Linux命令大全

# 系统管理

## 进程和作业管理

### init

切换执行等级。

init命令是Linux下的进程初始化工具，init进程是所有Linux进程的父进程，它的进程号为1。init命令是Linux操作系统中不可缺少的程序之一，init进程是Linux内核引导运行的，是系统中的第一个进程。

**语法**

init (选项) (参数)

**选项**

-b：不执行相关脚本而直接进入单用户模式；

-s：切换到单用户模式。

**参数**

运行等级：指定Linux系统要切换到的运行等级。

**实例**

几个常用的命令

查看系统进程命令： ps -ef | head

查看init的配置文件： more /etc/inittab

查看系统当前运行的级别： runlevel

**运行级别**

到底什么是运行级呢？简单的说，运行级就是操作系统当前正在运行的功能级别。这个级别从0到6 ，具有不同的功能。你也可以在/etc/inittab中查看它的英文介绍。

#0 关机（千万不能把initdefault 设置为0）

#1 单用户模式

#2 多用户，没有 NFS(和级别3相似，会停止部分服务)

#3 纯命令行模式

#4 没有用到

#5 含有图形界面模式 x11(Xwindow)

#6 重新启动（千万不要把initdefault 设置为6）

### ps

ps命令用于报告当前系统的进程状态。可以搭配kill指令随时中断、删除不必要的程序。ps命令是最基本同时也是非常强大的进程查看命令，使用该命令可以确定有哪些进程正在运行和运行的状态、进程是否结束、进程有没有僵死、哪些进程占用了过多的资源等等，总之大部分信息都是可以通过执行该命令得到的。

格式：ps [选项]

范例：

1.ps aux

查看系统中的所有进程。

## 用户和工作组管理

### useradd

useradd命令用于Linux中创建的新的系统用户。useradd可用来建立用户帐号。帐号建好之后，再用passwd设定帐号的密码．而可用userdel删除帐号。使用useradd指令所建立的帐号，实际上是保存在/etc/passwd文本文件中。

在Slackware中，adduser指令是个script程序，利用交谈的方式取得输入的用户帐号资料，然后再交由真正建立帐号的useradd命令建立新用户，如此可方便管理员建立用户帐号。在Red Hat Linux中，adduser命令则是useradd命令的符号连接，两者实际上是同一个指令。

**语法**

useradd (选项) (参数)

**选项**

-c<备注>：加上备注文字。备注文字会保存在passwd的备注栏位中；

-d<登入目录>：指定用户登入时的启始目录；

-D：变更预设值；

-e<有效期限>：指定帐号的有效期限；

-f<缓冲天数>：指定在密码过期后多少天即关闭该帐号；

-g<群组>：指定用户所属的群组；

-G<群组>：指定用户所属的附加群组；

-m：自动建立用户的登入目录；

-M：不要自动建立用户的登入目录；

-n：取消建立以用户名称为名的群组；

-r：建立系统帐号；

-s：指定用户登入后所使用的shell；

-u：指定用户id。

**参数**

用户名：要创建的用户名。

**实例**

新建用户加入组：

useradd –g sales jack –G company,employees //-g：加入主要组、-G：加入次要组

建立一个新用户账户，并设置ID：

useradd zheye -u 544

需要说明的是，设定ID值时尽量要大于500，以免冲突。因为Linux安装后会建立一些特殊用户，一般0到499之间的值留给bin、mail这样的系统账号。

### passwd

passwd命令用于设置用户的认证信息，包括用户密码、密码过期时间等。系统管理者则能用它管理系统用户的密码。只有管理者可以指定用户名称，一般用户只能变更自己的密码。

**语法**

passwd (选项) (参数)

**选项**

-d：删除密码，仅有系统管理者才能使用；

-f：强制执行；

-k：设置只有在密码过期失效后，方能更新；

-l：锁住密码；

-S：列出密码的相关信息，仅有系统管理者才能使用；

-u：解开已上锁的帐号。

**参数**

用户名：需要设置密码的用户名。

**知识扩展**

* **与用户、组账户信息相关的文件**

存放用户信息：

/etc/passwd /etc/shadow

存放组信息：

/etc/group /etc/gshadow

* **用户信息文件分析（每项用:隔开）**

例如：jack:X:503:504:::/home/jack/:/bin/bash

jack　　 //用户名

X　　 //口令、密码

503　　 //用户id（0代表root、普通新建用户从500开始）

504　　 //所在组

:　　 //描述

/home/jack/　　 //用户主目录

/bin/bash　　 //用户缺省Shell

* **组信息文件分析**

例如：jack:$!$:???:13801:0:99999:7:\*:\*:

jack　　//组名

$!$　　//被加密的口令

13801　　//创建日期与今天相隔的天数

0　　//口令最短位数

99999　　//用户口令

7　　//到7天时提醒

\*　　//禁用天数

\*　　//过期天数

**实例**

如果是普通用户执行passwd只能修改自己的密码。如果新建用户后，要为新用户创建密码，则用passwd用户名，注意要以root用户的权限来创建。

[root@localhost ~]# passwd linuxde //更改或创建linuxde用户的密码；

Changing password for user linuxde. New UNIX password: //请输入新密码；

Retype new UNIX password: //再输入一次；

passwd: all authentication tokens updated successfully. //成功；

普通用户如果想更改自己的密码，直接运行passwd即可，比如当前操作的用户是linuxde。

[linuxde@localhost ~]$ passwd

Changing password for user linuxde. //更改linuxde用户的密码；

(current) UNIX password: //请输入当前密码； 时

New UNIX password: //请输入新密码；

Retype new UNIX password: //确认新密码；

passwd: all authentication tokens updated successfully. //更改成功；

比如我们让某个用户不能修改密码，可以用-l选项来锁定：

[root@localhost ~]# passwd -l linuxde //锁定用户linuxde不能更改密码；

Locking password for user linuxde. passwd: Success //锁定成功；

[root@localhost ~]# su linuxde //通过su切换到linuxde用户；

[linuxde@localhost ~]$ passwd //linuxde来更改密码；

Changing password for user linuxde.

Changing password for linuxde (current) UNIX password: //输入linuxde的当前密码；

passwd: Authentication token manipulation error //失败，不能更改密码；

再来一例：

[root@localhost ~]# passwd -d linuxde //清除linuxde用户密码；

Removing password for user linuxde. passwd: Success //清除成功；

[root@localhost ~]# passwd -S linuxde //查询linuxde用户密码状态；

Empty password. //空密码，也就是没有密码；

注意：当我们清除一个用户的密码时，登录时就无需密码，这一点要加以注意。

### su

su命令用于切换当前用户身份到其他用户身份，变更时须输入所要变更的用户帐号与密码。

**语法**

su (选项) (参数)

**选项**

-c<指令>或--command=<指令>：执行完指定的指令后，即恢复原来的身份；

-f或——fast：适用于csh与tsch，使shell不用去读取启动文件；

-l或——login：改变身份时，也同时变更工作目录，以及HOME,SHELL,USER,logname。此外，也会变更PATH变量；

-m,-p或--preserve-environment：变更身份时，不要变更环境变量；

-s或--shell=：指定要执行的shell；

--help：显示帮助；

--version；显示版本信息。

**参数**

用户：指定要切换身份的目标用户。

**实例**

变更帐号为root并在执行ls指令后退出变回原使用者：

su -c ls root

变更帐号为root并传入-f选项给新执行的shell：

su root -f

变更帐号为test并改变工作目录至test的家目录：

su -test

## 文件系统管理

### mount

格式：mount [选项] 设备源 目标目录

范例：

1. mount /dev/cdrom /mnt

将光驱挂载到/mnt目录下

### umount

格式：mount 目标目录

范例：

1. umount /mnt （对应mount例）

取消光驱在/mnt下的挂载

## 系统关机和重启

### halt

### ctrlaltdel

### reboot

reboot命令用来重新启动正在运行的Linux操作系统。

**语法**

reboot (选项)

**选项**

-d： 重新开机时不把数据写入记录文件/var/tmp/wtmp。本参数具有“-n”参数效果；

-f： 强制重新开机，不调用shutdown指令的功能；

-i： 在重开机之前，先关闭所有网络界面；

-n： 重开机之前不检查是否有未结束的程序；

-w： 仅做测试，并不真正将系统重新开机，只会把重开机的数据写入/var/log目录下的wtmp记录文件。

**实例**

reboot //重开机。

reboot -w //做个重开机的模拟（只有纪录并不会真的重开机）。

### shutdown

shutdown命令用来系统关机命令。shutdown指令可以关闭所有程序，并依用户的需要，进行重新开机或关机的动作。

**语法**

shutdown (选项) (参数)

选项

-c： 当执行“shutdown -h 11:50”指令时，只要按+键就可以中断关机的指令；

-f： 重新启动时不执行fsck；

-F： 重新启动时执行fsck；

-h： 将系统关机；

-k： 只是送出信息给所有用户，但不会实际关机；

-n： 不调用init程序进行关机，而由shutdown自己进行；

-r： shutdown之后重新启动；

-t<秒数>： 送出警告信息和删除信息之间要延迟多少秒。

**参数**

[时间]：设置多久时间后执行shutdown指令；

[警告信息]：要传送给所有登入用户的信息。

**实例**

指定现在立即关机：

shutdown -h now

或者

shutdown now

指定5分钟后关机，同时送出警告信息给登入用户：

shutdown +5 "System will shutdown after 5 minutes"

shutdown -h now //立刻关机，其中now相当于时间为0的状态

shutdown -h 20:25 //系统在今天的20:25分关机，若在21:25才执行此命令，则隔天关机

shutdown -h +10 //系统再过十分钟后自动关机

shutdown -r now //系统立刻重启

shutdown -r +30 ‘The system will reboot’

//再过三十分钟系统会重启，并显示后面的消息给所有在线的用户

shutdown -k now ‘This system will reboot’

//仅发出警告新建参数，系统并不会关机

### poweroff

# 网络管理

## 网络测试

### netstat

netstat命令用来打印Linux中网络系统的状态信息，可让你得知整个Linux系统的网络情况。

**语法**

netstat (选项)

选项

-a或--all：显示所有连线中的Socket；

-A<网络类型>或--<网络类型>：列出该网络类型连线中的相关地址；

-c或--continuous：持续列出网络状态；

-C或--cache：显示路由器配置的快取信息；

-e或--extend：显示网络其他相关信息；

-F或--fib：显示FIB；

-g或--groups：显示多重广播功能群组组员名单；

-h或--help：在线帮助；

-i或--interfaces：显示网络界面信息表单；

-l或--listening：显示监控中的服务器的Socket；

-M或--masquerade：显示伪装的网络连线；

-n或--numeric：直接使用ip地址，而不通过域名服务器；

-N或--netlink或--symbolic：显示网络硬件外围设备的符号连接名称；

-o或--timers：显示计时器；

-p或--programs：显示正在使用Socket的程序识别码和程序名称；

-r或--route：显示Routing Table；

-s或--statistice：显示网络工作信息统计表；

-t或--tcp：显示TCP传输协议的连线状况；

-u或--udp：显示UDP传输协议的连线状况；

-v或--verbose：显示指令执行过程；

-V或--version：显示版本信息；

-w或--raw：显示RAW传输协议的连线状况；

-x或--unix：此参数的效果和指定"-A unix"参数相同；

--ip或--inet：此参数的效果和指定"-A inet"参数相同。

**实例**

* **列出所有端口 (包括监听和未监听的)**

netstat -a #列出所有端口

netstat -at #列出所有tcp端口

netstat -au #列出所有udp端口

* **列出所有处于监听状态的 Sockets**

netstat -l #只显示监听端口

netstat -lt #只列出所有监听 tcp 端口

netstat -lu #只列出所有监听 udp 端口

netstat -lx #只列出所有监听 UNIX 端口

* **显示每个协议的统计信息**

netstat -s 显示所有端口的统计信息

netstat -st 显示TCP端口的统计信息

netstat -su 显示UDP端口的统计信息

* **在netstat输出中显示 PID 和进程名称**

netstat -pt

netstat -p可以与其它开关一起使用，就可以添加“PID/进程名称”到netstat输出中，这样debugging的时候可以很方便的发现特定端口运行的程序。

* **在netstat输出中不显示主机，端口和用户名(host, port or user)**

当你不想让主机，端口和用户名显示，使用netstat -n。将会使用数字代替那些名称。同样可以加速输出，因为不用进行比对查询。

netstat -an

如果只是不想让这三个名称中的一个被显示，使用以下命令:

netsat -a --numeric-ports

netsat -a --numeric-hosts

netsat -a --numeric-users

* **持续输出netstat信息**

netstat -c #每隔一秒输出网络信息

* **显示系统不支持的地址族(Address Families)**

netstat --verbose

在输出的末尾，会有如下的信息：

netstat: no support for `AF IPX' on this system.

netstat: no support for `AF AX25' on this system.

netstat: no support for `AF X25' on this system.

netstat: no support for `AF NETROM' on this system.

* **显示核心路由信息**

netstat -r

使用netstat -rn显示数字格式，不查询主机名称。

* **找出程序运行的端口**

并不是所有的进程都能找到，没有权限的会不显示，使用 root 权限查看所有的信息。

netstat -ap | grep ssh

找出运行在指定端口的进程：

netstat -an | grep ':80'

* **显示网络接口列表**

netstat -i

显示详细信息，像是ifconfig使用netstat -ie。

**IP和TCP分析**

查看连接某服务端口最多的的IP地址：

netstat -ntu | grep :80 | awk '{print $5}' | cut -d: -f1 | awk '{++ip[$1]} END {for(i in ip) print ip[i],"\t",i}' | sort -nr

TCP各种状态列表：

netstat -nt | grep -e 127.0.0.1 -e 0.0.0.0 -e ::: -v | awk '/^tcp/ {++state[$NF]} END {for(i in state) print i,"\t",state[i]}'

查看phpcgi进程数，如果接近预设值，说明不够用，需要增加：

netstat -anpo | grep "php-cgi" | wc -l

## 网络配置

### ifconfig

ifconfig命令被用于配置和显示Linux内核中网络接口的网络参数。用ifconfig命令配置的网卡信息，在网卡重启后机器重启后，配置就不存在。要想将上述的配置信息永远的存的电脑里，那就要修改网卡的配置文件了。

**语法**

ifconfig (参数)

**参数**

add<地址>：设置网络设备IPv6的ip地址；

del<地址>：删除网络设备IPv6的IP地址；

down：关闭指定的网络设备；

<硬件地址>：设置网络设备的类型与硬件地址；

io\_addr：设置网络设备的I/O地址；

irq：设置网络设备的IRQ；

media<网络媒介类型>：设置网络设备的媒介类型；

mem\_start<内存地址>：设置网络设备在主内存所占用的起始地址；

metric<数目>：指定在计算数据包的转送次数时，所要加上的数目；

mtu<字节>：设置网络设备的MTU；

netmask<子网掩码>：设置网络设备的子网掩码；

tunnel<地址>：建立IPv4与IPv6之间的隧道通信地址；

up：启动指定的网络设备；

-broadcast<地址>：将要送往指定地址的数据包当成广播数据包来处理；

-pointopoint<地址>：与指定地址的网络设备建立直接连线，此模式具有保密功能；

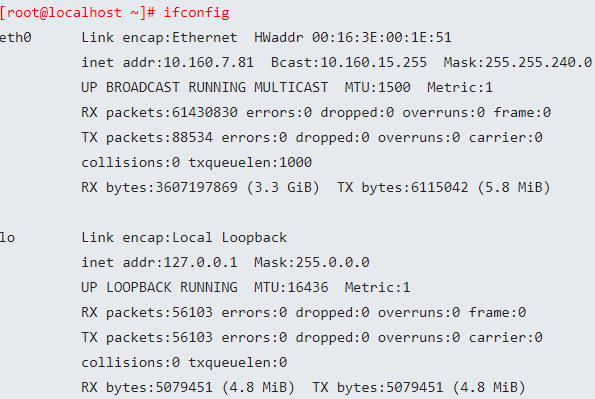
-promisc：关闭或启动指定网络设备的promiscuous模式；

IP地址：指定网络设备的IP地址；

网络设备：指定网络设备的名称。

**实例**

显示网络设备信息（激活状态的）：



**说明：**

**eth0**表示第一块网卡，其中HWaddr表示网卡的物理地址，可以看到目前这个网卡的物理地址(MAC地址）是00:16:3E:00:1E:51。 （我的centos上是ens33）

**inet addr**用来表示网卡的IP地址，此网卡的IP地址是10.160.7.81，广播地址Bcast:10.160.15.255，掩码地址Mask:255.255.240.0。

lo是表示主机的回坏地址，这个一般是用来测试一个网络程序，但又不想让局域网或外网的用户能够查看，只能在此台主机上运行和查看所用的网络接口。比如把 httpd服务器的指定到回坏地址，在浏览器输入127.0.0.1就能看到你所架WEB网站了。但只是您能看得到，局域网的其它主机或用户无从知道。

第一行：连接类型：Ethernet（以太网）HWaddr（硬件mac地址）。

第二行：网卡的IP地址、子网、掩码。

第三行：UP（代表网卡开启状态）RUNNING（代表网卡的网线被接上）MULTICAST（支持组播）MTU:1500（最大传输单元）：1500字节。

第四、五行：接收、发送数据包情况统计。

第七行：接收、发送数据字节数统计信息。

启动关闭指定网卡：

ifconfig eth0 up

ifconfig eth0 down

ifconfig eth0 up为启动网卡eth0，ifconfig eth0 down为关闭网卡eth0。ssh登陆linux服务器操作要小心，关闭了就不能开启了，除非你有多网卡。

为网卡配置和删除IPv6地址：

ifconfig eth0 add 33ffe:3240:800:1005::2/64 #为网卡eth0配置IPv6地址

ifconfig eth0 del 33ffe:3240:800:1005::2/64 #为网卡eth0删除IPv6地址

用ifconfig修改MAC地址：

ifconfig eth0 hw ether 00:AA:BB:CC:dd:EE

配置IP地址：

[root@localhost ~]# ifconfig eth0 192.168.2.10

[root@localhost ~]# ifconfig eth0 192.168.2.10 netmask 255.255.255.0

[root@localhost ~]# ifconfig eth0 192.168.2.10 netmask 255.255.255.0 broadcast 192.168.2.255

启用和关闭arp协议：

ifconfig eth0 arp #开启网卡eth0 的arp协议

ifconfig eth0 -arp #关闭网卡eth0 的arp协议

设置最大传输单元：

ifconfig eth0 mtu 1500 #设置能通过的最大数据包大小为 1500 bytes

# 软件 | 打印 | 开发 | 工具

## 常用工具命令

### date

date命令是显示或设置系统时间与日期。

很多shell脚本里面需要打印不同格式的时间或日期，以及要根据时间和日期执行操作。延时通常用于脚本执行过程中提供一段等待的时间。日期可以以多种格式去打印，也可以使用命令设置固定的格式。在类UNIX系统中，日期被存储为一个整数，其大小为自世界标准时间（UTC）1970年1月1日0时0分0秒起流逝的秒数。

**语法**

date (选项) (参数)

**选项**

-d<字符串>： 显示字符串所指的日期与时间。字符串前后必须加上双引号；

-s<字符串>： 根据字符串来设置日期与时间。字符串前后必须加上双引号；

-u： 显示GMT；

--help： 在线帮助；

--version： 显示版本信息。

**参数**

<+时间日期格式>：指定显示时使用的日期时间格式。

**日期格式字符串列表**

%H 小时，24小时制（00~23）

%I 小时，12小时制（01~12）

%k 小时，24小时制（0~23）

%l 小时，12小时制（1~12）

%M 分钟（00~59）

%p 显示出AM或PM

%r 显示时间，12小时制（hh:mm:ss %p）

%s 从1970年1月1日00:00:00到目前经历的秒数

%S 显示秒（00~59）

%T 显示时间，24小时制（hh:mm:ss）

%X 显示时间的格式（%H:%M:%S）

%Z 显示时区，日期域（CST）

%a 星期的简称（Sun~Sat）

%A 星期的全称（Sunday~Saturday）

%h,%b 月的简称（Jan~Dec）

%B 月的全称（January~December）

%c 日期和时间（Tue Nov 20 14:12:58 2012）

%d 一个月的第几天（01~31）

%x,%D 日期（mm/dd/yy）

%j 一年的第几天（001~366）

%m 月份（01~12）

%w 一个星期的第几天（0代表星期天）

%W 一年的第几个星期（00~53，星期一为第一天）

%y 年的最后两个数字（1999则是99）

**实例**

格式化输出：

date +"%Y-%m-%d"

2009-12-07

输出昨天日期：

date -d "1 day ago" +"%Y-%m-%d"

2012-11-19

2秒后输出：

date -d "2 second" +"%Y-%m-%d %H:%M.%S"

2012-11-20 14:21.31

传说中的 1234567890 秒：

date -d "1970-01-01 1234567890 seconds" +"%Y-%m-%d %H:%m:%S"

2009-02-13 23:02:30

普通转格式：

date -d "2009-12-12" +"%Y/%m/%d %H:%M.%S"

2009/12/12 00:00.00

apache格式转换：

date -d "Dec 5, 2009 12:00:37 AM" +"%Y-%m-%d %H:%M.%S"

2009-12-05 00:00.37

格式转换后时间游走：

date -d "Dec 5, 2009 12:00:37 AM 2 year ago" +"%Y-%m-%d %H:%M.%S"

2007-12-05 00:00.37

加减操作：

date +%Y%m%d //显示年月日

date -d "+1 day" +%Y%m%d //显示前一天的日期

date +%Y%m%d --date="-1 day"

date -d "-1 day" +%Y%m%d //显示后一天的日期

date -d "-1 month" +%Y%m%d //显示上一月的日期

date -d "+1 month" +%Y%m%d //显示下一月的日期

date -d "-1 year" +%Y%m%d //显示前一年的日期

date -d "+1 year" +%Y%m%d //显示下一年的日期

设定时间：

date -s //设置当前时间，只有root权限才能设置，其他只能查看

date -s 20120523 //设置成20120523，这样会把具体时间设置成空00:00:00

date -s 01:01:01 //设置具体时间，不会对日期做更改

date -s "01:01:01 2012-05-23" //这样可以设置全部时间

date -s "01:01:01 20120523" //这样可以设置全部时间

date -s "2012-05-23 01:01:01" //这样可以设置全部时间

date -s "20120523 01:01:01" //这样可以设置全部时间

有时需要检查一组命令花费的时间，举例：

#!/bin/bash

start=$(date +%s)

nmap man.linuxde.net &> /dev/null

end=$(date +%s)

difference=$(( end - start ))

echo $difference seconds.

### cal

cal命令用于显示当前日历，或者指定日期的日历。

**语法**

cal (选项) (参数)

**选项**

-l： 显示单月输出；

-3： 显示临近三个月的日历；

-s： 将星期日作为月的第一天；

-m： 将星期一作为月的第一天；

-j： 显示“julian”日期；

-y： 显示当前年的日历。

参数

月： 指定月份；

年： 指定年份。

实例

单独执行cal命令会打印出日历：

[root@localhost ShellFile]# cal

十月 2017

日 一 二 三 四 五 六

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 31

[root@localhost ShellFile]# cal -j

十月 2017

日 一 二 三 四 五 六

274 275 276 277 278 279 280

281 282 283 284 285 286 287

288 289 290 291 292 293 294

295 296 297 298 299 300 301

302 303 304

[root@localhost ShellFile]# cal -3

九月 2017 十月 2017 十一月 2017

日 一 二 三 四 五 六 日 一 二 三 四 五 六 日 一 二 三 四 五 六

1 2 1 2 3 4 5 6 7 1 2 3 4

3 4 5 6 7 8 9 8 9 10 11 12 13 14 5 6 7 8 9 10 11

10 11 12 13 14 15 16 15 16 17 18 19 20 21 12 13 14 15 16 17 18

17 18 19 20 21 22 23 22 23 24 25 26 27 28 19 20 21 22 23 24 25

24 25 26 27 28 29 30 29 30 31 26 27 28 29 30

### bc

bc命令是一种支持任意精度的交互执行的计算器语言。bash内置了对整数四则运算的支持，但是并不支持浮点运算，而bc命令可以很方便的进行浮点运算，当然整数运算也不再话下。

**语法**

bc (选项) (参数)

**选项**

-i：强制进入交互式模式；

-l：定义使用的标准数学库；

-w：对POSIX bc的扩展给出警告信息；

-q：不打印正常的GNU bc环境信息；

-v：显示指令版本信息；

-h：显示指令的帮助信息。

**参数**

文件：指定包含计算任务的文件。

**实例**

直接输入bc命令进行运算

[root@localhost ShellFile]# bc

bc 1.06.95

Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.

This is free software with ABSOLUTELY NO WARRANTY.

For details type `warranty'.

**2/3**

0

**scale=3**

**2/3**

.666

**1+2**

3

**2^3**

8

**quit //输入quit退出**

[root@localhost ShellFile]#

算术操作高级运算bc命令它可以执行浮点运算和一些高级函数：

echo "1.212\*3" | bc 3.636

设定小数精度（数值范围）

echo "scale=2;3/8" | bc

0.37

参数scale=2是将bc输出结果的小数位设置为2位。

进制转换

#!/bin/bash

abc=192

echo "obase=2;$abc" | bc

执行结果为：11000000，这是用bc将十进制转换成二进制。

#!/bin/bash

abc=11000000

echo "obase=10;ibase=2;$abc" | bc

执行结果为：192，这是用bc将二进制转换为十进制。

计算平方和平方根：

echo "10^10" | bc

echo "sqrt(100)" | bc

### man

man命令是Linux下的帮助指令，通过man指令可以查看Linux中的指令帮助、配置文件帮助和编程帮助等信息。

## 软件包管理

### rpm

格式：rpm [选项] [安装文件]

范例：

1. rpm -ivh tftp.rpm

安装名字为tftp的文件

2. rpm –qa

列出所有已安装rpm包

3. rpm –e name

卸载名字为name的rpm包

# 文件和目录管理

## 文件查找和比较

### which

which命令用于查找并显示给定命令的绝对路径，环境变量PATH中保存了查找命令时需要遍历的目录。which指令会在环境变量$PATH设置的目录里查找符合条件的文件。也就是说，使用which命令，就可以看到某个系统命令是否存在，以及执行的到底是哪一个位置的命令。

**语法**

which (选项) (参数)

**选项**

-a : 将所有由PATH目录中可以找到的命令均列出，而不止第一个被找到的命令名称

-n<文件名长度>：制定文件名长度，指定的长度必须大于或等于所有文件中最长的文件名；

-p<文件名长度>：与-n参数相同，但此处的<文件名长度>包含了文件的路径；

-w：指定输出时栏位的宽度；

-V：显示版本信息。

**参数**

指令名：指令名列表。

**实例**

查找文件、显示命令路径：

[root@localhost ~]# which pwd

/bin/pwd

[root@localhost ~]# which adduser

/usr/sbin/adduser

说明：which是根据使用者所配置的 PATH 变量内的目录去搜寻可运行档的！所以，不同的（用户） PATH 配置内容所找到的命令当然不一样的！

### locate/slocate

locate命令和slocate命令都用来查找文件或目录。 locate命令其实是find -name的另一种写法，但是要比后者快得多，原因在于它不搜索具体目录，而是搜索一个数据库/var/lib/locatedb，这个数据库中含有本地所有文件信息。Linux系统自动创建这个数据库，并且每天自动更新一次，所以使用locate命令查不到最新变动过的文件。为了避免这种情况，可以在使用locate之前，先使用updatedb命令，手动更新数据库。

updatedb：根据/etc/updatedb.conf的设置去查找系统硬盘内的文件名，并更新/var/lib/mlocate内的数据库文件；

locate：依据/var/lib/mlocate内的数据库记载，找出用户输入的关键字文件名。

locate -ir keyword

**语法**

locate/slocate (选项) (参数)

**选项**

-i ： 忽略大小写的差异；

-r ： 后面可接正则表达式的显示方式

-d<目录>或--database=<目录>：指定数据库所在的目录；

-u：更新slocate数据库；

--help：显示帮助；

--version：显示版本信息。

**参数**

查找字符串：要查找的文件名（包括目录）中含有的字符串。

**实例**

搜索etc目录下所有以sh开头的文件：

locate /etc/sh

搜索用户主目录下，所有以m开头的文件：

locate ~/m

搜索用户主目录下，所有以m开头的文件，并且忽略大小写：

locate -i ~/m

### find

推荐：<https://www.cnblogs.com/huninglei/p/5824205.html>

find命令用来在指定目录下查找文件。任何位于参数之前的字符串都将被视为欲查找的目录名。如果使用该命令时，不设置任何参数，则find命令将在当前目录下查找子目录与文件。并且将查找到的子目录和文件全部进行显示。

**语法**

find (选项) (参数)

**选项**

-amin<分钟>：查找在指定时间曾被存取过的文件或目录，单位以分钟计算；

-anewer<参考文件或目录>：查找其存取时间较指定文件或目录的存取时间更接近现在的文件或目录；

-atime<24小时数>：查找在指定时间曾被存取过的文件或目录，单位以24小时计算；

-cmin<分钟>：查找在指定时间之时被更改过的文件或目录；

-cnewer<参考文件或目录>查找其更改时间较指定文件或目录的更改时间更接近现在的文件或目录；

-ctime<24小时数>：查找在指定时间之时被更改的文件或目录，单位以24小时计算；

-daystart：从本日开始计算时间；

-depth：从指定目录下最深层的子目录开始查找；

-expty：寻找文件大小为0 Byte的文件，或目录下没有任何子目录或文件的空目录；

-exec<执行指令>：假设find指令的回传值为True，就执行该指令；

-false：将find指令的回传值皆设为False；

-fls<列表文件>：此参数的效果和指定“-ls”参数类似，但会把结果保存为指定的列表文件；

-follow：排除符号连接；

-fprint<列表文件>：此参数的效果和指定“-print”参数类似，但会把结果保存成指定的列表文件；

-fprint0<列表文件>：此参数的效果和指定“-print0”参数类似，但会把结果保存成指定的列表文件；

-fprintf<列表文件><输出格式>：此参数的效果和指定“-printf”参数类似，但会把结果保存成指定的列表文件；

-fstype<文件系统类型>：只寻找该文件系统类型下的文件或目录；

-gid<群组识别码>：查找符合指定之群组识别码的文件或目录；

-group<群组名称>：查找符合指定之群组名称的文件或目录；

-help或——help：在线帮助；

-ilname<范本样式>：此参数的效果和指定“-lname”参数类似，但忽略字符大小写的差别；

-iname<范本样式>：此参数的效果和指定“-name”参数类似，但忽略字符大小写的差别；

-inum：查找符合指定的inode编号的文件或目录；

-ipath<范本样式>：此参数的效果和指定“-path”参数类似，但忽略字符大小写的差别；

-iregex<范本样式>：此参数的效果和指定“-regexe”参数类似，但忽略字符大小写的差别；

-links<连接数目>：查找符合指定的硬连接数目的文件或目录；

-iname<范本样式>：指定字符串作为寻找符号连接的范本样式；

-ls：假设find指令的回传值为Ture，就将文件或目录名称列出到标准输出；

-maxdepth<目录层级>：设置最大目录层级；

-mindepth<目录层级>：设置最小目录层级；

-mmin<分钟>：查找在指定时间曾被更改过的文件或目录，单位以分钟计算；

-mount：此参数的效果和指定“-xdev”相同；

-mtime<24小时数>：查找在指定时间曾被更改过的文件或目录，单位以24小时计算；

-name<范本样式>：指定字符串作为寻找文件或目录的范本样式；

-newer<参考文件或目录>：查找其更改时间较指定文件或目录的更改时间更接近现在的文件或目录；

-nogroup：找出不属于本地主机群组识别码的文件或目录；

-noleaf：不去考虑目录至少需拥有两个硬连接存在；

-nouser：找出不属于本地主机用户识别码的文件或目录；

-ok<执行指令>：此参数的效果和指定“-exec”类似，但在执行指令之前会先询问用户，若回答“y”或“Y”，则放弃执行命令；

-path<范本样式>：指定字符串作为寻找目录的范本样式；

-perm<权限数值>：查找符合指定的权限数值的文件或目录；

-print：假设find指令的回传值为Ture，就将文件或目录名称列出到标准输出。格式为每列一个名称，每个名称前皆有“./”字符串；

-print0：假设find指令的回传值为Ture，就将文件或目录名称列出到标准输出。格式为全部的名称皆在同一行；

-printf<输出格式>：假设find指令的回传值为Ture，就将文件或目录名称列出到标准输出。格式可以自行指定；

-prune：不寻找字符串作为寻找文件或目录的范本样式;

-regex<范本样式>：指定字符串作为寻找文件或目录的范本样式；

-size<文件大小>：查找符合指定的文件大小的文件；

-true：将find指令的回传值皆设为True；

-typ<文件类型>：只寻找符合指定的文件类型的文件；

-uid<用户识别码>：查找符合指定的用户识别码的文件或目录；

-used<日数>：查找文件或目录被更改之后在指定时间曾被存取过的文件或目录，单位以日计算；

-user<拥有者名称>：查找符和指定的拥有者名称的文件或目录；

-version或——version：显示版本信息；

-xdev：将范围局限在先行的文件系统中；

-xtype<文件类型>：此参数的效果和指定“-type”参数类似，差别在于它针对符号连接检查。

**参数**

起始目录：查找文件的起始目录。

**实例**

* **根据文件或者正则表达式进行匹配**

列出当前目录及子目录下所有文件和文件夹

find .

在/home目录下查找以.txt结尾的文件名

find /home -name "\*.txt"

同上，但忽略大小写

find /home -iname "\*.txt"

当前目录及子目录下查找所有以.txt和.pdf结尾的文件

find . \( -name "\*.txt" -o -name "\*.pdf" \)

或

find . -name "\*.txt" -o -name "\*.pdf"

匹配文件路径或者文件

find /usr/ -path "\*local\*"

基于正则表达式匹配文件路径

find . -regex ".\*\(\.txt\|\.pdf\)$"

同上，但忽略大小写

find . -iregex ".\*\(\.txt\|\.pdf\)$"

* **否定参数**

找出/home下不是以.txt结尾的文件

find /home ! -name "\*.txt"

* **根据文件类型进行搜索**

find . -type 类型参数

类型参数列表：

f 普通文件

l 符号连接

d 目录

c 字符设备

b 块设备

s 套接字

p Fifo

* **基于目录深度搜索**

向下最大深度限制为3

find . -maxdepth 3 -type f

搜索出深度距离当前目录至少2个子目录的所有文件

find . -mindepth 2 -type f

* **根据文件时间戳进行搜索**

find . -type f 时间戳

UNIX/Linux文件系统每个文件都有三种时间戳：

访问时间（-atime/天，-amin/分钟）：用户最近一次访问时间。

修改时间（-mtime/天，-mmin/分钟）：文件最后一次修改时间。

变化时间（-ctime/天，-cmin/分钟）：文件数据元（例如权限等）最后一次修改时间。

搜索最近七天内被访问过的所有文件

find . -type f -atime -7

搜索恰好在七天前被访问过的所有文件

find . -type f -atime 7

搜索超过七天内被访问过的所有文件

find . -type f -atime +7

搜索访问时间超过10分钟的所有文件

find . -type f -amin +10

找出比file.log修改时间更长的所有文件

find . -type f -newer file.log

* **根据文件大小进行匹配**

find . -type f -size 文件大小单元

文件大小单元：

b —— 块（512字节）

c —— 字节

w —— 字（2字节）

k —— 千字节

M —— 兆字节

G —— 吉字节

搜索大于10KB的文件

find . -type f -size +10k

搜索小于10KB的文件

find . -type f -size -10k

搜索等于10KB的文件

find . -type f -size 10k

* **删除匹配文件**

删除当前目录下所有.txt文件

find . -type f -name "\*.txt" -delete

* **根据文件权限/所有权进行匹配**

当前目录下搜索出权限为777的文件

find . -type f -perm 777

找出当前目录下权限不是644的php文件

find . -type f -name "\*.php" ! -perm 644

找出当前目录用户tom拥有的所有文件

find . -type f -user tom

找出当前目录用户组sunk拥有的所有文件

find . -type f -group sunk

* **借助-exec选项与其他命令结合使用**

找出当前目录下所有root的文件，并把所有权更改为用户tom

find .-type f -user root -exec chown tom {} \;

上例中，{} 用于与-exec选项结合使用来匹配所有文件，然后会被替换为相应的文件名。

找出自己家目录下所有的.txt文件并删除

find $HOME/. -name "\*.txt" -ok rm {} \;

上例中，-ok和-exec行为一样，不过它会给出提示，是否执行相应的操作。

查找当前目录下所有.txt文件并把他们拼接起来写入到all.txt文件中

find . -type f -name "\*.txt" -exec cat {} \;> all.txt

将30天前的.log文件移动到old目录中

find . -type f -mtime +30 -name "\*.log" -exec cp {} old \;

找出当前目录下所有.txt文件并以“File:文件名”的形式打印出来

find . -type f -name "\*.txt" -exec printf "File: %s\n" {} \;

因为单行命令中-exec参数中无法使用多个命令，以下方法可以实现在-exec之后接受多条命令

-exec ./text.sh {} \;

搜索但跳出指定的目录 查找当前目录或者子目录下所有.txt文件，但是跳过子目录sk

find . -path "./sk" -prune -o -name "\*.txt" -print find

* **其他技巧收集**

要列出所有长度为零的文件

find . -empty

## 文件处理

### basename

basename命令用于打印目录或者文件的基本名称。basename和dirname命令通常用于shell脚本中的命令替换来指定和指定的输入文件名称有所差异的输出文件名称。

**语法**

basename (选项) (参数)

**选项**

--help：显示帮助；

--version：显示版本号。

**参数**

文件：带路径信息的文件；

后缀：可选参数，指定要去除的文件后缀字符串。

**实例**

1、要显示一个shell变量的基本名称，请输入：

basename $WORKFILE

此命令显示指定给shell变量WORKFILE的值的基本名称。如果WORKFILE变量的值是/home/jim/program.c文件，则此命令显示program.c。

要构造一个和另一个文件名称相同（除了后缀）的文件名称，请输入：

OFILE=`basename $1 .c`.o

此命令指定给 OFILE 文件第一个位置上的参数（$1）的值，但它的 .c 后缀更改至 .o。如果 $1 是 /home/jim/program.c 文件，则 OFILE 成为 program.o。因为 program.o 仅是一个基本文件名称，它标识在当前目录中的文件。

**补充：**

**basename与dirname**

[czha@localhost scripts]$ basename /server/scripts/m2.sh

m2.sh

[czha@localhost scripts]$ basename /server/scripts/

scripts

[czha@localhost scripts]$ basename /server/scripts

scripts

[czha@localhost scripts]$ dirname /server/scripts/m2.sh

/server/scripts

[czha@localhost scripts]$ dirname /server/scripts/

/server

[czha@localhost scripts]$ dirname /server/scripts

/server

### dirname

dirname命令去除文件名中的非目录部分，仅显示与目录有关的内容。dirname命令读取指定路径名保留最后一个/及其后面的字符，删除其他部分，并写结果到标准输出。如果最后一个/后无字符，dirname 命令使用倒数第二个/，并忽略其后的所有字符。dirname 和 basename 通常在 shell 内部命令替换使用，以指定一个与指定输入文件名略有差异的输出文件名。

语法

dirname (选项) (参数)

**选项**

--help：显示帮助；

--version：显示版本号。

**实例**

dirname //

结果为 /

dirname /a/b/

结果为：/a

dirname a

结果为 .

dirname a/b

结果为路径名 a

### touch

touch命令有两个功能：一是用于把已存在文件的时间标签更新为系统当前的时间（默认方式），它们的数据将原封不动地保留下来；二是用来创建新的空文件。

**语法**

touch (选项) (参数)

**选项**

-a：或--time=atime或--time=access或--time=use 只更改访问时间（用于更新文件的时间，见基础篇）；

-c：或--no-create 不建立任何文件；

-d：<时间日期> 使用指定的日期时间，而非现在的时间；

-f：此参数将忽略不予处理，仅负责解决BSD版本touch指令的兼容性问题；

-m：或--time=mtime或--time=modify 只更该变动时间；

-r：<参考文件或目录> 把指定文件或目录的日期时间，统统设成和参考文件或目录的日期时间相同； -t：<日期时间> 使用指定的日期时间，而非现在的时间；

--help：在线帮助；

--version：显示版本信息。

参数

文件：指定要设置时间属性的文件列表。

**实例**

touch ex2

在当前目录下建立一个空文件ex2，然后，利用ls -l命令可以发现文件ex2的大小为0，表示它是空文件。

### cat

cat命令连接文件并打印到标准输出设备上，cat经常用来显示文件的内容，类似于下的type命令。

注意：当文件较大时，文本在屏幕上迅速闪过（滚屏），用户往往看不清所显示的内容。因此，一般用more等命令分屏显示。为了控制滚屏，可以按Ctrl+S键，停止滚屏；按Ctrl+Q键可以恢复滚屏。按Ctrl+C（中断）键可以终止该命令的执行，并且返回Shell提示符状态。

**语法**

cat (选项) (参数)

**选项**

-n或-number： 从1开始对所有输出的行数编号；

-b或--number-nonblank： 和-n相似，只不过对于空白行不编号；

-s或--squeeze-blank： 当遇到有连续两行以上的空白行，就代换为一行的空白行；

-A： 显示不可打印字符，行尾显示“$”；

-e： 等价于"-vE"选项；

-t： 等价于"-vT"选项；

**参数**

文件列表：指定要连接的文件列表。

**实例**

设ml和m2是当前目录下的两个文件

cat m1 （在屏幕上显示文件ml的内容）

cat m1 m2 （同时显示文件ml和m2的内容）

cat m1 m2 > file （将文件ml和m2合并后放入文件file中）

## 目录基本操作

### cd

cd命令用来切换工作目录至dirname（change directory的缩写）。 其中dirName表示法可为绝对路径或相对路径。若目录名称省略，则变换至使用者的home directory(也就是刚login时所在的目录)。另外，~也表示为home directory的意思，.则是表示目前所在的目录，..则表示目前目录位置的上一层目录。

**语法**

cd (选项) (参数) 选项

-p 如果要切换到的目标目录是一个符号连接，直接切换到符号连接指向的目标目录

-L 如果要切换的目标目录是一个符号的连接，直接切换到字符连接名代表的目录，而非符号连接所指向的目标目录。 - 当仅实用"-"一个选项时，当前工作目录将被切换到环境变量"OLDPWD"所表示的目录。

**实例**

cd 进入用户主目录；

cd ~ 进入用户主目录；（两者一个意思）

cd - 返回进入此目录之前所在的目录；

cd .. 返回上级目录（若当前目录为“/“，则执行完后还在“/"；".."为上级目录的意思）；

cd ../.. 返回上两级目录；

cd !$ 把上个命令的参数作为cd参数使用。

### mkdir

mkdir命令用来创建目录。该命令创建由dirname命名的目录。如果在目录名的前面没有加任何路径名，则在当前目录下创建由dirname指定的目录；如果给出了一个已经存在的路径，将会在该目录下创建一个指定的目录。在创建目录时，应保证新建的目录与它所在目录下的文件没有重名。

注意：在创建文件时，不要把所有的文件都存放在主目录中，可以创建子目录，通过它们来更有效地组织文件。最好采用前后一致的命名方式来区分文件和目录。例如，目录名可以以大写字母开头，这样，在目录列表中目录名就出现在前面。

在一个子目录中应包含类型相似或用途相近的文件。例如，应建立一个子目录，它包含所有的数据库文件，另有一个子目录应包含电子表格文件，还有一个子目录应包含文字处理文档，等等。目录也是文件，它们和普通文件一样遵循相同的命名规则，并且利用全路径可以唯一地指定一个目录。

**语法**

mkdir (选项) (参数)

**选项**

-Z：设置安全上下文，当使用SELinux时有效；

-m<目标属性>或--mode<目标属性>建立目录的同时设置目录的权限；

-p或--parents 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；

--version 显示版本信息。

**参数**

目录：指定要创建的目录列表，多个目录之间用空格隔开。

**实例**

在目录/usr/meng下建立子目录test，并且只有文件主有读、写和执行权限，其他人无权访问

mkdir -m 700 /usr/meng/test

在当前目录中建立bin和bin下的os\_1目录，权限设置为文件主可读、写、执行，同组用户可读和执行，其他用户无权访问

mkdir -p -m 750 bin/os\_1

### rmdir

rmdir命令用来删除空目录。当目录不再被使用时，或者磁盘空间已到达使用限定值，就需要删除失去使用价值的目录。利用rmdir命令可以从一个目录中删除一个或多个空的子目录。该命令从一个目录中删除一个或多个子目录，其中dirname表示目录名。如果dirname中没有指定路径，则删除当前目录下由dirname指定的目录；如dirname中包含路径，则删除指定位置的目录。删除目录时，必须具有对其父目录的写权限。

注意：子目录被删除之前应该是空目录。就是说，该目录中的所有文件必须用rm命令删除，另外，当前工作目录必须在被删除目录之上，不能是被删除目录本身，也不能是被删除目录的子目录。

虽然还可以用带有-r选项的rm命令递归删除一个目录中的所有文件和该目录本身，但是这样做存在很大的危险性。

**语法**

rmdir (选项) (参数)

**选项**

-p或--parents：删除指定目录后，若该目录的上层目录已变成空目录，则将其一并删除；

--ignore-fail-on-non-empty：此选项使rmdir命令忽略由于删除非空目录时导致的错误信息；

-v或-verboes：显示命令的详细执行过程；

--help：显示命令的帮助信息；

--version：显示命令的版本信息。

**参数**

目录列表：要删除的空目录列表。当删除多个空目录时，目录名之间使用空格隔开。

**实例**

删除子目录os\_1和其父目录bin

cd /usr/meng/test

rmdir -p bin/os\_1

### ls

ls命令用来显示目标列表，在Linux中是使用率较高的命令。ls命令的输出信息可以进行彩色加亮显示，以分区不同类型的文件。

**语法**

ls （选项） （参数）

选项

-a： 全部的文件，连同隐藏文件（开头为 . 的文件）一起列出来（常用）

-A： 显示除除文件“.”和“..”以外的所有文件列表；

-C： 多列显示输出结果。这是默认选项；

-l： 与“-C”选项功能相反，所有输出信息用单列格式输出，不输出为多列；

-F： 在每个输出项后追加文件的类型标识符，具体含义：

“\*”表示具有可执行权限的普通文件，

“/”表示目录，

“@”表示符号链接，

“|”表示命令管道FIFO，

“=”表示sockets套接字。

当文件为普通文件时，不输出任何标识符；

-b： 将文件中的不可输出的字符以反斜线“”加字符编码的方式输出；

-c： 与“-lt”选项连用时，按照文件状态时间排序输出目录内容，排序的依据是

文件的索引节点中的ctime字段。与“-l”选项连用时，则排序的一句是文件

的状态改变时间；

-d： 仅显示目录名，而不显示目录下的内容列表。显示符号链接文件本身，而

不显示其所指向的目录列表；

-f： 此参数的效果和同时指定“aU”参数相同，并关闭“lst”参数的效果；

-h: 将文件容量以人类较易读的方式（例如KB， GB等）列出来；

-i： 显示文件索引节点号（inode）。一个索引节点代表一个文件；

-k： 以KB（千字节）为单位显示文件大小；

-l： 以长格式显示目录下的内容列表。输出的信息从左到右依次包括文件名，文

件类型、权限模式、硬连接数、所有者、组、文件大小和文件的最后修改时

间等；

-m： 用“,”号区隔每个文件和目录的名称；

-n： 以用户识别码和群组识别码替代其名称；

-r： 以文件名反序排列并输出目录内容列表；

-s： 显示文件和目录的大小，以区块为单位；

-t： 用文件和目录的更改时间排序；

-L： 如果遇到性质为符号链接的文件或目录，直接列出该链接所指向的原始文件或目录；

-R： 递归处理，将指定目录下的所有文件及子目录一并处理；

--file-type： 与“-F”选项的功能相同，但是不显示“\*”；

--full-time： 列出完整的日期与时间；

--color[=WHEN]：使用不同的颜色高亮显示不同类型的。

### cp

cp命令用来将一个或多个源文件或者目录复制到指定的目的文件或目录。它可以将单个源文件复制成一个指定文件名的具体的文件或一个已经存在的目录下。cp命令还支持同时复制多个文件，当一次复制多个文件时，目标文件参数必须是一个已经存在的目录，否则将出现错误。

**语法**

cp (选项) (参数)

cp [options] source1 source2 … directory

**选项**

-a：此参数的效果和同时指定"-dpR"参数相同；

-d：当复制符号连接时，把目标文件或目录也建立为符号连接，并指向与源文件或目录连接的原始文件或目录；

-f：强行复制文件或目录，不论目标文件或目录是否已存在；

-i：覆盖既有文件之前先询问用户；

-l：对源文件建立硬连接，而非复制文件；

-p：保留源文件或目录的属性；

-R/r：递归处理，将指定目录下的所有文件与子目录一并处理；

-s：对源文件建立符号连接（复制成为符号链接文件，即快捷方式文件），而非复制文件；

-u：使用这项参数后只会在源文件的更改时间较目标文件更新时或是名称相互对应的目标文件并不存在时，才复制文件；

-S：在备份文件时，用指定的后缀“SUFFIX”代替文件的默认后缀；

-b：覆盖已存在的文件目标前将目标文件备份；

-v：详细显示命令执行的操作。

**参数**

源文件：制定源文件列表。默认情况下，cp命令不能复制目录，如果要复制目录，则必须使用-R选项； 目标文件：指定目标文件。当“源文件”为多个文件时，要求“目标文件”为指定的目录。

**实例**

如果把一个文件复制到一个目标文件中，而目标文件已经存在，那么，该目标文件的内容将被破坏。此命令中所有参数既可以是绝对路径名，也可以是相对路径名。通常会用到 点. 或 点点.. 的形式。例如，下面的命令将指定文件复制到当前目录下：

cp ../mary/homework/assign .

所有目标文件指定的目录必须是己经存在的，cp命令不能创建目录。如果没有文件复制的权限，则系统会显示出错信息。

将文件file复制到目录/usr/men/tmp下，并改名为file1 cp file /usr/men/tmp/file1 将目录/usr/men下的所有文件及其子目录复制到目录/usr/zh中

cp -r /usr/men /usr/zh

交互式地将目录/usr/men中的以m打头的所有.c文件复制到目录/usr/zh中

cp -i /usr/men m\*.c /usr/zh

我们在Linux下使用cp命令复制文件时候，有时候会需要覆盖一些同名文件，覆盖文件的时候都会有提示：需要不停的按Y来确定执行覆盖。文件数量不多还好，但是要是几百个估计按Y都要吐血了，于是折腾来半天总结了一个方法：

* cp aaa/\* /bbb

复制目录aaa下所有到/bbb目录下，这时如果/bbb目录下有和aaa同名的文件，需要按Y来确认并且会略过aaa目录下的子目录。

* cp -r aaa/\* /bbb

这次依然需要按Y来确认操作，但是没有忽略子目录。

* cp -r -a aaa/\* /bbb

依然需要按Y来确认操作，并且把aaa目录以及子目录和文件属性也传递到了/bbb。

* \cp -r -a aaa/\* /bbb

成功，没有提示按Y、传递了目录属性、没有略过目录。

### rm

rm命令可以删除一个目录中的一个或多个文件或目录，也可以将某个目录及其下属的所有文件及其子目录均删除掉。对于链接文件，只是删除整个链接文件，而原有文件保持不变。

注意：使用rm命令要格外小心。因为一旦删除了一个文件，就无法再恢复它。所以，在删除文件之前，最好再看一下文件的内容，确定是否真要删除。rm命令可以用-i选项，这个选项在使用文件扩展名字符删除多个文件时特别有用。使用这个选项，系统会要求你逐一确定是否要删除。这时，必须输入y并按Enter键，才能删除文件。如果仅按Enter键或其他字符，文件不会被删除。

**语法**

rm (选项) (参数)

**选项**

-d：直接把欲删除的目录的硬连接数据删除成0，删除该目录；

-f：强制删除文件或目录；

-i：删除已有文件或目录之前先询问用户；

-r或-R：递归处理，将指定目录下的所有文件与子目录一并处理；

--preserve-root：不对根目录进行递归操作；

-v：显示指令的详细执行过程。

**参数**

文件：指定被删除的文件列表，如果参数中含有目录，则必须加上-r或者-R选项。

**实例**

交互式删除当前目录下的文件test和example

rm -i test example

Remove test ?n（不删除文件test)

Remove example ?y（删除文件example)

删除当前目录下除隐含文件外的所有文件和子目录

# rm -r \*

应注意，这样做是非常危险的!

### mv

mv命令用来对文件或目录重新命名，或者将文件从一个目录移到另一个目录中。source表示源文件或目录，target表示目标文件或目录。如果将一个文件移到一个已经存在的目标文件中，则目标文件的内容将被覆盖。

mv命令可以用来将源文件移至一个目标文件中，或将一组文件移至一个目标目录中。源文件被移至目标文件有两种不同的结果：

1. 如果目标文件是到某一目录文件的路径，源文件会被移到此目录下，且文件名不变。
2. 如果目标文件不是目录文件，则源文件名（只能有一个）会变为此目标文件名，并覆盖己存在的同名文件。如果源文件和目标文件在同一个目录下，mv的作用就是改文件名。当目标文件是目录文件时，源文件或目录参数可以有多个，则所有的源文件都会被移至目标文件中。所有移到该目录下的文件都将保留以前的文件名。

注意事项：mv与cp的结果不同，mv好像文件“搬家”，文件个数并未增加。而cp对文件进行复制，文件个数增加了。

**语法**

mv (选项) (参数)

**选项**

--backup=<备份模式>：若需覆盖文件，则覆盖前先行备份；

-b：当文件存在时，覆盖前，为其创建一个备份；

-f：若目标文件或目录与现有的文件或目录重复，则直接覆盖现有的文件或目录；

-i：交互式操作，覆盖前先行询问用户，如果源文件与目标文件或目标目录中的文件同名，则询问用户是否覆盖目标文件。用户输入”y”，表示将覆盖目标文件；输入”n”，表示取消对源文件的移动。这样可以避免误将文件覆盖。

--strip-trailing-slashes：删除源文件中的斜杠“/”；

-S<后缀>：为备份文件指定后缀，而不使用默认的后缀；

--target-directory=<目录>：指定源文件要移动到目标目录；

-u：当源文件比目标文件新或者目标文件不存在时，才执行移动操作。

**参数**

源文件：源文件列表。

目标文件：如果“目标文件”是文件名则在移动文件的同时，将其改名为“目标文件”；如果“目标文件”是目录名则将源文件移动到“目标文件”下。

**实例**

将文件ex3改名为new1

mv ex3 new1

将目录/usr/men中的所有文件移到当前目录（用.表示）中：

mv /usr/men/\* .

### pwd

pwd命令以绝对路径的方式显示用户当前工作目录。命令将当前目录的全路径名称（从根目录）写入标准输出。全部目录使用/分隔。第一个/表示根目录，最后一个目录是当前目录。执行pwd命令可立刻得知您目前所在的工作目录的绝对路径名称。

**语法**

pwd（选项）

**选项**

--help：显示帮助信息；

--version：显示版本信息。

**实例**

[root@localhost ~]# pwd /root

## 文件权限属性设置

### chown

chown命令改变某个文件或目录的所有者和所属的组，该命令可以向某个用户授权，使该用户变成指定文件的所有者或者改变文件所属的组。用户可以是用户或者是用户D，用户组可以是组名或组id。文件名可以使由空格分开的文件列表，在文件名中可以包含通配符。

只有文件主和超级用户才可以便用该命令。

**语法**

chown (选项) (参数)

**选项**

-c或——changes：效果类似“-v”参数，但仅显示更改的部分；

-f或--quite或——silent：不显示错误信息；

-h或--no-dereference：只对符号连接的文件作修改，而不更改其他任何相关文件；

-R或——recursive：递归处理，将指定目录下的所有文件及子目录一并处理；

-v或——verbose：显示指令执行过程；

--dereference：效果和“-h”参数相同；

--help：在线帮助；

--reference=<参考文件或目录>：把指定文件或目录的拥有者与所属群组全部设成和参考文件或目录的拥有者与所属群组相同； -

-version：显示版本信息。

**参数**

用户:组：指定所有者和所属工作组。当省略“：组”，仅改变文件所有者；

文件：指定要改变所有者和工作组的文件列表。支持多个文件和目标，支持shell通配符。

**实例**

将目录/usr/meng及其下面的所有文件、子目录的文件主改成 liu：

chown -R liu /usr/meng

[root@localhost ShellFile]# ls -al sh01.sh

-rw-r--r--. 1 root czha 269 10月 2 23:53 sh01.sh

[root@localhost ShellFile]# chown czha sh01.sh //将sh01.sh的所有者改为czha这个账号

[root@localhost ShellFile]# ls -al sh01.sh

-rw-r--r--. 1 czha czha 269 10月 2 23:53 sh01.sh

[root@localhost ShellFile]# su czha

[czha@localhost ShellFile]$ vim sh01.sh //此时，发现czha用户下，该文件也可以进行修改

[czha@localhost ShellFile]$ chown root:root sh01.sh

chown: 正在更改"sh01.sh" 的所有者: 不允许的操作

[czha@localhost ShellFile]$ su root

密码：

//将sh01.sh的所有者与用户组改回为root，注意是切换为root用户的权限下才能执行成功

[root@localhost ShellFile]# chown root:root sh01.sh

[root@localhost ShellFile]# ls -al sh01.sh

-rw-r--r--. 1 root root 269 10月 12 15:33 sh01.sh

chown也可以单纯的修改用户组。

例如：

[root@localhost ShellFile]# ls -al sh01.sh

-rw-r--r--. 1 root root 269 10月 12 15:33 sh01.sh

[root@localhost ShellFile]# chown .czha sh01.sh //“.”点号+组名

[root@localhost ShellFile]# ls -al sh01.sh

-rw-r--r--. 1 root czha 269 10月 12 15:33 sh01.sh

**使用场景**

假设你要讲.bashrc这个文件复制成为.bashrc\_test文件名，并且是要给bin这个用户：

cp .bashrc .bashrc\_test

由于复制行为（cp）会复制执行者的属性和权限，两个文件的属性一样，bashrc\_test还是属于root所有，即使你将文件给予bin这个用户了，那他仍然无法修改的，所以你必须要修改这个文件的所有者与用户组。

书本p180-181两个例题，可以看一下。

### chgrp

chgrp命令用来改变文件或目录所属的用户组。该命令用来改变指定文件所属的用户组。其中，组名可以是用户组的id，也可以是用户组的组名。文件名可以是由空格分开的要改变属组的文件列表，也可以是由通配符描述的文件集合。如果用户不是该文件的文件主或超级用户(root)，则不能改变该文件的组。

在UNIX系统家族里，文件或目录权限的掌控以拥有者及所属群组来管理。您可以使用chgrp指令去变更文件与目录的所属群组，设置方式采用群组名称或群组识别码皆可。

**语法**

chgrp (选项) (参数)

**选项**

-c或——changes： 效果类似“-v”参数，但仅回报更改的部分；

-f或--quiet或——silent： 不显示错误信息；

-h或--no-dereference： 只对符号连接的文件作修改，而不是该其他任何相关文件；

-R或——recursive： 递归处理，将指令目录下的所有文件及子目录一并处理；

-v或——verbose： 显示指令执行过程；

--reference=<参考文件或目录>： 把指定文件或目录的所属群组全部设成和参考文件或目录的所属群组相同；

**参数**

组：指定新工作名称；

文件：指定要改变所属组的文件列表。多个文件或者目录之间使用空格隔开。

**实例**

将/usr/meng及其子目录下的所有文件的用户组改为mengxin

chgrp -R mengxin /usr/meng

### chmod

chmod命令用来变更文件或目录的权限。在UNIX系统家族里，文件或目录权限的控制分别以读取、写入、执行3种一般权限来区分，另有3种特殊权限可供运用。用户可以使用chmod指令去变更文件与目录的权限，设置方式采用文字或数字代号皆可。符号连接的权限无法变更，如果用户对符号连接修改权限，其改变会作用在被连接的原始文件。

权限范围的表示法如下：

u User，即文件或目录的拥有者；

g Group，即文件或目录的所属群组；

o Other，除了文件或目录拥有者或所属群组之外，其他用户皆属于这个范围；

a All，即全部的用户，包含拥有者，所属群组以及其他用户；

r 读取权限，数字代号为“4”;

w 写入权限，数字代号为“2”；

x 执行或切换权限，数字代号为“1”；

- 不具任何权限，数字代号为“0”；

s 特殊功能说明：变更文件或目录的权限。

**语法**

chmod (选项) (参数)

**选项**

-c或——changes：效果类似“-v”参数，但仅回报更改的部分；

-f或--quiet或——silent：不显示错误信息；

-R或——recursive：递归处理，将指令目录下的所有文件及子目录一并处理；

-v或——verbose：显示指令执行过程；

--reference=<参考文件或目录>：把指定文件或目录的所属群组全部设成和参考文件或目录的所属群组相同；

<权限范围>+<权限设置>：开启权限范围的文件或目录的该选项权限设置；

<权限范围>-<权限设置>：关闭权限范围的文件或目录的该选项权限设置；

<权限范围>=<权限设置>：指定权限范围的文件或目录的该选项权限设置；

**参数**

权限模式：指定文件的权限模式；

文件：要改变权限的文件。

**知识扩展和实例**

Linux用户分为：拥有者、组群(Group)、其他（other），Linux系统中，预设的情況下，系统中所有的帐号与一般身份使用者，以及root的相关信息，都是记录在/etc/passwd文件中。每个人的密码则是记录在/etc/shadow文件下。 此外，所有的组群名称记录在/etc/group內！

例：rwx　rw-　r—

r=读取属性 //值＝4

w=写入属性　　 //值＝2

x=执行属性　　 //值＝1

chmod u+x,g+w f01 //为文件f01设置自己可以执行，组员可以写入的权限

chmod u=rwx,g=rw,o=r f01

chmod 764 f01

chmod a+x f01　　 //对文件f01的u,g,o都设置可执行属性

文件的属主和属组属性设置

chown user:market f01　　 //把文件f01给uesr，添加到market组

ll -d f1 //查看目录f1的属性

### umask

umask命令用来设置限制新建文件权限的掩码。当新文件被创建时，其最初的权限由文件创建掩码决定。用户每次注册进入系统时，umask命令都被执行， 并自动设置掩码mode来限制新文件的权限。用户可以通过再次执行umask命令来改变默认值，新的权限将会把旧的覆盖掉。

**语法**

umask (选项) (参数)

**选项**

-p：输出的权限掩码可直接作为指令来执行；

-S：以符号方式输出权限掩码。

**参数**

权限掩码：指定权限掩码。

**实例**

利用umask命令可以指定哪些权限将在新文件的默认权限中被删除。例如，可以使用下面的命令创建掩码，使得组用户的写权限，其他用户的读、写和执行权限都被取消：

umask u=, g=w, o=rwx

执行该命令以后，对于下面创建的新文件，其文件主的权限未做任何改变，而组用户没有写权限，其他用户的所有权限都被取消。

应注意：操作符“=”在umask命令和chmod命令中的作用恰恰相反。在chmod命令中，利用它来设置指定的权限，而其余权限则被删除；但是在umask命令中，它将在原有权限的基础上删除指定的权限。

不能直接利用umask命令创建一个可执行的文件，用户只能在其后利用chmod命令使它具有执行权限。假设执行了命令umask u=, g=w, o=rwx，虽然在命令行中，没有删去文件主和组用户的执行权限，但默认的文件权限还是640（即 rw-r-----），而不是750(rwxr-x---）。但是，如果创建的是目录或者通过编译程序创建的一个可执行文件，将不受此限制。在这种情况 下，会设置文件的执行权限。

也可以使用八进制数值来设置mode。由于在umask中所指定的权限是要从文件中删除的，所以，如果该文件原来的初始化权限是777，那么执行命令umask 022以后，该文件的权限将变为755：如果该文件原来的初始化权限是666，那么该文件的权限将变为644。

可以使用下面的命令检查新创建文件的默认权限：

umask -s

选项-s表示以字符形式显示当前的掩码。如果直接输入umask命令，不带任何参数，那么将以八进制形式显示当前的掩码。系统默认的掩码是0022。

### chattr

chattr命令用来改变文件属隐藏性。这项指令可改变存放在**ext2文件系统**上的文件或目录属性，这些属性共有以下8种模式：

a：让文件或目录仅供附加用途；

b：不更新文件或目录的最后存取时间；

c：将文件或目录压缩后存放；

d：将文件或目录排除在倾倒操作之外；

i：不得任意更动文件或目录；

s：保密性删除文件或目录；

S：即时更新文件或目录；

u：预防意外删除。

**语法**

chattr (选项)

**选项**

-R：递归处理，将指令目录下的所有文件及子目录一并处理；

-v<版本编号>：设置文件或目录版本；

-V：显示指令执行过程；

+<属性>：开启文件或目录的该项属性；

-<属性>：关闭文件或目录的该项属性；

=<属性>：指定文件或目录的该项属性。

**实例**

用chattr命令防止系统中某个关键文件被修改：

chattr +i /etc/fstab

然后试一下rm、mv、rename等命令操作于该文件，都是得到Operation not permitted的结果。

让某个文件只能往里面追加内容，不能删除，一些日志文件适用于这种操作：

chattr +a /data1/user\_act.log

### lsattr

lsattr命令用于查看文件的第二扩展文件系统属性（即文件隐藏属性）。

**语法**

lsattr (选项) (参数)

**选项**

-E：可显示设备属性的当前值，但这个当前值是从用户设备数据库中获得的，而不是从设备直接获得的。

-D：显示属性的名称，属性的默认值，描述和用户是否可以修改属性值的标志。

-R：递归的操作方式；

-V：显示指令的版本信息；

-a：列出目录中的所有文件，包括隐藏文件。

lsattr经常使用的几个选项-D，-E，-R这三个选项不可以一起使用，它们是互斥的，经常使用的还有-l,-H，使用lsattr时，必须指出具体的设备名，用-l选项指出要显示设备的逻辑名称，否则要用-c，-s，-t等选项唯一的确定某个已存在的设备。

**参数**

文件：指定显示文件系统属性的文件名。

**实例**

lsattr -E -l rmt0 -H lsattr -EO -l rmt0

### file

file命令用来探测给定文件的类型。file命令对文件的检查分为文件系统、魔法幻数检查和语言检查3个过程。

如果你想知道某个文件的基本数据类型，例如是ASCII或者是data文件，或者是binary，且其中没有使用到动态库函数（share library）等等的信息，就可以利用file这个命令来查看。

file xxx

**语法**

file (选项) (参数)

**选项**

-b：列出辨识结果时，不显示文件名称；

-c：详细显示指令执行过程，便于排错或分析程序执行的情形；

-f<名称文件>：指定名称文件，其内容有一个或多个文件名称时，让file依序辨识这些文件，格式为每列一个文件名称；

-L：直接显示符号连接所指向的文件类别；

-m<魔法数字文件>：指定魔法数字文件；

-v：显示版本信息；

-z：尝试去解读压缩文件的内容。

**参数**

文件：要确定类型的文件列表，多个文件之间使用空格分开，可以使用shell通配符匹配多个文件。

**实例**

显示文件类型

[root@localhost ~]# file install.log

install.log: UTF-8 Unicode text

[root@localhost ~]# file -b install.log <== 不显示文件名称

UTF-8 Unicode text

[root@localhost ~]# file -i install.log <== 显示MIME类别。

install.log: text/plain; charset=utf-8

[root@localhost ~]# file -b -i install.log

text/plain; charset=utf-8

显示符号链接的文件类型

[root@localhost ~]# ls -l /var/mail

lrwxrwxrwx 1 root root 10 08-13 00:11 /var/mail -> spool/mail

[root@localhost ~]# file /var/mail

/var/mail: symbolic link to `spool/mail'

[root@localhost ~]# file -L /var/mail

/var/mail: directory

[root@localhost ~]# file /var/spool/mail

/var/spool/mail: directory

[root@localhost ~]# file -L /var/spool/mail

/var/spool/mail: directory

### 文件过滤分割与合并

### grep

## 文件压缩与解压

若说gzip是为了替代compress并提供更好的压缩比例而成立的，那么bzip2则是为了取代gzip并提供更佳的压缩比而来的。

### tar

tar命令可以为linux的文件和目录创建档案。利用tar，可以为某一特定文件创建档案（备份文件），也可以在档案中改变文件，或者向档案中加入新的文件。tar最初被用来在磁带上创建档案，现在，用户可以在任何设备上创建档案。利用tar命令，可以把一大堆的文件和目录全部打包成一个文件，这对于备份文件或将几个文件组合成为一个文件以便于网络传输是非常有用的。

首先要弄清两个概念：打包和压缩。

打包是指将一大堆文件或目录变成一个总的文件；

压缩则是将一个大的文件通过一些压缩算法变成一个小文件。

为什么要区分这两个概念呢？这源于Linux中很多压缩程序只能针对一个文件进行压缩，这样当你想要压缩一大堆文件时，你得先将这一大堆文件先打成一个包（tar命令），然后再用压缩程序进行压缩（gzip bzip2命令）。

**语法**

tar (选项) (参数)

**选项**

-A或--catenate：新增文件到以存在的备份文件；

-B：设置区块大小；

-c或--create：建立新的备份文件；

-C <目录>：这个选项用在解压缩，若要在特定目录解压缩，可以使用这个选项。

-d：记录文件的差别；

-x或--extract或--get：从备份文件中还原文件；

-t或--list：列出备份文件的内容；

-z或--gzip或--ungzip：通过gzip指令处理备份文件；

-Z或--compress或--uncompress：通过compress指令处理备份文件；

-f<备份文件>或--file=<备份文件>：指定备份文件；

-v或--verbose：显示指令执行过程；

-r：添加文件到已经压缩的文件；

-u：添加改变了和现有的文件到已经存在的压缩文件；

-j：支持bzip2解压文件；

-v：显示操作过程；

-l：文件系统边界设置；

-k：保留原有文件不覆盖；

-m：保留文件不被覆盖；

-w：确认压缩文件的正确性；

-p或--same-permissions：用原来的文件权限还原文件；

-P或--absolute-names：文件名使用绝对名称，不移除文件名称前的“/”号；

-N <日期格式> 或 --newer=<日期时间>：只将较指定日期更新的文件保存到备份文件里；

--exclude=<范本样式>：排除符合范本样式的文件。

**参数**

文件或目录：指定要打包的文件或目录列表。

**实例**

将文件全部打包成tar包：

tar -cvf log.tar log2012.log 仅打包，不压缩！

tar -zcvf log.tar.gz log2012.log 打包后，以 gzip 压缩

tar -jcvf log.tar.bz2 log2012.log 打包后，以 bzip2 压缩

在选项f之后的文件档名是自己取的，我们习惯上都用 .tar 来作为辨识。 如果加z选项，则以.tar.gz或.tgz来代表gzip压缩过的tar包；如果加j选项，则以.tar.bz2来作为tar包名。

查阅上述tar包内有哪些文件：

tar -ztvf log.tar.gz

由于我们使用 gzip 压缩的log.tar.gz，所以要查阅log.tar.gz包内的文件时，就得要加上z这个选项了。

将tar包解压缩：

tar -zxvf /opt/soft/test/log.tar.gz

在预设的情况下，我们可以将压缩档在任何地方解开的

只将tar内的部分文件解压出来：

tar -zxvf /opt/soft/test/log30.tar.gz log2013.log

我可以透过tar -ztvf来查阅 tar 包内的文件名称，如果单只要一个文件，就可以透过这个方式来解压部分文件！

文件备份下来，并且保存其权限：

tar -zcvpf log31.tar.gz log2014.log log2015.log log2016.log

这个-p的属性是很重要的，尤其是当您要保留原本文件的属性时。

在文件夹当中，比某个日期新的文件才备份：

tar -N "2012/11/13" -zcvf log17.tar.gz test

备份文件夹内容是排除部分文件：

tar --exclude scf/service -zcvf scf.tar.gz scf/\*

其实最简单的使用 tar 就只要记忆底下的方式即可：

bzip2类型文件：

**压　缩**：tar -jcv -f filename.tar.bz2 要被压缩的文件或目录名称

**查　询**：tar -jtv -f filename.tar. bz2

**解压缩**：tar -jxv -f filename.tar. bz2 -C 欲解压缩的目录

解压到当前目录，直接 tar -jxf filename.tar.gz

gzip类型文件：

**压　缩**：tar -zcv -f filename.tar.gz 要被压缩的文件或目录名称

**查　询**：tar -ztv -f filename.tar.gz 2

**解压缩**：tar -zxv -f filename.tar.gz -C 欲解压缩的目录

其中，v选项只是显示指令执行过程，不用也可以。

# 硬件 | 监测 | Shell | 内核

## Shell内建命令

### kill

kill命令用来删除执行中的程序或工作。kill可将指定的信息送至程序。预设的信息为SIGTERM(15),可将指定程序终止。若仍无法终止该程序，可使用SIGKILL(9)信息尝试强制删除程序。程序或工作的编号可利用ps指令或job指令查看。

格式：kill [选项] 进程号

范例：

1.kill –s SIGKILL 4096

杀死4096号进程

## 性能检测与优化

### top

格式：top

范例：

1.top

查看系统中的进程对cpu、内存等的占用情况。

## 内核与模块管理

### uname

uname命令用于打印当前系统相关信息（内核版本号、硬件架构、主机名称和操作系统类型等）。

**语法**

uname (选项)

**选项**

-a或--all： 显示全部的信息；

-m或--machine： 显示电脑类型；

-n或-nodename： 显示在网络上的主机名称；

-r或--release： 显示操作系统的发行编号；

-s或--sysname： 显示操作系统名称；

-v： 显示操作系统的版本；

-p或--processor： 输出处理器类型或"unknown"；

-i或--hardware-platform： 输出硬件平台或"unknown"；

-o或--operating-system： 输出操作系统名称；

--help： 显示帮助；

--version： 显示版本信息。

**实例**

使用uname命令查看全部信息：

[root@localhost ~]# uname

Linux

[root@localhost ~]# uname -a

Linux localhost.localdomain 3.10.0-514.el7.x86\_64 #1 SMP Tue Nov 22 16:42:41 UTC 2016 x86\_64 x86\_64 x86\_64 GNU/Linux

[root@localhost ~]# uname -m

x86\_64

[root@localhost ~]# uname -n

localhost.localdomain

[root@localhost ~]# uname -r

3.10.0-514.el7.x86\_64

[root@localhost ~]# uname -s

Linux

[root@localhost ~]# uname -v

#1 SMP Tue Nov 22 16:42:41 UTC 2016

[root@localhost ~]# uname -p

x86\_64

[root@localhost ~]# uname -i

x86\_64

[root@localhost ~]# uname -o

GNU/Linux

[root@localhost ~]# uname --version

uname (GNU coreutils) 8.22

Copyright (C) 2013 Free Software Foundation, Inc.

许可证：GPLv3+：GNU 通用公共许可证第3 版或更新版本<http://gnu.org/licenses/gpl.html>。

本软件是自由软件：您可以自由修改和重新发布它。

在法律范围内没有其他保证。

由David MacKenzie 编写。

## 磁盘管理

### df

df命令用于显示磁盘分区上的可使用的磁盘空间。默认显示单位为KB。可以利用该命令来获取硬盘被占用了多少空间，目前还剩下多少空间等信息。

**语法**

df (选项) (参数)

**选项**

-a或--all：包含全部的文件系统；

--block-size=<区块大小>：以指定的区块大小来显示区块数目；

-h或--human-readable：以可读性较高的方式来显示信息；

-H或--si：与-h参数相同，但在计算时是以1000 Bytes为换算单位而非1024 Bytes；

-i或--inodes：显示inode的信息；

-k或--kilobytes：指定区块大小为1024字节；

-l或--local：仅显示本地端的文件系统；

-m或--megabytes：指定区块大小为1048576字节；

--no-sync：在取得磁盘使用信息前，不要执行sync指令，此为预设值；

-P或--portability：使用POSIX的输出格式；

--sync：在取得磁盘使用信息前，先执行sync指令；

-t<文件系统类型>或--type=<文件系统类型>：仅显示指定文件系统类型的磁盘信息；

-T或--print-type：显示文件系统的类型；

-x<文件系统类型>或--exclude-type=<文件系统类型>：不要显示指定文件系统类型的磁盘信息；

--help：显示帮助；

--version：显示版本信息。

**参数**

文件：指定文件系统上的文件。

**实例**

查看系统磁盘设备，默认是KB为单位：

[root@localhost home]# df

文件系统 1K-块 已用 可用 已用% 挂载点

/dev/mapper/cl-root 17811456 5993112 11818344 34% /

devtmpfs 917984 0 917984 0% /dev

tmpfs 933632 156 933476 1% /dev/shm

tmpfs 933632 17352 916280 2% /run

tmpfs 933632 0 933632 0% /sys/fs/cgroup

/dev/sda1 1038336 176296 862040 17% /boot

tmpfs 186728 28 186700 1% /run/user/1000

/dev/sr0 4276440 4276440 0 100% /run/media/czha/CentOS 7 x86\_64

使用-h选项以KB以上的单位来显示，可读性高：

[root@localhost home]# df -h

文件系统 容量 已用 可用 已用% 挂载点

/dev/mapper/cl-root 17G 5.8G 12G 34% /

devtmpfs 897M 0 897M 0% /dev

tmpfs 912M 156K 912M 1% /dev/shm

tmpfs 912M 17M 895M 2% /run

tmpfs 912M 0 912M 0% /sys/fs/cgroup

/dev/sda1 1014M 173M 842M 17% /boot

tmpfs 183M 28K 183M 1% /run/user/1000

/dev/sr0 4.1G 4.1G 0 100% /run/media/czha/CentOS 7 x86\_64

查看全部文件系统：

[root@localhost home]# df -a

### du

du命令也是查看使用空间的，但是与df命令不同的是Linux du命令是**对文件和目录**磁盘**使用的空间的查看**，还是和df命令有一些区别的。

**语法**

du [选项] [文件]

**选项**

-a或-all 显示目录中个别文件的大小。

-b或-bytes 显示目录或文件大小时，以byte为单位。

-c或--total 除了显示个别目录或文件的大小外，同时也显示所有目录或文件的总和。

-k或--kilobytes 以KB(1024bytes)为单位输出。

-m或--megabytes 以MB为单位输出。

-s或--summarize 仅显示总计，只列出最后加总的值。

-h或--human-readable 以K，M，G为单位，提高信息的可读性。

-x或--one-file-xystem 以一开始处理时的文件系统为准，若遇上其它不同的文件系统目录则略过。

-L<符号链接>或--dereference<符号链接> 显示选项中所指定符号链接的源文件大小。

-S或--separate-dirs 显示个别目录的大小时，并不含其子目录的大小。

-X<文件>或--exclude-from=<文件> 在<文件>指定目录或文件。

--exclude=<目录或文件> 略过指定的目录或文件。

-D或--dereference-args 显示指定符号链接的源文件大小。

-H或--si 与-h参数相同，但是K，M，G是以1000为换算单位。

-l或--count-links 重复计算硬件链接的文件。

**实例**

显示目录或者文件所占空间：

[root@localhost test]# du

608 ./test6

308 ./test4

4 ./scf/lib

4 ./scf/service/deploy/product

4 ./scf/service/deploy/info

12 ./scf/service/deploy

16 ./scf/service 4 ./scf/doc

4 ./scf/bin

32 ./scf

8 ./test3

1288 .

只显示当前目录下面的子目录的目录大小和当前目录的总的大小，最下面的1288为当前目录的总大小

显示指定文件所占空间：

[root@localhost test]# du log2012.log

300 log2012.log

查看指定目录的所占空间：

[root@localhost test]# du scf

4 scf/lib

4 scf/service/deploy/product

4 scf/service/deploy/info

12 scf/service/deploy

16 scf/service 4 scf/doc

4 scf/bin 32 scf

显示多个文件所占空间：

[root@localhost test]# du log30.tar.gz log31.tar.gz

4 log30.tar.gz

4 log31.tar.gz

只显示总和的大小:

[root@localhost test]# du -s

1288 .

[root@localhost test]# du -s scf

32 scf

[root@localhost test]# cd ..

[root@localhost soft]# du -s test

1288 test