发生错误。

注：如果向任意成功打开的File操作数的写入不成功，写入其它成功打开的File操作数，并且标准输出会继续，但是退出值将会是>0。

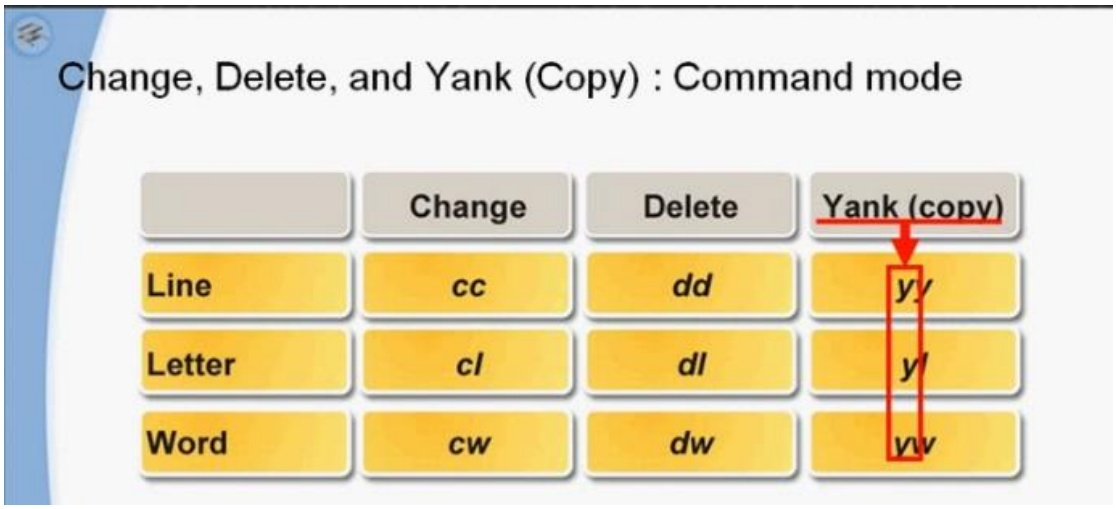
```
[root@localhost ~]# cut -f1 -d: /etc/passwd |tee cut.txt | sort | tee sort.txt 1>/dev/null
[root@localhost ~]# head -3 cut.txt
root
bin
daemon
[root@localhost ~]# head -3 sort.txt

adm
apache
[root@localhost ~]#
```

文件或目录的属主和属组只记录uid和gid,显示属主和属组的名称原因是通过/etc/passwd中uid和gid匹配的

execute表示执行的意思

w 对于文件可以修改，对于目录可以移除或新增文件或目录
x 对于文件有执行的权限，对于目录可切换进去和使用 ls 查看目录中的内容

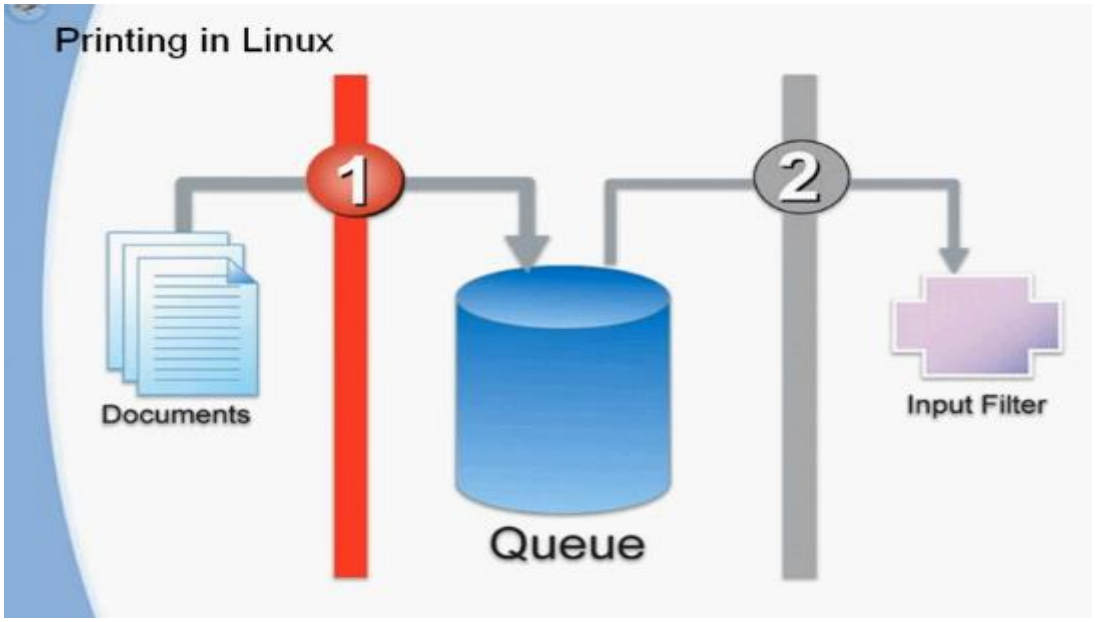


change 的修改功能是删除光标所在行，然后进入编辑模式

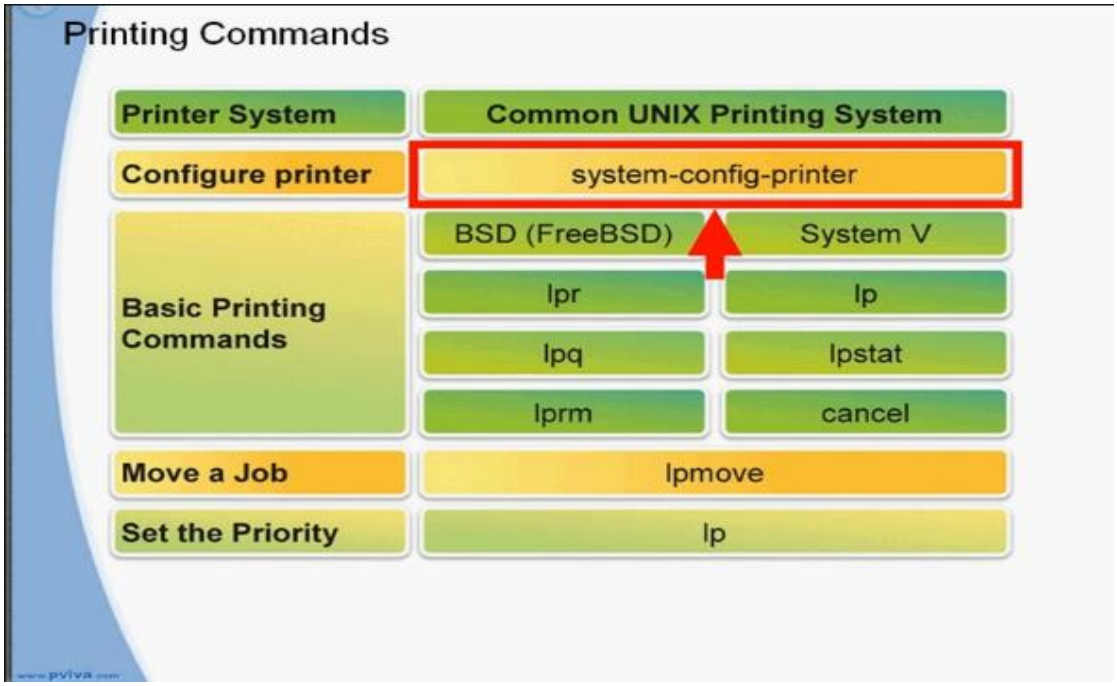
vi 编辑器中的撤销和恢复键
u 键：撤销
ctrl+r ：恢复
U：大写的 U 是取消所有的光标所在行的所有改变

命令模式：
dtc：删除从光标开始到 c 之间的字符
rc：替换光标所在字符为 c
6x：删除光标所在字符后的 6 个字符（包括光标所在字符自身的 6 个字符）

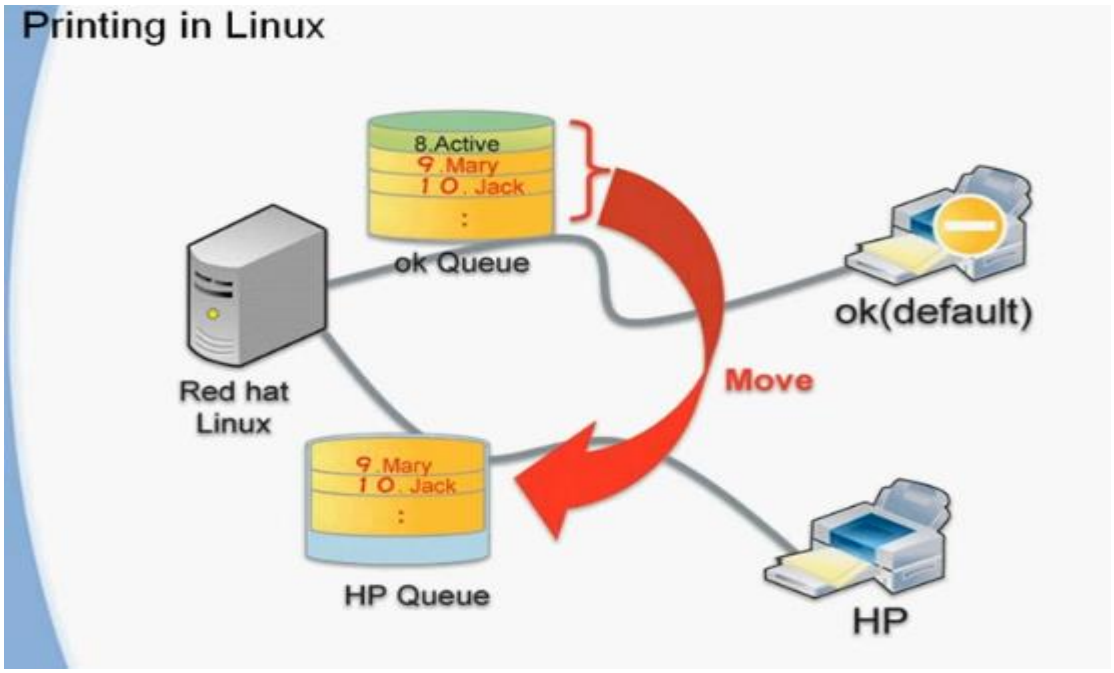
Print in Linux Network print local print



文档传到 Queue 中，在 Queue 里排队，等待打印，先进先出
reject ok 关闭第一通道 accept ok 为打开第一道
disable ok 关闭第二通道，/usr/bin/enable ok 打开第二个通道
enable 加路径是因为直接运行 enable 是其它的命令



打印的一些命令



使用 lpmove ok-9 hp

插队：lpq 查看

```
[root@localhost ~]# lpq
ok is ready and printing
Rank    Owner   Job     File(s)                         Total Size
active  root    8       README                          13312 bytes
1st     root    10      inittab                         2048 bytes
2nd     root    11      passwd.test                     2048 bytes
3rd     root    12      passwd.test                     2048 bytes
4th     root    13      README                          13312 bytes
5th     root    14      passwd.test                     2048 bytes
6th     root    15      testvi                          1024 bytes
7th     root    16      testvi                          1024 bytes
8th     root    17      boss.file                       13312 bytes
[root@localhost ~]#
```

lp -i17 -q51（默认的权限是 50）

打印机识别文档是因为打印机的驱动
PostScript 最初的文件格式，这种文档可以不用安装打印机驱动就可被识别
enscript filename -p filename.ps
把纯文档转化为 ps 格式

ps2pdf filename.ps > filename.pdf
把 ps 文档转化为 pdf 格式

pdf2ps
把 pdf 文档转化为 ps 格式

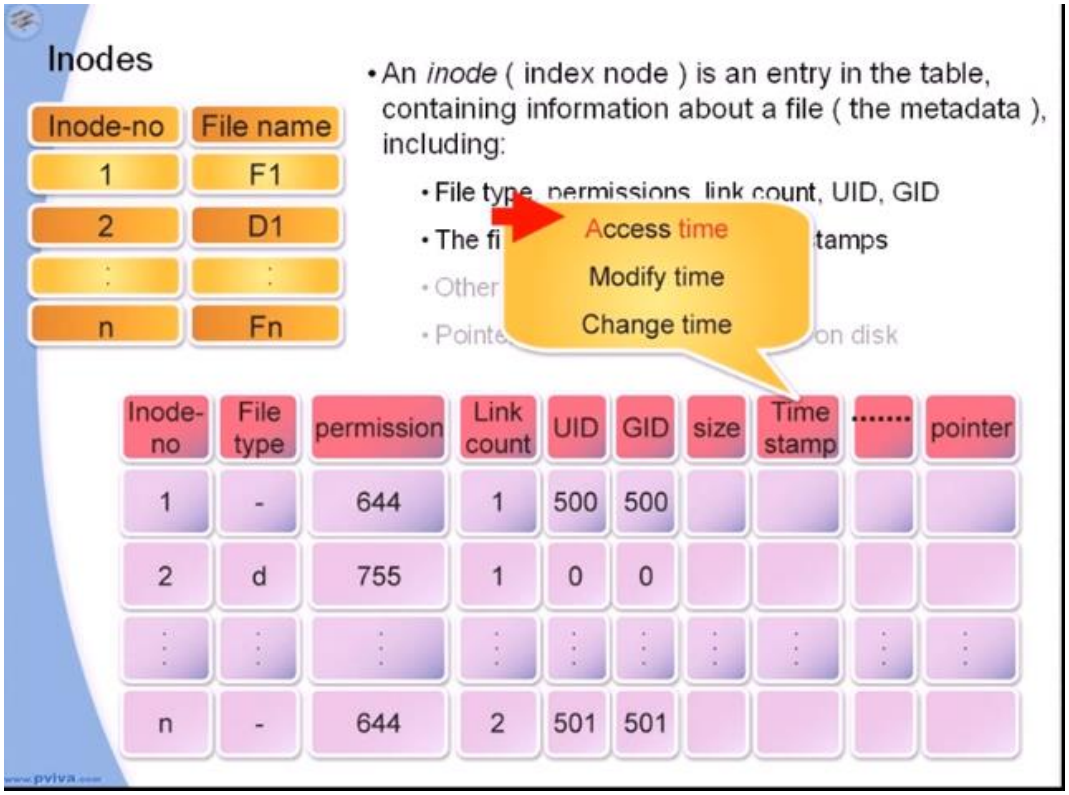
mpage 可以将多页压缩打印
mpage -4 filename.ps > mpage.filename.ps

unit8-12

默认一个 block 占 4k，一个 block 最多存放一个文件

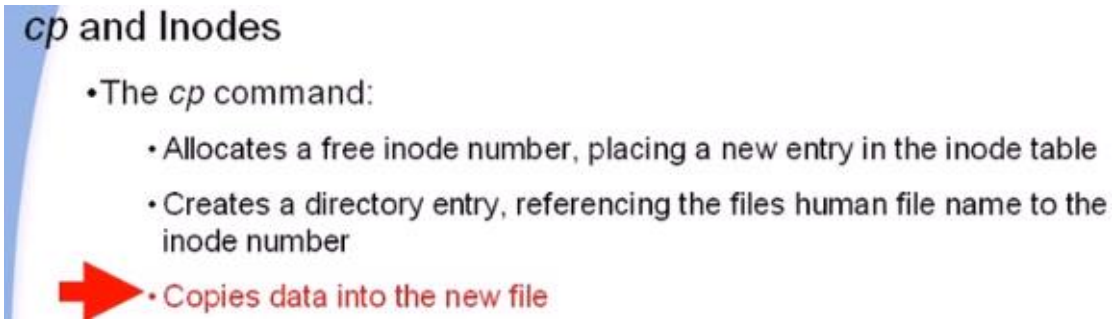
iso9660（typically userd for CD）

inode 包含的内容：

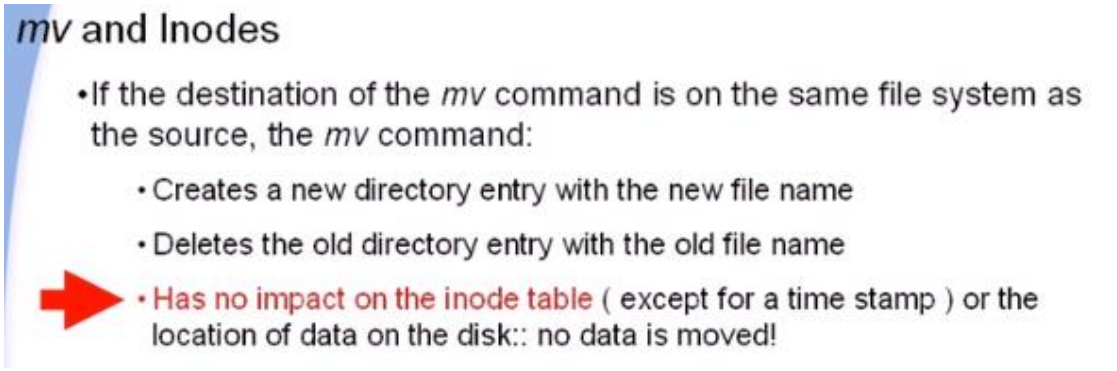


ls -i filename 查看文件或目录的 inode 号

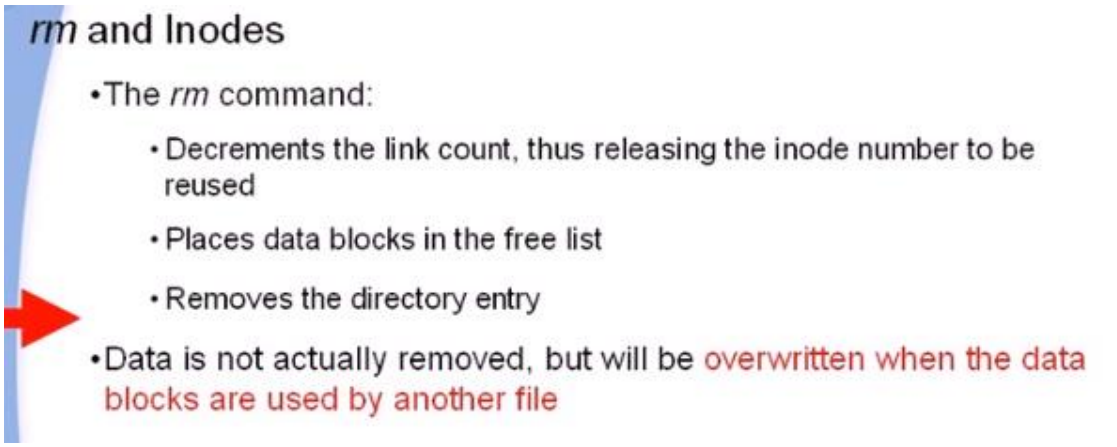
cp 和 inodes 的关系：



mv 和 inodes 关系：（数据不移动）



rm 和 inode 的关系：（当 rm 执行后，其实不是真的删除，只是当有文件存储的时候会被覆盖）



软链接占用一个 inode

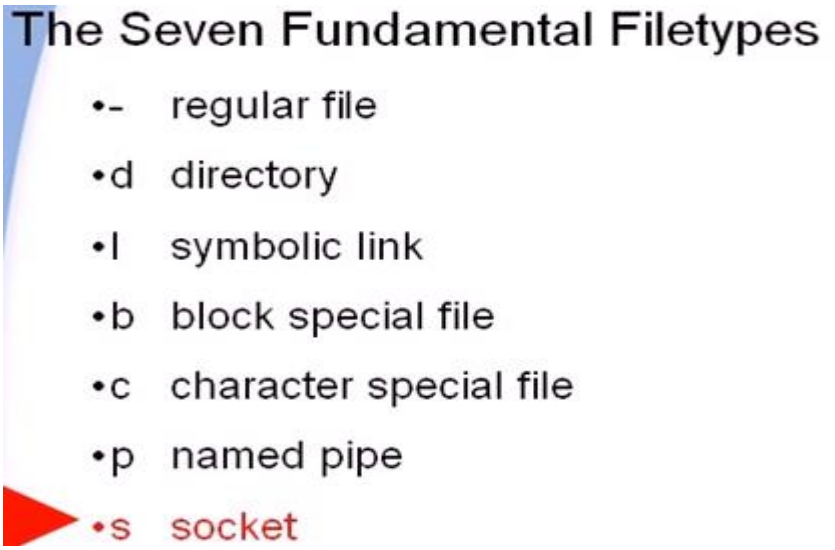
```
[root@localhost ~]# ll -i passwd*
1212454 -rw-r--r-- 1 root  root  2141 07-31 14:53 passwd
[root@localhost ~]# ls -il lnpasswd
1212456 lrwxrwxrwx 1 root root 6 07-31 16:46 lnpasswd -> passwd
```

硬链接不占用 inode（且不能用于目录）,硬链接只是一个指向 inode 的指针

```
[root@localhost ~]# ll -i passwd
1212454 -rw-r--r-- 2 root root 2141 07-31 14:53 passwd
```

```
[root@localhost ~]# ll -i lnpasswd1
1212454 -rw-r--r-- 2 root root 2141 07-31 14:53 lnpasswd1
```

cp -s 表示建立文件软链接
cp -l 表示建立文件硬链接



eject 弹出 cd

fdformat 低级格式化，危险！很少使用

tar -M 表示分片处理打包的档案

linux 下有两种变量：自定义变量和环境变量

set 可以查看所有的变量，env 则只能显示系统中的环境变量

set -o noclobber:设置之后文件中的内容部能被覆盖
$ set -o noclobber // 使 noclobber 变量开
$ set o noclobber // 使 noclobber 变量关

```
[wl@localhost ~]$ ll > ls.txt
[wl@localhost ~]$ set -o noclobber
[wl@localhost ~]$ ll > ls.txt
-bash: ls.txt: cannot overwrite existing file
[wl@localhost ~]$
```
当取消的时候，要使该命令生效需要 logout 一下，不然的话 noclobber 作用依然存在

USER 登录时执行的脚本：
1、/etc/profile(/etc/profile.d)
[root@localhost ~]# vi /etc/profile

```
for i in /etc/profile.d/*.sh ; do
    if [ -r "$i" ]; then
        . $i
    fi
done
```

2、~/.bash_profile(~/.bashrc(/etc/bashrc))

```
[root@localhost ~]# cat ~/.bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

[root@localhost ~]# cat ~/.bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
```

在以上提到的脚本中添加各个文件的路径，执行的时候就会显示路径
echo 路径

```
[wl@localhost ~]$ su - root
口令：
/etc/profile
/root/.bash_profile
/root/.bashrc
/etc/bashrc
[root@localhost ~]#
```

USER 退出时执行的脚本：
1、~/.bashrc
2、/etc/bashrc
3、/etc/profile.d
如果想要验证的话，可以使用上面的方法，在脚本中添加路径

/etc/profile:
使用者登录第一个执行的脚本
退出登录不执行该脚本
脚本中定义了用户的预设路径，登录使用者，LOGNAME，MAIL，HOSTNAME,HOSTSIZE

/etc/profile.d

包含应用程序所需要的脚本
/etc/profile 有个 for-loop 来呼叫其中的脚本

~/.bashrc_profile 和~/.bashrc
用户的自定义的设置、别名

~/.bash_logout
使用者登出时执行这个程序

root 用户可以设定不安全（6 个字以下）的密码，而其它用户不可以

VI 进阶
z（ENTER）把所在行换到第一行
z-把所在的行换到最后一行

Filtering:
!!ls -l
!!date
!}sort
!}fmt -w60

s/hello/hi/g 第一行中的 hello 取代为 hi
1,10s/hello/hi/g 第一行到 10 行中 hello 取代为 hi
1,$s/hello/hi/g 第一行到最后一行中 hello 取代为 hi
%s/hello/hi/g 同上一个一样的，第一行到最后一行中 hello 取代为 hi
.,.+10s/hello/hi/g 光标所在行下的 10 行（包括当前行）中 hello 取代为 hi
.,.-10s/hello/hi/g 反向取代
其中 s 是 sed 的意思

:r file1
:1,20w file2
:1,$w file3
:1,20w>>file4
其中 w 表示 write

:set all 可以查看一些设置

sdiff 与 diff

aspell chell 拼字检查（/usr/share/dict/words）

RHCE133

软件安装

yum localinstall

- **yum install** *package...*
- **yum localinstall** *rpmfile...*
- **yum groupinstall** *packagegroup...*
- **yum remove** *package...*
- **yum update** *[package...]*

```
[root@station36 tmp]# rpm2cpio initscripts-8.45.17.EL-1.i386.rpm |cpio -imd
8976 blocks
[root@station36 tmp]# ls
bin  etc  initscripts-8.45.17.EL-1.i386.rpm  lib  sbin  usr  var
```

rpm 包简单来说就是一个脚本

```
[root@station36 ~]# rpm -q  --scripts vsftpd
postinstall scriptlet (using /bin/sh):
/sbin/chkconfig --add vsftpd
#/usr/sbin/usermod -d /var/ftp ftp >/dev/null 2>&1 || :
preuninstall scriptlet (using /bin/sh):
if [ $1 = 0 ]; then
 /sbin/service vsftpd stop > /dev/null 2>&1
 /sbin/chkconfig --del vsftpd
fi
[root@station36 ~]#
```

rpm -F 软件 ——》有该软件则更新，无则不改动

rpm -U 软件 ——》有该软件则更新，无则安装

```
[root@station36 ~]# rpm -ivh --nodeps dovecot-1.0-1.2.rc15.el5.i386.rpm
Preparing...                ########################################### [100%]
   1:dovecot                ########################################### [100%]
[root@station36 ~]#
```

--nodeps 不检查包的依赖关系

--oldpackage：相当于回滚，不破坏原配置信息回滚软件的上一个版本

```
[root@station36 ~]# rpm -ivh http://192.168.0.254/pub/Server/vsftpd-2.0.5-10.el5.i386.rpm
Retrieving http://192.168.0.254/pub/Server/vsftpd-2.0.5-10.el5.i386.rpm
Preparing...                ########################################### [100%]
   1:vsftpd                 ########################################### [100%]
[root@station36 ~]#
```

rpm 支持 http 和 ftp

rpm -qa:查询已经安装软件包

yum list installed

```
[root@station36 ~]# rpm -qa  |grep vsftpd
vsftpd-2.0.5-10.el5
[root@station36 ~]# rpm -q vsftpd
vsftpd-2.0.5-10.el5
```

解决一个文件损坏的方法：

rpm -qf 文件名——》某个文件对应的软件包

```
[root@station36 ~]# rpm -qf /etc/inittab
initscripts-8.45.17.EL-1
[root@station36 ~]# rm /etc/inittab
rm: remove regular file `/etc/inittab'? y
[root@station36 ~]# vim /etc/inittab
[root@station36 ~]# touch /etc/inittab
[root@station36 ~]# rpm -qf /etc/inittab
initscripts-8.45.17.EL-1
[root@station36 ~]# cd /tmp/
[root@station36 tmp]# ls
bin  etc  gconfd-root  initscripts-8.45.17.EL-1.i386.rpm  lib  orbit-root  sbin  scim-bridge-0.3.0.lockfile-0@localhost:0.0  usr  var
[root@station36 tmp]# rm [^init]* -rf
[root@station36 tmp]# ls
initscripts-8.45.17.EL-1.i386.rpm
[root@station36 tmp]# rpm2cpio initscripts-8.45.17.EL-1.i386.rpm |cpio -imd
8976 blocks
[root@station36 tmp]# ls
bin  etc  initscripts-8.45.17.EL-1.i386.rpm  lib  sbin  usr  var
[root@station36 tmp]# cd etc/
[root@station36 etc]# ls
adjtime       inittab  profile.d  rc0.d  rc2.d  rc4.d  rc6.d  rc.local    rwtab    sysconfig    udev
initlog.conf  ppp      rc         rc1.d  rc3.d  rc5.d  rc.d   rc.sysinit  rwtab.d  sysctl.conf  X11
[root@station36 etc]# cp inittab /etc/
cp: overwrite `/etc/inittab'? y
[root@station36 etc]# vim /etc/intt
[root@station36 etc]# vim /etc/init
init.d/       initlog.conf  inittab
[root@station36 etc]# vim /etc/inittab
[root@station36 etc]#
```

-qpi 某个软件包的详细信息（-qi 则是已安装的软件信息）：

```
[root@station36 ~]# rpm -qpi vsftpd-2.0.5-10.el5.i386.rpm
Name        : vsftpd                       Relocations: (not relocatable)
Version     : 2.0.5                            Vendor: Red Hat, Inc.
Release     : 10.el5                       Build Date: Thu 18 Jan 2007 02:25:23 AM CST
Install Date: (not installed)              Build Host: hs20-bc2-4.build.redhat.com
Group       : System Environment/Daemons   Source RPM: vsftpd-2.0.5-10.el5.src.rpm
Size        : 289641                          License: GPL
Signature   : DSA/SHA1, Thu 18 Jan 2007 04:41:43 AM CST, Key ID 5326810137017186
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
URL         : http://vsftpd.beasts.org/
Summary     : vsftpd - Very Secure Ftp Daemon
Description :
vsftpd is a Very Secure FTP daemon. It was written completely from
scratch.
[root@station36 ~]#
```

```
[root@station36 ~]# rpm -qc  vsftpd
/etc/logrotate.d/vsftpd.log
/etc/pam.d/vsftpd
/etc/vsftpd/ftpusers
/etc/vsftpd/user_list
/etc/vsftpd/vsftpd.conf
/etc/vsftpd/vsftpd_conf_migrate.sh
[root@station36 ~]# rpm -q --scripts vsftpd
postinstall scriptlet (using /bin/sh):
/sbin/chkconfig --add vsftpd
#/usr/sbin/usermod -d /var/ftp ftp >/dev/null 2>&1 || :
preuninstall scriptlet (using /bin/sh):
if [ $1 = 0 ]; then
 /sbin/service vsftpd stop > /dev/null 2>&1
 /sbin/chkconfig --del vsftpd
fi
[root@station36 ~]#
```

语法：

rpm -q what_packages what_information

安装的软件包选项：

rpm -qa 命令会列出所有已安装的软件包

rpm -qf filename 命令显示拥有该文件的软件包

rpm -qi package_name 显示一般信息

rpm -ql package_name 列出软件包中所有文件的名称

查询（未安装）软件包的选项：

rpm -qip package_file.i386.rpm

rpm -qlp package_file.i686.rpm

```
[root@station36 ~]# rpm -V vsftpd
[root@station36 ~]# vim /etc/vsftpd/vsftpd.conf
[root@station36 ~]# rpm -V vsftpd
S.5....T c /etc/vsftpd/vsftpd.conf
[root@station36 ~]#
```

-V 校验

man rpm：

```
    S file Size differs
    M Mode differs (includes permissions and file type)
    5 MD5 sum differs
    D Device major/minor number mismatch
    L readLink(2) path mismatch
    U User ownership differs
    G Group ownership differs
    T mTime differs
```

```
[root@station36 ~]# rpm -Vp vsftpd-2.0.5-10.el5.i386.rpm
S.5....T c /etc/vsftpd/vsftpd.conf
```

```
[root@station36 ~]# cd /etc/pki/rpm-gpg/
[root@station36 rpm-gpg]# ls
RPM-GPG-KEY-fedora              RPM-GPG-KEY-redhat-former
RPM-GPG-KEY-fedora-test         RPM-GPG-KEY-redhat-release
RPM-GPG-KEY-redhat-auxiliary    RPM-GPG-KEY-redhat-rhx
RPM-GPG-KEY-redhat-beta
[root@station36 rpm-gpg]# rpm --import RPM-GPG-KEY-redhat-release
```

——》导入 key

检测软件是否经过数字签名 OK

```
[root@station36 ~]# rpm -K vsftpd-2.0.5-10.el5.i386.rpm
vsftpd-2.0.5-10.el5.i386.rpm: (sha1) dsa sha1 md5 gpg OK
[root@station36 ~]# echo "123456" >> vsftpd-2.0.5-10.el5.i386.rpm
[root@station36 ~]# rpm -K vsftpd-2.0.5-10.el5.i386.rpm
vsftpd-2.0.5-10.el5.i386.rpm: (sha1) dsa sha1 MD5 GPG NOT OK
[root@station36 ~]#
```

```
[root@station36 ~]# md5sum  dovecot-1.0-1.2.rc15.el5.i386.rpm
ce3409d3a3318509d2d0769008c82d16  dovecot-1.0-1.2.rc15.el5.i386.rpm
[root@station36 ~]#
```

查看某个软件 md5 值

Updating a Kernel RPM

- Make sure to install kernel updates
  - **yum** handles this transparently with either **update** or **install**
  - Do not use **rpm -U** or **rpm -F** ! Use **rpm -i** !
- Updating a kernel
  - **yum update kernel**
  - Boot new kernel to test
  - Revert to old kernel if a problem arises
  - **yum remove kernel-*oldversion*** if no problems

rpm -U 或-F 会破坏原内核,所以不使用，而是使用直接安装，这样的话会保留原来的内核

系统启动流程：

服务器启动系统比较慢是因为系统对服务器硬件要求比较高

系统加载的模块存放在 initrd.---.img 文件中：

```
[root@station33 tmp]# mkinitrd --with=raid1 /tmp/initrd-$(uname -r).img $(uname
-r)
WARNING: using /tmp for temporary files
[root@station33 tmp]# ls
initrd-2.6.18-164.el5.img
[root@station33 tmp]# zcat initrd-2.6.18-164.el5.img |cpio -imd
11497 blocks
[root@station33 tmp]# ls
bin  dev  etc  init  initrd-2.6.18-164.el5.img  lib  proc  sbin  sys  sysroot
[root@station33 tmp]# cd lib/
[root@station33 lib]# ls
ahci.ko          dm-mirror.ko       dm-zero.ko   libata.ko    uhci-hcd.ko
ata_piix.ko      dm-mod.ko          ehci-hcd.ko  ohci-hcd.ko
dm-log.ko        dm-raid45.ko       ext3.ko      raid1.ko
dm-mem-cache.ko  dm-region_hash.ko  firmware     scsi_mod.ko
dm-message.ko    dm-snapshot.ko     jbd.ko       sd_mod.ko
[root@station33 lib]#
```

[root@localhost ~]# grub-md5-crypt
Password:
Retype password:
1fLacV0$FPYtx9QyHbQApJD1eECah/
[root@localhost ~]#

```
default=0
timeout=5
splashimage=(hd0,6)/grub/splash.xpm.gz
hiddenmenu
password --md5 $1$lZTkH/$vHLRDHdwMOb3ggM9FBksB.
title Red Hat Enterprise Linux Server (2.6.18-164.el5)
password --md5 $1$lZTkH/$vHLRDHdwMOb3ggM9FBksB.
        root (hd0,6)
        kernel /vmlinuz-2.6.18-164.el5 ro root=LABEL=/1 rhgb quiet
        initrd /initrd-2.6.18-164.el5.img
title winXP
        rootnoverify (hd0,0)
        chainloader +1
~
```

RHCE133 台湾

unit1-3

从光盘启动，输入 linux askmethod 可以选择其他安装方式

Device Node

Block Devices

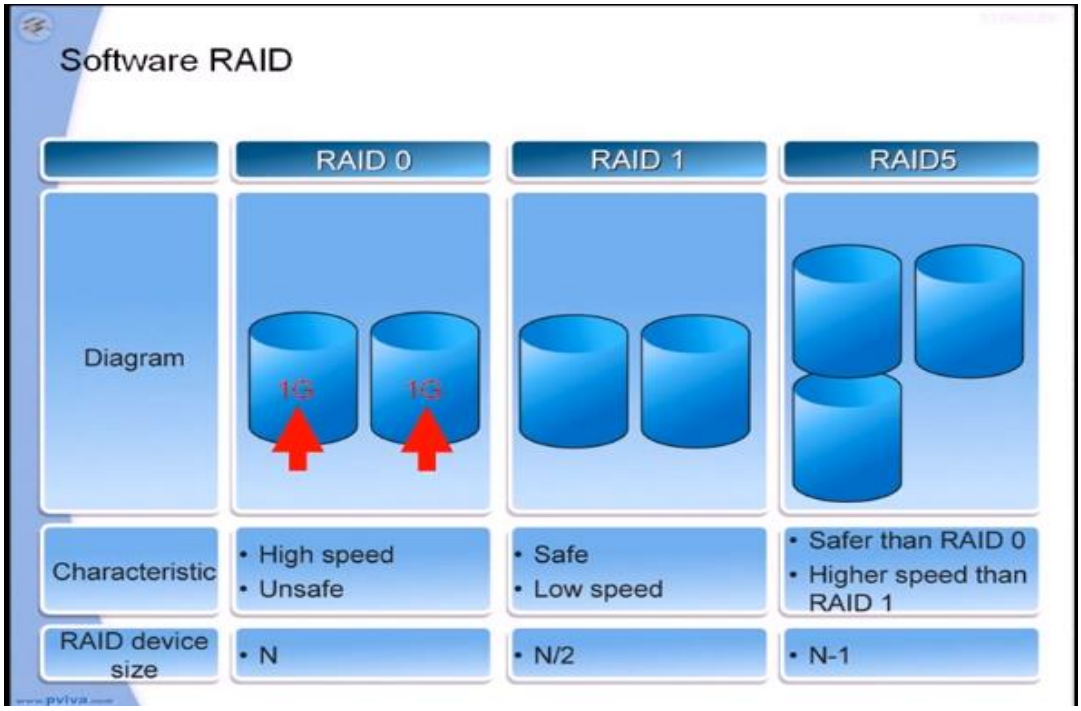
/dev/hda (IDE drives)

/dev/sda (SCSI drives)

/dev/fd0 (-standard floppy drives)

Character Devices
/dev/tty[0-6]
/dev/st0 SCSI 影碟机

	Block device	Character Device
Unit	•Block (512/1024 bytes) •One block is accessed or occurred each time	•Character (1 byte) •One character for each access
Characteristic	•Fast access •Random access	•Slow access •Sequential access
Property	b rwx r-x r--	c rwx r-x r--



Unit4 文件系统管理

IDE 最多分区：63
SCSI 最多分区：15

分区的作用：1、资料方便控制和管理
2、考虑到效能的因素
3、磁盘配额

4、资料备份等

mke2fs –N 指定 inode 的数
mke2fs –m 10 指定%10 保留给 super-user 的大小

安全：物理+本地+远程+个人
需要考虑很多的因素，特别是内部的因素

安全 → 需要知道自己的计算机有多少的服务在运行
经常要监控主机的系统，不要运行一些不受信任的软件
网络监控和分析
文件系统监控和分析
进程的监控和分析

Ip
Nmap 扫描（nmapfe 为图形界面版）→ 探测远程计算机的端口等信息
tcpdump，wireshark

netstat -tulnpa
注意：telnet 是 TCP 传输的协议，只能连接 tcp 的端口，不能连接 UDP 的服务端口

sar –u –o sar.txt 5 5 #导入到某个文件中去