

简介

简介说明

随意写一下。

快速开始

高频Linux命令大总结

- [Linux-commands-cheat-sheet-by-PhoenixNAP.pdf](#)

关机/重启/注销

常用命令	作用
shutdown -h now	即刻关机
shutdown -h 10	10分钟后关机
shutdown -h 11:00	11: 00关机
shutdown -h +10	预定时间关机 (10分钟后)
shutdown -c	取消指定时间关机
shutdown -r now	重启
shutdown -r 10	10分钟之后重启
shutdown -r 11:00	定时重启
reboot	重启
init 6	重启
init 0	立刻关机
telinit 0	关机
poweroff	立刻关机
halt	关机
sync	buff数据同步到磁盘
logout	退出登录Shell

注意点：比如同样是关机， shutdown、 poweroff、 halt、 init 0有什么区别呢？这个有兴趣可以自己了解一下，它们是有区别的。

系统信息和性能查看

这里命令其实平时用得是非常之多的，因为一旦系统或者后台服务除了问题，我们经常要登上去查看，包括很多的系统信息，比如：系统版本、内核版本、处理器架构、计算机名、环境变量、用户情况、负载情况、内存用量、磁盘信息、进程、网络连接...

常用命令	作用
uname -a	查看内核/OS/CPU信息
uname -r	查看内核版本
uname -m	查看处理器架构
arch	查看处理器架构
hostname	查看计算机名
who	显示当前登录系统的用户
who am i	显示登录时的用户名
whoami	显示当前用户名
cat /proc/version	查看linux版本信息
cat /proc/cpuinfo	查看CPU信息
cat /proc/interrupts	查看中断
cat /proc/loadavg	查看系统负载
uptime	查看系统运行时间、用户数、负载
env	查看系统的环境变量
lsusb -tv	查看系统USB设备信息
lspci -tv	查看系统PCI设备信息
lsmod	查看已加载的系统模块
grep MemTotal /proc/meminfo	查看内存总量
grep MemFree /proc/meminfo	查看空闲内存量
free -m	查看内存用量和交换区用量
date	显示系统日期时间
cal 2021	显示2021日历表
top	动态显示cpu/内存/进程等情况
vmstat 1 20	每1秒采一次系统状态，采20次
iostat	查看io读写/cpu使用情况
sar -u 1 10	查询cpu使用情况（1秒一次，共10次）
sar -d 1 10	查询磁盘性能

磁盘和分区

这是和日常使用息息相关的一些常用命令，在Windows系统里面我们点点鼠标，图形化界面上就能查看，但是在Linux中我们应该熟练掌握用命令的方式来查看，比如：各种分区信息查看、磁盘使用情况、文件和目录大小、各种挂载和卸载...

常用命令	作用
fdisk -l	查看所有磁盘分区
swapon -s	查看所有交换分区
df -h	查看磁盘使用情况及挂载点
df -hl	查看磁盘剩余空间
du -sh /dir	查看指定目录大小
du -sk * sort -rn	从高到低依次显示文件和目录大小
mount /dev/hda2 /mnt/hda2	挂载hda2盘
mount -t ntfs /dev/sdc1 /mnt/usbhd1	指定文件系统类型挂载（如ntfs）
mount -o loop xxx.iso /mnt/cdrom	挂载iso文件
mount /dev/sda1 /mnt/usbdisk	挂载usb盘/闪存设备
umount -v /dev/sda1	通过设备名卸载
umount -v /mnt/mymnt	通过挂载点卸载
fuser -km /mnt/hda1	强制卸载(慎用)

用户和用户组

Linux系统里用户组和用户本身也是一个非常重要的概念，这部分命令主要就是关于：用户的CRUD、用户组的CURD、然后还包括查用户、切换用户、改密码、查用户登录日志...

常用命令	作用
useradd codesheep	创建用户
userdel -r codesheep	删除用户
usermod -g group_name user_name	修改用户的组
usermod -aG group_name user_name	将用户添加到组
usermod -s /bin/ksh -d /home/codepig -g dev codesheep	修改用户codesheep的登录Shell、主目录以及用户组
groups test	查看test用户所在的组
groupadd group_name	创建用户组
groupdel group_name	删除用户组
groupmod -n new_name old_name	重命名用户组
su - user_name	完整切换到一个用户环境
passwd	修改口令
passwd codesheep	修改某用户的口令
w	查看活动用户
id codesheep	查看指定用户信息
last	查看用户登录日志
crontab -l	查看当前用户的计划任务
cut -d: -f1 /etc/passwd	查看系统所有用户
cut -d: -f1 /etc/group	查看系统所有组

网络和进程管理

我们作为一个后端开发，这些命令平时用到的概率很大，比如：查看网络、查看连接、查看端口服务、配置网卡/防火墙/路由表/DNS，查看和过滤进程，采集系统状态，还有一些系统性能的监控和排查命令...等等。这些命令，在后端开发连到公司服务器上去排查问题时就很有用了，用得也很频繁。

常用命令	作用
ifconfig	查看网络接口属性
ifconfig eth0	查看某网卡的配置
route -n	查看路由表
netstat -lntp	查看所有监听端口
netstat -antp	查看所有已经建立的连接
netstat -lntp	查看TCP/UDP的状态信息
ifup eth0	启用eth0网络设备
ifdown eth0	禁用eth0网络设备
iptables -L	查看iptables规则
ifconfig eth0 192.168.1.1 netmask 255.255.255.0	配置ip地址
dhclient eth0	以dhcp模式启用eth0
route add -net 0/0 gw Gateway_IP	配置默认网关
route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1	配置静态路由到达网络'192.168.0.0/16'
route del 0/0 gw Gateway_IP	删除静态路由
hostname	查看主机名
host www.codesheep.cn	解析主机名
nslookup www.codesheep.cn	查询DNS记录，查看域名解析是否正常
ps -ef	查看所有进程
ps -ef grep codesheep	过滤出你需要的进程
kill -s name	kill指定名称的进程
kill -s pid	kill指定pid的进程
top	实时显示进程状态
vmstat 1 20	每1秒采一次系统状态，采20次
iostat	查看io读写/cpu使用情况
sar -u 1 10	查询cpu使用情况（1秒一次，共10次）
sar -d 1 10	查询磁盘性能

常见系统服务命令

这类命令平时接触得也很多，比如我们自己在安装和部署各种各样的基础编程环境和服务时就常用，比如装JDK、MySQL数据库、redis缓存、nginx服务器...

常用命令	作用
chkconfig --list	列出系统服务
service <服务名> status	查看某个服务
service <服务名> start	启动某个服务
service <服务名> stop	终止某个服务
service <服务名> restart	重启某个服务
systemctl status <服务名>	查看某个服务
systemctl start <服务名>	启动某个服务
systemctl stop <服务名>	终止某个服务
systemctl restart <服务名>	重启某个服务
systemctl enable <服务名>	开启自启动
systemctl disable <服务名>	关闭自启动

文件和目录操作

这类命令全部是Linux系统使用的基本操作，也是平时用到频率最高的一些命令，关于文件操作的、关于目录操作的、关于路径的。比如：对文件和目录的各种高频操作，创建、查看、查找、删除、重命名、复制、软连接、快速定位和查找... 等等。都是些高频的实用命令。

常用命令	作用	
cd <目录名>	进入某个目录	
cd ..	回上级目录	
cd ../../	回上两级目录	
cd	进个人主目录	
cd -	回上一步所在目录	
pwd	显示当前路径	
ls	查看文件目录列表	
ls -F	查看目录中内容（显示是文件还是目录）	
ls -l	查看文件和目录的详情列表	
ls -a	查看隐藏文件	
ls -lh	显示权限	
ls -lSr	more	按大小查看文件/目录
tree	查看文件和目录的树形结构	
mkdir <目录名>	创建目录	

命令	作用
mkdir -p /tmp/dir1/dir2	创建目录树
rm -f file1	删除'file1'文件
rmdir dir1	删除'dir1'目录
rm -rf dir1	删除'dir1'目录和其内容
rm -rf dir1 dir2	同时删除两个目录及其内容
mv old_dir new_dir	重命名/移动目录
cp file1 file2	复制文件
cp dir/* .	复制某目录下的所有文件至当前目录
cp -a dir1 dir2	复制目录
cp -a /tmp/dir1 .	复制一个目录至当前目录
ln -s file1 link1	创建指向文件/目录的软链接
ln file1 link1	创建指向文件/目录的物理链接
find / -name file1	从跟目录开始搜索文件/目录
find / -user user1	搜索用户user1的文件/目录
find /dir -name *.bin	在目录/dir中搜带有.bin后缀的文件
locate 关键词	快速定位文件
locate *.mp4	寻找.mp4结尾的文件
whereis halt	显示某二进制文件/可执行文件的路径
which halt	查找系统目录下的二进制文件
chmod ugo+rwx dir1	设置目录所有者(u)、群组(g)及其他(o)的读(r)写(w)执行(x)权限
chmod go-rwx dir1	移除群组(g)与其他(o)对目录的读写执行权限
chown user1 file1	改变文件的所有者属性
chown -R user1 dir1	改变目录的所有者属性
chgrp group1 file1	改变文件群组
chown user1:group1 file1	改变文件的所有人和群组

文件查看和处理

这部分命令主要针对性地关于文件处理或者说文本处理，比如我们作为一个程序员，那操作最多的就是代码源文件，各种花式查看、比较、增加、删除、替换、合并... 等等一系列快速操作。

常用命令	作用	
cat file1	查看文件内容	
cat -n file1	查看内容并标示行数	
cat xxx.txt	awk 'NR%2==1'	查看文件中的所有奇数行
tac file1	从最后一行开始反看文件内容	
more file1	查看一个长文件的内容	
less file1	类似more命令，但允许反向操作	
head -2 file1	查看文件前两行	
tail -2 file1	查看文件后两行	
tail -f /log/msg	实时查看添加到文件中的内容	
grep codesheep hello.txt	在文件hello.txt中查找关键词codesheep	
grep ^sheep hello.txt	在文件hello.txt中查找以sheep开头的内容	
grep [0-9] hello.txt	选择hello.txt文件中所有包含数字的行	
sed 's/s1/s2/g' hello.txt	将hello.txt文件中的s1替换成s2	
sed '/^\$/d' hello.txt	从hello.txt文件中删除所有空白行	
sed '/ *#/d; /^\$/d' hello.txt	从hello.txt文件中删除所有注释和空白行	
sed -e '1d' hello.txt	从文件hello.txt 中排除第一行	
sed -n '/s1/p' hello.txt	查看只包含关键词"s1"的行	
sed -e 's/ *\$//'" hello.txt	删除每一行最后的空白字符	
sed -e 's/s1//g' hello.txt	从文档中只删除词汇s1并保留剩余全部	
sed -n '1,5p;5q' hello.txt	查看从第一行到第5行内容	
sed -n '5p;5q' hello.txt	查看第5行	
paste file1 file2	合并两个文件或两栏的内容	
paste -d '+' file1 file2	合并两个文件或两栏的内容，中间用"+"区分	
sort file1 file2	排序两个文件的内容	
sort file1 file2	uniq	取出并集(重复的行只保留一份)
sort file1 file2	uniq -u	删除交集，留下其他行
sort file1 file2	uniq -d	取交集
comm -1 file1 file2	比较两个文件的内容(去除'file1'所含内 容)	

常用命令	作用
comm -2 file1 file2	比较两个文件的内容(去除'file2'所含内容)
comm -3 file1 file2	比较两个文件的内容(去除两文件共有部分)

打包和解压

这部分内容主要关于文件或者目录的打包压缩和解压，涉及好几种压缩包格式和文件，这部分命令在平时用得也是非常非常频繁的。

常用命令	作用
zip xxx.zip file	压缩至zip包
zip -r xxx.zip file1 file2 dir1	将多个文件+目录压成zip包
unzip xxx.zip	解压zip包
tar -cvf xxx.tar file	创建非压缩tar包
tar -cvf xxx.tar file1 file2 dir1	将多个文件+目录打tar包
tar -tf xxx.tar	查看tar包的内容
tar -xvf xxx.tar	解压tar包
tar -xvf xxx.tar -C /dir	将tar包解压至指定目录
tar -cvfj xxx.tar.bz2 dir	创建bz2压缩包
tar -jxvf xxx.tar.bz2	解压bz2压缩包
tar -cvfz xxx.tar.gz dir	创建gzip压缩包
tar -zxfv xxx.tar.gz	解压gzip压缩包
bunzip2 xxx.bz2	解压bz2压缩包
bzip2 filename	压缩文件
gunzip xxx.gz	解压gzip压缩包
gzip filename	压缩文件
gzip -9 filename	最大程度压缩

然后接下来就是一些常见的包管理器命令。首先什么是包管理器，大家应该都清楚。

这么说吧，如果没有包管理器这个东西的存在，那估计仅仅是Linux系统上的软件安装，怕是就要劝退很多用户了，因为很多的软件依赖处理会让人抓狂。因此简单来说，我们可以将包管理器理解为，用来为Linux系统上的软件安装、卸载、升级、查询提供支持的组件，所以对于用户使用来说，一般就是一组工具命令集。

我们平常使用最广的比如，红帽子的包结构RPM包管理器，像RedHat、CentOS等系统都在用，典型的命令就是rpm命令、yum命令；然后就是DPKG包管理器，像Debian、Ubuntu等系统都用，典型的命令比如

dpkg命令、apt软件工具。

rpm包管理命令

常用命令	作用
rpm -qa	查看已安装的rpm包
rpm -q pkg_name	查询某个rpm包
rpm -q --whatprovides xxx	显示xxx功能是由哪个包提供的
rpm -q --whatrequires xxx	显示xxx功能被哪个程序包依赖的
rpm -q --changelog xxx	显示xxx包的更改记录
rpm -qi pkg_name	查看一个包的详细信息
rpm -qd pkg_name	查询一个包所提供的文档
rpm -qc pkg_name	查看已安装rpm包提供的配置文件
rpm -ql pkg_name	查看一个包安装了哪些文件
rpm -qf filename	查看某个文件属于哪个包
rpm -qR pkg_name	查询包的依赖关系
rpm -ivh xxx.rpm	安装rpm包
rpm -ivh --test xxx.rpm	测试安装rpm包
rpm -ivh --nodeps xxx.rpm	安装rpm包时忽略依赖关系
rpm -e xxx	卸载程序包
rpm -Fvh pkg_name	升级确定已安装的rpm包
rpm -Uvh pkg_name	升级rpm包(若未安装则会安装)
rpm -V pkg_name	RPM包详细信息校验

yum包管理命令

常用命令	作用
yum repolist enabled	显示可用的源仓库
yum search pkg_name	搜索软件包
yum install pkg_name	下载并安装软件包
yum install --downloadonly pkg_name	只下载不安装
yum list	显示所有程序包
yum list installed	查看当前系统已安装包
yum list updates	查看可以更新的包列表
yum check-update	查看可升级的软件包
yum update	更新所有软件包
yum update pkg_name	升级指定软件包
yum deplist pkg_name	列出软件包依赖关系
yum remove pkg_name	删除软件包
yum clean all	清除缓存
yum clean packages	清除缓存的软件包
yum clean headers	清除缓存的header

dpkg包管理命令

常用命令	作用
dpkg -c xxx.deb	列出deb包的内容
dpkg -i xxx.deb	安装/更新deb包
dpkg -r pkg_name	移除deb包
dpkg -P pkg_name	移除deb包(不留配置)
dpkg -l	查看系统中已安装deb包
dpkg -l pkg_name	显示包的大致信息
dpkg -L pkg_name	查看deb包安装的文件
dpkg -s pkg_name	查看包的详细信息
dpkg –unpack xxx.deb	解开deb包的内容

apt软件工具

常用命令	作用
apt-cache search pkg_name	搜索程序包
apt-cache show pkg_name	获取包的概览信息
apt-get install pkg_name	安装/升级软件包
apt-get purge pkg_name	卸载软件（包括配置）
apt-get remove pkg_name	卸载软件（不包括配置）
apt-get update	更新包索引信息
apt-get upgrade	更新已安装软件包
apt-get clean	清理缓存

参考资料

- 硬核！高频Linux命令大总结，建议收藏~ - SegmentFault 思否 [↗](#)
- 常考的 21 条 Linux 命令 [↗](#)
- [linux-command](#) [url](#) ↗ 这个开源项目是 Linux 命令大全搜索工具，当前搜集了 570 多个 Linux 命令，主要内容包含：Linux 基础命令分类、Linux 学习资源整理（社区网站、知识相关、软件工具、开源镜像站点、游戏玩家发行版）等；
- [Awesome-Linux-Software](#) [url](#) ↗ 这个开源项目适用于所有人员，主要内容包含：Linux 应用程序、命令行应用程序、桌面环境、窗口管理器等多种软件工具和其他资料的列表集合，可以说内容是非常的丰富，而且这个开源项目是一名大学生在校期间的所总结出来的，真的不一般。
- [How-To-Secure-A-Linux-Server](#) ↗ 这个开源项目的目的是教您如何保护 Linux 服务器安全的方法指南，也希望能从中教给您一些有关安全性及其重要性的知识，包括：在你开始使用前的一些指南、SSH 远程服务器需要注意的点、网络、基础、审计方面的操作以及危险操作的注意事项等，同时希望你通过这个开源项目能够对 Linux 服务器安全有一定的认识并运用到实际情况之中。
- [explainshell](#) ↗ explainshell 是一个可以解析 Linux 命令的网站，它可以给出命令的解释和其参数的解释。对 Linux 小白来说，可以有效的将一条长命令进行拆分加以理解，很有帮助。
- [the-practical-linux-hardening-guide](#) ↗ 这个开源项目是一份详细介绍了创建安全 Linux 生产系统所涉及的规划和工具指南。
- [Linux-Tutorial](#) [在线阅读](#) ↗ 这个开源项目是 Java 程序员眼中的 Linux，主要内容包含：Linux / Ubuntu 介绍与安装、Linux 环境下的基础命令操作、Vim 的安装 / 配置 / 快捷键、日常维护与监控、Linux 环境下的各应用服务安装和配置、高可用、黑客入侵检查等；
- [linuxtools_rst](#) [在线阅读](#) ↗ 这个开源项目的作者是 大CC，专注于 Linux 工具最常用的用法并希望读者能够应用到实际工作中。主要内容包含：Linux 基础（命令、文件与目录管理、文本处理、磁盘 / 进程管理、性能监控、网络工具）、Linux 工具进阶（程序构建、调试、优化）、工具参考等；
- [instantbox](#) ↗ 这个开源项目主要是通过在任何浏览器的即时 Web Shell 访问来启动临时 Linux 系统。目前支持 Ubuntu、CentOS、Arch Linux、Debian、Fedora 和 Alpine 的各种版本。
- [linuxupskillchallenge](#) [课程网址](#) ↗ 这个开源项目是一份开源的 Linux 服务器管理教程，主要包括 20 课的所有源材料，该课程可以让开发者在通过一个月时间快速掌握 Linux 基础使用技巧，而且这份教程过去是付费的，不过现在不仅免费而且还开源了。
- [TLCL](#) [在线阅读](#) ↗ 这个开源项目是：快乐的 Linux 命令行，主要内容包含：Shell、配置文件和 Shell 环境、常见任务和基本工具、编写 Shell 脚本（if、while、case、for 语句）等；
- [Linux命令大全](#) ↗
- [Linux系统大全](#) ↗

linux命令英文全称

转载自网络

su: Swith user 切换用户，切换到root用户

cat: Concatenate 串联

uname: Unix name 系统名称

df: Disk free 空余硬盘

du: Disk usage 硬盘使用率

chown: Change owner 改变所有者

chgrp: Change group 改变用户组

ps: Process Status 进程状态

tar: Tape archive 解压文件

chmod: Change mode 改变模式

umount: Unmount 卸载

ldd: List dynamic dependencies 列出动态相依

insmod: Install module 安装模块

rmmmod: Remove module 删除模块

lsmod: List module 列表模块

alias :Create your own name for a command

bash :GNU Bourne-Again Shell linux内核

grep:global regular expression print

httpd :Start Apache

ipcalc :Calculate IP information for a host

ping :Send ICMP ECHO_Request to network hosts

reboot: Restart your computer

sudo:Superuser do

/bin = BINaries

/dev = DEVices

/etc = ETCEterna

/lib = LIBrary

/proc = PROCesses

/sbin = Superuser BINaries

/tmp = TeMPorary

/usr = Unix Shared Resources

/var = VARiable ?

FIFO = First In, First Out

GRUB = GRand Unified Bootloader

IFS = Internal Field Separators

LILO = LIinux LOader

MySQL = My最初作者的名字SQL = Structured Query Language

PHP = Personal Home Page Tools = PHP Hypertext Preprocessor

PS = Prompt String

Perl = "Practical Extraction and Report Language" = "Pathologically Eclectic Rubbish Lister"

Python Monty Python's Flying Circus

Tcl = Tool Command Language

Tk = ToolKit

VT = Video Terminal

YaST = Yet Another Setup Tool

apache = "a patchy" server

apt = Advanced Packaging Tool

ar = archiver

as = assembler

bash = Bourne Again SHell

bc = Basic (Better) Calculator

bg = BackGround

cal = CALendar

cat = CATenate

cd = Change Directory

chgrp = CHange GRouP

chmod = CHange MODe
chown = CHange OWNer
chsh = CHange SHell
cmp = compare
cobra = Common Object Request Broker Architecture
comm = common
cp = CoPy
cpio = CoPy In and Out
cpp = C Pre Processor
cups = Common Unix Printing System
cvs = Current Version System
daemon = Disk And Execution MONitor
dc = Desk Calculator
dd = Disk Dump
df = Disk Free
diff = DIFFERENCE
dmesg = diagnostic message
du = Disk Usage
ed = editor
egrep = Extended GREP
elf = Extensible Linking Format
elm = ELectronic Mail
emacs = Editor MACroS
eval = EVALuate
ex = EXtended
exec = EXECute
fd = file descriptors
fg = ForeGround
fgrep = Fixed GREP
fmt = format

fsck = File System Check

fstab = FileSystem TABLE

fvwm = F*** Virtual Window Manager

gawk = GNU AWK

gpg = GNU Privacy Guard

groff = GNU troff

hal = Hardware Abstraction Layer

joe = Joe's Own Editor

ksh = Korn SHell

lame = Lame Ain't an MP3 Encoder

lex = LEXical analyser

lisp = LiSt Processing = Lots of Irritating Superfluous Parentheses

ln = LiNk

lpr = Line PRint

ls = list

lsof = LiSt Open Files

m4 = Macro processor Version 4

man = MANual pages

mawk = Mike Brennan's AWK

mc = Midnight Commander

mkfs = MaKe FileSystem

mknod = MaKe NODE

motd = Message of The Day

mozilla = MOsaic GodZILLA

mtab = Mount TABle

mv = MoVe

nano = Nano's ANOther editor

nawk = New AWK

nl = Number of Lines

nm = names

nohup = No HangUP

nroff = New ROFF

od = Octal Dump

passwd = PASSWorD

pg = pager

pico = PIne's message COnposition editor

pine = "Program for Internet News & Email" = "Pine is not Elm"

ping = Packet InterNet Grouper

pirntcap = PRINTer CAPability

popd = POP Directory

pr = pre

printf = PRINT Formatted

ps = Processes Status

pty = pseudo tty

pushd = PUSH Directory

pwd = Print Working Directory

rc = runcom = run command, shell

rev = REVerse

rm = ReMove

rn = Read News

roff = RunOFF

rpm = RPM Package Manager = RedHat Package Manager

rsh, rlogin, = Remote

rxvt = ouR XVT

sed = Stream EDitor

seq = SEQuence

shar = SHell ARchive

sln = S-Lang rn

ssh = Secure SHell

ssl = Secure Sockets Layer

stty = Set TTY

su = Substitute User

svn = SubVersioN

tar = Tape ARchive

tcsh = TENEX C shell

telnet = TEminaL over Network

termcap = terminal capability

terminfo = terminal information

tr = traslate

troff = Typesetter new ROFF

tsort = Topological SORT

tty = TeleTypewriter

twm = Tom's Window Manager

tz = TimeZone

udev = Userspace DEV

ulimit = User's LIMIT

umask = User's MASK

uniq = UNIQue

vi = VIsual = Very Inconvenient

vim = Vi IMproved

wall = write all

wc = Word Count

wine = WINE Is Not an Emulator

xargs = eXtended ARGuments

xdm = X Display Manager

xlfd = X Logical Font Description

xmms = X Multimedia System

xrdb = X Resources DataBase

xwd = X Window Dump

yacc = yet another compiler compiler

Linux系统启动U盘制作工具

UNetbootin

UNetbootin 让你创建 Ubuntu 或者其他 Linux 发行版的可引导 Live U 盘，而无需烧录 CD。

你既能让 UNetbootin 为你下载众多开箱即用的发行版，或者提供你自己的 Linux 的 .iso 文件。支持Linux 和windows系统，官网有安装方法。

官方网址: <https://unetbootin.github.io/>

Rufus(亲测过)

Rufus是一个实用程序，可帮助格式化和创建可启动的U盘。它对以下情况特别有用：您需要从可启动的 ISO（Windows, Linux, UEFI等）创建USB安装媒体，您需要在没有安装操作系统的系统上工作，你需要从DOS中刷新BIOS或其他固件，你想运行一个低级实用程序，尽管体积小，但Rufus提供您所需的一切！
目前之支持Windows平台

官方网站: https://rufus.ie/en_IE.html

universal-usb-installer

Universal USB Installer（通用USB安装程序）是一个自启动Linux U盘创建工具，您可从大量精选的Linux发行版中挑选一个安装到您的U盘上。通用USB安装程序使用方便，只需选择自启动Linux发行版，ISO文件，和您的U盘，单击“安装”即可。其他功能包括，持续保存（如果可用的话），以FAT32格式格式化U盘（推荐）确保一个干净的安装。安装完成后，您即拥有了一个安装了您所喜欢的Linux版本的自启动U盘。
目前之支持Windows平台。

官方网址: <https://www.pendrivelinux.com/universal-usb-installer-easy-as-1-2-3>

Ubuntu系统的自带软件 Startup Disk Creator（启动盘创建器）

这个是Ubuntu系统的自带软件，安装完Ubuntu系统会默认安装。

usb-creator-common: 使用CD或磁盘映像创建启动磁盘（通用文件） usb-creator-gtk: 使用CD或磁盘映像创建启动磁盘（适用于GNOME） usb-creator-kde: 使用CD或磁盘映像创建启动磁盘（适用于KDE）

没有安装可以通过下面的命令安装：

```
root@Ubuntu:~# apt install usb-creator-gtk
```

sh

ab - Apache服务器的性能测试工具

ab命令 是一个测试你 Apache http 服务器的工具，你可以通过这个工具，指定一个单位时间内向 apache 发出的请求数量来看看你的 Apache 和机器配合的性能如何。

ab指令是apache的性能测试工具，它可以测试当前apache服务器的运行性能，显示每秒中可以处理多少个http请求。

适用范围

RedHat RHEL Ubuntu CentOS Fedora

语法

```
ab [OPTION]
ab [ -A auth-username:password ] [ -c concurrency ] [ -C cookie-name=value
] [ -d ] [ -e csv-file ] [ -g gnuplot-file ] [ -h ] [ -H custom-header ] [
-i ] [ -k ] [ -n requests ] [ -p POST-file ] [ -P proxy-auth-user-
name:password ] [ -q ] [ -s ] [ -S ] [ -t timelimit ] [ -T content-type ]
[ -v verbosity ] [ -V ] [ -w ] [ -x <table>-attributes ] [ -X proxy[:port]
] [ -y <tr>-attributes ] [ -z <td>-attributes ] [http://]host-
name[:port]/path
```

sh

选项

```
-A name: pass          # 向服务器提供用户名和密码
-b                   # tcp连接的缓冲区大小
-c                   # 并发请求数目，默认1个
-C cookie-name=value # 添加cookie
-e csv-file          # 指定产生的csv文件
-g                   # 把测试结果写入到指定的gnuplot文件
-h                   # 显示帮助信息
-H                   # 为请求附加额外的头信息
-i                   # 执行http中的HEAD请求而不是GET
-k                   # 启动keepalive功能
-n                   # 指定测试会话中的请求次数
-p                   # 指定包含post数据的文件
-q                   # 如果请求数大于150，ab指令在处理10%或者100个请求后显示进度
-r                   # 当有socket接收错误的时候，不退出
-t                   # 指定测试的最大描述
-T                   # 指定Content-type信息
-v                   # 设置显示信息的级别
-V                   # 显示版本信息
-x                   # 设置table标记属性的字符串
-X                   # 设置代理服务器
-y                   # 设置tr标签
-z                   # 设置td标签
```

sh

举例

测试性能

```
[root@localhost ~]# ab -n 10 -c 10 http://www.qq.com/      #10个请求，并发10个
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/
Benchmarking www.qq.com (be patient)...apr_poll: The timeout specified has expired (70007)
Total of 6 requests completed
```

accept - 指示打印系统接受发往指定目标打印机的打印任务

accept命令 属于CUPS套件，用于指示打印系统接受发往指定目标打印机的打印任务。

accept指令用来设置允许向目标打印机发送打印任务。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
cupsaccept [ -E ] [ -U username ] [ -h hostname[:port] ] destination(s) sh
```

选项

-E	# 强制加密
-U	# 连接打印机的时候，发送用户名
-h	# 选择目标打印机ip和端口

举例

允许向目标打印机发送打印任务

```
[root@localhost /]$ accept printer01 sh
```

alias - 定义或显示别名

Alias不带参数或使用-p选项在标准输出上以“name=value”的形式打印别名列表。当提供参数时，为其值给定的每个名称定义一个别名。值中的尾随空格将导致在扩展别名时检查下一个单词是否替换别名。对于参数列表中没有为其提供值的每个名称，将打印别名的名称和值。别名返回true，除非给出没有为其定义别名的名称。

主要用途

- 简化较长的命令。
- 定义一个或多个别名。
- 修改一个或多个已定义别名的值。
- 显示一个或多个已定义别名。
- 显示全部已定义的别名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
alias [-p] [name[=value] ...]
```

sh

选项

```
-p # 以“key=val”的方式列出所有别名
```

sh

举例

查看已经定义别名

```
[root@localhost ~]$ alias
alias cp='cp -i'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mv='mv -i'
alias rm='rm -i'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

apachectl - Apache服务器前端控制工具

apachectl命令 是Apache的Web服务器前端控制工具，用以启动、关闭和重新启动Web服务器进程。

apachectl指令是apache http服务器的前端控制程序，可以协助控制apache服务的守护进程httpd。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
apachectl cmd
```

sh

选项

start	# 启动apache httpd守护进程	sh
restart	# 重启apache httpd守护进程	
stop	# 停止apache httpd守护进程	
status	# 显示apache服务的简要信息	
graceful	# 优雅的重启apache服务，它和restart不一样，不会中断当前已经打开的http连接，不会立刻关闭日志	
graceful-stop	# 优雅的停止apache服务，它和stop不一样，不会中断当前已经打开的http连接，不会立刻关闭日志	
configtest	# 运行apache配置语法检测	

举例

运行apache配置文件测试

```
[root@localhost ~]$ apachectl configtest #配置文件检测，没有错误  
httpd: Could not reliably determine the server's fully qualified domain name, using localhost.localdomain.  
Syntax OK
```

sh

重启服务

```
[root@localhost ~]$ apachectl restart #重启
```

sh

ar - 建立或修改备份文件，或是从备份文件中抽取文件

ar命令 是一个建立或修改备份文件，或是从备份文件中抽取文件的工具，ar可让您集合许多文件，成为单一的备份文件。在备份文件中，所有成员文件皆保有原来的属性与权限。

ar指令可以创建、修改库，也可以从库中提取单个模块。库是一个单独的文件，里面包含了按照特定结构组织起来的其他文件，我们称作member。归档文件通常是一个二进制文件，我们一般将归档文件当作库来使用。原始文件的内容、模式(权限)、时间戳、所有者和组保存在存档中，并可在提取时恢复。

GNU ar可以维护其成员具有任意长度的名称的档案；但是，根据您的系统上对ar的配置方式，可以对成员名长度进行限制，以便与其他工具维护的存档格式兼容。如果存在，限制通常是15个字符(典型的与a.out相关的格式)或16个字符(典型的与coff相关的格式)。

当您指定修饰符的时候，ar会为存档中可重定位的对象模块中定义的符号创建一个索引。创建之后，每当ar对其内容进行更改时，这个索引就会在存档中更新(除了Q更新操作外)。具有这样一个索引的归档可以加速链接到库，并允许库中的例程相互调用，而不考虑它们在存档中的位置。您可以使用“nm -s”或“nm --print-armap”列出此索引表。如果存档缺少表，则可以使用另一种称为ranlib的ar形式来添加表。

GNU ar可以随意创建一个瘦存档，其中包含一个符号索引和对档案成员文件的原始副本的引用。这样的存档对于构建用于本地构建的库非常有用，因为在本地构建中，可重新定位的对象将保持可用，而复制每个对象的内容只会浪费时间和空间。薄档案也是扁平的，因此将一个或多个档案添加到一个瘦归档中将单独添加嵌套归档的元素。存档元素的路径是相对于归档本身存储的。

GNU Ar被设计成与两个不同的设施兼容。您可以使用命令行选项来控制它的活动，比如Unix系统上的不同类型的ar；或者，如果您指定了单一命令行选项-M，您可以使用标准输入提供的脚本来控制它，比如MRI "librarian"程序。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
ar  [--plugin name]  [-X32_64]  [-]p[mod  [relpos]  [count]]  archive  [member...]
```

sh

选项

GNU ar允许您在第一个命令行参数中按任意顺序混合操作代码p和修饰符标志mod。如果您愿意，可以用“-”开始第一个命令行参数。p键字母指定要执行的操作；它可能是以下任一操作，但您必须仅指定其中一个操作：

```

c # 创建归档文件
d # 删除归档中的成员文件，将要删除的模块的名称指定为member。如果没有指定要删除的文件，则存档将保持不变
m # 改变成员文件在归档中的顺序。如果在多个成员中定义了一个符号，则存档中成员的排序会改变程序使用库链接的方式
p # 将存档的指定成员打印到标准输出文件。如果指定了v修饰符，请在将其内容复制到标准输出之前显示成员名。
q # 将文件附加在归档文件的最后。修饰符a、b和i不影响此操作；新成员总是放在归档的末尾。修饰符v在追加时使每个文件可选
r # 将文件‘member’插入到归档文件，如果归档中存在要插入的文件，那么就覆盖。此操作与q的不同之处在于，如果现有成员与之同名，则不插入
t # 显示归档文件包含的文件列表。通常只显示成员名称；如果还希望查看模式(权限)、时间戳、所有者、组和大小，则通过-t或-tv
x # 从归档文件中提取成员文件，您可以在此操作中使用v修饰符，在提取时请求ar列出每个名称。如果不指定成员，则提取所有成员

```

许多修饰符(Mod)可以紧跟p键，以指定操作行为的变化

```

a # 在存档的现有成员之后添加新文件。如果使用修饰符a，则现有存档成员的名称必须作为relpos参数出现在归档规范之后
b # 在存档的现有成员之前添加新文件。如果使用修饰符b，则现有存档成员的名称必须作为relpos参数出现在归档规范之前
c # 创建档案。当您请求更新时，如果指定的存档不存在，则始终创建它。但是，除非事先指定通过使用此修饰符来创建警报
D # 在确定性模式下操作。当添加文件和存档索引时，对UID、GID、时间戳使用零，并对所有文件使用一致的文件模式。这将导致ar命令与某些系统上的本机ar程序不兼容
f # 截断存档中的名称。GNU ar通常允许任意长度的文件名。这将导致它创建与某些系统上的本机ar程序不兼容的档案
i # 在存档的现有成员之前插入新文件。如果使用修饰符i，则现有存档成员的名称必须作为relpos参数出现在归档规范之前
l # 没有用的修饰符
N # 使用计数参数。如果存档中有多个具有相同名称的条目，则使用此方法。从存档中提取或删除给定名称的实例计数
o # 提取成员时保留成员的原始日期。如果不指定此修饰符，则从档案中提取的文件将加盖提取时间。
P # 在存档中匹配名称时使用完整路径名称。GNU ar不能创建具有完整路径名的存档(这类档案不是POSIX投诉)，但其他工具链可能支持
s # 将对象文件索引写入存档，或更新现有的索引，即使没有对归档进行其他更改。您可以在任何操作中使用此修饰符标志
S # 不要生成存档符号表。这可以加快构建一个大型库的几个步骤。结果存档不能与链接器一起使用。为了构建符号表，必须使用ar -S
T # 将指定的归档文件设置为瘦存档。如果它已经存在并且是一个常规存档，则现有成员必须与存档目录相同。
u # 通常，ar r将列出的所有文件插入存档中。如果只想插入比同名的现有成员更新的文件，请使用此修饰符。u修饰符仅适用于存档
v # 显示详细执行过程
V # 显示ar的版本

```

AR忽略初始选项-x32_64，以便与AIX兼容。此选项产生的行为是GNU ar的默认行为。ar不支持任何其他-X选项；特别是，它不支持-x32，这是AIX ar的默认设置。

可选的命令行开关“--plugin name”使ar加载名为name的插件，这增加了对更多文件格式的支持。只有在工具链已启用插件支持的情况下，此选项才可用。

@file选项，从文件中读取命令行选项。已读取的选项被插入以代替原始的@file选项。如果文件不存在或无法读取，则将按字面处理该选项，而不删除该选项。file中的选项用空格分隔。可以在选项中包含空格字符，方法是将整个选项包围在单引号或双引号中。任何字符(包括反斜杠)都可以通过前缀所包含的字符串来包括在反斜杠中。file本身可能包含额外的@file选项；任何此类选项都将被递归处理。

```
--plugin <p> - load the specified plugin
```

ar: 支持的目标: elf64-x86-64 elf32-i386 elf32-x86-64 a.out-i386-linux pei-i386 pei-x86-64 elf64-l1om
elf64-k1om elf64-little elf64-big elf32-little elf32-big plugin srec symbolsrec verilog tekhex binary ihex

举例

sh

```
创建归档文件
[sogrey@bogon newDir3]$ ll
总用量 16
-rw-----. 1 sogrey sogrey 38 1月 27 23:42 1.txt
-rw-----. 1 sogrey sogrey 62 1月 24 01:03 students.txt
-rw-----. 1 sogrey sogrey 11 1月 24 01:08 test2.txt
-rw-----. 1 sogrey sogrey 135 1月 24 01:05 test.txt
[sogrey@bogon newDir3]$ ar -rc mlib.a test.txt test2.txt # 创建归档
[sogrey@bogon newDir3]$ ls
1.txt mlib.a students.txt test2.txt test.txt
[sogrey@bogon newDir3]$ ar -t mlib.a # 查看归档的文件列表
test.txt
test2.txt
[sogrey@bogon newDir3]$
```

在归档中插入文件

sh

```
[sogrey@bogon newDir3]$ ar -rc mlib.a 1.txt # 插入1.txt, 此时归档中还没有1.txt
[sogrey@bogon newDir3]$ ar -t mlib.a
test.txt
test2.txt
1.txt
[sogrey@bogon newDir3]$
```

在归档的最后插入文件

sh

```
[sogrey@bogon newDir3]$ ar -q mlib.a students.txt # 是用选项q来追加到最后, 注意这个和r是不一样的
[sogrey@bogon newDir3]$ ar -t mlib.a
test.txt
test2.txt
1.txt
students.txt
[sogrey@bogon newDir3]$
```

从归档中删除文件,只删除第一个

sh

```
[sogrey@bogon newDir3]$ ar -d mlib.a students.txt
[sogrey@bogon newDir3]$ ar -t mlib.a
test.txt
test2.txt
1.txt
[sogrey@bogon newDir3]$
```

arch - 显示当前主机的硬件架构类型

Arch指令主要用于显示当前主机的硬件结构类型，我们可以看到它输出的结果有：i386、i486、mips、alpha等。

打印机器架构信息；arch 命令输出结果有：i386、i486、i586、alpha、sparc、arm、m68k、mips、ppc、i686等。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
arch [OPTION]
```

sh

选项

```
--help          # 显示帮助文档  
--version       # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon ~]$ arch  
x86_64  
[sogrey@bogon ~]$ arch --help  
用法: arch [选项]...  
输出机器的体系结构。  
  
--help  显示此帮助信息并退出  
--version  显示版本信息并退出  
  
GNU coreutils online help: <http://www.gnu.org/software/coreutils/>  
请向<http://translationproject.org/team/zh\_CN.html> 报告arch 的翻译错误  
要获取完整文档, 请运行: info coreutils 'arch invocation'  
[sogrey@bogon ~]$ arch --version  
arch (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 和Karel Zak 编写。  
[sogrey@bogon ~]$
```

sh

arp - arp 命令用于显示和修改 IP 到 MAC 转换表

arp 命令是 Address Resolution Protocol，地址解析协议，是通过解析网络层地址来找寻数据链路层地址的一个网络协议包中极其重要的网络传输协议。而该命令可以显示和修改 arp 协议解析表中的缓冲数据。

这个核心协议模块实现RFC826中定义的 Address Resolution Protocol [译注：即TCP/IP的第三层到第一层的地址转换协议]，用于在直接相连的网络中将第二层硬件地址和Ipv4 协议地址之间的转换。用户除非想对其进行配置，否则一般不会直接操作这个模块。

实际上，它提供对核心中其它协议的服务。

用户进程可以使用 packet(7) 的 sockets，收到 ARP 包（译注：一译分组）。还有一种机制是使用 netlink(7) sockets，在用户空间管理 ARP 缓存的机制。我们也可以通过 ioctl (2) 控制任意 PF_INET socket 上的 ARP 表

ARP 模块维护一个硬件地址到协议地址映射的缓存。这个缓存有大小限制，所以不常用的和旧的记录（Entry）将被垃圾收集器清除（garbage-collected），垃圾收集器永远不能删除标为永久的记录。我们可以使用ioctls直接操纵缓冲，并且其性状可以用下面定义的 sysctl 调节。

如果在限定的时间（见下面的sysctl）内，一条现存映射没有肯定反馈时，则认为相邻层的缓存记录失效。为了再次向目标发送数据，ARP将首先试着询问本地arp进程 app_solicit 次，获取更新了的 MAC（介质访问控制）地址。如果失败，并且旧的MAC地址是已知的，则发送 ucast_solicit 次的 unicast probe。如果仍然失败，则将向网络广播一个新的ARP请求，此时要有待发送数据的队列

如果 Linux 接到一个地址请求，而且该地址指向 Linux 转发的地址，并且接收接口打开了代理 arp 时，Linux 将自动添加一条非永久的代理 arp 记录；如果存在拒绝到目标的路由，则不添加代理 arp 记录。

arp指令用来管理系统的arp缓冲区，可以显示、删除、添加静态mac地址。ARP以各种方式操纵内核的 ARP缓存。主要选项是清除地址映射项并手动设置。为了调试目的，ARP程序还允许对ARP缓存进行完全转储。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
arp [-evn] [-H type] [-i if] -a [hostname]
arp [-v] [-i if] -d hostname [pub]
arp [-v] [-H type] [-i if] -s hostname hw_addr [temp]
arp [-v] [-H type] [-i if] -s hostname hw_addr [netmask nm] pub
arp [-v] [-H type] [-i if] -Ds hostname ifa [netmask nm] pub
arp [-vnD] [-H type] [-i if] -f [filename]
```

选项

```
-H type  
--hw-type type  
-t type          # 在设置或读取ARP缓存时，这个可选参数告诉ARP应该检查哪类条目。此参数的默  
-a [hostname], --all [hostname] # 显示本机的arp缓冲区内容  
-d hostname, --delete hostname # 从缓冲区删除指定的地址类型  
-D, -use-device      # 使用指定接口的mac地址  
-e                  # 使用Linux风格显示  
-i if, --device if    # 显示指定设备的arp缓冲区  
-s hostname hw_addr   # 设置指定主机的mac地址映射  
-f filename, --file filename # 类似于-s选项，只是这次地址信息是从filename设置的。数据文件的名称通常是指定的  
-n, --numeric        # 使用数字方式显示  
-v, --verbose        # 显示执行过程  
  
--help             # 显示帮助文档  
--version          # 显示命令版本信息
```

在所有需要主机名的地方，人们也可以用虚线小数点表示法输入IP地址。作为兼容性的特例，主机名和硬件地址的顺序可以交换。ARP缓存中的每个完整条目都将被标记为C标志。永久条目用M标记，已发布的条目带有P标志。

举例

添加静态映射

```
[root@localhost ~]$ arp -i eth0 -s 192.168.1.6 ff:ee:ee:ee:ee:ee #将目标ip地址映射固定mac  
[root@localhost ~]$ arp -a                         #查看arp缓冲区  
? (10.0.2.2) at 52:54:00:12:35:02 [ether] on eth0  
? (192.168.1.6) at ff:ee:ee:ee:ee:ee [ether] PERM on eth0
```

以数字方式显示

```
[root@localhost ~]$ arp -vn  
Address           Hwtype  Hwaddress          Flags Mask     Iface  
10.0.2.2          ether    52:54:00:12:35:02  C          eth0  
192.168.1.6       ether    ff:ee:ee:ee:ee:ee CM         eth0  
Entries: 2 Skipped: 0 Found: 2
```

arping - 通过发送ARP协议报文测试网络

arping命令是用于发送arp请求到一个相邻主机的工具，arping使用arp数据包，通过ping命令检查设备上的硬件地址。能够测试一个ip地址是否是在网络上已经被使用，并能够获取更多设备信息。功能类似于ping。

arping指令用于发送arp请求到一个相邻的主机，在指定网卡上发送arp请求指定地址，源地址使用-s指定。该指令可以直奔MAC地址，找出哪些地址被哪些电脑使用了。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
arping [-AbDfhqUV] [-c count] [-w deadling] [-s source] -I interface destination
```

sh

选项

-A	# 与-U相同，但是发送的是ARP RELAY报文
-b	# 只发送物理层的广播报文
-c count	# 指定发送测试数据包的次数
-D	# 重复地址检测模式
-f	# 如果目的主机有响应，立刻停止发送测试数据包
-h	# 显示帮助信息
-I interface	# 指定网络设备接口
-q	# 静默模式
-s source	# 指定发送测试包的源地址
-U	# 更新邻居主机arp缓存
-V	# 显示版本信息
-w	# 指定超时时间

sh

举例

向邻居主机发送请求

```
[root@localhost ~]$ arping -I eth0 192.168.1.8          #指定网卡，向目标ip发送请求
ARPING 192.168.1.8 from 192.168.1.9 eth0
Unicast reply from 192.168.1.8 [98:01:A7:9F:5E:9D]  0.817ms  #可以看到目标ip的mac地址
Unicast reply from 192.168.1.8 [98:01:A7:9F:5E:9D]  0.749ms
Unicast reply from 192.168.1.8 [98:01:A7:9F:5E:9D]  0.741ms
Unicast reply from 192.168.1.8 [98:01:A7:9F:5E:9D]  0.787ms
```

sh

arpwatch - 监听网络上ARP的记录

arpwatch命令 用来监听网络上arp的记录。

arpwatch指令可以监听网络设备和ip地址的对应关系，将发现的信息发送到系统日志"/var/log/message"。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
arpwatch [选项]
```

sh

选项

```
-d                # 打开调试模式，调试信息发送到终端  
-f file         # 设置arp记录的存储文件，默认arp.dat。开始使用这个指令之前，必须创建一个空arp.dat  
-i interface    # 指定网络接口  
-n                # 说明本地网络  
-N                # -N标志禁止报告  
-r                # 从指定的文件中读取arp记录  
-e                # 指定发送mail的目标用户，默认是root  
-s                # 指定发送mail的源用户，默认root
```

sh

报告信息

信息	说明
new activity	这对ethernet/ip地址对第一次被使用了六个月或更长时间。
new station	此ip地址以前从未使用过此以太网地址。
flip flop	以太网地址已从最近看到的地址改为第二最近看到的地址。(如果旧的或新的以太网地址是DECnet地址，并且不到24小时，则报告的电子邮件版本将被取消。)
changed ethernet address	主机切换到新的以太网地址。

系统日志信息

信息	说明
ethernet broadcast	主机的mac以太网地址是广播地址。
ip broadcast	主机的IP地址是广播地址。
bogon	源ip地址不是本地子网的本地地址。
ethernet broadcast	源mac或arp以太网地址均为1或全部为零。
ethernet mismatch	源Mac以太网地址与ARP数据包中的地址不匹配。
reused old ethernet address	以太网地址已从最近看到的地址更改为第三(或更大)最近看得最少的地址。(这类似于触发器。)
suppressed DECnet flip flop	由于两个地址之一是DECnet地址，“flip flop”报告被禁止。

举例

监听arp信息

```
[root@localhost ~]$ arpwatch -i eth0          # 监听网卡eth0
[root@localhost ~]$ tail -n 3 /var/log/messages    # 查看最近的日志信息
Sep 30 08:29:59 localhost arpwatch: listening on eth0
Sep 30 08:30:01 localhost arpwatch: new station 192.168.1.1 c8:41:29:f4:4a:20
Sep 30 08:30:12 localhost arpwatch: new station 192.168.1.9 8:0:27:14:33:57
You have new mail in /var/spool/mail/root
```

bc - 算术操作精密运算工具

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。

bc是一种算数语言，其语法和c语言类似，可以交互执行。通过命令行选项可以获得一个标准的数学库。如果请求，在处理任何文件之前定义数学库。BC从处理所有文件的代码开始。命令行中列出的文件按所列顺序排列。在处理完所有文件后，BC从标准输入中读取。所有代码都在读取时执行。(如果文件包含停止处理器的命令，BC将永远不会从标准输入中读取。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
bc [ -hlwsqv ] [long-options] [ file ... ]
```

sh

选项

```
-h, --help          # 帮助信息
-v, --version       # 显示命令版本信息
-l, --mathlib        # 定义标准数学库
-i, --interactive   # 强制交互
-w, --warn          # 显示POSIX的警告信息
-s, --standard       # 使用POSIX标准来处理
-q, --quiet          # 不显示欢迎信息
```

sh

说明

1. 数据

bc中最基本的元素是数字。数字是任意精度的数字。这种精度既包括整数部分，也包括分数部分。所有的数字在内部用十进制表示，所有的计算都用十进制来表示。(此版本截断除法和乘运算的结果。)数字有两个属性：长度和小数位。长度是数字中有效位的总数，小数位是小数点之后的有效位数。例如，0.000001的长度是6，小数位是6；1935.000的长度是7，小数位是3。

2. 变量

数字存储在两种类型的变量中，简单变量和数组。变量名称以字母开头，后面跟着任意数量的字母、数字和下划线。所有字母都必须小写。(全字母-数字名是扩展名。在POSIX bc中，所有名称都是一个小写字母。)变量的类型在上下文中是明确的，因为所有数组变量名称后面都会有方括号([])。

有四个特殊的变量：scale、ibase、obase和last。scale定义了一些操作在小数点之后是如何使用数字的，

默认值是0。ibase和obase定义输入和输出数字的转换基。默认值都是基数10。last(扩展)是一个变量。它是最后打印的数字的值。所有这些变量可能都有分配给它们的值，以及表达式中使用的值。

3. 注释

bc中的注释以字符"/ "开头，以字符" /"结尾。注释可以从任何地方开始，并显示为输入中的单个空格。(这会导致注释分隔其他输入项。例如，在变量名的中间找不到注释。)注释包括注释开始和结束之间的任何新行(行尾)。

为了支持bc脚本的使用，添加了行注释作为扩展。行注释从"#"字符开始，继续到行的结束。行结束字符不是注释的一部分，而是正常处理的。

如果命令行上的任何文件无法打开，bc将报告该文件不可用并终止。

关于表达式

这些数字由表达式和语句操作。由于语言设计为交互式，所以语句和表达式会尽快执行。没有“主”程序，而是在遇到时执行代码。

一个简单的表达式只是一个常量。BC使用变量ibase指定的当前输入基将常量转换为内部十进制数。(函数中有一个例外。)ibase的有效值为2到16，将超出此范围之后，分配给ibase的值可能是2或16。输入数字可能包含字符0-9和A-F。(注：它们必须是大写字母，小写字母是变量名称。)无论ibase的值是多少，单数数字总是有该数字的值。(即A=10。)对于多位数字，bc将所有大于或等于ibase的输入数字更改为IBASE-1的值。这使得FFF始终输入的最大3位数。

完全表达式类似于许多其他高级语言。由于只有一种数字，所以没有混合类型的规则。相反，有关于表达式精度的规则。每个表达式都有一个精度。这是从原始数字的精度、所执行的操作以及在许多情况下变量scale的值导出的。变量scale的有效值为0到可由c语言整数表达的最大值。

在以下合法表达式的描述中，“expr”指的是一个完整的表达式，“var”指的是一个简单的或数组变量。除非特别提到，结果的精度是所涉及的表达式的最大精度。

普通表达式	说明
-expr	结果是对表达式的否定。
++var	变量增加1，而新值是表达式的结果
--var	变量减1，而新值是表达式的结果
var++	表达式的结果是变量的值，然后变量增加1
var--	表达式的结果是变量的值，然后变量减1
expr * expr	表达式的结果是这两个表达式的乘积
expr + expr	表达式的结果是这两个表达式的和
expr - expr	表达式的结果是这两个表达式的差
expr / expr	表达式的结果是这两个表达式的商。结果的精度是变量scale的值。
expr % expr	表达式的结果是“余数”
expr ^ expr	表达式的结果是n次方，第二个表达式必须是整数。如果指数是负数，那么结果的精度是scale
(expr)	强制对表达式进行计算
var = expr	变量就是表达式的值
var = expr	相当于“var=var expr”，例如“var -= expr”等价于“var=var-expr”

关系表达式	关系表达式是一种特殊的表达式，计算结果总是0或1。如果关系为真，则计算为1；如果关系为假，则结果是0。这些表达式可能出现在任何合法表达式中。 (POSIX bc要求关系表达式仅用于if、while和语句，并且只能在其中进行一次关系测试。)		
expr1 < expr2	expr1 <= expr2	expr1 > expr2	expr1 >= expr2
expr1 == expr2	expr1 != expr2	expr1 && expr2	expr1 expr2
!expr			

运算符的优先级如下，从上到下依次增加：

运算符	结合方式
	左结合
&&	左结合
!	不结合
关系运算符	左结合
赋值运算符	右结合
+和-	左结合
*、/、%	左结合
^	右结合
一元运算符 -	不结合
++和--	不结合

选择此优先级是为了使符合POSIX的bc程序能够正确运行。这将导致关系运算符和逻辑运算符在与赋值表达式一起使用时有一些不寻常的行为。例如下面的表达式：

```
a = 3<5
```

大多数C程序员会假设这会将“3<5”(值1)的结果赋给变量“a”，这在bc中所做的是将值3赋给变量“a”，然后比较3到5。在使用关系运算符和逻辑运算符与赋值运算符时，最好使用括号。

在bc中还提供了一些特殊的表达式。这些表达式与用户定义的函数和标准函数有关。它们都以“名称(参数)”的形式出现。有几个标准函数：

1. length (expr)，计算表达式结果的有效位数。
2. read ()，Read函数(一个扩展)将从标准输入中读取一个数字，而不管该函数发生在何处。注意，这可能会导致标准输入中的数据和程序混合出现问题。这个函数的最佳使用是在一个已经编写好的程序中，这个程序需要用户输入，但绝不允许从用户输入程序代码。读函数的值是从标准输入中读取的数字，使用转换基的变量ibase的当前值。
3. scale (expr)，这个函数的值是expr表达式中小数点之后的位数。
4. sqrt (expression)，函数的结果是表达式的开方值。

关于语句

语句(在大多数代数语言中)提供表达式计算的顺序。在bc中，语句被“尽快”执行。执行发生在遇到的换行符的时候，并且有一个或多个完整的语句。由于这种立即执行，换行符在bc中非常重要。事实上，分号和换行符都用作语句分隔符。如果换行符放置不当，将导致语法错误。因为换行符是语句分隔符，所以可以使用反斜杠字符隐藏换行符。()在bc中显示为空格而不是新行。语句列表是由分号和换行符分隔的一系列语句。

1. 表达式

这条语句做两件事之一。如果表达式以“<变量><赋值>.”开头，则被认为是赋值语句。如果表达式不是赋值语句，则计算表达式并将表达式打印到输出。在打印数字之后，将打印换行符。例如，“a=1”是一个赋值语句和“(a=1)”是一个具有内嵌赋值的表达式。输出基obase的有效值是2~ BC_BASE_MAX。对于基数2至

16，通常采用书写数字的方法。对于大于16的基数，bc使用多字符数字方法将每个较高的基数打印成以10为基数的数据。由于数字具有任意精度，一些数字可能无法在一条输出线上打印。这些长数字将被分割，以\"作为一行上的最后一个字符，每行打印的最大字符数为70个。由于bc的交互性，打印一个数字会导致最后将打印值赋值给特殊变量"last"的副作用。这允许用户恢复打印的最后一个值，而不必重新键入打印数字的表达式。将last变量赋值为"最后一个值"是合法的，并将最后一个打印的值用指定的值覆盖。新赋值将保持不变，直到打印下一个数字或将另一个值分配给"last"为止。

2. 字符串

字符串被打印到输出。字符串以双引号开始，包含所有字符直到下一个双引号字符。所有字符都是字面意思，包括任何换行符。字符串后不打印换行符。

3. 打印列表

print语句(扩展)提供了另一种输出方法。"list"是由逗号分隔的字符串和表达式的列表。每个字符串或表达式都按列表的顺序打印。不打印终止换行符。表达式的将被计算出值，最后将其值打印并分配给变量"last"。打印语句中的字符串将打印到输出中，并可能包含特殊字符。特殊字符以反斜杠字符\"开始。bc识别的特殊字符是"a"(警报或钟)、"b"(反斜杠)、"f"(表单提要)、"n"(换行符)、"r"(回车)、"q"(双引号)、"t"(制表符)和"\"(反斜杠)。反斜杠后面的任何其他字符都将被忽略。

4. 语句列表

这是复合语句。它允许将多个语句组合在一起执行。

5. if (表达式) statement 1 [else statement 2]

if语句根据表达式的值决定执行statement 1或statement 2。如果表达式为非零，则执行statement 1。如果存在statement 2，且表达式的值为0的时候执行statement 2。

6. while (expression) statement

while语句将在表达式为非零时执行语句。它在每次执行语句之前计算表达式。循环的终止是由零表达式值或break语句的执行引起的。

7. for ([expression1] ; [expression2] ; [expression3]) statement

for语句控制语句的重复执行。表达式1是在循环之前计算的。表达式2是在每次语句执行之前计算的。如果表达式2为非零，则计算语句；如果为零，则终止循环。每次执行语句后，计算表达式3。在重新计算表达式2之前，如果未找到表达式1或表达式3，则不会在计算值的点上对其进行任何计算。

8. break

break语句用来强制退出，通常用在for语句或者while语句中。

9. continue

continue语句用来结束本次循环。

10. halt

halt语句会导致bc程序退出。

11. return

函数返回0.

12. return expr

返回表达式的值。

13. 伪语句，这些语句不会执行，他们在编译的时候才会起作用。下面列出伪语句

- a. limits，打印由于bc版本而产生的限制 - b. quit，遇到quit指令的时候就会退出bc，无论它出现在什么地方。例如“if (0 == 1) quit”就会导致退出bc - c. warranty，打印较长的授权通知

函数

1. 函数

bc中的函数总是计算一个值并将其返回给调用者。函数定义是“动态的”，在输入中遇到定义之前，函数是未定义的。然后使用该定义，直到遇到相同名称的另一个定义函数。然后，新定义取代旧的定义。函数定义方式如下：

```
define name ( parameters ) { newline  
auto_list statement_list }
```

sh

函数的调用很简单“name (parameters) ”。

2. 参数

参数是数字或数组。在函数定义中，可以有0个或者多个参数，通过逗号分隔开。所有参数都是通过值参数调用的。数组是通过符号“name[]”在参数定义中指定的。在函数调用中，实参是数字参数的完整表达式。相同的符号。数组的定义和传值使用相同的符号。命名数组通过值传递给函数。由于函数定义是动态的，因此在调用函数时会检查参数号和类型。参数数量或类型的不匹配都会导致运行时错误。对未定义函数的调用也会出现运行时错误。

3. auto_list

“auto_list”是供“本地”使用的变量的可选列表。auto_list的语法(如果存在)是“autoname, ...;”。(分号是可选的。)每个名称都是自动变量的名称。数组可以使用与参数相同的表示法来指定。这些变量的值在函数开始时被推入堆栈中。然后将变量初始化为零，并在函数的整个执行过程中使用。在函数退出时，这些变量被弹出，以便恢复这些变量的原始值(在函数调用时)。这些参数实际上是自动变量，它们被初始化为函数调用中提供的值。自动变量不同于传统局部变量，因为如果函数A调用函数B，B可以使用相同的名称访问函数A的自动变量，除非函数B调用它们为自动变量。由于自动变量和参数被推到堆栈上，bc支持递归函数。

4. 函数体

函数体是一系列bc语句的列表。同样，语句用分号或换行符分隔。返回语句导致函数的终止和值的返回。返回语句有两个版本。第一个形式“return”将值0返回给调用表达式。第二种形式“return (表达式)”计算表达式的值并将该值返回给调用表达式。在每个函数的末尾有一个隐含的“return (0)”。这允许一个函数终止并返回0，而不需要显式返回语句。

函数还会改变变量ibase的用法。函数体中的所有常量都将在函数调用时使用ibase的值进行转换。在函数执行过程中，ibase的更改将被忽略，但标准函数读取除外，后者将始终使用ibase的当前值来转换数字。

当前版本的bc，在函数中添加了几个扩展。首先，定义的格式稍微放松了一些。标准要求开始大括号与定义关键字在同一行，所有其他部分必须在下面的行上。这个版本的bc将允许之前的任何数目的换行符。在函数的开头支撑之后，例如，下面的定义是合法的：

```
define d (n) { return (2*n); }
define d (n)
{ return (2*n); }
```

sh

5. void类型

函数可以定义为void。空函数不返回值，因此可能不会在任何需要值的地方使用。空函数在输入行调用时不会产生任何输出。关键字void放在关键字定义和函数名称之间。例如，请考虑下面的例子

```
define py (y) { print ">>>", y, "<---", "0; "}
define void px (x) { print ">>>", x, "<---", "0; "}
py(1)
-->1<---
0           //由于py不是void，因此有默认返回值，因此这里打印了它的返回值，
px(1)
-->1<---      //px是void类型，最后不会打印返回值
```

sh

此外，还为数组添加了按变量调用。为了申明一个数组变量，函数中的数组参数是这样定义的“name[]”。

数学库

1.如果使用“-l”选项调用bc，则预加载一个数学库，并将默认精度设置为20。数学库中有一下函数：

- s (x)，计算x的正弦值，x是弧度值。
- c (x)，计算x的余弦值，x是弧度值。
- a (x)，计算x的反正切值，返回弧度。
- l (x)，计算x的自然对数。
- e (x)，e的x次方。
- j (n, x)，从n到x的阶数。

2.例子

下面的句子可以将“pi”的值赋值给shell变量pi

```
pi = $(echo "scale=10; 4*a(1)" , bc -l)
```

下面的句子就是数学库中e的次方定义方式

```

scale = 20
/* Uses the fact that e^x = (e^(x/2))^2
   When x is small enough, we use the series:
   e^x = 1 + x + x^2/2! + x^3/3! + ...
*/
define e(x) {
    auto a, d, e, f, i, m, v, z

    /* Check the sign of x. */
    if (x<0) {
        m = 1
        x = -x
    }

    /* Precondition x. */
    z = scale;
    scale = 4 + z + .44*x;
    while (x > 1) {
        f += 1;
        x /= 2;
    }

    /* Initialize the variables. */
    v = 1+x
    a = x
    d = 1
    for (i=2; 1; i++) {
        e = (a *= x) / (d *= i)
        if (e == 0) {
            if (f>0) while (f--) v = v*v;
            scale = z
            if (m) return (1/v);
            return (v/1);
        }
        v += e
    }
}

```

下面的语句实现一个计算支票簿余额的简单程序

```

scale=2
print "\nCheck book program!\n"
print " Remember, deposits are negative transactions.\n"
print " Exit by a 0 transaction.\n\n"

print "Initial balance? "; bal = read()
bal /= 1
print "\n"
while (1) {
    "current balance = "; bal
    "transaction? "; trans = read()
    if (trans == 0) break;
    bal -= trans
    bal /= 1
}
quit

```

下面的语句采用递归的方式计算x的阶乘

```
define f (x) {
    if (x <= 1) return (1);
    return (f(x-1) * x);
}
```

readline和libedit选项

可以编译GNU bc(通过一个配置选项)来使用GNU readline输入编辑器库或bsd libedit库。这允许用户在将行发送到bc之前进行编辑。它还允许保存以前键入的行的历史记录。当选择此选项时，bc还有一个特殊变量。变量“history”是保留的历史记录行数。对于readline，值-1表示不限制历史记录的行数，0将禁用历史记录功能，默认值为100。

差别

这个版本的bc是从POSIX P 1003.2/D11草案中实现的，包含了与草案和传统实现相比的一些区别和扩展，它不是以传统的方式使bc(1)实现的，这个版本是一个解析和运行程序字节代码转换的单一进程。这里有一个“无文档”选项(-c)，它导致程序将字节码输出到标准输出，而不是运行它。它主要用于调试解析器和准备数学库。差异的一个主要来源是扩展，下面列出一些差异和扩展：

1. LANG环境变量，此版本在处理lang环境变量和从lc_开始的所有环境变量时不符合POSIX标准。
2. 名字，传统和POSIX bc都有用于函数、变量和数组的单字母名称。它们被扩展为以字母开头的多字符名称，可以包含字母、数字和下划线字符。
3. 字符串，字符串不允许包含NUL字符。POSIX表示所有字符都必须包含在字符串中。
4. last，POSIX bc中没有last变量。
5. 比较，POSIX bc只允许在if语句、while语句和for语句的第二个表达式中进行比较。
6. if语句，POSIX bc中if语句没有else。
7. for语句，POIX bc中要求for语句中的3个表达式都必须具备。
8. &&, ||, !，POSIX bc中没有逻辑运算。
9. read，POSIX bc没有read功能。
10. 打印语句，POSIX bc没有打印语句。
11. continue语句，POSIX bc没有continue语句。
12. return，POSIX bc要求return的表达式加括号。
13. 数组参数，POSIX bc不(目前)完全支持数组参数。POSIX语法允许函数定义中的数组，但没有提供将数组指定为实际参数的方法。(这很可能是语法上的疏忽。)传统的bc实现只通过值数组参数进行调用。
14. 函数，POSIX bc要求函数开头的大括号和define关键字在同一行，语句在下一行。
15. =+, =-, =*, =/, =%, =^。POSIX bc不要求定义这些“旧样式”赋值操作符。此版本可能允许这些“旧样式”赋值。使用限制语句查看安装的版本是否支持它们。如果它确实支持“旧样式”赋值运算符，则“a=-1”语句将使a减少1，而不是将a设置为值-1。
16. 数字中的空格，bc的其他实现允许数字空格。例如，“x=1 3”将值13赋值给变量x。相同的语句将导致bc版本中的语法错误。
17. 错误和执行，在程序中发现语法和其他错误时，此实现与其他实现的代码不同。如果在函数定义中发现语法错误，则错误恢复机制将尝试查找语句的开头并继续解析函数。一旦在函数中发现语法错误，该函数将不可调用并变为未定义。交互执行代码中的语法错误将使当前执行块失效。执行块由在完整语句序列之后出现的行尾终止。例如

a = 1 b = 2

这个语句有两个执行块，而下面的语句

```
{a = 1 b = 2}
```

只有一个执行块。任何运行时错误都会终止当前的执行块，而警告则不会。

18. 中断，在交互会话期间，SIGINT信号(通常由终端上的“ctrl+c”生成)将导致当前执行块的执行中断。它将显示一个“运行时”错误，指示哪个功能被中断。在所有运行时结构被清除后，将打印一条消息通知用户bc准备好接收更多的输入。所有先前定义的函数都保留定义，所有非自动变量的值是中断点的值。在清理过程中，所有自动变量和函数参数都会被移除。对于一个非交互式会话，SIGINT信号将终止bc的整个运行。

限制

下面列出当前bc程序的一些限制，有一些限制可能已经被用户修改过。

1. BC_BASE_MAX，最大输出基设置为999。最大输入基为16。
2. BC_DIM_MAX，这是当前分布的65535以内的任意限制，每个机器可能都不一样。
3. BC_SCALE_MAX，小数点前后的位数都由INT_MAX限制。
4. BC_STRING_MAX，字符串中的字符字数由INT_MAX限制。
5. exponent，指数运算中的指数值由LONG_MAX限制。
6. variable names，当前对每个简单变量、数组和函数名字的限制32767。

环境变量

下面的环境变量由bc程序来控制

1. POSIXLY_CORRECT，和“-s”选项一样。
2. BC_ENV_ARGS，这是另一种获取bc参数的机制。格式与命令行参数相同。这些参数是先处理的，因此环境参数中列出的任何文件在任何命令行参数文件之前都会被处理。这允许用户设置“标准”选项和文件，以便在每次调用环境变量中的文件通常包含用户希望在每次运行bc时定义的函数定义。
3. BC_LINE_LENGTH，这应该是一个整数，指定数字输出行中的字符数。这包括用于长数字的反斜杠和换行符。，如果值是0，将禁用多行功能。此变量的任何其他值如果小于3，则将行长设置为70。

举例

简单计算

```
[root@192 ~]$ bc
bc 1.06.95  # 欢迎语句
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type 'warranty'.
12+23      # 输入加法表达式, 回车
35        # 得到结果
100/25    # 输入除法表达式, 回车
4        # 得到结果
quit      # 退出指令
[root@192 ~]$
```

执行for循环语句

```
for(i=0; i<3; i++){print "hello\n"} #这是一个打印语句
hello
hello
hello
```

sh

从文件读取内容并且执行bc

```
[root@localhost /]$ cat test.c    #查看文件的内容，里面全是bc语句
/*define 3 functions add,sub,mul*/
define add(x,y){
    return x+y;
}

define sub(x,y){
    return x-y;
}

define mul(x,y){
    return x*y;
}

/*for statement*/
for(i=0;i<3;i++){
    print "bc test ",i,"\n";
}
/*print statement*/
print "10+5=",add(10,5),"\n"
print "10-5=",sub(10,5),"\n"
print "10&5=",mul(10,5),"\n"
/*quit bc program*/
quit

[root@localhost /]$ bc test.c  #bc程序从文件获取到代码，然后执行
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'.
bc test 0
bc test 1
bc test 2
10+5=15
10-5=5
10&5=50
```

sh

bg - 将前台终端作业移动到后台运行

bg指令用来将挂起的进程在后台执行。

主要用途

- 用于将作业放到后台运行，使前台可以执行其他任务。该命令的运行效果与在指令后面添加符号&的效果是相同的，都是将其放到系统后台执行。
- 若后台任务中只有一个，则使用该命令时可以省略任务号。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
bg [jobs]
```

sh

选项

无

举例

后台运行

```
[root@localhost ~]$ find / -name passwd          #执行查找指令, ctrl+z挂起任务
find: 探测到文件系统循环; “/var/named/chroot/var/named” 是与 “/var/named” 相同的文件系统循环的一部分。
^Z
[1]+  Stopped                  find / -name passwd
You have new mail in /var/spool/mail/root
[root@localhost ~]$ bg           #任务在后台执行, 结果任然会输出
[1]+  find / -name passwd &
[root@localhost ~]$ /usr/bin/passwd
/etc/passwd
/etc/squid/passwd
/etc/pam.d/passwd
```

bind - 显示或设置键盘按键与其相关的功能

dirs命令用于显示当前shell堆栈记录的目录。使用pushd可以向堆栈中增加记录，popd可以删除记录

主要用途

- 显示目录堆栈。
- 清空目录堆栈。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
dirs [+n] [-n] [-cp1v]
```

sh

选项

```
+n      # 显示从左起的第n个目录, 从0开始计数  
-n      # 显示从右起的第n个目录, 从0开始计数  
-c      # 删除所有记录  
-l      # 以完整的格式显示  
-p      # 以每行一个记录的方式显示  
-v      # 以每行一个记录的方式显示, 并加上序号
```

sh

举例

显示当前所有记录

```
[root@localhost ~]$ dirs -l -v          #显示所有记录, 每行一个, 并且加序号  
0  /root  
[root@localhost ~]$
```

sh

增加记录并显示

```
[root@localhost ~]$ pushd /doc          #增加记录  
/doc ~  
[root@localhost ~]$ dirs -l -v          #查看记录, 已经成功添加  
0  /doc  
1  /root  
[root@localhost ~]$
```

sh

注意

- `bash` 的目录堆栈命令包括 `dirs` `popd` `pushd` 。
- 当前目录始终是目录堆栈的顶部。
- 该命令是 `bash` 内建命令，相关的帮助信息请查看 `help` 命令。

bunzip2 - 解压缩**bzip2**压缩过的文件

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
bunzip2 [-fkvsVL] 文件
```

sh

选项

```
-f, --force # 强制执行  
-k, --keep # 解压之后, 保留源文件  
-v, --verbose # 显示详细信息  
-s, --small # 减少内存使用, 用于压缩、解压缩和测试。使用修改的算法对文件进行解压缩和测试, 该算法每个块字节  
-L, --license # 显示许可条款和条件  
-V, --version # 显示软件版本
```

sh

举例

1) 解压, 保留压缩包

```
[root@localhost weijie]$ bunzip2 -k 2.c.bz2 #解压之后保留压缩包  
[root@localhost weijie]$ ls  
11.c 1.c.gz 1.gz 2.c 2.c.bz2 3.c 4.c 5.c 6.c~ rec000012.c.bz2 res.zip
```

sh

2) 解压

```
[root@localhost weijie]$ bzip2 -d res.bz2 #解压  
[root@localhost weijie]$ ls  
11.c 1.c.bz2 2.c 3.c 4.c 5.c 6.c~ bak res
```

sh

3) 将两个文件压缩到一个文件中

sh

```
[root@localhost weijie]$ cat 1.c 2.c          #输出两个文件的内容
hello world,
i am david.
i love linux,
love code.

123
23
212

[root@localhost weijie]$ bzip2 -c 1.c > foo.gz      #将1.c压缩到foo
[root@localhost weijie]$ bzip2 -c 2.c >> foo.gz     #将2.c压缩到foo
[root@localhost weijie]$ bgzip2 -d foo.gz           #解压foo
[root@localhost weijie]$ cat foo                   #显示foo的内容

hello world,
i am david.
i love linux,
love code.

123
23
212
```

bzcat - 解压缩指定的.bz2文件

bzcat命令 解压缩指定的.bz2文件，并显示解压缩后的文件内容。保留原压缩文件，并且不生成解压缩后的文件。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
bzcat [-s] 文件
```

sh

选项

```
-s # 降低程序运行时内存使用
```

sh

举例

解压文件到标准输出

```
[root@localhost weijie]$ cat 2.c
123
23
212
[root@localhost weijie]$ bzcat -s 2.c.bz2 #将文件解压到标准输出
123
23
212
```

sh

bzip2 - 将文件压缩成bz2格式

bzip2命令 用于创建和管理（包括解压缩）".bz2"格式的压缩包。

bzip2 采用 Burrows-Wheeler 块排序文本压缩算法和 Huffman 编码方式压缩文件。压缩率一般比基于 LZ77/LZ78 的压缩软件好得多，其性能接近 PPM 族统计类压缩软件。

命令行参数有意设计为非常接近 GNU gzip 的形式，但也不完全相同。

bzip2 从命令行读入文件名和参数。每个文件被名为 "原始文件名.bz2" 的压缩文件替换。每个压缩文件具有与原文件相同的修改时间、权限，如果可能的话，还具有相同的属主，因此在解压缩时这些特性将正确地恢复。在某些文件系统中，没有权限、属主或时间的概念，或者对文件名的长度有严格限制，例如 MSDOS，在这种情况下，bzip2 没有保持原文件名、属主、权限以及时间的机制，从这个意义上说，bzip2 对文件名的处理是幼稚的。

bzip2 和 bunzip2 在缺省情况下不覆盖已有的文件。如果想覆盖已有的文件，要指定 -f 选项。

如果未指定文件名，bzip2 将压缩来自标准输入的数据并写往标准输出。在这种情况下，bzip2 会拒绝将压缩结果写往终端，因为这完全无法理解并且是没有意义的。

bunzip2 (以及 bzip2 -d) 对所有指定的文件进行解压缩处理。不是由 bzip2 产生的文件将被忽略，同时发出一个警告信息。bzip2 按下列方式由压缩文件名确定解压后的文件名：

filename.bz2	解压成	filename
filename.bz	解压成	filename
filename.tbz2	解压成	filename.tar
filename.tbz	解压成	filename.tar
anyothername	解压成	anyothername.out

如果文件名的后缀不是下列之一：.bz2, .bz, .tbz2 或 .tbz, .bzip2 将抱怨无法确定原始文件名，并采用原文件名加.out 作为解压缩文件名。

在压缩时，如果不提供文件名，bzip2 将从标准输入读取数据，压缩结果写往标准输出。

bzip2 采用 32 位 CRC 校验码作自我检查，以确认解压后的文件与原始文件相同。这可用于检测压缩文件是否损坏，并防止 bzip2 中未知的缺陷（运气好的话这种可能性非常小）。数据损坏而未检测到的几率非常之小，对于每个被处理的文件大约是四十亿分之一。检查是在解压缩时进行的，因此它只能说明某个地方出问题了。它能帮助恢复原始未压缩的数据。可以用 bzip2recover 来尝试从损坏的文件中恢复数据。

返回值：正常退出返回 0，出现环境问题返回 1（文件未找到，非法的选项，I/O 错误等），返回 2 表明压缩文件损坏，出现导致 bzip2 紧急退出的内部一致性错误（例如缺陷）时返回 3。

适用范围

RedHat openSUSE	RHEL Fedora	Ubuntu Linux Mint	CentOS Alpine Linux	Debian Arch Linux	Deepin	SUSE
--------------------	----------------	----------------------	------------------------	----------------------	--------	------

语法

```
bzip2 [ -cdfkqstvzVL123456789 ] [ filenames ... ]
```

sh

选项

```
-c --stdout
    # 将数据压缩或解压缩至标准输出。

-d --decompress
    # 强制解压缩。 bzip2, bunzip2 以及 bzcat 实际上是同一个程序，进行何种操作将根据程序名确定。 指定该选

-z --compress
    # -d 选项的补充：强制进行压缩操作，而不管执行的是哪个程序。

-t --test
    # 检查指定文件的完整性，但并不对其解压缩。 实际上将对数据进行实验性的解压缩操作，而不输出结果。

-f --force
    # 强制覆盖输出文件。通常 bzip2 不会覆盖已经存在的文件。该选项还强制 bzip2 打破文件的硬连接，缺省情况下

-k --keep
    # 在压缩或解压缩时保留输入文件（不删除这些文件）。

-s --small
    # 在压缩、解压缩及检查时减少内存用量。采用一种修正的算法进行压缩和测试，每个数据块仅需要 2.5 个字节。这

    # 的内存中进行解压缩，尽管速度只有通常情况下的一半。

    # 在压缩时，-s 将选定 200k 的块长度，内存用量也限制在 200k 左右，代价是压缩率会降低。总之，如果机器的

    # 可对所有操作都采用-s 选项。参见下面的内存管理。

-q --quiet
    # 压制不重要的警告信息。属于 I/O 错误及其它严重事件的信息将不会被压制。

-v --verbose
    # 详尽模式 -- 显示每个被处理文件的压缩率。命令行中更多的 -v 选项将增加详细的程度，使 bzip2 显示出许多

-L --license -V --version
    # 显示软件版本，许可证条款及条件。

-1 to -9
    # 在压缩时将块长度设为 100 k、200 k .. 900 k。对解压缩没有影响。参见下面的内存管理。

-- # 将所有后面的命令行变量看作文件名，即使这些变量以减号"-"打头。可用这一选项处理以减号"-"打头的文件名，

--repetitive-fast --repetitive-best
    # 这些选项在 0.9.5 及其以上版本中是多余的。在较早的版本中，这两个选项对排序算法的行为提供了一些粗糙的控

    # 及其以上版本采用了改进的算法而与这些选项无关。
```

举例

压缩指定文件filename：

```
bzip2 filename  
# 或  
bzip2 -z filename
```

sh

这里，压缩的时候不会输出，会将原来的文件filename给删除，替换成filename.bz2.如果以前有filename.bz2则不会替换并提示错误（如果想要替换则指定-f选项，例如bzip2 -f filename；如果filename是目录则也提醒错误不做任何操作；如果filename已经是压过的了有bz2后缀就提醒一下，不再压缩，没有bz2后缀会再次压缩。

解压指定的文件**filename.bz2**：

```
bzip2 -d filename.bz2  
# 或  
bunzip2 filename.bz2
```

sh

这里，解压的时候没标准输出，会将原来的文件filename.bz2给替换成filename。如果以前有filename则不会替换并提示错误（如果想要替换则指定-f选项，例如bzip2 -df filename.bz2）。

压缩解压的时候将结果也输出：

```
bzip2 -v filename
```

sh

输入之后，输出如下：

```
filename: 0.119:1, 67.200 bits/byte, -740.00% saved, 5 in, 42 out.
```

这里，加上-v选项就会输出了,只用压缩举例了，解压的时候同理bzip2 -dv filename.bz2不再举例了。

模拟解压实际并不解压：

```
bzip2 -tv filename.bz2
```

sh

输入之后，输出如下：

```
filename.bz2: ok
```

这里，-t指定要进行模拟解压，不实际生成结果，也就是说类似检查文件,当然就算目录下面有filename也不会有什么错误输出了，因为它根本不会真的解压文件。为了在屏幕上输出，这里加上-v选项了,如果是真的解压bzip2 -dv filename.bz2则输出的是把"ok"替换了"done"。

压缩解压的时候，除了生成结果文件，将原来的文件也保存：

```
bzip2 -k filename
```

sh

这里，加上-k就保存原始的文件了，否则原始文件会被结果文件替代。只用压缩举例了，解压的时候同理\$bzip2 -dk filename.bz2不再举例了。

解压到标准输出：

```
bzip2 -dc filename.bz2
```

sh

输入之后，输出如下：

```
hahahahaahahaha
```

这里，使用-c指定到标准输出，输出的是文件filename的内容，不会将filename.bz2删除。

压缩到标准输出：

```
bzip2 -c filename  
bzip2: I won't write compressed data to a terminal.  
bzip2: For help, type: `bzip2 --help'.
```

sh

这里，使用-c指定压缩到标准输出不删除原有文件，不同的是，压缩后的文件无法输出到标准输出。

使用**bzip2**的时候将所有后面的看作文件(即使文件名以'-'开头)：

```
bzip2 -- -myfilename
```

sh

这里主要是为了防止文件名中-产生以为是选项的歧义。

bzip2recover - 恢复被破坏的.bz2压缩包中的文件

bzip2recover命令 可用于恢复被破坏的".bz2"压缩包中的文件。

修复损坏的bzip2文件，bzip2以区块的方式来压缩，每一个区块都可以当做一个单位，当区块损坏之后，可以使用bzip2recover修复。

bzip2是以区块的方式来压缩文件，每个区块视为独立的单位。因此，当某一区块损坏时，便可利用bzip2recover，试着将文件中的区块隔开来，以便解压缩正常的区块。通常只适用在压缩文件很大的情况。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
bzip2recover file
```

sh

选项

无

举例

```
[sogrey@bogon 文档]$ bzip2recover 2.c.bz2
bzip2recover 1.0.5: extracts blocks from damaged .bz2 files.
bzip2recover: searching for block boundaries ...
    block 1 runs from 80 to 273
bzip2recover: splitting into blocks
    writing block 1 to `rec000012.c.bz2' ...
bzip2recover: finished
```

sh

bzmore - 查看bzip2压缩过的文本文件的内容

bzmore命令 用于查看bzip2压缩过的文本文件的内容，当下一屏显示不下时可以实现分屏显示。

将bzip压缩过的文件解压到标准输出，同时也可以将普通文件显示到标准输出。该指令可以实现分屏显示，并且不会删除压缩包。bzmore是一个过滤器，它允许在软拷贝终端上一次检查压缩或纯文本文件。bzmore可以处理使用bzip2压缩的文件，也可以处理未压缩的文件。如果文件不存在，bzmore将查找同名文件，并添加.bz2后缀。Bzmore通常在每个屏幕后暂停，打印-更多-在屏幕底部。如果用户然后键入回车，则会显示多一行。如果用户点击一个空格，则会显示另一个屏幕。其他可能性将在后面列举。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
bzmore 文件
```

sh

选项

Bzmore在文件"/etc/Tercap"中查找以确定终端特性，并确定默认窗口大小。在能够显示24行的终端上，默认窗口大小为22行。在bzmore暂停时可能键入的其他序列及其效果如下(i是可选的整数参数，默认为1)：

```
i<space>      # 多显示i行(或另一个屏幕，如果没有给出参数)
ctrl+D        # 多显示11行(“滚动”)。如果给定i，则滚动大小设置为i。
d             # 同上
iz            # 与键入空格相同，但如果有i，则会成为新的窗口大小。请注意，窗口大小将恢复到当前文件末尾的i行。
is            # 跳过i行，然后打印出满屏幕
if            # 跳过i屏幕，打印一整行的屏幕
q, Q          # 停止读取当前文件；继续下一个文件(如果有的话)
e, q          # 当输出提示符--More-- (Next file: file时，此命令将导致bzmore退出。
s             # 当输出提示符--More-- (Next file: file时，此命令将导致bzmore跳过下一个文件，然后继续读取当前文件
=             # 显示当前的行号
i/expr         # 搜索正则表达式expr的第i次出现。如果找不到模式，bzmore将继续到下一个文件(如果有的话)。
in            # 搜索输入的最后一个正则表达式的第i次出现
!command       # 使用命令唤醒shell。字符‘!’在“命令”中，将替换为前面的shell命令。序列“\! ”改为“! ”
:q, :Q         # 停止读取当前文件；转到下一个文件(如果有的话)(与q或Q相同)。
.             # 重复前面的命令
```

这些命令立即生效，即不需要键入回车。在给出命令字符本身的时间之前，用户可以点击行终止字符来取消正在形成的数值参数。此外，用户还可以点击擦除字符来重新显示-更多的消息。

在任何时候，当输出被发送到终端时，用户可以按退出键(通常是Control-)。Bzmore将停止发送输出，并显示通常的"--More--"提示。然后，用户可以正常方式输入上述命令之一。不幸的是，当这样做时，会丢失一些输出，因为当退出信号发生时，终端输出队列中等待的任何字符都会被刷新。

该程序将终端设置为noecho模式，以便输出可以连续。因此，您键入的内容将不会显示在您的终端上，除了'/'和'!'命令。

如果标准输出不是teletype，那么bzmore就像bzcat一样，只是在每个文件之前打印一个头。

举例

1) 解压bz2文件

```
[root@localhost weijie]# bzmore 2.c.bz2 //将内容解压到标准输出，不删除压缩包  
-----> 2.c.bz2 <-----  
123  
23  
212
```

sh

2) 显示普通文件

```
[root@localhost weijie]# bzmore 1.c //直接显示普通文件  
-----> 1.c <-----  
hello world,  
i am david.  
i love linux,  
love code.
```

sh

cal - 显示当前日历或指定日期的日历

cal命令 用于显示当前日历，或者指定日期的日历，如果没有指定参数，则显示当前月份。

一个单一的参数指定要显示的年份(1 - 9999)；注意年份必须被完全地指定: cal 89 不会显示1989年的日历。

两个参数表示月份(1 - 12)和年份。如果没有指定参数，则显示当前月份的日历。

一年从Jan 1 (1月1日)开始。

格里高利历法改革(Gregorian Reformation)被认为发生于1752年9月3日。在此之前，多数国家已经认可这项改革(尽管有一些直到20世纪初才认可它)。那天之后的10天在这项改革被略去了，所以那个月的日历有点不太寻常。

cal指令可以显示一个日历信息，如果没有指定选项和参数，那么就会显示当前的月份。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
cal [-smjy13] [[ [day] month] year]
```

sh

选项

-V	# 显示命令版本信息
-1	# 显示一个月的日历信息，这是默认值
-3	# 显示上个月、这个月、下个月的日历信息
-s	# 将星期日作为第一天来显示
-m	# 将星期一作为一周的第一天，默认星期日是第一天
-j	# 显示儒略历（显示的是从1月1日起，到当前的天数）
-y	# 显示当年的日历信息

sh

一个参数指定要显示的年份(1-9999)；请注意，必须指定完整的年份：“cal 89”将不显示1989年的日历。

两个参数表示月份(1-12)和年份。三个参数表示日期(1-31)、月份和年份，如果在终端上显示日历，则会突出显示日期。如果没有参数，则显示当前月份的日历。一年从1月1日开始。一周的第一天由地区决定。

举例

显示儒略历日历

sh

```
[sogrey@bogon ~]$ cal -j # 这里显示的不是“日”，而是一年的第几天
      六月 2021
 日 一 二 三 四 五 六
 152 153 154 155 156
157 158 159 160 161 162 163
164 165 166 167 168 169 170
171 172 173 174 175 176 177
178 179 180 181
```

```
[sogrey@bogon ~]$
```

将“星期日”作为第一天显示，显示最近3个月

sh

```
[sogrey@bogon ~]$ cal -3 -s
      五月 2021          六月 2021          七月 2021
 日 一 二 三 四 五 六 日 一 二 三 四 五 六 日 一 二 三 四 五 六
           1           1 2 3 4 5           1 2 3
 2 3 4 5 6 7 8   6 7 8 9 10 11 12   4 5 6 7 8 9 10
 9 10 11 12 13 14 15  13 14 15 16 17 18 19  11 12 13 14 15 16 17
16 17 18 19 20 21 22  20 21 22 23 24 25 26  18 19 20 21 22 23 24
23 24 25 26 27 28 29  27 28 29 30           25 26 27 28 29 30 31
30 31
[sogrey@bogon ~]$
```

将星期一作为一个星期的第一天

sh

```
[sogrey@bogon ~]$ cal -m
      六月 2021
一 二 三 四 五 六 日
 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
```

```
[sogrey@bogon ~]$
```

cancel - 取消已存在的打印任务

cancel命令 用于取消已存在的打印任务。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
cancel [ -E ] [ -U username ] [ -a ] [ -h hostname[:port] ] [ -u username ] [id] [ destination ]
```

打印任务号：指定要取消的打印任务编号。

选项

-E	# 使用加密模式
-a	# 取消所有打印任务
-h	# 指定远程服务器
-U	# 设置别名
-u	# 取消被指定用户拥有的打印任务

举例

取消打印任务

```
[root@localhost /]$ lpq # 查看打印队列
printer01 已准备就绪, 正在打印
顺序    所有者    作业    文件          总大小
1st      root      4       4.c          1024 字节
active   root      2       5.c          1024 字节
[root@localhost /]$ cancel 4 # 取消4号任务
You have new mail in /var/spool/mail/root
[root@localhost /]$ lpq # 查看打印队列, 4号任务已经取消
printer01 已准备就绪, 正在打印
顺序    所有者    作业    文件          总大小
active   root      2       5.c          1024 字节
[root@localhost /]$
```

cat - 连接多个文件并打印到标准输出

主要用途:

- 显示文件内容，如果没有文件或文件为-则读取标准输入。
- 将多个文件的内容进行连接并打印到标准输出。
- 显示文件内容中的不可见字符（控制字符、换行符、制表符等）。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
cat [OPTION]... [FILE]...
cat [选项]  file
cat file1  file2 > file3
```

sh

选项

```
-A, --show-all          # 等价于"-vET"组合选项。
-b, --number-nonblank  # 只对非空行编号，从1开始编号，覆盖"-n"选项。
-e                      # 等价于"-vE"组合选项。
-E, --show-ends         # 在每行的结尾显示'$'字符。
-n, --number            # 对所有行编号，从1开始编号。
-s, --squeeze-blank    # 压缩连续的空行到一行。
-t                      # 等价于"-vT"组合选项。
-T, --show-tabs         # 使用"^I"表示TAB（制表符）。
-u                      # POSIX兼容性选项，无意义。
-v, --show-nonprinting   # 使用"^"和"M-"符号显示控制字符,
                         # 除了LFD（line feed，即换行符'\n'）和TAB（制表符）。

--help                  # 显示帮助信息并退出。
--version                # 显示版本信息并退出。
```

sh

返回

返回状态为成功除非给出了非法选项或非法参数。

举例

常用用例

sh

```
# 合并显示多个文件
cat ./1.log ./2.log ./3.log
# 显示文件中的非打印字符、tab、换行符
cat -A test.log
# 压缩文件的空行
cat -s test.log
# 显示文件并在所有行开头附加行号
cat -n test.log
# 显示文件并在所有非空行开头附加行号
cat -b test.log
# 将标准输入的内容和文件内容一并显示
echo ##### | cat - test.log
```

sh

```
[sogrey@bogon 文档]$ ls
test.txt

[sogrey@bogon 文档]$ ls -lh test.txt
-rwxrwxrwx. 1 sogrey sogrey 250 1月 12 00:19 test.txt
[sogrey@bogon 文档]$ cat -b test.txt
1 石家庄今日新增16例确诊病例
2 中国留美博士遇害 美驻华使馆慰问
3 特朗普夫人发文谴责国会暴乱
4 理塘文旅公司回应丁真抽烟
5 北京一确诊者隐瞒行程不配合流调
6 山西晋中新增2例无症状感染者

[sogrey@bogon 文档]$ cat test.txt test2.txt
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
特朗普夫人发文谴责国会暴乱
```

1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。

```
[sogrey@bogon 文档]$ cat test.txt test2.txt > test3.txt
[sogrey@bogon 文档]$ cat -b test3.txt
1 石家庄今日新增16例确诊病例
2 中国留美博士遇害 美驻华使馆慰问
3 特朗普夫人发文谴责国会暴乱
4 理塘文旅公司回应丁真抽烟
5 北京一确诊者隐瞒行程不配合流调
6 山西晋中新增2例无症状感染者
7 特朗普夫人发文谴责国会暴乱
```

8 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。

```
[sogrey@bogon 文档]$
```

cd - 切换用户当前工作目录

切换目录，在切换之前确保有权利进入该目录。将当前目录更改为dir。变量HOME是默认的dir，变量CDPATH定义包含dir在内的目录的搜索路径。CDPATH中的替代目录名由冒号(:)分隔。CDPATH中的空目录名与当前目录相同，即“.”。如果dir以斜杠(/)开头，则不使用CDPATH。

主要用途：

- 切换工作目录至dir。其中dir的表示法可以是绝对路径或相对路径。
- 若参数dir省略，则默认为使用者的shell变量HOME。
- 如果dir指定为~时表示为使用者的shell变量HOME，.表示当前目录，..表示当前目录的上一级目录。
- 环境变量CDPATH是由冒号分割的一到多个目录，你可以将常去的目录的上一级加入到CDPATH以便方便访问它们；如果dir以/开头那么CDPATH不会被使用。
- 当shopt选项cdable_vars打开时，如果dir在CDPATH及当前目录下均不存在，那么会把它当作变量，读取它的值作为要进入的目录。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
cd [-L|-P] [dir]                                     sh
```

参数'-'相当于\$OLDPWD。如果使用来自CDPATH的非空目录名，或者如果'-'是第一个参数，并且目录更改成功，则将新工作目录的绝对路径名写入标准输出。如果成功更改目录，则返回值为true；否则为false。

选项

```
-L      # 强制遵循符号链接
-P      # 使用物理目录结构，而不是下面的符号链接
-      # 当前工作目录将被切换到环境变量OLDPWD所表示的目录，也就是前一个工作目录。                                     sh
```

举例

```
cd      # 进入用户主目录;
cd /    # 进入根目录
cd ~    # 进入用户主目录;
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。                                     sh
```


chattr - 改变文件的属性

用来改变文件属性。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。

这个命令只有超级用户才能使用。这个指令适用于ext2、ext3、ext4、xfs、ubifs、reiserfs、jfs系统。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
chattr [-RVf] [ -v version ] + | - | =[属性] file
```

sh

运算符'+'使所选属性被添加到文件的现有属性中；'-'使它们被删除；'='使它们成为文件所拥有的唯一属性。

选项

sh

```

-R          # 递归的方式修改目录及其子目录下的文件属性
-V          # 显示详细执行过程
-f          # 跳过错误信息
-v version # 设置文件或者目录的版本号

# 属性
a           # 只能以append的方式打开
A           # 最后修改时间不被记录
c           # 对文件进行压缩，而读取的时候会得到解压数据
D           # 同步目录更新
d           # 不进行备份
E           # 压缩文件有错误，这个属性不能被用户手动修改
e           # 文件使用扩展区来映射到磁盘
-I          # 目录使用hash树来索引，这个属性不能被用户手动修改
h           # 文件大小超过2TB，这个属性不能被用户手动修改
i           # 文件不能被修改，但是可以删除或者重命名
s           # 将数据块清零并存入磁盘
S           # 同步更新。将文件修改之后，结果写入磁盘
u           # 预防意外删除
-T          # 目录层次结构顶部
-X          # 压缩原始存取。虽然lsattr(1)可以显示它，  
# 但它目前不能使用chattr(1)进行设置或重置。
-Z          # 压缩脏文件。虽然它可以由lsattr(1)显示，  
# 但不能使用chattr(1)设置或重置

+<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。

```

举例

用chattr命令防止系统中某个关键文件被修改:

```
[sogrey@bogon 文档]$ chattr +i /etc/fstab
/home/sogrey/文档
```

sh

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:

```
[sogrey@bogon 文档]$ chattr +a /data1/user_act.log
/home/sogrey/文档
```

sh

增加a属性，设置版本号。显示详细的执行过程:

```
[sogrey@bogon demos]$ chattr -v 1 -V +a test.txt
chattr 1.45.0 (6-Mar-2019)
test.txt的标志被设为 -----a-----e-----
chattr: 不允许的操作 设置 test.txt 的标志时
[sogrey@bogon demos]$ > test.c
[sogrey@bogon demos]$ chattr -Vv 2 -a +c test.c
chattr 1.45.0 (6-Mar-2019)
test.c的标志被设为 -----c-----e-----
test.c 的版本被设置为 2
[sogrey@bogon demos]$
```

chfn - 用来改变finger命令显示的信息

chfn命令 用来改变finger命令显示的信息。这些信息都存放在/etc目录里的passwd文件里。若不指定任何选项，则chfn命令会进入问答式界面。

chfn指令可以改变通过finger指令查看到的信息。此信息存储在/etc/passwd文件中，并由Finger程序显示。Linux Finger命令将显示可由chfn更改的四条信息：您的真名、您的工作区和电话以及您的家庭电话。这四条信息中的任何一条都可以在命令行上指定。如果命令行上没有提供任何信息，chfn将进入交互模式。在交互模式下，chfn将提示每个字段。在提示符下，您可以输入新信息，也可以按“返回”使字段保持不变。输入关键字“None”使字段为空。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
chfn [选项] user
```

sh

选项

-f, --full-name	# 设置真实姓名
-o, --office	# 设置办公室地址
-h, --home-phone	# 设置家庭电话
-p, --office-phone	# 设置办公室电话
-u, --help	# 显示帮助文档
-v, --version	# 显示命令版本信息

sh

举例

修改用户真实姓名和家庭电话

```
[sogrey@bogon 文档]$ chfn -f sogrey -h 110 root    #修改信息
Changing finger information for root.
Finger information changed.
[sogrey@bogon 文档]$ finger root                  #查看信息，已经修改成功
Login: root           Name: sogrey
Directory: /root      Shell: /bin/bash
Office: 110
On since 五 9月 7 21:02 (CST) on tty1 from :0
  14 days 21 hours idle
On since 六 9月 22 07:36 (CST) on pts/0 from :0.0
  9 hours 12 minutes idle
On since 四 9月 13 08:55 (CST) on pts/1 from :0.0
New mail received 六 9月 22 18:05 2018 (CST)
  Unread since 二 8月 21 09:22 2018 (CST)
No Plan.
```

chgrp - 用来变更文件或目录的所属群组

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。

改变文件或者目录所属的群组，使用参数“`--reference`”，可以改变文件的群组为指定的关联文件群组。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
chgrp [选项] group file
chgrp [选项] --reference=FILE file
```

sh

选项

```
-c, --changes          # 和verbose一样，但是只有在发生改变的时候才显示详细信息
--dereference         # 改变符号链接所指向的文件，而不是符号链接自己。这是默认选项
-h, --no-dereference # 修改符号链接，仅适用于可更改符号链接所有权的系统
--no-preserve-root    # 不要特殊处理“/”，默认选项
--preserve-root       # 未能对“/”进行递归操作
-f, --silent, --quiet # 忽略部分错误信息
--reference=file      # 使用关联文件所属的组，而不是指出一个具体的值
-R, --recursive       # 递归处理目录及其内部的文件
-v, --verbose          # 显示详细信息
-H                   # 如果命令行参数是指向目录的符号链接，请遍历它。配合”-R”使用
-L                   # 遍历到遇到的目录的每个符号链接。配合”-R”使用
-P                   # 不要遍历任何符号链接(默认)。配合”-R”使用

--help                # 显示帮助文档
--version             # 显示命令版本信息
```

sh

举例

修改文件所属的组

```
[root@bogon 文档]$ ls -l #使用ls查看详细信息  
总用量 1072  
-rw-r--r-- 1 root root 0 9月 7 09:11 1.c  
-rw-r--r-- 3 root root 358400 9月 7 15:46 link  
[root@bogon 文档]$ chgrp sogrey 1.c #修改组  
[root@bogon 文档]$ ls -l #再次查看，修改已经成功  
总用量 1072  
-rw-r--r-- 1 root sogrey 0 9月 7 09:11 1.c  
-rw-r--r-- 3 root root 358400 9月 7 15:46 link
```

2) 使用选项“--reference”

```
[root@bogon 文档]$ chgrp --reference=1.c my.iso #1.c的组已经是sogrey了  
[root@bogon 文档]$ ls -l  
总用量 1072  
-rw-r--r-- 1 root sogrey 0 9月 7 09:11 1.c  
-rw-r--r-- 3 root sogrey 358400 9月 7 15:46 my.iso #my.iso所属的组也变成sogrey
```

3) 只修改符号链接自己

```
[root@localhost wj]$ ls -l 1.c 11.c #查看文件信息  
lrwxrwxrwx 1 root root 3 10月 26 10:11 11.c -> 1.c  
-rw-r--r-- 1 root root 0 10月 24 10:12 1.c  
[root@localhost wj]$ chgrp -h sogrey 11.c #修改组  
[root@localhost wj]$ ls -l 1.c 11.c #查看文件信息，只有符号链接自己的组被改了  
lrwxrwxrwx 1 root sogrey 3 10月 26 10:11 11.c -> 1.c  
-rw-r--r-- 1 root root 0 10月 24 10:12 1.c
```

chkconfig - 检查或设置系统的各种服务

chkconfig命令 检查、设置系统的各种服务。这是Red Hat公司遵循GPL规则所开发的程序，它可查询操作系统在每一个执行等级中会执行哪些系统服务，其中包括各类常驻服务。谨记chkconfig不是立即自动禁止或激活一个服务，它只是简单的改变了符号连接。

启动或者关闭系统服务，设置服务的运行级别，该指令并不会立刻启动或者停止服务，而是在开机的时候发生效果。

chkconfig提供了一个简单的命令行工具，用于维护/etc/rc[0-6].d目录层次结构，使系统管理员不必直接操作这些目录中的许多符号链接。这个chkconfig的实现受到IRIX操作系统中的chkconfig命令的启发。但是，这个版本没有在/etc/rc[0-6].d层次结构之外维护配置信息，而是直接管理/etc/rc[0-6].d中的符号链接。这将留下所有有关在单个位置启动服务init的配置信息。

chkconfig有五个不同的功能：添加用于管理的新服务、从管理中删除服务、列出服务的当前启动信息、更改服务的启动信息以及检查特定服务的启动状态。

当chkconfig后面只有一个服务名称的时候，它会检查服务是否配置为在当前运行级中启动。如果是，则chkconfig返回true；否则返回false。“--level”选项可以用来使chkconfig查询指定运行级下的服务状态，而不是当前的运行级。当使用“--list”参数运行chkconfig或根本没有参数时，将显示所有服务及其当前配置的清单。

如果在服务名称之后指定了on、off、reset或resetpriorities之一，则chkconfig将更改指定服务的启动信息。on和off标志将分别导致服务在被更改的运行级别中启动或停止。reset标志将服务的所有运行级别的on/off状态重置为init脚本文件中指定的样子，重置的时候会有一个询问。而resetpriorities标志则会直接将服务的on/off状态重置为init脚本中指定样子。默认情况下，on和off选项只影响运行级别2、3、4和5，而reset和resetpriorities影响所有运行级别。请注意，对于每个服务，每个运行级都有一个开始脚本或一个停止脚本。当切换runlevel时，init将不会重新启动已经启动的服务，也不会重新停止未运行的服务。

chkconfig还可以通过xinetd.d配置文件管理xinetd脚本，xinetd服务只支持on、off、--list。

chkconfig支持一个“--type”参数，当几个服务共享名字的情况下，这个选项就有效。支持该选项的服务可以是sysv和xinetd。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
chkconfig [--list] [--type type][name]
chkconfig --add name
chkconfig --del name
chkconfig --override name
chkconfig [--level levels] [--type type] name <on|off|reset|resetpriorities>
chkconfig [--level levels] [--type type] name
```

选项

```
--list 服务名          # 此选项列出chkconfig所知道的所有服务，以及它们是在每个运行级别中停止还是启动。  
--add 服务名          # 增加服务。当添加新服务时，chkconfig确保服务在每个运行级别上都有一个启动项或一  
--del 服务名          # 将服务从chkconfig管理中删除，与它相关的/etc/rc[0-6].d中的任何符号链接都被删  
--level [levels] [name] [on|off|reset] # 设置指定服务在指定运行级别的开机状态，级别可以是0~6。例如：ch  
--override name        # 如果/etc chkconfig.d/name文件现在存在，并且与基础配置脚本不同，则更改服务名  
  
--help                 # 显示帮助文档  
--version              # 显示命令版本信息
```

关于运行级别

每个应该由chkconfig管理的服务都需要在其init.d脚本中添加两行或者多行注释。第一行告诉chkconfig默认应该在什么运行级别启动服务，以及启动和停止优先级级别。如果服务在默认情况下不应该在任何运行级别中启动，则应该使用“-”来代替runlevel列表。第二行包含服务的描述，并且可以通过反斜杠继续扩展多行。例如random.init中有三行注释

```
# chkconfig: 2345 20 80  
# description: Saves and restores system entropy pool for \  
# higher quality random number generation.
```

这意味着随机脚本应该在级别2、3、4和5中启动，它的开始优先级应该是20，它的停止优先级应该是80。

chkconfig还支持从左到右的“-”分隔符，并将优先于可用的“chkconfig:”行应用它们。例如下面的：

```
### BEGIN INIT INFO  
# Provides: foo  
# Required-Start: bar  
# Default-Start: 2 3 4 5  
# Default-Stop: 0 1 6  
# Description: Foo init script  
### END INIT INFO
```

在这种情况下，“foo”的启动优先级将被更改，它将高于“bar”启动优先级。在添加依赖项时必须小心，因为它们可能导致许多脚本的启动和停止优先级发生巨大变化。

举例

[查看所有服务](#)

sh

```
[sogrey@bogon ~]$ chkconfig --list  
  
注: 该输出结果只显示 SysV 服务, 并不包含  
原生 systemd 服务。SysV 配置数据  
可能被原生 systemd 配置覆盖。  
  
要列出 systemd 服务, 请执行 'systemctl list-unit-files'。  
查看在具体 target 启用的服务请执行  
'systemctl list-dependencies [target]'。  
  
netconsole      0:关 1:关 3:关 4:关 5:关 6:关  
network        0:关 1:关 2:开 3:开 4:开 5:开 6:关  
[sogrey@bogon ~]$
```

查看指定服务

sh

```
[sogrey@bogon ~]$ chkconfig --list vsftpd  
vsftpd        0:关闭 1:关闭 2:关闭 3:关闭 4:关闭 5:启用 6:关闭  
[sogrey@bogon ~]$
```

设置ftp服务在3和5这两个级别启动

sh

```
[sogrey@bogon ~]$ chkconfig --level 35 vsftpd on # 级别3和5启动  
[sogrey@bogon ~]$ chkconfig --list vsftpd          # 查看是否设置成功  
vsftpd        0:关闭 1:关闭 3:启用 4:关闭 5:启用 6:关闭
```

chmod - 用来变更文件或目录的权限

改变文件或者目录的权限，可以用数字或者字母来标识权限。

- 通过符号组合的方式更改目标文件或目录的权限。
- 通过八进制数的方式更改目标文件或目录的权限。
- 通过参考文件的权限来更改目标文件或目录的权限。

在数字模式下：

- 0 代表没有权限；
- 1 代表可执行；
- 2 代表可读；
- 4 代表可写；

多个权限可以相加。

在字符模式下：

- x 代表执行；
- r 代表读；
- w 代表写；
- g 代表组权限；
- o 代表组内其他用户权限；
- u 代表用户权限

ls命令中，看到的权限分别是： 用户权限 、 组权限 、 组内其他用户权限 。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
chmod [选项] mode file
chmod [选项] 八进制模式 file
chmod [选项] -reference=FILE file
```

sh

选项

```
[sogrey@bogon 文档]$ chmod --help
用法: chmod [选项]... 模式[,模式]... 文件...
或: chmod [选项]... 八进制模式 文件...
或: chmod [选项]... --reference=参考文件 文件...
Change the mode of each FILE to MODE.
With --reference, change the mode of each FILE to that of RFILE.

-c, --changes          # like verbose but report only when a change is made
-f, --silent, --quiet   # suppress most error messages
-v, --verbose           # output a diagnostic for every file processed
--no-preserve-root      # do not treat '/' specially (the default)
--preserve-root         # fail to operate recursively on '/'
--reference=RFILE        # use RFILE's mode instead of MODE values
-R, --recursive         # change files and directories recursively
--help                  # 显示此帮助信息并退出
--version               # 显示版本信息并退出
```

Each MODE is of the form '[ugoa]*([+-=]([rwxXst]*|[ugo]))+|[-+=][0-7]+'.
 GNU coreutils online help: <<http://www.gnu.org/software/coreutils/>>
 请向<http://translationproject.org/team/zh_CN.html> 报告chmod 的翻译错误
 要获取完整文档, 请运行: info coreutils 'chmod invocation'

以上信息来自 CentOS Linux 8

补充说明

chmod根据模式更改每个给定文件的文件模式位，该模式可以是要进行的更改的符号表示，也可以是表示新模式位的位模式的八进制数。符号模式的格式是[ugoa...][[+-=][perms...]...], 其中perms为集合rwxXst的零个或多个字母，或来自结合ugo的单个字母。可以给出多种符号模式，用逗号分隔。

字母ugoa控制哪些用户访问文件的权限将被更改: (u)拥有该文件的用户、(g)文件组中的其他用户、(o)不属于文件组的其他用户或所有用户(A)。如果所有这些都没有给出，那么效果就好像给定了“a”，但是在umask中设置的位不受影响。

运算符'+'使所选的文件模式位被添加到每个文件的现有文件模式位中；'-'使它们被删除；'='使它们被添加，并使未提及的位被删除，除非目录的未提及的设置用户和组ID位不受影响。

字母“rwxXst”为受影响的用户选择文件模式位: (r)读、(w)写、(x)执行(或搜索目录)、(X)只在文件是目录或已对某些用户具有执行权限、(s)在执行时设置用户或组ID、(t)限制删除标志或粘性位时执行/搜索。您可以指定“ugo”其中的一个或多个字母: (u)授予拥有文件的用户的权限, (g)授予属于文件组的其他用户的权限, (o)授予上述两个类别中任何一个用户的权限。

数字模式是1到4个八进制数字(0-7)，通过将值4、2和1的位相加而得。省略的数字被假定为前导零。第一个数字选择设置用户ID(4)和设置组ID(2)，并限制删除或粘贴(1)属性。第二位数为拥有该文件的用户选择权限：读(4)、写(2)和执行(1)；第三位数选择文件组中具有相同值的其他用户的权限；第四位数字选择对不属于文件组的其他用户具有相同值的权限。

chmod从不更改符号链接的权限；chmod系统调用不能更改它们的权限。这不是一个问题，因为符号链接的权限从未被使用过。但是，对于命令行中列出的每个符号链接，chmod会更改指向文件的权限。相反，chmod忽略递归目录遍历过程中遇到的符号链接。

设置uid和gid

如果文件的组ID与用户的有效组ID或用户的辅助组ID不匹配，则chmod将清除常规文件的“set-group-ID”位，除非用户具有适当的权限。附加限制可能导致MODE或RFILE的“set-user-ID”和“set-group-ID”位被忽略。此行为取决于底层chmod系统调用的策略和功能。当有疑问时，检查底层系统行为。

chmod保留目录的“set-user-ID”和“set-group-ID”位，除非您显式地另外指定。您可以使用“u+s”和“g-s”这样的符号模式设置或清除位，也可以用数字模式设置(但不清楚)这些位。

限制删除标志或粘性位

受限制的删除标志或粘性位是一个位，其解释取决于文件类型。对于目录，它防止非特权用户删除或重命名目录中的文件，除非他们拥有该文件或目录；这称为目录的限制删除标志，通常在/tmp之类的可写目录中找到。对于一些旧系统上的常规文件，这一位将程序的文本映像保存在交换设备上，以便在运行时更快地加载；这称为粘性位。

举例

```
[sogrey@bogon 文档]$ ls  
ee.txt  
[sogrey@bogon 文档]$ ls -lh ee.txt  
-rw-----. 1 sogrey sogrey 12 1月 10 18:45 ee.txt  
[sogrey@bogon 文档]$ chmod 777 ee.txt  
[sogrey@bogon 文档]$ ls -lh ee.txt  
-rwxrwxrwx. 1 sogrey sogrey 12 1月 10 18:45 ee.txt
```

chown - 用来变更文件或目录的拥有者或所属群组

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。用户可以是用户名或者是用户ID，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使用该命令。

改变文件或者目录的所有者，或者所属的群组。如果只给出一个所有者(用户名或数字用户ID)，则该用户将成为每个给定文件的所有者，并且文件的组不会被更改。如果所有者后面跟着冒号和组名(或数字组ID)，在它们之间没有空格，那么文件的组所有权也会被更改。如果用户名后面有冒号，但没有组名，则使该用户成为文件的所有者，并将文件组更改为该用户的登录组。如果给出冒号和组，但省略了所有者，则只更改文件组；在本例中，Chown执行与chgrp相同的功能。如果只给出冒号，或者整个操作数为空，则所有者和组都不会更改。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
chmod [选项] [owner: group], [owner], [.group] file
chmod [选项] -reference=FILE file
```

sh

选项

```
-c, --changes          # 和verbose一样，但是只有在发生改变的时候才显示详细信息
--dereference         # 修改符号链接所指的对象
-h, --no-dereference # 修改符号链接
--from=CURRENT_OWNER:CURRENT_GROUP # 仅当每个文件的当前所有者和组匹配此处指定的所有者和组时，才更改其所有者和组
--preserve-root        # 禁止对根目录递归炒作
--no-preserve-root    # 不特殊对待根目录，默认值
-f, --silent, --quiet # 忽略部分错误信息
--reference=file       # 使用关联文件所属的组
-v, --verbose          # 显示详细信息
-R, --recursive        # 用递归的方式改变所有的目录和子目录
-H                   # 如果命令行参数是指向目录的符号链接，请遍历它。配合“-R”使用。
-L                   # 遍历到遇到的目录的每个符号链接，配合“-R”使用。
-P                   # 不要遍历任何符号链接(默认)，配合“-R”使用。

--help                # 显示帮助文档
--version             # 显示命令版本信息
```

sh

举例

通过名称改变组信息

```
[root@bogon 文档]$ ls -l 1.c      #当前的组是sogrey  
-rwxr--r-- 1 root sogrey 0 9月 7 09:11 1.c  
[root@bogon 文档]$ chown .david 1.c    #把组改为david。注意语法，前面有个点  
[root@bogon 文档]$ ls -l 1.c  
-rwxr--r-- 1 root david 0 9月 7 09:11 1.c
```

2) 通过id修改组信息，id可以从/etc/group中找到

```
[root@bogon 文档]$ ls -l 1.c      #当前的组是sogrey  
-rwxr--r-- 1 root sogrey 0 9月 7 09:11 1.c  
[root@bogon 文档]$ chown .500 1.c  #把组改为500，这个id对应的名称就是david。注意语法，前面有个点  
[root@bogon 文档]$ ls -l 1.c  
-rwxr--r-- 1 root david 0 9月 7 09:11 1.c
```

3) 通过名称改所有者

```
[root@bogon 文档]$ ls -l 1.c      #当前的所有者，root  
-rwxr--r-- 1 root sogrey 0 9月 7 09:11 1.c  
[root@bogon 文档]$ chown david 1.c  #把所有者改为david。  
[root@bogon 文档]$ ls -l 1.c  
-rwxr--r-- 1 david sogrey 0 9月 7 09:11 1.c
```

4) 同时改变所有者和组

```
[root@bogon 文档]$ ls -l 1.c      #当前的组是sogrey  
-rwxr--r-- 1 root sogrey 0 9月 7 09:11 1.c  
[root@bogon 文档]$ chown 500:500 1.c  #把组和所有者都改为500，注意语法，中间有个冒号  
[root@bogon 文档]$ ls -l 1.c  
-rwxr--r-- 1 david david 0 9月 7 09:11 1.c
```

chsh - 用来更换登录系统时使用的shell

chsh命令 用来更换登录系统时使用的shell。若不指定任何参数与用户名，则chsh会以应答的方式进行设置。

改变用户登录时使用的shell， 默认使用bash。如果命令行上没有给出shell， chsh将提示输入一个shell。chsh将接受系统上任何可执行文件的完整路径名。但是， 如果shell未在“/etc/shell”文件中列出，则将发出警告。另一方面， 也可以将其配置为只接受此文件中列出的shell， 除非您是root用户。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
chsh [选项] user
```

sh

选项

-s, --shell	# 改变登录后使用的shell环境
-l, --list-shells	# 显示系统当前可以用的shell
-u, --help	# 显示帮助文档
-v, --version	# 显示命令版本信息

sh

举例

查看可用的shell

```
[sogrey@bogon 文档]$ chsh -l david #显示当可用shell  
/bin/sh  
/bin/bash  
/sbin/nologin  
/bin/dash  
/bin/tcsh  
/bin/csh
```

sh

改变用户shell

```
[sogrey@bogon 文档]$ chsh -s /bin/sh david #改变登录shell  
Changing shell for david.  
Shell changed.  
[sogrey@bogon 文档]$ su david #切换用户，shell已经改  
sh-4.1$
```

sh

把我的shell改成zsh

```
[sogrey@bogon 文档]$ chsh -s /bin/zsh  
Changing shell for rocrocket.  
Password:  
Shell changed.  
[sogrey@bogon 文档]$
```

sh

使用chsh加选项-s就可以修改登录的shell了！你会发现你现在执行echo \$SHELL后仍然输出为/bin/bash，这是因为你需要重启你的shell才完全投入到zsh怀抱中去。chsh -s其实修改的就是/etc/passwd文件里和你的用户名相对应的那一行。

把shell修改回/bin/bash

```
[sogrey@bogon 文档]$ chsh -s /bin/bash  
Changing shell for rocrocket.  
Password:  
Shell changed.
```

sh

cksum - 检查文件的crc是否正确

检查文件的crc是否正确，统计文件的字节数.

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
cksum [选项] file
```

sh

选项

```
--help          # 显示帮助文档  
--version       # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon 文档]$ ls  
backup demos  
[sogrey@bogon 文档]$ cd demos  
[sogrey@bogon demos]$ ls  
test2.txt test3.txt test4.txt test.c test.txt  
[sogrey@bogon demos]$ cksum test.c  
# crc校验 字节数 文件名  
4294967295 0 test.c  
[sogrey@bogon demos]$ cksum test.txt  
# crc校验 字节数 文件名  
3701171500 250 test.txt  
[sogrey@bogon demos]$
```

sh

clear - 清除当前屏幕终端上的任何信息

clear命令 用于清除当前屏幕终端上的任何信息。

清除屏幕信息。如果可能的话，清除屏幕。它会在环境中查找终端类型，然后在终端数据库中找出如何清除屏幕。clear忽略可能存在的任何命令行参数。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
clear                                         sh
```

选项

无

举例

输入clear指令后回车，屏幕的信息就会被清除。

```
clear                                         sh
```

cmp - 比较两个文件是否有差异

用字节的方式，比较两个文件是否存在差异，但是不保存运算结果。Cmp指令只会根据结果设置相关的标志位，这个指令之后往往会跟着一个条件跳转指令。

逐行比较两个已经排序过的文件。结果以3列显示：第1列显示只在file1出现的内容，第2列显示只在file2出现的内容，第3列显示同时出现的内容。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
cmp [OPTION]... FILE1 [FILE2 [SKIP1 [SKIP2]]]
```

sh

选项

```
-b, --print-bytes          # 输出不同的字节
-i num, --ignore-initial=num # 跳过开始的num个字节
-i num1 num2, --ignore-initial=num1: num2 #第一个文件跳过num1个字节，第二个文件跳过num2个字节
-l, --verbose               # 输出不同之处的字节序号，以及这个字节的值
-n LIMIT, --bytes=LIMIT     # 最多比较LIMIT字节
-s, --quiet, --silent       # 不输出；只输出退出状态

--help                      # 显示帮助文档
--version                   # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon backup]$ diff -y test.txt test2.txt
石家庄今日新增16例确诊病例           <
中国留美博士遇害 美驻华使馆慰问           <
特朗普夫人发文谴责国会暴乱      特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟           |
123          | 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上
北京一确诊者隐瞒行程不配合流调           <
山西晋中新增2例无症状感染者           <

[sogrey@bogon backup]$ cmp test.txt test2.txt
test.txt test2.txt 不同: 第 2 字节, 第 1 行
[sogrey@bogon backup]$ cmp -b test.txt test2.txt
test.txt test2.txt 不同: 第 1 行, 第 2 字节为 237 M-^_ 211 M-^I
[sogrey@bogon backup]$ cmp -lb test.txt test2.txt
 2 237 M-^_ 211 M-^I
 3 263 M-3  271 M-9
 4 345 M-e   346 M-f
```

sh

5 256 M-. 234 M-^\n6 266 M-6 227 M-^W\n7 345 M-e 346 M-f\n8 272 M-: 231 M-^Y\n9 204 M-^D 256 M-.\n10 344 M-d 345 M-e\n11 273 M-; 244 M-\$\n12 212 M-^J 253 M-+\n13 346 M-f 344 M-d\n14 227 M-^W 272 M-:\n15 245 M-% 272 M-:\n16 346 M-f 345 M-e\n17 226 M-^V 217 M-^O\n18 260 M-0 221 M-^Q\n19 345 M-e 346 M-f\n20 242 M-'' 226 M-^V\n21 236 M-^^ 207 M-^G\n22 61 1 350 M-h\n23 66 6 260 M-0\n24 344 M-d 264 M-4\n25 276 M-> 350 M-h\n26 213 M-^K 264 M-4\n27 347 M-g 243 M-#\n28 241 M-! 345 M-e\n29 256 M-. 233 M-^[\n30 350 M-h 275 M-=\n31 257 M-/ 344 M-d\n32 212 M-^J 274 M-
\n33 347 M-g 232 M-^Z\n34 227 M-^W 346 M-f\n35 205 M-^E 232 M-^Z\n36 344 M-d 264 M-4\n37 276 M-> 344 M-d\n38 213 M-^K 271 M-9\n39 12 ^J 261 M-1\n40 344 M-d 12 ^J\n41 270 M-8 12 ^J\n42 255 M-- 61 1\n43 345 M-e 346 M-f\n44 233 M-^[234 M-^\\\n45 275 M-= 210 M-^H\n46 347 M-g 61 1\n47 225 M-^U 61 1\n48 231 M-^Y 346 M-f\n49 347 M-g 227 M-^W\n50 276 M-> 245 M-%\n51 216 M-^N 357 M-o\n52 345 M-e 274 M-
\n53 215 M-^M 214 M-^L\n54 232 M-^Z 347 M-g\n55 345 M-e 276 M->\n56 243 M-# 216 M-^N\n57 253 M-+ 345 M-e\n58 351 M-i 233 M-^[\n59 201 M-^A 275 M-=\n60 207 M-^G 347 M-g\n61 345 M-e 254 M-,\n62 256 M-. 254 M-,\n63 263 M-3 344 M-d

64	40	270	M-8	
65	347	M-g	200	M-^@
66	276	M->	345	M-e
67	216	M-^N	244	M-\$
68	351	M-i	253	M-+
69	251	M-)	344	M-d
70	273	M-;	272	M-:
71	345	M-e	272	M-:
72	215	M-^M	346	M-f
73	216	M-^N	242	M-"
74	344	M-d	205	M-^E
75	275	M-=	346	M-f
76	277	M-?	213	M-^K
77	351	M-i	211	M-^I
78	246	M-&	345	M-e
79	206	M-^F	260	M-0
80	346	M-f	274	M-<
81	205	M-^E	345	M-e
82	260	M-0	250	M-(
83	351	M-i	205	M-^E
84	227	M-^W	302	M-B
85	256	M-.	267	M-7
86	12	^J	347	M-g
87	347	M-g	211	M-^I
88	211	M-^I	271	M-9
89	271	M-9	346	M-f
90	346	M-f	234	M-^\ 91 234 M-^\ 92 227 M-^W 346 M-f 93 346 M-f 231 M-^Y 94 231 M-^Y 256 M-. 95 256 M-. 351 M-i 96 345 M-e 200 M-^@ 97 244 M-\$ 232 M-^Z 98 253 M-+ 350 M-h 99 344 M-d 277 M-? 100 272 M-: 207 M-^G 101 272 M-: 347 M-g 102 345 M-e 231 M-^Y 103 217 M-^O 275 M-= 104 221 M-^Q 345 M-e 105 346 M-f 256 M-. 106 226 M-^V 253 M-+ 107 207 M-^G 345 M-e 108 350 M-h 217 M-^O 109 260 M-0 221 M-^Q 110 264 M-4 350 M-h 111 350 M-h 241 M-! 112 264 M-4 250 M-(113 243 M-# 345 M-e 114 345 M-e 243 M-# 115 233 M-^[260 M-0 116 275 M-= 346 M-f 117 344 M-d 230 M-^X 118 274 M-< 216 M-^N 119 232 M-^Z 357 M-o 120 346 M-f 274 M- 121 232 M-^Z 214 M-^L 122 264 M-4 350 M-h 123 344 M-d 260 M-0

```
124 271 M-9 264 M-4
125 261 M-1 350 M-h
126 12 ^J 264 M-4
127 347 M-g 243 M-#
128 220 M-^P 344 M-d
129 206 M-^F 270 M-8
130 345 M-e 212 M-^J
131 241 M-! 345 M-e
132 230 M-^X 221 M-^Q
133 346 M-f 250 M-( 
134 226 M-^V 345 M-e
135 207 M-^G 217 M-^O
136 346 M-f 221 M-^Q
137 227 M-^W 347 M-g
138 205 M-^E 224 M-^T
139 345 M-e 237 M-^_
140 205 M-^E 345 M-e
141 254 M-, 234 M-^\
142 345 M-e 250 M-( 
143 217 M-^O 347 M-g
144 270 M-8 276 M->
145 345 M-e 216 M-^N
146 233 M-^[ 345 M-e
147 236 M-^^ 233 M-^[
148 345 M-e 275 M-= 
149 272 M-: 345 M-e
150 224 M-^T 233 M-^[
151 344 M-d 275 M-= 
152 270 M-8 344 M-d
153 201 M-^A 274 M-<
154 347 M-g 232 M-^Z
155 234 M-^` 347 M-g
156 237 M-^_ 232 M-^Z
157 346 M-f 204 M-^D
158 212 M-^J 346 M-f
159 275 M-= 232 M-^Z
160 347 M-g 264 M-4
161 203 M-^C 344 M-d
162 237 M-^_ 271 M-9
163 12 ^J 261 M-1
164 61 1 343 M-c
165 62 2 200 M-^@
166 63 3 202 M-^B
cmp: test2.txt 已结束
[sogrey@bogon backup]$
```

col - 过滤控制字符

col命令是一个标准输入文本过滤器，它从标注输入设备读取文本内容，并把内容显示到标注输出设备。在许多UNIX说明文件里，都有RLF控制字符。当我们运用shell特殊字符>和>>，把说明文件的内容输出成纯文本文件时，控制字符会变成乱码，col命令则能有效滤除这些控制字符。

过滤掉影响阅读的控制字符，使用重定向符把说明手册的内容输出到文本文件时，控制字符就成乱码。col指令可以过滤掉控制字符，使文本可读。col从标砖输入读取内容，输出到标准输出。col在读取字符时跟踪字符集，并确保字符集在输出时是正确的。如果输入试图备份到最后一条刷新行，col将显示一条警告消息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
col [-bfp] [-lnum] file
```

sh

选项

```
-b          # 过滤所有的控制字符  
-f          # 过滤掉RLF字符，忽略HRLF字符  
-p          # 忽略未知的控制字符  
-x          # 将多个空格用tab代替  
-lnum       # 设置缓冲区大小，默认128行  
  
--help      # 显示帮助文档  
--version   # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon demo4]$ man  
您需要什么手册页?  
[sogrey@bogon demo4]$ man col  
[sogrey@bogon demo4]$ man col > col.txt # 将手册从定向到文件  
<standard input>:58: warning [p 1, 2.0i]: cannot adjust line  
<standard input>:74: warning [p 1, 3.5i]: cannot adjust line  
[sogrey@bogon demo4]$ ls  
col.txt  hello.txt  test2.txt  
[sogrey@bogon demo4]$ cat col.txt  
col(1)                                     General Commands Manual           col(1)
```

sh

NAME(名
 col - 过滤掉输入中的反向换行符

SYNOPSIS(总
 col [**-bfx**] [**-l num**]

DESCRIPTION(描
 col 过滤掉反向(以及半反向)换行符(LF: line feed or NL: new line),
 这样输出按正常顺序, 即只包括正向和半正向换行符,
 而且在可能的地方使用tab替换白空格.这对 nroff(**1**) 和 tbl(**1**) 的输出处理很有用处.

col 从标准输出读入, 并写出到标准输出上.

选项如下:

-b 不输出任何退格符, 在每列的位置上只打印最后写的那个字符.

-f 允许正向半换行符(``fine''模式).
 通常, 处于平行分界线上的字符打印在下一行.

-x 输出多个空格以替换tab.

-l num 在内存中至少缓冲 num 行. 默认情况下, 缓冲128行.

col 所能理解的用于回车操作的控制序列以及它们的十进制值都列在下面的表中:

ESC-7 反向换行符(**escape**后接7)

ESC-8 反向半换行符(**escape**后接8)

ESC-9 正向半换行符(**escape**后接9)

backspace
 反向移动一列(**8**);在第一列则忽略.

carriage return
 (**13**)

newline
 正向换行符(**10**);同时执行回车(**carriage return**)操作

shift in
 转到正常字符集(**15**)

shift out
 转到备选的字符集(**14**)

space 正向移动一列(**32**)

tab 正向移动到下一个tab(**9**)

vertical tab
 反向换行符(**11**)

丢弃所有不被承认的控制字符和**escape**序列.

当读取字符时, **col** 保持着与字符集的联系, 而且在输出时确保字符集是正确的.

如果输入设备试图回复到最近被刷新的行, **col** 会显示一条警告消息.

SEE ALSO(另

expand(1) nroff(1) tbl(1)

HISTORY(历

col 命令出现于AT&T UNIX版本6.

[中

riser <boomer@ccidnet.com>

[中

2000/12/6

《

<http://cmpp.linuxforum.net>

COL 1

1991年6月17日

col(1)

[sogrey@bogon demo4]\$

colrm - 删除文件中的指定列

从标准输入读取数据，删除指定的列，然后送到标准输出。如果用一个参数调用，则将从指定的列开始删除每一行的列。如果使用两个参数调用，则将删除从第一列到最后一列的列。列编号以第1列开始。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
colrm [startcol] [endcol]
```

sh

举例

```
[sogrey@bogon demo4]$ cat hello.txt # 显示内容
Hello world!
Ok,fine.
[sogrey@bogon demo4]$ colrm 3 < hello.txt # 删除第3列之后
He
Ok
[sogrey@bogon demo4]$ colrm 3 5 < hello.txt # 删除3-5列
He world!
Okne.
[sogrey@bogon demo4]$ colrm < hello.txt # 不指定列，显示所有的
Hello world!
Ok,fine.
[sogrey@bogon demo4]$
```

sh

comm - 逐行比较两个已经排序过的文件

逐行比较两个已经排序过的文件。结果以3列显示：第1列显示只在file1出现的内容，第2列显示只在file2出现的内容，第3列显示同时出现的内容。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
comm [OPTION]... FILE1 FILE2
```

sh

选项

```
-1          # 不显示第一个文件中出现的内容
-2          # 不显示第二个文件出现的内容
-3          # 不显示同时出现的内容
--check-order      # 检查输入是否正确排序，即使所有输入行都已经配对
--nocheck-order    # 不检查输入是否正确排序
--output-delimiter=STR  # 使用STR将列分割

--help        # 显示帮助文档
--version     # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon backup]$ diff -y test.txt test2.txt
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱    特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
123 | 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者

[sogrey@bogon backup]$ comm test.txt test2.txt
石家庄今日新增16例确诊病例
特朗普夫人发文谴责国会暴乱
comm: 文件2 没有被正确排序
```

```
1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。
中国留美博士遇害 美驻华使馆慰问
comm: 文件1 没有被正确排序
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
123
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者

[sogrey@bogon backup]$
```

compress - 使用Lempress-Ziv编码压缩数据文件

compress命令 使用“Lempress-Ziv”编码压缩数据文件。compress是个历史悠久的压缩程序，文件经它压缩后，其名称后面会多出“.Z”的扩展名。当要解压缩时，可执行uncompress指令。事实上uncompress是指向compress的符号连接，因此不论是压缩或解压缩，都可通过compress指令单独完成。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
compress [OPTION] [参数]
```

sh

选项

```
-f:            # 不提示用户，强制覆盖掉目标文件;  
-c:            # 将结果送到标准输出，无文件被改变;  
-r:            # 递归的操作方式;  
-b<压缩效率> # 压缩效率是一个介于9~16的数值，预设值为"16"，指定愈大的数值，压缩效率就愈高;  
-d:            # 对文件进行解压缩而非压缩;  
-v:            # 显示指令执行过程;  
-V:            # 显示指令版本及程序预设值。
```

sh

举例

```
[sogrey@bogon 文档]$ compress man.config # 压缩  
[sogrey@bogon 文档]$ ls -l  
-rw-r--r-- 1 sogrey sogrey 2605 Jul 27 11:43 man.config.Z  
[sogrey@bogon 文档]$ compress -d man.config.Z # 解压
```

sh

将 man.config 压缩成另外一个文件来备份

```
[sogrey@bogon 文档]$ compress -c man.config > man.config.back.Z  
[sogrey@bogon 文档]$ ll man.config*  
-rw-r--r-- 1 sogrey sogrey 4506 Jul 27 11:43 man.config  
-rw-r--r-- 1 sogrey sogrey 2605 Jul 27 11:46 man.config.back.Z
```

sh

这个-c的选项比较有趣！会将压缩过程的资料输出到屏幕上，而不是写入成为file.Z文件。所以，我们可以透过资料流重导向的方法将资料输出成为另一个档名。

cp - 复制文件

可以将一个文件复制到另外一个地方，也可以将多个文件复制到目录。

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。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
cp [option] [-T] src dst
cp [option] src dir
cp [option] -t dir src
```

sh

选项

```

-a, --archive          # 等价于“-dR --preserve=all”
--backup[=CONTROL]    # 为每一个存在的目标文件创建备份
-b                   # 类似“--backup”，但是没有参数
--copy-contents       # 递归时复制特殊文件的内容
-d                   # 等价于“--no-dereference --preserve=links”
-f, --force           # 强制执行
-i, --interactive     # 交互模式，覆盖文件之前询问
-H                   # 遵循src中的命令行符号链接。
-l, --link             # 创建链接，不复制
-L, --dereference     # 始终遵循src中的符号链接
-n, --no-clobber      # 不覆盖已经存在的文件
-R, -r                # 递归模式，复制子目录
-s, --symbolic-link   # 创建符号链接，不复制
-P, --no-dereference  # 不遵循src中的符号链接
-p                   # 等价于“--preserve=mode”
--preserve[=ATTR_LIST] # 保留指定的属性(默认：模式、所有权、时间戳)，
                      # 如果可能的话，其他属性：上下文、链接、xattr、all
-c                   # 等价于“--preserve=context”
--no-preserve=ATTR_LIST # 不保留指定的属性
--parents            # 使用目录下的完整源文件名
-R, -r, --recursive   # 递归复制子目录
--reflink[=WHEN]       # 控制拷贝
--remove-destination  # 在尝试打开每个现有目标文件之前移动它(与“--force”相反)
--sparse=WHEN         # 控制稀疏文件的创建
--strip-trailing-slashes # 从每个源参数中移除任何尾随斜线。
-s, --symbolic-link   # 创建符号链接，不复制
-S, --suffix=SUFFIX    # 重写通常的备份后缀
-t, --target-directory=DIRECTORY # 将所有源参数复制到DIRECTORY中。
-T, --no-target-directory # 将目标文件当做普通文件
-t, --target-directory=dir # 复制所有的源文件到目录
-u, --update           # 以更新的方式复制
-v, --verbose          # 显示详细执行过程
-x, --one-file-system  # 保持在这个文件系统上
-Z, --context=CONTEXT   # 将副本的安全上下文设置为上下文

--help                 # 显示帮助文档
--version              # 显示命令版本

```

补充说明

当 `--reflink[=always]` 被指定时，执行一个轻量级副本，其中数据块仅在修改时被复制。如果这是不可能的，复制失败，或者如果 `--reflink=auto` 被指定，则返回到标准副本。

备份后缀为 `~`，除非设置 `--suffix` 或 `SIMPLE_BACKUP_SUFFIX`。版本控制方法可以通过“`--backup`”选项或通过`VERSION_CONTROL`环境变量来选择。以下是这些值：

- 1) `none, off`, 从不备份，即使指定了 `--backup`。
- 2) `numbered, t`, 数字版本控制。
- 3) `existing, nil`, 如果有数字备份，那么就使用数字备份，否则使用简单备份。
- 4) `simple, never`, 简单备份。

作为特例，当提供强制和备份选项时，`cp`对源进行备份，`src`和`dst`名字相同，都是常规文件。

举例

```
[sogrey@bogon 文档]$ ls
demos test2.txt test3.txt test.txt
[sogrey@bogon 文档]$ ls ./demos
[sogrey@bogon 文档]$ cp test.txt test4.txt
[sogrey@bogon 文档]$ ls
demos test2.txt test3.txt test4.txt test.txt
[sogrey@bogon 文档]$ cp -t demos *.txt
[sogrey@bogon 文档]$ cd ./demos
[sogrey@bogon demos]$ ls
test2.txt test3.txt test4.txt test.txt
[sogrey@bogon demos]$ cd ..
[sogrey@bogon 文档]$ cp -r ./demos/ ./backup/
[sogrey@bogon 文档]$ ls ./backup/
test2.txt test3.txt test4.txt test.txt
[sogrey@bogon 文档]$
```

cpio - 用来建立、还原备份档的工具程序

cpio命令主要是用来建立或者还原备份档的工具程序，cpio命令可以复制文件到归档包中，或者从归档包中复制文件。

从归档中复制文件，或者复制文件到归档中。

Cpio命令有三种工作模式：

1. copy-out mode, cpio指令将文件复制到归档。它读取标准输入上的文件名列表(每行一个)，将归档包写到标准输出。
2. copy-in mode, cpio指令从归档中复制文件，从标准输入读取归档包。
3. cpio-pass mode, 从目录树复制文件到另一个目录，它从标准输入中读取要复制的文件列表。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
cpio -o namelist > archive
cpio -i < archive
cpio -p dst-dir < namelist
```

选项

```
-0, --null                                # 接受新增列控制字符，通常配合find指令的“-print0”参数使用;
-a, --rest-access-time
-A, --append
-b, --awap
-B
-c
-C<区块大小>, --io-size=<区块大小>    # 设置输入/输出的区块大小，单位是Byte;
-d, --make-directories                      # 如有需要cpio会自行建立目录;
-E<范本文件>, --pattern-file=<范本文件># 指定范本文件，其内含有一个或多个范本样式，让cpio解开符合范本条件
-f, --nonmatching                           # 让cpio解开所有不符合范本条件的文件;
-F<备份档>, --file=<备份档>
-H<备份格式>
-i, --extract
-l<备份档>
-k
-l, --link
-L, --dereference                          # 不建立符号连接，直接复制该连接所指向的原始文件;
-m, preserve-modification-time            # 不去更改文件的更改时间;
-M<回传信息>, --message=<回传信息>    # 设置更换保存媒体的信息;
-n, --numeric-uid-gid
-o, --create
-O<备份档>
-p, --pass-through                         # 执行copy-pass模式，略过备份步骤，直接将文件复制到目的目录;
-r, --rename
-R<拥有者><.:><所属群组>, ----owner<拥有者><.:><所属群组> # 在copy-in模式还原备份档，或copy-pass模式下运行;
-s, --swap-bytes                           # 交换每队字节的内容;
-S, --swap-halfwords                       # 交换每半个字节的内容;
-t, --list
-u, --unconditional                        # 置换所有文件，不论日期时间的新旧与否，皆不予询问而直接覆盖;
-v, --verbose
-V, --dot
--block-size=<区块大小>
--force-local
--no-absolute-filenames
--no-preserve-owner
-only-verify-crc
--quiet
--sparse
--help
--version
```

sh

用法

将/etc下的所有普通文件都备份到/opt/etc.cpio，使用以下命令：

```
find /etc -type f | cpio -ocvB >/opt/etc.cpio
```

sh

将系统上所有资料备份到磁带机内，使用以下命令：

```
find / -print | cpio -covB > /dev/st0
```

sh

这里的/dev/st0是磁带的设备名，代表SCSI磁带机。

查看上例磁带机上备份的文件，使用以下命令：

```
cpio -icdvt < /dev/st0 > /tmp/st_content
```

sh

有时可能因为备份的文件过多，一个屏幕无法显示完毕，此时我们利用下面命令，让磁带机的文件信息输出到文件。

将示例1中的备份包还原到相应的位置，如果有相同文件进行覆盖，使用以下命令：

```
cpio -icduv < /opt/etc.cpio
```

sh

注意，cpio恢复的路径，如果cpio在打包备份的时候用的是绝对路径，那么在恢复的时候会自动恢复到这些绝对路径下，本例就会将备份文件全部还原到/etc路径下对应的目录中。同理，如果在打包备份用的是相对路径，还原时也将恢复到相对路径下。

通过上面的示例，可以看出，cpio无法直接读取文件，它需要每个文件或者目录的完整路径名才能识别读取，而find命令的输出刚好做到了这点，因此，cpio命令一般和find命令配合使用。其实，上面的示例我们已经看到了它们的组合用法。

归档当前目录下的内容，并且制定输出文件

```
% ls | cpio -ov > directory.cpio
```

sh

存档整个目录树，find命令可以将文件列表提供给cpio。这将获取当前目录中的所有文件，以及下面的目录，并将它们放置在归档目录tree.cpio中

```
% find . -print -depth | cpio -ov > tree.cpio
```

sh

这将检索存档在文件directory.cpio中的文件，并将它们放在当前目录中

```
% cpio -iv < directory.cpio
```

sh

这将获取存档tree.cpio的内容，并将其解压缩到当前目录。

```
% cpio -idv < tree.cpio
```

sh

将当前目录的文件和子目录复制到一个名为new-dir的新目录中

```
% find . -depth -print0 | cpio --null -pvd new-dir
```

sh

实例

将当前目录归档

sh

```
[sogrey@bogon 文档]$ ll  
总用量 12  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 9 00:23 run.sh  
[sogrey@bogon 文档]$ ls | cpio -o > bak # 将ls的输出定向到cpio, 然后归档  
3 块  
[sogrey@bogon 文档]$ ll  
总用量 16  
-rw----- 1 sogrey sogrey 1536 3月 9 00:39 bak  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 9 00:23 run.sh  
[sogrey@bogon 文档]$
```

从归档中提取文件

sh

```
[sogrey@bogon 文档]$ cpio -i < bak # 从归档中提取  
cpio: 未创建 bak: 已有更新或同样新的版本存在  
cpio: 未创建 file1.txt: 已有更新或同样新的版本存在  
cpio: 未创建 file2.txt: 已有更新或同样新的版本存在  
cpio: 未创建 run.sh: 已有更新或同样新的版本存在  
3 块  
[sogrey@bogon 文档]$ ll  
总用量 16  
-rw----- 1 sogrey sogrey 1536 3月 9 00:39 bak  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 9 00:23 run.sh  
[sogrey@bogon 文档]$
```

拷贝文件

```
[sogrey@bogon 文档]$ ll
总用量 24
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 3.c
-rw-----. 1 sogrey sogrey 1536 3月 9 00:39 bak
drwx-----. 2 sogrey sogrey 4096 3月 9 00:46 bakDir
-rw-----. 1 sogrey sogrey 415 3月 8 23:50 file1.txt
-rw-----. 1 sogrey sogrey 576 3月 8 23:51 file2.txt
-rw-----. 1 sogrey sogrey 12 3月 9 00:43 list.txt
-rwx-----. 1 sogrey sogrey 90 3月 9 00:23 run.sh
[sogrey@bogon 文档]$ cat list.txt
1.c
2.c
3.c
[sogrey@bogon 文档]$ cpio -p bakDir/ < list.txt # 将list.txt中的内容当做文件列表，然后拷贝到bakDir目录下
0 块
[sogrey@bogon 文档]$ ll -R
.:
总用量 24
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 3.c
-rw-----. 1 sogrey sogrey 1536 3月 9 00:39 bak
drwx-----. 2 sogrey sogrey 4096 3月 9 00:46 bakDir
-rw-----. 1 sogrey sogrey 415 3月 8 23:50 file1.txt
-rw-----. 1 sogrey sogrey 576 3月 8 23:51 file2.txt
-rw-----. 1 sogrey sogrey 12 3月 9 00:43 list.txt
-rwx-----. 1 sogrey sogrey 90 3月 9 00:23 run.sh

./bakDir:
总用量 0
-rw-----. 1 sogrey sogrey 0 3月 9 00:46 1.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:46 2.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:46 3.c
[sogrey@bogon 文档]$
```

csplit - 将一个大文件分割成小的碎片文件

csplit命令 用于将一个大文件分割成小的碎片，并且将分割后的每个碎片保存成一个文件。碎片文件的命名类似“xx00”，“xx01”。csplit命令是split的一个变体，split只能够根据文件大小或行数来分割，但csplit能够根据文件本身特点来分割文件。

将文件按照指定的模式分割，默认的输出文件名是xx00、xx01、xx02等，指令会显示每个输出文件的大小。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
csplit [选项] file 格式
```

sh

选项

```
-b, --suffix-format=FORMAT      # 使用sprintf格式而不是%02d
-f name, --prefix=PREFIX        # 指定输出文件的前缀名字，而不是使用“xx”
-k, --keep-files                # 不要删除错误的输出文件
-n num                          # 指定输出文件名的字符数，这里指文件名中的序号长度
-z, --elide-empty-files         # 删除空文件
-s, --quiet, --silent           # 不显示输出文件的大小

--help                           # 显示帮助文档
--version                        # 显示命令版本信息
```

sh

格式

- 整数： 不包括指定的行，并以其为文件分块边界
- /表达式/[偏移量]： 不包括匹配到的行，并以其为文件分块边界
- %表达式%[偏移量]： 预先跳过匹配的行数，以其为文件分块边界
- {整数}： 将之前指定的模式重复指定的次数
- {*}： 将之前指定的模式重复尽可能多的次数

举例

```
[sogrey@bogon newDir]$ ls
test.txt
[sogrey@bogon newDir]$ cat test.txt # 查看文件内容
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon newDir]$ csplit test.txt 3 # 从第3行开始分割，输出分割后的文件大小
86
164
[sogrey@bogon newDir]$ ls
test.txt  xx00  xx01
[sogrey@bogon newDir]$ cat xx00  # 分别查看每个片段文件输出文件内容
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon newDir]$ cat xx01
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon newDir]$ csplit -n 3 -f new test.txt 3  # 指定输出文案前缀te，名字长度3个。这里指文件名
86
164
[sogrey@bogon newDir]$ ls
new000  new001  test.txt  xx00  xx01
[sogrey@bogon newDir]$
```

cupsenable - 启动指定的打印机

cupsenable命令 用于启动指定的打印机。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
cupsenable [OPTION] [参数]
```

sh

目标：指定目标打印机。

选项

```
-E      # 当连接到服务器时强制使用加密;  
-U      # 指定连接服务器时使用的用户名;  
-u      # 指定打印任务所属的用户;  
-h      # 指定连接的服务器名和端口号;
```

sh

cut - 连接文件并打印到标准输出设备上

cut 命令 用来显示行中的指定部分，删除文件中指定字段。cut 经常用来显示文件的内容，类似于 type 命令。

说明：该命令有两项功能，其一是用来显示文件的内容，它依次读取由参数 file 所指明的文件，将它们的内容输出到标准输出上；其二是连接两个或多个文件，如cut f1 f2 > f3 将把文件 f1 和 f2 的内容合并起来，然后通过输出重定向符“>”的作用，将它们放入文件 f3 中。

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。

pwd - 显示当前工作目录

查看用户当前的工作目录，输出完整路径。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
pwd [OPTION] [参数]
```

sh

选项

```
-b,--bytes=LIST          # 仅显示行中指定直接范围的内容;
-c,--characters=LIST     # 仅显示行中指定范围的字符;
-d,--delimiter=DELIM     # 指定字段的分隔符，默认的字段分隔符为“TAB”;
-f,--fields=LIST          # 显示指定字段的内容;
-n                         # 与“-b”选项连用，不分割多字节字符;
--complement                # 补足被选择的字节、字符或字段;
--out-delimiter= 字段分隔符 # 指定输出内容是的字段分割符;
-s, --only-delimited        # 如果该行没有分隔字符，那么不显示这行

--help                      # 显示帮助文档
--version                   # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ cat students.txt
No,Name,Percent
01,tom,69,91
02,jack,71,87
03,alex,68,98
[sogrey@bogon newDir3]$ cut -f2 -d"," students.txt # 使用 -d 选项指定字段分隔符
Name
tom
jack
alex
[sogrey@bogon newDir3]$
```

指定字段的字符或者字节范围

cut 命令可以将一串字符作为列来显示，字符字段的记法：

- N-：从第 N 个字节、字符、字段到结尾；
- N-M：从第 N 个字节、字符、字段到第 M 个（包括 M 在内）字节、字符、字段；
- -M：从第 1 个字节、字符、字段到第 M 个（包括 M 在内）字节、字符、字段。

上面是记法，结合下面选项将摸个范围的字节、字符指定为字段：

- -b 表示字节；
- -c 表示字符；
- -f 表示定义字段。

```
[sogrey@bogon newDir3]$ cat test.txt
abcdefghijklmнопqrstuvwxyz
abcdefghijklmнопqrstuvwxyz
abcdefghijklmнопqrstuvwxyz
abcdefghijklmнопqrstuvwxyz
abcdefghijklmнопqrstuvwxyz
[sogrey@bogon newDir3]$ cut -c1-3 test.txt # 打印第 1 个到第 3 个字符
abc
abc
abc
abc
abc
[sogrey@bogon newDir3]$ cut -c-2 test.txt # 打印前 2 个字符
ab
ab
ab
ab
ab
[sogrey@bogon newDir3]$ cut -c5- test.txt # 打印从第 5 个字符开始到结尾
efghijklmнопqrstuvwxyz
efghijklmнопqrstuvwxyz
efghijklmнопqrstuvwxyz
efghijklmнопqrstuvwxyz
efghijklmнопqrstuvwxyz
[sogrey@bogon newDir3]$
```

date - 显示或设置系统时间与日期

以给定的格式显示当前的日期，或者设置系统时间。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
date [选项]
date [+格式]
date -u [参数]
```

sh

参数格式[MMDDhhmm[[CC]YY][.ss]]，分别对应（月、日、时、分、年前两位、年后两位、秒）

选项

```
-f, --file=DATEFILE      # 和“--date”一样，文件的每一行都设置一次
-d, --date=STRING        # 显示字符串代表的时间，注意不是当前时间
-r, --reference=FILE     # 显示文件的最后修改时间
-R, --rfc-2822            # 以rfc-2822的方式输出日期和时间,
--rfc-3339=TIMESPEC       # 以rfc-3339的方式输出日期和时间，精度可以是date、seconds、ns，日期和时间
-s, --set=STRING          # 用指定字符串设置时间
-u, --utc, --universal    # 输出或者设置通用时间

--help                    # 显示帮助文档
--version                 # 显示命令版本信息
```

sh

format可用的转义序列如下：

%%	百分号
%a	当地缩写的工作日名称（例如，Sun）
%A	当地完整的工作日名称（例如，Sunday）
%b	当地缩写的月份名称（例如，Jan）
%B	当地完整的月份名称（例如，January）
%c	当地的日期和时间（例如，Thu Mar 3 23:05:25 2005）
%C	世纪，和%Y类似，但是省略后两位（例如，20）
%d	一月中的一天（例如，01）
%D	日期，等价于%m/%d/%y
%e	一月中的一天，格式使用空格填充，等价于%d
%F	完整的日期；等价于%+4Y-%m-%d
%g	ISO标准计数周的年份的最后两位数字
%G	ISO标准计数周的年份，通常只对%V有用
%h	等价于%b
%H	小时，范围(00..23)
%I	小时，范围(00..23)
%j	一年中的一天，范围(001..366)
%k	小时，使用空格填充，范围(0..23)，等价于%_H
%l	小时，使用空格填充，范围(1..12)，等价于%_I
%m	月，范围(01..12)
%M	分钟，范围(00..59)
%n	换行符
%N	纳秒，范围(000000000..000000000)
%p	用于表示当地的AM或PM，如果未知则为空白
%P	类似于%p，但用小写表示
%q	季度，范围(1..4)
%r	当地以12小时表示的时钟时间（例如，11:11:04 PM）
%R	24小时每分钟；等价于%H:%M
%s	自协调世界时1970年01月01日00时00分以来的秒数
%S	秒数，范围(00..60)
%t	水平制表符
%T	时间；等价于%H:%M:%S
%u	一周中的一天(1..7)，1代表星期一
%U	一年中的第几周，周日作为一周的起始(00..53)
%V	ISO标准计数周，该方法将周一作为一周的起始(01..53)
%w	一周中的一天(0..6)，0代表星期天
%W	一年中的第几周，周一作为一周的起始(00..53)
%x	当地的日期表示（例如，12/31/99）
%X	当地的时间表示（例如，23:13:48）
%y	年份后两位数字，范围(00..99)
%Y	年份
%z	+hhmm格式的数值化时区格式（例如，-0400）
%:z	+hh:mm格式的数值化时区格式（例如，-04:00）
%::z	+hh:mm:ss格式的数值化时区格式（例如，-04:00:00）
%:::z	数值化时区格式，相比上一个格式增加':'以显示必要的精度（例如，-04，+05:30）
%Z	时区缩写（如EDT）

说明

默认情况下，使用数字0来填补数字时间中的空缺。当然也可以指定其他方式：“%–”，不填补；“%_”，使用空格填补；“%0”，使用0；“%^”使用大写字母；“%#”，使用相反的字母

“--date=String”是一种自由格式，是一种方便读取的日期字符串，例如“Sun, 2月29日16: 21: 42-0800”或“2004-02-29 16: 21: 42: 42”，甚至是“下星期四”。日期字符串可能包含指示日历日期、时间、时区、周中日、相对时间、相对日期和数字的项。空字符串表示一天的开始。日期字符串可以包含指示日

历日期、时间、时区、星期数、相对时间、相对日期和数字的项。日期字符串格式比这里容易记录的要复杂，但是在info文档中有完整的描述。

举例

不适用任何参数，直接显示日期和时间

```
[sogrey@bogon ~]$ date  
2021年 06月 23日 星期三 23:24:34 CST # 注意，这里显示的是CST时间  
[sogrey@bogon ~]$
```

显示UTC时间

```
[sogrey@bogon ~]$ date -u  
2021年 06月 23日 星期三 15:24:57 UTC  
[sogrey@bogon ~]$
```

设置CST日期时间

```
[sogrey@bogon ~]$ sudo date -s 2018-9-4 # 这里可以看到支持的日期格式。如果没有设置时间，那么默认就是0点  
2018年 09月 04日 星期二 00:00:00 CST  
[sogrey@bogon ~]$ sudo date -s 20180905  
2018年 09月 05日 星期三 00:00:00 CST  
[sogrey@bogon ~]$ date -s 2018/9/6  
date: 无法设置日期：不允许的操作  
2018年 09月 06日 星期四 00:00:00 CST  
[sogrey@bogon ~]$ date -s 11:56 # 修改时间  
date: 无法设置日期：不允许的操作  
2021年 06月 23日 星期三 11:56:00 CST  
[sogrey@bogon ~]$ date -s 11:54:40  
date: 无法设置日期：不允许的操作  
2021年 06月 23日 星期三 11:54:40 CST  
[sogrey@bogon ~]$
```

显示当前是一年中的第几周，第几天

```
[sogrey@bogon ~]$ date +第%U周第%j天  
第25周第174天  
[sogrey@bogon ~]$
```

显示12小时制度下的时间

```
[sogrey@bogon ~]$ date +%r  
下午 11时28分49秒  
[sogrey@bogon ~]$ date +%p%H:%M:%S  
下午23:28:57  
[sogrey@bogon ~]$
```

显示当前日期

sh

```
[sogrey@bogon ~]$ date +%x  
2021年06月23日  
[sogrey@bogon ~]$ date +%F  
2021-06-23  
[sogrey@bogon ~]$ date +%Y-%m-%d # 多格式拼接  
2021-06-23  
[sogrey@bogon ~]$
```

declare - 声明变量，设置或显示变量的值和属性

declare用来定义shell变量或者给变量增加属性，如果没有指定变量，那么就显示所有的变量。declare定义的变量仅能在当前的shell环境中使用，退出shell之后无效。

主要用途

- 显示包含指定属性的全部变量和值
- 显示包含指定属性的一到多个变量和值
- 显示一到多个变量的属性和值
- 显示所有变量的属性和值并显示函数的定义
- 显示所有变量的属性和值
- 显示所有全局变量的属性和值
- 显示全部函数名和函数定义
- 只显示全部函数名
- 显示一到多个函数名和函数定义
- 只显示一到多个函数名
- 声明全局变量（可选：赋值）
- 声明变量（可选：赋值、属性）
- 增加、删除变量的属性（可选：赋值）

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
declare [-aAfFilrtux] [-p] [name[=value] ...]
```

sh

选项

```
-a      # 声明数组变量  
-f      # 仅显示函数  
-F      # 不显示函数定义  
-i      # 先计算表达式，把结果赋值给变量  
-p      # 仅显示变量定义的方法和值。使用此选项时，其他选项被忽略  
-r      # 定义只读变量  
-x      # 将指定的变量转换成环境变量  
-l      # 当变量分配值的时候，所有的大写字母变成小写  
-u      # 当变量分配值的时候，所有的小写字母变成大写
```

sh

使用“-”是赋值属性，使用“+”可以关闭属性。例如使用“-x”可以将变量改为环境变量，而使用“+x”可以将其变为普通变量

举例

定义环境变量

```
[root@localhost ~]$ declare -x var_1=100      #定义环境变量  
[root@localhost ~]$ declare -x | tail -n 1    #查看环境变量  
declare -x var_1="100"  
  
[root@localhost ~]$ declare +x var_1          #去除var_1的环境变量属性  
You have new mail in /var/spool/mail/root  
[root@localhost ~]$ declare -x | tail -n 1    #查看环境变量, 已经成功去除  
declare -x XMODIFIERS="@im=ibus"
```

sh

使用"-u"属性定义变量

```
[root@localhost ~]$ declare -u var_1="hello"  #定义变量并赋值  
[root@localhost ~]$ declare | grep var_1      #查看变量, 赋值时的小写字母都变成大写  
_=var_1=hello  
var_1=HELLO  
[root@localhost ~]$
```

sh

使用"-i"属性定义变量

```
[root@localhost ~]$ declare -i var_2=100+200  #定义变量var_2, 结果是300  
[root@localhost ~]$ declare | grep var_2      #查看变量  
_=var_2=100+200  
var_2=300
```

sh

depmod - 分析可载入模块的相依性

depmod命令 可产生模块依赖的映射文件，在构建嵌入式系统时，需要由这个命令来生成相应的文件，由modprobe使用。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
depmod [OPTION]
```

sh

选项

-a, --all	# 分析所有可用的模块;
-d, debug	# 执行排错模式;
-e	# 输出无法参照的符号;
-i	# 不检查符号表的版本;
-m<文件>, system-map<文件>	# 使用指定的符号表文件;
-s, --system-log	# 在系统记录中记录错误;
-v, --verbose	# 执行时显示详细的信息;
-V, --version	# 显示版本信息;
--help	# 显示帮助。

sh

举例

无

df - 显示磁盘的相关信息

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。

显示磁盘分区上的磁盘使用状况，可以显示出文件系统名称、大小、挂载点等信息。df显示包含每个文件名参数的文件系统上可用的磁盘空间。如果不给出文件名，则显示所有当前挂载的文件系统上可用的空间。默认情况下，磁盘空间显示在1K的块中，除非设置了环境变量POSIXLY_RIDER，在这种情况下使用512个字节块。

如果参数是包含已挂载文件系统的磁盘设备节点的绝对文件名，df将显示该文件系统上的可用空间，而不是包含设备节点的文件系统(始终是根文件系统)。这样做的各种系统都需要非常不可移植的文件系统结构知识。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
df [OPTION]... [FILE]...
```

sh

选项

```
-a, --all          # 显示所有的文件系统，包含虚拟文件系统
[file]            # 显示指定文件所在的文件系统信息
-B, --block-size=SIZE # 设置显示时的块大小
--direct          # 显示文件的统计信息，而不是挂载点
--total           # 产生一个总和
-h, --human-readable # 以更加易读的方式来显示
-H, --si           # 以更加易读的方式显示，但是使用1000为一个单位，而不是1024
-i, --inodes        # 显示inode信息
-k                 # 等价于"--block-size=1k"
-l, --local         # 显示本地文件系统
--no-sync          # 在获取使用信息之前，不唤醒同步
--sync             # 在获取信息之前唤醒同步
-P, --portability   # 使用POSIX输出格式
-t, --type=TYPE      # 显示指定类型的文件系统信息
-T, --print-type     # 显示文件系统类型
-x, --exclude-type=TYPE # 不显示指定的文件系统

--help              # 显示帮助文档
--version           # 显示命令版本信息
```

sh

该指令显示的值的单位是"--block-size"、"DF_BLOCK_SIZE"、"BLOCK_SIZE"，、BLOCKSIZE"这是四个值中第一个可用的值。除了第一个值是用户设定的，其他3个都是环境变量。如果这四个值没有可用的，那么

默认是1024（如果设置了POSIXLY_CORRECT，那么就是512）。

显示的单位可能是：KB，1000；K，1024；MB，100 100；M，1024 1024。

举例

使用“-h”选项，以方便阅读的方式显示文件系统信息

```
[sogrey@bogon /]$ df -h
文件系统          容量  已用  可用  已用% 挂载点
/dev/mapper/euleros-root  49G  4.9G   42G   11% /
devtmpfs           2.1G    0  2.1G    0% /dev
tmpfs             2.1G    0  2.1G    0% /dev/shm
tmpfs             2.1G  18M  2.1G    1% /run
tmpfs             2.1G    0  2.1G    0% /sys/fs/cgroup
/dev/sda1          976M 151M  759M   17% /boot
/dev/mapper/euleros-home 362G 197M  343G    1% /home
tmpfs            428M  28K  428M    1% /run/user/1000
/dev/sr0            59M   59M    0  100% /run/media/sogrey/VBox_GAs_6.1.22
[sogrey@bogon /]$
```

使用“-T”选项，显示出文件系统的类型

```
[sogrey@bogon /]$ df -T
文件系统      类型      1K-块  已用      可用  已用% 挂载点
/dev/mapper/euleros-root ext4  51343840 5033576 43672440  11% /
devtmpfs       devtmpfs  2172124    0  2172124    0% /dev
tmpfs          tmpfs    2190360    0  2190360    0% /dev/shm
tmpfs          tmpfs    2190360  17840  2172520    1% /run
tmpfs          tmpfs    2190360    0  2190360    0% /sys/fs/cgroup
/dev/sda1        ext4    999320 154068  776440  17% /boot
/dev/mapper/euleros-home ext4  379160352 201128 359629176    1% /home
tmpfs          tmpfs    438076    28  438048    1% /run/user/1000
/dev/sr0          iso9660  59590   59590    0  100% /run/media/sogrey/VBox_GAs_6.1.22
[sogrey@bogon /]$
```

使用“-B”选项，指定显示时的块大小是2048kb。注意结果可以和上面对比一下

```
[sogrey@bogon /]$ df -B 2048
文件系统      2K-块  已用      可用  已用% 挂载点
/dev/mapper/euleros-root 25671920 2516868 21836140  11% /
devtmpfs       1086062    0  1086062    0% /dev
tmpfs          1095180    0  1095180    0% /dev/shm
tmpfs          1095180  8920  1086260    1% /run
tmpfs          1095180    0  1095180    0% /sys/fs/cgroup
/dev/sda1        499660  77034  388220  17% /boot
/dev/mapper/euleros-home 189580176 100564 179814588    1% /home
tmpfs          219038    14  219024    1% /run/user/1000
/dev/sr0          29795   29795    0  100% /run/media/sogrey/VBox_GAs_6.1.22
[sogrey@bogon /]$
```

使用“-t”选项，指定显示“ext4”类型的文件系统信息

sh

```
[sogrey@bogon /]$ df -t ext4
文件系统          1K-块    已用      可用  已用% 挂载点
/dev/mapper/euleros-root  51343840 5033872  43672144  11% /
/dev/sda1           999320 154068   776440  17% /boot
/dev/mapper/euleros-home 379160352 201128 359629176  1% /home
[sogrey@bogon /]$
```

diff - 比较给定的两个文件的不同

逐行比较两个文本文件，把文件的差异显示到标准输出。如果要指定要比较目录，那么diff命令会比较目录中相同文件名的文件，不会比较子目录。

diff命令 在最简单的情况下，比较给定的两个文件的不同。如果使用“-”代替“文件”参数，则要比较的内容将来自标准输入。diff命令是以逐行的方式，比较文本文件的异同处。如果该命令指定进行目录的比较，则将会比较该目录中具有相同文件名的文件，而不会对其子目录文件进行任何比较操作。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
diff [选项] files
```

sh

选项

```

-m, --mode=MODE          # 设置目录的权限
-i, --ignore-case        # 比较的时候，忽略大小写
--ignore-file-name-case # 在比较文件名的时候，忽略大小写
--no-ignore-file-name-case # 比较文件名的时候，不能忽略大小写
-E, --ignore-tab-expansion # 不比较tab
-b, --ignore-space-change # 不比较空格数
-w, --ignore-all-space   # 忽略所有的空格
-B, --ignore-blank-lines # 不比较空白行
-I res, --ignore-matching-lines=res # 不比较含有指定字符串res的行
--strip-trailing-cr      # 去除输入行尾随的东西
-a, --text                # 将所有的文件都当做文本文件
-c -C NUM --context[=NUM] # 显示不同之处的前后部分内容，默认是3行
-u -U NUM --unified[=NUM] # 显示相同之处的前后部分内容，默认是3行
--label LABEL             # 使用文件的标签，而不是名字
-p, --show-c-function     # 比较c语言文件的时候，显示不同之处所在的函数
-F RE, --show-function-line=RE # 显示匹配RE的最近的行
-q, --brief               # 只显示是否有差异，不显示详细内容
-e, --ed                  # 输出一个ed脚本
--normal                 # 输出一个正常的diff
-n, --rcs                 # 结果以rcs的方式显示
-y, --side-by-side        # 将两个文件已并列方式显示比较结果
-W num, --width=NUM       # 使用“-y”选项的时候，指定列宽
--left-column             # 只输出公共行的左列
--suppress-common-lines   # 不要输出公共行
-D NAME, --ifdef=NAME     # 输出合并文件以显示‘#ifdef NAME’的差异
--GTYPE-group-format=GFMT # 同上，但用GFMT格式化GTYPE输入组
--line-format=LFMT         # 同上，但用LFMT格式化GTYPE输入组
--LTYPE-line-format=LFMT   # 同上，但用LFMT格式化LTYPE输入行
-l, --paginate            # 将输出传递给pr”以分页
-t, --expand-tabs          # 将制表符展开为输出中的空格
-T, --initial-tab          # 通过预置选项卡使制表符对齐
-N, --new-file              # 将缺席文件视为空文件
--unidirectional-new-file # 将缺席的第一批文件视为空文件
-s, --report-identical-files # 当两个文件相同时报告
-X FILE, --exclude-from=FILE # 排除与文件中任何模式匹配的文件
-S FILE, --starting-file=FILE # 从文件开始比较目录时
-r, --recursive             # 用递归的方式比较子目录下的所有文件
-x path                    # 不比较指定的文件
--from-file=FILE1           # 将FILE 1与所有操作数进行比较。FILE 1可以是一个目录
--to-file=FILE2              # 将所有操作数与文件2进行比较。文件2可以是一个目录
--horizontal-lines=NUM       # 保持通用前缀和后缀的NUM行
-d, --minimal                # 努力找出一组较小的变更
--speed-large-files          # 假设文件很大，并且有许多零散的小更改。

--help                      # 显示帮助文档
--version                   # 显示命令版本信息

```

举例

```
[sogrey@bogon demos]$ diff test.txt test2.txt
1,2d0
< 石家庄今日新增16例确诊病例
< 中国留美博士遇害 美驻华使馆慰问
4,6c2,3
< 理塘文旅公司回应丁真抽烟
< 北京一确诊者隐瞒行程不配合流调
< 山西晋中新增2例无症状感染者
---
>
> 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。
[sogrey@bogon demos]$ diff -y test.txt test2.txt
石家庄今日新增16例确诊病例 <
中国留美博士遇害 美驻华使馆慰问 <
特朗普夫人发文谴责国会暴乱 特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟 | <
北京一确诊者隐瞒行程不配合流调 | 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上
山西晋中新增2例无症状感染者 <
[sogrey@bogon demos]$
[sogrey@bogon 文档]$ diff backup demos
只在 demos 存在: test.c
[sogrey@bogon 文档]$ diff -q backup/test.txt demos/test.txt
文件 backup/test.txt 和 demos/test.txt 不同
[sogrey@bogon 文档]$
```

diff3 - 比较3个文件不同的地方

diff3命令 用于比较3个文件，将3个文件的不同的地方显示到标准输出。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
diff3 [选项] MYFILE OLDFILE YOURFILE
```

sh

选项

```
-e, --ed          # 输出从OLDFILE到YOURFILE到MYFILE的未合并更改
-E, --show-overlap # 输出未合并的更改，将冲突括起来
-A, --show-all    # 输出所有更改，将冲突括起来
-x, --overlap-only # 输出重叠变化
-X,              # 输出重叠变化，将冲突括起来
-3, --easy-only   # 输出未合并的不重叠更改
-m, --merge        # 输出合并文件而不是ed脚本(默认-A)。
-L LABEL, --label=LABEL # 使用文件LABEL
-i                # 将‘w’和‘q’命令附加到ed脚本中
-a, --text         # 将所有文件当做文本
-T --initial-tab   # 通过预置选项卡使制表符对齐
--diff-program=PROGRAM # 使用PROGRAM来比较文件

--help            # 显示帮助文档
--version         # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon backup]$ diff3 test.txt test2.txt test3.txt
```

```
====
```

```
1:1,7c
```

石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟

```
123
```

北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者

```
2:1,3c
```

特朗普夫人发文谴责国会暴乱

1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。

```
3:1,9c
```

石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
特朗普夫人发文谴责国会暴乱

1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。

```
[sogrey@bogon backup]$
```

diffstat - 显示diff命令输出信息的柱状图

读取diff的输出，并显示每个文件的插入、删除和修改的直方图。Diffstat是一个用于检查大型复杂修补程序文件的程序。它从包含diff输出的一个或多个输入文件中读取，生成针对引用的每个文件更改的总行的直方图。如果输入文件名以.bz 2、.gz、.lzma、.z或.z结尾，Diffstat将通过管道从相应的程序读取未压缩数据。它还可以根据标准输入的管道文件推断压缩类型。

Diffstat识别来自diff的最流行的输出类型：

- unified，修补程序首选。
- context，最好的可读性，但不太紧凑。
- default，不是很好，但很容易产生。

Diffstat检测由diff输出的行，以判断比较了哪些文件，然后在第一列中计数表示更改类型(插入、删除或修改)的标记。这些在直方图中显示为“+”、“-”和“!”字符。如果命令行上没有指定文件名，Diffstat将读取标准输入中的差异。

diffstat命令 用来显示diff命令输出信息的柱状图，用以显示diff命令比较两个文件的不同统计信息。用户也可以直接使用|将diff命令所输出的结果直接送给diffstat命令进行统计结果的显示。使用该命令时，若所比较的文件或者子目录不在当前目录下，则应该使用其完整路径。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
diffstat [options] [file-specifications]
```

sh

选项

```

-b          # 忽略diff中的“二进制文件XXX和YYY不同”匹配的行
-c          # 在每一行输出前加上“#”，使其成为shell脚本的注释行。
-D destination # 指定一个目录，其中包含可作为应用差异的结果而引用的文件。
              # Diffstat将计算相应文件中的行数(在通过-p选项调整名称之后),
              # 以获得每个文件中的总行数。
-e file     # 将标准错误重定向到文件
-f format   # 指定直方图的格式:
              #   0, 为了简洁, 它只显示值和一个直方图代码insert (+),
              #       delete (-)或modify (!)。
              #   1, 正常格式。
              #   2, 用点填充直方图。
              #   4, 使用直方图打印每个值。
              #   任何非零值都会给出直方图。
              #   点和个别值可以结合在一起, 例如-f6给出两者。
-h          # 显示帮助信息并且退出
-k          # 禁止合并报表中的文件名
-l          # 只列出文件名。不生成直方图。
-m          # 从修补程序文件的每个“块”中合并插入/删除计数,
              # 以接近修改行的计数
-n number   # 指定用于文件名的最小宽度。如果不指定这一点,
              # Diffstat在去掉常见前缀后使用最长文件名的长度。
-N number   # 指定用于文件名的最大宽度。超过此限制的名称在左侧被截断。
              # 如果您没有指定此选项, 下面将检查-n选项。
-o file     # 将标准输出重定向到文件
-p number   # 重写删除公共路径名的逻辑, 模拟修补程序“-p”选项。
-q          # 抑制空差异的“0文件更改”消息
-r code     # 提供直方图中显示的数据的可选舍入, 而不是通过错误调整截断数据:
              #   0, 是默认的。不执行舍入操作, 但累积错误将添加到下列中。
              #   1, 舍入数据。
              #   2, 对数据进行舍入并调整直方图, 以确保在有任何差异的情况下显示
              #       某些内容, 即使这些差异通常被四舍五入为零。
-S source   # 这类似于-D选项, 但指定了一个可以找到原始文件(在应用差异之前)的位置。
-t          # 重写直方图, 生成逗号分隔值的输出。
-u          # 禁止对报表中的文件名进行排序。
-v          # 显示进度, 例如, 如果输出重定向到文件, 则将进度消息写入标准错误
-V          # 显示版本号, 并且退出
-w number   # 指定直方图的最大宽度。直方图将永远不会小于10列, 以防文件名过大。

```

环境变量

Diffstat运行在可移植的UNIX环境中。您可以通过设置与输入文件名称相对应的环境变量来覆盖用于解压缩输入文件的程序的编译路径。然而, Diffstat假设生成的程序使用相同的命令行选项, 例如“-c”来解压缩到标准输出。

- DIFFSTAT_BZCAT_PATH
- DIFFSTAT_BZIP2_PATH
- DIFFSTAT_COMPRESS_PATH
- DIFFSTAT_GZIP_PATH
- DIFFSTAT_LZCAT_PATH
- DIFFSTAT_PCAT_PATH
- DIFFSTAT_UNCOMPRESS_PATH
- DIFFSTAT_ZCAT_PATH

举例

```
[sogrey@bogon demos]$ diff test.txt test2.txt
1,2d0
< 石家庄今日新增16例确诊病例
< 中国留美博士遇害 美驻华使馆慰问
4,6c2,3
< 理塘文旅公司回应丁真抽烟
< 北京一确诊者隐瞒行程不配合流调
< 山西晋中新增2例无症状感染者
---
>
> 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。
[sogrey@bogon demos]$ diff test.txt test2.txt | diffstat -f 2
unknown |    7 +-----.
1 file changed, 2 insertions(+), 5 deletions(-)
[sogrey@bogon demos]$ diff test.txt test2.txt | diffstat -f 1
unknown |    7 +-----.
1 file changed, 2 insertions(+), 5 deletions(-)
[sogrey@bogon demos]$ diff test.txt test2.txt | diffstat -f 0
unknown |    7 2 + 5 - 0 !
1 file changed, 2 insertions(+), 5 deletions(-)
[sogrey@bogon demos]$
```

dig - 域名查询工具

dig命令是常用的域名查询工具，可以用来测试域名系统工作是否正常。

dig是一个DNS查询工具，多数管理员会使用dig命令来解决DNS的问题。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
dig [选项]
```

sh

选项

```
@server      # 指定服务器地址  
-b host      # 指定通过哪个主机查询  
-f file      # 从指定文件来查询  
-p port      # 指定使用的端口  
-t type      # 指定要查询的DNS类型，例如A\MX\PRT  
-x ip        # 指定DNS你想查询，输入ip得到域名  
-4           # 使用ipv4  
-6           # 使用ipv6
```

sh

举例

sh

```
[sogrey@bogon ~]$ dig

; <>> DiG 9.9.4-RedHat-9.9.4-61.1.h4.eulerosv2r7 <>>
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43944
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;.

;; ANSWER SECTION:
. 98105 IN NS f.root-servers.net.
. 98105 IN NS b.root-servers.net.
. 98105 IN NS g.root-servers.net.
. 98105 IN NS j.root-servers.net.
. 98105 IN NS d.root-servers.net.
. 98105 IN NS e.root-servers.net.
. 98105 IN NS m.root-servers.net.
. 98105 IN NS k.root-servers.net.
. 98105 IN NS a.root-servers.net.
. 98105 IN NS l.root-servers.net.
. 98105 IN NS i.root-servers.net.
. 98105 IN NS h.root-servers.net.
. 98105 IN NS c.root-servers.net.

;; Query time: 18 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: 五 7月 16 00:34:56 CST 2021
;; MSG SIZE rcvd: 228

[sogrey@bogon ~]$
```

查询域名信息

sh

```
[sogrey@bogon ~]$ dig www.baidu.com

; <>> DiG 9.9.4-RedHat-9.9.4-61.1.h4.eulerosv2r7 <>> www.baidu.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3867
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.baidu.com. IN A

;; ANSWER SECTION:
www.baidu.com. 942 IN CNAME www.a.shifen.com.
www.a.shifen.com. 15 IN CNAME www.wshifen.com.
www.wshifen.com. 44 IN A 103.235.46.39

;; Query time: 66 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: 五 7月 16 00:35:32 CST 2021
;; MSG SIZE rcvd: 111

[sogrey@bogon ~]$
```

反向查询

sh

```
[sogrey@bogon ~]$ dig -t a -x 103.235.46.39

; <>> DiG 9.9.4-RedHat-9.9.4-61.1.h4.eulerosv2r7 <>> -t a -x 103.235.46.39
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 15873
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;39.46.235.103.in-addr.arpa. IN A

;; AUTHORITY SECTION:
103.in-addr.arpa. 1799 IN SOA ns.apnic.net. read-txt-record-of-zone-first-dns-admin.apnic.net. 5

;; Query time: 269 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: 五 7月 16 00:36:45 CST 2021
;; MSG SIZE rcvd: 143

[sogrey@bogon ~]$
```

dirs - 显示目录堆栈

bind命令用来显示或者设置当前的键盘设置，可以通过bind了解当前组合按键功能，也可以自己设置。

bind命令 用于显示和设置命令行的键盘序列绑定功能。通过这一命令，可以提高命令行中操作效率。您可以利用bind命令了解有哪些按键组合与其功能，也可以自行指定要用哪些按键组合。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
bind [-m keymap] [-lpsvPSV]
bind [-m keymap] [-q function] [-u function] [-r keyseq]
bind [-m keymap] -f filename
bind [-m keymap] -x keyseq:shell-command
bind [-m keymap] keyseq:function-name
bind readline-command
```

sh

选项

```
-m keymap      # 使用键盘设置，可以使用的按键: emacs, emacs-standard, emacs-meta, emacs-ctlx, vi,
-1             # 显示所有的readline函数
-p             # 以可以重读的方式列出readline函数名和绑定情况
-P             # 列出当前readline函数和绑定情况
-s             # 列出可被重复读取的宏和字符串
-S             # 列出正在使用的宏和字符串
-v             # 以重读的方式列出readline变量名和数值
-V             # 列出正在使用的readline变量名和数值
-f             # 从文件中读取按键配置
-q             # 查询函数的唤醒按键
-u             # 接触函数的所有按键绑定
```

sh

举例

显示当前所有的readline函数

sh

```
[root@localhost ~]$ bind -l      # 显示当前所有的readline函数
abort
accept-line
alias-expand-line
arrow-key-prefix
backward-byte
backward-char
backward-delete-char
backward-kill-line
backward-kill-word
backward-word
beginning-of-history
...
...
```

查看清屏函数的组合键

sh

```
[root@localhost ~]$ bind -q clear-screen    # 查看清屏函数的组合键
clear-screen can be invoked via "\C-l".
```

dmesg - 显示Linux系统启动信息

dmesg命令被用于检查和控制内核的环形缓冲区。kernel会将开机信息存储在ring buffer中。您若是开机时来不及查看信息，可利用dmesg来查看。开机信息保存在/var/log/dmesg文件里。

dmesg指令用来打印和控制内核的输出信息，这些信息保存在ring buffer中。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
dmesg [-c] [-r] [-n level] [-s bufsize]
```

sh

选项

```
-c # 打印之后清除缓冲区  
-r # 打印raw信息  
-s # 设置缓冲区大小，默认16392  
-n # 指定记录信息的等级
```

sh

举例

查看开机内核输出信息

```
[sogrey@bogon ~]$ sudo dmesg -c # 查看信息，之后删除缓冲区内容  
...  
[ 220.943025] [monitor_netdev] Deleting route 127.0.0.0/8, curr comm: kworker/u2:0, curr pid: 6  
[ 220.943032] [monitor_netdev] Deleting route 0.0.0.0/32, curr comm: kworker/u2:0, curr pid: 6  
[ 220.943034] [monitor_netdev] Deleting route 127.255.255.255/32, curr comm: kworker/u2:0, curr pid: 6  
[ 220.943035] [monitor_netdev] Deleting route 127.0.0.0/32, curr comm: kworker/u2:0, curr pid: 6  
[ 224.895993] print_netdev_status: 2 callbacks suppressed  
[ 224.895996] [monitor_netdev] devname : lo, status : down, pid : 685, cmd : kworker/u2:3, ppid : 6  
[ 224.900318] [monitor_netdev] Deleting route 127.0.0.0/8, curr comm: kworker/u2:3, curr pid: 685  
[ 224.900326] [monitor_netdev] Deleting route 0.0.0.0/32, curr comm: kworker/u2:3, curr pid: 685  
[ 224.900328] [monitor_netdev] Deleting route 127.255.255.255/32, curr comm: kworker/u2:3, curr pid: 685  
[ 224.900330] [monitor_netdev] Deleting route 127.0.0.0/32, curr comm: kworker/u2:3, curr pid: 685  
[sogrey@bogon ~]$
```

sh

打印raw信息

sh

```
[sogrey@bogon ~]$ sudo dmesg -r
<4>[ 280.842432] print_netdev_status: 2 callbacks suppressed
<4>[ 280.842435] [monitor_netdev] devname : lo, status : up, pid : 14497, cmd : (ostnamed), ppi
<4>[ 280.842453] [monitor_netdev] Insert table=255 127.0.0.1/32, curr comm: (ostnamed), curr pid
<4>[ 280.842460] [monitor_netdev] Insert table=255 127.0.0.0/8, curr comm: (ostnamed), curr pid
<4>[ 280.842465] [monitor_netdev] Insert table=255 127.0.0.0/32, curr comm: (ostnamed), curr pid
<4>[ 280.842468] [monitor_netdev] Insert table=255 127.255.255.255/32, curr comm: (ostnamed), curr pid
<4>[ 311.036619] print_netdev_status: 5 callbacks suppressed
<4>[ 311.036624] [monitor_netdev] devname : lo, status : down, pid : 6, cmd : kworker/u2:0, ppi
<4>[ 311.041683] [monitor_netdev] Deleting route 127.0.0.0/8, curr comm: kworker/u2:0, curr pid
<4>[ 311.041691] [monitor_netdev] Deleting route 0.0.0.0/32, curr comm: kworker/u2:0, curr pid
<4>[ 311.041693] [monitor_netdev] Deleting route 127.255.255.255/32, curr comm: kworker/u2:0, curr pid
<4>[ 311.041694] [monitor_netdev] Deleting route 127.0.0.0/32, curr comm: kworker/u2:0, curr pid
[sogrey@bogon ~]$
```

dnsdomainname - 定义DNS系统中FQDN名称的域名

dnsdomainname命令 用于定义DNS系统中FQDN名称中的域名。

dnsdomainname将打印FQDN(完全限定域名)的域部分。系统的完整FQDN以“hostname --fqdn”返回。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
dnsdomainname [-v]
```

sh

选项

-v	# 显示详细执行过程
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

显示主机域名

```
[root@localhost ~]$ dnsdomainname -v    #-v选项显示详细执行过程
gethostname()=`localhost.localdomain'
Resolving `localhost.localdomain' ...
Result: h_name=`localhost'
Result: h_aliases=`localhost.localdomain'
Result: h_aliases=`localhost4'
Result: h_aliases=`localhost4.localdomain4'
Result: h_aliases=`localhost.localdomain'
Result: h_aliases=`localhost6'
Result: h_aliases=`localhost6.localdomain6'
Result: h_addr_list=`127.0.0.1'
Result: h_addr_list=`127.0.0.1'
```

sh

domainname - 显示和设置系统的NIS域名

domainname命令 用于显示和设置系统的NIS域名。

domainname指令显示由函数“getdomainname”返回的主机域名，使用这个指令也可以设置一个主机域名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
domainname [-v] [-F filename] [name]
```

sh

选项

-v	# 显示详细执行过程
-F, --file filename	# 从指定文件读取主机域名
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

显示主机域名

```
[root@localhost ~]$ domainname -v
getdomainname()='(none)'
(none)
[root@localhost ~]$ domainname www.sogrey.top
You have new mail in /var/spool/mail/root
[root@localhost ~]$ domainname
www.sogrey.top
```

sh

du - 显示每个文件和目录的磁盘使用空间

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。

以块为单位，显示当前目录下，所有目录、文件、子目录的磁盘使用情况。总结每个文件的磁盘使用情况，对目录进行递归处理

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
du [选项] [目录]
```

sh

选项

```
-a, --all          # 显示所有的文件大小，包含目录、文件、子目录。默认情况下不显示文件的大小
--apparent-size   # 打印表观大小，而不是磁盘使用量；虽然表观大小通常较小，但由于(“稀疏”)文件中的漏
-B, --block-size=SIZE # 设置显示时的块大小
-b, --bytes        # 等价于“--apparent-size --block-size=1”
-c, --total         # 产生一个总和统计
-D, --dereference-args, -H # 只引用命令行中列出的符号链接
--files0-from=F     # 总结文件F中指定的以NUL结尾的文件名的磁盘使用情况；如果F是“-”，则从标准输入中
-h, --human-readable # 以更加易读的方式来显示
--si                  # 和“-h”一样，只是显示单位是1000，而不是1024
-k                   # 相当于-block-size=1k
-m                   # 相当于-block-size=1M
-l, --count-links    # 如果是硬链接，那么记录次数
-L, --dereference     # 取消引用所有符号链接
-P, --no-dereference  # 不要跟随任何符号链接，这是默认的
-0, --null            # 以0字节(而不是换行符)结束每一行输出
-S, --separate-dirs    # 不包括子目录的大小
-s, --summarize       # 只显示每个参数的总数
-x, --one-file-system  # 跳过不同文件系统上的目录
-X, --exclude-from=FILE # 排除与文件中任何模式匹配的文件
--exclude=PATTERN      # 排除匹配模式的文件
--max-depth=N           # 只有目录层数少于N，才打印目录(或文件)的总数(或带有-all)；“--max-depth=0 ‘
--time                 # 显示最后修改的时间
--time=WORD              # 将时间显示为指定的内容而不是修改时间，可以是： atime、access、use、ctime或
--time-style=STYLE        # 使用指定的格式显示时间，时间格式可以是full-iso, long-iso, iso, +FORMAT

--help                # 显示帮助文档
--version             # 显示命令版本信息
```

该指令显示的值的单位是“--block-size”、“DF_BLOCK_SIZE”、“BLOCK_SIZE”，、BLOCKSIZE”这是四个值中

第一个可用的值。除了第一个值是用户设定的，其他3个都是环境变量。如果这四个值没有可用的，那么默认是1024（如果设置了POSIXLY_CORRECT，那么就是512）。

显示的单位可能是：KB，1000；K，1024；MB，100 100；M，1024 1024。

举例

查看home目录总大小

```
[sogrey@bogon ~]$ sudo du /home -s  
131412 /home  
[sogrey@bogon ~]$
```

sh

使用"--time"选项，显示最后的修改时间

```
[sogrey@bogon ~]$ sudo du /home --time  
16 2020-12-17 01:44 /home/lost+found  
4 2020-12-17 01:56 /home/sogrey/模板  
4 2020-12-17 01:56 /home/sogrey/公共  
4 2021-01-12 00:16 /home/sogrey/桌面  
4 2021-03-28 23:43 /home/sogrey/DirTest/Dir1  
8 2021-03-28 23:54 /home/sogrey/DirTest/Dir2  
16 2021-03-28 23:54 /home/sogrey/DirTest  
4 2020-12-17 01:56 /home/sogrey/图片  
48 2021-06-10 23:10 /home/sogrey/文档  
4 2020-12-17 01:56 /home/sogrey/视频  
70196 2021-01-12 00:02 /home/sogrey/下载  
4 2020-12-17 01:56 /home/sogrey/音乐  
131336 2021-06-16 22:56 /home/sogrey  
28 2021-03-21 23:03 /home/other  
28 2021-03-21 23:00 /home/userTmp  
131412 2021-06-16 22:56 /home  
[sogrey@bogon ~]$
```

sh

使用"-m"选项，以Mb为单位显示

sh

```
[sogrey@bogon ~]$ sudo du /home -m
1 /home/lost+found
1 /home/sogrey/模板
1 /home/sogrey/公共
1 /home/sogrey/桌面
1 /home/sogrey/DirTest/Dir1
1 /home/sogrey/DirTest/Dir2
1 /home/sogrey/DirTest
1 /home/sogrey/图片
1 /home/sogrey/文档
1 /home/sogrey/视频
69 /home/sogrey/下载
1 /home/sogrey/音乐
129 /home/sogrey
1 /home/other
1 /home/userTmp
129 /home
[sogrey@bogon ~]$
```

指定"--time"的显示

sh

```
[sogrey@bogon DirTest]$ ll
总用量 8
drwx----- 2 sogrey sogrey 4096 3月 28 23:43 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:54 Dir2
[sogrey@bogon DirTest]$ du ./Dir1/ --time=use
4 2021-06-16 22:48 ./Dir1/
[sogrey@bogon DirTest]$ du ./Dir1/ --time=status
4 2021-03-28 23:54 ./Dir1/
[sogrey@bogon DirTest]$
```

dump - 用于备份ext2或者ext3文件系统

dump命令 用于备份ext2或者ext3文件系统。可将目录或整个文件系统备份至指定的设备，或备份成一个大文件。

检查ext2/3/4文件系统，确定哪些文件需要备份，这些需要备份的文件将会被复制到指定的磁盘或者其他存储介质。dump检查Ext 2/3/4文件系统上的文件，并确定哪些文件需要备份。这些文件被复制到给定的磁盘、磁带或其他存储介质中以确保安全保存(请参阅下面的-f选项以进行远程备份)。大于输出介质的转储被分解为多个卷。在大多数媒体上，大小是通过写入来确定的，直到返回媒体结束指示为止。

在无法可靠地返回媒体结束指示(例如一些盒式磁带驱动器)的媒体上，每个卷都是固定大小的；实际大小是通过指定墨盒介质或通过下面的磁带大小、密度和/或块计数选项来确定的。默认情况下，在提示操作员更改媒体后，每个卷都使用相同的输出文件名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
dump [选项] [-f 目标文件] 源文件
dump [-level#] [-ackMnqSuv] [-A file] [-B records]
      [-b blocksize] [-d density] [-D file] [-e inode numbers]
      [-E file] [-f file] [-F script] [-h level] [-I nr errors]
      [-jcompression level] [-L label] [-Q file] [-s feet] [-T date]
      [-y] [-zcompression level] files-to-dump
dump [-W | -w]
```

sh

files-to-dump要么是文件系统的挂载点，要么是要作为文件系统子集备份的文件和目录列表。在前一种情况下，可以使用安装文件系统的路径或卸载文件系统的设备。在后一种情况下，对备份设置了某些限制：-u是不允许的，唯一支持的转储级别是0，所有文件和目录都必须驻留在同一个文件系统中。

选项

```
sh
-l[evel]#          # 备份级别。如果是0，那么就备份整个文件系统；如果大于0，那么就会备份相对于上一个
-a                 # “自动尺寸”。绕过所有的磁带长度计算，并写入，直到媒体结束指示返回。这是最适合大
-A archive_file   # 存档指定文件中的“内容转储表”，由RESTORE(8)使用，以确定文件是否在正在恢复的转
-b blocksize       # 每个转储记录的千字节数。默认的块大小为10，除非-d选项已用于指定6250 BPI或更高的
-B records        # 每卷1 kB块的数目。通常不需要，因为转储可以检测到媒体的结束。达到指定大小时，du
-c                 # 更改默认使用的磁带驱动器，密度为8000 BPI，长度为1700英尺。指定墨盒驱动器将覆
-d density         # 设置磁带密度。默认为1600 BPI。指定磁带密度将覆盖媒体端检测。
-D file            # 设置文件的路径名，该文件存储有关前一个完整转储和增量转储的信息。默认位置是/etc/dump
-e inodes          # 不包含指定的inodes。inodes参数是一个逗号分隔的inode编号列表
-E file            # 从文本文件中读取备份时要排除的inode列表。该文件应该是一个普通文件，其中包含由
-f filename        # 将备份写入文件；文件可能是一个特殊的设备文件，如/dev/st0(atAPE驱动器)、/dev/
-F script          # 在每个磁带的末尾运行脚本(最后一个除外)。设备名称和当前卷号将在命令行上传递。
-h level           # 只对给定级别或以上的转储授予用户节点标志UF_NODUMP。默认荣誉级别为1，因此增
-I nr_errors       # 默认情况下，在请求操作符干预之前，转储将忽略文件系统上的前32次读取错误。可以
-jcompression_level # 使用bzlib库压缩要写入磁带上的每个块。只有在将文件或管道转储到文件或管道时，才
-k                 # 使用Kerberos身份验证与远程磁带服务器对话。(只有在编译转储时启用此选项时才可
-L label           # 用户提供的文本字符串标签label被放置到转储头中，在这里，像RESTORE(8)和FILE
-m                 # 如果指定了此标志，dump将优化自上次转储以来已更改但未修改的inode的输出(“已更
-M                 # 如果使用此选项，请注意，许多从档案中解压缩文件的程序(例如tar、rpm、unzip、c
-n                 #
-q                 #
-Q file            # 启用多卷功能。使用“-f”指定的名称被视为前缀，并按顺序写入<prefix>001, <prefix>002,
-s feet            # 每当转储需要操作员注意时，以类似于wall(1)的方式通知组操作符中的所有操作符。
-T date            # 每当需要操作员注意时，立即使dump中止，而无需提示写入错误、磁带更改等。
-u                 #
-v                 #
-w                 # 启用快速文件访问支持。每个inode的磁带位置存储在RESTORE使用的文件中(如果用参
-w                # 建议将st驱动程序设置为在调用带参数的转储/还原之前返回逻辑磁带位置而不是物理磁
-y                 # 此选项可在转储到本地磁带(见上文)或本地文件时使用。
-zcompression level # 试图计算在特定密度下所需的磁带数量。如果超过此数量，则转储提示输入新磁带。建
-s                # 大小估计。确定在没有实际执行转储的情况下执行转储所需的空间数量，并显示它将占
-t                # 指定备份的日期。使用指定的日期作为转储的开始时间，而不是从查看/etc/dumpdate
-u                # 备份完成后，在/etc/dumpdates中记录备份的文件系统、日期。人们可以阅读/etc/dumpda
-v                # 显示详细的信息，帮助调试错误
-w                # 显示出最近的备份时间、层级，检测需要备份的文件。当W选项出现时，其他的选项都被
-w                # 类似于“-W”，但只在“/etc/mtab”和“/etc/fstat”中打印需要转储的可识别的文件系
-y                # 使用lzo库压缩要写入磁带的每个块。这不能像 zlib库那样压缩，但速度要快得多。只
-z                # 使用 zlib库压缩要写入磁带上的每个块。此选项仅在将文件或管道转储到文件或管道时
```

说明

dump需要操作员对以下条件进行干预：磁带结束、转储结束、磁带写入错误、磁带打开错误或磁盘读取错误(如果存在nr错误的阈值以上)。除了通知-n键所暗示的所有操作符之外，在转储不能继续时，或者在发生严重错误时，转储与转储控制终端上的操作符进行交互。所有的问题转储构成必须通过键入“是”或“否”，适当地回答。

因为做一个dump需要花费大量的时间和精力，所以在每个磁带卷的开头都会转储检查点。如果由于某种原因写入该卷失败，则在重新装入和删除旧磁带并安装新磁带之后，在操作员许可下，转储将从检查点重新启动。

dump以周期性的间隔告诉操作符正在发生什么，包括通常对要写入的块数、所需磁带的数量、完成时间和磁带更改时间的低估计。输出是冗长的，因此其他人知道终端控制转储是繁忙的，而且会持续一段时间。

在发生灾难性磁盘事件时，将所有必要的备份磁带或文件恢复到磁盘所需的时间可以通过错位增量转储来保持在最低限度。一种将磁带数量降到最小的增量式转储的有效方法如下：

- 总是从0级备份开始，例如：“/sbin/dump -0u -f /dev/st0 /usr/src”。这应该每隔一段时间进行一次，比如每月一次或每两个月一次，并在一套永久保存下来的新磁带上进行。
- 在级别0之后，每天都会对活动文件系统进行转储，并使用此转储级别序列：“3 2 5 4 7 6 9 8 9 9 ...”。

经过几个月左右的时间，每天和每周的磁带都应该从转储周期中轮换出来，新的磁带应该被带进来。

环境变量

- TAPE。如果没有指定“-f”选项，dump将使用通过TAPE指定的设备作为转储设备。TAPE可以是磁带名字、host:tapename、user@host:tapename。
- RMT。环境变量RMT将用于确定远程rmt(8)程序的路径名。
- RSH。dump使用此变量的内容来确定远程备份时要使用的远程shell命令的名称(rsh、ssh等)。如果未设置此变量，则将使用rcmd(3)，但只有root才能进行远程备份。

文件

- /dev/st0 要备份到的默认磁带单元。
- /etc/dumpdates dump的日期记录。
- /etc/fstab 转储表：文件系统和频率
- /etc/mtab 转储表：挂载的文件系统
- /etc/group 查找组操作符

退出码

成功时dump状态为零。启动错误用退出代码1表示；异常终止用退出代码3表示。

当发生读取错误时，转储输出相应的物理磁盘块和扇区号以及Ext 2/3/4逻辑块号。它不打印相应的文件名，甚至不打印inode编号。用户必须使用调试器(8)，命令nCheck和lcheck将由转储输出的ext2blk数字转换为inode号，然后转换为文件名。

每个卷轴都需要一个新的进程，所以卷轴的父进程已经写的只是挂在一起，直到写了整个磁带。如果压缩为ON，则估计的磁带数不正确。如果dump知道dump序列，就会很好，保持跟踪的磁带，告诉操作员要安装什么磁带，并为运行恢复的操作员提供更多的帮助。

由于转储的安全历史，它无法在不以root用户身份运行的情况下进行远程备份。目前，如果设置setuid(与以前一样)，它就能工作，但这可能会构成安全风险。请注意，可以将rsh设置为使用远程shell程序。

注意

这个版本的dump只能处理Ext 2/3/4文件系统，这可能被认为是一个错误。具体来说，它不适用于FAT文件系统。忽略文件系统上少于32个读取错误(用-i更改此错误)。如果注意读取错误很重要，则可以解析转储的输出，以查找包含文本“读取错误”的行。

举例

将/home目录所有内容备份到/tmp/homeback.bak文件中，备份层级为0并在/etc/dumpdates中记录相关信息：

```
dump -0u -f /tmp/homeback.bak /home
```

sh

将/home目录所有内容备份到/tmp/homeback.bak文件中，备份层级为1（只备份上次使用层次0备份后发生过改变的数据）并在/etc/dumpdates中记录相关信息：

```
dump -1u -f /tmp/homeback.bak /home
```

sh

通过dump命令的备份层级，可实现完整+增量备份、完整+差异备份，在配合crontab可以实现无人值守备份。

dumpe2fs - 显示ext2、ext3、ext4文件系统的超级快和块组信息

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
dumpe2fs [ -bfhixV ] [ -o superblock=superblock ] [ -o blocksize=blocksize ] device
```

sh

选项

```
-b # 显示文件系统中保留的损坏块  
-o superblock=superblock # 检查文件系统时，使用指定大小的超级块。此选项通常不需要，除非文件系统向导正在检查严  
-o blocksize=blocksize # 检查文件系统时，指定块大小。此选项通常不需要，除非文件系统向导正在检查严  
-f # 强制执行  
-h # 只显示超级块信息，而不显示任何块组描述符详细信息  
-i # 显示从e2image获得的文件系统信息，使用设备作为image文件的路径名。  
-x # 以十六进制显示文件系统信息  
-V # 显示命令版本信息，并且退出
```

举例

显示sdb4的组块信息

```
[root@localhost ~]$ dumpe2fs /dev/sdb4
dumpe2fs 1.41.12 (17-May-2010)
Filesystem volume name: <none>
Last mounted on: <not available>
Filesystem UUID: e2a0cb30-f3ca-47de-92b8-780296960d93
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
...
Group 0: (Blocks 1-8192)
主 superblock at 1, Group descriptors at 2-2
保留的GDT块位于 3-65
Block bitmap at 66 (+65), Inode bitmap at 67 (+66)
Inode表位于 68-323 (+67)
7855 free blocks, 2037 free inodes, 2 directories
可用块数: 338-8192
可用inode数: 12-2048
Group 1: (Blocks 8193-16383)
...
```

e2image - 将文件系统元数据保存到由图像文件指定的文件中

e2Image程序将位于设备上的ext2、ext3或ext4文件系统元数据保存到由图像文件指定的文件中。通过对这些程序使用-i选项，image文件可以由dupe2fs和调试器来检查。这可以帮助专家恢复严重损坏的文件系统。

如果image文件是“-”，那么e2image的输出将被发送到标准输出，以便输出可以管道到另一个程序，如gzip(1)。(请注意，目前只有在使用-r选项创建原始image文件时才支持这一点，因为创建普通image文件或QCOW 2映像的过程目前需要对文件进行随机访问，这不能使用管道进行。)

最好为系统上的所有文件系统创建映像文件，并定期保存分区布局(可以使用fdisk-l命令生成)。image文件应该存储在它所包含的数据的文件系统以外的其他文件系统上，以确保在文件系统严重损坏的情况下可以访问这些数据。

为了节省磁盘空间，e2Image将image文件创建为稀疏文件，或以QCOW2格式创建。因此，如果需要将稀疏image文件复制到另一个位置，则应该首先对其进行压缩，或者使用GNU版本的cp “sparse=always”选项。这不适用于QCOW2映像，它并不稀疏。

ext2映像文件的大小主要取决于文件系统的大小和正在使用的inode数量。对于一个典型的10GB文件系统，120万个节点中有20万个节点在使用，image文件将大约为35兆字节；在55万个节点中使用15000个节点的4G文件系统将产生一个3MB的image文件。image文件通常是可压缩的；占用磁盘上32 MB空间的image文件通常会压缩到3或4MB。

适用范围

RedHat openSUSE	RHEL Fedora	Ubuntu Linux Mint	CentOS Alpine Linux	Debian Arch Linux	Deepin	SUSE
--------------------	----------------	----------------------	------------------------	----------------------	--------	------

语法

```
e2image [ -rsI ] device image-file
```

sh

选项

```
-I      # 将文件中的元数据恢复到分区  
-r      # 创建raw格式的image  
-Q      # 创建QCOW2格式的image
```

sh

说明

1. “-I”选项

“-I”选项将导致e2Image将存储在映像文件中的元数据重新安装到设备上。它可用于在紧急情况下将文件系

统元数据还原回设备。只有当其他选择失败时，才应该使用"-l"选项作为一种绝望措施。如果文件系统在创建image文件后发生了更改，数据将丢失。通常，您应该首先对文件系统进行完整的映像备份，以便以后尝试其他恢复策略。

2. "-r"选项

"-r"选项将创建一个原始image文件，而不是普通的image文件。原始image文件与普通image文件有两种不同之处。首先，将文件系统元数据放置在适当的位置，以便e2fsck、dupe2fs、调试器等。可以直接在原始image文件上运行。为了尽量减少原始映像文件占用的磁盘空间，将该文件创建为稀疏文件。(请注意使用不了解如何创建稀疏文件的实用程序复制或压缩/解压缩该文件；该文件将与文件系统本身一样大！)其次，原始image文件还包括标准image文件没有的间接块和目录块，尽管这在将来可能会发生变化。

当将文件系统作为bug报告的一部分发送给维护人员时，有时会使用原始映像文件到e2fsprogs。当以这种方式使用时，建议的命令如下(用适当的设备替换hda1)：“e2image -r /dev/hda1 - | bzip2 > hda1.e2i.bz2”。这将只发送元数据信息，没有任何数据块。但是，目录块中的文件名仍然可以显示关于文件系统内容的信息，bug报告人员可能希望将这些信息保密。要解决此问题，可以指定"-s"选项。这将导致e2Image对目录条目进行置乱，并在写入image文件之前将目录块中任何未使用的部分清零。但是，“-s”选项将防止分析与散列树索引目录相关的问题

请注意，即使您将"/dev/hda 1"替换为另一个原始磁盘映像或先前由e2Image创建的QCOW2映像，这也是可行的。

3. "-Q"选项

"-Q"选项将创建一个QCOW2 image文件，而不是普通的或原始的image文件。QCOW2 image包含原始image所做的所有信息，但是与原始image不同的是，它并不稀疏。QCOW 2映像通过将数据以特殊格式存储，并将数据紧密地打包在一起，从而将磁盘空间的数量降到最低，从而避免了漏洞，同时仍然最小化了大小。

为了将文件系统作为bug报告的一部分发送给维护人员到e2fsprogs，请使用以下命令：“e2image -Q /dev/hda1 hda1.qcow2”、“bzip2 -z hda1.qcow2”。这将只发送元数据信息，没有任何数据块。但是，目录块中的文件名仍然可以显示关于文件系统内容的信息，bug报告人员可能希望将这些信息保密。要解决此问题，可以指定"-s"选项。这将导致e2Image对目录条目进行置乱，并在写入image文件之前将目录块中任何未使用的部分清零。但是，“-s”选项将防止分析与哈希树索引目录相关的问题.

请注意，e2Image创建的qcow2映像是规则的qcow2映像，可以通过知道qcow2格式的工具进行处理，例如qemu-img。

举例

保存sdb4元数据

```
[root@localhost ~]$ e2image /dev/sdb4 sdb4
e2image 1.41.12 (17-May-2010)
[root@localhost ~]$ file sdb4
sdb4: Linux rev 1.0 ext2 filesystem data
```

e2label - 设置第二扩展文件系统的卷标

e2label命令 用来设置第二扩展文件系统的卷标。

修改ext2、ext3、ext4文件系统的标签，如果没有指定标签，那么会显示当前的标签。文件系统标签最长16个字符。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
e2label device [new-label]
```

sh

选项

```
device      # 设备名称  
label       # 标签名称。不指定名称，会显示当前名称。标签名称最多16个字符，超过16个字符将会被自动截断
```

sh

举例

指定sdb4的标签

```
[root@localhost ~]$ e2label /dev/sdb4 hello #指定标签名字  
[root@localhost ~]$ e2label /dev/sdb4          #显示标签名字  
hello  
[root@localhost ~]$
```

sh

echo - 输出指定的字符串或者变量

echo命令 用于在shell中打印shell变量的值，或者直接输出指定的字符串。linux的echo命令，在shell编程中极为常用，在终端下打印变量value的时候也是常常用到的，因此有必要了解下echo的用法echo命令的功能是在显示器上显示一段文字，一般起到一个提示的作用。

echo指令可以输出内容到标准输出，以空白分割字符串，并且后面增加换行。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
echo [-neE] [arg ...]
```

sh

选项

-n	# 输出字符串不换行
-e	# 处理某些特殊字符 # \a 蜂鸣器警报 # \b 删除前一个字符 # \c 最后不加换行 # \e 输出esc字符 # \f 换行，光标停在原处 # \n 换行 # \r 光标移动到首行，不换行 # \t 水平tab # \v 垂直tab # \\ 输出\ # \0nnn 八进制nnn代表的ASCII字符 # \xHH 十六进制数HH代表的ASCII字符
-E	# 禁用转义解释

sh

举例

使用"\f"换行

```
[root@localhost ~]$ echo -e "hello\fworld"      #必须使用-e选项，\f换行之后，光标还在结尾
hello
world
[root@localhost ~]$
```

sh

使用"\n"换行

```
[root@localhost ~]$ echo -e "hello\nworld"      # 必须使用-e选项, \n换行之后, 光标在开头
hello
world
[root@localhost ~]$
```

sh

输出ascii字符

```
[root@localhost ~]$ echo -e "\x31"    # 十六进制的31, 换算成49, 代表的ascii字符就是1
1
[root@localhost ~]$
```

sh

ed - 简单的单行文本编辑程序

ed是简单的单行文本编辑程序，一次只能编辑一行。Ed有两种工作模式，命令模式和输入模式。在输入模式下输入"."并按下回车就可以回到命令模式。

内置命令：

```
a      # 进入输入模式，在最后一行之后输入新内容  
c      # 进入输入模式，输入内容替代最后一行  
i      # 进入输入模式，在当前行之前加入新行输入内容  
d      # 删除最后一行  
n      # 显示最后一行的行号和内容  
w      # 保存，可指定文件名  
q      # 退出
```

sh

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
ed [-GVhs] [-p string] file
```

sh

选项

```
--version    # 显示命令版本信息  
--help       # 显示帮助文档  
-G          # 与老版本兼容  
-p string   # 指定命令模式的提示符  
-s          # 打开文件时不执行检查功能
```

sh

举例

```
[sogrey@bogon demo4]$ cat test.txt # 查看内容
eeeiee eeiee
[sogrey@bogon demo4]$ ed test.txt # 编辑文件
13                                # 显示文件字节数
a                                    # 输入命令a, 进入输入模式
hello world!                         # 输入内容
.                                    # 输入“.”, 并回车, 回到命令模式
w                                    # 输入命令w, 保存文件
26                                # 显示字节数
q                                    # 输入命令q, 退出编辑
[sogrey@bogon demo4]$ cat test.txt
eeeiee eeiee
hello world!
[sogrey@bogon demo4]$
```

eject - 用来退出抽取式设备

eject命令 用来退出抽取式设备。若设备已挂入，则eject命令会先将该设备卸除再退出。

eject允许可移动介质（典型是cd-ROM、软盘、磁带、或者JAZ以及zip磁盘）在软件控制下弹出。该命令也可以控制一些多盘片CD-ROM控制器，控制一些设备支持的自动弹出功能，以及控制一些CD-ROM驱动器磁盘托盘的关闭。与name相应的设备将被弹出，name可以为设备文件或者其挂载点，也可以为完整路径或者省略前面的/dev或者/mnt设备文件名。如果没有指定name，缺省使用cdrom。

有四种不同的弹出的方法，具体要看设备是CD-ROM，SCSI设备，可移动软盘，还是磁带而定。默认的弹出会依次尝试所有四种方法，直到成功为止。如果设备当前是挂载上来的，那么在弹出前要先卸载。

eject指令允许在软件控制下弹出可移动媒体(通常是光盘、软盘、磁带或Jaz或ZIP磁盘)。该命令还可以控制一些由某些设备支持的自动弹出功能的多光盘转换器，并关闭一些光盘驱动器的盘。

对应于 的设备被弹出。名称可以是一个设备文件或挂载点，可以是一个完整的路径，也可以是前面省略的"/dev"、"/media"或"/mnt"。如果未指定名称，则使用默认名称"cdrom"。

根据设备是CD ROM、SCSI设备、可移动软盘还是磁带，有四种不同的弹出方法。默认情况下，弹出将按顺序尝试所有四种方法，直到成功为止。如果该设备目前已安装，则在弹出前将其卸载。

其他说明

eject指令执行成功之后会返回0，如果失败就返回1。

eject指令只适用于支持四种弹出方法中的一种或多种方法的设备。这包括大多数光盘驱动器(IDE、SCSI和专有)、一些SCSI磁带驱动器、Jaz驱动器、ZIP驱动器(并行口、SCSI和IDE版本)和LS 120可移动软盘。用户还报告说，在Sun SPARC和Apple Macintosh系统上，软盘驱动器也取得了成功。如果弹出无法工作，很可能是因为设备的内核驱动程序的限制，而不是弹出程序本身的限制。

-r、-s、-f和-q选项允许控制用于弹出的方法。可以指定多个方法。如果没有指定这些选项，则会尝试所有四个选项(在大多数情况下，这很好)。

eject并不总是能够确定设备是否已安装(例如，它是否有多个名称)。如果设备名称是一个符号链接，弹出将跟随该链接并使用它所指向的设备。

如果eject确定该设备可以具有多个分区，则它将尝试在弹出之前卸载该设备的所有已安装分区。如果卸载失败，程序将不会尝试弹出媒体。

你可以弹出一张音频CD。如果驱动器是空的，一些CDROM将拒绝打开托盘。有些设备不支持托盘关闭命令。

如果启用了自动弹出功能，则在运行此命令后，驱动器将始终弹出。并不是所有的linux内核CDROM驱动程序都支持自动弹出模式，无法找到自动弹出模式的状态。

您需要适当的权限才能访问设备文件。要弹出某些设备(例如SCSI设备)，需要以root或setuidroot的形式运行。

用于查找设备的启发式方法(给定名称)如下所示。如果名称以尾随斜杠结尾，则删除它(这是为了支持使用shell文件名完成生成的文件名)。如果名称以"."或"/"开头，则尝试将其作为设备文件或挂载点打开。如果

失败，它会尝试将’/dev/’、’/media/’、’/mnt/’、’/dev/cdroms’、’/dev/rdsk/’、’/dev/dsk/’和最后’.’放在名称前面，直到找到可以打开的设备文件或挂载点。安装设备的程序检查/etc/mtab。如果失败，它还会检查/etc/fstab以查找当前未挂载设备的挂载点。

建议创建符号链接，如/dev/cdrom或/dev/zip，以便eject可以使用容易记住的名称来确定合适的设备。

要保存类型，可以为特定设置工作的弹出选项创建shell别名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
sh
eject -h
eject [-vnrsfmp] [<name>]
eject [-vn] -d
eject [-vn] -a on|off|1|0 [<name>]
eject [-vn] -c slot [<name>]
eject [-vn] -i on|off|1|0 [<name>]
eject [-vn] -t [<name>]
eject [-vn] -T [<name>]
eject [-vn] -x <speed> [<name>]
eject [-vn] -X [<name>]
eject -V
```

选项

```
-d | --default
```

显示默认的设备名字 (cdrom)

```
-a on|1|off|0
```

这个选项控制自动弹出模式，只有某些设备才支持。如果是能这个开关，设备在关闭的时候会自动弹出。

```
-c <slot> | --changerSlot
```

使用此选项，可以从ATAPI/IDE CD-ROM转换器中选择CD插槽。使用此特性需要Linux2.0或更高版本。当一个更改请求工作

```
-i on|1|off|0
```

此选项可以锁定弹出按钮，使其不工作。当启用时，当按下按钮时，驱动器将不会弹出。这是有用的，当您携带笔记本电脑在

```
-t | --trayclose
```

有了这个选项，驱动器被赋予一个CD-ROM托盘关闭命令。并非所有设备都支持此命令。

```
-T | --traytoggle
```

如果CD-ROM托盘已经打开，那么它将关闭；如果CD-ROM托盘已经关闭，那么它将弹出。并非所有设备都支持此命令，因为它

```
-x <speed> | --cdspeed
```

使用此选项，CDROM驱动器可以进行选择速度。速度参数是一个指示所需速度的数字(例如，8表示8X速度)，或0表示最大数

-X | --listspeed

显示cdrom的可用速度。使用此选项，将探测CD-ROM驱动器以检测可用的速度。输出一个速度列表，可用作-x选项的参数。

-n | --noop

显示所选的设备，但是不执行任何操作

-r | --cdrom

弹出 cdrom设备

-s | --scsi

弹出SCSI设备

-f | --floppy

弹出 floppy设备

-q | --tape

弹出 磁带设备

-p | --proc

允许使用/proc/mounts代替/etc/mtab

-m | --no-unmount

此选项允许eject与自动挂载可移动媒体的设备驱动程序一起工作，因此这些设备必须总是已挂载的。该选项告诉eject不要
-v, --verbose # 执行指令的时候显示详细信息，可以在命令行看到指令在干什么

-h, --help # 显示帮助文档

-V, --version # 显示命令版本信息

举例

sh

```
#弹出默认设备
 eject 

#弹出一个名字为cdrom的设备或者挂载点
 eject cdrom 

#使用设备名来弹出
 eject /dev/cdrom 

#使用挂载点弹出
 eject /mnt/cdrom 

#弹出第4个IDE设备
 eject hdd 

#弹出第一个SCSI设备
 eject sda 

#使用SCSI分区名称弹出
 eject sda4 

#在多盘交换机上选择第5盘
 eject -v -c4 /dev/cdrom 

#打开声音放映机CD-ROM上的自动弹出功能
 eject -a on /dev/sbpcd 
```

没有指定设备类型，直接弹出cdrom。此种情况下会依次尝试所有的方式，直到弹出为止。

sh

```
[sogrey@bogon ~]$  eject -v 
 eject: 使用默认设备“/dev/sr0”
 eject: 设备名称为“/dev/sr0”
 eject: /dev/sr0: 已挂载于 /run/media/sogrey/VBox_GAs_6.1.22
 eject: /dev/sr0: 是全磁盘设备
 eject: /dev/sr0 不是可移动设备
 eject: /run/media/sogrey/VBox_GAs_6.1.22: 正在卸载
 eject: /dev/sr0: 正在尝试使用 CD-ROM 弹出命令弹出
 eject: CD-ROM 弹出命令成功
 [sogrey@bogon ~]$
```

使用"-r"选项，弹出cdrom

```
[sogrey@bogon ~]$ mount # 查看是否有光盘挂载
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime,seclabel)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
devtmpfs on /dev type devtmpfs (rw,nosuid,seclabel,size=2172124k,nr_inodes=543031,mode=755)
...
/dev/sr0 on /run/media/sogrey/VBox_GAs_6.1.22 type iso9660 (ro,nosuid,nodev,relatime,uid=1000,gid=1000,allow_other)
[sogrey@bogon ~]$ eject -v -r # 弹出光盘
eject: 使用默认设备“/dev/sr0”
eject: 设备名称为“/dev/sr0”
eject: /dev/sr0: 已挂载于 /run/media/sogrey/VBox_GAs_6.1.22
eject: /dev/sr0: 是全磁盘设备
eject: /dev/sr0 不是可移动设备
eject: /run/media/sogrey/VBox_GAs_6.1.22: 正在卸载
eject: /dev/sr0: 正在尝试使用 CD-ROM 弹出命令弹出
eject: CD-ROM 弹出命令成功
[sogrey@bogon ~]$
```

eLinks - 纯文本界面的WWW浏览器

eLinks指令是一个纯文本格式的浏览器，支持颜色、表格、鼠标、菜单操作。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
eLinks [OPTION]
```

sh

选项

```
-anonymous      # 匿名使用  
-auto-submit   # 对于遇到的第一个表格，是否自动提交  
-config-dump   # 将配置文件打印到标准输出  
-config-file   # 指定配置文件  
-h              # 显示帮助信息
```

sh

举例

以文本方式访问网站

```
[sogrey@bogon 文档]$ eLinks www.baidu.com
```

sh

访问本地目录

```
[sogrey@bogon 文档]$ eLinks ./demo/
```

sh

emacs - 功能强大的全屏文本编辑器

emacs命令 是由GNU组织的创始人Richard Stallman开发的一个功能强大的全屏文本编辑器，它支持多种编程语言，具有很多优良的特性。有众多的系统管理员和软件开发者使用emacs。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
emacs [选项] [参数]
```

sh

选项

+<行号>: 启动emacs编辑器，并将光标移动到制定行号的行；
-q: 启动emacs编辑器，而不加载初始化文件；
-u<用户>: 启动emacs编辑器时，加载指定用户的初始化文件；
-t<文件>: 启动emacs编辑器时，把指定的文件作为中端，不适用标准输入（stdin）与标准输出（stdout）；
-f<函数>: 执行指定lisp（广泛应用于人工智能领域的编程语言）函数；
-l<lisp代码文件>: 加载指定的lisp代码文件；
-batch: 以批处理模式运行emacs编辑器。

sh

emacs命令操作大全

基本命令

```
C-x C-c : 退出Emacs  
C-x C-f : 打开一个文件，如果文件不存在，则创建一个文件  
C-g : 取消未完成的命令
```

编辑

```
C-z (redefined): Undo; 原来C-z是挂起Emacs（然后用fg命令调出）；C-x u 是默认的命令； 移动一下光标，再C-z)  
M-d : 删除光标后的词语
```

移动光标

```
C-v : 向前翻页  
M-v : 向后翻页  
M-r : 将光标移动到屏幕中间那行  
C-a : 移到行首  
M-a : 移到句首, 从行首到句首之间可能有空格  
C-e : 移到行尾  
M-e : 移到句尾  
M-{ : 向上移动一段  
M-} : 向下移动一段  
C-right : 向前移动一个单词  
C-left : 向后移动一个单词  
C-up : 向前移动一段  
C-down : 向后移动一段  
M-< : 移到整个文本开头  
M-> : 移到整个文本末尾  
C-u 数字 命令 : 执行多次(数字表示次数)该命令; "M-数字 命令" 也可以  
M-x goto-line : 移动到某一行  
C-l : 重绘屏幕, 效果就是当前编辑行移动窗口中央
```

Buffer 相关

```
C-x k : 关闭当前buffer  
C-x b : 切换到前一个编辑的buffer  
C-x C-b : 列出当前所有buffer  
C-x C-s : 保存当前buffer  
C-x s : 保存所有未保存的buffer, 会提示你是否需要保存  
C-x C-w : 文件另存为
```

拷贝与粘贴

```
M-space (redefined): 设置mark; C-@ 是默认命令  
C-w (redefined) : 剪切一块区域; 如果没有设置mark, 则是剪切一行  
M-w (redefined) : 拷贝一块区域; 如果没有设置mark, 则是拷贝一行  
C-k : 从当前位置剪切到行尾  
C-y : 粘贴  
M-y : 用C-y拉回最近被除去的文本后, 换成 M-y可以拉回以前被除去的文本。键入多次的M-y可以拉回更早以前被除去的文本  
C-x r k : 执行矩形区域的剪切  
C-x r y : 执行矩形区域的粘贴
```

窗口操作

```
C-x 0 : 关闭当前窗口  
C-x 1 : 将当前窗口最大化  
C-x 2 : 垂直分割窗口  
C-x 3 : 水平分割窗口  
M-o (redefined) : 在窗口之间切换; C-x o 是默认命令  
C-x 5 1/2/3/0 : 对frame类似的操作  
C-x < : 窗口内容右卷  
C-x > : 窗口内容左卷 (这两个命令在垂直分割窗口后比较有用)  
(C-u) C-x ^ : 加高当前窗口, 如果有C-u, 则每次加高4行  
(C-u) C-x } : 加宽当前窗口  
(C-u) C-x { : 压窄当前窗口  
ESC C-v : 在其它窗口进行卷屏操作
```

搜索和替换

```
C-s : 向前搜索 (增量式搜索); 连续C-s, 跳到下一个搜索到的目标  
C-s RET : 普通搜索  
C-r : 向前搜索  
C-s RET C-w : 按单词查询  
M-% : 查询替换, 也就是替换前会询问一下  
M-x replace-string : 普通替换
```

Tags

```
M-! etags .c .h : 创建TAGS文件  
M-. : 跳到tag所在位置  
M-x list-tags : 列出tags
```

书签

```
C-x r m : 设置书签bookmark  
C-x r b : 跳到bookmark处
```

帮助

```
C-h ? : 查看帮助信息  
C-h f : 查看一个函数  
C-h v : 查看一个变量  
C-h k : 查看一个键绑定 (C-h c 也是查看键绑定, 但是信息较简略)  
C-h C-f : 查看一个函数的info, 非常有用  
C-h i : 看Info
```

其它

```
C-M-\ : 对选中区域, 按照某种格式(比如C程序)进行格式化  
C-x h : 全部选中  
M-! : 执行外部shell命令  
M-x shell : 模拟shell的buffer  
M-x term : 模拟terminal, C-c k 关闭terminal  
C-x C-q : 修改buffer的只读属性
```


enable - 启动或禁用shell内建命令

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
enable [-a] [-dnp] [-f filename] [name ...]
```

sh

- **filename:** 动态库文件名。
- **name (可选) :** 内建命令, 可以为多个。

主要用途

- 禁用一到多个内建命令。
- 启用一到多个内建命令。
- 直接调用与禁用的内建命令同名且在\$PATH路径下找到的外部命令。
- 打印所有内建命令, 无论是否禁用。
- 打印处于启用状态的内建命令。
- 打印处于禁用状态的内建命令。
- 打印处于启用状态的posix标准内建命令。
- 打印处于禁用状态的posix标准内建命令。
- 打印posix标准内建命令, 无论是否禁用。
- 从动态库中加载内建命令。
- 移除从动态库中加载的内建命令。

选项

```
-a      # 打印所有内建命令, 无论是否禁用。  
-d      # 移除从动态库中加载的内建命令。  
-n      # 禁用内建命令或显示已禁用的内建命令。  
-p      # 以可复用格式打印。  
-s      # 只显示处于启动状态的posix标准内建命令。  
-f      # 动态库中加载内建命令。  
-ns     # 打印处于禁用状态的posix标准内建命令。  
-as     # 打印posix标准内建命令, 无论是否禁用。
```

sh

举例

```
# posix special builtin
# 假设没有任何内建命令被禁用
# 禁用两个posix标准内建命令
enable -n set source
# 打印处于禁用状态的posix标准内建命令
enable -ns
# 打印posix标准内建命令，无论是否禁用。
enable -as
# 打印处于启用状态的posix标准内建命令
enable -s
# 假设没有任何内建命令被禁用
# 禁用一到多个内建命令
enable -n echo pwd
# 打印所有内建命令，无论是否禁用。
enable -a
# 打印处于启用状态的内建命令
enable
# 打印处于禁用状态的内建命令
enable -n
# 启用一到多个内建命令
enable pwd
```

eval - eval会调用shell，将参数作为指令来自行

这个指令经常在shell脚本文件中用到。args被读取并连接到一个命令中。然后，shell读取并执行此命令，并将其退出状态作为val的值返回。如果没有args，或者只有空参数，val将返回0。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
eval [arg ...]
```

sh

选项

无

举例

执行指令

```
[root@localhost ~]$ cat wj.txt      #直接执行
1  zhangsan
2  lisi
3  wangwu
4  zhangliu
[root@localhost ~]$ eval cat wj.txt    #通过eval执行，结果一样
1  zhangsan
2  lisi
3  wangwu
4  zhangliu
```

ex - 启动vim编辑器的ex编辑模式

在 ex 模式下启动vim文本编辑器。ex执行效果如同vi -E，适用于法及参数可参照vi指令，如要从Ex模式回到普通模式，则在vim中输入:vi或:visual即可。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
ex [选项] file
```

sh

选项

+num	# 从文本的指定行开始显示
-b	# 进入二进制模式
-c	# 第一个文件编辑完成偶执行指定的指令
-d	# 进入diff模式，编辑多个文件时，显示差异部分
-m	# 不允许修改文件
-n	# 不使用缓存
-o	# 同时打开n个文件
-p	# 以tab形式显示每个文件
-r	# 列出缓存，并显示恢复的信息
-R	# 以只读模式打开
-s	# 静默模式，不显示任何错误信息
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

```
[sogrey@bogon demo4]$ ls
col.txt  hello.txt  test2.txt
[sogrey@bogon demo4]$ ex test.txt  # 进入ex模式编辑
"test.txt" [New File]
Entering Ex mode. Type "visual" to go to Normal mode.
:visual  # visual
```

sh

ex2fsck - 检查ext2、ext3、ext4文件系统

检查ext2、ext3、ext4文件系统，如果系统已经挂载了，那么不建议去检查，因为这样是不安全的。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
e2fsck [-pacnyrdfkvtDFV] [-b superblock]
        [-B blocksize] [-L bad_blocks_file ]
        [-C fd] [-j external-journal] [-E extended_options ] device
```

设备是存储文件系统的设备文件(例如/dev/hdc 1)。

选项

```
sh
-a, -p      # 自动修复文件系统
-b superblock # 指定块大小。不要使用普通的超级块，而是使用由superblock指定的替代超级块。此选项通常在主超
              # 其他备份超级块可以通过使用mke2fs程序使用-n选项打印出创建超级块的位置来确定。mke2fs的-b选
              # 如果指定了另一个超级块，并且文件系统不是只读的，e2fsck将确保主超级块在完成文件系统检查后
              # 正常情况下，e2fsck将在不同块大小下搜索超级块，以试图找到适当的块大小。在某些情况下，此搜
              # 此选项会导致e2fsck使用badblock(8)程序对设备执行只读扫描，以查找任何坏块。如果发现任何坏
              # 此选项导致e2fsck将完成信息写入指定的文件描述符，以便监视文件系统检查的进度。运行e2fsck的
              # 显示调试信息
              # 优化文件系统中的目录。此选项导致e2fsck尝试优化所有目录，如果文件系统支持目录索引，则通过
              # 设置e2fsck扩展选项。扩展选项是逗号分隔的，可以使用等于('=')号进行参数设置。
              # ea_ver=extended_attribute_version，在检查文件系统时，设置e2fsck所需的扩展属性块
              # fragcheck，在传递1期间，为文件系统中的文件打印任何不连续块的详细报告。
              # discard，在进行完整的文件系统检查后，尝试丢弃空闲块和未使用的inode块。
              # nodiscard，不要试图丢弃空闲块和未使用的inode块。这个选项与discard选项正好相反。此
-f          # 强制检查
-F          # 在开始检查之前，清空缓冲区
-I file     # 指定文件中的块添加在损坏列表
-j file     # 设置文件系统在日志文件的路径
-k          # 当与-c选项相结合时，坏块列表中的任何现有坏块将被保留，而通过运行badblocks (8)发现的
              # 将文件名指定的文件中所列的块号添加到坏块的列表中。此文件的格式与badblocks (8)程序生成的
              # 将坏块列表设置为文件名指定的块列表。(此选项与-l选项相同，除非在将文件中列出的块添加到坏
              # 打开文件系统只读，并假设对所有问题的答案为“no”。允许非交互地使用e2fsck。此选项不能与-y选
              # 自动修复(“preen”)文件系统。此选项将导致e2fsck自动修复任何不需要人工干预就能安全修复的
              # 不执行任何操作，提供向后的兼容性
              # 交换文件系统的字节顺序
              # 打印e2fsck的时间统计信息。如果使用此选项两次，则附加的时间统计信息将按传递方式打印。
              # 显示执行的详细过程
              # 显示命令版本号，并且退出
              # 所有的交回应答都回答yes，此选项不能与-n或-p选项同时指定。
```

说明

e2fsck用于检查ext 2/ext 3/ext 4系列文件系统。对于使用日志的ext 3和ext 4文件系统，如果系统在没有任何错误的情况下被不干净地关闭，通常在日志中重播提交的事务之后，文件系统应该标记为干净。因此，对于使用日志的文件系统，e2fsck通常会重播日志并退出，除非它的超级块表明需要进一步检查。

请注意，通常在挂载的文件系统上运行e2fsck是不安全的。唯一的例外是指定了-n选项，并且没有指定-c、-l或-L选项。然而，即使这样做是安全的，如果挂载了文件系统，e2fsck打印的结果也是无效的。

返回值

e2fsck可以返回以下值：

- 0 没有错误。
- 1 文件系统错误更正。
- 2 文件系统错误更正，系统应该重启。
- 4 文件系统错误没有更正。
- 8 操作错误。
- 16 语法错误。
- 32 用户取消了操作。
- 128 共享库错误

信号

对e2fsck发送以下信号，并产生相应的结果：

- SIGUSR1 此信号导致e2fsck开始显示完成栏或发出进度信息。(见对-C选项的讨论。)
- SIGUSR2 此信号导致e2fsck停止显示完成栏或发出进度信息。。

举例

以只读的方式检查sda1

```
[root@bogon ~]$ e2fsck -n /dev/sda1
e2fsck 1.41.12 (17-May-2010)
Warning! /dev/sda1 is mounted.      #文件系统已经挂载，因此会有警告
Warning: skipping journal recovery because doing a read-only filesystem check.    #以只读的方式执
/dev/sda1: clean, 39/128016 files, 49152/512000 blocks
```

exec - 调用并执行指定的命令

exec指令用于执行给定的指令然后退出当前shell， exec并没有创建新的进程，只是替换了原来进程的上下文， 进程PID保持不变。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
exec [-cl] [-a name] [command [arguments]]
```

sh

选项

-l	# 在第0个参数之前增加一个~，类似登录shell一样
-c	# 不使用任何环境变量执行命令
-a	# 将name作为命令的第0个参数

sh

举例

执行指令后退出

```
[root@localhost bak]$ exec cat 1.c      #打印1.c
123
321
#上面的指令执行完毕之后， shell就会退出
```

sh

exit - 退出当前的shell

使shell以状态码n退出。如果省略n，则退出状态是执行的最后一个命令的状态。退出时的陷阱在shell终止之前执行。

主要用途

- 执行exit可使shell以指定的状态值退出。若不设置参数，则以最后一条命令的返回值作为exit的返回值退出。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
exit [n]
```

sh

选项

无

返回值

返回值为你指定的参数n的值，如果你指定的参数大于255或小于0，那么会通过加或减256的方式使得返回值总是处于0到255之间。

举例

退出shell

```
[root@localhost ~]$ tcsh          #切换shell
[root@localhost /~]$ exit 9        #已经切换。退出shell
exit
You have new mail in /var/spool/mail/root
[root@localhost ~]$                      #回到原来的shell
```

sh

expand - 将文件的制表符转换为空白字符

expand命令 用于将文件的制表符（TAB）转换为空白字符（space）， 将结果显示到标准输出设备。

将文件中的tab转换成空格， 结果送到标准输出。如果没有指定文件，那么从标准输入读取。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
expand [选项] file
```

sh

选项

-i, --initial	# 不转换空白行的tab
-t, --tabs	# 指定tab代表的字符数，默认8个
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

```
[sogrey@bogon newDir3]$ ll
总用量 16
-rw-----. 1 sogrey sogrey 38 1月 24 22:46 1.txt
-rw-----. 1 sogrey sogrey 62 1月 24 01:03 students.txt
-rw-----. 1 sogrey sogrey 11 1月 24 01:08 test2.txt
-rw-----. 1 sogrey sogrey 135 1月 24 01:05 test.txt
[sogrey@bogon newDir3]$ cat 1.txt
hello world,
i love linux,
love code.
[sogrey@bogon newDir3]$ expand -t 1 1.txt # 将tab用1个空格代替
hello world,
i love linux,
love code.
[sogrey@bogon newDir3]$
```

sh

export - 为shell变量或函数设置导出属性

export用来设置、删除、修改环境变量，改指令仅在本次登录有效。

主要用途

- 定义一到多个变量并设置导出属性。
- 修改一到多个变量的值并设置导出属性。
- 删除一到多个变量的导出属性。
- 显示全部拥有导出属性的变量。
- 为一到多个已定义函数新增导出属性。
- 删除一到多个函数的导出属性。
- 显示全部拥有导出属性的函数。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
export [-fn] [name[=word]] ...
export -p
```

sh

选项

```
-f      # 设置变量名称为某一函数名称
-n      # 删除某环境变量
-p      # 显示所有环境变量
-o      # 限制变量的值必须是使用-o选项定义的set内置变量
```

sh

如果变量名后面跟着“=word”，则变量的值设置为word。除非遇到无效的选项，否则导出返回退出状态为0，其中一个名称不是有效的shell变量名，或者-f提供的名称不是函数。

举例

显示所有环境变量

```
[root@localhost ~]$ export  
declare -x CLASSPATH=".:/usr/local/src/jdk1.8.0_181/jre/lib/rt.jar:/usr/local/src/jdk1.8.0_181/  
/usr/local/src/jdk1.8.0_181/lib/tools.jar"  
declare -x COLORTERM="gnome-terminal"  
declare -x CVS_RSH="ssh"
```

添加环境变量

```
[root@localhost ~]$ declare var_1="/root"      #定义变量  
[root@localhost ~]$ export var_1                #将变量导出为环境变量  
[root@localhost ~]$ export | grep var_1        #查看环境变量，已经导出  
declare -x var_1="/ROOT"  
[root@localhost ~]$
```

删除环境变量

```
[root@localhost ~]$ export -n var_1          #删除环境变量  
[root@localhost ~]$ export | grep var_1        #查看环境变量，已经删除  
[root@localhost ~]$
```

exportfs - 管理NFS共享文件系统列表

exportfs主要用于管理当前NFS服务器的文件系统。

exportfs 命令用来管理当前NFS共享的文件系统列表。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
/usr/sbin/exportfs [-avi] [-o options,...] [client:/path ..]  
/usr/sbin/exportfs -r [-v]  
/usr/sbin/exportfs [-av] -u [client:/path ..]  
/usr/sbin/exportfs [-v]  
/usr/sbin/exportfs -f
```

sh

选项

```
-a      # 共享nfs配置文件中所有的共享目录  
-i      # 忽略/etc(exports配置文件，只使用exportfs指令的默认值和命令行指定的参数  
-r      # 重新共享所有的nfs文件系统  
-u      # 取消一个或者多个NFS共享文件系统的共享  
-v      # 显示详细执行信息
```

sh

举例

阅读用户david邮件

```
[root@localhost ~]$ exportfs -u 192.168.1.12:/media/test
```

sh

fc - 显示历史列表中的命令或修改指定的历史命令并执行

fc指令用来显示、编辑、执行历史命令。

主要用途

- 显示历史列表中的命令。
- 编辑并重新执行历史列表的命令。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
fc  [-e ename]  [-lnr]  [first]  [last]
fc  -s  [pat=rep]  [cmd]
```

sh

选项

```
-l      # 显示到标准输出
-n      # 不显示行号
-r      # 反序显示
-e      # 编辑命令的文本编辑器, 默认vi
-s      # 找到命令并执行
```

sh

说明

修正命令。在第一种形式中，从历史记录列表中选择从第一到最后的一系列命令。可以将“first”和“last”指定为字符串(查找以该字符串开头的最后一个命令)或作为一个数字(历史列表中的索引，其中负数用作当前命令号的偏移)。如果未指定“last”，则将其设置为当前命令。如果没有指定“first”，则将其设置为用于编辑的前一个命令，并且列出16个记录“-l 16”。

在第二种形式中，在每个PAT实例被REP替换后，重新执行命令。

如果使用第一个形式，则返回值为0，除非遇到无效选项或第一次或最后一次指定超出范围的历史记录线。如果提供了-e选项，则返回值是最后执行的命令的值，如果临时命令文件出现错误，则返回失败。如果使用第二种形式，则返回状态是重新执行的命令的状态，除非cmd没有指定有效的历史记录行，在这种情况下，fc返回失败。

举例

显示最后5条历史命令

```
[root@localhost ~]$ fc -l -5          ##-l指明到标准输出， -5指明显示最后5条
51  fc -l 1 10
52  fc -l 1 4
53  fc -l 1 3
54  fc -l 0 3
55  fc -l 5
[root@localhost ~]$
```

sh

显示最后5条历史命令

```
[root@localhost ~]$ fc -l 51 55      #将51到55条历史命令显示
51  fc -l 1 10
52  fc -l 1 4
53  fc -l 1 3
54  fc -l 0 3
55  fc -l 5
You have new mail in /var/spool/mail/root
[root@localhost ~]$
```

sh

找到命令并执行

```
[root@localhost ~]$ fc -s ps          #找到以ps开头的命令，并执行
ps
  PID TTY          TIME CMD
12137 pts/0    00:00:00 bash
12965 pts/0    00:00:00 ps
```

sh

fg - 将后台作业移动到前台终端运行

fg指令用来将挂载的任务在前台运行。

主要用途

- 用于将后台作业（在后台运行的或者在后台挂起的作业）放到前台终端运行。
- 若后台任务中只有一个，则使用该命令时可以省略任务号。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
fg [jobs]
```

sh

选项

无

举例

切换前台

```
[root@localhost ~]$ jobs      #查看工作
[1]-  Stopped                  wc
[2]+  Stopped                  find / -name passwd
[root@localhost ~]$ fg 2        #将2号工作切换到前台
find / -name passwd
/etc/passwd
/etc/squid/passwd
/etc/pam.d/passwd
[root@localhost ~]$
```

sh

file - 判断指定文件的文件类型

判断指定文件的文件类型，它依据文件内容判断，并不依据扩展名。

file对每个参数进行测试，试图对其进行分类。按照这个顺序执行的测试有三组：文件系统测试、魔术测试和语言测试。成功的第一步将导致打印文件类型。打印的类型通常包含一个单词text(该文件只包含打印字符和几个常见的控制字符，并可能安全地在ASCII终端上读取)、executable(该文件包含以某种UNIX内核可以理解的形式编译程序的结果)，或包含任何其他含义的数据(数据通常是“二进制”或不可打印的)。异常是众所周知的包含二进制数据的文件格式(核心文件、tar存档)。在修改魔术文件或程序本身时，请确保保留这些关键字。用户依赖于知道目录中的所有可读文件都打印了单词“text”。不要像Berkeley那样，把“shell命令文本”改为“shell脚本”

文件系统测试基于检查STAT(2)系统调用的返回。该程序检查该文件是否为空文件，或是否为某种特殊文件。如果在系统头文件中定义了适合您正在运行的系统的任何已知文件类型(套接字、符号链接或有名管道)，则会直观地显示它们。

魔术测试用于检查具有特定固定格式的数据的文件。这方面的典型示例是二进制可执行文件(已编译的程序)a.out文件，其格式在标准include目录中的#include<a.out.h>中定义，或者#include <exec.h>。这些文件具有一个“魔术号”，存储在文件开头附近的某个特定位置，该位置告诉UNIX操作系统该文件是二进制可执行文件，以及其中的几种类型中的哪一种。“魔术”的概念已经通过扩展应用于数据文件。任何在一个小的固定偏移量处具有一些不变标识符的文件通常都可以用这种方式来描述。标识这些文件的信息是从已编译的魔术文件/usr/share/misc/magic.mgc中读取的，如果编译的文件不存在，则从/usr/share/misc/magic目录中读取这些文件。此外，如果\$HOME/.magic.mgc或\$home/.magic存在，则它将优先用于系统魔术文件。如果/etc/magic存在，它将与其他魔术文件一起使用。

如果文件与魔术文件中的任何条目不匹配，则检查它是否为文本文件。ASCII、ISO-8859-x、非ISO 8位扩展-ASCII字符集(例如在Macintosh和IBM PC系统上使用的字符集)、UTF-8编码Unicode、UTF-16编码Unicode和EBCDIC字符集可以通过构成每组可打印文本的不同范围和字节序列来区分。如果一个文件通过了这些测试中的任何一个，就会报告它的字符集。ASCII、ISO-8859-x、UTF-8和Extended-ASCII文件被识别为“Text”，因为它们在几乎任何终端上都是可读的；UTF-16和EBCDIC仅仅是“字符数据”，因为它们包含文本，但是文本需要翻译才能被读取。此外，文件将试图确定文本类型文件的其他特征。如果文件的行为被CR、CRLF或NEL终止，而不是Unix标准LF，这将被报告。还将识别包含嵌入转义序列或过度醒目的文件。

一旦文件确定了在文本类型文件中使用的字符集，它将试图确定该文件是用哪种语言编写的。语言测试寻找特定的字符串(Cf. #include<names.h>)，它可以出现在文件的前几个块中的任何地方。例如，关键字.br表示文件很可能是troff(1)输入文件，正如关键字struct表示C程序一样。这些测试不如前两组可靠，因此它们是最后执行的。语言测试例程还测试一些混类(例如tar(1)存档)。

任何无法标识为已在上述任何字符集中写入的文件，都被简单地称为“data”。

适用范围

RedHat openSUSE	RHEL Fedora	Ubuntu Linux Mint	CentOS Alpine Linux	Debian Arch Linux	Deepin	SUSE
--------------------	----------------	----------------------	------------------------	----------------------	--------	------

语法

```
file [选项] filename
```

选项

```

-b, --brief          # 列出结果的时候, 不显示文件名
-C, --compile        # 编写一个magic.mgc输出文件,
                     # 其中包含魔术文件或目录的预解析版本
-c, --checking-printout # 以检查魔术文件的解析形式的打印输出。
                     # 这通常与 -m 标志一起使用,
                     # 用于在安装新的魔术文件之前调试它。
-e, --exclude testname # 从确定文件类型的测试列表中排除在 testname
                     # 中指定的测试。有效的测试名称是:
                     # apptype, EMX 应用程序类型(仅在EMX上)。
                     # text, 各种类型的文本文件(此测试将尝试猜测
                     # 文本编码, 而不管“编码”选项的设置如何)
                     # encoding, 用于软魔术测试的不同文本编码
                     # tokens, 在文本文件中查找已知的令牌
                     # cdf, 打印复合文档文件的详细信息
                     # compress, 检查并查看压缩文件
                     # elf, 打印 ELF 文件的细节。
                     # soft, 查阅魔法档案。
                     # tar, 检查 tar 文件
-F, --separator      # 指定文件名和结果之间的分隔符, 默认: (冒号)
-f namefile, --files-from namefile # 从给定的文件中, 读取文件名, 然后操作
-h, --no-dereference # 选项导致符号链接不被遵循(在支持符号链接的系统上)。
                     # 如果没有定义环境变量 POSIXLY_TRIDER, 则这是默认的
                     # 导致文件命令输出 MIME 类型字符串, 而不是更传统的人类
                     # 可读的字符串。因此, 它可以说 'text/plain;;'
                     # charset=us-ascii' 而不是 'ASCII 文本'。为了使此选项工作,
                     # 文件更改了它处理命令本身识别的文件的方式(例如许多文本文
                     # 件类型、目录等), 并使用了另一个“魔术”文件。(见下文档案部分)
                     # 类似 -i, 但只打印指定的元素。
                     # 第一次匹配别停, 继续。随后的匹配将具有字符串 '\012-'。
                     # (如果需要换行符, 请参见 '-r' 选项。)
-L, --dereference   # 选项导致符号链接被遵循, 就像 ls(1) 中的同名选项(在支持符号
                     # 链接的系统上)。如果定义了环境变量 POSIXLY_TRIDER, 则这是默认的
                     # 指定包含魔术的文件和目录的备用列表。这可以是单个项目,
                     # 也可以是冒号分隔的列表。如果在文件或目录旁边找到已编译的魔术文件,
                     # 则将使用它。
-m, --magic-file magicfiles
                     # 让文件名在输出中对齐
                     # 检查每个文件后, 强迫 stdout 被刷新。这只有在检查文件列表时才有用。
                     # 它用于希望从管道输出文件类型的程序。
                     # 在支持 utime(2) 或 utime(2) 的系统上, 尝试保留分析过的文件的访问时间,
                     # 假装文件从未读取过它们。
                     # 不要将不可打印的字符翻译为 \ooo。通常, 文件将不可打印的字符转换为
                     # 它们的八进制表示形式
                     # 通常, 文件只尝试读取和确定 STAT(2) 报告是普通文件的参数文件的类型。
                     # 这可以防止出现问题, 因为读取特殊文件可能会产生特殊的后果。
                     # 指定 -s 选项会导致 file 也读取参数文件, 这些参数文件是块文件或字符特殊
                     # 这对于确定原始磁盘分区中数据的文件系统类型非常有用, 这些分区是块特殊
                     # 此选项还会导致文件忽略 stat(2) 所报告的文件大小, 因为在某些系统上,
                     # 它报告原始磁盘分区的大小为零。
                     # 试着查看压缩文件
                     # 在文件名结束后输出空字符 '\0'。很好地削减了产量。
                     # 这不影响仍然打印的分隔符。
-z                 # 尝试读取压缩文件的内容
--help             # 显示帮助文档
--version          # 显示命令版本信息

```

返回值

文件在成功时返回0，在错误时返回非零。如果文件操作数命名的文件不存在，无法读取，或者无法确定由文件操作数命名的文件的类型，则不认为这是影响退出状态的错误。

环境变量

环境变量MAGIC可以用来设置默认的魔术文件名。如果设置了该变量，那么文件将不会尝试打开\$HOME/.magic。文件酌情将'.mgc'添加到此变量的值中。环境变量POSIXLY_RIDER控制(在支持符号链接的系统上)，文件是否会尝试遵循符号链接。如果设置了，那么文件将遵循符号链接，否则就不会。这也是由-l和-h选项控制的。

举例

```
[sogrey@bogon demos]$ file test.txt  
test.txt: UTF-8 Unicode text  
[sogrey@bogon demos]$ file -F. test.txt  
test.txt. UTF-8 Unicode text
```

sh

find - 在指定目录下查找文件

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使用该命令。

在指定的目录下查找文件，并可对找到的文件执行指定的操作。Find指定会从指定的目录向下递归搜索各个子目录。GNU find根据优先级规则从左到右计算给定表达式，从而搜索根植于每个给定文件名的目录树，直到找到结果，此时find移到下一个文件名。如果在安全很重要的环境中使用find(例如，如果您使用它搜索其他用户可写的目录)，则应阅读findutils文档中的“Security Considerations”一章。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
find [-H] [-L] [-P] [-D debugopts] [-Olevel] [path...] [expression]
```

sh

选项

```
-P # 找到符号链接的时候，所有的属性都来自符号链接。这是默认选项  
-L # 找到符号链接的时候，所有的属性来自文件本身，而不是符号链接  
-H # 找到符号链接的时候，所有的属性都来自符号链接。  
-D debugoptions # 打印诊断信息；这可能有助于诊断问题，为什么查找不到您想做的事情。调试选项列表应  
# - help，解释调试选项  
# - tree，以其原始和优化的形式显示表达式树  
# - stat，使用stat和lstat系统调用检查作为文件的打印消息。find程序试图将此类调  
# - opt，打印有关表达式树优化的诊断信息；请参见-O选项。  
# - rates，打印一个摘要，指示每个谓词成功或失败的频率。  
-Olevel # 启用查询优化。Find程序重新排序测试以加快执行速度，同时保持总体效果；也就是说，  
# - 0，相当于优化级别1  
# - 1，这是默认的优化级别，与传统行为相对应。表达式被重新排序，以便首先执行仅基于  
# - 2，任何-type或-xtype的测试都是在只基于文件名的任何测试之后，而是在需要inod  
# - 3，在此优化级别上，启用了完全基于成本的查询优化器。对测试顺序进行了修改，以便  
# 提高速度并降低内存使用量。  
-help, --help # 显示帮助文档  
-version, --version # 显示命令版本信息
```

说明

-H, -L和-P选项控制符号链接的处理。后面的命令行参数被视为要检查的文件或目录的名称，直到

以'-'或'('或'!'开头的第一个参数为止。该参数和随后的任何参数都被视为描述要搜索的内容的表达式。如果没有提供路径，则使用当前目录。如果没有给出表达式，则使用表达式"-print"。

这些选项控制find的行为，但在上一个路径名之后立即指定。五个"实"选项-H, -L, -P, -D和-O必须出现在第一个路径名之前，如果有的话。“--”也可以用来表示任何剩余的参数都不是选项(但如果在起始点列表中使用通配符，则确保所有起始点都以“./”或“/”开头通常更安全)。

如果指定了-H、-L和-P中的一个，最后一个出现在命令行上的操作将生效。由于-P选项是默认的，因此，除非指定了-H或-L，否则-P选项应视为有效。在命令行本身的处理过程中，在搜索开始之前，GNU经常查找stats文件。当-H或-L选项生效时，列出的任何作为-newer参数的符号链接将被取消引用，并且时间戳将从符号链接指向的文件中提取。同样的考虑也适用于-newerxy, -anewer和-cnewer。

表达式

表达式由OPTION(这些选项影响整体操作而不是处理特定文件，并且始终返回true)、TEST(返回真或假值)和ACTION(有副作用并返回真或假值)组成，所有这些选项都由操作符'-'分隔。如果表达式不包含"-prune"以外的操作，则对表达式为true的所有文件执行"-print"。

1. OPTION

所有选项总是返回true。除了-daystart、-follow和-regextype之外，这些选项会影响所有测试，包括在选项之前指定的测试。这是因为这些选项是在分析命令行时处理的，而测试在检查文件之前不会执行任何操作。在这方面，-daystart、-follow和-regextype选项是不同的，并且只对稍后出现在命令行中的测试有影响。因此，为了清晰起见，最好将它们放在表达式的开头。如果不这样做，就会发出警告。

```
sh
-d          # depth的同义词，与FreeBSD、NetBSD、MacOSX和OpenBSD兼容。
-daystart   # 测量时间(用于-amin, -atime, -cmin, -ctime, -mmin和-mtime)从今天开始，而不是从24小时
-depth      # 在目录本身之前处理每个目录的内容。
-follow     # 使用-L选项代替。解除引用符号链接。“follow”选项只影响在命令行中出现的测试。
-help, --help # 打印出帮助信息，并且退出
-ignore_readdir_race # 通常情况下，Find将在无法统计文件时发出错误消息。如果您提供了此选项，并且在find从目
-maxdepth levels  # 查找的最大深度，“-maxdepth 0”只对命令行出现的TEST和ACTION有效
-mindepth levels  # 不要在低于levels的级别上应用任何测试或操作，“-maxdepth 1”意味着处理除命令行参数以外
-mount       # 不要将目录降到其他文件系统上。-xdev的替代名称，用于与find的其他版本兼容。
-noignore_readdir_race # 关闭-ignore_readdir_race。
-noleaf       # 不要通过假设目录包含的子目录比硬链接数少两个来进行优化。在搜索不遵循Unix目录链接约定
-regextype type    # 更改命令行后面发生的-regex和-iregex测试所理解的正则表达式语法。
-version, --version # 打印find命令的版本信息，并且退出
-warn, -nowarn    # 关闭或者打开警告信息
-xautofs        # 不要降低autofs文件系统上的目录
-xdev           # 不要降低其他文件系统上的目录
```

2. TEST

有些测试，例如newerXY和-samefile，允许比较目前正在检查的文件和命令行上指定的引用文件。当使用这些测试时，引用文件的解释由选项H、-L和-P以及前面的任何选项决定，但是在解析命令行时，只检查一次引用文件。如果无法检查引用文件(例如，stat(2)系统调用失败)，则会发出错误消息，并以非0状态退出。

```

+n          # 大于n
-n          # 小于n
n           # n
-amin m    # 查找m分钟之前被访问过的文件
-anewer file # 最近被访问的文件，而不是最近被修改的文件
-ctime m    # 最近m天前被访问的文件
-cmin m    # 查找m分钟之前被修改过文件状态的文件
-cnewer file # 最近被未改过文件状态的文件，而不是最近被修改的文件
-ctime m    # 最近m天前被修改文件状态的文件
-mmin m    # 查找最近m分钟前被修改过内容的文件
-mtime m    # 查找最近m天前被修改过内容的文件
-empty      # 查找大小为0的目录和文件
-executable # 查找可以被执行的文件，或者可以被搜索的目录
-fstype type # 查找在指定文件系统上的文件
-gid id     # 查找属于指定组id的文件
-group name # 查找属于指定组的文件
-ilname pattern # 和“-lname”一样，但匹配是不区分大小写的。如果-L选项或-follow选项生效，则除非符号链接中
-iname pattern # 和“-name”一样，但匹配是不区分大小写的
-inum n      # 文件inode编号n。通常情况下，使用-samefile测试更容易。
-ipath pattern # 和“-iwholename”一样，不建议使用
-iregex pattern # 和“-regex”一样，但是匹配不区分大小写
-iwholename pattern # 和“-wholename”一样，但是匹配不区分大小写
-links n     # 有n个链接的文件
-lname pattern # 找找符合指定匹配模式的符号链接文件
-name pattern # 文件名的基本(删除了前导目录的路径)与shell模式匹配。元字符(“*”、“?”和“[ ]”)与“.”匹配。
-newer file   # 比file更近修改的文件
-newerXY reference # 将当前文件的时间戳与引用进行比较。引用参数通常是文件的名称(其中一个时间戳用于比较),
                  # a, 文件的访问时间。
                  # B, 文件的产生时间
                  # c, 文件inode状态改变的时间
                  # m, 文件的修改时间
                  # t, 引用被直接解释为时间。
-nogroup     # 没有组对应于文件的组ID
-nouser      # 没有用户对应于文件的用户ID。
-path pattern # 文件名与shell模式匹配。元字符不处理“/”或‘.’，例如“find . -path ‘./sr*sc’”为
-perm -mode   # 为文件设置了所有权限位模式。符号模式以这种形式被接受，这通常是想要使用它们的方式。如
-perm /mode   # 为该文件设置任何权限位模式。符号模式以这种形式被接受。如果使用符号模式，则必须指定“u
-perm +mode   # 不推荐使用这种模式集中任何权限位搜索文件的旧方法。你应该改用“-perm /mode”。例如,
-size n[cwbkMG] # 查找指定大小的文件。文件单位默认是块，512字节。有以下单位:
                  # - ‘b’, 521字节的块，默认
                  # - ‘c’, bytes
                  # - ‘w’, 2个字节
                  # - ‘k’, 1024字节
                  # - ‘M’, 1048576字节
                  # - ‘G’, 1073741824字节
-type 类型   # 查找指定类型的文件。c, 字符设备; d, 目录文件; p, 有名管道; f, 普通文件; l, 符号链接
-uid n        # 指定文件的uid
-used n       # 最后一次访问文件是在最后一次更改状态后的n天。
-user name    # 查找属于指定用户的文件
-wholename pattern # 等价于“-path”
-writable     # 匹配可写的文件。
-xtype c      # 和“-type”相同，除非文件是符号链接。对于符号链接，-xtype检查-type不检查的文件的类
-context pattern # 文件的安全上下文与GLOB模式相匹配。

```



3. ACTIONS

```
sh
-delete          # 删除文件；如果删除成功，则为true。如果删除失败，则发出错误消息。如果“-delete”失败
-exec command    # 执行命令；如果返回0状态，则为true。以下所有要查找的参数都被视为命令的参数，直到遇到
-exec command {} + # -exec操作的这个变体在选定的文件上运行指定的命令，但是命令行是通过在末尾追加每个选定
-execdir command ; #
-execdir command {} + # 类似-exec，但指定的命令是从包含匹配文件的子目录运行的，该子目录通常不是您开始查找
-fls file        # 真，类似“-ls”，但是写文件类似“-fprint”。输出文件总是被创建，即使谓词永远不匹配。
-fprint file      # 真，将完整的文件名打印到文件中。如果运行find时文件不存在，则创建文件；如果文件存
-fprintf0 file    # 真，类似“-printf0”，但是写文件类似“-fprint”。输出文件总是被创建，即使谓词永远不
-fprintf file format # 真，类似“-printf”但是写到文件类似“-fprint”。输出文件总是被创建，即使谓词永远不
-ls               # 真，在标准输出上以“ls -dils”格式列出当前文件。块计数为1K块，除非设置了环境变量P
-ok command ;     # 类似“-exec”，但是先问用户。如果用户同意，运行命令。否则只会返回FALSE。如果运行该
-okdir command ; # 类似“-execdir”，但是和“-ok”一样先询问用户。如果用户不同意，只需返回false。如果
-print             # 真，在标准输出上打印完整的文件名，然后是换行符。
-print0            # 真，在标准输出上打印完整的文件名，然后是空字符
-printf format     # 真，在标准输出上打印格式，解释‘\’转义和‘%’指令。字段宽度和精度可以用“printf C”
-prune             # 如果该文件是一个目录，则不要下降到它
-quit              # 马上退出。不会继续运行子进程，但不会处理命令行上指定的路径。
```

4. 操作符，按优先次序排列

```
sh
(expr)           # 强制优先
! expr            # 取反
-not expr         # 取反，但不适合POSIX
expr1 expr2       # 隐含的“与”操作
expr1 -a expr2   # 和“expr1 expr2”一样
expr1 -and expr2 # 和“expr1 expr2”一样，但是不适合POSIX
expr1 -o expr2   # “或”操作
expr1 -or expr2  # “或”操作，不适合POSIX
expr1 , expr2    # 始终对expr1和expr2进行计算。expr1的值被丢弃；列表的值是expr2的值。逗号运算符对于
```

环境变量

- `LANG` 为未设置或空的全局变量提供默认值。
- `LC_ALL` 如果设置为非空字符串值，则重写所有其他全局变量的值。
- `LC_COLLATE` POSIX标准指定此变量影响用于-name选项的模式匹配。对'LC_COLLATE'的支持取决于系
统库。这个变量还会影响对“-ok”的响应的解释；当'LC_MESSAGES'变量选择用于将响应解释为“-ok”的
实际模式时，模式中任何括号表达式的解释都会受到'LC_COLLATE'的影响。
- `LC_CTYPE` 如果系统的fnmatch(3)库函数支持该变量，则此变量将影响正则表达式中使用的字符类的
处理以及名称测试。此变量还会影响用于解释“-ok”发出的提示的响应的正则表达式中任何字符类的解
释。当打印文件名时，'LC_CTYPE'环境变量还将影响哪些字符被认为是不可打印的；
- `LC_MESSAGES` 确定要用于全局消息的区域设置。如果设置了'POSIXLY_REVERT'环境变量，这还将确定
对“-ok”操作所作提示的响应的解释
- `NLSPATH` 确定全局信息目录的位置。
- `PATH` 影响搜索以查找-exec、-execdir、-ok和-okdir调用的可执行文件的目录。
- `POSIXLY_CORRECT` 确定-ls和-fls使用的块大小。如果设置了POSIXLY_RIDERT，则块是512字节的单
位。否则，它们是1024字节的单位。
- `TZ` 影响用于-printf和-fprintf的一些与时间相关的格式指令的时区。

返回值

如果成功，返回0；如果失败，返回大于0的数。

举例

```
find /tmp -name core -type f -print, xargs /bin/rm -f
```

sh

在/tmp目录中或下面查找名为core的文件并删除它们。请注意，如果存在包含换行符、单引号或双引号或空格的文件名，则此操作将不正确。

```
find /tmp -name core -type f -print0, xargs -0 /bin/rm -f
```

sh

在目录/tmp中或下面查找名为core的文件，并删除它们，处理文件名，以便正确处理包含单引号或双引号、空格或换行符的文件或目录名称。名称测试出现在-type测试之前，以避免对每个文件调用stat(2)。

```
find . -type f -exec file '{}' \;
```

sh

对当前目录中或当前目录下的每个文件运行“file”。注意，大括号被用单引号括起来，以保护它们不被解释为shell脚本标点符号。分号同样受到反斜杠的保护，尽管在这种情况下也可以使用单引号。

```
find / \(\ -perm -4000 -fprintf /root/suid.txt '%#m %u %p\n' \), \(\ -size +100M -fprintf /root/big.txt '%#m %u %p\n' \)
```

sh

只遍历文件系统一次，将setuid文件和目录列出到/root/suid.txt，并将大型文件列出到/root/big.txt

```
find $HOME -mtime 0
```

sh

在您的主目录中搜索在过去24小时内已经修改的文件。此命令的工作方式是这样的，因为每个文件上次修改后的时间被除以24小时，其余部分被丢弃。这意味着要匹配-mtime 0，文件必须在过去进行修改，而修改时间不到24小时前。

```
find /sbin /usr/sbin -executable \! -readable -print
```

sh

搜索可执行但不可读的文件

```
find . -perm 664
```

sh

搜索所有者和组有读写权限，但其他用户可以读取但不能写入的文件。满足这些条件但设置了其他权限的文件(例如，如果有人可以执行该文件)将不匹配。

```
find . -perm -664
```

sh

搜索所有者和组有读写权限的文件，以及其他用户可以读取的文件，而不考虑是否存在任何额外的权限位

(例如可执行位)。例如，这将匹配具有模式0777的文件。

```
find . -perm /222
```

sh

搜索可写的文件

```
find . -perm /220  
find . -perm /u+w,g+w  
find . -perm /u=w,g=w
```

sh

所有这三个命令都执行相同的操作，但是第一个命令使用文件模式的八进制表示，另外两个命令使用符号形式。这些命令都搜索其所有者或组可写的文件。文件不必由所有者和组同时写入才能匹配，两者都可以。

```
find . -perm -220  
find . -perm -g+w,u+w
```

sh

这两个命令执行相同的操作；搜索它们的所有者和组都可以写的文件。

```
find . -perm -444 -perm /222 ! -perm /111  
find . -perm -a+r -perm /a+w ! -perm /a+x
```

sh

这两个命令都搜索每个人都可读的文件(-perm -444或-perm -a+r)，至少设置了一个写入位集(-perm /222或-perm /a+w)，但任何人都不能执行(! -perm /111和! -perm /a+x)。

```
cd /source-dir  
find . -name .snapshot -prune -o \(\ ! -name *~ -print0 \|  
cpio -pmd0 /dest-dir
```

sh

此命令将/source-dir的内容复制到/dest-dir，但省略了名为".snapshot"的文件和目录(以及其中的任何内容)。它还省略了名称以~结尾的文件或目录，而不是其内容。

```
find repo/ -exec test -d {}/.svn -o -d {}/.git -o -d {}/CVS ; -print -prune
```

sh

给定以下项目目录及其相关的SCM管理目录，高效地搜索项目的根

```
repo/project1/CVS  
repo/gnu/project2/.svn  
repo/gnu/project3/.svn  
repo/gnu/project3/src/.svn  
repo/project4/.git
```

在根目录下查找doc后缀的文件

```
[sogrey@bogon 文档]$ find / -name *.doc
find: 探测到文件系统循环; “/var/named/chroot/var/named” 是与 “/var/named” 相同的文件系统循环的一部分。
/usr/share/cvs/contrib/intro.doc
/usr/lib/python2.6/pdb.doc
/lib/kbd/keymaps/i386/qwerty/no-latin1.doc
```

将查找到的内容输出到res.txt

```
[sogrey@bogon 文档]$ find / -name *.doc -fprint res.txt # 找到文件, 输出到res.txt
find: 探测到文件系统循环; “/var/named/chroot/var/named” 是与 “/var/named” 相同的文件系统循环的一部分。
[sogrey@bogon 文档]$ cat res.txt    #查看输出结果
/usr/share/cvs/contrib/intro.doc
/usr/lib/python2.6/pdb.doc
/lib/kbd/keymaps/i386/qwerty/no-latin1.doc
```

查找属于指定用户的文件

```
[sogrey@bogon 文档]$ find / -user david -perm 777 #查找属于用户david, 并且权利是777的文件
find: 探测到文件系统循环; “/var/named/chroot/var/named” 是与 “/var/named” 相同的文件系统循环的一部分。
/wj/_主页logo魏杰it教育.jpg
find: “/proc/7990/task/7990/fd/5”: 没有那个文件或目录
find: “/proc/7990/task/7990/fdinfo/5”: 没有那个文件或目录
find: “/proc/7990/fd/5”: 没有那个文件或目录
find: “/proc/7990/fdinfo/5”: 没有那个文件或目录
```

findfs - 标签或UUID查找文件系统

findfs命令 依据卷标（Label）和UUID查找文件系统所对应的设备文件。findfs命令会搜索整个磁盘，看是否有匹配的标签或者UUID没有，如果有则打印到标注输出上。findfs命令也是e2fsprogs项目的一部分。

查找指定卷标或者UUID的文件系统对应的设备文件。findfs将搜索系统中的磁盘，寻找具有标签匹配标签或与UUID相等的文件系统。如果找到文件系统，文件系统的设备名称将打印在stdout上。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
findfs LABEL=label  
findfs UUID=uuid
```

sh

选项

无

举例

查找指定UUID的文件系统的设备

```
[root@localhost ~]# findfs UUID=059facc9-c58e-42d0-b8f5-7644c4574888  
/dev/sda1  
You have new mail in /var/spool/mail/root
```

sh

查找指定LABEL的文件系统的设备

```
[root@localhost ~]$ findfs LABEL=wj  
/dev/sdb4
```

sh

finger - 用于查找并显示用户信息

finger命令 用于查找并显示用户信息。包括本地与远端主机的用户皆可，帐号名称没有大小写的差别。单独执行finger指令，它会显示本地主机现在所有的用户的登陆信息，包括帐号名称，真实姓名，登入终端机，闲置时间，登入时间以及地址和电话。

finger指令用来查找、显示指定用户的信息。查询远程主机信息是，可以用user@localhost来指定用户。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
finger [-lmsp] user
```

sh

选项

-s	# 显示用户的用户名、真实姓名、登录终端、闲置时间、登录时间、地址、电话
-l	# 显示用户的用户名、真实姓名、用户家目录、登录后的shell、登录时间、电子邮件、计划文件
-p	# 和-l一样，但是不显示“.plan”、“.project”、“.pgpkey”文件
-m	# 不查找用户真实姓名

sh

举例

查看用户root信息：

```
[sogrey@bogon 文档]$ finger -l root
Login: root                      Name: root
Directory: /root                  Shell: /bin/bash
On since 五 9月 7 21:02 (CST) on tty1 from :0
    14 days 20 hours idle
On since 六 9月 22 07:36 (CST) on pts/0 from :0.0
    9 hours 5 minutes idle
On since 四 9月 13 08:55 (CST) on pts/1 from :0.0
New mail received 六 9月 22 18:00 2018 (CST)
    Unread since 二 8月 21 09:22 2018 (CST)
No Plan.
```

sh

fmt - 读取文件后优化处理并输出

fmt命令 读取文件的内容，根据选项的设置对文件格式进行简单的优化处理，并将结果送到标准输出设备。

将指定文件的内容，按照指定的格式重新排版，结果送到标准输出。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
fmt [width] [选项] file
```

sh

选项

```
-c, --crown-margin      # 每段前两行缩排
-p, --prefix=STRING     # 重新排版以指定字符串开头的行
-s, --split-only        # 将长行分割开
-t, --tagged-paragraph  # 将第一行缩进
-u, --uniform-spacing   # 字与字之间一个空白，句子后两个空白
-w, --width=WIDTH       # 设置每行字符数，默认75]

--help                  # 显示帮助文档
--version               # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ ll
总用量 16
-rw-----. 1 sogrey sogrey 38 1月 27 23:42 1.txt
-rw-----. 1 sogrey sogrey 62 1月 24 01:03 students.txt
-rw-----. 1 sogrey sogrey 11 1月 24 01:08 test2.txt
-rw-----. 1 sogrey sogrey 135 1月 24 01:05 test.txt
[sogrey@bogon newDir3]$ cat 1.txt
Hello linux,
I'm Sogrey.
Hello world!
[sogrey@bogon newDir3]$ fmt 1.txt # 直接格式化, 将所有的内容合并成行
Hello linux, I'm Sogrey. Hello world!
[sogrey@bogon newDir3]$ cat 1.txt
Hello linux,
I'm Sogrey.
Hello world!
[sogrey@bogon newDir3]$ fmt -5 -t 1.txt # 每行5个字符, 第一行缩进
Hello
    linux,
I'm
    Sogrey.
Hello
    world!
[sogrey@bogon newDir3]$
```

fold - 按照指定的宽度显示文件

fold命令 用于控制文件内容输出时所占用的屏幕宽度。fold命令会从指定的文件里读取内容，将超过限定列宽的列加入增列字符后，输出到标准输出设备。若不指定任何文件名称，或是所给予的文件名为“-”，则fold指令会从标准输入设备读取数据。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
fold [选项] file
```

sh

选项

```
-b, --bytes          # 以字节为单位, 指定宽度  
-c, --characters    # 以字符为单位, 指定宽度  
-s, --space          # 以空格分割  
-w, --width          # 指定列宽, 默认30  
  
--help                # 显示帮助文档  
--version             # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir]$ cat test.txt
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon newDir]$ fold -b16 test.txt # 每16个字节一行输出
石家庄今日
新增16例确
诊病例
中国留美博
士遇害 美驻
华使馆慰问
特朗普夫人
发文谴责国
会暴乱
理塘文旅公
司回应丁真
抽烟
北京一确诊
者隐瞒行程
不配合流调
山西晋中新
增2例无症状
感染者
[sogrey@bogon newDir]$
```

free - 显示内存的使用情况

free命令 可以显示当前系统未使用的和已使用的内存数目，还可以显示被内核使用的内存缓冲区。

free指令用来显示内存的使用情况，显示系统中可用和已使用的物理和交换内存的总量，以及内核使用的缓冲区。应该忽略共享内存列；它已经过时了。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
free [OPTION]
```

sh

选项

```
-b      # 显示内存使用情况，单位byte  
-k      # 显示内存使用情况，单位KB  
-m      # 显示内存使用情况，单位MB  
-o      # 不显示缓冲区调节行  
-t      # 显示内存总和  
-s      # 按照指定的时间间隔显示内存使用情况  
-l      # 显示详细的低内存和高内存提高统计数据  
-v      # 显示命令的版本并退出
```

sh

举例

以kb为单位显示

```
[sogrey@bogon ~]$ free -k # 以kb为单位显示  
total        used        free        shared    buff/cache    available  
Mem:        4380708      819580     2827380      12040      733748     3320896  
Swap:       4587516        0        4587516  
[sogrey@bogon ~]$
```

sh

以MB为单位显示

```
[sogrey@bogon ~]$ free -m -t # 以MB为单位显示，同时显示总和  
total        used        free        shared    buff/cache    available  
Mem:        4278        801        2760        11        716        3242  
Swap:       4479        0        4479  
[sogrey@bogon ~]$
```

sh

fsck - 检查并且试图修复文件系统中的错误

fsck命令被用于检查并且试图修复文件系统中的错误。当文件系统发生错误时，可用fsck指令尝试加以修复。

检查或者修复指定的文件系统，可以是设备名、挂载点，还可以是一个ext2的label，或者是一个UUID。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
fsck [选项] -t systype device  
fsck [-sAVRTMNP] [-C [fd]] [-t fstype] [filesys...] [--] [fs-specific-options]
```

选项

```
-s          # 检查的序列。如果有多个文件系统需要检查，那么按照顺序来操作。注意：e2fsck(8)默认以交互模式运行。  
-t fslist   # 指定要检查的文件系统的类型。当指定-A标志时，只检查与fslist匹配的文件系统。fslist参数是以逗号分隔的。  
           # 选项说明符可能包含在逗号分隔的fslist中。它们必须具有“opts=s-option”的格式。如果存在选项说明符，则  
           # 为了兼容Mandrake发行版，它的引导脚本依赖于对fsck程序的未经授权的ui更改，如果在fslist中找到  
           # 通常，文件系统类型是通过在“/etc/fstab”文件中搜索文件并使用相应的条目来推断的。如果无法推断出  
-A          # 遍历“/etc/fstab”文件，并尝试在一次运行中检查所有文件系统。此选项通常来自“/etc/rc”系统初始化脚本。  
           # fsck不与任何其他设备并行检查堆叠设备。因此，在“/etc/fstab”文件中非常常见的配置是将根文件系统设为  
           # fsck通常不会在调用文件系统的检查器之前检查设备是否实际存在。因此，如果文件系统特定的检查器  
-C [fs]      # 显示检查的进度。显示支持这些文件系统检查程序的完成/进度条(目前只用于ext 2和ext 3)。Fsck将  
           # 不要检查已安装的文件系统，并返回安装文件系统的退出代码0。  
-M          # 不执行检查操作，只是演示一下。  
-N          # 不设置-A标志时，与其他文件系统并行检查根文件系统。这不是世界上最安全的事情，因为如果根文件系统  
           # 设置-A标志时，与其他文件系统并行检查根文件系统。这不是世界上最安全的事情，因为如果根文件系统  
-P          # 当使用-a标志检查所有文件系统时，跳过根文件系统。  
-R          # 启动时不要显示标题。  
-T          # 显示执行过程。  
-V          # 显示执行过程。  
fs-specific-options # fsck不理解的选项被传递给特定于文件系统的检查器。这些参数不能使用参数，因为fsck无法正确地  
           # 处理它们。请注意，fsck的设计并不是为了将任意复杂的选项传递给特定于文件系统的检查器。如果您正在做一些  
           # 不同文件系统特定fsck的选项没有标准化。如果有疑问，请查阅文件系统特定检查器的手册页。虽然没有  
-a          # 自动修复文件系统。请注意e2fsck(8)只支持向后兼容。此选项映射到e2fsck的-p选项，与某些文件系  
           # 对于某些特定于文件系统的检查程序，-n选项将导致fs特定的fsck避免试图修复任何问题，但只需将此  
-n          # 以交互方式修复文件系统(请求确认)。注意：如果多个fsck并行运行，使用此选项通常是个坏主意。还  
           # 对于某些特定于文件系统的检查程序，-y选项将导致fs特定的fsck总是试图自动修复任何检测到的文件  
-r          # 系统。  
-y          # 对于某些特定于文件系统的检查程序，-y选项将导致fs特定的fsck总是试图自动修复任何检测到的文件  
           # 系统。
```

说明

如果命令行上没有指定任何文件系统，并且没有指定-A选项，fsck将默认为串行地检查/etc/fstab中的文件系统。fsck指令可以有以下的返回值：

- 0 没有错误。
- 1 文件系统错误更正。
- 2 系统应该重启。
- 4 系统错误没有更正。
- 8 操作错误。
- 16 语法错误。
- 32 用户取消fsck。
- 128 共享库错误。

检查多个文件系统时返回的退出代码是所检查的每个文件系统的退出代码的逐位OR。

实际上，fsck只是linux下可用的各种文件系统检查器(fsck.fstype)的前端。文件系统特定的检查器首先在/sbin中搜索，然后在/etc/fs和/etc中搜索，最后在PATH环境变量中列出的目录中搜索。

环境变量

fsck的执行收到以下环境变量的影响：

1. FSCK_FORCE_ALL_PARALLEL，如果设置了此环境变量，fsck将尝试并行运行所有指定的文件系统，而不管文件系统是否位于同一设备上。(这对于RAID系统或高端存储系统(如IBM或EMC等公司销售的存储系统)非常有用。)请注意，fs_passno值仍被使用。
2. FSCK_MAX_INST，这个环境变量将限制一次运行的文件系统检查器的最大数量。这允许具有大量磁盘的配置避免fsck一次启动过多的文件系统检查器，这可能会使系统上可用的CPU和内存资源超载。为零，则可以生成无限数量的进程。这是当前的默认情况，但未来版本的fsck可能会尝试根据从操作系统收集会计数据自动确定可以运行多少个文件系统检查。
3. PATH，PATH环境变量用于查找文件系统检查器，首先搜索一组系统目录：/sbin、/sbin/fs.d、/sbin/fs、/etc/fs和/etc/fs，然后搜索路径环境中的一组目录。
4. FSTAB_FILE，这个环境变量允许系统管理员覆盖/etc/fstab文件的标准位置，对于正在测试fsck的开发人员也很有用。

举例

检查sdb4

```
[root@localhost ~]$ fsck -t swap /dev/sdb4
fsck from util-linux-ng 2.17.2
e2fsck 1.41.12 (17-May-2010)
/dev/sdb4 was not cleanly unmounted, 强制检查.
Resize inode not valid. 重建<y>? 是
```

第一步：检查inode,块,和大小

第二步：检查目录结构

第3步：检查目录连接性

Pass 4: Checking reference counts

第5步：检查簇概要信息

Free 块s count wrong for 簇 #0 (7854, counted=7855).

处理<y>? 是

Free 块s count wrong (15722, counted=15723).

处理<y>? 是

/dev/sdb4: ***** 文件系统已修改 *****

/dev/sdb4: 11/4096 files (0.0% non-contiguous), 661/16384 blocks

```
[root@localhost ~]$
```

ftp - 用来登录远程ftp服务器

ftp指令可以用来登录远程ftp服务器。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ftp [选项] [host]
```

sh

选项

-A	# 活动模式，对于不支持密码连接的服务器，可以用来传输文件
-P	# 被动模式，允许在有防火墙的服务器中使用
-i	# 关闭互动模式
-n	# 不使用自动登录
-g	# 关闭本地主机文件名称支持特殊字符的扩展性
-v	# 显示详细过程
-d	# 使能调试

sh

举例

登录ftp服务器

```
[root@localhost ~]$ ftp 192.168.1.8          #登录
Connected to 192.168.1.8 (192.168.1.8).
220 (vsFTPd 2.2.2)
Name (192.168.1.8:root): ftp           #用户名
331 Please specify the password.
Password:                      #密码
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls                          #查看内容
227 Entering Passive Mode (192,168,1,8,44,30).
150 Here comes the directory listing.
drwxr-xr-x  2 0      0          4096 Aug 14 06:38 pub
226 Directory send OK.
ftp>
```

sh

ftpcount - 显示目前已FTP登入的用户人数

显示目前已ftp登入的用户人数。执行这项指令可得知目前用FTP登入系统的人数以及FTP登入人数的上限。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ftpcount
```

sh

ftpsht - 在指定的时间关闭FTP服务器

功能说明：在指定的时间关闭ftp服务器。本指令提供系统管理者在设置的时间关闭FTP服务器，且能在关闭之前发出警告信息通知用户。关闭时间若设置后为"none"，则会马上关闭服务器。如果采用"+30"的方式来设置表示服务器在30分钟之后关闭。依次类推，假设使用"1130"的格式则代表服务器会在每日的11时30分关闭，时间格式为24 小时制。FTP服务器关闭后，在/etc目录下会产生一个名称为shutmsg的文件，把它删除后即可再度启动FTP服务器的功能。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ftpsht [-d<分钟>][-l<分钟>][关闭时间]["警告信息"]
```

sh

选项

```
-d<分钟>    # 切断所有FTP连线时间。  
-l<分钟>    # 停止接受FTP登入的时间。
```

sh

ftpwho - 显示当前每个ftp会话信息

ftpwho命令 ftp服务器套件proftpd的工作指令，用于显示当前每个ftp会话信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
ftpwho [OPTION]
```

sh

选项

```
-h    # 显示帮助信息;  
-v    # 详细模式，输出更多信息。
```

sh

get - 登录ftp服务器之后从服务器获取文件

使用lftp登录ftp服务器之后，可以使用get指令从服务器获取文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
get [-E] [-a] [-c] [-O base] rfile [-o lfile]
```

sh

选项

```
-o      # 指定输出文件的名字，不指定则使用原来的名字  
-c      # 如果失败，持续获取  
-E      # 获取之后，删除源文件  
-a      # 使用ascii模式  
-O      # 指定输出文件存放的目录
```

sh

举例

获取文件，指定存储的名字

```
[root@localhost ~]$ lftp 192.168.1.8          #登录ftp服务器  
lftp 192.168.1.8:> ls  
drwxr-xr-x    2 0          0          4096 Aug 14 06:38 pub  
lftp 192.168.1.8:/> cd pub                  #切换目录  
lftp 192.168.1.8:/pub> ls                  #查看文件  
-rwxrwxrwx    1 0          0          2375494044 Aug 14 06:38 1.zip  
-rw-r--r--    1 0          0          0 Aug 14 03:38 test.c  
lftp 192.168.1.8:/pub> get test.c -o testtt.c  #获取文件，存储名字为testtt.c  
lftp 192.168.1.8:/pub> quit                #退出  
[root@localhost ~]$ ls                      #查看内容，已经获取到文件。文件存储在当前目录  
1   11.c  1.zip  2.c.bz2  4.c  6.c~  rec000012.c.bz2  testtt.c  
1.  1.c   2.c    3.c     5.c   col    res.zip
```

sh

获取文件，指定存储位置

sh

```
[root@localhost ~]$ lftp 192.168.1.8          #登录服务器
lftp 192.168.1.8:~> cd pub/                 #切换目录
lftp 192.168.1.8:/pub> get -O / test.c      #获取文件，指定存储位置到根目录
lftp 192.168.1.8:/pub> quit                  #退出
[root@localhost ~]$ ls /                      #查看更目录，已经得到文件
bak    dev    lib     misc   opt    sbin    sys    usr    wj
bin    etc    lost+found  mnt   proc   selinux test.c  var
boot   home   media    net    root   srv    tmp    admin
```

gpasswd - Linux下工作组文件的管理工具

gpasswd指令用来管理组文件"/etc/group"和"/etc/gshadow"，每个组可以设置管理员、组员、密码。系统管理员可以使用-A选项定义组管理员，使用-M选项定义成员。他们拥有组管理员和成员的所有权利。由具有组名的组管理员调用的gpasswd只提示输入组的新密码。如果设置了密码，则成员仍然可以在没有密码的情况下使用newgrp(1)，而非成员必须提供密码。

组密码是一个固有的安全问题，因为允许多个人知道密码。然而，群组是允许不同用户之间合作的有用工具。

警告，此工具仅对"/etc/group"和"/etc/gshadow"文件进行操作。因此，您不能更改任何NIS或LDAP组。这必须在相应的服务器上执行。

gpasswd命令 是Linux下工作组文件/etc/group和/etc/gshadow管理工具。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
gpasswd [选项] group
```

sh

选项

```
-a, --add user          # 向组中添加用户
-d, --delete user       # 删除成员
-r, --remove-password   # 删除组密码
-R, --restrict           # 限制对命名组的访问。只允许组成员使用newgrp加入命名组。
-M, --members user,...    # 向组添加多个成员
-A, --administrators user,.. # 设置组管理员

--help                  # 显示帮助文档
--version                # 显示命令版本信息
```

sh

配置

下面"/etc/login.defs"中的配置变量更改了该工具的行为：

ENCRYPT_METHOD (string) , 这定义了用于加密密码的系统默认加密算法(如果命令行上没有指定算法)。可用的算法有：DES（默认）、MD5、SHA256、SHA512。注意：此参数重写MD5_CRYPT_ENAB变量。

MAX_MEMBERS_PER_GROUP (number) , 每个组条目的最大成员数。达到最大值时，在/etc/group中启动一个新的组条目(行)(具有相同的名称、相同的密码和相同的GID)。默认值为0，这意味着组中的成员

数没有限制。此功能(拆分组)允许限制组文件中的行长度。这对于确保NIS组的行不大于1024个字符非常有用。如果你需要执行这样的限制，你可以使用25。注意：拆分组可能不支持所有的工具(即使在阴影工具集中)。除非你真的需要这个变量，否则你不应该使用它。

MD5_CRYPT_ENAB (boolean)，指示是否必须使用基于MD5的算法加密密码。如果设置为“是”，新密码将使用基于MD5的算法进行加密，该算法与FreeBSD最新版本使用的算法兼容。它支持无限长度的密码和更长的盐字符串。如果您需要将加密密码复制到其他不了解新算法的系统，则设置为“否”。默认为否。该变量由ENCRYPT_METHOD变量或用于配置加密算法的任何命令行选项取代。不推荐这个变量。您应该使用ENCRYPT_METHOD。

SHA_CRYPT_MIN_ROUNDS (number)、**SHA_CRYPT_MAX_ROUNDS** (number)，当Encrypt_Method设置为SHA 256或SHA 512时，默认情况下这将定义加密算法使用的SHA轮数(当命令行上未指定轮数时)。有很多回合，这是比较困难的暴力强制密码。但是还要注意的是，需要更多的CPU资源来对用户进行身份验证。如果没有指定，libc将选择默认的回合数(5000)。数值必须在1000-999999999范围内。如果只设置了SHA_CRYPT_MIN_ROUNDS或SHA_CRYPT_MAX_ROUNDS中的一个，则将使用此值。如果SHA_CRYPT_MIN_ROUNDS>SHA_CRYPT_MAX_ROUNDS，则将使用最高值。

相关文件

- `/etc/group`，组账户信息。
- `/etc/gshadow`，安全组账户信息。

用法

如系统有个peter账户，该账户本身不是groupname群组的成员，使用newgrp需要输入密码即可。

```
gpasswd groupname
```

sh

让使用者暂时加入成为该组成员，之后peter建立的文件group也会是groupname。所以该方式可以暂时让peter建立文件时使用其他的组，而不是peter本身所在的组。

所以使用gpasswd groupname设定密码，就是让知道该群组密码的人可以暂时切换具备groupname群组功能的。

```
gpasswd -A peter users
```

sh

这样peter就是users群组的管理员，就可以执行下面的操作：

```
gpasswd -a mary users  
gpasswd -a allen users
```

sh

注意：添加用户到某一个组 可以使用usermod -G group_name user_name这个命令可以添加一个用户到指定的组，但是以前添加的组就会清空掉。

所以想要添加一个用户到一个组，同时保留以前添加的组时，请使用gpasswd这个命令来添加操作用户：

```
gpasswd -a user_name group_name
```

sh

举例

将用户 `usr01` 添加到组 `sogrey`

```
[sogrey@bogon 文档]$ gpasswd -a usr01 sogrey #向sogrey组添加用户usr01
Adding user usr01 to group sogrey
[sogrey@bogon 文档]$ cat /etc/group      #查看组信息，已经成功添加
sogrey:123456:1000:usr01
```

sh

设置组管理员

```
[sogrey@bogon 文档]$ sudo gpasswd -A sogrey sogrey #设置管理员
[sogrey@bogon 文档]$ sudo cat /etc/gshadow          #查看组信息，管理员已经设置
sogrey:!:1000:sogrey
```

sh

grep - 强大的文本搜索工具

grep (global search regular expression(RE) and print out the line, 全面搜索正则表达式并把行打印出来) 是一种强大的文本搜索工具, 它能使用正则表达式搜索文本, 并把匹配的行打印出来。用于过滤/搜索的特定字符。可使用正则表达式能配合多种命令使用, 使用上十分灵活。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
grep [选项] PATTERN files  
grep [OPTIONS] [-e PATTERN | -f FILE] [FILE...]
```

sh

选项

```
-a --text # 不要忽略二进制数据。  
-A <显示行数> --after-context=<显示行数> # 除了显示符合范本样式的那一行之外, 并显示该行之后的内容。  
-b --byte-offset # 在显示符合范本样式的那一行之外, 并显示该行之前的内容。  
-B <显示行数> --before-context=<显示行数> # 除了显示符合样式的那一行之外, 并显示该行之前的内容。  
-c --count # 计算符合范本样式的列数。  
-C <显示行数> --context=<显示行数>或-<显示行数> # 除了显示符合范本样式的那一列之外, 并显示该列之前后的内容。  
-d <进行动作> --directories=<动作> # 当指定要查找的是目录而非文件时, 必须使用这项参数, 否则grep命令将回报  
-e <范本样式> --regexp=<范本样式> # 指定字符串作为查找文件内容的范本样式。  
-E --extended-regexp # 将范本样式为延伸的普通表示法来使用, 意味着使用能使用扩展正则表达式。  
-f <范本文件> --file=<规则文件> # 指定范本文件, 其内容有一个或多个范本样式, 让grep查找符合范本条件的文件。  
-F --fixed-regexp # 将范本样式视为固定字符串的列表。  
-G --basic-regexp # 将范本样式视为普通的表示法来使用。  
-h --no-filename # 在显示符合范本样式的那一列之前, 不标示该列所属的文件名称。  
-H --with-filename # 在显示符合范本样式的那一列之前, 标示该列的文件名称。  
-i --ignore-case # 忽略字符大小写的差别。  
-l --file-with-matches # 列出文件内容符合指定的范本样式的文件名称。  
-L --files-without-match # 列出文件内容不符合指定的范本样式的文件名称。  
-n --line-number # 在显示符合范本样式的那一列之前, 标示出该列的编号。  
-P --perl-regexp # PATTERN 是一个 Perl 正则表达式  
-q --quiet或--silent # 不显示任何信息。  
-R/-r --recursive # 此参数的效果和指定“-d recurse”参数相同。  
-s --no-messages # 不显示错误信息。  
-v --revert-match # 反转查找。  
-V --version # 显示版本信息。  
-w --word-regexp # 只显示全字符符合的列。  
-x --line-regexp # 只显示全列符合的列。  
-y # 此参数效果跟“-i”相同。  
-o # 只输出文件中匹配到的部分。  
-m <num> --max-count=<num> # 找到num行结果后停止查找, 用来限制匹配行数
```

sh

规则表达式

```
^      # 锚定行的开始 如: '^grep' 匹配所有以grep开头的行。  
$      # 锚定行的结束 如: 'grep$' 匹配所有以grep结尾的行。  
.     # 匹配一个非换行符的字符 如: 'gr.p' 匹配gr后接一个任意字符, 然后是p。  
*     # 匹配零个或多个先前字符 如: '*grep' 匹配所有一个或多个空格后紧跟grep的行。  
./* # 一起用代表任意字符。  
[]    # 匹配一个指定范围内的字符, 如'[Gg]rep' 匹配Grep和grep。  
[^]   # 匹配一个不在指定范围内的字符, 如: '[^A-FH-Z]rep' 匹配不包含A-R和T-Z的一个字母开头, 紧跟rep的行。  
\(..\)\# 标记匹配字符, 如'\(love\)', love被标记为1。  
\<    # 锚定单词的开始, 如: '\<grep' 匹配包含以grep开头的单词的行。  
\>    # 锚定单词的结束, 如'grep\>' 匹配包含以grep结尾的单词的行。  
x\{m\}\# 重复字符x, m次, 如: 'o\{5\}' 匹配包含5个o的行。  
x\{m,\}\# 重复字符x, 至少m次, 如: 'o\{5,\}' 匹配至少有5个o的行。  
x\{m,n\}\# 重复字符x, 至少m次, 不多于n次, 如: 'o\{5,10\}' 匹配5--10个o的行。  
\w    # 匹配文字和数字字符, 也就是[A-Za-z0-9], 如: 'G\w*p' 匹配以G后跟零个或多个文字或数字字符, 然后是p。  
\W    # \w的反置形式, 匹配一个或多个非单词字符, 如点号句号等。  
\b    # 单词锁定符, 如: '\bgrep\b' 只匹配grep。
```

常见用法

在文件中搜索一个单词, 命令会返回一个包含 "match_pattern" 的文本行:

```
grep match_pattern file_name  
grep "match_pattern" file_name
```

sh

在多个文件中查找:

```
grep "match_pattern" file_1 file_2 file_3 ...
```

sh

输出除之外的所有行 `-v` 选项:

```
grep -v "match_pattern" file_name
```

sh

标记匹配颜色 `--color=auto` 选项:

```
grep "match_pattern" file_name --color=auto
```

sh

使用正则表达式 `-E` 选项:

```
grep -E "[1-9]+"
```

或

```
egrep "[1-9]+"
```

sh

使用正则表达式 `-P` 选项:

```
grep -P "(\d{3}\-){2}\d{4}" file_name
```

sh

只输出文件中匹配到的部分 `-o` 选项:

```
echo this is a test line. | grep -o -E "[a-z]+\."  
line.  
  
echo this is a test line. | egrep -o "[a-z]+\."  
line.
```

sh

统计文件或者文本中包含匹配字符串的行数 `-c` 选项:

```
grep -c "text" file_name
```

sh

输出包含匹配字符串的行数 `-n` 选项:

```
grep "text" -n file_name  
# 或  
cat file_name | grep "text" -n
```

sh

多个文件

```
grep "text" -n file_1 file_2
```

sh

打印样式匹配所位于的字符或字节偏移:

```
echo gun is not unix | grep -b -o "not"  
7:not  
# 一行中字符串的字符便宜是从该行的第一个字符开始计算, 起始值为0。选项 ***-b -o*** 一般总是配合使用。
```

sh

搜索多个文件并查找匹配文本在哪些文件中:

```
grep -l "text" file1 file2 file3...
```

sh

grep递归搜索文件 在多级目录中对文本进行递归搜索:

```
grep "text" . -r -n
```

sh

.表示当前目录

忽略匹配样式中的字符大小写:

```
echo "hello world" | grep -i "HELLO"  
# hello
```

sh

选项 `-e` 制动多个匹配样式:

```
echo this is a text line | grep -e "is" -e "line" -o  
is  
line  
  
#也可以使用 **-f** 选项来匹配多个样式，在样式文件中逐行写出需要匹配的字符。  
cat patfile  
aaa  
bbb
```

```
echo aaa bbb ccc ddd eee | grep -f patfile -o
```

在grep搜索结果中包括或者排除指定文件:

只在目录中所有的.php和.html文件中递归搜索字符"main()"

```
grep "main()" . -r --include *.{php,html}
```

在搜索结果中排除所有README文件

```
grep "main()" . -r --exclude "README"
```

在搜索结果中排除filelist文件列表里的文件

```
grep "main()" . -r --exclude-from filelist
```

使用0值字节后缀的grep与xargs:

测试文件

```
echo "aaa" > file1  
echo "bbb" > file2  
echo "aaa" > file3  
  
grep "aaa" file* -lZ | xargs -0 rm
```

执行后会删除file1和file3, grep输出用-Z选项来指定以0值字节作为终结符文件名 (\0), xargs -0 读取输入并用0值字节分隔。

grep静默输出

```
grep -q "test" filename  
# 不会输出任何信息, 如果命令运行成功返回0, 失败则返回非0值。一般用于条件测试。
```

打印出匹配文本之前或者之后的行:

显示匹配某个结果之后的3行, 使用 `-A` 选项

```
seq 10 | grep "5" -A 3
5
6
7
8
```

sh

显示匹配某个结果之前的3行，使用 **-B** 选项

```
seq 10 | grep "5" -B 3
2
3
4
5
```

sh

显示匹配某个结果的前三行和后三行，使用 **-C** 选项

```
seq 10 | grep "5" -C 3
2
3
4
5
6
7
8
```

sh

如果匹配结果有多个，会用“--”作为各匹配结果之间的分隔符

```
echo -e "a\nb\nc\na\nb\nc" | grep a -A 1
```

sh

环境变量

grep的行为受到以下环境变量的影响。

GREP_OPTIONS，此变量指定放置在任何显式选项前面的默认选项。选项规范由空格分隔。反斜杠转义下一个字符，因此它可以指定包含空格或反斜杠的选项。例如，如果GREP_OPTIONS是'--binary-files=without-match --directories=skip'，那么grep执行的时候就假设已经有了这两个选项。

GREP_COLOR，此变量指定用于突出显示匹配(非空)文本的颜色。

GREP_COLORS，指定用于突出显示输出的各个部分的颜色和其他属性。它的值是一个以冒号分隔的功能列表，默認為ms=01; 31:mc=01; 31:sl=: cx=: fn=35:ln=32:bn=32:se=36，省略了rv和ne布尔功能(即false)。支持的功能如下所示。

sl=，用于整个选定行的SGR子字符串(即-v命令行选项省略时的匹配行，或指定-v时不匹配的行)。但是，如果指定了布尔rv功能和-v命令行选项，则它将应用于上下文匹配行。默认值为空(即终端的默认颜色对)。

cx=，用于整个上下文行的SGR子字符串(即省略-v命令行选项时的非匹配行，或指定-v时的匹配行)。但是，如果指定了布尔RV功能和-v命令行选项，则它将适用于选定的非匹配行。默认值为空(即终端的默认颜色对)。

`rv`, 当指定`-v`命令行选项时, 逆转(掉期)`"sl=`和`"cx=`功能的布尔值。缺省值为`false`(即省略了功能)。

`mt=01; 31`, 用于匹配任何行中的非空文本的`sgr`子字符串。(这仅在省略`-v`命令行选项时使用。)当启动时, `sl=(`或`cx=`)能力的效果保持活跃。默认值是当前行背景上的粗体红色文本前景。

`ms=01; 31`, 用于匹配选定行中的非空文本的`sgr`子字符串。(这仅在省略`-v`命令行选项时使用。)当启动时, `sl=(`或`cx=`)能力的效果保持活跃。默认值是当前行背景上的粗体红色文本前景。

`mc=01; 31`, 用于匹配上下文行中的非空文本的`sgr`子字符串。(这仅在省略`-v`命令行选项时使用。)当启动时, `sl=(`或`cx=`)能力的效果保持活跃。默认值是当前行背景上的粗体红色文本前景。

`fn=35`, 用于任何内容行前缀的文件名的`SGR`子字符串。默认值是终端默认背景上的洋红色文本前景。

`ln=32`, 任何内容行前缀的行号的`SGR`子字符串。默认值是终端默认背景上的绿色文本前景。

`bn=32`, 用于任何内容行前缀的字节偏移的`SGR`子字符串。默认值是终端默认背景上的绿色文本前景。

`se=36`, 当指定了非零上下文`(--)`, `SGR`子字符串用于在选定的行字段`(:)`、上下文行字段之间`(-)`和相邻行组之间插入分隔符。默认值是终端默认背景上的青色文本前景。

`ne`, 布尔值, 该值防止在每次彩色项结束时使用擦除入行`(EL)`对右`(\33[K]`清除到行尾的值。这是在不支持`EL`的终端上需要的。对于没有应用`Back_COLOR_ERASE(BCE)`布尔终止功能的终端、所选择的高亮颜色不影响背景、或者当`EL`太慢或导致过多闪烁时, 它在其他情况下是有用的。默认值为`false`(即省略功能)

`LC_ALL, LC_COLLATE, LANG`, 这些变量指定`LC_COLLATE`类别的区域设置, 该类别确定用于解释范围表达式(如`[a-z]`)的排序序列。

`LC_ALL, LC_CTYPE, LANG`, 这些变量指定`LC_CTYPE`类别的区域设置, 它决定字符的类型, 例如, 哪些字符是空格。

`LC_ALL, LC_MESSAGES, LANG`, 这些变量指定`LC_MESSAGES`类别的区域设置, 它确定`grep`用于消息的语言。默认的C语言环境使用美式英语消息。

`POSIXLY_CORRECT`, 如果设置, `grep`的行为与POSIX.2所要求的一样; 否则, `grep`的行为更像其他GNU程序。POSIX.2要求必须将文件名后面的选项视为文件名; 默认情况下, 这些选项被排到操作数列表的前面, 并被视为选项。此外, POSIX.2还要求将未被承认的选项诊断为“非法”, 但由于它们并不真正违反法律, 默认情况是将它们诊断为“无效”。`POSIXLY_RIDER`还禁用`_N_GNU_NOOPTION_ARGV_LANGS_`, 如下所述。

`N_GNU_nonoption_argv_flags`, (这里N是`grep`的数字进程ID。)如果此环境变量值的ith字符为1, 则不要将`grep`的ith操作数视为选项, 即使它似乎是选项之一。`shell`可以为其运行的每个命令在环境中放置此变量, 指定哪些操作数是文件名通配符展开的结果, 因此不应被视为选项。此行为仅在GNU C库中可用, 且仅在未设置`POSIXLY_RIDER`时才可用。

退出码

通常, 如果找到选定的行, 则退出状态为0, 否则为1。但是, 如果发生错误, 退出状态为2, 除非使用`-q`、`--quite`、`--slient`选项, 并找到选定的行。但是, 请注意, 对于`grep`、`CMP`和`diff`等程序, POSIX只要求在出现错误时的退出状态大于1; 因此, 出于可移植性的考虑, 建议使用对此一般条件进行测试的逻辑, 而不是与2严格相等的逻辑。

举例

```
[sogrey@bogon newDir3]$ cat test2.txt  
123  
23  
212  
[sogrey@bogon newDir3]$ grep ^2 test2.txt # 显示以2开头的行  
23  
212  
[sogrey@bogon newDir3]$ grep -v 23 test2.txt # 显示不包含23的行  
212  
[sogrey@bogon newDir3]$ grep -w 23 test2.txt # 显示整个字都匹配的行  
23  
[sogrey@bogon newDir3]$
```

sh

groupadd - 用于创建一个新的工作组

groupadd命令 用于创建一个新的工作组，新工作组的信息将被添加到系统文件中。

指定群组名称来建立新的群组账号，需要时可以从系统中取得新的群组值。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
groupadd [选项] group
```

sh

选项

```
-g gid          # 指定组id  
-r             # 创建系统组  
-f             # 新增一个已经存在的群组帐号，系统会出现错误讯息然后结束groupadd。如果是这  
--help          # 显示帮助文档  
--version       # 显示命令版本信息
```

sh

相关文件

- `/etc/group` 群组信息。
- `/etc/gshadow` 群组加密信息。

举例

```
[sogrey@bogon ~]$ sudo groupadd groupTest -g 1996 # 创建组，指定id  
[sogrey@bogon ~]$ tail -n 2 /etc/group # 查看组信息  
sogrey:123456:1000:sogrey  
groupTest:x:1996:  
[sogrey@bogon ~]$
```

sh

groupdel - 用于删除指定的工作组

groupdel命令 用于删除指定的工作组，本命令要修改的系统文件包括/etc/group和/etc/gshadow。若该群组中仍包括某些用户，则必须先删除这些用户后，方能删除群组。

删除组，如果组内有用户，那么必须先删除用户。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
groupdel group
```

sh

选项

```
--help          # 显示帮助文档  
--version      # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon ~]$ tail -n 2 /etc/group # 查看组信息 存在groupTest组  
sogrey:123456:1000:sogrey  
groupTest:x:1996:  
[sogrey@bogon ~]$ sudo groupdel groupTest # 删除组  
[sogrey@bogon ~]$ tail -n 2 /etc/group # 查看组信息，已经删除  
vboxsf:l:983:  
sogrey:123456:1000:sogrey  
[sogrey@bogon ~]$
```

sh

groupmod - 更改群组识别码或名称

groupmod命令 更改群组识别码或名称。需要更改群组的识别码或名称时，可用groupmod指令来完成这项工作。

修改组的基本信息，包括组名称、组ID等信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
groupmod [选项] group
```

sh

选项

-g gid	# 指定组id
-n name	# 指定组名
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

```
[sogrey@bogon 文档]$ tail -n 2 /etc/group # 查看组信息
sogrey:123456:1000:sogrey
vboxsf:!:983:
[sogrey@bogon 文档]$ groupmod -g 955 -n sogrey01 sogrey
bash: /usr/sbin/groupmod: 权限不够
[sogrey@bogon 文档]$ sudo groupmod -g 955 -n sogrey01 sogrey #修改组id: 955, 组名字: sogrey01
[sudo] sogrey 的密码:
[sogrey@bogon 文档]$ tail -n 2 /etc/group
vboxsf:!:983:
sogrey01:123456:955:sogrey #查看组信息, 已经修改
[sogrey@bogon 文档]$
```

sh

groups - 打印指定用户所在组的名称

打印指定用户所在组的名称。

groups指令可以查看用户所属的组。如果未指定用户名，则打印当前进程的组成员资格。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
groups [OPTION]... [username]...
```

sh

- username（可选）：可以是一到多个，不提供时默认为当前用户。

选项

```
--help          # 显示帮助文档  
--version      # 显示命令版本信息
```

sh

返回值

返回0表示成功，返回非0值表示失败。

举例

显示sogrey用户所属的组

```
[sogrey@bogon ~]$ groups sogrey  
sogrey : sogrey wheel  
[sogrey@bogon ~]$
```

sh

注意

1. 该命令等价于 `id -Gn` 。
2. 每个用户属于 `/etc/passwd` 中指定的一个组和在 `/etc/group` 中指定的其他组。
3. 该命令是 `GNU coreutils` 包中的命令，相关的帮助信息请查看 `man -s 1 groups` , `info coreutils 'groups invocation'` 。

grpck - 用于验证组文件的完整性

grpck命令 用于验证组文件的完整性，在验证之前，需要先锁定（lock）组文件/etc/group和/etc/shadow。

grpck命令检查数据是否正确存放，每条记录是否都包含足够的信息，是否有一个唯一的组名，是否包含正确的用户，是否正确设置了组的管理员等。grpck检查发现错误以后，在命令行提示用户是否删除错误的记录。如果用户没有明确回答删除记录，grpck终止运行。

grpck指令可以验证组文件"/etc/group"和"/etc/gshadow"的完整性。检查的内容包括：正确的字段数、唯一有效的组名称、有效的组标识符、成员和管理员的有效列表、"/etc/gshadow"文件中的相应条目。检查正确的字段数和唯一的组名是致命的。如果条目有错误的字段数，则会提示用户删除整行。如果用户没有肯定地回答，所有进一步的检查都会被绕过。提示删除具有重复组名的条目，但仍将进行其余检查。所有其他错误都是警告，并鼓励用户运行groupmod命令来更正错误。

对"/etc/group"和"/etc/gshadow"文件进行操作的命令不能更改损坏或重复的条目。在这种情况下，应该使用grpck来删除违规条目。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
grpck [选项] group
```

sh

选项

-r	# 以只读模式运行
-s	# 使用gid对group和gshadow进行排序
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

配置

下面"/etc/login.defs"中的配置变量更改了该工具的行为：

MAX_MEMBERS_PER_GROUP (number)，每个组条目的最大成员数。达到最大值时，在"/etc/group"中启动一个新的组条目(行)(具有相同的名称、相同的密码和相同的GID)。默认值为0，这意味着组中的成员数没有限制。此功能(拆分组)允许限制组文件中的行长度。这对于确保NIS组的行不大于1024个字符非常有用。如果你需要执行这样的限制，你可以使用25。注意：拆分组可能不支持所有的工具(即使在阴影工具集中)。除非您真的需要这个变量，否则不应该使用它。

相关文件

- `/etc/group` 组账户信息。
- `/etc/gshadow` 安全组账户信息。
- `/etc/passwd` 用户账户信息。

举例

检查组信息

```
[sogrey@bogon ~]$ sudo grpck # 检查组信息, 返回0, 没有任何错误
[sudo] sogrey 的密码:
[sogrey@bogon ~]$ echo $?
0
[sogrey@bogon ~]$
```

对/etc/group排序

```
[sogrey@bogon ~]$ tail /etc/group # 查看组信息
pulse-access:::986:
pulse-rt:::985:
pulse:::171:
gdm:::42:
gnome-initial-setup:::984:
sshd:::74:
avahi:::70:
slocate:::21:
vboxsf:::983:
sogrey:123456:1000:sogrey
[sogrey@bogon ~]$ sudo grpck -s /etc/group # 按照gid排序
[sogrey@bogon ~]$ tail /etc/group # 查看信息, 已经排序
geoclue:::992:
ssh_keys:::993:
cgred:::994:
colord:::995:
libstoragemgmt:::996:
printadmin:::997:
polkitd:::998:
input:::999:
sogrey:123456:1000:sogrey
nfsnobody:::65534:
[sogrey@bogon ~]$
```

grpconv - 用来开启群组的投影密码

grpconv命令 用来开启群组的投影密码。Linux系统里的用户和群组密码，分别存放在/etc目录下的passwd和group文件中。因系统运作所需，任何人都得以读取它们，造成安全上的破绽。投影密码将文件内的密码改存在/etc目录下的shadow和gshadow文件内，只允许系统管理者读取，同时把原密码置换为"x"字符。投影密码的功能可随时开启或关闭，您只需执行grpconv指令就能开启群组投影密码。

grpconv可以开启群组的影子文件。Linux系统的用户和群组密码分别存在在"/etc/passwd"、"/etc/group"文件中。在系统运行的时候任何用户都可以读取它们，这样难免不安全。影子文件将口令文件的内容存在在"/etc/shadow"和"/etc/gshadow"中，只允许系统管理者读取。

grpconv命令从组和gshadow创建组，然后移除gshadow。

pwconv命令从passwd和一个可选的现有shadow创建shadow。

pwunconv命令从passwd和shadow创建passwd，然后删除shadow。

每个程序在转换前都会获得必要的锁。pwconv和grpconv相似。首先，删除主文件中不存在的阴影文件中的条目。然后，不以"x"作为主文件中的密码的阴影条目将被更新。添加任何缺失的阴影项。最后，主文件中的密码被替换为'x'。如果主文件是手工编辑的，这些程序也可以用于初始转换以更新阴影文件。

当向"/etc/shadow"添加新条目时，pwconv将使用"/etc/login.defs"文件中定义的PASS_MIN_DAYS、PASS_MAX_DAYS和PASS_WARN_AGE的值。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
grpconv
```

sh

实例

设置sogrey组密码

```
[sogrey@bogon ~]$ sudo groupmod --password 123456 sogrey
[sudo] sogrey 的密码:
[sogrey@bogon ~]$ sudo cat /etc/gshadow | grep sogrey # 查看sogrey组密码
wheel:::sogrey
sogrey:123456::sogrey
[sogrey@bogon ~]$
```

sh

启动影子系统

```
[sogrey@bogon ~]$ sudo grpconv
[sogrey@bogon ~]$ cat /etc/group | grep sogrey #看出密码段已经被x替代
wheel:x:10:sogrey
sogrey:x:1000:sogrey
[sogrey@bogon ~]$ sudo cat /etc/gshadow | grep sogrey # 已经移到影子文件了
wheel:::sogrey
sogrey:123456::sogrey
[sogrey@bogon ~]$
```

注：gshadow, shadow只有root权限才可以查看。

grpunconv - 用来关闭群组的投影密码

grpunconv命令 用来关闭群组的投影密码。它会把密码从gshadow文件内，回存到group文件里。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
grpunconv
```

sh

实例

未关闭的情况

```
[sogrey@bogon ~]$ sudo cat /etc/gshadow | grep sogrey
[sudo] sogrey 的密码:
wheel:::sogrey
sogrey:123456::sogrey
[sogrey@bogon ~]$
```

sh

关闭影子密码

```
[sogrey@bogon ~]$ sudo grpunconv
[sogrey@bogon ~]$ sudo cat /etc/gshadow
cat: /etc/gshadow: 没有那个文件或目录
[sogrey@bogon ~]$
```

sh

查看密码已经复制到/etc/group中了。

```
[sogrey@bogon ~]$ cat /etc/group | grep sogrey
wheel::10:sogrey
sogrey:123456:1000:sogrey
[sogrey@bogon ~]$
```

sh

grub - 多重引导程序grub的命令行shell工具

grub命令 是多重引导程序grub的命令行shell工具。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
grub [OPTION]
```

sh

选项

```
--batch          # 打开批处理模式;
--boot-drive=<驱动器>    # 指定stage2的引导驱动器;
--config-file<配置文件>    # 指定stage2的配置文件;
--device-map=<文件>      # 指定设备的映射文件;
--help            # 显示帮助信息;
--install-partition=<分区> # 指定stage2安装分区;
--no-config-file   # 不使用配置文件;
--no-pager        # 不使用内部分页器;
--preset-menu     # 使用预设菜单;
--probe-second-floppy # 检测第二个软盘驱动器;
--read-only       # 只读模式。
```

sh

实例

利用grub命令来启动损坏的Linux系统，可能你的电脑因为某些原因损坏不能自动启动了。当然原因很多，可能的现象也很多。

这里说一下这种情况下的处理方法，即：屏幕上提示 `grub>`，但你的硬盘上数据没有丢失，各分区都是好的。这种情况是你的grub信息损坏了，但比较严重的是系统启动不了。

当然，在正常启动情况下，屏幕上出现grub的启动项选择菜单时按 `c` 键也是可以进入 `grub>` 状态的。这时候我们需要用grub的命令来手工启动系统。

只需要用到四个命令`boot`、`kernel`、`initrd`、`boot`。

但grub本身命令很多，比如查看文件内容的`cat`，你输入`help`会得到。

首先，输入“`root (hd`”，然后按两次TAB键；/* 这会列出你电脑上可能的磁盘设备，硬盘为 `hd0/hd1` 或 `sd0/sd1` 等 */

然后，选择你的安装 Linux 系统的硬盘，比如 `hd0`，输入 “`root (hd0,`” 再按两次TAB键；/* 这会列出你的第一块硬盘上的分区情况，你会知道哪个是 swap 交换分区，`0x82`，哪个是 Linux 分区 `0x83` */

选择你认为可能的 /boot 目录所在的分区，输入 `root (hd0, 1)` 回车；

接着，输入 `cat /boot/vmlinuz`，按两次 TAB 键，如果出现一些 `vm` 开头的文件，比如 `vmlinuz-2.6.15-26-386` 说明这里是 /boot 所在的分区。

删除上一次的输入，再输入 `cat /boot/initrd`，按两次 TAB 键，如果出现一些 `initrd` 开头的文件，比如 `initrd.img-2.6.15-26-386` 说明这个 /boot 所在的分区有 `initrd`，即 ramdisk 镜像；

删除上一次的输入，再输入 `cat /sbin/init`，按两次 TAB 键，如果出现一些 `init` 开头的文件，比如 `/sbin/init` 说明这个分区是 / 所在的分区；

如果没有出现 `/sbin/init` 文件，说明 `(hd0,1)` 分区仅仅是 `/boot` 分区而不是 / 分区。重新输入 `root (hd0,N)` 命令，这里 N 是某个 Linux 分区，然后再试 `cat /sbin/init`，直到屏幕上出现 `/sbin/init`，说明你找到了 / 分区，严格来说，应该是 `/sbin` 目录所在的分区；

依次输入命令：

```
root (hd0,1) /* 假设 /dev/hda2 是你的 /boot 所在的分区 */
kernel /boot/vmlinuz-2.6.15-26-386 ro dev=/dev/hda3 /* 假设 /dev/hda3 是你的 / 所在的分区 */
initrd /boot/initrd.img-2.6.15-26-386
boot
```

即可启动系统。

这里的关键问题是如何确定系统的几个分区： `/boot` / `/sbin`

gunzip - 用来解压缩文件

gunzip命令 用来解压缩文件。gunzip是个使用广泛的解压缩程序，它用于解开被gzip压缩过的文件，这些压缩文件预设最后的扩展名为.gz。事实上gunzip就是gzip的硬连接，因此不论是压缩或解压缩，都可通过gzip指令单独完成。

解压缩被gzip压缩过的文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux		Arch Linux

语法

```
gunzip [-acfhlLnNrtvV] [-S suffix] 文件
```

sh

选项

-f, --force	# 强制执行
-a, --ascii	# 文本模式。此选项仅在某些非Unix系统上支持。
-c, --stdout, --to-stdout	# 将解压的文件写到标准输出，源文件不变。如果有多个输入文件，则输出由一系列文件组成
-l, --list	# 列出压缩文件的信息
-L, --licence	# 列出gzip的许可证并且退出
-n, --no-name	# 解压缩时，如果存在，不要还原原始文件名(仅从压缩文件名中删除gzip后缀)，而是将文件名设为文件的内容
-N, --name	# 解压缩时，如果存在，请还原原始文件名和时间戳。此选项对于限制文件名长度很有用
-r, --recursive	# 递归遍历目录结构。如果命令行中指定的任何文件名都是目录，则gzip将下降到该目录并递归地处理所有子目录
-t, --test	# 测试压缩文件完整性
-v, --verbose	# 显示详细执行过程
-S	# 解压缩时，在从输入文件名派生输出文件名时，将.suf添加到后缀列表的开头以生成新的输出文件名
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

解压

```
[sogrey@bogon 文档]$ gunzip -v 1.gz #解压，显示详细执行过程
1.gz:   9.4% -- replaced with 1
```

sh

指定文件后缀

sh

```
[sogrey@bogon 文档]$ gunzip -v 1.mygz    # 解压, 后缀不是gz, 报错
gzip: 1.mygz: unknown suffix -- ignored
[sogrey@bogon 文档]$ gunzip -v -S "mygz" 1.mygz  # 指定后缀名, 不报错
1.mygz:      9.4% -- replaced with 1.
```

gzexe - 用来压缩可执行文件

gzexe命令 用来压缩可执行文件，压缩后的文件仍然为可执行文件，在执行时进行自动解压缩。当您去执行被压缩过的执行文件时，该文件会自动解压然后继续执行，和使用一般的执行文件相同。这个命令也可以看成是gunzip命令的一个扩展。

压缩可执行文件，在执行程序的时候可以自动实现解压。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
gzexe file
```

sh

选项

-d	# 解压
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

压缩ls指令

```
[sogrey@bogon 文档]$ ll  
总用量 12  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$ gzexe run.sh # 压缩，原来的指令文件变为run.sh~  
run.sh: 7.8%  
[sogrey@bogon 文档]$ ll  
总用量 16  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 909 3月 9 00:19 run.sh  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh~  
[sogrey@bogon 文档]$
```

sh

解压ls指令

sh

```
[sogrey@bogon 文档]$ ll  
总用量 16  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 909 3月 9 00:19 run.sh  
[sogrey@bogon 文档]$ gzxex -d run.sh # 解压  
[sogrey@bogon 文档]$ ll # 解压之后, 原来的压缩包变为run.sh~  
总用量 20  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 9 00:23 run.sh  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh~  
[sogrey@bogon 文档]$
```

gzip - 用来压缩文件

gzip命令 用来压缩文件。 gzip是个使用广泛的压缩程序，文件经它压缩过后，其名称后面会多处".gz"扩展名。

gzip是在Linux系统中经常使用的一个对文件进行压缩和解压缩的命令，既方便又好用。 gzip不仅可以用来压缩大的、较少使用的文件以节省磁盘空间，还可以和tar命令一起构成Linux操作系统中比较流行的压缩文件格式。据统计， gzip命令对文本文件有60%~70%的压缩率。减少文件大小有两个明显的好处，一是可以减少存储空间，二是通过网络传输文件时，可以减少传输的时间。

gzip通过Lempel-ziv算法来压缩文件，压缩的时候保留每个文件的所有者、权限、修改时间。对于符号链接， gzip将会忽略它。

如果压缩的文件名对其文件系统来说太长，则gzip将截断它。 Gzip试图只截断文件名中超过3个字符的部分。(部分由点分隔。)如果名称仅由小部件组成，最长的部分将被截断。例如，如果文件名限制为14个字符，则" gzip.msdos.exe"压缩为" gzi.msd.exe.gz"。在没有文件名长度限制的系统中，名称不会被截断。

默认情况下， gzip将原始文件名和时间戳保存在压缩文件中。这些在使用"-N"选项解压缩文件时使用。当压缩文件名被截断或文件传输后没有保留时间戳时，这是非常有用的。压缩文件可以使用" gzip -d"或"gunzip"或"zcat"恢复到它们的原始形式。如果保存在压缩文件中的原始名称不适合其文件系统，则从原始文件中构造新名称以使其合法。

gunzip在其命令行中获取一个文件列表，并替换其名称以.gz、 -z、 -z、 _z或.z结尾的每个文件，该文件以正确的魔术号开头，文件的未压缩文件没有原来的扩展名。 gunzip还将特殊的扩展名.tgz和.taz分别识别为.tar.gz和.tar.z的缩写。压缩时， gzip在必要时使用.tgz扩展名，而不是截断扩展名为.tar的文件。

gunzip目前可以解压缩由gzip， zip， compress-H或Pack创建的文件。输入格式的检测是自动的。当使用前两种格式时， gunzip检查32位CRC。对于包，枪拉链检查未压缩长度。标准压缩格式的设计不是为了允许一致性检查。然而， gunzip有时能够检测到一个坏的.z文件。如果在解压缩.z文件时出现错误，请不要仅仅因为标准解压缩不抱怨而认为.Z文件是正确的。这通常意味着标准解压缩不检查其输入，并愉快地生成垃圾输出。上海合作组织压缩-H格式(lzh压缩方法)不包括一个CRC，但也允许一些一致性检查。

由zip创建的文件只有在使用"通缩"方法压缩单个成员的情况下才能被gzip解压缩。此特性仅用于帮助将tar.zip文件转换为tar.gz格式。要使用单个成员提取zip文件，可以使用诸如"gunzip < foo.zip"或"gunzip -S .zip foo.zip"之类的命令。要提取包含多个成员的zip文件，请使用解压缩而不是gunzip。

Zcat和"gunzip -c"是一样的。(在某些系统上，可以将zcat安装为gzcat，以保留原始链接以进行压缩。)zcat解压缩命令行上的文件列表或其标准输入，并将未压缩的数据写入标准输出。无论是否有.gz后缀， zcat都会解压缩具有正确魔术号的文件。

Gzip使用用于zip和PKZIP的Lempel-Ziv算法。获得的压缩量取决于输入的大小和常用子字符串的分布。通常，像源代码或英语这样的文本会减少60-70%.压缩通常比LZW(用于压缩)、 Huffman编码(用于Pack)或自适应Huffman编码(紧凑)要好得多。

压缩总是被执行，即使压缩文件比原始文件稍大。最坏的情况是gzip文件头的几个字节，加上每32K块5个字节，或者大文件的扩展率为0.015%。注意，使用过的磁盘块的实际数量几乎从未增加。 gzip在压缩或解压缩时保留文件的模式、所有权和时间戳。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
gzip [ -acdfhlLnNrtvV19 ] [-S suffix] [ name ... ]
```

sh

选项

```
-a, --ascii          # 文本模式, 只适用于某些系统
-c, --stdout, --to-stdout # 将解压文件写到标准输出, 源文件不变。如果有多个输入文件, 则输出由一系列独立的压缩文件
-d, --decompress, --uncompress # 解压
-f, --force          # 强制执行
-l, --list            # 对于每一个压缩的文件, 列出压缩文件大小、解压大小、压缩比列、压缩前的文件名
-L, --license         # 列出gzip的许可证
-n, --no-name        # 压缩文件的时候, 不保留原始文件名字和时间属性
-N, --name            # 压缩的时候, 保留原始文件和时间属性
-q, --quite           # 跳过所有的警告信息
-r, --recursive       # 递归压缩子目录
-S,.suf, --suffix, .suf # 指定压缩文件后缀。压缩时, 使用后缀.suf代替.gz。可以提供任何非空后缀, 但不能与--name一起使用
-t, --test             # 测试
-v, --verbose          # 显示执行过程
-num, --fast, --best   # 使用指定的数字num调整压缩速度, 其中-1或--fast表示最快的压缩方法(较小的数)
--help                # 显示帮助文档
--version              # 显示命令版本信息
```

sh

环境变量

环境变量GZIP可以保存gzip的一组默认选项。这些选项首先被解释, 可以被显式命令行参数覆盖。例如

```
for sh:  GZIP="-8v --name"; export GZIP
for csh:  setenv GZIP "-8v --name"
for MSDOS: set GZIP=-8v --name
```

在VAX/VMS上, 环境变量的名称为GZIP_OPT, 以避免与用于调用程序的符号设置冲突。

举例

```
[sogrey@bogon 文档]$ ll  
总用量 12  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$ cat file1.txt  
报道称，7日上午，近3000人身穿白衣，祈求上天降甘霖，其中台湾“农田水利署”台中管理处长陈荣福也出席参加。参加祈雨  
[sogrey@bogon 文档]$ cat file2.txt  
国民党“立委”赖士葆表示，干旱不雨，担心科技业没水用，“要凿井取水”、要办祈雨法会求助神明解决水荒，政府却毫无专业  
也有岛内网友讽刺民进党当局，“这个‘政府’竟然有办法搞到只剩乞求神明保佑这招了...怎么有办法无能至此...”↓  
[sogrey@bogon 文档]$ gzip -c file1.txt > FILE.gz # 压缩  
[sogrey@bogon 文档]$ gzip -c file2.txt >> FILE.gz # 可以连接多个压缩文件  
[sogrey@bogon 文档]$ gunzip -c FILE # 等价于cat file1 file2  
报道称，7日上午，近3000人身穿白衣，祈求上天降甘霖，其中台湾“农田水利署”台中管理处长陈荣福也出席参加。参加祈雨  
国民党“立委”赖士葆表示，干旱不雨，担心科技业没水用，“要凿井取水”、要办祈雨法会求助神明解决水荒，政府却毫无专业  
也有岛内网友讽刺民进党当局，“这个‘政府’竟然有办法搞到只剩乞求神明保佑这招了...怎么有办法无能至此...”↓  
[sogrey@bogon 文档]$ ll  
总用量 16  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rw----- 1 sogrey sogrey 782 3月 8 23:54 FILE.gz  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$ cat file1.txt file2.txt | gzip > FOO.gz # 通过一次压缩所有成员来获得更好的压缩  
[sogrey@bogon 文档]$ ll  
总用量 20  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rw----- 1 sogrey sogrey 782 3月 8 23:54 FILE.gz  
-rw----- 1 sogrey sogrey 682 3月 8 23:59 FOO.gz  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$
```

如果您想要重新压缩连接的文件以获得更好的压缩

```
[sogrey@bogon 文档]$ gzip -cd old.gz | gzip > new.gz # 想要重新压缩连接的文件以获得更好的压缩
```

如果需要所有成员的未压缩大小

```
[sogrey@bogon 文档]$ ll  
总用量 20  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rw----- 1 sogrey sogrey 782 3月 8 23:54 FILE.gz  
-rw----- 1 sogrey sogrey 682 3月 8 23:59 FOO.gz  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$ gzip -cd FOO.gz | wc -c  
991  
[sogrey@bogon 文档]$
```

如果希望创建一个包含多个成员的单个归档文件，以便以后可以独立提取成员，请使用一个归档程序(如tar或zip)。GNUtar支持-z选项来透明地调用gzip。gzip是作为tar的补充，而不是替代。

压缩文件

```
[sogrey@bogon 文档]$ ll  
总用量 12  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$ gzip file1.txt  
[sogrey@bogon 文档]$ ll  
总用量 12  
-rw----- 1 sogrey sogrey 337 3月 8 23:50 file1.txt.gz  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh
```

sh

解压

```
[sogrey@bogon 文档]$ gzip -ld file1.txt.gz # 解压文件  
      compressed      uncompressed   ratio   uncompressed_name  
            337                  415  25.5% file1.txt  
[sogrey@bogon 文档]$ gzip -dv * # 解压每个压缩的文件  
file1.txt.gz: 25.5% -- replaced with file1.txt  
gzip: file2.txt: unknown suffix -- ignored  
gzip: run.sh: unknown suffix -- ignored  
[sogrey@bogon 文档]$ ll  
总用量 12  
-rw----- 1 sogrey sogrey 415 3月 8 23:50 file1.txt  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rwx----- 1 sogrey sogrey 90 3月 7 12:08 run.sh  
[sogrey@bogon 文档]$
```

sh

其他

把test6目录下的每个文件压缩成.gz文件

```
gzip *
```

sh

把上例中每个压缩的文件解压，并列出详细的信息

```
gzip -dv *
```

sh

详细显示例1中每个压缩的文件的信息，并不解压

```
gzip -l *
```

sh

压缩一个tar备份文件，此时压缩文件的扩展名为.tar.gz

```
gzip -r log.tar
```

sh

递归的压缩目录

```
gzip -rv test6
```

sh

这样，所有test下面的文件都变成了*.gz，目录依然存在只是目录里面的文件相应变成了*.gz.这就是压缩，和打包不同。因为是对目录操作，所以需要加上-r选项，这样也可以对子目录进行递归了。

递归地解压目录

```
gzip -dr test6
```

sh

保留原始文件，把压缩/解压流重定向到新文件

```
gzip -c aa > aa.gz  
gzip -dc bb.gz > bb
```

sh

hash - 用来显示和清除指定运行时系统查询的hash表

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
hash [-lr] [-p filename] [-dt] [name]
```

sh

选项

-p #将具有完整路径的指令加入到hash表
-r #清除所有的hash表
-d #在hash表中删除某记录
-t #显示hash表中的完整记录
-l #显示hash表中的指令

sh

举例

显示hash表中的命令

```
[root@localhost ~]$ hash -l          #列出hash表中的命令  
builtin hash -p /bin/grep grep
```

sh

增加命令

```
[root@localhost ~]$ hash -p /usr/bin/tail tail      #将完整路径的命令加入  
[root@localhost ~]$ hash -l                      #列出hash表中命令，已经成功添加  
builtin hash -p /bin/grep grep  
builtin hash -p /usr/bin/tail tail
```

sh

head - 显示文件的开头部分

显示文件开头的几行， 默认显示10行， 可以使用选项-n来指定行数。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
head [选项] files
```

sh

选项

```
-c, --bytes=[-]K # 显示文件开头的n个字节; -n显示所有内容，但是不包含最后n字节  
-n, --lines=[-]K # 显示开头的k行; -k显示所有行，但是不包含最后k行  
-q, --quite, --silent # 不显示文件名  
-v, --verbose # 显示文件名  
  
--help # 显示帮助文档  
--version # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ cat 1.txt  
hello world,  
i love linux,  
love code.  
[sogrey@bogon newDir3]$ head -c 10 1.txt # 显示前10个字节  
hello worl  
[sogrey@bogon newDir3]$ head -n 2 1.txt # 使用-n选项指定显示前3行  
hello world,  
i love linux,  
[sogrey@bogon newDir3]$
```

sh

history - 显示或操作历史列表

history指令用来显示用户以前执行过的命令，也可以对历史命令进行追加和删除。

主要用途

- 显示历史列表。
- 操作历史列表。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
history [n]
history -c
history -d offset
history -anrw [filename]
history -p arg [arg ...]
history -s arg [arg ...]
```

sh

选项

```
-c      # 清空当前历史命令
-d      # 删除指定序号的历史命令
-a      # 追加新的历史命令
-n      # 从历史命令的文件中读取本次shell会话没有读取的命令
-r      # 读取历史命令文件到当前的历史命令缓冲区
-s      # 将指令作为单独的条目存储到历史命令缓冲区，并且在添加之前删除缓冲区中的最后一条命令
-w      # 将当前shell历史命令缓冲区写入到历史命令文件
```

sh

举例

显示当前历史命令

```
[root@localhost ~]$ history      #显示当前所有历史命令
...
1008 hash -p /usr/bin/lsusb
1009 hash -p /usr/bin/tail
1010 hash -l
1011 hash -p /usr/bin/tail tail
1012 hash -l
1013 history
```

删除指定的历史命令

```
[root@localhost ~]$ history      # 查看历史命令
992 hash -p /usr/bin/lsusb
993 hash -l
994 hash -p /usr/bin/tail
995 hash -l
996 hash -l
997 history
998 history -d 1008
999 history
1000 history -d 996
1001 history
[root@localhost ~]$ history -d 994      #删除指令的历史命令
[root@localhost ~]$ history      #查看历史命令, 已经删除
991 ls /usr/bin/
992 hash -p /usr/bin/lsusb
993 hash -l
994 hash -l
995 history
996 history -d 1008
997 history
998 history -d 996
999 history
```

注意

- 在命令行中，可以使用符号!执行指定序号的历史命令。例如，要执行第2个历史命令，则输入!2。
- 关闭终端后，历史列表将被写入历史文件~/bash_history。
- 环境变量HISTSIZE决定了历史文件中命令的存储数量，默认存储1000条。
- 环境变量HISTTIMEFORMAT如果是非空值，则使用其值作为strftime(3)打印相关时间戳的格式字符串添加在每个显示的历史记录之前；否则不会打印时间戳。
- 该命令是bash内建命令，相关的帮助信息请查看help命令。

host - 常用的分析域名查询工具

host命令 是常用的分析域名查询工具，可以用来测试域名系统工作是否正常。

host是一个常用的DNS查询工具，经常用来查询域名、检查域名解析是否正确。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
host [选项] name [server]
```

sh

选项

-a	# 查询所有的信息
-c	# 设置查询类型
-C	# 查询完整的SOA记录
-d, -v	# 显示详细过程
-l	# 列表模式
-t	# 选择查询类型: CNAME NS SOA SIG KEY AXFR
-w	# 永久等待
-W	# 设置等待超时

sh

举例

查询域名

```
[sogrey@bogon ~]$ host www.baidu.com
www.baidu.com is an alias for www.a.shifen.com.
www.a.shifen.com has address 14.215.177.38
www.a.shifen.com has address 14.215.177.39
www.a.shifen.com is an alias for www.wshifen.com.
www.a.shifen.com is an alias for www.wshifen.com.
[sogrey@bogon ~]$
```

sh

查询所有信息

sh

```
[sogrey@bogon ~]$ host -a www.baidu.com
Trying "www.baidu.com"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11677
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.baidu.com. IN ANY

;; ANSWER SECTION:
www.baidu.com. 0 IN A 14.215.177.38

Received 47 bytes from 192.168.0.1#53 in 12 ms
[sogrey@bogon ~]$
```

hostid - 显示当前主机的十六进制数字标识

打印出主机的id标识，将会输出一个十六进制的数字。

该命令是GNU coreutils包中的命令，相关的帮助信息请查看man -s 1 hostid, info coreutils 'hostid invocation'。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
hostid [OPTION]
```

sh

选项

```
--help          # 显示帮助文档  
--version      # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon ~]$ hostid  
000a0f02  
[sogrey@bogon ~]$
```

sh

hostname - 显示和设置系统的主机名

hostname命令 用于显示和设置系统的主机名称。

- 环境变量 HOSTNAME 也保存了当前的主机名。
- 在使用 hostname 命令设置主机名后，系统并不会永久保存新的主机名，重启之后还是原来的主机名。如果需要永久修改主机名，需要修改 /etc/hosts 和 /etc/sysconfig/network 的相关内容并进行重启；也可以使用 hostnamectl 命令进行永久修改。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
hostname [OPTION]
hostname [-b] {hostname|-F file}          # 设置主机名称（或从文件获取）
hostname [-a|-A|-d|-f|-i|-I|-s|-y]       # 显示格式化的名称
hostname                                     # 显示主机名称

{yp,nis,}domainname {nisdomain|-F file}    # 设置 NIS 主机名称（或从文件获取）
{yp,nis,}domainname                         # 显示 NIS 主机名称

dnsdomainname                                # 显示 DNS 主机名称

hostname -V|--version|-h|--help             # 打印信息并退出
```

sh

选项

```
-v                                         # 显示详细执行过程
-a, --alias                               # 显示主机别名
-d, --domain                             # 显示主机dns域名
-F file                                  # 从文件读取
-f, --fqdn, --long                        # 显示完全格式的域名
-A, --all-fqdns                          # 显示机器的全部FQDNs
-i, --ip-address                         # 显示指定主机的ip地址
-I, --all-ip-address                     # 显示主机所有的地址
-s, --short                               # 以短格式显示，仅显示从第一个点分开的部分
-y, --yp, --nis                           # 显示nis域名

-h, --help                                 # 显示帮助文档
-V, --version                            # 显示命令版本信息
```

sh

FQDN

不能用此命令更改FQDN(由hostname-fqdn返回)或DNS域名(由dnsdomainname返回)。系统的FQDN是

resolver为主机名返回的名称。技术上：FQDN是由gethostname(2)返回的主机名。DNS域名是第一个点之后的部分。因此，如何更改取决于配置文件(通常在"/etc/host.conf"中)。通常(如果在DNS或NIS之前解析主机文件)，您可以在"/etc/host"中更改它。

如果一台机器有多个网络接口/地址，或者在移动环境中使用，那么它可能有多个FQDN/域名，或者根本没有。因此，避免使用"hostname -fqdn"、"hostname --domain"和"dnsdomainname"。"hostname --ip-address"地址也受到同样的限制，因此也应该避免。

举例

显示主机名

```
[root@localhost ~]$ hostname #显示完整名字  
localhost.localdomain  
[root@localhost ~]$ hostname -s #显示短格式名字  
localhost  
[root@localhost ~]$ hostname -a #显示主机别名  
localhost.localdomain localhost4 localhost4.localdomain4 localhost.localdomain loc
```

sh

显示主机ip

```
[root@localhost ~]$ hostname -i  
127.0.0.1 127.0.0.1
```

sh

hostnamectl - 查询或更改系统主机名

hostnamectl可用于查询和更改系统主机名和相关设置。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
hostnamectl [选项...] 指令 ...
```

sh

指令

```
status                # 显示当前主机名设置  
set-hostname NAME    # 设置系统主机名  
set-icon-name NAME   # 设置主机的图标名称  
set-chassis NAME    # 设置主机的机箱类型  
set-deployment NAME # 设置主机的部署环境  
set-location NAME    # 设置主机位置
```

sh

选项

```
-h --help                # 显示此帮助  
--version                # 显示包的版本  
--no-ask-password      # 不提示输入密码  
-H --host=[USER@]HOST   # 在远程主机上操作  
-M --machine=CONTAINER # 在本地容器上执行操作。指定要连接到的容器名称。  
--transient, --static, --pretty  
                        # 如果调用了status（或者没有给出显式命令）并且指定了其中一个开关，hostnamectl将
```

sh

举例

显示主机名设置

```
$ hostnamectl status
```

sh

改变主机名(永久修改,不用重启哦~)

```
$ sudo hostnamectl set-hostname newname
```

sh

htdigest - Apache服务器内置工具

htdigest指令用来建立和更新apache服务器用于摘要认证的存放用户认证信息的文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
htdigest [-c] passfile realm username
```

sh

选项

```
-c # 创建密码文件，如果存在，首先删除
```

sh

举例

创建摘要认证文件

```
[root@localhost ~]$ htdigest -c htfile qq.com weijie #用户weijie在qq.com的认证文件
Adding password for weijie in realm qq.com.
New password:                                     #输入密码
Re-type new password:                            #确认密码
You have new mail in /var/spool/mail/root
[root@localhost ~]$ cat htfile                  #显示认证文件
weijie:qq.com:3d3feff0cf3f031cf3652349b7249d59
```

sh

htpasswd - apache服务器创建密码认证文件

htpasswd命令 是Apache的Web服务器内置工具，用于创建和更新储存用户名、域和用户基本认证的密码文件。

htpasswd指令用来创建和更新用于基本认证的用户认证密码文件。 htpasswd指令必须对密码文件有读写权限，否则会返回错误码。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
htpasswd [ -c ] [ -m ] [ -D ] passwdfile username
htpasswd -b [ -c ] [ -m | -d | -p | -s ] [ -D ] passwdfile username password
htpasswd -n [ -m | -d | -s | -p ] username
htpasswd -nb [ -m | -d | -s | -p ] username password
```

sh

选项

```
-b      # 使用批处理方式，直接从命令行获取密码，不提示用户输入
-c      # 创建密码文件，如果文件存在，那么内容被清空重写
-n      # 将结果送到标准输出
-m      # 使用MD5加密
-s      # 使用crypt()加密
-p      # 使用文本密码
-D      # 从认证文件中删除用户记录
```

sh

举例

创建基本认证文件

```
[root@localhost ~]$ htpasswd -cm htpfile1 sogrey      #创建认证文件，使用md5加密
New password:
Re-type new password:
Adding password for user sogrey
You have new mail in /var/spool/mail/root
[root@localhost ~]$ cat htpfile1                  #显示认证文件
sogrey:$apr1$/RxQ5LT9$L1WJPkxknMizG5DwGVGv4.
```

sh

创建基本认证文件，使用文本密码

httpd - apache超文本传输协议的主程序

httpd是apache超文本传输协议的主程序，它被设计成一个独立运行的守护进程。httpd会建立一个线程池来处理http请求。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
httpd [OPTION]
```

sh

选项

```
-d serverroot          # 设置服务器根目录。对应配置文件中的ServerRoot指令
-f config              # 定apache服务器的配置文件。如果配置文件不使用绝对路径，那么就是相对于ServerRoot
-k start|restart|graceful|stop|graceful-stop # 向httpd进程发送信息，可以控制httpd
-C                   # 在读取配置文件之前，先处理指定的指令
-c                   # 在读取配置文件之后，处理指定的指令
-D param              # 设置参数，它可以配合apache的配置文件中<IfDefine>一起使用
-e level              # 设置日志等级
-E file               # 置错误信息文件
-h                   # 显示简短的说明选项
-l                   # 显示静态编译的httpd模块列表
-L                   # 显示apache服务配置文件中的指令列表
-M                   # 显示httpd模块列表
-S                   # 显示虚拟主机配置
-t                   # 检查配置文件语法
-v                   # 显示httpd版本
-V                   # 显示编译时的配置参数和版本信息
-X                   # 运行调试模式
```

sh

举例

重启httpd服务

```
[root@localhost ~]$ httpd -k restart      #重启服务
You have new mail in /var/spool/mail/root
[root@localhost ~]$
```

sh

检测配置文件

```
[root@localhost ~]$ httpd -t          #检测配置文件，没有错误
httpd: Could not reliably determine the server's fully qualified domain name, using localhost.localdomain
Syntax OK
[root@localhost ~]$
```

显示apache中的模块

```
[root@localhost ~]$ httpd -M
httpd: Could not reliably determine the server's fully qualified domain name, using localhost.localdomain
Loaded Modules:
core_module (static)
mpm_prefork_module (static)
http_module (static)
so_module (static)
auth_basic_module (shared)
auth_digest_module (shared)
authn_file_module (shared)
authn_alias_module (shared)
authn_anon_module (shared)
authn_dbm_module (shared)
```

hwclock - 显示与设定硬件时钟

hwclock命令是一个硬件时钟访问工具，它可以显示当前时间、设置硬件时钟的时间和设置硬件时钟为系统时间，也可设置系统时间为硬件时钟的时间。

在Linux中有硬件时钟与系统时钟等两种时钟。硬件时钟是指主机板上的时钟设备，也就是通常可在BIOS画面设定的时钟。系统时钟则是指kernel中的时钟。当Linux启动时，系统时钟会去读取硬件时钟的设定，之后系统时钟即独立运作。所有Linux相关指令与函数都是读取系统时钟的设定。

hwclock是一种访问硬件时钟的工具，可以显示当前时间，将硬件时钟设置为指定的时间，将硬件时钟设置为系统时间，以及从硬件时钟设置系统时间。您还可以定期运行hwlock以插入或删除硬件时钟中的时间，以补偿系统漂移(如果继续运行，则该时钟始终以一定的速率获得或丢失时间)。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
hwclock [functions] [options]
```

sh

选项

下面的选项告诉hwclock指令该执行那个函数

```
-r, --show          # 读取硬件时钟并在标准输出上打印时间。显示的时间总是本地时间，即使您将硬件时钟保持  
--set              # 将硬件时钟设置为-date选项指定的时间  
-s, --hctosys      # 从硬件时钟设置系统时间。  
                   # 还将内核的时区值设置为由TZ环境变量和/usr/share/zoneinfo指示的本地时区，正  
                   # 这是在系统启动脚本中使用的一个很好的选项  
-w, --systohc      # 将硬件时钟设置为当前的系统时间  
--systz            # 根据当前时区设置内核的时区并重置系统时间，系统时间仅在启动后的第一次调用时重置。  
                   # 本地时区被认为是TZ环境变量和/usr/share/zoneinfo所指示的，tzset(3)将解释它  
                   # 这是-hctosys的另一个选项，它不读取硬件时钟，并且可以在最近2.6内核的系统启动脚本  
                   # 从硬件时钟中增加或减去时间，以说明自上次时钟设置或调整以来的系统漂移。请参阅下面  
                   # 将内核的硬件时钟时纪元打印到标准输出。这是在AD中引用硬件时钟中的零年值的年份数。  
                   # 每当hwlock读取或设置硬件时钟时，就会使用这个纪元值。epoch只有在Alpha机器上有。  
--adjust           # 将内核的硬件时钟纪元值设置为--epoch选项指定的值。  
--getepoch         # 在标准输出上显示hwclock的版本  
--setepoch         # 如果指定-set选项，则需要此选项。否则，将忽略该选项。这将指定设置硬件时钟的时间。  
-v, --version       # 参数在本地时间，即使您将硬件时钟保持在协调的世界时间。请参见-UTC选项。  
--date=date_string  # 指定硬件时钟时代开始的年份，即在AD中，硬件时钟的年份计数器中的零值所指的进入AD的  
--epoch=year        # 指定硬件时钟时代开始的年份，即在AD中，硬件时钟的年份计数器中的零值所指的进入AD的
```

下面的选项配合函数使用

```
-u, --utc, --localtime # 指示硬件时钟分别保持在协调的世界时间或本地时间。您可以选择是否将时钟保持在协调世界时  
# 如果您指定了这些选项中的一个错误的(或者没有指定，并且选择了错误的默认值)，那么硬件时钟将被设置为本地时间。  
# 如果您没有指定-utc或-localtime，则默认值以最后一次使用hwlock设置时钟时指定的值为准。  
--noadjfile # 禁用/etc/adjtime.hwlock提供的工具，不使用此选项读取或写入该文件。在使用此选项时，  
# 覆盖默认的/etc/adjtime文件。  
--adjfile=filename # 重写默认/dev文件名，在许多平台上为/dev/rtc，但可能是/dev/rtc0、/dev/rtc1，等等。  
# 只有在ISA机器或Alpha上才有意义(粗略地说，它实现了足够的ISA机器来实现hwlock功能)。  
--directisa # 表示硬件时钟无法存储超出1994-1999年范围的年份。在一些BIOSes(几乎所有在4/26/99之后的)  
# 为了弥补这一点(没有BIOS更新，这肯定更好)，请始终使用-如果您有这些机器中的一台，  
# 虽然hwlock在读取硬件时钟时忽略了年份值，但当它设置时钟时，它会设置年份值，将其设为1999。  
# 此选项等价于--epoch=1900，用于使用srn控制台指定alphad中最常见的历元。  
--badyear # 此选项等价于--epoch=1980，用于使用ARC控制台指定ALPHS上最常用的历次(但Ruffans除外)。  
--srm # 这两个选项指定了您拥有的Alpha机器的类型。如果您没有Alpha，则它们是无效的，如果  
# --jensen代表运行在Jensen模式。  
--arc # --funky-toy意味着在您的机器上，必须使用UF位而不是硬件时钟中的UIP位来检测时间转换。  
# 测试程序，不改变任何设置。  
--test # 显示大量关于hwlock内部正在做什么的信息，其中一些功能是复杂的，这个输出可以帮助您  
# 调试。  
--debug
```

说明

一般在操作系统中都会有两个时钟，硬件时钟是主板上的定时器时钟，系统时钟是系统的内核时钟，它们相互不影响。

1. 硬件时钟

这个时钟，运行独立于任何控制程序运行在CPU中，甚至当机器关闭。在ISA系统中，这个时钟被指定为ISA标准的一部分。控制程序可以读取或设置这个时钟为整秒，但控制程序也可以检测1秒时钟的边缘，因此该时钟实际上具有无限的精度。

这种时钟通常被称为硬件时钟、实时时钟、RTC、BIOS时钟和CMOS时钟。硬件时钟以其大写的形式被hwlock所发明，因为其他所有的名称都不适合误导。例如，一些非ISA系统有几个实时时钟。一个非常低功耗的I2C或SPI时钟芯片可以与备用电池一起作为硬件时钟，以初始化一个功能更好的集成实时时钟，用于大多数其他用途。

2. 系统时钟

这是由Linux内核内的时钟保持的时间，由计时器中断驱动。(在ISA机器上，计时器中断是ISA标准的一部分)。它只有在linux在机器上运行时才有意义。系统时间是从1970年世界协调时(UTC)1月1日00: 00开始的秒数(或者更简洁地说，是1969年以来的秒数)。不过，系统时间不是整数，它实际上是无限的。系统时间是重要的时间。Linux系统中硬件时钟的基本目的是在Linux不运行时保持时间。在Linux启动时，将系统时间从硬件时钟初始化，然后不再使用硬件时钟。请注意，在设计ISA的DOS中，硬件时钟是唯一的实时时钟。

重要的是，当系统运行时，系统时间不存在任何不连续性，比如使用date命令来设置它。但是，在系统运行时，您可以对硬件时钟做任何您想做的事情，而下一次Linux启动时，它将使用硬件时钟的调整时间进行设置。

Linux内核维护系统的本地时区的概念。但是不要被误导-几乎没有人关心内核认为它在哪个时区。相反，关心时区的程序(可能因为他们想为您显示本地时间)几乎总是使用更传统的方法。确定时区：它们使用“tz”环境变量或“/usr/share/zoneinfo”目录，如tzset(3)的手册页所解释的那样。时区值是错误的，vFAT文件系统会在文件上报告并设置错误的时间戳。

当您使用“--hctosys”选项设置系统时间时，hwlock将内核时区设置为“tz”或“/usr/share/zoneinfo”所指示的值。

时区值实际上由两部分组成：1)字段“tz_minutesWest”表示本地时间(未根据DST进行调整)滞后于UTC；2)字段“tz_dsttime”，指示当前在本地有效的夏令时(DST)约定的类型。第二个字段不在Linux下使用，始终为零。

3. **hwclock**如何访问硬件时钟

hwlock使用多种不同的方法来获取和设置硬件时钟值，最常见的方法是对设备特殊文件“/dev/rtc”执行I/O操作，假定该文件是由rtc设备驱动程序驱动的。然而，这种方法并不总是可用的。首先，rtc驱动程序是linux中比较新的一种。此外，虽然有一些版本的rtc驱动程序可以在decalpha上工作，但似乎有大量的alpha无法工作(常见的症状是时钟挂起)。此外，最近的linux系统对rtc有更多的通用支持，甚至支持不止一个的系统，所以您可能需要通过指定“/dev/rtc 0或/dev/rtc 1”来覆盖默认值。

在旧系统中，访问硬件时钟的方法取决于系统硬件。

在ISA系统中，hwlock通过对端口0x70和0x71进行I/O操作，可以直接访问构成时钟的“CMOS存储器”寄存器。它使用实际的I/O指令，因此只有在超级用户有效用户ID的情况下才能这样做。(对于jensen Alpha，hwlock无法执行这些I/O指令，因此它使用设备文件“/dev/port”，它提供了与I/O子系统几乎一样低的接口)。这是一种非常糟糕的访问时钟的方法，因为用户空间程序通常不应该进行直接I/O和禁用中断。但是在ISA和Alpha系统中，这是唯一的方式。

在m68k系统上，hwlock可以通过控制台驱动程序访问时钟，通过设备文件“/dev/tty1”访问时钟。

hwlock尝试使用文件“/dev/rtc”。如果内核没有编译“/dev/rtc”，或者它无法打开“/dev/rtc”，那么hwlock将返回到另一种方法(如果可用的话)。在ISA或Alpha计算机上，您可以强制hwclock使用CMOS寄存器的直接操作，而无需通过指定“--directisa”选项。

4. 校准功能**adjust**

硬件时钟通常不是很精确，但是它的许多不准确是完全可以预测的，它每天得到或失去相同的时间。这被称为系统漂移。hwlock的“调整”功能允许您进行系统校正以纠正系统漂移。它的工作方式如下：hwlock保存了一个文件“/etc/adjtime”，它保存了一些历史信息。

假设您从没有adjtime文件开始，发出hwlock-set命令将硬件时钟设置为真实的当前时间。hwlock创建adjtime文件，并在其中记录当前时间，作为最后一次校准时钟。5天后，时钟增加了10秒，因此您可以发出另一个“hwlock --set”命令来设置它。返回10秒。hwlock更新adjtime文件，显示当前时间作为最后一次校准时钟，并以系统漂移速率记录每天2秒。24小时过去，然后发出“hwlock --adjust”命令。hwlock查阅adjtime文件，看到时钟离开时每天增加2秒。一个人呆了整整一天。所以它从硬件时钟中减去2秒。然后，它记录当前时间作为最后一次调整时钟的时间。又过了24小时，你又发出了另一个“hwclock --adjust”指令。hwclock做了同样的事情：减去2秒，用当前时间更新adjtime文件，这是最后一次调整时钟。

每次您校准时钟(使用--set或--systohc)时，hwlock根据上次校准的时间、上次调整后的时间、在任何中间的调整中假定的漂移率以及时钟当前的关闭量，重新计算系统漂移率。在hwclock使用的任何时候，都会出现少量的误差，因此它不会进行小于1秒的调整。稍后，当您再次请求调整时，累积漂移将超过1秒钟，而hwlock则会进行调整。

在系统启动时，在“hwlock --hctosys”之前进行hwlock的调整是很好的，并且在系统通过cron运行时也可以定期进行调整。

虽然adjtime文件的命名仅仅是为了控制时间调整的历史记录，但它实际上包含了hwlock在从一个调用到

下一个调用时记忆信息时使用的其他信息。adjtime文件的格式是ASCII:

第1行的3个数字数字，用空格隔开，分别代表：a)系统漂移率，每天以秒为单位，浮点小数点；b)自1969年世界协调时以来最近调整或校准的秒数，小数整数；c)零(与时钟(8)兼容)为十进制整数。

第2行一个数字，代表自1969年世界协调时以来最近一次校准产生的秒数。如果还没有校准，或者已知任何先前的校准都是没有意义的，那么值就是0(例如，因为在校准之后，硬件时钟已经被找到，不包含有效时间)。这是一个十进制整数。

第3行是“utc”或“local”。指示硬件时钟是设置为协调世界时间还是设置为本地时间。

5. 内核如何自动同步硬件时钟

在某些系统中，您应该注意到硬件时钟保持同步的另一种方式。Linux内核有一种模式，它每11分钟将系统时间复制一次到硬件时钟。这是一个很好的模式，当您使用一些复杂的东西，比如NTP来保持系统时间同步时。(NTP是一种保持系统时间同步的方法，它可以与网络上的某个时间服务器或连接到您的系统的无线电时钟保持同步。参见RFC 1305)。

这个模式(我们称之为“11分钟模式”)是关闭的，直到有东西打开它。ntp守护进程xntpd就可以打开它。您可以通过运行任何东西来关闭它，包括“hwclock --hctosys”，它以老式的方式设置系统时间。

如果你的系统以11分钟的模式运行，不要使用“hwclock --adjust”或“hwclock-hctosys”。在启动时使用“hwclock --hctosys”来获得一个合理的系统时间是可以接受的，直到您的系统能够运行为止。从外部源设置系统时间并启动11分钟模式

举例

不适用任何参数，直接查看硬件时钟

```
[root@localhost ntop-4.0.1]$ hwclock  
2018年08月23日 星期四 15时01分28秒 -0.577410 seconds
```

设置硬件时钟

```
[root@localhost ntop-4.0.1]$ hwclock --set --date="0904" # 设置硬件时钟，需要date参数来配合使用  
[root@localhost ntop-4.0.1]$ hwclock  
2018年09月04日 星期二 09时04分09秒 -0.479386 seconds
```

将硬件时钟设置成本地时间格式

```
[root@localhost ntop-4.0.1]$ hwclock --localtime  
2018年09月04日 星期二 01时05分46秒 -0.462990 seconds
```

将硬件时钟设置成系统时间

```
[root@localhost ntop-4.0.1]$ date #查看当前系统时间  
2018年 09月 04日 星期二 12:25:15 CST  
[root@localhost ntop-4.0.1]$ hwclock -w #将硬件时钟设置为系统时间  
[root@localhost ntop-4.0.1]$ hwclock #查看硬件时钟  
2018年09月04日 星期二 12时25分48秒 -0.263687 seconds
```


ifcfg - 配置Linux中的网络接口参数

ifcfg命令 是一个Bash脚本程序，用来设置Linux中的网络接口参数。

ifcfg是一个简单的脚本替换iconfig命令，它可以设置网络接口的ip地址。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ifcfg [device] [cmd] [address]
```

sh

device就是网卡设备，它可能有别名。cmd可以是add、delete、stop。address就是ip地址。

选项

无

举例

添加ip地址

```
[root@localhost ~]$ ifcfg eth0 add 192.168.0.250/24      #添加地址250
Forwarding is ON or its state is unknown (4). OK, No RDISC.
[root@localhost ~]$ ifconfig                                #查看网络信息
eth0      Link encap:Ethernet HWaddr 08:00:27:14:33:57
          inet addr:192.168.0.250  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe14:3357/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:107276 errors:0 dropped:0 overruns:0 frame:0
            TX packets:72250 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:83580745 (79.7 MiB)  TX bytes:5842176 (5.5 MiB)
```

sh

删除ip地址

sh

```
[root@localhost ~]$ ifcfg eth0 delete 192.168.0.250/24      #删除网卡地址
Forwarding is ON or its state is unknown (4). OK, No RDISC.
[root@localhost ~]$ ifconfig                                #查看网卡信息, ip地址已经删除
eth0      Link encap:Ethernet  HWaddr 08:00:27:14:33:57
          inet6 addr: fe80::a00:27ff:fe14:3357/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:107276 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72251 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:83580745 (79.7 MiB)  TX bytes:5842218 (5.5 MiB)
```

ifconfig - 配置和显示Linux系统网卡的网络参数

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。

ifconfig指令用来配置网络接口参数，同时还可以显示当前内核网络接口的工作状态。如果没有提供参数，则ifconfig将显示当前活动接口的状态。如果给定单个接口参数，则只显示给定接口的状态；如果给定单个“-a”参数，则显示所有接口的状态，即使是关闭的接口也是如此。否则，它会配置一个接口。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ifconfig [interface | up | down]
```

sh

选项

```
interface          # 接口的名称。这通常是一个驱动程序名，后面跟着一个单元号，例如用于第一个以太网接口的e1
up                # 此标志将导致激活接口。如果将地址分配给接口，则会隐式指定该地址。
down              # 此标志导致关闭此接口的驱动程序。
[-]arp             # 启用或禁用在此接口上使用ARP协议。
[-]promisc         # 启用或禁用接口的混杂模式。如果选中，网络上的所有数据包都将由接口接收。
[-]allmulti        # 启用或禁用所有多播模式。如果选中，则接口将接收网络上的所有多播数据包。
metric N           # 此参数设置接口度量。它在GNU/Linux下不可用
mtu N              # 此参数设置接口的最大传输单元(MTU)。
dstaddr addr       # 为点对点链路(如PPP)设置远程IP地址。这个关键字现在已经过时了；使用pointopoint关键字
netmask addr       # 设置此接口的IP网络掩码。此值默认为通常的A、B或C类网络掩码(从接口IP地址派生)，但
add addr/prefixlen # 向接口添加IPv 6地址
del addr/prefixlen# 从接口中删除IPv 6地址
tunnel ::aa.bb.cc.dd# 创建一个新的SIT(IPv6-in-IPv4)设备，通过隧道到达给定的目的地。
irq addr            # 设置此设备使用的中断行。并非所有设备都可以动态更改其IRQ设置。
io_addr addr        # 为该设备设置I/O空间中的起始地址
mem_start addr      # 设置此设备使用的共享内存的起始地址。只有少数几个设备需要这个
media type          # 设置设备要使用的物理端口或介质类型。并非所有设备都可以更改此设置，以及那些可以更改
[-]broadcast [addr] # 如果地址参数给定，则为该接口设置协议广播地址。否则，设置(或清除)接口的IFF_BROADCAST
[-]pointopoint [addr]# 这个关键字启用了接口的点对点模式，这意味着它是两台机器之间的直接链接，没有其他人
hw class address    # 如果设备驱动程序支持此操作，则设置此接口的硬件地址。关键字后面必须跟着硬件类的名称
multicast           # 在接口上设置多播标志。这通常不应该需要，因为驱动程序本身设置正确的标志。
address             # 要分配给此接口的IP地址。
txqueuelen length  # 设置设备的传输队列的长度。对于具有高延迟(调制解调器链路，ISDN)的较慢设备，将其设
```

地址族

如果接口名称之后的第一个参数被识别为受支持地址族的名称，则该地址族用于解码和显示所有协议地址。目前支持的地址族包括Internet(TCP/IP， 默认值)、inet6(IPv 6)、Axis25(AMPR分组无线电)、ddp(AppleTalk相位2)、IPX(Novell IPX)和netrom(AMPR分组无线电)。在IPv4虚线小数表示法中提供的所有数字都可以是十进制、八进制或十六进制，正如ISO C标准所指定的那样(即，前导0x或0X表示十六进制；否则，前导“0”表示八进制；否则，该数字被解释为十进制)。使用十六进制和八进制数字是不符合RFC的，因此它的使用是不鼓励的，可能会消失。

举例

查看当前网络状态

```
[root@localhost ~]$ ifconfig      #没有任何参数选项，显示当前所有网络状态
eth0      Link encap:Ethernet HWaddr 08:00:27:14:33:57
          inet6 addr: fe80::a00:27ff:fe14:3357/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:107276 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72251 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:83580745 (79.7 MiB)  TX bytes:5842218 (5.5 MiB)

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:7347 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7347 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:786270 (767.8 KiB)  TX bytes:786270 (767.8 KiB)
```

查看指定网卡的状态

```
[root@localhost ~]$ ifconfig eth0      #显示指定网卡状态
eth0      Link encap:Ethernet HWaddr 08:00:27:14:33:57
          inet6 addr: fe80::a00:27ff:fe14:3357/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:107276 errors:0 dropped:0 overruns:0 frame:0
          TX packets:72251 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:83580745 (79.7 MiB)  TX bytes:5842218 (5.5 MiB)
```

3) 启动网卡

```
[root@localhost ~]$ ifconfig eth0 down      #关闭eth0
[root@localhost ~]$ ifconfig eth0 up        #开启eth0
[root@localhost ~]$ ifconfig eth0          #查看eth0, 已经分配ip
eth0      Link encap:Ethernet  HWaddr 08:00:27:14:33:57
          inet  addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe14:3357/64  Scope:Link
             UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
             RX packets:107280 errors:0 dropped:0 overruns:0 frame:0
             TX packets:72262 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:83582067 (79.7 MiB)  TX bytes:5843588 (5.5 MiB)
```

sh

其他

启动关闭指定网卡：

```
ifconfig eth0 up
ifconfig eth0 down
```

sh

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
```

sh

用ifconfig修改MAC地址：

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

sh

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
```

sh

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
```

sh

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
```

sh

ifdown - 禁用指定的网络接口

ifdown命令 用于禁用指定的网络接口。

ifdown指令用来关闭网络接口设备，设备必须是定义在"/etc/sysconfig/network-scripts/ifcfg-ethX"或者"/etc/sysconfig/network"的文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ifdown interface
```

sh

选项

无

举例

关闭网卡

```
[root@localhost ~]$ ifdown eth0    #关闭eth0  
设备状态: 3 (断开连接)
```

sh

ifup - 激活指定的网络接口

ifup命令 用于激活指定的网络接口。

ifup指令用来启动网络接口设备，设备必须是定义在"/etc/sysconfig/network-scripts/ifcfg-ethX"或者"/etc/sysconfig/network"的文件。这些脚本通常使用一个参数：配置的名称(例如eth0)。在引导序列中，使用"boot"的第二个参数调用它们，以便在引导过程中不想打开的设备(ONBOOT=no)此时可以被忽略。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ifup interface
```

sh

选项

无

举例

激活网卡

```
[root@localhost ~]$ ifup eth0    #激活网卡eth0
活跃连接状态: 激活中
活跃连接路径: /org/freedesktop/NetworkManager/ActiveConnection/2
状态: 激活的
连接被激活
```

sh

indent - 格式化C语言的源文件

indent命令 可辨识C的原始代码文件，并加以格式化，以方便程序员阅读、修改等操作。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
indent [选项] [源文件]  
indent [选项] [源文件] [-o 目标文件]
```

sh

选项

```

-bad:                                # 在声明区加上空白行;
-bap:                                # 添加空白行;
-bbb:                                # 在注释后面添加空白行;
-bc:                                 # 在声明段中, 如果出现逗号就换行;
-bl:                                 # if (或是else、for等) 与后面执行区段的“{”不同行, 且“}”
                                         # 自成一行-bli<缩排格数>设置{}缩排的格数;
-br:                                 # if (或是else、for等) 与后面执行区段的“{”同行, 且“}”自成一行;
-bs:                                 # 在sizeof之后空一格;
-c<栏数>:                            # 将注释置于程序右侧指定的栏位;
-cd<栏数>:                            # 将注释置于声明右侧指定的栏位;
-cdb:                                # 注释符号自成一行;
-ce:                                 # 将else置于“}”(if执行区段的结尾)之后;
-ci: <缩排格数>;                   # 叙述过长而换行时, 指定换行后缩排的格数;
-cli<缩排格数>:                     # 使用case时, switch缩排的格数;
-cp<栏数>:                            # 将注释置于else与elseif叙述右侧指定的栏位;
-cs:                                 # 在case之后空一格;
-d<缩排格数>:                      # 针对不是放在程序码右侧的注释, 设置其缩排格数;
-di<栏数>:                            # 将声明区段的变量置于指定的栏位;
-fc1:                                # 针对放在每行最前端的注释, 设置其格式;
-fca:                                # 设置所有注释的格式;
-gnu:                                 # 使用指定的GNU格式, 该参数为默认值;
-i<格数>:                            # 设置缩排的格数;
-ip<格数>:                            # 设置参数的缩排格数;
-kr:                                 # 指定使用Kernighan&Ritchie的格式;
-lp:                                 # 叙述过长而换行, 且叙述中包含了括号时,
                                         # 将括号中的每行起始栏位内容垂直对其排列;
-nbad:                               # 在声明区段后不要加上空白行;
-nbap:                               # 在程序后面不添加空白行;
-nbbb:                               # 在注释段后面不添加空白行;
-nbc:                                # 在声明段中, 即使出现逗号, 也不换行;
-ncdb:                               # 注释符号不自成一行;
-nce:                                 # 不将else置于“}”后面;
-ncs:                                 # 不在case后面空一格;
-nfc1:                               # 不要格式化放在每行最前端的注释;
-nfca:                               # 不用格式化任何的注释;
-nip:                                 # 参数不要缩排;
-nlp:                                 # 叙述过长而换行, 且叙述中包含了括号时,
                                         # 不用将括号中的每行起始栏位垂直对其排列;
-npcs:                               # 在调用函数名之后, 不要添加空格;
-npro:                               # 不要读取indent的配置文件“.indent.pro”;
-npsl:                               # 程序类型与程序名称放在同一行;
-nsc:                                 # 注释左侧不要添加星号;
-nsob:                               # 不用处理多余的空白行;
-nss:                                 # 若for或while区段仅有一行时, 在分号前不加空格;
-nv:                                  # 不显示详细的信息;
-orig:                               # 使用berkeley格式;
-pcs:                                 # 在调用函数名与“{”之间添加空格;
-psl:                                 # 程序类型置于程序名称的前一行;
-sc:                                  # 在每行注释左侧添加星号;
-sob:                                 # 删除多余的空白行;
-ss:                                  # 若for或swile区段仅有一行时, 在分号前加上空格;
-st:                                  # 将结果显示在标准输出设备上;
-T:                                   # 数据类型名称缩排;
-ts<格数>:                           # 设置tab的长度;

--help                                # 显示帮助文档
--version                             # 显示命令版本信息

```

举例

使用indent命令将C语言源文件"test.c"中所有的sizeof后面添加一个空格，输入如下命令：

```
[sogrey@bogon newDir]$ > test.c
[sogrey@bogon newDir]$ cat test.c
#include "stdio.h"
#include "string.h"
#include "stdlib.h"
int main()
{
    short int sa=10;
    int a=10;
    long la=10;
    float f = 20;
    double d=20;
    char ch='c';
    char str[]="ABC";
    char *p=str;
    struct str{
        double d;
        char ch;
        int data;
    }str_wu;
    struct str1{
        char ch;
        double d;
        int data;
    }str_wu1;
    printf("sizeof(short):%d\n",sizeof(sa));
    printf("sizeof(int):%d\n",sizeof(a));
    printf("sizeof(long):%d\n",sizeof(la));
    printf("sizeof(float):%d\n",sizeof(f));
    printf("sizeof(double):%d\n",sizeof(d));
    printf("sizeof(char):%d\n",sizeof(ch));
    printf("sizeof(string):%d\n",sizeof(str));
    printf("sizeof(point address):%d\n",sizeof(p));
    printf("sizeof(Point):%d\n",sizeof(*p));
    printf("sizeof(Struct):%d\n",sizeof(str_wu));
    printf("sizeof(Struct):%d\n",sizeof(str_wu1));
    system("pause");
}
[sogrey@bogon newDir]$ indent -bs test.c
[sogrey@bogon newDir]$ cat test.c
#include "stdio.h"
#include "string.h"
#include "stdlib.h"
int
main ()
{
    short int sa = 10;
    int a = 10;
    long la = 10;
    float f = 20;
    double d = 20;
    char ch = 'c';
    char str[] = "ABC";
```

```
char *p = str;
struct str
{
    double d;
    char ch;
    int data;
} str_wu;
struct str1
{
    char ch;
    double d;
    int data;
} str_wu1;
printf ("sizeof(short):%d\n", sizeof (sa));
printf ("sizeof(int):%d\n", sizeof (a));
printf ("sizeof(long):%d\n", sizeof (la));
printf ("sizeof(float):%d\n", sizeof (f));
printf ("sizeof(double):%d\n", sizeof (d));
printf ("sizeof(char):%d\n", sizeof (ch));
printf ("sizeof(string):%d\n", sizeof (str));
printf ("sizeof(point address):%d\n", sizeof (p));
printf ("sizeof(Point):%d\n", sizeof (*p));
printf ("sizeof(Struct):%d\n", sizeof (str_wu));
printf ("sizeof(Struct):%d\n", sizeof (str_wu1));
system ("pause");
}
[sogrey@bogon newDir]$
```

执行上面的命令后，用户可以打开指定的源文件查看在`sizeof`后面是否都添加了一个空格。由于该命令的参数非常多，所以用户可以根据实际需要选择适合的参数进行使用即可。

init - init进程是所有Linux进程的父进程

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。

init是所有进程的父进程，它由内核执行，可以启动其他所有的进程。init指令在启动时会参考/etc/inittab文件的配置，完成其他进程的启动。init通常不会由用户进程执行，并且期望进程id为1。如果不是这样，它将实际执行telinit(8)并将所有参数传递给它。

init管理的进程称为作业，并由/etc/init目录中的文件定义。init(8)是一个基于事件的init守护进程。这意味着作业将通过系统状态发生的更改自动启动和停止，包括作业的启动和停止。这与基于依赖项的init守护进程不同，后者启动一组指定的目标作业，并通过迭代它们的依赖项来解决它们应该启动的顺序和其他作业所需的顺序。主要事件是startup (7)事件，在守护进程加载完其配置后发出。其他有用的事件是以作业更改状态发出的starting (7)、started (7)、stopping (7)和stopped (7)事件。

init有7种运行等级：

- 0 关机
- 1 单用户模式
- 2 多用户模式，不启动nfs
- 3 多用户模式，有网络功能
- 4 保留
- 5 图形界面
- 6 重启

Upstart init(8)守护进程不跟踪运行级别本身，而是完全由用户空间工具实现。为表示运行级的更改而发出的事件是runlevel(7)事件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
init [选项] [参数]
```

sh

选项

```
--verbose          # 将有关作业状态更改和事件释放的详细消息输出到系统控制台或日志，这对于调试非常有用
```

```
--help            # 显示帮助文档
```

```
--version         # 显示命令版本信息
```

sh

insmod - 将给定的模块加载到内核中

insmod命令 用于将给定的模块加载到内核中。Linux有许多功能是通过模块的方式，在需要时才载入kernel。如此可使kernel较为精简，进而提高效率，以及保有较大的弹性。这类可载入的模块，通常是设备驱动程序。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
insmod [OPTION] [参数]
```

sh

内核模块：指定要加载的内核模块文件。

选项

```
-f                # 不检查目前kernel版本与模块编译时的kernel版本是否一致，强制将模块载入;  
-k                # 将模块设置为自动卸除;  
-m                # 输出模块的载入信息;  
-o<模块名称>    # 指定模块的名称，可使用模块文件的文件名;  
-p                # 测试模块是否能正确地载入kernel;  
-s                # 将所有信息记录在系统记录文件中;  
-v                # 执行时显示详细的信息;  
-x                # 不要汇出模块的外部符号;  
-X                # 汇出模块所有的外部符号，此为预设置。
```

sh

举例

加载RAID1阵列级别模块，如下所示

```
[root@localhost boot]$ insmod /lib/modules/2.6.  
18-8.el5/kernel/drivers/md/raid1.ko  
  
[root@localhost boot]$ lsmod | grep raid1  
raid1                25153  0
```

sh

从以上显示结果可知，RAID1模块已加载成功。只是在使用insmod命令加载模块时，需要使用绝对路径方能加载，且加载时无法自动解决依赖关系。

iostat - 监视系统输入输出设备和CPU的使用情况

iostat命令 被用于监视系统输入输出设备和CPU的使用情况。它的特点是汇报磁盘活动统计情况，同时也会汇报出CPU使用情况。同vmstat一样，iostat也有一个弱点，就是它不能对某个进程进行深入分析，仅对系统的整体情况进行分析。

iostat指令用来显示cpu状态，系统IO设备的状态，以及相关磁盘和NFS使用状态。iostat命令通过观察设备相对于其平均传输速率的活动时间来监视系统输入/输出设备负载。iostat命令生成可用于更改系统配置的报告，以更好地平衡物理磁盘之间的输入/输出负载。

iostat命令生成的第一个报告提供了自系统启动以来的统计数据，除非在省略该第一个报告时使用-y选项。每一份后续报告都涵盖自上次报告以来的时间。每次运行iostat命令时都会报告所有统计信息。报告由CPU标题行和CPU统计数据行后面的一行组成。在多处理器系统中，CPU统计数据作为所有处理器之间的平均值计算在系统范围内。设备标题行后面显示配置的每个设备的一行统计信息。当使用选项-n时，会显示NFS标题行，并为每个已挂载的网络文件系统显示一行统计信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
iostat [OPTION]
```

sh

选项

```
-c # 显示cpu情况  
-d # 显示设备利用率  
-h # 配合-n选项使用，让输出更加易读  
-j # 显示指定设备的名字、id、label  
-k # 以“kb/s”为单位显示，默认使用“块/s”为单位  
-m # 以Mb/s为单位  
-N # 显示注册设备的映射名字  
-n # 显示NFS状态  
-p # 显示块设备状态  
-t # 显示报告时间  
-x # 显示扩展信息  
-v # 显示版本信息，并且退出  
-y # 如果在给定间隔内显示多个记录，则自系统启动以来省略带有统计信息的第一次报告。  
-z # 告诉iostat，对于在示例期间没有活动的任何设备，都要省略输出。
```

sh

报告

iostat命令生成三种类型的报告：CPU利用率报告、设备使用报告和网络文件系统报告。

1) CPU利用率报告

iostat命令生成的第一个报告是CPU利用率报告。对于多处理器系统，CPU值是所有处理器之间的全局平均值。报告的格式如下：

报告内容	说明
%user	显示在用户级别(应用程序)执行时CPU利用率的百分比。
%nice	以良好的优先级在用户级别执行时显示CPU利用率的百分比。
%system	显示在系统级(内核)执行时出现的CPU利用率百分比。
%iowait	显示CPU或CPU空闲的时间百分比，在此期间，系统有未执行的磁盘I/O请求。
%steal	显示虚拟机管理程序为另一个虚拟处理器服务时，虚拟CPU或CPU在非自愿等待中花费的时间百分比。
%idle	显示CPU或CPU空闲的时间百分比，并且系统没有未执行的磁盘I/O请求。

2) 设备使用报告

iostat命令生成的第二个报告是设备使用报告。设备报告提供每个物理设备或分区的统计信息。可以在命令行上输入要显示统计信息的块设备。如果不使用-x选项，也可以在命令行中输入分区。如果没有输入设备或分区，则为系统使用的每个设备显示统计信息，并提供内核为其维护统计信息。如果在命令行上给出ALL关键字，则会显示系统定义的每个设备的统计信息，包括从未使用过的设备。报告可能会显示以下字段，这取决于所使用的标志

报告内容	说明
Device:	该列给出了nth设备的设备(或分区)名称，它以devm-n的形式显示，内核为2.4，其中m是设备的主要数字，n是一个独特的数字。对于较新的内核，将显示/dev目录中列出的设备名称。
tps	指示每秒发送给设备的传输次数。传输是对设备的I/O请求。多个逻辑请求可以组合成对设备的单个I/O请求。转移是不确定的大小。
Blk_read/s	指示从设备读取的数据量，以每秒多个块表示。块等效于内核2.4及更高版本的扇区，因此其大小为512字节。对于较老的核，块的大小是不确定的。
Blk_wrtn/s	指示写入设备的数据量，以每秒多个块表示。
Blk_read	读入的总块数
Blk_wrtn	写入的总块数
kB_read/s	指示从设备读取的数据量，以每秒千字节表示。
kB_wrtn/s	指示从设备写入的数据量，以每秒千字节表示。
kB_read	读取的总量，kb
kB_wrtn	写入的总量，kb
MB_read/s	指示写入设备的数据量，以每秒兆字节表示。
MB_wrtn/s	指示读取设备的数据量，以每秒兆字节表示。
MB_read	读取的总量，Mb

报告内容	与八的总重， MB	说明
rrqm/s	每秒合并到设备的读取请求数。	
wrqm/s	每秒合并到设备的写入请求数。	
r/s	每秒向设备发出的读取请求数。	
w/s	每秒向设备发出的写入请求数。	
rsec/s	每秒从设备读取的扇区数。	
wsec/s	每秒从设备写入的扇区数。	
rkB/s	每秒从设备读取的千字节数。	
wkB/s	每秒从设备写入的千字节数。	
rMB/s	每秒从设备读取的兆字节数。	
wMB/s	每秒从设备写入的兆字节数。	
avgrq-sz	向设备发出的请求的平均大小(按扇区)	
avgqu-sz	向设备发出的请求的平均队列长度。	
await	向要服务的设备发出I/O请求的平均时间(毫秒)。这包括请求在队列中花费的时间和服务它们的时间。	
svctm	向设备发出的I/O请求的平均服务时间(毫秒)。警告！不要再相信这个领域了。此字段将在以后的sysstat版本中删除。	
%util	向设备发出I/O请求的CPU时间百分比(设备的带宽利用率)。当此值接近100%时，设备饱和发生。	

3) NFS报告

NetworkFilessystem(NFS)报告为每个挂载的网络文件系统提供统计信息。报告显示了以下领域：

报告内容	说明
Filesystem:	此列显示NFS服务器的主机名，后面是冒号，以及安装网络文件系统的目录名。
rBlk_nor/s	指示应用程序通过Read(2)系统调用接口读取的块数。块的大小为512字节。
wBlk_nor/s	指示应用程序通过写(2)系统调用接口编写的块数。块的大小为512字节。
rBlk_dir/s	指示从使用O_DIRECT标志打开的文件中读取的块数。
wBlk_dir/s	指示写入使用O_DIRECT标志打开的文件的块数。
rBlk_svr/s	指示NFS客户端通过NFS读取请求从服务器读取的块数。
wBlk_svr/s	指示NFS客户端通过NFS读取请求从服务器写入的块数。
rkB_nor/s	指示应用程序通过Read(2)系统调用接口读取的千字节数。
wkB_nor/s	指示应用程序通过write(2)系统调用接口编写的千字节数。
rkB_dir/s	指示从使用O_DIRECT标志打开的文件中读取的千字节数。
wkB_dir/s	指示写入到使用O_DIRECT标志打开的文件中的千字节数。
rkB_svr/s	指示NFS客户端通过NFS读取请求从服务器读取的千字节数。
wkB_svr/s	指示NFS客户端通过NFS读取请求从服务器写入的千字节数。
rMB_nor/s	指示应用程序通过Read(2)系统调用接口读取的兆字节数。
wMB_nor/s	指示应用程序通过write(2)系统调用接口编写的兆字节数。
rMB_dir/s	指示从使用O_DIRECT标志打开的文件中读取的兆字节数。
wMB_dir/s	指示写入到使用O_DIRECT标志打开的文件中的兆字节数。
rMB_svr/s	指示NFS客户端通过NFS读取请求从服务器读取的兆字节数。
wMB_svr/s	指示NFS客户端通过NFS读取请求从服务器写入的兆字节数。
ops/s	指示每秒向文件系统发出的操作数。
rops/s	指示每秒向文件系统发出的“读”操作数。
wops/s	指示每秒向文件系统发出的“写”操作数。

环境变量

iostat命令考虑了以下环境变量：

S_TIME_FORMAT，如果存于此变量，且其值为ISO，则在报表标题中打印日期时，将忽略当前区域设置。
iostat命令将使用ISO 8601格式(YYYY-MM-DD)。选项-t显示的时间戳也将符合ISO 8601格式。

举例

```
iostat      # 为所有CPU和设备显示自启动报告以来的单个历史记录  
iostat -d 2 # 以两秒钟间隔显示连续设备报告  
iostat -d 2 6 # 对所有设备每隔两秒钟播放六次报告  
iostat -x hda hdb 2 6 # 以两秒钟的间隔显示六份扩展统计报告，用于设备HDA和HDB。  
iostat -p sda 2 6   # 为设备SDA及其所有分区(sda 1等)以两秒钟间隔显示六个报告。
```

sh

显示cpu情况

```
[sogrey@bogon ~]$ iostat -c  
Linux 3.10.0-862.14.1.0.h209.eulerosv2r7.x86_64 (bogon) 2021年06月29日 _x86_64_ (1 CPU)  
  
avg-cpu: %user  %nice %system %iowait  %steal    %idle  
      51.41     0.04   15.55     0.50     0.00   32.50  
  
Device:          tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn  
[sogrey@bogon ~]$
```

sh

显示nfs系统状态

```
[sogrey@bogon ~]$ iostat -n -h -t # 显示nfs状态，要求显示出时间  
用法: iostat [ 选项 ] [ <时间间隔> [ <次数> ] ]  
选项:  
[ -c ] [ -d ] [ -h ] [ -k | -m ] [ -N ] [ -t ] [ -V ] [ -x ] [ -y ] [ -z ]  
[ -j { ID | LABEL | PATH | UUID | ... } ]  
[ [ -T ] -g <用户组名> ] [ -p [ <设备> [,...] | ALL ] ]  
[ <设备> [...] | ALL ]  
[sogrey@bogon ~]$
```

sh

ip - 网络配置工具

ip指令可以显示或操作路由、网路设备，设置路由策略和通道。

ip命令用来显示或操纵Linux主机的路由、网络设备、策略路由和隧道，是Linux下较新的功能强大的网络配置工具。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ip [选项] OBJECT COMMAND [help]
```

sh

OBJECT对象可以是：link，网络设备；addr，设备的协议地址；route，路由表；rule，策略；neigh，arp缓存；tunnel，ip通道；maddr，多播地址；mroute，多播路由

COMMAND是操作命令，不同的对象有不同的命令配置。

link对象支持的命令：set、show。

addr对象支持的命令：add、del、flush、show。

route对象支持的命令：list、flush、get、add、del、change、append、replace、monitor。

rule对象支持的命令：list、add、del、flush。

neigh对象支持的命令：add、del、change、replace、show、flush。

tunnel对象支持的命令：add、change、del、show。

maddr支持的命令：add、del。

mroute支持的命令：show

选项

```
-s, -stats, -statistics      # 显示详细的信息  
-f, -family                 # 指定协议类型  
-4                           # 等同-family inet  
-6                           # 等同-family inet6  
-0                           # 等同-family link  
-o, -oneline                # 每条记录输出一行  
-r, -resolve                 # 使用系统名字解析DNS  
  
--help                       # 显示帮助文档  
--version                    # 显示命令版本信息
```

sh

ip link---网络设备配置

链路是一种网络设备，相应的命令显示和改变设备的状态。

1) ip link set， 改变设备属性

dev NAME (default)， NAME指定要操作的网络设备。配置SR-IOV虚拟功能(VF)设备时，此关键字应指定关联的物理功能(PF)设备。

up， **down**， 改变设备的状态，开或者关。

arp on， **arp off**， 更改设备的NOARP标志。

multicast on， **multicast off**， 更改设备的MULTICAST标志。

dynamic on， **dynamic off**， 更改设备的DYNAMIC标志。

name NAME， 更改设备的名字，如果设备正在运行或者已经有一个配置好的地址，那么操作无效。

txqueuelen NUMBER， **txqlen** NUMBER， 更改设备发送队列的长度。

mtu NUMBER， 更改设备MTU。

address LLADDRESS， 更改接口的站点地址

broadcast LLADDRESS， **brd** LLADDRESS， **peer** LLADDRESS， 当接口为POINTOPOINT时，更改链路层广播地址或对等地址。

netns PID， 将设备移动到与进程PID关联的网络命名空间

alias NAME， 给设备一个符号名以便于参考

vf NUM， 指定要配置的虚拟功能设备。必须使用**dev**参数指定关联的pf设备。

警告：如果请求更改多个参数，则在任何更改失败后立即中止IP。这是IP能够将系统移动到不可预测状态的唯一情况。解决方案是避免使用一个ip链路集调用来更改多个参数。

2) ip link show， 显示设备属性

dev NAME (default)， NAME指定要显示的网络设备。如果省略此参数，则列出所有设备。

up， 只显示运行的设备。

ip address---协议地址管理

该地址是附加到网络设备上的协议(IP或IPv 6)地址。每个设备必须至少有一个地址才能使用相应的协议。可以将几个不同的地址附加到一个设备上。这些地址不受歧视，因此别名一词不太适合它们，我们在本文件中也没有使用它。**ip addr** 命令显示地址及其属性，添加新地址并删除旧地址。

1) ip address add， 增加新的协议地址

dev NAME， 要向其添加地址的设备的名称。

local ADDRESS (default)， 接口的地址。地址的格式取决于协议。它是一个用于IP的虚线四边形和一系列十六进制半字，用冒号分隔用于IPv 6。地址后面可以是斜杠和十进制数，它们编码网络前缀长度。

peer ADDRESS, 点对点接口的远程端点的地址。同样, 地址后面可以是斜杠和十进制数, 编码网络前缀长度。如果指定了对等地址, 则本地地址不能具有前缀长度。网络前缀与对等端相关联, 而不是与本地地址相关联。

broadcast ADDRESS, 接口的广播地址。可以使用特殊符号“和”-代替广播地址。在这种情况下, 通过设置/重置接口前缀的主机位来导出广播地址。

label NAME, 每个地址都可以用标签字符串标记。为了保持与Linux2.0网络别名的兼容性, 此字符串必须与设备名称重合, 或者必须以设备名后跟冒号作为前缀。

scope SCOPE_VALUE, 地址有效的区域的范围。可用的作用域列在文件“/etc/iproute2/rt_scopes”中。预定义的范围值是:

- I) **global**, 地址全局有效。
- II) **site**, (仅IPv 6)该地址为站点本地地址, 即该地址在此站点内有效。
- III) **link**, 该地址是本地链接, 即它仅在此设备上有效。
- IV) **host**, 该地址仅在此主机内有效。

2) **ip address delete**, 删除协议地址

Arguments: 与“**ip addr add**”的参数一致。设备名称是必需的参数。其余的都是可选的。如果没有提供参数, 则删除第一个地址。

3) **ip address show**, 显示协议地址

dev NAME (default), 设备名字

scope SCOPE_VAL, 仅列出具有此作用域的地址。

to PREFIX, 仅列出匹配PREFIX的地址。

label PATTERN, 只列出与模式匹配的标签的地址。

dynamic, **permanent**, 仅IPv 6)仅列出由于无状态地址配置而安装的地址, 或只列出永久(非动态)地址。

tentative, (仅IPv 6)仅列出未通过重复地址检测的地址。

deprecated, (仅IPv 6)仅列出废弃地址

primary, **secondary**, 只列出主(或辅助)地址。

4) **ip address flush**, 刷新协议地址

此命令刷新由某些条件选择的协议地址。此命令具有与Show相同的参数。不同之处在于, 当不给出参数时, 它不会运行。警告: 这个命令(以及下面描述的其他刷新命令)非常危险。如果你犯了一个错误, 它不会原谅它, 而是会残酷地清除所有的地址。

使用**-* **statistics** 选项, 命令变得详细。它打印出已删除地址的数量和为刷新地址列表而进行的轮次数。如果提供了两次此选项, 则“**ip addr flush**”也会以上一小节描述的格式转储所有已删除的地址。

ip addrlabel--协议地址标签管理

IPv 6地址标签用于RFC 3484中描述的地址选择。优先级由用户空间管理, 只有标签存储在内核中。

1) **ip addrlabel add** , 增加地址标签

prefix PREFIX, **dev** DEV, 输出接口。

label NUMBER, prefix的标签, 0xffffffff保留。

2) **ip addrlabel del** , 删除地址标签

该命令删除内核中的一个地址标签条目。参数: 与“**ip addrlabel add”**的参数一致, 但不需要标签。

3) **ip addrlabel list** , 列出地址标签

显示地址标签的内容。

4) **ip addrlabel flush** , 刷新地址标签

刷新地址标签的内容, 并且不保存默认设置。

ip neighbour---邻居/ARP表管理

邻居对象为共享相同链路的主机建立协议地址和链路层地址之间的绑定。邻接条目被组织成表。IPv4邻居表的另一个名称是ARP表。相应的命令显示邻居绑定及其属性, 添加新的邻居项并删除旧条目。

1) **ip neighbour add** , 增加邻居表

2) **ip neighbour change** , 改变已经存在的邻居表

3) **ip neighbour replace** , 增加一个表或者修改已经存在的表

这些命令创建新的邻居记录或更新现有记录。上面的三个命令使用方法如下:

to ADDRESS (default), 邻居的协议地址。它要么是IPv4, 要么是IPv6地址。

dev NAME, 连接到邻居的接口。

lladdr LLADDRESS, 邻居的链路层地址, 可以是null。

nud NUD_STATE, 邻居的状态, 可以是下面的值:

I) **permanent** , 邻居项永远有效, 只能由管理员删除。

II) **noarp** , 邻居项有效。将不会尝试验证此条目, 但可以在其生存期届满时删除该条目。

III) **reachable** , 邻居项在可达超时过期之前是有效的。

IV) **stale** , 邻居的进入是有效的, 但却是可疑的。如果邻居状态有效且此命令未更改地址, 则此选项不会更改邻居状态。

4) **ip neighbour delete** , 删除邻居表

此命令使邻居项无效。这些参数与“**ip neigh add**”相同, 只是将忽略 **lladdr** 和 **nud**。警告: 试图删除或手动更改内核创建的noarp条目可能会导致不可预测的行为。特别是, 即使在NOARP接口上, 如果地址是多播或广播的, 内核也可以尝试解析此地址。

5) **ip neighbour show** , 显示邻居表

to ADDRESS (default), 选择要列出的邻居的前缀

dev NAME, 只列出与此设备相连的邻居

unused, 只列出当前未使用的邻居

nud NUD_STATE, 只列出此状态中的相邻项。NUD_STATE接受下面列出的值或特殊值 **all** , 这意味着所有状态。此选项可能发生不止一次。如果没有此选项，则IP列出除None和noarp以外的所有条目。

6) **ip neighbour flush** , 刷新邻居表

此命令刷新相邻表，根据某些条件选择要刷新的条目。此命令具有与 **show** 相同的参数。不同之处在于，当不给出参数时，它不会运行，而要刷新的默认邻居状态不包括 **permanent** 和 **noarp** 。

ip route—路由表管理

操纵内核路由表中的路由条目保存其他网络节点的路径信息。路由类型可以是：

I) **unicast** , 路由条目描述到路由前缀所涵盖的目的地的实际路径。

II) **unreachable** , 这些目的地是无法到达的。丢弃数据包，生成不可访问的ICMP消息主机。本地发件人得到一个EHOSTUNREACH错误。

III) **blackhole** , 这些目的地是无法到达的。数据包被静默丢弃。本地发送者得到一个EINVAL错误

IV) **prohibit** , 这些目的地是无法到达的。丢弃数据包并生成ICMP消息通信，该ICMP消息通信在管理上被禁止。本地发件人得到一个EACCES错误。

V) **local** , 目的地分配给此主机。数据包被环回并在本地传送。

VI) **broadcast** , 目的地是广播地址。数据包作为链路广播发送。

VII) **throw** , 与策略规则一起使用的特殊控制路径。如果选择这样的路由，则将终止此表中的查找，假装没有找到路由。如果没有策略路由，则相当于路由表中没有路由。丢包并生成不可到达的ICMP消息网。本地发送者得到一个ENETUNREACH错误。

VIII) **nat** , 一条特殊的NAT路线。前缀覆盖的目的地被认为是虚拟地址(或外部地址)，需要在转发之前转换为真实地址(或内部地址)。选择要转换到的地址，并附带属性警告：Linux2.6中不再支持路由NAT。

IX) **via** , **anycast** , 未实现目标是分配给此主机的任意广播地址。它们主要等同于本地地址，但有一个不同之处：当将这些地址用作任何数据包的源地址时，这些地址是无效的。

multicast , 用于多播路由的一种特殊类型。它不存在于普通路由表中。

路由表：Linux-2.x可以将路由打包到从1到255范围内的数字标识的多个路由表中，或者根据文件/etc/iucte2/rt_tables的名称，默认情况下，所有普通路由都插入主表(ID 254)，内核只在计算路由时使用此表。实际上，另一个表总是存在的，这是不可见的，但更重要的是。它是本地表(ID 255)。此表由本地地址和广播地址的路由组成。内核自动维护这个表，管理员通常不需要修改它，甚至不需要查看它。使用策略路由时，多个路由表进入游戏。

1) **ip route add** , 增加路由

2) **ip route change** , 修改路由

3) **ip route replace** , 改变或者增加路由

to TYPE PREFIX (default) , 路由的目标前缀。如果省略类型，则IP采用类型单播。以上列出了其他类型的

值。前缀是一个IP或IPv 6地址，后面有斜杠和前缀长度。如果前缀的长度丢失，则IP将采用全长主机路由。还有一个特殊的前缀默认值-相当于“**IP 0/0/**或者“**to IPv6 ::/0**”。

tos TOS, **dsfield** TOS, 服务类型(TOS)密钥。该密钥没有关联的掩码，最长的匹配被理解为：首先，比较路由和数据包的TOS。如果它们不相等，那么数据包仍然可以匹配为零TOS的路由。TOS要么是8位十六进制数字，要么是“**/etc/iproute2/rt_dsfield**”中的标识符。

metric NUMBER, **preference** NUMBER, 路由的首选值。NUMBER是任意32位数。

table TABLEID, 要添加此路由的表。TABLEID可能是文件“**/etc/iproute2/rt_tables**”中的一个数字或字符串。如果省略此参数，则IP假定主表，但本地路由、广播路由和NAT路由除外，默认情况下这些路由被放入本地表中。

dev NAME, 输出设备名字。

via ADDRESS, 下一个路由器的地址。实际上，这个字段的意义取决于路由类型。对于普通单播路由，它要么是真正的下一跳路由器，要么是安装在BSD兼容模式下的直接路由，它可以是接口的本地地址。对于NAT路由，它是已翻译的IP目的地块的第一个地址。

src ADDRESS, 发送到路由前缀所涵盖的目的地时要首选的源地址。

realm REALMID, 指定此路由的域。REALMID可能是文件“**/etc/iproute2/rt_realms**”中的一个数字或字符串。

mtu MTU, **mtu lock** MTU, 沿着到达目的地的路径的MTU。如果未使用修饰符锁，则由于路径MTU发现，内核可能更新MTU。如果使用修饰符锁，则不会尝试路径MTU发现，所有数据包都将在IPv4情况下不使用DF位发送，或者在IPv6中碎片到MTU。

window NUMBER, TCP向这些目的地广告的最大窗口，以字节为单位。它限制了TCP对等点允许发送给我们的最大数据突发。

rtt TIME, 最初的RTT('往返时间')估计。如果没有指定后缀，则单元是直接传递给路由代码的原始值，以保持与以前版本的兼容性。否则，如果使用s、sec或secs后缀指定秒，使用ms、msec或msecs指定毫秒。

rttvar TIME (2.3.15+ only), 初始RTT方差估计。值与上述RTT指定的值相同。

rto_min TIME (2.6.23+ only), 与此目标通信时要使用的最小TCP重传超时。值与上述RTT指定的值相同。

ssthresh NUMBER (2.3.15+ only), 初始慢启动阈值的估计。

cwnd NUMBER (2.3.15+ only), 阻塞窗口的夹子。如果不使用锁标志，则忽略它

initcwnd NUMBER, TCP连接的最大初始拥塞窗口(CWND)大小。

inirwnd NUMBER (2.6.33+ only), 连接到此目标的初始接收窗口大小。实际窗口大小是此值乘以连接的MSS。默认值为零，意味着使用慢速开始值。

advmss NUMBER (2.3.15+ only), MSS("最大段大小")在建立TCP连接时向这些目的地做广告。如果未给出，Linux将使用从第一跳设备MTU中计算出来的默认值。(如果到达这些目的地的路径是不对称的，则这种猜测可能是错误的。)

reordering NUMBER (2.3.15+ only), 到达此目标的路径上的最大重排序。如果未给出，Linux将使用sysctl变量“**net/ipv4/tcp_reordering**”所选择的值。

nexthop NEXTHOP, 多径路径的下一个。NEXTHOP是一个复杂的值，其语法类似于顶级参数列表：

I) **via** ADDRESS, 下一个路由器。

II) **dev** NAME, 输出设备

III) **weight** NUMBER, 是反映其相对带宽或质量的多径路由的此元素的权重。

scope SCOPE_VAL, 路由前缀所涵盖的目的地的范围。SCOPE_VAL可以是文件“*/etc/iproute2/rt_scopes*”中的一个数字或字符串。如果省略此参数，则IP假定所有网关单播路由的作用域全局、直接单播和广播路由的范围链接以及本地路由的范围主机。

protocol RTPROTO, 此路由的路由协议标识符。RTPROTO可以是文件“*/etc/iproute2/rt_protos*”中的一个数字或字符串。如果未给出路由协议ID，则IP假定协议启动(即假定路由是由不了解他们正在做的事情的人添加的)。一些协议值有固定的解释：

I) **redirect**, 该路由是由于icmp重定向而安装的。

II) **kernel**, 该路由是由内核在自动配置期间安装的。

III) **boot**, 该路由是在启动过程中安装的。如果路由守护进程启动，它将清除所有这些守护进程。

IV) **static**, 管理员安装了该路由以覆盖动态路由。路由守护进程将尊重它们，甚至可能会向其对等方发布广告。

V) **ra**, 路由是由路由器发现协议安装的。

onlink, 假装Nextthop直接连接到此链接，即使它不匹配任何接口前缀。

equalize, 允许在多径路由上逐包随机化。如果没有这个修饰符，路由将被冻结到一个选定的下一个，这样负载拆分将只发生在每个流基上。只有当内核被修补时，均衡化才能工作。

4) **ip route delete**, 删除路由

“**ip route del**”与“**ip route add**”具有相同的参数，但它们的语义略有不同。键值(to、tos、首选项和表)选择要删除的路由。如果存在可选属性，则IP验证它们是否与要删除的路由的属性一致。如果没有找到具有给定密钥和属性的路由，则“**ip route del**”将失败

5) **ip route show**, 显示路由

to SELECTOR (default), 仅从给定的目的地范围中选择路由。SELECTOR由一个可选修饰符(**root**、**match**、**exact**)和一个前缀组成。root选择前缀不小于PREFIX的路由。例如，“**root 0/0**”选择整个路由表。match选择前缀长度不超过PREFIX的路由。例如，“**match 10.0/16**”选择10.0/16、10/8和0/0，但未选择10.1/16和10.0.0/24。exact(或仅仅前缀)选择具有此前缀的路由。如果这两个选项都没有出现，则IP假设为根0/0，即它列出了整个表。

tos TOS, 只选择具有给定tos的路由。

table TABLEID, 显示此表中的路线。默认设置是显示tablemain。TABLEID可以是实表的ID，也可以是特殊值之一：

I) **all**, 列出所有的表。

II) **cache**, 备份路由缓存。

cloned, **cached**, 列出克隆的路由，即由于某些路由属性(F.E)而从其他路由动态分叉的路由。(MTU)已更新。实际上，它等同于“**table cache**”。

from SELECTOR, 语法与 **to** 相同，但它绑定源地址范围而不是目的地。请注意，FROM选项仅适用于克

隆路由。

protocol RTPROTO，仅列出此路由的协议。

scope SCOPE_VAL，仅列出具有此范围的路由

type TYPE，只列出此类型的路由。

dev NAME，只列出经过此设备的路由

via PREFIX，只列出通过前缀选择的下一个路由器的路由

src PREFIX，只列出由前缀选择的首选源地址的路由。

realm REALMID, **realms** FROMREALM/TOREALM，只列出这些领域的路由。

6) **ip route flush**，刷新路由表

此命令刷新由某些标准选择的路由，参数具有与“**ip route show**”的参数相同的语法和语义，但是路由表没有列出，而是被清除。唯一的区别是默认操作：显示转储所有IP主路由表，但刷新打印助手页。

使用“**-statistics**”选项，命令变得详细。它打印出已删除路由的数目和刷新路由表的轮数。如果该选项被给予两次，IP路由刷新也会以上一小节描述的格式转储所有已删除的路由。

7) **ip route get**，获取一个单独的路由

此命令获取一条到达目标的路由，并按照内核所看到的那样打印其内容。

to ADDRESS (default)，目标地址。

from ADDRESS，源地址。

tos TOS, **dsfield** TOS，服务类型。

iif NAME，预期将从该包到达的设备。

oif NAME，强制将此数据包路由的输出设备。

connected，如果没有提供源地址(选项从)，则重新查找从第一次查找中接收到的源设置为首选地址的路由。如果使用策略路由，则可能是不同的路由。

请注意，此操作不等同于“**ip route show**”。**show** 显示现有路线。如果必要的话，**get** 解决它们并创建新的克隆。

ip rule---路由策略数据库管理

rule规则在路由策略数据库中控制路由选择算法。Internet中使用的经典路由算法只根据数据包的目的地地址(理论上，而不是实际中的TOS字段)进行路由决策。在某些情况下，我们希望通过不同的方式路由数据包，这不仅取决于目的地地址，还取决于其他数据包字段：源地址、IP协议、传输协议端口，甚至包有效负载。此任务称为“策略路由”。为了解决这一问题，传统的基于目标的路由表按照最长的匹配规则排序，被替换为“路由策略数据库”(RPDB)，该数据库通过执行一组规则来选择路由。

每个策略路由规则由一个选择器和一个动作谓词组成。RPDB按照增加优先级的顺序进行扫描。每个规则的选择器应用于{源地址、目标地址、传入接口、tos、fwmark}，如果选择器与数据包匹配，则执行操作。动作谓词可能会成功返回。在这种情况下，它将给出路由或故障指示，并终止RPDB查找。否则，RPDB程序将继续执行下一条规则。

语义上，自然动作是选择下一个和输出设备。在启动时，内核配置由三条规则组成的默认rpdb:

I) Priority: 0。 Selector: 匹配任何内容, Action: 查找本地路由表(ID 255)。本地表是包含本地地址和广播地址的高优先级控制路由的特殊路由表。

II) Priority: 32766。 Selector: 匹配任何内容, Action: 查找路由表主(ID 254)。主表是包含所有非策略路由的普通路由表.管理员可以删除和/或用其他规则重写此规则。

III) Priority: 32767。 Selector: 匹配任何内容, Action: 查找路由表默认值(ID 253)。默认表为空。如果没有先前的默认规则选择数据包，则保留用于某些后处理。这一规则也可以删除。

RPDB 可能包含以下类型的规则:

I) **unicast**, 该规则规定返回在规则引用的路由表中找到的路由。

II) **blackhole**, 这条规则规定要悄悄丢弃数据包。

III) **unreachable**, 该规则规定生成“网络不可达”错误。

IV) **prohibit**, 该规则规定产生“在行政上禁止通信”错误。

V) **nat**, 该规则规定将ip数据包的源地址转换为其他值。

1) **ip rule add**, 增加规则。

2) **ip rule delete**, 删除规则。

type TYPE (default), 这个规则的类型

from PREFIX, 选择要匹配的源前缀

to PREFIX, 选择要匹配的目标前缀

iif NAME, 选择要匹配的传入设备。如果接口是回送的，则该规则只匹配来自此主机的数据包。这意味着您可以为转发包和本地数据包创建单独的路由表，从而完全隔离它们。

tos TOS, **dsfield** TOS, 选择要匹配的TOS值。

fwmark MARK, 选择要匹配的fwmark值。

priority PREFERENCE, 这条规则的优先级。每个规则都应该有一个显式设置的唯一优先级值。选项、偏好和顺序是优先级的同义词。

table TABLEID, 如果规则选择器匹配，则查找路由表标识符。还可以使用查找而不是表。

realms FROM/TO, 规则匹配和路由表查找成功时要选择的区域。只有当路由没有选择任何领域时，才使用要使用的领域。

nat ADDRESS, 要翻译的IP地址块的基(用于源地址)。该地址可以是NAT地址块的开始(由NAT路由选择)，也可以是本地主机地址(甚至为零)。在最后一种情况下，路由器不会翻译数据包，而是将它们伪装成这个地址。使用map-to而不是nat意味着同样的事情。

3) **ip rule flush**, 刷新规则，还转储所有已删除的规则。

没有参数。

4) **ip rule show**, 显示规则

没有参数。

ip maddress---多播地址管理

1) **ip maddress show**， 显示多播地址

dev NAME (default)， 设备名字

2) **ip maddress add**， 增加多播地址

3) **ip maddress delete**， 删造除多播地址

这些命令附加/分离一个静态链路层多播地址，以便在接口上侦听。请注意，不可能静态地加入协议多播组。此命令仅管理链接层地址

address LLADDRESS (default)， 链路层多播地址。

dev NAME， 加入/删除多播地址的设备

ip mroute---多播路由缓存管理

mroute对象是由用户级mrouting守护进程创建的多播路由缓存条目。由于组播路由引擎当前接口的局限性，无法对多播路由对象进行管理更改，因此只能显示对象

ip mroute show， 列出mroute缓存项

to PREFIX (default)， 选择要列出的目标多播地址的前缀。

iif NAME， 接收多播数据包的接口。

from PREFIX， 选择多播路由的IP源地址的前缀

ip tunnel---通道配置

tunnel 对象是隧道，它将数据包封装在IP包中，然后通过IP基础结构发送。加密(或外部)地址族由“-f”选项指定。默认的是ipv4。

1) ip tunnel add， 增加一个新隧道

2) ip tunnel change， 修改一个已经存在的隧道

3) ip tunnel delete， 删造除隧道

name NAME (default)， 隧道设备名字。

mode MODE， 设置隧道模式。可用的模式取决于封装地址系列。IPv 4封装可用的模式：ipip、SIT、isatap和grep；IPv6封装的模式：ip6ip6、ipip6和any。

remote ADDRESS， 设置隧道的远程端点

local ADDRESS， 设置隧道数据包的固定本地地址。它必须是此主机的另一个接口上的地址。

ttl N，在隧道化的数据包上设置固定的TTL N。N是介于1-255范围内的一个数字。0是一个特殊值，意味着数据包继承TTL值。IPv 4隧道的默认值是：Inherence。IPv6隧道的默认值是：64。

tos T, dsfield T, tclass T, 在隧道数据包上设置固定的TOS(或IPv 6中的流量类)T。默认值是: inherit。

dev NAME, 将隧道绑定到设备名称, 以便隧道数据包只能通过此设备路由, 并且在到达端点的路由发生更改时无法逃逸到另一个设备。

nopmtudisc, 禁用此隧道上的路径MTU发现。默认情况下启用它。请注意, 固定的ttl与此选项不兼容: 使用固定的ttl进行隧道操作总是会使pmtu发现。

key K, ikey** K**, okey** K**, (只有GRE隧道)使用键控GRE与密钥K, K要么是一个数字或一个类似IP地址的虚线四边形。key参数设置在两个方向上使用的键。ikey和okey参数为输入和输出设置不同的键。

csum, icsum, ocsum, (只有GRE隧道)生成/要求隧道数据包的校验和。 **ocsu**m 标志计算传出数据包的校验和。 **icsum** 标志要求所有输入数据包都具有正确的校验和。 **csum** 标志等效于组合 **icsum ocsu**m 。

seq, iseq, oseq, (只有GRE隧道)序列化数据包。 **oseq** 标志允许对传出数据包进行排序。 **iseq** 标志要求对所有输入数据包进行序列化。 seq标志等效于 **iseq oseq** 组合。这不是工作。不要用它

dscp inherit, (只有IPv 6隧道)在内部和外部报头之间继承DS字段

encaplim ELIM, 设置固定的封装限制。缺省值为4

flowlabel FLOWLABEL, (只有IPv 6隧道)设置固定的流标签。

4) **ip tunnel prl**, 潜在路由器列表(只有ISATAP)

dev NAME,

prl-default ADDR,

prl-nodefault ADDR,

prl-delete ADDR,

添加或删除addr作为潜在的路由器或默认路由器

5) **ip tunnel show**, 列出隧道

没有参数

ip monitor and rtmon---状态监控

IP实用程序可以连续地监视设备、地址和路由的状态。此选项的格式略有不同。也就是说, 监视器 command是命令行中的第一个命令, 然后对象列表如下:

ip monitor [all, LISTofOBJECTS]

OBJECT-LIST是我们要监视的对象类型的列表。它可能包含链接、地址和路由。如果没有提供文件参数, 则IP将打开RTNETLINK, 倾听该参数, 并以前面部分描述的格式转储状态更改。

如果给定文件名, 则不会倾听RTNETLINK, 而是打开包含以二进制格式保存的RTNETLINK消息的文件, 并将其转储。可以使用rtmon实用程序生成这样的历史文件。此实用程序具有类似于IP监视器的命令行语法。理想情况下, 应该在发出第一个网络配置命令之前启动rtmon。例如, 如果你在一个启动脚本中插入:

rtmon file /var/log/rtmon.log

稍后您将能够查看完整的历史记录。当然，可以随时启动rtmon。它在历史记录的前面加上在启动时转储的状态快照。

ip xfrm---设置xfrm

xfrm是一个IP框架，它可以转换数据报的格式，即用某种算法对数据包进行加密。xfrm策略和xfrm状态通过模板tmpl_list相关联。该框架被用作IPsec协议的一部分。

- 1) **ip xfrm state add**， 增加新的状态
- 2) **ip xfrm state update**， 更新已经存在的状态
- 3) **ip xfrm state allocspi**， 分配SPI数值

MODE， 设置为默认传输，但可以设置为 **tunnel**， **ro** 或者 **beet**。

FLAG-LIST，包含一个或多个标志。

FLAG，可以设置为 **noecn**、**decap-dscp**、**wildrecv**

ENCAP，封装设置为封装类型ENCAP-TYPE、源端口SPORT、目标端口DPORT和OADDR。

ENCAP-TYPE，可以是 **espinudp** 或者 **espinudp-nonike**。

ALGO-LIST，包含一个或多个算法Algo，该算法依赖于Algo_type设置的算法类型。它可以使用这些算法 **enc**、**auth**、**comp**。

- 4) **ip xfrm policy add**， 增加新策略
- 5) **ip xfrm policy update**， 更新已经存在的策略
- 6) **ip xfrm policy delete**， 删除存在的策略
- 7) **ip xfrm policy get**， 过去存在的策略
- 8) **ip xfrm policy deleteall**， 删除所有的xfrm策略
- 9) **ip xfrm policy list**， 打印策略列表
- 10) **ip xfrm policy flush**， 舒心策略

dir DIR，目录可以是 **inp**、**out**、**fwd**

SELECTOR，选择将设置策略的地址。选择器由源地址和目标地址定义。

UPSPEC，由源端口 **sport**、目的端口 **dport**、**type** 和 **code** 定义。

dev DEV，指定网络设备。

index INDEX，索引策略的数量。

ptype PTYPE，默认是 **main**，可以切换为 **sub**。

action ACTION，默认是 **allow**，可以切换为 **block**。

priority PRIORITY，级别是一个数字，默认0。

LIMIT-LIST，限制以秒、字节或数据包数量为单位设置。

TMPL-LIST，模板列表基于 **ID**、**mode**、**reqid**、**level**。

ID，由源地址、目标地址、proto和spi的值指定。

XFRM_PROTO，值可以是 **esp**、**ah**、**comp**、**route2**、**hao**。

MODE，默认 **transport**，还可以是 **tunnel**，**beet**。

LEVEL，默认 **required**，还可以是 **use**。

UPSPEC，由 **sport**、**dport**、**type**、**code** 指定。

11) **ip xfrm monitor**，用于列出所有对象或定义的对象组。

xfrm monitor 可以监视所有对象或其中定义的组的策略。

ip token

IPv 6令牌化接口标识支持用于向节点分配众所周知的主机部分地址，同时仍然从路由器广告获得全局网络前缀。令牌标识符的主要目标是服务器平台，其中的地址通常是手动配置的，而不是使用DHCPv 6或SLAAC。通过使用令牌化标识符，主机仍然可以使用SLAAC确定其网络前缀，但如果其网络前缀更改，则更容易自动重新编号[1]。

1) **ip token set**，设置接口令牌

TOKEN，接口标识符令牌地址

dev DEV，网络接口

2) **ip token get**，从内核获取接口令牌

显示特定网络设备的令牌化接口标识符。参数：与“**ip token set”**的参数一致，但必须省略该令牌。

3) **ip token list**，列出所有接口令牌

列出内核中网络接口的所有令牌化接口标识符

举例

```
ip link show          # 显示网络接口信息
ip link set eth0 up      # 开启网卡
ip link set eth0 down     # 关闭网卡
ip link set eth0 promisc on   # 开启网卡的混合模式
ip link set eth0 promisc offi  # 关闭网卡的混个模式
ip link set eth0 txqueuelen 1200 # 设置网卡队列长度
ip link set eth0 mtu 1400      # 设置网卡最大传输单元
ip addr show      # 显示网卡IP信息
ip addr add 192.168.0.1/24 dev eth0 # 设置eth0网卡IP地址192.168.0.1
ip addr del 192.168.0.1/24 dev eth0 # 删除eth0网卡IP地址

ip route show # 显示系统路由
ip route add default via 192.168.1.254    # 设置系统默认路由
ip route list           # 查看路由信息
ip route add 192.168.4.0/24 via 192.168.0.254 dev eth0 # 设置192.168.4.0网段的网关为192.168.0.254
ip route add default via 192.168.0.254 dev eth0          # 设置默认网关为192.168.0.254
ip route del 192.168.4.0/24    # 删除192.168.4.0网段的网关
ip route del default        # 删除默认路由
ip route delete 192.168.1.0/24 dev eth0 # 删除路由
```

用ip命令显示网络设备的运行状态

```
[root@localhost ~]$ ip link list
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:16:3e:00:1e:51 brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:16:3e:00:1e:52 brd ff:ff:ff:ff:ff:ff
```

显示更加详细的设备信息

```
[root@localhost ~]$ ip -s link list
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    RX: bytes packets errors dropped overrun mcast
        5082831 56145 0 0 0 0
    TX: bytes packets errors dropped carrier collsns
        5082831 56145 0 0 0 0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:16:3e:00:1e:51 brd ff:ff:ff:ff:ff:ff
    RX: bytes packets errors dropped overrun mcast
        3641655380 62027099 0 0 0 0
    TX: bytes packets errors dropped carrier collsns
        6155236 89160 0 0 0 0
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:16:3e:00:1e:52 brd ff:ff:ff:ff:ff:ff
    RX: bytes packets errors dropped overrun mcast
        2562136822 488237847 0 0 0 0
    TX: bytes packets errors dropped carrier collsns
        3486617396 9691081 0 0 0 0
```

显示核心路由表

```
[root@localhost ~]$ ip route list
112.124.12.0/22 dev eth1 proto kernel scope link src 112.124.15.130
10.160.0.0/20 dev eth0 proto kernel scope link src 10.160.7.81
192.168.0.0/16 via 10.160.15.247 dev eth0
172.16.0.0/12 via 10.160.15.247 dev eth0
10.0.0.0/8 via 10.160.15.247 dev eth0
default via 112.124.15.247 dev eth1
```

sh

显示邻居表

```
[root@localhost ~]$ ip neigh list
112.124.15.247 dev eth1 lladdr 00:00:0c:9f:f3:88 REACHABLE
10.160.15.247 dev eth0 lladdr 00:00:0c:9f:f2:c0 STALE
```

sh

获取主机所有网络接口

```
ip link, grep -E '^[0-9]', awk -F: '{print $2}'
```

sh

显示设备的各种协议地址

```
[root@localhost ~]$ ip addr show      #显示设备支持的协议的地址
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:14:33:57 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.9/24 brd 192.168.1.255 scope global eth0
        inet6 fe80::a00:27ff:fe14:3357/64 scope link
            valid_lft forever preferred_lft forever
```

sh

为目标设备添加地址

```
[root@localhost ~]$ ip addr help      #查看帮助文档
Usage: ip addr {add|change|replace} IFADDR dev STRING [ LIFETIME ]
                  [ CONFFLAG-LIST]
    ip addr del IFADDR dev STRING
    ip addr {show|flush} [ dev STRING ] [ scope SCOPE-ID ]
                  [ to PREFIX ] [ FLAG-LIST ] [ label PATTERN ]
```

sh

```
[root@localhost ~]$ ip addr add 192.168.1.110 dev eth0      #给eth0添加新的ip
```

```
[root@localhost ~]$ ip addr show dev eth0      #查看eth0的地址信息，多了一个ip
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:14:33:57 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.9/24 brd 192.168.1.255 scope global eth0
        inet 192.168.1.110/32 scope global eth0
        inet6 fe80::a00:27ff:fe14:3357/64 scope link
            valid_lft forever preferred_lft forever
```


ipcalc - 简单的IP地址计算器

ipcalc命令 是一个简单的ip地址计算器，可以完成简单的IP地址计算任务。

ipcalc提供了一种计算主机IP信息的简单方法。各种选项指定ipcalc应该在标准输出上显示什么信息。可以指定多个选项。必须始终指定要操作的IP地址。大多数操作还需要一个网络掩码或CIDR前缀。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ipcalc [OPTION]... <IP address>[/prefix] [netmask]
```

sh

选项

-c, --check	# 检测ip地址
-4	# 指定ipv4
-6	# 指定ipv6
-b, --broadcast	# 显示指定ip的广播地址和网络掩码
-h, --hostname	# 显示指定ip的主机名
-m, --netmask	# 计算给定地址的掩码
-p, --prefix	# 显示给定掩码或者ip的前缀
-n, --network	# 显示给定ip和掩码的网络地址
-s, --client	# 不显示任何错误信息

sh

举例

计算网络地址

```
[root@localhost ~]$ ipcalc -n 192.168.1.9/24  
NETWORK=192.168.1.0
```

sh

计算广播地址

```
[root@localhost ~]$ ipcalc -b 192.168.1.9/24  
BROADCAST=192.168.1.255
```

sh

计算子网掩码

[root@localhost ~]\$ ipcalc -4 -m 192.168.1.9
NETMASK=255.255.255.0

sh

ipcs - 分析消息队列共享内存和信号量

ipcs命令 用于报告Linux中进程间通信设施的状态，显示的信息包括消息列表、共享内存和信号量的信息。

ipcs指令用来显示进程间通信状况。“-i”选项允许指定特定的资源id。将只打印有关此id的信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
ipcs [-asmq] [-tclup]  
ipcs [-smq] -i id  
ipcs -h
```

sh

选项

```
-i      # 显示指定id的信息  
-m      # 显示共享内存段的信息  
-q      # 显示队列信息  
-s      # 显示信号灯信息  
-a      # 显示所有信息， 默认选项  
-t      # 显示使用时间  
-p      # 显示进程pid  
-c      # 显示进程创造者  
-u      # 显示总和  
-l      # 显示限制
```

sh

举例

显示信号灯信息

sh

```
[sogrey@bogon ~]$ ipcs -s # 显示信号灯使用情况
```

----- 消息队列 -----

键	msqid	拥有者	权限	已用字节数	消息
---	-------	-----	----	-------	----

----- 共享内存段 -----

键	shmid	拥有者	权限	字节	nattch	状态
0x00000000	327680	sogrey	600	16777216	2	目标
0x00000000	360449	sogrey	600	524288	2	目标
0x00000000	458754	sogrey	600	393216	2	目标
0x00000000	622595	sogrey	600	524288	2	目标
0x00000000	720900	sogrey	600	524288	2	目标

----- 信号量数组 -----

键	semid	拥有者	权限	nsems
---	-------	-----	----	-------

```
[sogrey@bogon ~]$
```

显示共享内存使用情况

sh

```
[sogrey@bogon ~]$ ipcs -m -p # 显示共享内存信息，并且显示进程pid
```

----- 共享内存段 -----

键	shmid	拥有者	权限	字节	nattch	状态
0x00000000	327680	sogrey	600	16777216	2	目标
0x00000000	360449	sogrey	600	524288	2	目标
0x00000000	458754	sogrey	600	393216	2	目标
0x00000000	622595	sogrey	600	524288	2	目标
0x00000000	720900	sogrey	600	524288	2	目标

```
[sogrey@bogon ~]$
```

iptables - Linux上常用的防火墙软件

iptables命令是Linux上常用的防火墙软件，是netfilter项目的一部分。可以直接配置，也可以通过许多前端和图形界面配置。

iptables指令用来设置Linux内核的ip过滤规则以及管理nat功能。iptables用于在Linux内核中设置、维护和检查IPv4数据包过滤规则表。可以定义几个不同的表。每个表包含许多内置链，也可能包含用户定义的链。每个链都是一个规则列表，可以匹配一组数据包。每条规则都指定如何处理匹配的数据包。这被称为“目标”，它可能是跳转到同一表中的用户定义链。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
iptables [-t table] {-A|-D} chain rule-specification
iptables [-t table] -I chain [rulenumber] rule-specification
iptables [-t table] -R chain rulenumber rule-specification
iptables [-t table] -D chain rulenumber
iptables [-t table] -S [chain [rulenumber]]
iptables [-t table] {-F|-L|-Z} [chain [rulenumber]] [options...]
iptables [-t table] -N chain
iptables [-t table] -X [chain]
iptables [-t table] -P chain target
iptables [-t table] -E old-chain-name new-chain-name rule-specification [= [matches...]] [target = -m matchname [per-match-options] target = -j targetname [per-target-options]]
```

目前Linux内核支持3个相互独立的表：filter，过滤ip数据包；nat，配置nat功能；mangle，修改ip数据包。

filter是默认表，包含INPUT（发送给本机）、OUTPUT（本机向外发送）、FORWARD（被路由出去）三个链。

nat表包含PREROUTING（修改刚收到的数据包）、OUTPUT（在路由之前处理本机产生的数据包）、POSTROUTING（修改将要发送的数据包）三个链。

mangle表包含PREROUTING（路由之前，修改收到的包）、OUTPUT（路由之前，修改本机产生的包）、INPUT（修改发送到本机的包）、FORWARD（修改路由之后的包）、POSTROUTING（修改将被本机发送的包）五个链。

Linux系统中的内置目标包括：ACCEPT（允许数据包通过）DROP（丢弃数据包）QUEUE（传递包到用户空间）RETURN（停止向后检测其他的规则，返回之前的条用规则处）

选项

基本参数

```
-P          # 设置默认策略:iptables -P INPUT (DROP  
-F          # 清空规则链  
-L          # 查看规则链  
-A          # 在规则链的末尾加入新规则  
-I          # num 在规则链的头部加入新规则  
-D          # num 删除某一条规则  
-s          # 匹配来源地址IP/MASK, 加感叹号!"!"表示除这个IP外。  
-d          # 匹配目标地址  
-i          # 网卡名称 匹配从这块网卡流入的数据  
-o          # 网卡名称 匹配从这块网卡流出的数据  
-p          # 匹配协议,如tcp,udp,icmp  
--dport num # 匹配目标端口号  
--sport num # 匹配来源端口号
```

sh

命令

```
-t table          # 指定要管理的表  
-A, --append chain rule-specification    # 追加记录  
-D, --delete chain rule-specification     # 删除记录  
-I, --insert chain [rulenumber] rule-specification # 插入记录  
-R, --replace chain [rulenumber] rule-specification # 替换记录  
-L, --list [chain]           # 列出记录  
-S, --list-rules [chain]        # 列出已选择链的所有规则。如果没有选择任何链, 则所有链  
-F, --flush [chain]           # 删除指定的记录  
-Z, --zero [chain [rulenumber]]    # 将数据计数和字节计数清零  
-N, --new-chain chain         # 用户自定义新链  
-X, --delete-chain [chain]      # 删除用户自定义链  
-P, --policy chain target     # 为指定的链设置策略  
-E, --rename-chain old new    # 重命名链  
-h                         # 显示帮助信息
```

sh

参数

```
[!] -p, --protocol protocol          # 指定协议类型tcp、udp、icmp、all, 协议前加! 标识否定  
[!] -s, --source address[/mask][,...]   # 源地址  
[!] -d, --destination address[/mask][,...] # 目标  
-j, --jump                          # 指定跳转的目标  
-g, --goto chain                     # 这指定应在用户指定的链中继续处理。与“--jump”选项不同, 返回  
[!] -i, --in-interface name          # 接收数据包的接口名称(仅用于输入、转发和PREROUTING链的数据  
-o, --out-interface name            # 指定数据包离开的网络接口  
[!] -f, --fragment                  # 这意味着该规则仅指分段数据包的第二段和更多的片段。  
-c, --set-counters packets bytes    # 这使管理员能够初始化规则的数据包和字节计数器(在插入、追加、
```

sh

其他选项

```
-v, --verbose          # 兀长的输出, 该选项使List命令显示接口名称、规则选项(如果有有的)
-n, --numeric          # 数字输出IP地址和端口号将以数字格式打印。默认情况下, 程序将
--exact                # 扩大数字。显示数据包和字节计数器的确切值, 而不是只显示K's()
--line-numbers          # 当列出规则时, 将行号添加到每条规则的开头, 对应于该规则在链中
--modprobe=command      # 在向链中添加或插入规则时, 使用命令加载任何必要的模块(目标、

```

举例

清空当前的所有规则和计数

```
iptables -F  # 清空所有的防火墙规则
iptables -X  # 删除用户自定义的空链
iptables -Z  # 清空计数
```

配置允许ssh端口连接

```
iptables -A INPUT -s 192.168.1.0/24 -p tcp --dport 22 -j ACCEPT
# 22为你的ssh端口, -s 192.168.1.0/24表示允许这个网段的机器来连接, 其它网段的ip地址是登陆不了你的机器的。
```

允许本地回环地址可以正常使用

```
iptables -A INPUT -i lo -j ACCEPT
#本地圆环地址就是那个127.0.0.1, 是本机上使用的, 它进与出都设置为允许
iptables -A OUTPUT -o lo -j ACCEPT
```

设置默认的规则

```
iptables -P INPUT DROP # 配置默认的不让进
iptables -P FORWARD DROP # 默认的不允许转发
iptables -P OUTPUT ACCEPT # 默认的可以出去
```

配置白名单

```
iptables -A INPUT -p all -s 192.168.1.0/24 -j ACCEPT # 允许机房内网机器可以访问
iptables -A INPUT -p all -s 192.168.140.0/24 -j ACCEPT # 允许机房内网机器可以访问
iptables -A INPUT -p tcp -s 183.121.3.7 --dport 3380 -j ACCEPT # 允许183.121.3.7访问本机的3380端口
```

开启相应的服务端口

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT # 开启80端口, 因为web对外都是这个端口
iptables -A INPUT -p icmp --icmp-type 8 -j ACCEPT # 允许被ping
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT # 已经建立的连接得让它进来
```

保存规则到配置文件中

```
cp /etc/sysconfig/iptables /etc/sysconfig/iptables.bak # 任何改动之前先备份, 请保持这一优秀的习惯
iptables-save > /etc/sysconfig/iptables
cat /etc/sysconfig/iptables
```

列出已设置的规则

```
iptables -L [-t 表名] [链名]
```

- 四个表名 raw , nat , filter , mangle
- 五个规则链名 INPUT 、 OUTPUT 、 FORWARD 、 PREROUTING 、 POSTROUTING
- filter表包含 INPUT 、 OUTPUT 、 FORWARD 三个规则链

```
iptables -L -t nat          # 列出 nat 上面的所有规则
#           ^ -t 参数指定, 必须是 raw, nat, filter, mangle 中的一个
iptables -L -t nat --line-numbers # 规则带编号
iptables -L INPUT

iptables -L -nv # 查看, 这个列表看起来更详细
```

清除已有规则

```
iptables -F INPUT # 清空指定链 INPUT 上面的所有规则
iptables -X INPUT # 删除指定的链, 这个链必须没有被其它任何规则引用, 而且这条上必须没有任何规则。
# 如果没有指定链名, 则会删除该表中所有非内置的链。
iptables -Z INPUT # 把指定链, 或者表中的所有链上的所有计数器清零。
```

删除已添加的规则

```
# 添加一条规则
iptables -A INPUT -s 192.168.1.5 -j DROP
```

将所有iptables以序号标记显示, 执行:

```
iptables -L -n --line-numbers
```

比如要删除INPUT里序号为8的规则, 执行:

```
iptables -D INPUT 8
```

开放指定的端口

```
iptables -A INPUT -s 127.0.0.1 -d 127.0.0.1 -j ACCEPT          #允许本地回环接口(即运行本机访问
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT      #允许已建立的或相关连的通行
iptables -A OUTPUT -j ACCEPT           #允许所有本机向外的访问
iptables -A INPUT -p tcp --dport 22 -j ACCEPT      #允许访问22端口
iptables -A INPUT -p tcp --dport 80 -j ACCEPT      #允许访问80端口
iptables -A INPUT -p tcp --dport 21 -j ACCEPT      #允许ftp服务的21端口
iptables -A INPUT -p tcp --dport 20 -j ACCEPT      #允许FTP服务的20端口
iptables -A INPUT -j reject        #禁止其他未允许的规则访问
iptables -A FORWARD -j REJECT      #禁止其他未允许的规则访问
```

屏蔽IP

```
iptables -A INPUT -p tcp -m tcp -s 192.168.0.8 -j DROP    # 屏蔽恶意主机（比如，192.168.0.8
iptables -I INPUT -s 123.45.6.7 -j DROP                  #屏蔽单个IP的命令
iptables -I INPUT -s 123.0.0.0/8 -j DROP                #封整个段即从123.0.0.1到123.255.255.254的命令
iptables -I INPUT -s 124.45.0.0/16 -j DROP              #封IP段即从123.45.0.1到123.45.255.254的命令
iptables -I INPUT -s 123.45.6.0/24 -j DROP              #封IP段即从123.45.6.1到123.45.6.254的命令是
```

指定数据包出去的网络接口

只对 OUTPUT, FORWARD, POSTROUTING 三个链起作用。

```
iptables -A FORWARD -o eth0
```

查看已添加的规则

```
iptables -L -n -v
Chain INPUT (policy DROP 48106 packets, 2690K bytes)
pkts bytes target     prot opt in     out      source               destination
 5075  589K ACCEPT    all   --  lo      *       0.0.0.0/0            0.0.0.0/0
 191K   90M ACCEPT    tcp   --  *       *       0.0.0.0/0            0.0.0.0/0      tcp dpt:*
1499K  133M ACCEPT    tcp   --  *       *       0.0.0.0/0            0.0.0.0/0      tcp dpt:*
4364K 6351M ACCEPT   all   --  *       *       0.0.0.0/0            0.0.0.0/0      state RE
 6256  327K ACCEPT    icmp  --  *       *       0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target     prot opt in     out      source               destination

Chain OUTPUT (policy ACCEPT 3382K packets, 1819M bytes)
pkts bytes target     prot opt in     out      source               destination
 5075  589K ACCEPT    all   --  *       lo      0.0.0.0/0            0.0.0.0/0
```

启动网络转发规则

公网 210.14.67.7 让内网 192.168.188.0/24 上网

```
iptables -t nat -A POSTROUTING -s 192.168.188.0/24 -j SNAT --to-source 210.14.67.127
```

端口映射

本机的 2222 端口映射到内网 虚拟机的22 端口

```
iptables -t nat -A PREROUTING -d 210.14.67.127 -p tcp --dport 2222 -j DNAT --to-dest 192.168.188.0/24
```

字符串匹配

比如，我们要过滤所有TCP连接中的字符串 `test`，一旦出现它我们就终止这个连接，我们可以这么做：

```
iptables -A INPUT -p tcp -m string --algo kmp --string "test" -j REJECT --reject-with tcp-reset
iptables -L

# Chain INPUT (policy ACCEPT)
# target     prot opt source          destination
# REJECT     tcp   --  anywhere       anywhere        STRING match "test" ALGO name kmp
#
# Chain FORWARD (policy ACCEPT)
# target     prot opt source          destination
#
# Chain OUTPUT (policy ACCEPT)
# target     prot opt source          destination
```

阻止Windows蠕虫的攻击

```
iptables -I INPUT -j DROP -p tcp -s 0.0.0.0/0 -m string --algo kmp --string "cmd.exe"
```

防止SYN洪水攻击

```
iptables -A INPUT -p tcp --syn -m limit --limit 5/second -j ACCEPT
```

添加SECMARK记录

```
iptables -t mangle -A INPUT -p tcp --src 192.168.1.2 --dport 443 -j SECMARK --selctx system_u:object_r:myauth_packet_t
```

显示filter表的记录

```
[root@localhost ~]$ iptables -t filter -L      #显示指定表的记录
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
ACCEPT    all  --  anywhere        anywhere         state RELATED,ESTABLISHED
ACCEPT    icmp --  anywhere        anywhere
...
Chain FORWARD (policy ACCEPT)
target     prot opt source          destination
ACCEPT    all  --  anywhere        anywhere         state RELATED,ESTABLISHED
ACCEPT    icmp --  anywhere        anywhere
...
Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
```

禁止端口135的tcp数据包

```
[root@localhost ~]$ iptables -t filter -A INPUT -p tcp --dport 135 -j DROP  #添加记录，忽略135端口
[root@localhost ~]$ iptables -L      #查看表，已经加入规则
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
ACCEPT    all  --  anywhere        anywhere         state RELATED,ESTABLISHED
DROP      tcp  --  anywhere        anywhere         tcp dpt:epmap
```

禁止目标地址访问本机

```
[root@localhost ~]$ iptables -A INPUT -s 192.168.1.110 -j DROP  #禁止110地址访问本机
[root@localhost ~]$ iptables -L |grep DROP                      #查看filter表，已经添加记录
DROP      tcp  --  anywhere        anywhere         tcp dpt:epmap
DROP      all  --  192.168.1.110      anywhere
```

iptables-restore - 还原iptables表的配置

iptables-restore指令将之前备份的iptables信息还原，这个指令需要配置shell重定向功能。

iptables-restore命令 用来还原iptables-save命令所备份的iptables配置。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
iptables-restore [选项] < iptables.bak
```

sh

选项

```
-c, --counters # 还原所有数据包和字节的计数器  
-n, --noflush # 不删除之前的iptables配置
```

sh

举例

还原iptables

```
[root@localhost ~]$ iptables-restore < iptables.bak
```

sh

iptables.bak是iptables-save命令所备份的文件。

iptables-save - 备份iptables的表配置

iptables-save命令 用于将linux内核中的iptables表导出到标准输出设备，通常，使用shell中I/O重定向功能将其输出保存到指定文件中。

iptables-save指令可以将内核中当前的iptables配置导出到标准输出。通过IO重定向功能来定向输出到文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux		Arch Linux

语法

```
iptables-save [OPTION]
```

sh

选项

-c, --counters	# 指定要保存的iptables表时，保存当权的数据包计算器和字节计数器的值；
-t, --table	# 指定要保存的表的名称。
-M, --modprobe modprobe_program	# 指定调制解调器探测程序的路径。默认情况下，iptables-Save将检查“/proc/

举例

显示iptables配置

```
[root@localhost ~]$ iptables-save -c #显示计数器信息
# Generated by iptables-save v1.4.7 on Sun Sep 30 16:20:14 2018
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [1802:148673]
[14011:18173274] -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
[27:2268] -A INPUT -p icmp -j ACCEPT
[585:30820] -A INPUT -i lo -j ACCEPT
[4526:2073410] -A INPUT -i eth+ -j ACCEPT
[0:0] -A INPUT -i wlan+ -j ACCEPT
[0:0] -A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
[0:0] -A INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
```

sh

导出指定表的信息

sh

```
[root@localhost ~]$ iptables-save -t nat # 输出nat表的信息
# Generated by iptables-save v1.4.7 on Sun Sep 30 16:21:17 2018
*nat
:PREROUTING ACCEPT [6181:2947647]
:POSTROUTING ACCEPT [573:30328]
:OUTPUT ACCEPT [2466:479800]
-A POSTROUTING -o eth+ -j MASQUERADE
-A POSTROUTING -o wlan+ -j MASQUERADE
COMMIT
# Completed on Sun Sep 30 16:21:17 2018
```

备份iptables的表配置

sh

```
[root@localhost ~]$ iptables-save -t filter > iptables.bak
[root@localhost ~]$ cat iptables.bak
# Generated by iptables-save v1.3.5 on Thu Dec 26 21:25:15 2013
*filter
:INPUT DROP [48113:2690676]
:FORWARD accept [0:0]
:OUTPUT ACCEPT [3381959:1818595115]
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A OUTPUT -o lo -j ACCEPT
COMMIT
```

ispell - 检查文件中出现的拼写错误

ispell命令 用于检查文件中出现的拼写错误。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
ispell file
```

sh

举例

Ubuntu上安装 ispell:

```
sudo apt install ispell
```

sh

```
ispell test.txt
```

sh

jed - 主要用于编辑代码的编辑器

jed命令是由Slang所开发，其主要用途是编辑程序的源代码。它支持彩色语法加亮显示，可以模拟emacs, EDT, wordstar和Brief编辑器。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
jed [OPTION] [参数]
```

sh

选项

```
-2:          # 显示上下两个编辑区;  
-batch:      # 以批处理模式来执行;  
-f<函数>:   # 执行Slang函数;  
-g<行数>:   # 移到缓冲区中指定的行数;  
-i<文件>:   # 将指定的文件载入缓冲区;  
-n:          # 不要载入jed.rc配置文件;  
-s<字符串>: # 查找并移到指定的字符串。
```

sh

举例

以上下两个编辑区的方式，开启 mysource.c 原始代码文件。若要切换编辑区，可利用稍后介绍的命令，开启操作命令，开启功能表后，按 3，再按 2，即可切换编辑区：

```
jed -2 mysource.c
```

sh

操作

有些Emacs的组合键和jed菜单组合键冲突例如Alt+f在Emacs中应该是“前进一个单词”，而在jed中则是“文件菜单”想使用Emacs风格的组合键的话，编辑/usr/share/jed/lib/menus.slc找到如下段落：

```
unsetsetkey ("selectmenubar", "\em");
unsetsetkey ("@\emF", "\ef");
unsetsetkey ("@\emE", "\ee");
unsetsetkey ("@\emo", "\eo");
% Mode menu unsetsetkey ("@\emS", "\es");
unsetsetkey ("@\emB", "\eb");
unsetsetkey ("@\emi", "\ei");
unsetsetkey ("@\emH", "\eh");
unset_setkey ("@\emy", "\ey");
```

可以根据自己的需要修改，也可以简单的注释掉；使用菜单可以用F10键。

由于Jed可模拟多种编辑器，其各自按键指令也有所不同。这里以模拟 Emacs 为例，说明在编辑器中的操作方法。

文件

```
/usr/share/jed/lib/*.sl 这是默认的运行jed slang的文件。
/usr/share/jed/lib/site.sl 这是默认的启动文件。
/etc/jed.rc 这是全局系统配置文件。
~/.jedrc 这是用户配置文件。
```

jobs - 显示作业的状态

主要用途

- 显示作业的状态。
- 列出活动的作业。
- 列出停止的作业。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
jobs [-lnprs] [jobspec ...]  
jobs -x command [args]
```

sh

选项

```
-l    # 在作业信息中额外的列出PID。  
-n    # 只列出最近一次通知以来状态变更的作业。  
-p    # 只列出PID。  
-r    # 只输出处于运行状态的作业。  
-s    # 只输出处于停止状态的作业。
```

sh

返回值

返回状态为成功除非给出了非法选项、执行出现错误。

如果使用`jobs -x command [args]`形式执行，那么返回值为`command`的退出状态。

举例

```
[user2@pc] ssh 192.168.1.4
pc@192.168.1.4's password:
# 此时按下ctrl+z使得交互停止。
[1]+  Stopped                  ssh 192.168.1.4

[user2@pc] sleep 60 &
[2] 13338

[user2@pc] jobs
[1]-  Stopped                  ssh 192.168.1.4
[2]  Running                   sleep 60 &

[user2@pc] jobs -l
[1]- 12927 Stopped              ssh 192.168.1.4
[2]  13338 Running              sleep 60 &

[user2@pc] jobs -p
12927
13338

[user2@pc] jobs -s
[1]-  Stopped                  ssh 192.168.1.4

[user2@pc] jobs -r
[2]  Running                   sleep 60 &

[user2@pc] kill -9 12927
[2]  Done                      sleep 60

[user2@pc] jobs -n -l
[1]+ 12927 Killed               ssh 192.168.1.4

[user2@pc] jobs -n -l
```

joe - 强大的纯文本编辑器

joe命令 是一款功能强大的纯文本编辑器，拥有众多编写程序和文本的优良特性。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
joe [OPTION] [参数]
```

sh

选项

```
-force: 强制在最后一行的结尾处加上换行符号;  
-lines<行数>: 设置行数;  
-lightoff: 选取的区块在执行完区块命令后, 就会恢复成原来的状态;  
-autoindent: 自动缩排;  
-backpath: <目录>; 指定备份文件的目录;  
-beep: 编辑时, 若有错误即发出哔声;  
-columns<栏位>: 设置栏数;  
-csmode: 可执行连续查找模式;  
-dopadding: 是程序跟tty间存在缓冲区;  
-exask: 在程序中, 执行“Ctrl+k+x”时, 会先确认是否要保存文件;  
-force: 强制在最后一行的结尾处加上换行符号;  
-help: 执行程序时一并显示帮助;  
-keepup: 在进入程序后, 画面上方为状态列;  
-marking: 在选取区块时, 反白区块会随着光标移动;  
-mid: 当光标移出画面时, 即自动卷页, 使光标回到中央;  
-nobackups: 不建立备份文件;  
-nonotice: 程序执行时, 不显示版本信息;  
-nosta: 程序执行时, 不显示状态列;  
-noxon: 尝试取消“Ctrl+s”和“Ctrl+q”键的功能;  
-orphan: 若同时开启一个以上的文件, 则其他文件会置于独立的缓冲区, 而不会另外开启编辑区;  
-pg<行数>: 按“PageUp”或“PageDown”换页时, 所要保留前一页的行数;  
-skiptop<行数>: 不使用屏幕上方指定的行数。
```

sh

join - 两个文件中指定栏位内容相同的行连接起来

找出两个文件中相同的字段，根据相同字段合并两个文件，将合并结果显示到标准输出。

join命令 用来将两个文件中，制定栏位内容相同的行连接起来。找出两个文件中，指定栏位内容相同的行，并加以合并，再输出到标准输出设备。

pwd - 显示当前工作目录

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
join [选项] file1 file2
```

sh

选项

```
-a 1或2          # 将文件中没有相同字段的行显示出来
-e string        # 如果在文件中找不到指定的字符串，在输出中填入选项中的字符串
-I, --ignore-case # 忽略大小写
-j FIELD         # 等价于“-1 FIELD -2 FIELD”
-o format        # 用指定的格式显示结果
-t CHAR          # 指定分隔符
-v 1或2          # 和-a一样，但是只显示没有相同字段的行
-1 FIELD         # 连接文件1指定的字段
-2 FIELD         # 连接文件2指定的字段
--check-order    # 检查输入是否正确排序，即使所有输入行都是可修的。
--nocheck-order  # 不检查输入是否正确排序

--help            # 显示帮助文档
--version         # 显示命令版本信息
```

sh

举例

```
sogrey@bogon demo4]$ ls
hello.txt  test2.txt
[sogrey@bogon demo4]$ cat hello.txt test2.txt
Hello world!
I love linux
[sogrey@bogon demo4]$ join hello.txt test2.txt
[sogrey@bogon demo4]$ cat hello.txt test2.txt
Hello world!
Ok,fine.
I love linux
Hello world
HaHa
[sogrey@bogon demo4]$ join hello.txt test2.txt # 合并相同行
join: test2.txt:2: is not sorted: Hello world
[sogrey@bogon demo4]$ join hello.txt test2.txt -v 1 # 显示文件1中不没有相同字段的行
Hello world!
join: test2.txt:2: is not sorted: Hello world
Ok,fine.
[sogrey@bogon demo4]$ join hello.txt test2.txt -v 2 # 显示文件2中没有相同字段的行
I love linux
join: test2.txt:2: is not sorted: Hello world
Hello world
HaHa
[sogrey@bogon demo4]$
```

kernelversion - 打印当前内核的主版本号

`kernelversion`命令 用于打印当前内核的主版本号。

语法

```
kernelversion
```

sh

选项

无

kill - 发送信号到进程

kill指令用来向指定的进程发送一个信号

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

内建命令

语法

```
kill [-s sigspec | -n signum | -sigspec] [pid | jobspec] ...
kill -l [sigspec | exit_status]
```

sh

选项

```
-s      # 指定信号名称,如果不指定信号名称,默认的是SIGTERM
-n      # 指定信号编号
-l      # 列出信号
```

sh

下面是常用的信号。

只有第9种信号(SIGKILL)才可以无条件终止进程, 其他信号进程都有权利忽略。

```
HUP    1    终端挂断
INT    2    中断(同 Ctrl + C)
QUIT   3    退出(同 Ctrl + \)
KILL   9    强制终止
TERM   15   终止
CONT   18   继续(与STOP相反, fg/bg命令)
STOP   19   暂停(同 Ctrl + Z)
```

sh

返回值

如果成功的发送了一个信息, 那么就返回真。如果出现了错误或者指定了无效的选项, 那么久返回假。

举例

查看信号列表

```
[root@localhost ~]$ kill -l
 1) SIGHUP 2) SIGINT 3) SIGQUIT   4) SIGILL 5) SIGTRAP
 6) SIGABRT   7) SIGBUS 8) SIGFPE 9) SIGKILL  10) SIGUSR1
11) SIGSEGV  12) SIGUSR2 13) SIGPIPE   14) SIGALRM  15) SIGTERM
16) SIGSTKFLT 17) SIGCHLD 18) SIGCONT   19) SIGSTOP  20) SIGTSTP
21) SIGTTIN  22) SIGTTOU 23) SIGURG    24) SIGXCPU  25) SIGXFSZ
26) SIGVTALRM 27) SIGPROF 28) SIGWINCH  29) SIGIO   30) SIGPWR
31) SIGSYS  34) SIGRTMIN 35) SIGRTMIN+1 36) SIGRTMIN+2 37) SIGRTMIN+3
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9 56) SIGRTMAX-8 57) SIGRTMAX-7
58) SIGRTMAX-6 59) SIGRTMAX-5 60) SIGRTMAX-4 61) SIGRTMAX-3 62) SIGRTMAX-2
63) SIGRTMAX-1 64) SIGRTMAX
```

```
[root@localhost ~]$ sleep 90 &
[1] 178420

# 终止作业标识符为1的作业。
[root@localhost ~]$ kill -9 %1

[root@localhost ~]$ jobs -l
[1]+ 178420 KILLED          ssh 192.168.1.4

[root@localhost ~]$ sleep 90 &
[1] 181357

# 发送停止信号。
[root@localhost ~]$ kill -s STOP 181357

[root@localhost ~]$ jobs -l
[1]+ 181357 Stopped (signal)      sleep 90

# 发送继续信号。
[root@localhost ~]$ kill -s CONT 181357

[root@localhost ~]$ jobs -l
[1]+ 181357 Running            sleep 90 &
```

外部命令

- 发送信号到进程（可以为多个）。
- 列出信号。

该命令是GNU coreutils包中的命令，相关的帮助信息请查看man -s 1 kill或info coreutils 'kill invocation'。

启动或关闭内建命令请查看enable命令，关于同名优先级的问题请查看builtin命令的例子部分的相关讨论。

与kill命令类似的有xkill, pkill, killall等，用于不同的目的和场景。

语法

```
kill [-signal|-s signal|-p] [-q value] [-a] [--] pid|name...
kill -l [number] | -L
```

sh

选项

```
-s, --signal signal      # 要发送的信号，可能是信号名称或信号对应的数字。
-l, --list [number]       # 打印信号名称或转换给定数字到信号名称。信号名称可参考文件 (/usr/include/linux/s
-L, --table               # 和'-l'选项类似，但是输出信号名称以及信号对应的数字。
-a, --all                 # 不要限制“命令名到pid”的转换为具有与当前进程相同的UID的进程。
-p, --pid                  # 打印目标进程的PID而不发送信号。
--verbose                # 打印信号以及接收信号的PID。
-q, --queue value         # 使用sigqueue(3)而不是kill(2)。参数value是信号对应的数字。
                           # 如果接收进程已为此信号安装了处理程序将SA_SIGINFO标记为sigaction(2)，则可以获取
                           # 该数据通过siginfo_t结构的si_signval字段。
--help                    # 显示帮助信息并退出。
--version                 # 显示版本信息并退出。
```

sh

参数

n 当n大于0时，PID为n的进程接收信号。
0 当前进程组中的所有进程均接收信号。
-1 PID大于1的所有进程均接收信号。
-n 当n大于1时，进程组n中的所有进程接收信号。当给出了一个参数的形式为“-n”，想要让它表示一个进程组，那么必须首

返回值

- 0 成功。
- 1 失败。
- 64 部分成功（当指定了多个进程时）。

举例

杀死进程

sh

```
[root@localhost ~]$ ps -a      #查看活动进程
 PID TTY      TIME CMD
 3629 pts/0    00:00:00 mysqld_safe
 3737 pts/0    00:00:11 mysqld
11092 pts/1    00:00:00 man
11095 pts/1    00:00:00 sh
11096 pts/1    00:00:00 sh
11100 pts/1    00:00:00 less
11602 pts/0    00:00:00 ps
[root@localhost ~]$ kill -s SIGKILL 11100  # 杀死进程11100
[root@localhost ~]$ ps -a                  # 查看活动进程，已经杀死
 PID TTY      TIME CMD
 3629 pts/0    00:00:00 mysqld_safe
 3737 pts/0    00:00:11 mysqld
11603 pts/0    00:00:00 ps
```

killall - 使用进程的名称来杀死一组进程

killall命令 使用进程的名称来杀死进程，使用此指令可以杀死一组同名进程。我们可以使用kill命令杀死指定进程PID的进程，如果要找到我们需要杀死的进程，我们还需要在之前使用ps等命令再配合grep来查找进程，而killall把这两个过程合二为一，是一个很好用的命令。

killall可以根据名字来杀死进程，它会给指定名字的所有进程发送信息。如果没有指定信号名，则发送SIGTERM。信号可以通过名称(例如-HUP或-SIGHUP)或数字(例如-1)或选项-s来指定。如果命令名不是正则表达式(选项-r)，并且包含斜杠(/)，则将选择执行该特定文件的进程，与其名称无关。如果每个列出的命令至少有一个进程被杀死，或者没有列出命令，并且至少有一个进程符合-u和-Z搜索条件，则KILLALL返回一个零返回代码。否则KILLALL返回非零。KILLALL进程永远不会杀死自己(但可能会杀死其他KILLALL进程)。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
killall [OPTION] [name]
```

sh

选项

```
-e, --exact          # 进程需要和名字完全相符
-I, --ignore-case   # 忽略大小写
-g, --process-group # 结束进程组
-i, --interactive    # 结束之前询问
-l, --list           # 列出所有的信号名称
-q, --quite          # 进程没有结束时，不输出任何信息
-r, --regexp         # 将进程名模式解释为扩展的正则表达式。
-s, --signal          # 发送指定信号
-u, --user            # 结束指定用户的进程
-v, --verbose         # 显示详细执行过程
-w, --wait             # 等待所有的进程都结束

--help                # 显示帮助文档
-V, --version          # 显示命令版本信息
```

sh

举例

结束所有进程

```
[root@localhost ~]$ ps      #查看进程, 有3个wc进程
  PID TTY      TIME CMD
8266 pts/0    00:00:00 bash
9781 pts/0    00:00:00 wc
9784 pts/0    00:00:00 wc
9785 pts/0    00:00:00 wc
9786 pts/0    00:00:00 ps

[root@localhost ~]$ killall -9 wc      #结束所有的wc进程

[root@localhost ~]$ ps      #查看进程, wc都被杀死
  PID TTY      TIME CMD
8266 pts/0    00:00:00 bash
9788 pts/0    00:00:00 ps
[1]  已杀死          nice -n 19 wc
[2]-  已杀死          nice -n 19 wc
[3]+  已杀死          nice -n 19 wc
```

last - 列出目前与过去登入系统的用户相关信息

last命令 用于显示用户最近登录信息。单独执行last命令，它会读取/var/log/wtmp的文件，并把该文件的内容记录的登入系统的用户名单全部显示出来。

显示以前登录过的用户信息，last指令会搜索/var/log/wtmp文件（或者是经过-f选项指定的文件），然后列出文件中所有的用户信息。如果执行last指令时提示“last /var/log/wtmp: NO such file or directory”,则需要使用指令touch /var/log/wtmp手工创建此文件

lastb指令用来显示登录失败的用户信息，其用法和last一样，对应的日志文件是/var/log/btmp

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
last  [-R]  [-num]  [ -n num ]  [-adFiowx]  [ -f file ]  [ -t YYYYMMDDHHMMSS ]  [name...]  [tty...]
lastb  [-R]  [-num]  [ -n num ]  [ -f file ]  [-adFiowx]  [name...]  [tty...]
```

选项

-f 文件名	# 指定登录的日志文件（默认是/var/log/wtmp）
-num	# 指定last显示多少行信息
-n num	# 和“-num”一样
-R	# 不显示主机名字
-a	# 在最后一列显示主机名
-d	# 将非本地登录的用户ip转换成主机名
-F	# 显示所有的登录和注销时间和日期
-o	# 读取旧的日志文件
-w	# 显示用户名和域名
-x	# 显示系统关机信息和运行级别的变化信息
-t [YYYYMMDDHHMMSS]	# 显示指定时间的登录信息
[name]	# 显示指定用户的登录信息
[tty]	# 显示指定终端的登录信息, last tty1 = last 1

举例

显示最近登录的5条信息

sh

```
[sogrey@bogon ~]$ last -5
sogrey pts/0 :0 Thu Jun 24 23:32 still logged in
sogrey :0 :0 Thu Jun 24 23:30 still logged in
reboot system boot 3.10.0-862.14.1. Thu Jun 24 23:29 - 23:33 (00:04)
sogrey pts/0 :0 Wed Jun 23 23:18 - 23:35 (00:16)
sogrey :0 :0 Wed Jun 23 23:12 - crash (1+00:16)

wtmp begins Thu Dec 17 01:55:51 2020
[sogrey@bogon ~]$
```

显示用户weijie和root在8月9号的登录信息

sh

```
[sogrey@bogon ~]$ last -t 2021062500000 sogrey
sogrey pts/0 :0 Thu Jun 24 23:32 still logged in
sogrey :0 :0 Thu Jun 24 23:30 still logged in
sogrey pts/0 :0 Wed Jun 23 23:18 - 23:35 (00:16)
sogrey :0 :0 Wed Jun 23 23:12 - crash (1+00:16)
sogrey pts/1 :0 Fri Jun 18 00:26 - 00:32 (00:05)
sogrey pts/0 :0 Fri Jun 18 00:21 - 00:31 (00:09)
sogrey :0 :0 Fri Jun 18 00:18 - crash (5+22:53)
sogrey pts/0 :0 Wed Jun 16 22:55 - 23:10 (00:14)
sogrey pts/0 :0 Wed Jun 16 22:48 - 22:54 (00:05)

...
wtmp begins Thu Dec 17 01:55:51 2020
[sogrey@bogon ~]$
```

显示终端tty1的登录信息

sh

```
[sogrey@bogon ~]$ last 1 # 等同于last tty1

wtmp begins Thu Dec 17 01:55:51 2020
[sogrey@bogon ~]$ last tty1

wtmp begins Thu Dec 17 01:55:51 2020
[sogrey@bogon ~]$
```

less - 分屏上下翻页浏览文件内容

less命令 的作用与more十分相似，都可以用来浏览文字档案的内容，不同的是less命令允许用户向前或向后浏览文件，而more命令只能向前浏览。用less命令显示文件时，用PageUp键向上翻页，用PageDown键向下翻页。要退出less程序，应按Q键。

分屏显示文本文件的内容，可以向前或者向后滚动显示。LESS是一个类似于More(1)的程序，但它允许文件中的反向移动以及向前移动。此外，在启动之前，less不需要读取整个输入文件，因此对于大型输入文件，它的启动速度比vi(1)这样的文本编辑器更快。less使用术语(或某些系统的终端)，因此它可以运行在各种终端上。甚至对硬拷贝终端的支持也很有限。(在硬拷贝终端上，应在屏幕顶部打印的行以插入符号作为前缀。)

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
less [选项] files
```

sh

选项

当使用less指令之后，屏幕进入了less的命令模式，我们需要借助一些命令才能查看所有的内容。当然，less指令之前使用pageup、pagedown按钮来切换。在下面的描述中，^X表示控制-X。ESC代表逃逸键，例如，esc-v表示两个字符序列“ESC”，然后是“v”。

```
空格, f, ctrl+V, ctrl+F # 向前滚动N行，默认一个窗口(见下面的选项-z)。如果N大于屏幕大小，则只显示最终屏幕  
z # 类似于空格，但如果指定了N，则它将成为新的窗口大小  
ESC-SPACE # 像空格一样，但是滚动一个完整的屏幕，即使它在过程中到达文件末尾。  
RETURN , ^N , e , ^E , j , ^J # 向前滚动N行，默认为1。即使N大于屏幕大小，也会显示整个N行。  
d , ^D # 向前滚动N行，默认屏幕大小的一半。如果指定了N，则它将成为后续d和u命令的新默认值。  
b , ^B , ESC-v # 向后滚动N行，默认为一个窗口(见下面的选项-z)。如果N大于屏幕大小，则只显示最终屏幕。  
w # 类似esc-v，但如果指定了N，则它将成为新的窗口大小。  
y , ^Y , ^P , k , ^K # 向后滚动N行，默认为1。即使N大于屏幕大小，也会显示整个N行。警告：一些系统使用'  
u , ^U # 向后滚动N行，默认屏幕大小的一半。如果指定了N，则它将成为后续d和u命令的新默认值。  
ESC-) , RIGHTARROW # 横向右N个字符，默认为屏幕宽度的一半(参见-#选项)。如果指定了一个数字N，则它将成为 R  
ESC-( , LEFTARROW # 向左滚动N个字符，默认为屏幕宽度的一半(参见-#选项)。如果指定了一个数字N，则它将成为 L  
r , ^R , ^L # 重新绘制屏幕  
R # 重新绘制屏幕，丢弃任何缓冲输入。如果文件正在被查看时正在更改，则非常有用。  
F # 向前滚动，并在到达文件结束时继续尝试读取。通常，在文件结束时将使用此命令。这是一种监视文件尾部的方法，该  
g , < , ESC-< # 转到文件中的第N行，默认为1(文件的开头)。(警告：如果N很大，这可能会很慢。)  
G , > , ESC-> # 转到文件中的第N行，默认为文件的末尾。(警告：如果N很大，或者没有指定N，并且正在读取标准输入)。  
p , % # 转到文件中的N%位置。N应该介于0到100之间，并且可能包含小数点。  
P # 转到文件中包含字节偏移量N的行  
{ # 如果在屏幕上显示的最上线中出现左括号，则{命令将转到匹配的右括号。匹配的右括号位于屏幕的底线上。如果上  
} # 如果在屏幕上显示的底线中出现右括号，则}命令将转到匹配的左括号。匹配的左括号位于屏幕的上线上。如果上  
( # 类似{，但适用于括号而不是花括号。
```

```
) # 类似}, 但适用于括号而不是花括号。
[ # 类似{, 但适用于方括号而不是花括号。
] # 类似}, 但适用于方括号而不是花括号。
ESC-^F # 后面跟着两个字符, 动作类似于{, 但分别使用这两个字符作为开括号和尾括号。例如, “ESC^F<>”可用于向前
ESC-^B # 后面跟着两个字符, 类似于}, 但分别使用这两个字符作为开括号和尾括号。例如, “ESC^B<>”可用于返回到与
m # 后面跟着任何小写字母, 用该字母标记当前位置。
' # (单引号)后面跟着任何小写字母, 返回以前用该字母标记的位置。后面跟着另一个单引号, 返回执行最后一个“大型”和
^X^X # 和单引号一样
/pattern # 在文件中搜索包含模式的第N行。N默认为1。模式是一个正则表达式, 正如系统提供的正则表达式库所识别的
    # ^N or !, 搜索与模式不匹配的行。
    # ^E or *, 搜索多个文件。也就是说, 如果搜索到达当前文件的末尾而没有找到匹配项, # 则搜索将在命令行
    # ^F or @, 在命令行列表中的第一个文件的第一行开始搜索, 而不考虑当前屏幕上显示的# 内容或-a或-j选项
    # ^K, 高亮显示任何与当前屏幕上的模式匹配的文本, 但不要移动到第一个匹配(保持当前# 位置)。
    # ^R不要解释正则表达式元字符; 也就是说, 做一个简单的文本比较
?pattern #在文件中向后搜索包含模式的第N行, 搜索从紧接显示的上线之前的行开始。
    #^N or !, 搜索与模式不匹配的行。
    #^E or *, 搜索多个文件。也就是说, 如果搜索到达当前文件的末尾而没有找到匹配项, 则搜索将在
    #^F or @, 在命令行列表中的第一个文件的第一行开始搜索, 而不考虑当前屏幕上显示的内容或-a或
    #^K, 正向搜索
    #^R, 正向搜索
ESC-/pattern # 等价于/*
ESC-?pattern # 等价于?*
n # 重复前面的搜索, 寻找包含最后一个模式的第N行.如果前面的搜索是由^N修改的, 则对不包含模式的第N行进行搜索。
N # 重复先前的搜索, 但方向相反
ESC-n # 重复以前的搜索, 但跨越文件边界。其效果就好像先前的搜索被*修改了一样。
ESC-N # 重复先前的搜索, 但方向相反, 并跨越文件边界。
ESC-u # 撤消搜索高亮显示。关闭与当前搜索模式匹配的字符串的高亮显示。如果由于前面的ESC-u命令, 高亮显示已经关
&pattern # 只显示与模式匹配的行; 不匹配模式的行不显示。如果模式是空的(如果您键入&后面紧跟着Enter), 则关闭
:e [filename] # 检查一个新文件。如果缺少文件名, 则重新检查命令行中文件列表中的“当前”文件(请参阅下面的: n和
^X^V or E # 和: e一样
:n # 检查下一个文件(从命令行中提供的文件列表)。如果指定数字N, 则将检查第N个下一个文件。
:p # 检查命令行列表中的前一个文件。如果指定数字N, 则检查第N个前一个文件。
:x # 检查命令行列表中的第一个文件。如果指定数字N, 则检查列表中的第N个文件。
:d # 从文件列表中删除当前文件
:t # 如果当前标记有多个匹配项, 请转到下一个标记
:T # 如果当前标记有多个匹配项, 请转到前一个标记。
= , ^G , :f # 打印有关正在查看的文件的一些信息, 包括其名称、所显示的底线的行号和字节偏移量。如果可能的话
- # 后面跟着一个命令行选项字母(请参阅下面的选项), 这将更改该选项的设置, 并打印一条描述新设置的消息。如果在破
-- # 类似于“-”命令, 但采用长选项名(请参阅下面的选项), 而不是单个选项字母。键入选项名称后, 必须按“返回”。第
-+ # 后面跟着一个命令行选项字母, 这将选项重置为默认设置, 并打印一条描述新设置的消息。(“-X”命令在命令行上执
--+ # 类似“-+”命令, 但采用长选项名, 而不是单个选项字母。
-! # 后面跟着一个命令行选项字母, 这将选项重置为其默认设置的“相反”, 并打印一条描述新设置的消息。这不适用于数
--! # 就像“-!“命令, 但是使用长选项名, 而不是单个选项字母。
-_ # (下划线)后面跟着一个命令行选项字母, 它将打印一条消息, 描述该选项的当前设置。选项的设置不会更改。
__ # (双下划线)类似于_(下划线)命令, 但是使用一个长选项名, 而不是一个选项字母。键入选项名称后, 必须按“返回”
+cmd # 导致每次检查新文件时都执行指定的cmd。例如, G导致开始时显示每个文件的次数较少, 从未尾开始, 而不是从开
V # 打印less的版本
q , Q , :q , :Q , ZZ # 退出less
v # 调用编辑器编辑正在查看的当前文件。如果定义了环境变量visual, 则从环境变量获取编辑器; 如果未定义visual,
! shell-command # 调用shell来运行给定的shell命令。命令中的百分比符号(%)被替换为当前文件的名称。一个磅号(
| <m> shell-command # <m>表示任何标记字母。将输入文件的一部分输送到给定的shell命令。要管道的文件的部分位
s filename # 将输入保存到文件中。这只有在输入是管道而不是普通文件时才有效。
```



```
-a, --search-skip-screen # 搜索的时候从屏幕的最后一行开始
-b, --buffers=n          # 指定以千字节(1024字节)为单位的每个文件将使用的缓冲区空间的大小。默认情况下,
-B, --auto-buffers       # 禁止读取pipe时自动开辟内存
-c, --clear-screen        # 清屏的时候从最上面开始, 默认从最下面开始清屏
-C, --CLEAR-SCREEN       # 和“-c”一样
-d, --dumb                # 如果终端是哑的, 则-d选项将抑制通常显示的错误消息; 也就是说, 缺少一些重要功能,
-e, --quit-at-eof         # 文件内容显示完毕后, 自动退出
-E, --QUIT-AT-EOF        # 和“-E”一样
-f, --force                # 强制打开文件
-F, --quit-if-one-screen  # 如果文件一屏就可以显示, 那么自动退出
-g                         # 不特别标识使用搜索指定的关键词
-G, --HILITE-SEARCH       # -G选项禁止搜索命令找到的字符串的所有高亮显示。
--old-dot                 # 返回到屏幕行为的旧底部。这有时是可取的, 如果长线没有正确包装时, 到达终端底部,
-hn , --max-back-scroll=n # 指定要向后滚动的最大行数。如果需要向后滚动超过n行, 则将屏幕改为正向绘制。
-i, --ignore-case          # 搜索时忽略大小写
-I, --IGNORE-CASE          # 类似-i, 但搜索忽略大小写, 即使模式包含大写字母。
-jn , --jump-target=n      # 指定屏幕上要放置“目标”行的一行。目标行是由任何命令指定的行, 用于搜索模式、跳转
-J, --status-column         # 在屏幕左侧显示状态栏。Status列显示与当前搜索匹配的行。如果-w或-W选项生效, 还
-N                         # 在每一行的开头显示行号
-Q                         # 不输出警告信息
-s                         # 连续的多个空行, 仅显示一行
-S                         # 行内容大于屏幕宽度时, 不换行

?,--help                   # 显示帮助文档
--version                  # 显示命令版本信息
```

lftp - 用来登录远程ftp服务器，是一个字符界面的文件传输工具

lftp指令可以用来登录远程ftp服务器，这是一个字符界面的文件传输工具。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lftp [选项] [host]
```

sh

选项

```
-d          # 打开调试模式  
-e cmd      # 执行给定的命令，不退出  
-p port     # 使用指定的端口登录  
-u user, pass # 使用给定的用户名和密码登录  
-f file     # 执行文件中的命令  
-c cmd      # 执行给定的命令，并且退出
```

sh

举例

登录ftp服务器

```
[root@localhost ~]$ lftp 192.168.1.8:21      #登录  
lftp 192.168.1.8:~> ls                      #查看内容，已经登录成功  
drwxr-xr-x    2 0          0          4096 Aug 14 06:38 pub  
lftp 192.168.1.8:/>
```

sh

以netstat格式显示

```
[root@localhost ~]$ lftp -u david 192.168.1.8 #使用指定用户名登录
口令:                                         #密码
lftp david@192.168.1.8:~> ls             #查看内容, 登录成功
drwxrwxr-x  3 500   500  4096 Aug 21 01:10 mail
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 下载
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 公共的
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 图片
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 文档
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 桌面
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 模板
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 视频
drwxr-xr-x  2 500   500  4096 Sep 22 12:44 音乐
```

lha - 压缩或解压缩lzh格式文件

lha命令 是从lharc演变而来的压缩程序，文件经它压缩后，会另外产生具有.lzh扩展名的压缩文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lha [OPTION] [文件或目录]
```

sh

选项

```
-a或a: 压缩文件，并加入到压缩文件内。  
-a<0/1/2>/u<0/1/2> 压缩文件时，采用不同的文件头。  
-c或c: 压缩文件，重新建构新的压缩文件后，再将其加入。  
-d或d: 从压缩文件内删除指定的文件。  
-<a/c/u>d或<a/c/u>d: 压缩文件，然后将其加入，重新建构，更新压缩文件或，删除原始文件，也就是把文件移到压缩文  
-e或e: 解开压缩文件。  
-f或f: 强制执行lha命令，在解压时会直接覆盖已有的文件而不加以询问。  
-g或g: 使用通用的压缩格式，便于解决兼容性的问题。  
-<e/x>i或<e/x>i: 解开压缩文件时，忽略保存在压缩文件内的文件路径，直接将其解压后存放在现行目录下或是指定的目  
-l或l: 列出压缩文件的相关信息。  
-m或m: 此选项的效果和同时指定"-ad"选项相同。  
-n或n: 不执行指令，仅列出实际执行会进行的动作。  
-<a/u>o或<a/u>o: 采用lharc兼容格式，将压缩后的文件加入，更新压缩文件。  
-p或p: 从压缩文件内输出到标准输出设备。  
-q或q: 不显示指令执行过程。  
-t或t: 检查备份文件内的每个文件是否正确无误。  
-u或u: 更换较新的文件到压缩文件内。  
-u<0/1/2>或u<0/1/2>: 在文件压缩时采用不同的文件头，然后更新到压缩文件内。  
-v或v: 详细列出压缩文件的相关信息。  
-<e/x>w=<目的目录>或<e/x>w=<目的目录>: 指定解压缩的目录。  
-x或x: 解开压缩文件。  
-<a/u>z或<a/u>z: 不压缩文件，直接把它加入，更新压缩文件。
```

sh

举例

```
lha -a abc.lhz a.b          #压缩a.b文件，压缩后生成 abc.lhz 文件  
lha -a abc2 /home/hnlinux   #压缩目录  
lha -xiw=agis abc          #解压文件abc，到当前目录
```

sh

lilo - 安装核心载入开机管理程序

lilo命令 用于安装核心载入，开机管理程序。lilo是个Linux系统核心载入程序，同时具备管理开机的功能。单独执行lilo指令，它会读取/etc/lilo.conf配置文件，然后根据其内容安装lilo。

Linux lilo已经成为所有 Linux 发行版的标准组成部分。作为一个较老的/最老的 Linux 引导加载程序，它那不断壮大的 Linux 社区支持使它能够随时间的推移而发展，并始终能够充当一个可用的现代引导加载程序。有一些新的功能，比如增强的用户界面，以及对能够突破原来 1024-柱面限制的新 BIOS 功能的利用。

虽然 LILO 仍在不断地发展，但 LILO 工作原理的基本概念保持不变。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lilo [OPTION]
```

sh

选项

```
-b<外围设备代号>      # 指定安装lilo之处的外围设备代号;
-c                      # 使用紧致映射模式;
-C<配置文件>          # 指定lilo的配置文件;
-d<延迟时间>          # 设置开机延迟时间;
-D<识别标签>          # 指定开机后预设启动的操作系统，或系统核心识别标签;
-f<几何参数文件>      # 指定磁盘的几何参数配置文件;
-i<开机磁区文件>      # 指定欲使用的开机磁区文件，预设是/boot目录里的boot.b文件;
-I<识别标签>          # 显示系统核心存放之处;
-l                      # 产生线形磁区地址;
-m<映射文件>          # 指定映射文件;
-P<fix/ignore>        # 决定要修复或忽略分区表的错误;
-q                      # 列出映射的系统核心文件;
-r<根目录>            # 设置系统启动时欲挂入成为根目录的目录;
-R<执行指令>          # 设置下次启动系统时，首先执行的指令;
-s<备份文件>          # 指定备份文件;
-S<备份文件>          # 强制指定备份文件;
-t                      # 不执行指令，仅列出实际执行会进行的动作;
-u<外围设备代号>      # 删除lilo;
-U<外围设备代号>      # 此选项的效果和指定"-u"参数类似，当不检查时间戳记;
-v                      # 显示指令执行过程;
-V                      # 显示版本信息。
```

实例

使用 LILO 作为引导加载程序

要使用 LILO 作为引导加载程序，需要做的事情取决于是要进行全新安装还是要让已经安装的 Linux 改为使用 LILO。如果是要进行全新安装，那么直接跳转到 配置 LILO 那一节。如果已经安装了某个 Linux 发行版，那么通常可以选择安装并配置 LILO（并可以将机器引导到新的 Linux 安装）。

要将现有的 Linux 迁移到 LILO，首先必须获得最新版本的 LILO（见 参考资料）。在做任何其他事情之前，建议您确保在手边拥有一张 Linux 引导盘——如果偶而弄错了某些地方，它可以提供很大的帮助，能够恢复到初始的 Linux 配置！将 LILO 安装到系统中之后，让它接管 MBR 非常简单。以 root 用户身份输入：

```
/sbin/lilo -v -v
```

这将使用当前的 LILO 默认值，抹去 MBR 中当前所有内容。不过，请阅读 配置 LILO，以确保能够按预期引导起来。也要注意，如果想要在同一机器上运行 Windows 和 Linux，那么应该先安装 Windows OS，然后再安装 Linux OS，这样，在 Linux 安装中所选择的引导加载程序就不会被 Windows 引导加载程序所覆盖。与 Linux 引导加载程序不同，多数 Window 引导加载程序不支持引导 Linux。如果已经先安装了 Linux，那么只需要自己创建一张 Linux 引导盘，这样就可以在安装完 Windows 之后，回到 Linux 安装中并重写 MBR。

配置 LILO

LILO 的配置都是通过位于 /etc/lilo.conf 的一个配置文件来完成的。清单 1 给出了一个示例配置，使用的是我的家用机器，支持 Linux 和 Windows 机器的双重引导。了解我的工作站的基本配置，就可以想像出这些配置是如何与实际机器相关联的：

主 HDD（物理磁盘 1）上安装了 Windows XP（最初机器上只有它）。在 Linux 术语中，这个 HDD 是 /dev/hda（在 grub 术语中是 hd0,0）。

从 HDD（物理磁盘 2）上安装了 Red Hat Linux；root 分区位于这个硬盘驱动器的第三个分区，即 /dev/hdb3（在 GRUB 术语中是 hd1,3）。

lilo.conf 示例文件：

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=100
compact
default=Linux
image=/boot/vmlinuz-2.4.18-14
    label=Linux
    root=/dev/hdb3
    read-only
    password=linux
other=/dev/hda
    label=WindowsXP
```

配置文件选项说明：

- **boot=** 行告诉 LILO 在哪里安装引导加载程序。在上面的示例中，将把它安装到第一块硬盘的 MBR。也可以选择将 LILO 安装到 /dev/hdb3（示例中的 Linux 分区），这样需要向 /dev/hda 安装另一个引导加载程序，并令其指向 LILO 引导加载程序；然后只需要让 LILO 作为二级引导加载程序。通常，引导加载程序应该位于 /dev/hda。还可以将这个参数指向软盘驱动器（最常见的是 /dev/fd0），来制作 LILO 软盘引导磁盘。
- **map=** 指向引导期间 LILO 内部使用的映射文件。当使用 /sbin/lilo 命令安装 LILO 时，它会自动生成这个文件，其中包含有描述符表（还有其他内容）。建议不要改动这个文件！
- **install=** 是 LILO 在引导过程中内部使用的文件之一。它同时包含有引导加载程序的主要部分和二级部分。boot.b 文件的一个片段被写入到 MBR（引导加载程序的主要部分），它会指向那个映射，接下来指向二级引导加载程序。同样，不要改动它！
- **prompt=** 告诉 LILO 使用用户界面（本例中给出了两个选择——Linux 和 WindowsXP）。除了使用 prompt/user 界面以外，在适当情况下还可以为 Linux 内核等指定具体的参数。如果不在配置文件中指定此选项，那么 LILO 将引导到默认的 OS，不发生任何用户交互，也不会等待。（但是请注意，如果在引导时按下了 SHIFT，那么还是可以得到提示，当不想把引导加载程序暴露给普通用户时，这非常有用）。
- **timeout=** 是引导提示在自动引导默认 OS（本例中是 Linux）之前的等待时间（以十分之一秒为单位）。如果在 lilo.conf 没有指定 prompt，那么这个参数就会被忽略。
- **compact** 选项可以大大加速引导过程，它会将连续的读磁盘的请求合并为一个单独的请求。不过，这可能是一件祸福参半的事情，因为我在论坛上看到过很多帖子提到了关于此选项的问题。当希望从软盘引导时，这个选项尤其有用。
- **default=** 选项告诉 LILO 默认使用哪个映像进行引导，比如在等待超时之后。这与 lilo.conf 文件中的某个映像的 **label** 相关联。如果没有在配置文件中指定此选项，那么它将引导文件中指定的第一个映像。
- 对于允许用户引导到的每一个 Linux 版本，都应该指定 **image=** 及以下三个选项。**image** 选项指定希望引导到的内核版本。
- **label=** 标明了在运行期间希望能够从用户界面引导的不同 OS。另外，这个标签用于指定引导的默认 OS。（注意：标签名称中避免出现空格；否则，引导那个文件时会出现无法预期的错误。）
- **root=** 告诉 LILO OS 文件系统实际所在的位置。在我们的示例中为 /dev/hdb3，即第二块硬盘上的第三个分区。
- **read-only** 告诉 LILO 以只读的方式初始引导到文件系统。OS 一旦完全引导起来，就会以读写方式挂载。
- **password=** 允许您为将要引导到的特定 OS 设置口令。不幸的是，这个口令是以可读文本的方式保存在 lilo.conf 文件中，所以，所有人都能够读取它。如果需要，还可以对想要引导的每个操作系统设置口令（在我们的示例中，只为 Linux 的引导设置了一个口令）。
- **other=** 的动作类似于 **image** 和 **root** 选项的组合，但是用于除了 Linux 以外的其他操作系统。在我们的示例中，它告诉 LILO 到哪里去找到 Windows OS（位于第一块硬盘的第一个分区）。如果先安装 Windows，后安装 Linux，通常会是这样。
- **label=** 与所有其他 **label** 选项相同。

在 lilo.conf 文件中可以使用很多其他参数，不过清单 1 中的参数就足以让机器可用。要获得关于 lilo.conf 的这些以及其他参数的进一步资料，请参考手册页（man lilo.conf）。由于在引导时不会读取 lilo.conf，所以，当这个文件有改动时，需要“更新”MBR。如果不完成此步骤就重新引导，那么对 lilo.conf 的修改不会在启动中反映出来。与先前将 LILO 写入 MBR 类似，需要运行：

```
/sbin/lilo -v -v
```

-v -v 标记会为您给出非常详细的输出。当像我们那样运行 LILO 时，有很多参数可以指定。参阅手册页以获得更进一步的信息（man lilo）。

初始引导过程

当 LILO 初始引导时，它会按次序打印出每个字母——L-I-L-O。如果所有字母都显示出来，那么第一阶段引导就成功了。缺少任何内容都表示出现了问题：

L: 第一阶段引导加载程序已经被加载。如果 LILO 停止在这里，那么是在引导第二阶段引导加载程序时出现了问题。这通常会伴随有一个错误代码。在这个阶段的常见问题是介质问题，或者在 lilo.conf 文件中指定了不正确的磁盘参数。

LI: 第二阶段引导加载程序已经被加载。LILO 在此处停止表示第二阶段引导加载程序不能被执行。同样，这可能是因为出现了与只显示 L 类似的问题：正在加载，或者因 boot.b 文件被破坏、移动或删除而不能加载。

LIL: 第二阶段引导加载程序正在被执行。此时，可能会再次出现介质问题，或者映射文件（如 lilo.conf 文件中所指定的）在寻找描述符表时可能会出现问题。

LIL?: 加载到与上面相同的阶段。这通常意味着加载第二阶段引导加载程序使用了错误的地址，最常见的原因是 boot.b 所在的位置与 lilo.conf 文件所指定的不同。

LIL-: 加载到与上面相同的阶段。加载描述符表时出现问题，最常见的原因是描述符表错误。

LILO: LILO 成功被加载，没有出现任何错误。

引导时的附加配置

LILO 被成功加载后，将看到 LILO 提示符。还是使用前面的示例 lilo.conf 文件，此时将有两个选择，可能对 LILO 新手来说并不直观。首先，可以让 LILO 超时（10 秒后），这将引导 `/dev/hdb3`，即 Linux 分区。另外，可以按下 TAB 键，这将列出将要引导的操作系统选项。在我们的示例 lilo.conf 中，将得到的选项是“Linux”和“Windows”。输入哪一个，就会引导到哪个 OS。指定加载 Linux 选项，会提示输入一个口令，在本例中是 linux。如果输入的口令有误，则会返回 LILO 提示符。

不幸的是，LILO 不支持引导期间的交互式配置，所以，只能在 lilo.conf 中或者运行 `/sbin/lilo` 时指定选项。

关于第一次尝试 LILO 的最后一点建议是：我发现使用软盘引导磁盘比使用硬盘实现 LILO 配置更为安全。为此，必须在 lilo.conf 文件中使用 `boot=/dev/fd0` 替换 `boot=/dev/hda`。那样，如果弄乱了 lilo.conf 文件中的任何配置，都可以取出引导磁盘并像先前一样引导到 Linux。当使用软盘进行引导一切正常以后，可以将 lilo.conf 修改回 `boot=/dev/hda`，然后最后一次运行 `/sbin/lilo` 来上传修改。

II - 用于显示指定工作目录下之内容

Linux ls (英文全拼: list files) 命令用于显示指定工作目录下之内容 (列出目前工作目录所含之文件及子目录)。可参考 [ls命令](#)。

ll不是命令，是ls -l的别名。

ll命令列出的信息更加详细，有时间，是否可读写等信息。

ll会列出该文件下的所有文件信息，包括隐藏的文件，而ls -l只列出显式文件，说明这两个命令还是不等同的！

Ubuntu默认不支持命令 `ll`，必须用 `ls -l`，这样使用起来不是很方便。

如果要使用此命令，可以作如下修改：

- 打开 `~/.bashrc`
- 找到 `#alias ll='ls -l'`，去掉前面的 `#` 就可以了。（关闭原来的终端才能使命令生效）

这样个人用户可以使用ll命令，当切换成超级用户后，使用ll命令时提示找不到命令，那是因为你只是修改了个人用户的配置，所以，切换成root后做相同的操作即可解决问题。

启示：我们可以通过修改 `~/.bashrc` 添加任何其他的命令别名。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
ll [options]
```

sh

选项

-a 列出目录下的所有文件，包括以 . 开头的隐含文件。
 -b 把文件名中不可输出的字符用反斜杠加字符编号(就象在C语言里一样)的形式列出。
 -c 输出文件的 i 节点的修改时间，并以此排序。
 -d 将目录象文件一样显示，而不是显示其下的文件。
 -i 输出文件的 i 节点的索引信息。
 -l 列出文件的详细信息。
 -m 横向输出文件名，并以“,”作分格符。
 -n 用数字的 **UID,GID** 代替名称。
 -o 显示文件的除组信息外的详细信息。
 -p -F 在每个文件名后附上一个字符以说明该文件的类型，
 - “*”表示可执行的普通文件；
 - “/”表示目录；
 - “@”表示符号链接；
 - “|”表示FIFOs；
 - “=”表示套接字(sockets)。
 -q 用?代替不可输出的字符。
 -r 对目录反向排序。
 -s 在每个文件名后输出该文件的大小。
 -t 以时间排序。
 -u 以文件上次被访问的时间排序。
 -A 显示除 “.” 和“..”外的所有文件。
 -B 不输出以 “~” 结尾的备份文件。
 -L 列出链接文件名而不是链接到的文件。
 -N 不限制文件长度。
 -Q 把输出的文件名用双引号括起来。
 -R 列出所有子目录下的文件。
 -S 以文件大小排序。
 -X 以文件的扩展名(最后一个 . 后的字符)排序。
 -1 一行只输出一个文件。
 --color=no 不显示彩色文件名

 --help 在标准输出上显示帮助信息。
 --version 在标准输出上输出版本信息并退出。

举例

```
[sogrey@localhost 文档]$ ll
总用量 16
drwx----- 3 sogrey sogrey 4096 7月 15 00:09 baidu
-rwxr-xr-x. 1 sogrey sogrey 74 12月 19 2020 demo.sh
-rw-----. 1 sogrey sogrey 283 12月 16 2020 demo.txt
-rw-----. 1 sogrey sogrey 39 12月 17 2020 test.js
[sogrey@localhost 文档]$
```

- 第一个栏位，表示文件的属性。Linux的文件基本上分为三个属性：可读(r)，可写(w)，可执行(x)。

这里有十个格子可以添（具体程序实现时，实际上是十个bit位）。第一个字母表示文件类型，

- “-”,普通文件.
- “d”目录,字母“d”,是dirtection(目录)的缩写.
- “l”符号链接。请注意,一个目录或者说一个文件夹是一个特殊文件,这个特殊文件存放的是其他文件和文件夹的相关信息.
- “b”块设备文件。

- "c"字符设备文件。

紧接着的3*3个字符分3组，各指示此文件的读、写、执行权限，对于owner、group、others而言。因为Linux是多用户多任务系统，所以一个文件可能同时被许多人使用，所以我们一定要设好每个文件的权限，其文件的权限位置排列顺序是（以-rwxr-xr-x为例）：

```
rwx(Owner)r-x(Group)r-x(Other)
```

这个例子表示的权限是：使用者自己可读，可写，可执行；同一组的用户可读，不可写，可执行；其它用户可读，不可写，可执行。另外，有一些程序属性的执行部分不是X,而是S,这表示执行这个程序的使用者，临时可以有和拥有者一样权力的身份来执行该程序。一般出现在系统管理之类的指令或程序，让使用者执行时，拥有root身份。

- 第二个栏位，表示文件个数。如果是文件的话，那这个数目自然是1了，如果是目录的话，那它的数目就是该目录中的文件个数了。
- 第三个栏位，表示该文件或目录的拥有者。若使用者目前处于自己的Home,那这一栏大概都是它的账号名称。
- 第四个栏位，表示所属的组（group）。每一个使用者都可以拥有一个以上的组，不过大部分的使用者应该都只属于一个组，只有当系统管理员希望给予某使用者特殊权限时，才可能会给他另一个组。
- 第五栏位，表示文件大小。文件大小用byte来表示，而空目录一般都是1024byte，当然可以用其它参数使文件显示的单位不同，如使用ls -k就是用kb来显示一个文件的大小单位，不过一般我们还是以byte为主。
- 第六个栏位，表示最后一次修改时间。以“月，日，时间”的格式表示，如Aug 15 5:46表示8月15日早上5:46分。
- 第七个栏位，表示文件名。我们可以用ls -a显示隐藏的文件名。

修改文件权限用，[chmod命令](#)

常用的linux文件权限：

```
444 r--r--r--
600 rw-----
644 rw-r--r--
666 rw-rw-rw-
700 rwx-----
744 rwxr--r--
755 rwxr-xr-x
777 rwxrwxrwx
```

从左至右，1-3位数字代表文件所有者的权限，4-6位数字代表同组用户的权限，7-9数字代表其他用户的权限。

而具体的权限是由数字来表示的，读取的权限等于4，用r表示；写入的权限等于2，用w表示；执行的权限等于1，用x表示；

通过4、2、1的组合，得到以下几种权限：0（没有权限）；4（读取权限）；5（4+1 | 读取+执行）；6（4+2 | 读取+写入）；7（4+2+1 | 读取+写入+执行）

只列出子目录

```
sh
ls -F | grep /$ 或者 alias sub = "ls -F | grep /$"(linux)
ls -l | grep "^d" 或者 ls -ll | grep "^d" (Solaris)
```

计算当前目录下的文件数和目录数

下面命令可以分别计算当前目录下的文件和目录个数:

```
sh
ls -l * |grep "^-"|wc -l ---- to count files
ls -l * |grep "^d"|wc -l ----- to count dir
```

显示彩色目录列表

打开/etc/bashrc, 加入如下一行:

```
sh
alias ls="ls --color"
```

In - 用来为文件创建链接

为指定的目录或者文件创建链接，如果没有指定链接名，那么会创建一个和源文件名字一样的链接。

In命令 用来为文件创建链接，链接类型分为硬链接和符号链接两种，默认的链接类型是硬链接。如果要创建符号链接必须使用"-s"选项。

注意：符号链接文件不是一个独立的文件，它的许多属性依赖于源文件，所以给符号链接文件设置存取权限是没有意义的。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ln [OPTION]... [-T] TARGET LINK_NAME      (1st form)
ln [OPTION]... TARGET                      (2nd form)
ln [OPTION]... TARGET... DIRECTORY         (3rd form)
ln [OPTION]... -t DIRECTORY TARGET...     (4th form)
```

sh

- 第1种用法，创建一个名字为LINK_NAME的目标链接；
- 第2种用法，创建指向当前目录中目标的链接；
- 第3和第4种用法，创建指向目录中每个目标的链接。
- 默认情况下创建硬链接，使用使用 "symbolic"创建符号链接。
- 创建硬链接时，每个目标都必须存在。
- 符号链接可以保存任意文本；如果稍后解析，则相对链接将根据其父目录进行解释。

选项

```
--backup[=CONTROL]          # 为已经存在的链接创建备份
-b                           # 和“–backup”一样，但是没有参数
-d, -F, --directory          # 允许超级用户创建硬链接
-f, --force                   # 强制创建，如果已经存在，删除原来的硬链接
-i, --interactive             # 确认是否删除目的文件
-L, --logical                 # 创建硬链接到符号链接的关联
-n, --no-dereference          # 处理与某个目录的symlink的目标，就像它是一个正常文件一样
-P, --physical                 # 创建符号链接的硬链接
-s, --symbolic                  # 创建符号链接
-S, --suffix=SUFFIX            # 重写通常的备份后缀
-t, --target-directory          # 指定要创建链接的目录
-T, --no-target-directory       # 将链接作为普通文件
-v, --verbose                   # 打印每个链接文件的名字

--help                         # 显示帮助文档
--version                       # 显示命令版本信息
```

sh

- 源文件：指定链接的源文件。如果使用-s选项创建符号链接，则“源文件”可以是文件或者目录。创建硬链接时，则“源文件”参数只能是文件；
- 目标文件：指定源文件的目标链接文件。

扩展

Linux具有为一个文件起多个名字的功能，称为链接。被链接的文件可以存放在相同的目录下，但是必须有不同的文件名，而不用在硬盘上为同样的数据重复备份。另外，被链接的文件也可以有相同的文件名，但是存放在不同的目录下，这样只要对一个目录下的该文件进行修改，就可以完成对所有目录下同名链接文件的修改。对于某个文件的各链接文件，我们可以给它们指定不同的存取权限，以控制对信息的共享和增强安全性。

文件链接有两种形式，即硬链接和符号链接。

硬链接

建立硬链接时，在另外的目录或本目录中增加目标文件的一个目录项，这样，一个文件就登记在多个目录中。如图所示的m2.c文件就在目录mub1和liu中都建立了目录项。

创建硬链接后，已经存在的文件的I节点号（Inode）会被多个目录文件项使用。一个文件的硬链接数可以在目录的长列表格式的第二列中看到，无额外链接的文件的链接数为l。

在默认情况下，ln命令创建硬链接。ln命令会增加链接数，rm命令会减少链接数。一个文件除非链接数为0，否则不会从文件系统中被物理地删除。

对硬链接有如下限制：

- 不能对目录文件做硬链接。
- 不能在不同的文件系统之间做硬链接。就是说，链接文件和被链接文件必须位于同一个文件系统中。

符号链接

符号链接也称为软链接，是将一个路径名链接到一个文件。这些文件是一种特别类型的文件。事实上，它只是一个文本文件（如图中的abc文件），其中包含它提供链接的另一个文件的路径名，如图中虚线箭头所示。另一个文件是实际包含所有数据的文件。所有读、写文件内容的命令被用于符号链接时，将沿着链接方向前进来看访问实际的文件。

!符号连接

与硬链接不同的是，符号链接确实是一个新文件，当然它具有不同的I节点号；而硬链接并没有建立新文件。

符号链接没有硬链接的限制，可以对目录文件做符号链接，也可以在不同文件系统之间做符号链接。

用ln -s命令建立符号链接时，源文件最好用绝对路径名。这样可以在任何工作目录下进行符号链接。而当源文件用相对路径时，如果当前的工作路径与要创建的符号链接文件所在路径不同，就不能进行链接。

符号链接保持了链接与源文件或目录之间的区别：

- 删除源文件或目录，只删除了数据，不会删除链接。一旦以同样文件名创建了源文件，链接将继续指向该文件的新数据。
- 在目录长列表中，符号链接作为一种特殊的文件类型显示出来，其第一个字母是l。

- 符号链接的大小是其链接文件的路径名中的字节数。
- 当用ln -s命令列出文件时，可以看到符号链接名后有一个箭头指向源文件或目录，例如lrwxrwxrwx ... 14 jun 20 10:20 /etc/motd->/original_file其中，表示“文件大小”的数字“14”恰好说明源文件名original_file由14个字符构成。

举例

创建一个硬链接

```
[sogrey@bogon 文档]$ ln -v my.iso link1  
"link" => "my.iso"
```

sh

创建一个符号链接

```
[sogrey@bogon 文档]$ ln -v -s my.iso link2  
"link2" -> "my.iso"  
[sogrey@bogon 文档]$ ls -l  
总用量 1068  
-rw-r--r-- 3 root root 358400 9月 7 15:46 link1  
lrwxrwxrwx 1 root root       6 9月 10 12:13 link2 -> my.iso
```

sh

logname - 打印当前终端登录用户的名称

显示当前登录的用户名。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
logname
```

sh

选项

```
--help                          # 显示帮助文档  
--version                        # 显示命令版本信息
```

sh

举例

查看当前登录用户

```
[sogrey@bogon ~]$ logname  
sogrey  
[sogrey@bogon ~]$
```

sh

注意

注意区分 whoami 和 logname 这两个命令；比如我们以用户 root 打开的终端，然后切换到了用户 user2。此时，whoami 返回的是当前用户 user2，logname 返回的是 root，大家可以自行实践验证一下。

该命令是GNU coreutils包中的命令，相关的帮助信息请查看man -s 1 logname, info coreutils 'logname invocation'。

logout - 退出当前登录的Shell

logout命令 用于退出当前登录的Shell， logout指令让用户退出系统，其功能和login指令相对应。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
logout
```

sh

look - 显示文件中以指定字符串开头的任意行

look命令 用于显示文件中以指定字符串开头的任意行。

显示文件中以特定字符串开始的行。在look执行二进制搜索时，必须对文件中的行进行排序。如果未指定文件，则使用文件"/usr/share/dict/words"，只比较字母数字字符，忽略字母字符的大小写。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
look [选项] string files
```

sh

选项

```
-d          # 只对比数字和英文字母，其他忽略  
-f          # 忽略字符的大小写  
-a          # 使用字典文件/usr/share/dict/web2  
-t          # 指定字符串结束符  
  
--help      # 显示帮助文档  
--version   # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ cat 1.txt  
hello world,  
i love linux,  
love code.  
[sogrey@bogon newDir3]$ look love 1.txt # 显示以hello开头的行  
love code.  
[sogrey@bogon newDir3]$
```

sh

lp - 打印文件或修改排队的打印任务

lp命令用于打印文件，或者修改排队的打印任务。与lpr命令类似，lp命令既支持文件输入也支持标准输入。它与lpr的不同之处在于它有一个不同（稍微复杂点）的参数选项设置。

lp指令用来打印文件，也可以修改存在的打印任务。使用该指令可以指定打印的页码、副本等。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lp [ -E ] [ -U username ] [ -c ] [ -d destination[/instance] ] [ -h hostname[:port] ] [ -m ]
[ -o option[=value] ] [ -q priority ] [ -s ] [ -t title ] [ -H handling ] [ -P page-list ] [ -n num-copies ]

lp [ -E ] [ -U username ] [ -c ] [ -h hostname[:port] ] [ -i job-id ] [ -n num-copies ] [ -o opt
[ -q priority ] [ -t title ] [ -H handling ] [ -P page-list ]
```

选项

```
--          # 标记选项的结尾；使用它打印以(-)开头的文件。
-E          # 使用加密模式
-U username # 设置用户名
-c          # 向后提供兼容
-d destination # 目标打印机
-h hostname[:port] # 远程打印机
-i job-id    # 指定要修改的打印任务
-n          # 设置打印副本的次数1~100
-m          # 打印完成之后发送邮件
-o "name=value [name=value ...]" # 设置打印选项
-q priority   # 设置打印级别1~100, 100最大, 默认50
-s          # 静默模式
-t "name"     # 设置打印任务名字
-u username   # 以指定的名字提交作业。
-H hh:mm     # 设置打印开始时间。可以是时间格式，也可以是hold, 等待打印; immediate, 立即打印; restart,
```

举例

打印指定的文件

```
[root@localhost /]$ lpq          #查看当前打印队列
printer01 已准备就绪, 正在打印
顺序    所有者   作业    文件          总大小
active  root     2      5.c           1024 字节
1st     root     3      P1            1024 字节
[root@localhost /]$ lp -H 10:00 -q 100 /weijie/4.c  #打印文件, 指定最高级别和时间
请求 id 是 printer01-4 (1 个文件)
You have new mail in /var/spool/mail/root
[root@localhost /]$ lpq          #查看打印队列, 可以看到刚才的任务是第一个要打印的
printer01 已准备就绪, 正在打印
顺序    所有者   作业    文件          总大小
1st     root     4      4.c           1024 字节
active  root     2      5.c           1024 字节
2nd     root     3      P1            1024 字节
[root@localhost /]$
```

lpadmin - 配置CUPS套件中的打印机和类

lpadmin命令 用于配置CUPS套件中的打印机和类，也被用来设置打印服务器默认打印机。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lpadmin [OPTION] [参数]
```

sh

打印机：指定要配置的打印机的名称。

选项

```
-c      # 将打印机加入类;  
-i      # 为打印机设置“system V”风格的接口脚本;  
-m      # 从mode目录设置一个标准的“system V”接口脚本或“PPD”文件;  
-o      # 为“PPD”或服务器设置选项;  
-r      # 从类中删除打印机;  
-u      # 设置打印机用户级的访问控制;  
-D      # 为打印机提供一个文字描述;  
-E      # 允许打印机接受打印任务;  
-L      # 为打印机位置提供一个文字描述;  
-P      # 为打印机指定一个ppd描述文件;  
-p      # 指定要配置的打印机名称;  
-d      # 设置默认打印机。
```

sh

lpc - 命令行方式打印机控制程序

lpc命令 式命令行方式打印机控制程序，有5个内置命令。

lpc指令用来控制打印机，它提供了一个命令行，用户可以输出命令来控制打印机。如果命令行上没有指定命令，lpc将从标准输入中显示提示符并接受命令。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lpc [ command [ parameter(s) ] ]
```

sh

选项

```
exit, quit      # 退出打印机命令行
?, help        # 显示帮助
status         # 显示打印机状态
```

sh

举例

进入lpc命令行

```
[root@localhost /]$ lpc          #进入命令行
lpc> help           #显示帮助
命令可能是缩写。命令是:
exit    help    quit    status  ?
lpc> ?           #显示帮助
命令可能是缩写。命令是:
exit    help    quit    status  ?
lpc> status        #显示打印机状态
printer01:
    打印机在设备 &apos;ipp&apos; 上, 速度 -1
    队列已停用
    打印已启用
    1 个条目
    监控程序已存在
lpc> exit          #退出
You have new mail in /var/spool/mail/root
[root@localhost /]$
```

sh

lpq -显示打印队列中的打印任务的状态信息

lpq命令 用于显示打印队列中的打印任务的状态信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
lpq [OPTION]
```

sh

选项

```
-E      # 强制使用加密方式与服务器连接;  
-P      # 显示中的打印机上的打印队列状态; ;  
-U      # 自动可选的用户名;  
-a      # 报告所有打印机的定义任务;  
-h      # 指定打印服务器信息;  
-l      # 使用长格式输出;  
+       # 指定显示状态的间隔时间。
```

sh

lpr - 将文件发送给指定打印机进行打印

lpr命令 用于将文件发送给指定打印机进行打印，如果不指定目标打印机，则使用默认打印机。

lpr指令从来打印文件，如果没有指定文件名，那么从标准输入读取内容。CUPS提供了许多设置默认目标的方法。首先查询“LPDEST”和“PRINTER”环境变量。如果没有设置，则使用lpoptions(1)命令的当前默认集，然后使用lpadmin(8)命令进行默认设置。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lpr [ -E ] [ -H server[:port] ] [ -U username ] [ -P destination[/instance] ] [ -# num-co sh
```

选项

```
-E          # 使用加密模式
-H          # 指定远程打印服务器
-C | -J | -T "name" # 设置打印任务名字
-P destination[/instance] # 指定打印机名字
-U username # 设置别名
-# copies   # 将要打印的副本数量从1份设置为100份。
-h          # 关闭标语打印
-l          # 指定文件已经被格式化，发送的时候不应该过滤
-m          # 打印完成之后发送邮件
-o option[=value] # 设置job的选项
-p          # 指定文件应该被格式化
-q          # 等待打印
-r          # 打印之后，文件被删除
```

举例

打印指定的文件

```
[root@localhost /]$ lpr -C P1 /weijie/4.c      #打印文件，设置打印的名字
[root@localhost /]$ lpq                         #查看打印队列
printer01 已准备就绪，正在打印
顺序    所有者    作业    文件                      总大小
active   root      2       5.c                      1024 字节
1st     root      3       P1                      1024 字节
[root@localhost /]$
```


lprm - 删除打印队列中的打印任务

lprm命令 用于删除打印队列中的打印任务。尚未完成的打印机任务会被放在打印机贮列之中，这个命令可用来将尚未送到打印机的任务取消。

lprm指令用来删除当前打印队列上的任务，如果没有指定，那么就删除当前打印任务。您可以指定一个或多个职务ID编号来取消这些职务，或者使用选项"-"-取消所有作业。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
lprm [ -E ] [ -U username ] [ -h server[:port] ] [ -P destination[/instance]] [ - ] [ job ]
```

选项

-E	# 使用加密模式
-P	# 指定打印机
-h	# 指定远程服务器
-U	# 设置别名

举例

删除打印任务

```
[root@localhost /]$ lpq          #查看当前打印队列
printer01 已准备就绪, 正在打印
顺序    所有者    作业    文件          总大小
1st      root      4      4.c          1024 字节
active   root      2      5.c          1024 字节
2nd      root      3      P1          1024 字节
[root@localhost /]$ lprm 3        #删除3号任务
You have new mail in /var/spool/mail/root
[root@localhost /]$ lpq          #查看打印队列, 3号任务已经删除
printer01 已准备就绪, 正在打印
顺序    所有者    作业    文件          总大小
1st      root      4      4.c          1024 字节
active   root      2      5.c          1024 字节
[root@localhost /]#
```


Ipstat - 显示CUPS中打印机的状态信息

Ipstat命令 用于显示CUPS中打印机的状态信息。

Ipstat指令用来显示当前任务、打印机的状态。如果没有参数，那么就显示打印队列。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lpstat [OPTION]
```

sh

选项

-E	# 使用加密模式
-H	# 显示远程打印机的名字和端口
-R	# 显示任务的顺序
-U	# 设置别名
-W	# 设置要显示哪个任务
-a	# 显示允许打印的打印机队列
-c	# 显示打印机类
-d	# 显示默认打印机
-h server[:port]	# 指定备用服务器
-l	# 显示打印机、类或作业的长列表。
-o	# 显示指定打印机的队列
-p	# 显示指定打印机，无论打印机是否激活
-r	# 显示CUPS是否在运行
-s	# 显示状态总和
-t	# 显示所有的状态。等价于"-r", "-d", "-c", "-v", "-a", "-p", "-o"一起使用
-u [user(s)]	# 显示由指定用户排队的打印作业列表。如果未指定用户，则列出当前用户排队的作业。
-v [printer(s)]	# 显示打印机及其连接的设备。如果没有指定打印机，则列出所有打印机。

sh

举例

查看CUPS是否运行

```
[root@localhost /]$ lpstat -r
调度程序正在运行
You have new mail in /var/spool/mail/root
[root@localhost /]$
```

sh

ls - 显示目录内容列表

ls命令用来显示目录列表，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。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
ls [选项] [文件名...]
[-1abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ]
[-w cols]
[-T cols]
[-I pattern]
[--full-time]
[--format={long,verbose,commas,across,vertical,single-column}]
[--sort={none,time,size,extension}]
[--time={atime,access,use,ctime,status}]
[--color[={none,auto,always}]]
[--help]
[--version]
[--]
```

sh

常用命令：

```
ls # 仅列出当前目录可见文件
ls -l # 列出当前目录可见文件详细信息
ls -hl # 列出详细信息并以可读大小显示文件大小
ls -al # 列出所有文件（包括隐藏）的详细信息
ls --human-readable --size -1 -S --classify # 按文件大小排序
du -sh * | sort -h # 按文件大小排序(同上)
```

sh

选项

```
[sogrey@bogon 文档]$ ls --help
用法: ls [选项]... [文件]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
-a, --all # 不隐藏任何以. 开始的项目
-A, --almost-all # 列出除. 及.. 以外的任何项目
--author # 与-l 同时使用时列出每个文件的作者
-b, --escape # 以八进制溢出序列表示不可打印的字符
--block-size=SIZE # scale sizes by SIZE before printing them; e.g.,
```

sh

```
# '--block-size=M' prints sizes in units of
# 1,048,576 bytes; see SIZE format below
-B, --ignore-backups
-c
-C
--color[=WHEN]
-d, --directory
-D, --dired
-f
-F, --classify
--file-type
--format=WORD
--full-time
-g          # 类似-l, 但不列出所有者
--group-directories-first
-G, --no-group      # 以一个长列表的形式, 不输出组名
-h, --human-readable    # 与-l一起, 以易于阅读的格式输出文件大小
# (例如 1K 234M 2G)
--si           # 同上面类似, 但是使用1000为基底而非1024
-H, --dereference-command-line
--dereference-command-line-symlink-to-dir
--hide=PATTERN
--indicator-style=WORD
-i, --inode
-I, --ignore=PATTERN
-k, --kibibytes
-l          # 使用较长格式列出信息
-L, --dereference
--m           # 所有项目以逗号分隔, 并填满整行宽
-n, --numeric-uid-gid   # 类似 -l, 但列出UID及GID号
-N, --literal
-o          # 类似 -l, 但不列出有关组的信息
-p, --indicator-style=slash # 对目录加上表示符号"/"
-q, --hide-control-chars
--show-control-chars
-Q, --quote-name
--quoting-style=WORD
-r, --reverse
-R, --recursive
-s, --size
-S
--sort=WORD
# '---block-size=M' prints sizes in units of
# 1,048,576 bytes; see SIZE format below
# do not list implied entries ending with ~
# with -lt: sort by, and show, ctime (time of last
# modification of file status information);
# with -l: show ctime and sort by name;
# otherwise: sort by ctime, newest first
# list entries by columns
# colorize the output; WHEN can be 'never', 'auto',
# or 'always' (the default); more info below
# list directories themselves, not their contents
# generate output designed for Emacs' direvd mode
# do not sort, enable -aU, disable -ls --color
# append indicator (one of */=>@|) to entries
# likewise, except do not append '*'
# across -x, commas -m, horizontal -x, long -l,
# single-column -1, verbose -l, vertical -C
# like -l --time-style=full-iso
# group directories before files;
# can be augmented with a --sort option, but any
# use of --sort=none (-U) disables grouping
# 以一个长列表的形式, 不输出组名
# 与-l一起, 以易于阅读的格式输出文件大小
# (例如 1K 234M 2G)
# 同上面类似, 但是使用1000为基底而非1024
# follow symbolic links listed on the command line
# follow each command line symbolic link
# that points to a directory
# do not list implied entries matching shell PATTERN
# (overridden by -a or -A)
# append indicator with style WORD to entry names:
# none (default), slash (-p),
# file-type (--file-type), classify (-F)
# print the index number of each file
# do not list implied entries matching shell PATTERN
# default to 1024-byte blocks for disk usage
# 使用较长格式列出信息
# 当显示符号链接的文件信息时, 显示符号链接所指示
# 的对象而并非符号链接本身的信息
# 所有项目以逗号分隔, 并填满整行宽
# 类似 -l, 但列出UID及GID号
# 输出未经处理的项目名称 (如不特别处理控制字符)
# 类似 -l, 但不列出有关组的信息
# 对目录加上表示符号"/"
# print ? instead of nongraphic characters
# show nongraphic characters as-is (the default,
# unless program is 'ls' and output is a terminal)
# enclose entry names in double quotes
# use quoting style WORD for entry names:
# literal, locale, shell, shell-always, c, escape
# 逆序排列
# 递归显示子目录
# 以块数形式显示每个文件分配的尺寸
# sort by file size
# sort by WORD instead of name: none (-U), size (-S),
# time (-t), version (-v), extension (-X)
```

```
--time=WORD          # with -l, show time as WORD instead of default
                     # modification time: atime or access or use (-u)
                     # ctime or status (-c); also use specified time
                     # as sort key if --sort=time
--time-style=STYLE    # with -l, show times using style STYLE:
                     # full-iso, long-iso, iso, locale, or +FORMAT;
                     # FORMAT is interpreted like in 'date'; if FORMAT
                     # is FORMAT1<newline>FORMAT2, then FORMAT1 applies
                     # to non-recent files and FORMAT2 to recent files;
                     # if STYLE is prefixed with 'posix-', STYLE
                     # takes effect only outside the POSIX locale
-t                  # sort by modification time, newest first
-T, --tabsize=COLS   # assume tab stops at each COLS instead of 8
-u                  # with -lt: sort by, and show, access time;
                     # with -l: show access time and sort by name;
                     # otherwise: sort by access time
-U                  # do not sort; list entries in directory order
-v                  # natural sort of (version) numbers within text
-w, --width=COLS    # assume screen width instead of current value
-x                  # list entries by lines instead of by columns
-X                  # sort alphabetically by entry extension
-1                  # list one file per line
```

SELinux options:

```
--lcontext          # Display security context. Enable -l. Lines
                     # will probably be too wide for most displays.
-Z, --context        # Display security context so it fits on most
                     # displays. Displays only mode, user, group,
                     # security context and file name.
--scontext           # Display only security context and file name.
--help               # 显示此帮助信息并退出
--version            # 显示版本信息并退出
```

SIZE is an integer and optional unit (example: 10M is 10*1024*1024). Units are K, M, G, T, P, E, Z, Y (powers of 1024) or KB, MB, ... (powers of 1000).

使用色彩来区分文件类型的功能已被禁用，默认设置和 `--color=never` 同时禁用了它。

使用 `--color=auto` 选项，`ls` 只在标准输出被连至终端时才生成颜色代码。

`LS_COLORS` 环境变量可改变此设置，可使用 `dirCOLORS` 命令来设置。

退出状态：

- 0 正常
- 1 一般问题（例如：无法访问子文件夹）
- 2 严重问题（例如：无法使用命令行参数）

GNU coreutils online help: <<http://www.gnu.org/software/coreutils/>>
请向<http://translationproject.org/team/zh_CN.html> 报告`ls` 的翻译错误
要获取完整文档，请运行：`info coreutils 'ls invocation'`

以上信息来自 CentOS Linux 8

补充说明

在默认情况下，使用颜色来区分文件类型是禁用的，并且使用“`--color=never`”。只有当标准输出连接到终

端时，ls才会发出颜色代码。LS_COLORS环境变量可以更改设置，使用dircolors命令来设置。

大小可以是KB, 1000; K, 1024,; MB, 1000 1000; M, 1024 1024

举例

```
[sogrey@bogon 文档]$ ls  
ee.txt  
[sogrey@bogon 文档]$ ls -lh ee.txt  
-rw-----. 1 sogrey sogrey 12 1月 10 18:45 ee.txt  
[sogrey@bogon 文档]$
```

sh

lsattr - 显示指定文件或者目录的属性

RT。 显示指定文件或者目录的属性。可用于查看文件的第二扩展文件系统属性。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
lsattr [选项] file
```

sh

选项

```
-v      # 列出文件版本号  
-R      # 递归列出所有子目录中文件的属性  
-a      # 列出所有文件的属性，包含隐藏文件  
-d      # 列出目录的属性，而不是它里面内容的属性  
-V      # 显示执行过程
```

sh

举例

```
[sogrey@bogon 文档]$ lsattr -a demos
-----e---- demos/test.txt
-----e---- demos/test3.txt
-----e---- demos/.
-----e---- demos/test2.txt
-----e---- demos/test4.txt
-----e---- demos/..
[sogrey@bogon 文档]$ lsattr -a -v demos
803300458 -----e---- demos/test.txt
803300456 -----e---- demos/test3.txt
803300302 -----e---- demos/.
803300455 -----e---- demos/test2.txt
803300457 -----e---- demos/test4.txt
3339769049 -----e---- demos/..
[sogrey@bogon 文档]$ lsattr -a -V demos
lsattr 1.45.0 (6-Mar-2019)
-----e---- demos/test.txt
-----e---- demos/test3.txt
-----e---- demos/.
-----e---- demos/test2.txt
-----e---- demos/test4.txt
-----e---- demos/..
[sogrey@bogon 文档]$ lsattr -d demos
-----e---- demos
[sogrey@bogon 文档]$ lsattr -R demos
-----e---- demos/test.txt
-----e---- demos/test3.txt
-----e---- demos/test2.txt
-----e---- demos/test4.txt
[sogrey@bogon 文档]$
```

lsmod - 显示已载入系统的模块

lsmod指令用来显示已经加载的内核模块。

lsmod命令 用于显示已经加载到内核中的模块的状态信息。执行lsmod命令后会列出所有已载入系统的模块。Linux操作系统的内核具有模块化的特性，应此在编译核心时，务须把全部的功能都放入核心。您可以将这些功能编译成一个个单独的模块，待需要时再分别载入。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
lsmod
```

sh

选项

无

举例

```
[sogrey@bogon ~]$ lsmod
Module           Size  Used by
nls_utf8          16384  1
isofs              49152  1
vboxvideo          24576  0
ghash_clmulni_intel  16384  0
snd_rawmidi        36864  1 snd_seq_midi
aesni_intel        372736  0
crypto_simd         16384  1 aesni_intel
cryptd              24576  2 crypto_simd,ghash_clmulni_intel
joydev              24576  0
vboxguest           45056  5
pata_acpi           16384  0
video               49152  0
...
sogrey@sogrey-VirtualBox:~/桌面$
```

lspci - 显示当前主机的所有PCI总线信息

lspci命令 用于显示当前主机的所有PCI总线信息，以及所有已连接的PCI设备信息。

lspci是一种实用程序，用于在系统中显示有关pci总线的信息以及连接到它们的设备。

默认情况下，它显示了一个简单的设备列表。使用下面描述的选项可以请求更详细的输出或其他程序用于解析的输出。

如果要报告PCI设备驱动程序或**lspci**本身中的bug，请使用选项“**lspci-vvx**”或更好的“**lspci-vvxxxx**”的输出(不过，可能会有警告)。

输出的某些部分，特别是在高度冗长的模式下，只有经验丰富的PCI黑客才能理解Proba-Bly。有关字段的确切定义，请参阅PCI规范或header.h和/usr/include/linux/pci.h文件。

在许多操作系统上，对PCI配置空间的某些部分的访问仅限于root用户，因此对于普通用户来说，**lspci**的功能是有限的。然而，**lspci**尽力显示尽可能多的可用信息，并将所有其他信息标记为<访问拒绝>文本

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
lspci [OPTION]
```

sh

lspci命令中，我们经常会看到一些“**0:0.0**”这样格式的数字，例如“**00:01.0**”，这一个参数是总线编号，第二个是插槽编号，第三个是功能编号，它们都是十六进制的数字。

选项

基础显示模式

```
-m      # 以向后兼容并且机器可读的方式转储设备信息  
-mm     # 以机器可读的方式转储设备信息，以便脚本解析  
-t      # 以树形结构显示pci设备的层次关系，包含所有总线、桥梁、设备和它们之间的连接
```

sh

显示选项

```
-n          # 显示pci设备的厂商和设备代码  
-v          # 显示所有设备的详细信息  
-vv         # 以更加详细的方式显示设备信息  
-k          # 显示处理每个设备的内核驱动程序以及能够处理该设备的内核模块。默认情况下，当-v以正常的输出  
-x          # 显示配置空间标准部分的十六进制转储(CardBus桥的前64字节或128字节)。  
-xxx        # 显示整个PCI配置空间的十六进制转储。当您试图读取配置空间的某些部分时，只有当几个PCI设备  
-xxxx       # 显示扩展(4096字节)PCI配置空间在PCI-X2.0和PCIEexpress总线上可用的十六进制转储  
-b          # 以总线为中心的视图。显示由pci总线上的卡看到的所有irq编号和地址。注意，不是有内核看到的  
-D          # 始终显示PCI域号。默认情况下，lspci在只有域0的机器上略过它们。
```

解析ID为名称的选项

```
-n          # 将PCI供应商和设备代码显示为编号，而不是在PCI ID列表中查找它们。  
-nn         # 显示pci供应商和设备的代码和名字  
-q          # 如果在本地pci.id文件中找不到设备，则使用DNS查询中央PCI ID数据库。如果DNS查询成功，结果  
-qq         # 和“-q”一样，但是本地缓存被重置  
-Q          # 查询中央数据库，即使是本地也有缓存数据可查。如果您怀疑显示的条目是错误的，请使用此方法
```

选择设备的选项

```
-s [域]:[总线].[插槽].[功能]  # 只显示指定域中的设备(如果您的计算机有几个主机桥接器，它们可以共享公共总线编址)  
-d [厂商: 设备]               # 显示指定厂商和设备的信息，厂商号和设备号都是十六进制。
```

其他选项

```
-i <file>      # 指定pci设备编号文件，默认文件是/usr/share/hwdata/pci.ids  
-p <file>      # 使用指定文件作为PCI ID的映射文件，默认使用/lib/Module/kernel_version/Modes.pcimap  
-M             # 调用总线映射模式，它对所有pci设备，包括配置错误的桥后面的设备进行彻底扫描。此选项只在直接硬件访问时有效
```

PCI设备访问选项

```
-A <method>      # 支持多种方法来访问PCI硬件。默认情况下，它使用第一个可用的访问方法，但您可以使用此选项  
-O <param>=<value> # 库的行为由多个命名参数控制。此选项允许设置任何参数的值。使用“-Ohelp”获取已知参数及其  
-H1            # 通过Intel配置机制1直接访问硬件  
-H2            # 通过Intel配置机制2直接访问硬件  
-F <file>       # 与其访问真正的硬件，不如从先前运行的lspci-x生成的给定文件中读取设备及其配置寄存器的值  
-G             # 提高库的调试级别
```

说明

1.关于“-m”选项

如果您打算自动处理lspci的输出，请使用本节中描述的机器可读的输出格式之一(-m、-vm、-vmm)。所有其他格式都可能在lspci的不同版本之间发生变化。所有的数字都是以十六进制打印的。如果要处理数字ID而不是名称，请添加-n开关。

在简单格式中，每个设备都在一行上进行描述，这些参数被格式化为适合传递给shell脚本的参数，即由空格分隔的值，必要时引用和转义。其中一些参数是位置：槽、类、供应商名称、设备名称、子系统名称和子系统名称(如果设备没有子系统，最后两个参数是空的)；其余的参数是选项

2.关于“-vmm”选项

详细的输出是由空行分隔的记录序列，每条记录用一行来描述一个设备，每一行包含一个‘tag: value’对。标记和值由单个制表符分隔。记录或记录中的行都不按任何特定顺序排列。标记区分大小写。下面是已经定义的tag：

- Slot, 设备所在的插槽名称
- Class, 类名
- Vendor, 厂商名
- Device, 设备名
- SVendor, 子系统供应商名字
- SDevice, 子设备名字
- PhySlot, 设备所在的物理插槽
- Rev, 修序号
- Proglf, 编程接口
- Driver, 当前正在处理设备的内核驱动程序
- Module, 内核模块的报告

举例

以机器可读的方式显示

```
[root@localhost ntop-4.0.1]$ lspci -m
00:00.0 "Host bridge" "Intel Corporation" "440FX - 82441FX PMC [Natoma]" -r02 "" ""
00:01.0 "ISA bridge" "Intel Corporation" "82371SB PIIX3 ISA [Natoma/Triton II]" "" ""
00:01.1 "IDE interface" "Intel Corporation" "82371AB/EB/MB PIIX4 IDE" -r01 -p8a "" ""
00:02.0 "VGA compatible controller" "InnoTek Systemberatung GmbH" "VirtualBox Graphics Adapter"
```

显示设备代码和名字

```
[root@localhost ntop-4.0.1]$ lspci -nn
#设备代码0600, 厂商代码8086: 1237
00:00.0 Host bridge [0600]: Intel Corporation 440FX - 82441FX PMC [Natoma] [8086:1237] (rev 02)
00:01.0 ISA bridge [0601]: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II] [8086:7000]
00:01.1 IDE interface [0101]: Intel Corporation 82371AB/EB/MB PIIX4 IDE [8086:7111] (rev 01)
```

以树形结构显示

```
[root@localhost ntop-4.0.1]$ lspci -t
#总线编号, 插槽, 功能编号
-[0000:00]-+-00.0
    +-01.0
    +-01.1
    +-02.0
```

显示指定位置的设备信息

```
[root@localhost ntop-4.0.1]$ lspci -s 0000:01.0  
00:01.0 ISA bridge: Intel Corporation 82371SB PIIIX3 ISA [Natoma/Triton II]
```

sh

显示指定厂商和设备号的设备信息

```
[root@localhost ntop-4.0.1]$ lspci -d 8086:1237  
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
```

sh

lsusb - 显示本机的USB设备列表信息

lsusb命令 用于显示本机的USB设备列表，以及USB设备的详细信息。

lsusb命令是一个学习USB驱动开发，认识USB设备的助手，推荐大家使用，如果您的开发板中或者产品中没有lsusb命令可以自己移植一个，放到文件系统里面。

显示本机的usb设备列表，可以显示出usb的详细信息，包括设备的读取速度和描述符。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
lsusb [OPTION]
```

sh

选项

-v	# 告诉lsusb详细显示所示设备的详细信息。这包括设备当前速度的配置描述符。如果可用，
-s [[bus]:][devnum]	# 显示指定总线和设备号的设备信息，总线和设备号用十进制标识。格式：lsusb -s 00:01
-d [vendor]:[product]	# 显示指定厂商和产品编号的设备，用十六进制表示编号。格式：lsusb -d 8086:
-D	# 显示指定设备文件的设备信息，例如：lsusb -D /proc/bus/usb/001/001。只有root才能执行
-t	# 以树状结构显示
-V	# 指令版本信息

如果指定的设备没有被找到，那么返回一个非0值。/usr/share/hwdata/usb.ids文件中记录了所有的USB设备节点的信息，包括制造商、产品号、类、子类、协议等等。

举例

直接显示简单的设备信息

```
[root@localhost ntop-4.0.1]$ lsusb
//总线号      设备号          厂商ID
Bus 001       Device 001:   ID 1d6b:0001 Linux Foundation 1.1 root hub
```

sh

显示详细信息

sh

```
[root@localhost ntop-4.0.1]$ lsusb -v
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Device Descriptor:      # 设备描述符
    bLength          18
    bDescriptorType   1
    ...
Hub Descriptor:        # 集线器描述符
    bLength          11
    ...
Hub Port Status:       # 集线器端口状态
    Port 1: 0000.0100 power
    Port 2: 0000.0100 power
    ...
Device Status:         0x0003  # 设备状态
    Self Powered
    Remote Wakeup Enabled
```

显示指定总线上的设备

sh

```
[root@localhost ntop-4.0.1]$ lsusb -s 001:001
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

显示指定厂商的设备信息

sh

```
[root@localhost ntop-4.0.1]$ lsusb -d 1d6b:001
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

以树状结构显示

sh

```
[root@localhost ntop-4.0.1]$ lsusb -t
Bus# 1
`-Dev# 1 Vendor 0x1d6b Product 0x0001
```

lynx - 纯文本模式的网页浏览器

lynx命令 是纯文本模式的网页浏览器，不支持图形、音视频等多媒体信息。

lynx是一个字符界面的全功能www浏览器，它没有图形界面，因此占用的资源较少。

适用范围

RedHat

RHEL

Ubuntu

Fedora

语法

```
lynx [OPTION] [参数]
```

sh

参数:

URL: 指定要访问的网站的URL地址。

选项

```
-case      # 在搜索字符串时，区分大小写;  
-ftp       # 关闭ftp功能;  
-nobrowse  # 关闭目录浏览功能;  
-noclor    # 关闭色彩显示模式;  
-reload    # 更新代理服务器的缓存，只对首页有效;  
--color    # 如果系统支持彩色模式，则激活彩色模式;  
--help     # 显示指令的帮助信息;  
--versionm # 显示指令的版本信息。
```

sh

内部命令

移动命令

```
下方向键: 页面上的下一个链接(用高亮度显示)。  
上方向键: 页面上的前一个链接(用高亮度显示)。  
回车和右方向键: 跳转到链接指向的地址。  
左方向键: 回到上一个页面。
```

滚动命令

+、Page-Down、Space、Ctrl+f: 向下翻页。
-、Page-Up、b、Ctrl+b: 向上翻页。
Ctrl+a: 移动到当前页的最前面。
Ctrl+e: 移动到当前页的最后面。
Ctrl+n: 向下翻两行。
Ctrl+p: 往回翻两行。
): 向下翻半页。
(: 往回翻半页。
#: 回到当前页的 Toolbar 或 Banner。

###文件操作命令

c: 建立一个新文件。
d: 下载选中的文件。
E: 编辑选中的文件。
f: 为当前文件显示一个选项菜单。
m: 修改选中文件的名字或位置。
r: 删除选中的文件。
t: Tag highlighted file.
u: 上载一个文件到当前目录。

其他命令

?、h: 帮助。
a: 把当前链接加入到一个书签文件里。
c: 向页面的拥有者发送意见或建议。
d: 下载当前链接。
e: 编辑当前文件。
g: 跳转到一个用户 指定的URL或文件。
G: 编辑当前页的URL，并跳转到这个URL。
i: 显示文档索引。
j: 执行预先定义的“短”命令。
k: 显示键盘命令列表。
l: 列出当前页上所有链接的地址。
m: 回到首页。
o: 设置选项。
p: 把当前页输出到文件, e-mail, 打印机或其他地方。
q: 退出。
/: 在当前页内查找字符串。
s: 在外部搜索输入的字符串。
n: 搜索下一个。
v: 查看一个书签文件。
V: 跳转到访问过的地址。
x: 不使用缓存。
z: 停止当前传输。
[backspace]: 跳转到历史页(同 V 命令)。
=: 显示当前页的信息。
: 查看当前页的源代码。
!: 回到shell提示符下。

举例

以文本方式访问网站

```
[sogrey@bogon 文档]$ lynx www.baidu.com
```

sh

mail - 一个邮件的管理程序

mail是一个邮件的管理程序，可以用来发送或者接收邮件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mail [选项] addr
```

sh

选项

```
-a file      # 将给定的文件发送出去
-b           # 指定邮件盲抄送地址
-c           # 指定抄送地址
-H           # 显示所有的邮件头并且推出
-i           # 忽略控制台的终端信号
-r           # 设置发送者地址
-s           # 指定邮件主题
-u           # 阅读指定用户的邮件列表
-v           # 显示详细信息
-V           # 显示版本信息，并退出
```

sh

举例

阅读用户david邮件

sh

```
[root@localhost ~]$ mail -u david      #指定用户，首先得到邮件列表
Heirloom Mail version 12.4 7/29/08. Type ? for help.
"/var/mail/david": 10 messages 3 new
  1 root          Thu Aug 16 17:07  21/692   "test"
  2 root          Thu Aug 16 17:08  20/631   "test"
  3 root          Thu Aug 16 17:10  20/602   "test"
  4 root          Fri Aug 17 08:15  20/570   "test3"
  5 root          Fri Aug 17 09:46  25/668   "test04"
  6 root          Tue Aug 21 09:14  20/609   "test04"
  7 root          Tue Aug 21 09:15  20/666   "test05"
>N  8 wejie       Fri Oct  5 21:32  13/403
N  9 wejie       Fri Oct  5 21:33  13/400
N 10 wejie       Fri Oct  5 21:37  13/406
```

& 10 #此处输入邮件编号，可以得到具体邮件内容

Message 10:

```
From weijie@david.cn  Fri Oct  5 21:37:11 2018
Return-Path: <weijie@david.cn>
X-Original-To: david
Delivered-To: david@david.cn
Date: Fri, 5 Oct 2018 21:37:07 +0800 (CST)
From: weijie@david.cn (weijie)
To: undisclosed-recipients:;
Status: RO
```

hehe

发送邮件

sh

```
[root@localhost ~]$ mail -r david weijie      #发送邮件给weijie，发信人david
Subject: test
123
.
EOT
You have new mail in /var/spool/mail/root
[root@localhost ~]$ tail /var/spool/mail/weijie    #查看weijie邮箱，收到david邮件
To: weijie@david.cn
Subject: test
User-Agent: Heirloom mailx 12.4 7/29/08
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
Message-Id: <20181005141158.B5D4914321B@mailsrv.david.cn>
```

123

mailq - 显示出待发送的邮件队列

mailq指令可以显示出待发送的邮件队列。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
mailq
```

sh

选项

无

举例

显示邮件队列

```
[root@localhost ~]$ sendmail -f weijie wj78080458@163.com #发送邮件  
1.23  
. You have new mail in /var/spool/mail/root  
[root@localhost ~]$ mailq #显示邮件队列  
-Queue ID- --Size-- ----Arrival Time---- -Sender/Recipient-----  
CFF2E14321A*    284 Fri Oct  5 21:55:21 weijie@david.cn  
                  wj78080458@163.com  
  
-- 0 Kbytes in 1 Request.
```

mget - 登录mftp服务器之后从服务器获取文件

使用lftp登录mftp服务器之后，可以使用mget指令从服务器获取文件。mget指令可以使用通配符，而get指令则不可以。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
mget [-E] [-a] [-c] [-d] [-O base] rfile
```

sh

选项

-d	# 创建与文件名字相同的目录，将文件存放于此
-c	# 如果失败，持续获取
-E	# 获取之后，删除源文件
-a	# 使用ascii模式
-O	# 指定输出文件存放的目录

sh

举例

使用通配符获取文件

```
[root@localhost ~]$ lftp 192.168.1.8      #登录服务器
lftp 192.168.1.8:~> cd pub/                #切换目录
lftp 192.168.1.8:/pub> ls                  #查看内容
-rwxrwxrwx    1 0        0          2375494044 Aug 14 06:38 1.zip
-rw-r--r--    1 0        0          0 Oct 02 01:19 11c
-rw-r--r--    1 0        0          0 Oct 02 01:19 22c
drwxr-xr-x    2 0        0          4096 Oct 02 01:12 testftp
lftp 192.168.1.8:/pub> mget *c            #获取名字包含c的文件
Total 2 files transferred
lftp 192.168.1.8:/pub> quit              #退出
You have new mail in /var/spool/mail/root
[root@localhost ~]$ ls                  #查看内容，已经获取到文件
1  11c  1.zip  2.c.bz2  4.c  6.c~  rec000012.c.bz2
1.  11.c  22c  3.c       5.c  col    res.zip
```

sh

mirror - 登录ftp服务器之后从服务器获取目录

使用lftp登录ftp服务器之后，可以使用mirror指令从服务器获取目录

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
mirror [OPTS] [source [target]]
```

sh

选项

```
-c      # 如果失败，持续获取  
-n      # 只下载新文件  
-r      # 不下载子目录  
-p      # 下载时不设置权限  
-L      # 将符号链接当做文件
```

sh

举例

获取目录

```
[root@localhost ~]$ lftp 192.168.1.8          #登录服务器  
lftp 192.168.1.8:> cd pub/                  #切换目录  
lftp 192.168.1.8:/pub> mirror testftp/        #下载目录  
Total: 1 directory, 2 files, 0 symlinks  
New: 2 files, 0 symlinks  
lftp 192.168.1.8:/pub> quit                  #退出  
You have new mail in /var/spool/mail/root  
[root@localhost ~]$ ls                         #查看内容，已经获取到目录  
1  11c  1.zip  2.c.bz2  4.c  6.c~  rec000012.c.bz2  testftp  
1.  11.c  22c  3.c       5.c  col    res.zip  
[root@localhost ~]$
```

sh

mkdir - 创建目录

如果目录不存在，那么就创建目录。

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。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
mkdir [选项] dir # 多个目录之间用空格隔开
```

sh

选项

```
-m, --mode=MODE      # 设置目录的权限  
-p, --parents        # 创建多层目录的时候，如果父目录不存在，那么首先创建父目录  
-v, --verbose         # 显示执行过程  
  
--help                # 显示帮助文档  
--version             # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon 文档]$ mkdir newDir
[sogrey@bogon 文档]$ ls
backup demos newDir
[sogrey@bogon 文档]$ ls -l
总用量 12
drwx----- 2 sogrey sogrey 4096 1月 12 00:52 backup
drwx----- 2 sogrey sogrey 4096 1月 14 23:23 demos
drwx----- 2 sogrey sogrey 4096 1月 15 23:42 newDir
[sogrey@bogon 文档]$ mkdir -m 777 newDir2
[sogrey@bogon 文档]$ ls -l
总用量 16
drwx----- 2 sogrey sogrey 4096 1月 12 00:52 backup
drwx----- 2 sogrey sogrey 4096 1月 14 23:23 demos
drwx----- 2 sogrey sogrey 4096 1月 15 23:42 newDir
drwxrwxrwx. 2 sogrey sogrey 4096 1月 15 23:43 newDir2
[sogrey@bogon 文档]$ mkdir -v newDir3/test.java
mkdir: 无法创建目录 "newDir3/test.java": 没有那个文件或目录
[sogrey@bogon 文档]$ mkdir -p -v newDir3/test.java
mkdir: 已创建目录 "newDir3"
mkdir: 已创建目录 "newDir3/test.java"
[sogrey@bogon 文档]$
```

mke2fs - 创建磁盘分区上的etc2/etc3文件系统

mke2fs命令 被用于创建磁盘分区上的“etc2/etc3”文件系统。

在磁盘分区上创建ext2、ext3、ext4文件系统，默认情况下会创建ext2。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mke2fs [选项] [设备]
mke2fs [ -c | -l filename ] [ -b block-size ] [ -f fragment-size ] [ -g blocks-per-group ]
mke2fs -O journal_dev [ -b block-size ] [ -L volume-label ] [ -n ] [ -q ] [ -v ] external
```

device是与设备相对应的特殊文件(例如: g/dev/hdXX)。blocks-count是设备上的块数。如果省略, mke2fs会自动配置文件系统的大小。如果调用为mkfs.ext3, 则创建日记, 好像指定了“-j”选项。新创建的文件系统参数的默认值(如果没有被下面列出的选项覆盖)由“/etc/mke2fs.conf”配置文件控制。

选项

```

-b block-size          # 指定文件系统上块的大小 (kb)，有效值1024、2048、4096。如果省略，块大小将由文件系
-c                      # 坏道检测。如果该选项被指定两次，则使用较慢的读写测试，而不是快速只读测试。
-E extended-options    # 为文件系统设置扩展选项。扩展选项是逗号分隔的，可以使用等号复制。在mke2fs的早期版
                         # stride=stride-size。使用stride-size个块来配置RAID数组，这是在移动到下一个磁盘
                         # stripe-width=stripe-width。使用stripe-width个块来配置RAID数组，这通常是stripe-size
                         # resize=max-online-resize。预留足够的空间，以便块组描述符能够增长以支持具有在线扩
                         # lazy_itable_init[= <0 to disable, 1 to enable>]。如果使能了，并且也启用了update
                         # test_fs。在文件系统超级块中设置一个标志，指示可以使用试验性内核代码(如ext4dev文
                         # discard。尝试在mkfs时丢弃块(在固态设备和稀疏/稀疏的Provisioned存储中丢弃块是有
                         # nodiscard。在mkfs时不会丢弃块。
-f fragment-size        # 设置文件系统碎片的大小。mke2fs接受“-f”选项，但目前忽略它，因为第二个扩展文件系统
-F                      # 强制mke2fs创建文件系统，即使指定的设备不是块特殊设备上的分区，或者其他参数没有意
-g blocks-per-group     # 指定块组中的块数。用户通常没有任何理由设置此参数，因为默认设置对文件系统是最佳的。
-G number-of-groups      # 指定组的数量，这些组将被打包在一起用来创建更大的虚拟块组。组数必须是2的幂，并且只
-i bytes-per-inode       # mke2fs为磁盘上每一个inode字节创建一个inode。bytes/inode比越大，创建的inode就
-I inode-size             # 指定每个inode的大小(以字节为单位)。mke2fs默认创建256字节的inode。在2.6.10之后
-j                      # 创建ext3文件系统。如果没有指定“-j”选项，则默认日志参数将用于创建存储在文件系统中的
-J journal-options        # 使用命令行中指定的选项创建ext 3日志。选项是逗号分隔的，可以使用相等号对参数赋值。
-K                      # size=journal-size。指定内部日志的大小，单位是MB。日志的大小必须至少为1024个文件
-l filename               # device=external-journal。将文件系统附加到位于指定“external-journal”的日志块
-L new-volume-label        # 保留，不要试图在mkfs时丢弃块
-m reserved-blocks-percentage # 从文件中读取磁盘坏块信息。注意，必须使用mke2fs使用的块大小来生成坏块列表中的块号。
-M last-mounted-directory   # 设置文件系统最后的挂载目录。一些实用程序可以从上一次挂载目录中选择键，以确定
-n                      # 不创建文件系统，而是演示创建文件系统时该怎么做
-N number-of-inodes         # 重写应为文件系统保留的inode数量的默认值。
-o creator-os              # 重写文件系统的“creator operating system”字段的默认值。默认情况下，creator字段
-O feature[,...]
                         # 创建具有给定功能的文件系统(文件系统选项)，覆盖默认的文件系统选项。默认情况下启用的
                         # 文件系统特性集将使用此选项指定的特性集进行进一步编辑，或者如果未给出此选项，则由正
                         # 文件系统功能集由一系列要启用的以逗号分隔的特性组成。要禁用一个功能，只需在特征名前
                         # 1) dir_index。使用hashed b-trees来加快大目录中的查找速度
                         # 2) extent。使用间extent块来存储inode中数据块的位置。这是一种更有效的编码，它
                         # 3) filetype。将文件类型信息存储在目录条目中。
                         # 4) flex_bg。允许将每个块组元数据(分配位图和inode表)放置在存储介质上的任何位置
                         # 5) has_journal。创建ext 3日志，和“-j”选项一样。
                         # 6) journal_dev。在给定设备上创建外部ext 3日志，而不是常规ext 2文件系统。
                         # 7) large_file。文件系统可以包含大于2GB的文件。(创建文件>2GB时，现代内核会自动
                         # 8) resize_inode。保留空间，这样块组描述符表在将来可能会增长。对于使用resize2fs
                         # 9) sparse_super。创建一个较少SuperBlock备份的文件系统(在大型文件系统上节省
                         # 10) uninit_bg。在不初始化所有块组的情况下创建一个文件系统。这个特性还允许校验
-q                      # 静默执行，通常用在脚本文档中。
-r revision               # 为新的文件系统设置文件系统修订号。
-S                      # 只写超级块和组描述符。如果所有的超级块和备份超级块都损坏了，并且需要一种最后的恢复
-t fs-type                 # 指定文件系统类型，默认ext2。此选项根据“/etc/mke2fs.conf(5)”中的fstypes配置节
                         # 如果使用“-O”选项显式地添加或删除应该在新创建的文件系统中设置的文件系统选项，则所
-T usage-type[,...]
                         # 指定文件系统的使用方式，以便mke2fs可以选择最佳的文件系统参数。所支持的usage-type
-U                      # 用指定UUID创建系统
-v                      # 显示详细执行过程
-V                      # 显示命令版本信息

```

举例

创建ext2文件系统

```
[root@localhost ~]$ mknod /dev/sdb4 b 1 1          //创建一个设备

[root@localhost ~]$ mke2fs /dev/sdb4              //创建文件系统, 没有指定类型, 默认是ext2
mke2fs 1.41.12 (17-May-2010)
文件系统标签=
操作系统:Linux
块大小=1024 (log=0)
分块大小=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
4096 inodes, 16384 blocks
819 blocks (5.00%) reserved for the super user
第一个数据块=1
Maximum filesystem blocks=16777216
2 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
     8193

正在写入inode表: 完成
Writing superblocks and filesystem accounting information: 完成
This filesystem will be automatically checked every 26 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.

[root@localhost ~]$ mount /dev/sdb4 /media/disk    #将文件系统挂载
[root@localhost ~]$ df -T      #查看已经使用的文件系统
Filesystem           Type      1K-blocks   Used   Available Use% Mounted on
/dev/mapper/VolGroup-lv_root ext4      25552764 13250844 11003900 55% /
tmpfs                 tmpfs       829656    268    829388  1% /dev/shm
/dev/sda1               ext4      495844    32996   437248  8% /boot
/dev/sr0                iso9660  56618      56618      0 100% /media/VBox_GAs_5.2.18
/dev/sdb1               vfat      15863     140    14904  1% /media/disk
/dev/sdb4               ext2      15863     140    14904  1% /media/disk      #可以
```

mkfs - 用于在设备上创建Linux文件系统

mkfs命令 用于在设备上（通常为硬盘）创建Linux文件系统。mkfs本身并不执行建立文件系统的工作，而是去调用相关的程序来执行。

在磁盘分区上创建ext2、ext3、ext4、ms-dos、vfat文件系统，默认情况下会创建ext2。mkfs用于在设备上构建Linux文件系统，通常是硬盘分区。文件要么是设备名称(例如/dev/hda1, /dev/sdb2)，要么是包含文件系统的常规文件。成功返回0，失败返回1。

实际上，mkfs只是Linux下可用的各种文件系统构建器(mkfs.fstype)的前端，在可能/sbin、/sbin/fs、/sbin/fs.d、/etc/fs、/etc/fs等多个目录中搜索特定于文件系统的生成器(编译时定义了精确的列表，但至少包含/sbin和/sbin/fs)，最后在PATH环境变量中列出的目录中搜索。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mkfs [-V] [-t fstype] [fs-options] filesystem [blocks]
```

sh

选项

```
-t          # 指定文件系统类型， 默认ext2  
fs-options # 传递给真正的文件系统构建器的特定选项。虽然没有保证，但大多数文件系统构建器都支持下列  
-l filename # 从指定文件中读取坏块列表  
-c          # 创建文件系统之前进行坏道检测  
-v          # 显示详细执行过程  
  
-V          # 显示命令版本信息
```

sh

举例

在/dev/hda5上建一个msdos的档案系统，同时检查是否有坏轨存在，并且将过程详细列出来：

```
mkfs -V -t msdos -c /dev/hda5  
mkfs -t ext3 /dev/sda6      #将sda6分区格式化为ext3格式  
mkfs -t ext2 /dev/sda7      #将sda7分区格式化为ext2格式
```

sh

mkisofs - 建立ISO 9660映像文件

mkisofs命令 用来将指定的目录与文件做成ISO 9660格式的映像文件，以供刻录光盘。

mkisofs指令可以创建ISO9660/Joliet/HFS文件系统，现在使用指令genisoimage代替它。genisoimage是一个预掌握程序，用于生成iso 9660/joliet/hfs混合文件系统。

genisoimage能够生成由Rock Ridge交换协议指定的系统使用共享协议记录(SUSP)，用于向unix主机进一步描述iso 9660文件系统中的文件，并提供诸如长文件名、UID/GID、POSIX权限、符号链接以及块和字符设备文件等信息。如果指定Joliet或HFS混合命令行选项，genisoimage将创建Joliet或HFS所需的附加文件系统元数据，否则genisoimage将生成纯ISO 9660文件系统。

genisoimage可以生成真正的(或共享的)HFS混合文件系统。这些文件从Macintosh访问时被视为HFS文件，从其他机器访问时被视为ISO 9660文件。HFS代表分层文件系统，是在Macintosh计算机上使用的本机文件系统作为替代，genisoimage可以为每个文件生成对ISO 9660的Apple扩展。当从Macintosh访问时，这些扩展为每个文件提供创建者、类型和某些Finder标志。请参阅下面的HFS Macintosh文件格式部分。

genisoimage获取给定目录树的快照，并生成二进制映像，当写入块设备时，该图像将对应于iso 9660和/或hfs文件系统。

写入iso 9660文件系统的每个文件都必须有一个8.3格式的文件名(最多8个字符.最多3个字符，全部大写)，即使Rock Ridge正在使用。这个文件名用于无法使用Rock Ridge扩展名的系统(如MS-DOS)，每个目录中的每个文件名必须与每个目录中的文件名不同。genisoimage通常试图通过强制unix文件名大写并根据需要截断而形成正确的名称，但当截断的名称并不都是唯一时，这通常会产生不满意的结果。genisoimage为每个文件名分配权重，如果找到两个其他相同的名称，则将优先级较低的名称重命名为包含一个3位数的数字(保证是唯一的)。例如，两个文件“foo.bar”和“foo.bar.~1~”可以呈现为“FOO.BAR；1”和“FOO000.bAR；1”

当与各种HFS选项一起使用时，genisoimage将尝试识别存储在多个Apple/Unix文件格式中的文件，并复制数据和资源派生以及任何相关的Finder信息。有关genisoimage支持的更多信息，请参见下面的HFS Macintosh文件格式部分。

请注意，genisoimage的设计并不是为了直接与作者通信。大多数编写人员都有专有的命令集，每个制造商的不同，您需要一个专门的工具才能真正地刻录盘。WODIM就是这样的工具之一。

适用范围

RedHat openSUSE	RHEL Fedora	Ubuntu Linux Mint	CentOS Alpine Linux	Debian	Deepin	SUSE Arch Linux
--------------------	----------------	----------------------	------------------------	--------	--------	--------------------

语法

```
mkisofs [选项] [-o file.iso] srcfile
genisoimage [options] [-o filename] pathspec [pathspec ...]
```

sh

pathspec是要复制到iso 9660文件系统中的目录树的路径。可以指定多个路径，genisoimage将所有指定路径组件中的文件合并成文件系统映像。

如果指定了“-graft-points”选项，则可以在root目录以外的点嫁接路径，也可以将文件或目录移植到CDRom映像上，并且源文件系统中的名称可以不同。我们首先假设存在一个本地文件“./old.lis”，并且希望将它包含在CDROM映像中。语句

```
foo/bar/=/..old.lis
```

可以使“/foo/bar/old.lis”的CDROM映像中包括“old.lis”。而语句

```
foo/bar/xxx=../old.lis
```

可以使“/foo/bar/xxx”的CDROM映像中包括“old.lis”。同样的语法也可以与目录一起使用。genisoImage将创建任何目录，以满足CDROM映像上存在嫁接点的需求。默认情况下，像这样动态创建的任何目录都具有权限0555，被genisoimage的运行者拥有。如果您希望其他权限或中间目录的所有者，请参阅-uid, -gid, -dir-mode, -file-mode and -new-dir-mode。

选项

```
-abstract file          # 设置摘要文件名称，可以有37个字符
-A AppId                # 指定描述光盘应用程序Id的文本字符串，可以有128个字符
-allow-limited-size      # 当处理不能在ISO 9660中容易表示的大于2GiB的文件时，用缩小的可见文件大小将它们
-allow-leading-dots-ldots # 允许ISO 9660文件名以句点开始。通常，为了保持MS-DOS的兼容性，使用下划线替换前
-allow-lowercase         # 允许iso9660文件名中出现小写字母。这违反了ISO 9660标准，但它恰好适用于某些系
-allow-multidot          # 允许iso9660文件名中出现多个点。引导点不受此选项的影响，可以单独使用“-allow-
-biblio file            # 指定书目文件名。有37个字符的空间。等效于“.genisoImagerc”文件中的BIBL。
-cache-inodes           #
-no-cache-inodes        # 启用或禁用inode缓存和以设备号查找指向文件的硬链接。如果genisoImage找到一个硬
-alpha-boot alpha _boot_image # 指定在制作Alpha/SRM可引导CD时使用的引导映像的路径和文件名。路径名必须相
-hppa-bootloader hppa_bootloader_image # 指定在制作HPPA可引导CD时使用的引导映像的路径和文件名。路径名
-hppa-cmdline hppa_boot_command_line # 指定在制作可引导CD时要传递给HPPA引导加载程序的命令行。用空格或
-hppa-kernel-32 hppa_kernel_32 #
-hppa-kernel-64 hppa_kernel_64 # 指定在制作HPPA可引导CD时使用的32位和/或64位内核映像的路径和文件名。路
-hppa-ramdisk hppa_ramdisk_image # 指定在制作HPPA可引导CD时使用的ramdisk映像的路径和文件名。路径名必须
-mips-boot mips_boot_image # 指定在生成SGI/大端MIPS可引导CD时使用的引导映像的路径和文件名。路径名必须相
-mipsel-boot mipsel_boot_image # 指定制作DEC/小端MIPS引导映像的路径和文件名。路径名必须相对于指定给gen
-b eltorito_boot_image # 指定在为x86 PC制作El Torito可引导映像的路径和文件名。路径名必须
# - 如果引导映像不是软盘映像，则需要添加“-hard-disk-boot”或“-no-emul-boot.”。
# - 如果未指定“-sort”，则引导映像将以低优先级(2)排序到介质的开头。如果您不喜欢这
-B img_sun4,img_sun4c,img_sun4m,img_sun4d,img_sun4e #
-sparc-boot img_sun4,img_sun4c,img_sun4m,img_sun4d,img_sun4e # 设置引导文件名，img_sun4,img_sun4c,
# - 指定为SPARC系统制作可引导cd所需的逗号分隔的引导映像列表。分区0用于iso 96
# - 实现的引导方法是在SunOS4.x和SunOS5.x中找到的。但是，它不依赖于SunOS内部
# - 如果指定了文件名，则将实际的和下面的所有引导分区映射到前一个分区。如果使用
-G generic_boot_image # 指定在制作通用可引导cd时使用的引导映像的路径和文件名。引导映像将放置在cd的前
-eltorito-alt-boot     # 从一组新的El Torito启动参数开始。最多63个El Torito引导项可以存储在一张CD_
-hard-disk-boot        # 指定用于创建El Torito可引导CD的引导映像是硬盘映像。映像必须以包含单个分区的
-no-emul-boot          # 指定用于创建El Torito可引导cd的引导映像是“无仿真”映像。系统将在不执行任何磁
-no-boot               # 指定应将创建的El Torito CD标记为不可引导。系统将提供一个模拟驱动器的图像，
-boot-load-seg segment_address # 指定非模拟El Torito cd的引导映像的加载段地址
-boot-load-size load_sectors # 指定在非模拟模式下加载的“虚拟”(512字节)扇区的数量。默认情况是加载整个
-boot-info-table        # 指定在引导文件中的偏移量8处修补一个56字节的表，其中包含CD-ROM布局的信
-C last_sess_start,next_sess_start # 此选项需要为多会话磁盘创建一个额外的CD或第二个会话或更高级别会话
-c boot_catalog         # 指定引导目录的路径和文件名，这是El Torito可引导CD所必需的。路径名必须相对于根
# - 如果未指定“-sort”，则引导目录以低优先级(1)排序到介质的开头。如果您不喜欢这
-check-oldnames         # 检查从旧会话导入的所有文件名是否符合iso 9660文件命名规则。如果没有此选项，只
-check-session file      # 检查所有旧会话是否符合实际的genisoImage iso 9660文件命名规则。这是一个高级
-copyright file          # 指定版权信息，通常是光盘上的文件名。有37个字符的空间。相当于在“.genisoImage
```

```
-d          # 不要将句点附加到没有句点的文件中。这违反了ISO 9660标准，但它恰好适用于许多系
-D          # 不要使用深度目录重新定位，而是按我们看到的方式打包它们。如果未选择iso 9660：
-dir-mode mode      # 重写用于创建镜像的目录mode，指定为chmod(1)中权限位的4位数字。
-dvd-video      # 生成一个符合DVD视频的UDF文件系统，这是通过排序适当文件的内容顺序和在需要时在文
# 注意，为了获得符合dvd视频的文件系统映像，您需要准备一个符合dvd视频的目录树，此树必须包含
# 在生成文件系统时遵循符号链接。当这个选项没有使用时，符号链接将使用Rock Ridge
# 重写用于创建image的常规文件的模式，使用4位模式
-gid gid      # 将从源文件读取的组ID重写为gid的值。指定此选项将自动启用RockRidge扩展
-gui          # 切换GUI的行为。这使得输出更加冗长，但将来可能会产生其他影响。
-graft-points      # 允许文件名使用嫁接点。如果使用此选项，将检查所有文件名是否有嫁接点。文件名在第
# 隐藏出现在ISO 9660或Rock Ridge目录中的与shell通配符模式GLOB匹配的任何文件
# 包含要隐藏的shell通配符列表的文件
-hide glob      # 为匹配GLOB的文件和目录添加隐藏(存在)ISO 9660目录属性，这是一个shell通配符模
# 包含获取隐藏属性的shell通配符列表的文件
-hide-joliet glob      # 隐藏在Joliet目录中看到的与shell通配符模式GLOB匹配的文件和目录。GLOB可能匹配
# 包含要向Joliet树隐藏的shell通配符列表的文件。
-hide-joliet-list file      # 将TRANS.TBL文件隐藏在Joliet树中。这些文件在Joliet世界中通常没有意义，因为它
# 将RockRidge树中目录“RR_MOVED”的“.rr_move”目。这样似乎不可能完全将“RR_MOV
# 定义本地文件名中使用的输入字符集。要获得有效字符集名称的列表，请调用“genisoI
# 定义Rock Ridge文件名中将使用的输出字符集，默认为输入字符集。有关详细信息，i
# 设置ISO 9660一致性级别。有效数字是1到4。
# - 对于第1级，文件只能由一个部分组成，文件名限制为8.3类型。
# - 对于第2级，文件只能由一个部分组成。
# - 对于第三级，没有任何限制 (ISO-9660: 1988除外)。
# 所有ISO 9660级别从1到3，所有文件名仅限于大写字母、数字和下划线(_)。文件名
# - 级别4正式不存在，但genisoImage将其映射到ISO-9660: 1999，这是ISO 9660版
# 使用第4级时，增强型卷描述符的版本号和文件结构的版本号设置为2。目录嵌套不限
# 创建第2版image时，genisoImage会发出一个增强的卷描述符，与主卷描述符相似但不
# 除了正常的ISO 9660文件名之外，还生成Joliet目录记录。在Windows机器上使用光盘
# 允许Joliet文件名最多为103个Unicode字符，而不是64个字符。这违反了Joliet规范
# “-J -input-charset charset”的组合。
# 允许完整的31个字符的文件名。通常iso 9660文件名将采用与MS-DOS兼容的8.3格式，
# 过时的选项，使用“-allow-leading-dots”代替。
-jigdo-jigdo jigdo_file      # 生成一个jigdo.jigdo元数据文件以及文件系统映像。
-jigdo-template template_file      # 生成一个jigdo.template元数据文件以及文件系统映像。
-jigdo-min-file-size size      # 指定要在.jigdo文件中列出的文件的最小size。默认值(和最小允许值)为1KB。
-jigdo-force-md5 path      # 指定文件模式，其中文件必须包含在由“-md5-list”提供的externally-supplied
-jigdo-exclude path      # 指定一个不再“.jigdo”中列出的文件模式
-jigdo-map path      # 为jigdo文件指定模式映射，例如“Debian=/mirror/debian”
-md5-list md5_file      # 指定一个文件，这个文件列出了包含在“.jigdo”文件列表中的文件的MD5sum、大小和
-jigdo-template-compress algorithm      # 指定模板date.gzip和bzip 2当前支持的压缩算法，默认情况是gzip。
-log-file log_file      # 将所有错误、警告和信息性消息重定向到log_file，而不是标准错误
-m glob      # 将与shell通配符模式glob匹配的文件排除到CD-ROM。blob可能与文件名组件或完整
-exclude-list file      # 包含要排除的shell通配符列表的文件
-max-iso9660-filenames      # 允许ISO 9660文件名长达37个字符。此选项启用-N，因为额外的名称空间是从为文件
-M path      # 指定要合并的现有ISO 9660映像的路径。另一种形式采用SCSI设备说明符，它使用
-N          # 从ISO 9660文件名中省略版本号。这违反了iso 9660标准，但没有人真正使用版本
-new-dir-mode mode      # 指定在文件系统映像中创建新目录时使用的mode，缺省值为0555。
-nobak      # 排除ISO 9660文件系统上的备份文件；也就是说，包含字符“~”或“#”或以.bak结尾
-force-rr      # 不要对以前的会话进行自动的Rock Ridge属性识别。这可以解决由Nero Burning
-no-rr          # 不要使用以前会话中的RockRidge属性。这可能有助于避免当genisoImage在旧会话
-no-split-symlink-components      # 不要拆分符号链接组件，而是开始一个新的连续区域(CE)。这可能会浪费一些空间，但
# 不要拆分符号链接字段，而是开始一个新的连续区域(CE)。这可能会浪费一些空间，但
-no-split-symlink-fields      # 指定iso 9660文件系统映像的输出文件。这可以是磁盘文件、磁带驱动器，也可以是
-o filename      # 将整个映像的末尾按150个扇区(300 KB)填充。默认启用此选项。如果与-B一起使
-pad          # 不要将结束部分按150个扇区(300 KB)填充，也不要使引导分区在16个扇区的倍数上
-no-pad          # 包含要添加到ISO 9660文件系统中的路径规范化目录和文件名列表的文件。此路径规范
```

```
-P # 过时的操作，参考“-publisher”
-publisher publisher_id # 指定要写入卷头的文本字符串。这应该描述CD-ROM的发行者，通常带有邮件地址和电
-p preparer_id # 指定要写入卷头的文本字符串。这应该描述CD-ROM的发行者，通常带有邮件地址和电
-print-size # 以扇区大小的倍数(2048字节)打印估计的文件系统大小并退出。这个选项是磁盘在一
cdblocks=' genisoimage -print-size -quiet ... '
genisoimage ... | wodim ... tsize=${cdblocks}s -
-quiet # 静默模式，只有少量的输出。
-R # 使用Rock Ridge协议生成SUSP和RR记录，以进一步描述ISO 9660文件系统上的文
-r # 这类似于-R选项，但是文件所有权和模式被设置为更有用的值。uid和gid设置为零。
-relaxed-filenames # 允许ISO 9660文件名包含除小写字母以外的所有7位ASCII字符。这违反了ISO 966
-root dir # 当写入多会话映像时，此选项是必需的，而上一次(甚至更早的)会话是用-root d
-old-root dir # -root和-old-root用于一起进行增量备份。初始会话将使用“genisoimage -root
-sort sort_file # 对媒体上的文件位置进行排序。排序由包含一对文件名和排序偏移加权的文件控制。
-sparc-boot img_sun4,img_sun4c,img_sun4m,img_sun4d,img_sun4e # 参考“-B”
-sparc-label label # 为使用“-sparc-boot”创建的Sun磁盘标签设置Sun磁盘标签名。
-split-output # 将输出image分割成几个文件，每个文件大约1GB。这有助于在没有大型文件支持的
-stream-media-size # 选择流操作并将媒体大小设置为#扇区。这允许您将tar(1)程序的输出输出到genis
-stream-file-name name # 保留
-sunx86-boot UFS_img,,,AUX1_img # 指定为Solaris x86系统制作可引导CD所需的以逗号分隔的文件系统映像列表
# 如果指定了“-sunx86-boot”，则结果映像的第一个扇区将包含一个带有Solaris类型
# Solaris x86引导CD使用1024字节大小的主引导，它使用ElTorito不模拟引导模式
# 为使用-sunx86-boot创建的svr 4磁盘标签设置svr 4磁盘标签名。
-sunx86-label label # 指定系统ID。有32个字符的空间。相当于“.genisoImagerc”文件中的SYSI。
-sysid ID # 在CD-ROM上的每个目录中生成一个TRANS.TBL文件，该文件可用于非Rock Ridge系
-T # 替代翻译表文件名(见上文)。如果要创建多会话映像，则必须使用与上一次会话相同的
-table-name table_name # 在JolietSVD中设置Unicode一致性级别。默认级别为3。可以使用此选项将其设置为
-ucs-level level # 在生成的文件系统映像中包含UDF文件系统支持。UDF支持目前处于alpha状态，因此
-udf # 将从源文件读取的uid重写。指定此选项将自动启用RockRidge扩展
-uid uid # -use-fileversion选项允许genisoImage使用文件系统中的文件版本号。如果未指
-use-fileversion # 允许“未翻译”文件名，完全违反了上面描述的ISO 9660标准。允许以下标志：-d -l
-no-iso-translate # 不要翻译ISO 9660文件名中无效的字符“#”和“~”。虽然这些字符无效，但Microsof
-V volid # 指定要写入主块的卷ID(卷名或标签)。有32个字符的空间。相当于“.genisoImagerc
-volset ID # 指定卷集ID。有128个字符的空间。相当于“.genisoImagerc”文件中的VOLS
-volset-size # # 将卷集大小设置为#。卷集大小是CD卷集中的CD数。卷集是一个或多个卷的集合，其中
-volset-seqno # # GenisoImage目前不支持大于1的-volset-size。选项-volset-size必须在每个命
-v # 将卷集序列号设置为#。卷集序列号是cd集中当前cd的索引号。选项-volset-size必
-x glob # 允长的执行。如果在命令行上给出两次，则会打印额外的调试信息。
-z # 与-m glob相同
# 为透明压缩的文件生成特殊的RRIP记录。这只是支持透明解压缩的主机(如Linux 2.
```

HFS选项

```

-hfs          # 创建ISO 9660/HFS混合CD。此选项应与-map、-magic或下列各种双破折号选项结合使用。
-apple        # 创建带有Apple扩展的ISO 9660 CD。类似于-hfs，只是添加了对ISO 9660的苹果映射。
-map mapping_file # 使用映射文件根据文件名的扩展名为文件设置创建者并键入文件的信息。只有当文件名是通过使用文件的magic数字(通常是文件的前几个字节)来设置的。
-magic magic_file # 创建者和类型信息是通过使用文件的magic数字(通常是文件的前几个字节)来设置的。
-hfs-creator creator # 设置所有文件的默认创建者。一定是4个字符。请参阅HFSCREATOR/TYPE。
-hfs-type type   # 设置所有文件的默认类型。必须精确为4个字符。请参阅HFSCREATOR/TYPE。
-probe        # 搜索所有已知的Apple/Unix文件格式的文件内容。然而，检测MacBinary and Apple/Macintosh文件。
-no-desktop    # 不要创建(空)桌面文件。当在Macintosh上使用CD时，将创建新的hfs桌面文件。默认情况下，将使用HFS文件名作为ISO 9660、Joliet和Rock Ridge文件名的起点。有关更多信息，请参阅Macintosh桌面文件。
-mac-name      # 使用HFS文件名作为ISO 9660、Joliet和Rock Ridge文件名的起点。有关更多信息，请参阅Macintosh桌面文件。
-boot-hfs-file driver_file # 安装驱动程序文件，使CD可在Macintosh上启动。
-part          # 生成一个HFS分区表。默认情况下，不生成分区表，但是一些较老的macintosh CD-ROM使用分区表。
-auto AutoStart_file # 让HFS CD使用QuickTime2.0自动启动功能来启动应用程序或文档。给定的文件名必须是正确的文件名，而不是文件扩展名。
-cluster-size size   # 以字节为单位设置PC Exchange文件的群集或分配单元的大小。
-hide-hfs glob     # 将shell通配符模式GLOB隐藏在HFS卷中。该文件或目录仍然存在于iso 9660和/或Joliet/Rock Ridge卷中。
genisoimage -o rom -hfs -hide-hfs '*.*' -hide-hfs foobar # 将会从HFS卷中排除所有以“.o”结尾或称为foobar的文件。
# 仅从src目录中排除名为html的文件或目录。树中任何其他名为html的文件或目录都将是可见的。
-hide-hfs-list file # 指定一个文件，该文件包含要隐藏的通配符模式的列表，如“-hide-hfs”中所示。
-hfs-volid hfs_volid # HFS分区的卷名称。这是分配给macintosh上的磁盘的名称，并用-V替换使用的volume参数。
-icon-position    # 如果存在图标位置信息，就使用。图标将显示在与Macintosh桌面相同的位置。文件名是可选的。
-root-info file    # 为HFS卷的根文件夹设置位置、屏幕上的大小、滚动位置、文件夹视图等。
-prep-boot file    # 准备引导映像文件。最多允许4次。
-chrp-boot        # 添加CHRP引导头。
-input-hfs-charset charset # 输入字符集，定义与“-mac-name”一起使用时在hfs文件名中使用的字符。默认字符集是ISO 8859-1。
-output-hfs-charset charset # 定义将在HFS文件名中使用的字符的输出字符集。默认输入字符集是ISO 8859-1。
-hfs-unlock       # 默认情况下，genisoImage将创建一个被锁定的hfs卷。此选项使卷不被锁定，以便于在Windows上使用。
-hfs-bless folder_name # “Bless”指定目录(文件夹)。这通常是系统文件夹，用于创建HFS可引导CD。目录的名称是可选的。
-hfs-parms parameters # 重写用于创建HFS文件系统的某些参数。不太可能在正常情况下使用。
--cap            # 查找AUFS CAP Macintosh文件。仅搜索CAP Apple/Unix文件格式。搜索其他可能的文件格式。
--netatalk       # 查找NETATALK Macintosh文件。
--double          # 查找AppleDouble Macintosh文件。
--ethershare      # 查找Helios EtherShare Macintosh文件。
--ushare          # 查找IPT UShare Macintosh文件。
--exchange         # 查找PC Exchange Macintosh文件。
--sgi             # 查找SGI Macintosh文件。
--xinet           # 查找XINET Macintosh文件。
--macbin          # 查找MacBinary Macintosh文件。
--single          # 查找AppleSingle Macintosh文件。
--dave            # 查找Thursby Software Systems DAVE Macintosh文件。
--sfm             # 查找Microsoft's Services for Macintosh文件。
--osx-double      # 查找Mac OS X AppleDouble Macintosh文件。
--osx-hfs         # 查找Mac OS X HFS Macintosh文件。

```

字符集

genisoimage以符合POSIX的方式将文件名处理为8位字符的字符串。要表示所有语言的所有编码，8位字符是不够的。Unicode或ISO-10646定义了至少需要21位才能表示所有已知语言的字符编码。它们可以用UTF-32、UTF-16或UTF-8编码表示。UTF-32使用普通的32位编码，但似乎不常见。Microsoft使用UTF-16时使用Win32，缺点是16位字符不符合POSIX文件系统接口。

现代Unix操作系统可以对文件名使用UTF-8编码。每个32位字符由一个或多个8位字符表示。如果一个字符用ISO-8859-1(在中欧和北美使用)编码，则以1:1映射到UTF-32或UTF-16编码的Unicode字符。如果一个字符是用7位ASCII编码(在美国和其他字符集有限的国家)将以1:1映射到UTF-32、UTF-16或UTF-8编码的Unicode字符。不能用UTF-8(如果值大于0x7F)中的单个字节表示的字符代码使用转义序列，这些转义序列映射到多个8位字符。

如果所有操作系统都使用utf-8, genisoimage将不需要在文件名中重新编码字符。不幸的是, Apple使用完全非标准的编码, 而microsoft使用与POSIX文件名接口不兼容的unicode编码。对于所有非utf-8编码的操作系统, 每个字节表示的实际字符取决于本地操作系统使用的字符集或代码页(microsoft使用的名称)。字符集中的字符将反映用户的区域或自然语言集。通常字符代码0x00-0x1f是控制字符, 代码0x20-0x7f是7位ASCII字符, (在PC和Mac上)0x80-0xff用于其他字符。

由于有超过256个字符/符号在使用, 只有一个小子集表示在一个字符集。因此, 相同的字符代码可以表示不同字符集中的不同字符。因此在中欧生成的文件名, 在东欧的机器上查看时可能不会显示相同的字符。为了使事情变得更复杂, 不同的操作系统对区域或语言使用不同的字符集。例如, 'é'的字符代码(带有尖锐口音的小e)可以是PC上的字符代码0x82、Macintosh上的代码0x8e、西欧Unix系统上的代码0xe9和Unicode中的代码0x000e9。

只要不是所有操作系统和应用程序都使用相同的字符集作为文件名的基础, 就可能需要指定文件名中使用的字符集以及文件名应该显示在CD上的字符集。有四个选项可以指定要使用的字符集:

- **-input-charset**。定义您在主机上使用的本地字符集。发生的任何字符集转换都将使用此字符集作为起点。默认输入字符集在基于MS-DOS的系统上为cp 437, 在所有其他系统上为iso 8859-1。如果给定-j, 则输入字符集的unicode等效项将在joliet目录中使用。“-jcharset”选项 和“-input-charset -J”选项时一样的。
- **-output-charset**。定义用于CD上Rock Ridge名称的字符集。默认为输入字符集。
- **-input-hfs-charset**。定义从各种Apple/Unix文件格式解码的用于HFS文件名的HFS字符集。只有当与“-mac-name”一起使用时才有用。
- **-output-hfs-charset**。定义用于从正在使用的输入字符集创建hfs文件名的hfs字符集。在大多数情况下, 这将来自“-input-charset”选项指定的字符集。默认为输入hfs字符集。

在genisoimage中内置了许多字符集。要获得列表, 请使用输入字符集帮助。此列表不包括从当前区域设置派生的字符集, 如果genisoimage是使用iconv支持构建的。通过将文件名作为选项的参数, 可以从任意字符集选项的文件中读取其他字符集。只有当文件的名称与内置字符集中的一个不匹配时, 才会读取给定的文件。

字符集文件的格式与“<http://www.unicode.org/Public/MAPPINGS>”提供的映射文件相同。格式如下:

```
Column #1 is the input byte code (in hex as 0xXX)
Column #2 is the Unicode (in hex as 0xXXXX)
The rest of the line is ignored.
```

sh

任何空白行、没有上述格式的两个(或更多)列的行或注释行(以#字符开始)都会被忽略, 而不会发出任何警告。任何丢失的输入代码都映射到Unicode字符0x0000。

请注意, 虽然支持UTF-8, 但其他Unicode编码(如UCS-2/UTF-16和UCS-4/UTF-32)则不支持, 因为POSIX操作系统无法本地处理它们。1: 1字符集映射可以通过使用关键字默认值作为任何字符集选项的参数来定义。从输入文件名生成的iso 9660文件名不会从输入字符集转换。iso 9660字符集是ASCII字符的非常有限的子集, 因此任何转换都是没有意义的。任何不能转换的字符都将被“_”字符替换。

HFS CREATOR/TYPE (创建者和类型)

Macintosh文件有两个与其相关的属性, 它们定义了创建文件的创建者和文件类型。两者都是(完全)4个字母字符串。通常, 这允许Macintosh用户双击文件并启动正确的应用程序等。可以通过在Macintosh上使用ResEdit(或类似的)之类的方法找到特定文件的创建者和类型。

创建者和类型信息存储在所有Apple/Unix编码的文件中。对于其他文件，可以使用文件的映射文件(带-map)或使用文件的魔术号(通常是前几个字节中的签名)，将创建者和输入文件的扩展名作为基础。如果这两个选项都给出了，那么它们在命令行上的顺序是有效的。如果“-map”是首先给出的，则在魔术号匹配之前尝试进行文件扩展名匹配。但是，如果先给出“-magic”，则在文件扩展名匹配之前尝试进行魔法号匹配。

如果没有使用映射或魔术文件，或者没有找到匹配，则可以通过使用“.genisolmagerc”文件中的条目或使用“-hfs-creator”或“-hfs-type,”来设置所有常规文件的默认创建者和类型，否则默认的创建者和类型是Unix和Text。

映射文件的格式与aufs使用的afpfile格式相同。该文件有五列用于扩展名、文件转换、创建者、类型和注释。以“#”字符开头的行是注释行，被忽略。下面就是一个例子

```
# Example filename mapping file
#
# EXTN  Xlate   Creator  Type    Comment
.tif    Raw     '8BIM'   'TIFF'  "Photoshop TIFF image"
.hqx    Ascii   'BnHq'   'TEXT'  "BinHex file"
.doc    Raw     'MSWD'   'WDBN'  "Word file"
.mov    Raw     'TVOD'   'MooV'  "QuickTime Movie"
*      Ascii   'ttxt'   'TEXT'  "Text file"
```

sh

第一列EXTN定义要映射的Unix文件名扩展名，任何不匹配的文件扩展名的默认映射都定义为“*”字符。第二列Xlate列定义了Unix和Macintosh文件之间的文本转换类型，它被genisolimage忽略，但与aufs(1)兼容。虽然genisolimage不改变文件的内容，但如果二进制文件的类型设置为文本，则可能会在Macintosh上错误地读取它。因此，默认类型的更好选择可能是“? ? ? ?”。CREATOR和TYPE关键字必须为4个字符长，并以单引号括起来。注释字段以双引号括起来-它被genisolimage忽略，但与aufs兼容。

魔术文件的格式几乎与file (1)命令使用的魔术(5)文件格式相同。该文件有偏移量、类型、测试和消息用四个字节的制表符分隔列。以'#'字符开头的行是注释行，被忽略。一个示例文件将类似于：

```
# Example magic file
#
# off   type    test      message
0      string   GIF8     8BIM GIFf  GIF image
0      beshort  0xffffd8  8BIM JPEG   image data
0      string   SIT!     SIT! SIT!  StuffIt Archive
0      string   \037\235  LZIV ZIVU  standard Unix compress
0      string   \037\213  GNUz ZIVU  gzip compressed data
0      string   %!      ASPS TEXT  Postscript
0      string   \004%!  ASPS TEXT  PC Postscript with a ^D to start
4      string   moov    txtt MooV  QuickTime movie file (moov)
4      string   mdat    txtt MooV  QuickTime movie file (mdat)
```

sh

文件的格式用magic(5)来描述。这里唯一的区别是，对于魔术文件中的每个条目，message的初始偏移量必须是4个字符的CREATOR，紧接着是4个字符的TYPE，空格是可选的。此行上的任何其他字符都将被忽略。延拓行(从'>'开始)也会被忽略，即只使用初始偏移行。

使用“-magic”可能会显著增加处理时间，因为每个文件必须打开和读取，以找到其魔术号码。

总之，对于所有文件，默认创建者是Unix，默认类型是TEXT。这些可以通过使用“.genisolmagerc”文件中的条目或使用“-hfs-creator”或“-hfs-type”选项来更改。如果文件是以已知的Apple/Unix格式之一(并且已

经选择了该格式), 则创建者和类型将从存储在Apple/Unix文件中的值中提取。其他文件的创建者和类型可以通过其文件扩展名(-map)或它们的魔术号(-magic)进行设置。如果在映射文件中使用了默认匹配, 则这些值将覆盖默认的创建者和类型。

HFS MACINTOSH FILE FORMATS (文件格式)

Macintosh文件有两个部分, 称为数据和资源叉。两者都可能是空的。unix(和许多其他开放源码软件)只能处理只有一部分(或分叉)的文件。另外, macintosh文件有许多与它们相关的属性-可能最重要的是类型和创建者。同样, unix对这些类型的属性没有概念。例如, Macintosh文件可以是JPEG图像, 其中图像存储在数据分叉中, 桌面缩略图存储在资源分叉中。数据分叉中的信息通常是跨平台有用的。因此, 要在Unix文件系统上存储Macintosh文件, 必须找到一种方法来处理两个分叉和额外的属性(称为Finder info)。不幸的是, 似乎每个在Unix上存储Macintosh文件的软件包都选择了一种完全不同的存储方法。

genisolvimage(部分)支持的Apple/Unix格式如下:

CAP AUFS format。数据存储在文件中, 资源存储在子目录。资源和数据有相同的文件名

AppleDouble/Netatalk。数据存储在文件中。资源存储在以'%'作为前缀的文件中, 查找器信息也存储在相同的'%'文件中

AppleSingle。数据结构类似于上面的结构, 除了分叉和Finder信息都存储在一个文件中。

Helios EtherShare。数据存储在文件中的, 资源和查找器信息一起存储在子目录.rsrc中, 与数据具有相同的文件名。

IPT UShare。与EtherShare格式类似, 但是Finder信息的存储方式略有不同

MacBinary。分叉和查找器信息都存储在一个文件中。

Apple PC Exchange。Macintoshes将苹果文件存储在DOS(FAT)磁盘上。数据分叉存储在文件中。资源分叉存储在子目录Resoures.frk(或RESOURCE.FRK)中。finder info作为文件finder.dat(或FINDER.DAT)中的一条记录。为每个数据叉目录分离finder.dat。注意: genisolvimage需要知道pc Exchange文件所在的磁盘的本机簇集群大小(或已从其复制)。此大小由"-cluster-size"确定。使用DOS实用程序chkdsk可以找到群集或分配大小。

SGI/XINET。SGI机器在安装HFS磁盘时使用。存储在文件中的数据分叉。.hsResource子目录中的资源叉, 具有相同的文件名.finder info作为文件中的一条记录。

Thursby Software Systems DAVE。允许Macintoshes将苹果文件存储在SMB服务器上。存储在文件中的数据分叉。使用AppleDouble格式存储资源叉。

genisolvimage将尝试从finder信息中设置创建者、类型、日期和可能的其他标志。此外, 如果存在, 则从finder info设置macintosh文件名, 否则macintosh名称基于unix文件名

Services for Macintosh。NT服务器存储在NTFS文件系统上的文件格式。数据分叉存储为文件名。资源叉存储为名为filename: afp_Resource的NTFS流。查找信息存储为名为filename: AFP_AfpInfo的NTFS流。NTFS流通常对用户不可见。注意: genisolvimage只部分支持SFM格式。如果存储在NT服务器上的HFS文件或文件夹在其名称中包含非法NT字符, 则NT将这些字符转换为私有使用Unicode字符。这些字符是: "*/<>? \和空格或句点(如果它是文件名的最后一个字符, 则编码0x01到0x1f)。(控制字符)和苹果的苹果标志。不幸的是, 这些私有Unicode字符无法被genisolvimageNT可执行文件读取。因此, 包含这些字符的任何文件或目录名称都将被忽略-包括任何此类目录的内容。

Mac OS X AppleDouble。当MacOSX将HFS/HFS文件复制或保存到非HFS文件系统(例如UFS、NFS等)时, 文件以AppleDouble格式存储。存储在以"."开头的同名文件中的资源叉。查找器信息也存储在相同的"."文

件中。

Mac OS X HFS (Alpha)。实际上不是Apple/Unix编码，而是MacOSX系统上的实际HFS/HFS文件。存储在文件中的数据分叉。资源叉存储在同名、后缀/rsrc的伪文件中。只有在MacOSX上使用时才能工作。如果发现一个文件具有零长度的资源分叉和空的finderinfo，则假定它没有任何Apple/Unix编码，因此可以使用其他方法设置类型和创建者。

genisolimage将尝试从finder信息中设置创建者、类型、日期和可能的其他标志。此外，如果存在，则从finder info设置macintosh文件名，否则macintosh名称基于unix文件名。

在使用"-apple"时，类型和创建者存储在ISO 9660目录记录中的可选系统使用或SUSP字段中，方式与Rock Ridge属性非常类似。实际上，为了简化生活，在现有Rock Ridge属性的开头添加了Apple扩展(即，为了获得Apple扩展，您也可以获得Rock Ridge扩展)。Apple扩展要求将资源叉存储为与ISO 9660关联的文件。这与存储在ISO 9660文件系统中的任何普通文件一样，只是相关的文件标志是在目录记录(位2)中设置的。该文件与数据分叉(由非苹果计算机看到的文件)具有相同的名称。相关文件通常被其他操作系统忽略。

使用"-hfs"时，类型和创建者以及其他查找器信息存储在单独的HFS目录中，在ISO 9660卷中不可见。HFS目录引用上述相同的数据和资源叉文件。在大多数情况下，最好使用"-hfs"而不是"-apple"，因为后者在文件名中使用了有限的ISO 9660字符。然而，Apple扩展名确实提供了这样的优势：文件被更高效地打包到磁盘上，并且可能在CD上安装更多的文件

HFS MACINTOSH FILENAMES

在可能的情况下，存储在Apple/Unix文件中的HFS文件名用于CD的HFS部分。但是，并不是所有的Apple/Unix编码都使用finderinfo存储HFS文件名。在这些情况下，使用Unix文件名-带有转义的特殊字符。特殊字符包括'/'和代码超过127的字符。AUFS使用': '转义这些字符，后面跟着字符代码作为两个十六进制数字。Netatalk和EtherShare有类似的方案，但使用'%'而不是': '。如果genisolimage无法找到HFS文件名，则使用Unix名称，将任何"%xx"或": xx"字符(xx是两个十六进制数字)转换为单个字符代码。如果xx不是十六进制数字([0-9a-FA-F])，则它们将被单独保留-尽管其余的": "转换为"%"，因为": "是hfs目录分隔符。必须小心，因为带有"%xx"或": xx"的普通Unix文件也将被转换。例如下面的例子

```
This:2fFile converted to This/File  
This:File converted to This%File  
This:t7File converted to This%t7File
```

sh

虽然HFS文件名似乎支持大写字母和小写字母，但文件系统不区分大小写，即文件名AbC和aBC是相同的。如果在具有相同HFS名称的目录中找到一个文件，genisolimage将尝试通过在其中一个文件名中添加"_"字符来创建唯一的名称。如果文件存在HFS文件名，genisolimage可以使用"-mac-name"选项，让此名称作为ISO 9660、Joliet和Rock Ridge文件名的起始点。没有HFS名称的普通Unix文件仍将使用它们的Unix名称。

如果在unix系统中，一个mac的二进制文件存储名字是这是someimage.gif.bin，但是系统中还有一个名字为someimage.gif的文件，那么这个名字是将出现在CD的HFS部分的名称。但是，由于genisolimage使用Unix名称作为其他名称的起点，生成的ISO 9660名称可能是SOMEIMAG.BIN，Joliet/Rock Ridge将是某某Image.gif.bin。此选项将使用HFS文件名作为起点，ISO 9660名称可能为SOMEIMAG.GIF，而Joliet/Rock Ridge可能是一些Image.gif。

"-mac-name"目前不能和"-T"一起使用。Unix名称将在TRANS.TBL文件中使用，而不是Macintosh名称。

用于将任何HFS文件名转换为Joliet/Rock Ridge文件名的字符集默认为cp 10000(Mac Roman)。使用的字符集可以使用

现有的genisoimage代码将过滤掉ISO 9660和Joliet文件名的任何非法字符，但随着genisoimage预期将直接处理Unix名称，它将保留Rock Ridge的名称。但"/"是一个合法的HFS文件名字符，“-mac-name”选项在Rock Ridge文件名中将"/"转换为"_"。

如果使用Apple扩展，则只会在Macintosh上显示ISO 9660文件名。然而，由于Macintosh ISO 9660驱动程序可以使用级别2文件名，您可以使用类似于"-allow-multidot"的选项，这在Macintosh上没有问题的。但是仍然要注意名称，例如>this.file.name"将转换为"THIS.FILE"，即只有一个'.'，也可以将文件名"abcdefg"视为"ABCDEFGH"，但"abcdefghi"将被视为"ABCDEFGHI."。也就是说，在结尾有一个"."不知道这是Macintosh问题还是genisoimage/mk混合问题。当在Macintosh上查看时，所有的文件名都是大写的。当然，DOS/Win3.X机器将无法看到二级文件名。

HFS CUSTOM VOLUME/FOLDER ICONS

若要为HFS CD提供自定义图标，请确保根(顶级)文件夹包含一个标准Macintosh卷图标文件。若要在Macintosh上为卷提供自定义图标，必须在卷的“获取信息”框中将一个图标粘贴到卷的图标上。这将在根文件夹中创建一个名为"Icon\r"(\r是回车字符)的不可见文件。自定义文件夹图标非常相似，文件夹本身存在一个名为"Icon\r"的不可见文件。

可能创建genisoimage可以使用的自定义图标的最简单方法是格式化Mac上的空白HFS软盘，并将图标粘贴到其“GET Info”框中。如果在安装hfs模块的情况下使用linux，请挂载软盘：“mount -t hfs /dev/fd0 /mnt/floppy”。默认情况下，软盘将作为CAP文件系统挂载。然后使用以下内容运行genisoimage：“genisoimage --cap -o output source_dir /mnt/floppy”。

如果您不使用Linux，您可以使用hfsutils从软盘复制图标文件。但是，必须小心，因为图标文件包含一个控制字符。例如：

```
hmount /dev/fd0
hdir -a
hcopy -m Icon^V^M icon_dir/ico
```

其中'^V^M'是"ctrl+v"，然后是控制"ctrl+m"。然后以下面的方式运行genisoimage：“genisoimage --macbin -o output source_dir icon_dir”

创建/使用自定义文件夹图标的过程非常相似-将图标粘贴到文件夹的“Get Info”框中，并将生成的"Icon\r"文件传输到genisoimage源树中的相关目录。您可能需要将图标文件隐藏在iso 9660和Joliet树中。

HFS BOOT DRIVER

可以在Macintosh上启动混合CD。可引导的HFS CD需要一个AppleCD-ROM(或兼容)驱动程序、一个可引导的HFS分区和必要的系统、Finder等文件。使用Apple_Driver实用程序，可以从任何其他Macintosh可引导CD-ROM中获得驱动程序。然后，这个文件可以与"-boot-hfs-file"一起使用。HFS分区(在本例中是混合磁盘)必须包含一个合适的系统文件夹，同样来自另一个光盘或磁盘。

要使分区可引导，必须设置其引导块。引导块位于分区的前两个块中。对于不可引导的分区，引导块中满是零。通常情况下，当系统文件被复制到Macintosh磁盘上的分区时，引导块会被许多必需的设置填充。因此，实用程序Apple_Driver还从它在给定的cd-ROM上找到的第一个hfs分区中提取引导块，这用于

genisoimage创建的hfs分区。

请注意：通过使用来自Apple CD的驱动程序并将Apple软件复制到您的CD上，您将很容易服从Apple Computer, Inc. 软件许可协议。

EL TORITO BOOT INFORMATION TABLE

当给出"-boot-info-table"时，genisoimage将修改-b指定的引导文件，方法是在文件的偏移量8处插入一个56字节的引导信息表。这一修改是在源文件系统中完成的，因此，如果该文件不易重新创建，请确保使用副本！此文件包含指针，这些指针在启动时可能不易或可靠地获得。本表的格式如下；所有整数均采用7.3.1("小Endian")格式：

Offset	Name	Size	Meaning
8	bi_pvd	4 bytes	LBA of primary volume descriptor
12	bi_file	4 bytes	LBA of boot file
16	bi_length	4 bytes	Boot file length in bytes
20	bi_csum	4 bytes	32-bit checksum
24	bi_reserved	40 bytes	Reserved

32位校验和是引导文件中从字节偏移64开始的所有32位字的总和。所有线性块地址(LBA)都以CD扇区(通常为2048字节)给出。

HFS PROBLEMS/LIMITATION

虽然HFS文件名似乎支持大写字母和小写字母，但文件系统不区分大小写，即文件名ABC和ABC是相同的。如果在具有相同HFS名称的目录中找到一个文件，genisoimage将尝试通过在其中一个文件名中添加"_"字符来创建唯一的名称。

共享前31个字符的HFS文件/目录名称有'_N'(十进制数字)替代最后几个字符以生成唯一的名称。在“嫁接”Apple/Unix文件或目录(所涉及的方法和语法见上文)时，必须小心。不能为Apple/Unix编码的文件/目录使用新名称。例如，如果要将名为oldname的Apple/Unix编码文件添加到CD中，则不能使用一下命令行：

```
"genisoimage -o output.raw -hfs -graft-points newname=oldname cd_dir"
```

genisoimage将无法解码旧名。但是，只要您不尝试像上面那样给它们命名，就可以移植Apple/unix编码的文件或目录。

当使用多会话选项-M和-C创建HFS卷时，仅上一次会话中的文件将位于HFS卷中。也就是说，genisoimage不能将以前会话中的现有文件添加到hfs卷中。但是，如果每个会话都是使用"-part"创建的，则在Mac上挂载时，每个会话将显示为单独的卷。在这种情况下，值得使用"-v"或"-hfs-volid"为每个会话提供一个唯一的卷名，否则每个“卷”都会以相同的名称出现在桌面上。符号链接(与所有其他非常规文件一样)不会添加到hfs目录中。

混合卷可能大于包含相同数据的纯ISO 9660卷。在某些情况下(例如DVD大小的卷)，差异可能很大。随着HFS卷的增大，分配块的大小(文件所能占用的最小空间)也会变大。对于650 MB的CD，分配块是10 kb，对于4.7GB的DVD，大约是70 kB。hfs卷中的最大文件数约为65500，但实际限制将略小于此。

生成的混合卷可以使用hfsutils例程在Unix机器上访问。但是，当卷被设置为锁定时，不能对其进行任何更改。这个选项"-hfs-unlock"将创建一个未锁定的输出映像-但是不应该对卷的内容进行任何更改(除非您真正知道自己在做什么)，因为它不是“真实的”hfs卷。

"-mac-name"目前不能和"-T"一起使用， -Unix名称将在TRANS.TBL文件中使用，而不是Macintosh名称。

虽然genisoimage不改变文件的内容，但如果二进制文件的类型设置为"TEXT"，则可能在Macintosh上不正确地读取它。因此，默认类型的更好选择可能是"? ? ? ?"。

"-mac-boot-file"选项可能完全不能工作。可能无法使用PC Exchangev2.2或更高版本的文件(MacOS8.1可用)。当使用Linux时，包含PC Exchange文件的DoS媒体应该以MSDOS(而不是vFAT)类型挂载。SFM格式仅部分支持。geniso映像应该能够创建超过4GB的hfs混合映像，尽管这还没有经过充分的测试。

HPPA说明

要为HPPA制作一个可引导的CD，至少必须指定一个引导加载程序文件(-hppa-boot-loader)、一个内核映像文件(32位、64位，或者两者兼而有之，视硬件而定)和一个引导命令行(-hppa-cmdline)。有些系统可以启动32位或64位的内核，如果两者都存在，固件将选择其中一种。还可以选择使用"-hppa-cmdline"对根文件系统使用ramdisk。

JIGDO说明

Jigdo是一个帮助分发大文件的工具，如cd和dvd镜像；更多细节请参见<http://atterer.net/jigdo/>。debian cd和dvd iso镜像以jigdo格式在网络上发布，以便最终用户更有效地下载它们。若要从genisoimage在ISO映像旁边创建jigdo和模板文件，您必须首先生成将使用的文件列表，格式如下：

MD5sum File size Path

32 chars 12 chars to end of line

MD5sum应该用jigdo的伪BASE 64格式编写。文件大小应该是十进制，文件的路径必须是绝对的。拥有此文件后，使用所有常规命令行参数调用genisoimage。使用"-jigdo-jigdo"和"-jigdo-Template"指定jigdo和模板文件的输出文件名，并使用"-md5-list"传递md5列表的位置。

如果您不希望将一些文件添加到jigdo文件中(例如，如果它们可能经常更改)，请使用-jigdo忽略指定它们。如果您想在将一些文件写入镜像时验证它们，请使用"-jigdo-force-md5"指定它们。如果任何文件不匹配，genisoimage就会中止。这两个选项都以正则表达式作为输入。可以根据大小，使用"-jigdo-min-file-size"选项限制将进一步使用的文件集。

最后，jigdo代码需要知道如何将给定的文件映射到镜像样式的配置中。指定如何使用"-jigdo-map"映射路径。使用"debian=/mirror/debian"将导致所有以"/mirror/debian"开头的路径映射到输出jigdo文件中的debian:。

例子代码

1) 若要在cd.iso文件中创建一个普通的ISO 9660文件系统映像，如果CD为cd.iso，目录cd_dir将成为根目录，使用命令：

```
% genisoimage -o cd.iso cd_dir
```

sh

2) 使用目录cd_dir的Rock Ridge扩展创建CD

```
% genisoimage -o cd.iso -R cd_dir
```

sh

3) 若要创建源目录cd_dir的RockRidge扩展名的CD，其中所有文件至少具有读取权限，且所有文件均为root所有，使用命令：

```
% genisoimage -o cd.iso -r cd_dir
```

sh

4) 将tar存档直接写入CD，该CD将包含带有tar存档的简单iso 9660文件系统。使用命令：

```
% tar cf - . | genisoimage -stream-media-size 333000 | wodim dev=b,t,l -dao tsize=333000s -
```

sh

5) 用源目录cd_dir的Joliet和Rock Ridge扩展创建HFS混合CD

```
% genisoimage -o cd.iso -R -J -hfs cd_dir
```

sh

6) 从包含Netatalk Apple/Unix文件的源目录cd_dir创建HFS混合CD

```
% genisoimage -o cd.iso --netatalk cd_dir
```

sh

7) 要从源目录cd_dir创建HFS混合CD，只需根据文件“映射”中列出的文件扩展名提供所有文件创建者和类型

```
% genisoimage -o cd.iso -map mapping cd_dir
```

sh

8) 要创建一个带有Apple扩展到ISO 9660的CD，可以从源目录cd_dir和another_dir。解码所有已知Apple/Unix格式的文件，并根据文件魔术中给出的魔术号给出任何其他文件的创建者和类型。

```
% genisoimage -o cd.iso -apple -magic magic -probe cd_dir another_dir
```

sh

9) 下面的示例在CD上放置不同的文件，这些文件都有自述的名称，但当被视为ISO 9660/Rock Ridge、Joliet或HFS CD时，它们的内容不同。当前目录包含“README.hfs README.joliet README.Unix cd_dir/”。下面的命令将cd_dir目录的内容连同三个自述文件放在CD上，但在这三个文件系统中只能看到一个

```
% genisoimage -o cd.iso -hfs -J -r -graft-points \
-hide README.hfs -hide README.joliet \
-hide-joliet README.hfs -hide-joliet README.Unix \
-hide-hfs README.joliet -hide-hfs README.Unix \
README=README.hfs README=README.joliet \
README=README.Unix cd_dir
```

sh

文件README.hfs将被视为HFS CD上的自述文件，另外两个自述文件将被隐藏。同样适用于Joliet和ISO 9660/Rock Ridge CD。隐藏选项的组合可能会产生各种奇怪的结果。

说明

genisoimage可以安全地安装在suid root中。这可能是为了允许genisoimage在创建多会话映像时读取上一次会话。如果genisoimage正在创建具有Rock Ridge属性的文件系统映像，而源目录树的目录嵌套级别对于ISO 9660来说太高，genisoimage将执行深度目录重定位。这将导致CD根目录中的一个名为RR_Move的目录。您无法避免此目录。不同平台的许多引导代码选项是互斥的，因为引导块不能共存，即不同平台共享映像中相同的数据位置。

举例

创建iso文件

```
[sogrey@bogon 文档]$ mkisofs -v -o my.iso /sogrey/
I: -input-charset not specified, using utf-8 (detected in locale settings)
genisoimage 1.1.9 (Linux)
Scanning /sogrey/
Scanning /sogrey/wjtpf1R
Writing: Initial Padblock                         Start Block 0
Done with: Initial Padblock                      Block(s) 16
Writing: Primary Volume Descriptor               Start Block 16
Done with: Primary Volume Descriptor            Block(s) 1
Writing: End Volume Descriptor                  Start Block 17
Done with: End Volume Descriptor                Block(s) 1
Writing: Version block                          Start Block 18
Done with: Version block                        Block(s) 1
Writing: Path table                            Start Block 19
Done with: Path table                          Block(s) 4
Writing: Directory tree                         Start Block 23
Done with: Directory tree                      Block(s) 2
Writing: Directory tree cleanup                Start Block 25
Done with: Directory tree cleanup              Block(s) 0
Writing: The File(s)                           Start Block 25
Total translation table size: 0
Total rockridge attributes bytes: 0
Total directory bytes: 2048
Path table size(bytes): 26
Done with: The File(s)                         Block(s) 0
Writing: Ending Padblock                       Start Block 25
Done with: Ending Padblock                     Block(s) 150
Max brk space used 0
175 extents written (0 MB)
You have new mail in /var/spool/mail/root
[root@localhost sogrey]$ ls
1.c  my.iso  wj123.kpET  wj123.oH2o4P  wj234.q1C  wjtpf1R
[root@localhost sogrey]$
```

mknod - 创建字符设备文件和块设备文件

mknod命令 用于创建Linux中的字符设备文件和块设备文件。

创建块设备或者字符设备文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mknod [选项] 设备名 设备类型 主设备号 次设备号
```

sh

选项

```
--help          # 显示帮助文档  
--version       # 显示命令版本信息  
  
-m, --mode=MODE # 设置权限  
-Z, --context=CTX # 设置SELinux的安全上下文  
设备类型  
设备号          # 只能是十进制和八进制, 如果是0x开头, 那么还是十六进制。对于b、c、u设备,
```

sh

当类型为b、c或u时，必须同时指定主设备号和次设备号；当类型为p时，必须省略它们。如果主设备号或次设备号以0x或0X开头，则解释为十六进制；否则，如果以0开头为八进制，其他为十进制。

扩展知识

Linux的设备管理是和文件系统紧密结合的，各种设备都以文件的形式存放在/dev目录下，称为设备文件。应用程序可以打开、关闭和读写这些设备文件，完成对设备的操作，就像操作普通的数据文件一样。

为了管理这些设备，系统为设备编了号，每个设备号又分为主设备号和次设备号。主设备号用来区分不同种类的设备，而次设备号用来区分同一类型的多个设备。对于常用设备，Linux有约定俗成的编号，如硬盘的主设备号是3。

Linux为所有的设备文件都提供了统一的操作函数接口，方法是使用数据结构struct file_operations。这个数据结构中包括许多操作函数的指针，如open()、close()、read()和write()等，但由于外设的种类较多，操作方式各不相同。Struct file_operations结构体中的成员为一系列的接口函数，如用于读/写的read/write函数和用于控制的ioctl等。

打开一个文件就是调用这个文件file_operations中的open操作。不同类型的文件有不同的file_operations成员函数，如普通的磁盘数据文件，接口函数完成磁盘数据块读写操作；而对于各种设备文件，则最终调用各自驱动程序中的I/O函数进行具体设备的操作。这样，应用程序根本不必考虑操作的是设备还是普通文

件，可一律当作文件处理，具有非常清晰统一的I/O接口。所以file_operations是文件层次的I/O接口

举例

创建块设备文件

```
[root@bogon 文档]$ mknod /dev/sdb4 b 1 1      #创建一个设备
[root@bogon 文档]$ ls -l /dev/sdb4
brw-r--r-- 1 root root 1, 1 9月  7 08:21 /dev/sdb4
```

sh

创建字符设备文件

```
[root@bogon 文档]$ mknod /dev/ttywj c 2 1    #创建一个设备
[root@bogon 文档]$ ls -l /dev/ttywj
crw-r--r-- 1 root root 2, 1 9月  7 09:42 /dev/ttywj
```

sh

mkswap - 建立和设置SWAP交换分区

mkswap命令 用于在一个文件或者设备上建立交换分区。在建立完之后要使用swapon命令开始使用这个交换区。最后一个选择性参数指定了交换区的大小，但是这个参数是为了向后兼容设置的，没有使用的必要，一般都将整个文件或者设备作为交换区。

在Linux设备或者文件中创建交换分区，创建完成之后必须使用swapon来使用它。一般在“/etc/fstab”中有一个交换分区列表，这样开机的时候就可以使用它。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mkswap [-c] [-f] [-p PSZ] [-L label] [-U uuid] device [size]
```

sh

参数device通常是一个磁盘分区(类似于/dev/sdb7)，但也可以是一个文件。Linux内核不查看分区ID，但是许多安装脚本将假定十六进制类型82(Linux_SWAP)的分区是交换分区。(警告：Solaris也使用此类型。小心不要关闭Solaris分区。)

参数size是多余的，但为了向后兼容性而保留

PSZ参数指定要使用的页大小。指定它几乎是不必要的(甚至是不明智的)，但是某些旧的libc版本是关于页面大小的，所以mkswap有可能弄错了它。症状是后续Swapon失败了，因为没有找到交换签名。PSZ的典型值为4096或8192。

选项

```
-c          # 创建交换分区之前，检测坏块。如果有，那么打印出数量。  
-f          # 强制执行。如果没有此选项，mkswap将拒绝擦除带有分区表的设备上的第一个块或整个磁盘上的第一个  
-p          # 设置页大小，默认4096。一般是不需要指定这个参数的  
-L label    # 指定一个label，方便swapon使用。只适用于新的风格交换区域  
-v1         # 创建v1版本的swap分区，2.5内核之后只支持这种的  
-U uuid     # 指定uuid，默认情况会生成uuid
```

sh

说明

交换头不触及第一个块。引导加载程序或磁盘标签可以在那里，但不建议安装。建议的设置是为Linux交换区域使用单独的分区。mkswap和许多其他类似mkfs的实用程序一样，擦除了第一块，这样就可以删除磁盘上的旧系统。mkswap拒绝擦除带有磁盘标签的设备上的第一个块，或者整个磁盘。

交换区域的最大有用大小取决于体系结构和内核版本，在i386、PPC、m68k、ARM、sparc上为1 GiB、MIPS上为512 MiB、alpha上为128 GiB、sparc64上为3 TiB。对于2.3.3以后的内核来说，没有这样的限

制。请注意，在2.1.117之前，内核为每个页面分配了一个字节，而现在它分配了两个字节，因此使用中的交换区域2 GiB可能需要2个MiB内核内存。

目前，Linux允许32个交换区域(这是Linux2.4.10之前的8个)。使用中的区域可以在文件“/proc/swaps”中看到(自2.1.25以来)。mkswap拒绝10页以下的区域。如果您不知道您的机器使用的页面大小，您可能可以使用“cat /proc/cpuinfo”查找它。

若要设置交换文件，必须在使用mkswap初始化该文件之前创建该文件，例如使用以下命令：

```
# dd if=/dev/zero of=swapfile bs=1024 count=65536
```

sh

注意，交换文件不能包含任何漏洞(因此，使用cp(1)创建该文件是不可接受的)

举例

创建交换分区，指定页大小2048

```
[root@bogon ~]$ mkswap -p 2048 /dev/sdb4      #这里指定页大小2048，取代了系统默认的4096
Using user-specified page size 2048, instead of the system value 4096
Setting up swap space version 1, size = 16382 KiB
no label, UUID=42f07b0e-0adb-47b6-a906-1209efabb981
```

sh

创建交换分区，指定页大小4096，指定label

```
[root@bogon ~]$ mkswap -p 4096 -L wj /dev/sdb4    #笔者当前的系统，不能使用2048大小的页
Setting up swap space version 1, size = 16380 KiB
LABEL=wj, UUID=c458a15d-50ee-4e10-a49b-b59add4879d5
```

sh

使用交换分区

```
[root@bogon ~]$ swapon /dev/sdb4          #使用指定的分区
[root@bogon ~]$ swapon -s                  #查看分区使用情况
Filename      Type      Size     Used   Priority
/dev/dm-1     partition 2940920  0       -1
/dev/sdb4     partition 16376   0       -2
```

sh

mktemp - 创建临时文件供shell脚本使用

mktemp命令 被用来创建临时文件供shell脚本使用。

创建临时文件或者目录，这样的创建方式是安全的。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mktemp [选项] [TEMPLATE]
```

sh

选项

```
-d, --directory          # 创建目录
-u, --dry-run            # 不要创建任何东西，只要打印一个名字（不安全）
-q, --quiet              # 发生错误的时候不显示提示信息
--suffix=SUFF            # 附加SUFF到模板中。SUFF不能包含斜杠。如果模板不以X结尾，则使用此选项。
--tmpdir[=dir]           # 指定临时文件的路径，如果tmpdir后面没有路径，那么使用变量$TMPDIR；如果这个变量未设置，那么使用/tmp或$XDG_RUNTIME_DIR
-p DIR                  # 使用DIR作为前缀
-t                      # 将模板解释为一个相对于目录$TMPDIR(如果设置)的单个文件名组件；否则通过-p指定的DIR
TEMPLATE               # 临时文件名，名字中必须包含至少3个字母X。如果没有指定，那么默认是tmp.XXXXXXXXXX

--help                  # 显示帮助文档
--version               # 显示命令版本信息
```

sh

举例

创建临时文件

```
[root@bogon 文档]$ mktemp wj123.XXXX      #名字包含4个X
wj123.kpET
You have new mail in /var/spool/mail/root
[root@bogon 文档]$ mktemp wj123.XXXXXX    #名字包含6个X
wj123.oH2o4P
[root@bogon 文档]$ ls
1.c  wj123.kpET  wj123.oH2o4P
```

sh

创建临时目录

sh

```
[root@bogon 文档]$ mktemp -d wjtp    #名字中没有X
mktemp: 模板"wjtp" 中X 太少
[root@bogon 文档]$ mktemp -d wjtpXXX    #名字中没有X, 这里可以看到X必须是大写的
mktemp: 模板"wjtpXXX" 中X 太少
[root@bogon 文档]$ mktemp -d wjtpXXX    #创建成功
wjtpf1R
[root@bogon 文档]$ ls -l
总用量 4
-rw-r--r-- 1 root root    0 9月   7 09:11 1.c
-rw----- 1 root root    0 9月   7 14:47 wj123.kpET
-rw----- 1 root root    0 9月   7 14:47 wj123.oH2o4P
drwx----- 2 root root 4096 9月   7 14:50 wjtpf1R
```

在/tmp中创建临时文件

sh

```
[root@bogon 文档]$ mktemp --tmpdir wj234.XXX #tmpdir没有指定路径, 在tmp下创建
/tmp/wj234.BNy
You have new mail in /var/spool/mail/root
```

在指定目录下创建临时目录

sh

```
[root@bogon 文档]$ mktemp --tmpdir=/weijie wj234.XXX #在tmpdir指定的路径下创建
/weijie/wj234.q1C
[root@bogon 文档]$ ls
1.c  wj123.kpET  wj123.oH2o4P  wj234.q1C  wjtpf1R
```

使用选项-u创建

sh

```
[root@bogon 文档]$ mktemp -u wj123.XXXXXX #使用-u选项
wj123.dSgIK1
[root@bogon 文档]$ ls      #看不到临时文件
1.c  wj123.kpET  wj123.oH2o4P  wj234.q1C  wjtpf1R
```

mmd - 在ms-dos系统中创建目录

在ms-dos系统中创建目录。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
mmd 目录
```

sh

选项

无

举例

在ms-dos文件系统中创建目录

```
[sogrey@bogon 文档]$ pwd
/home/sogrey/文档

[root@localhost test]$ mount /dev/sdb4 /media/test
[root@localhost test]$ cd /media/test #切换到ms-dos
[root@localhost test]$ mmd mydir #创建目录
[root@localhost test]$ ls #查看是否创建成功
mydir
```

sh

modprobe - 自动处理可载入模块

modprobe命令 用于智能地向内核中加载模块或者从内核中移除模块。

modprobe可载入指定的个别模块，或是载入一组相依的模块。modprobe会根据depmod所产生的相依关系，决定要载入哪些模块。若在载入过程中发生错误，在modprobe会卸载整组的模块。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
modprobe [OPTION] [参数]
```

sh

模块名：要加载或移除的模块名称。

选项

```
-a, --all          # 载入全部的模块;
-c, --show-conf   # 显示所有模块的设置信息;
-d, --debug        # 使用排错模式;
-l, --list         # 显示可用的模块;
-r, --remove       # 模块闲置不用时，即自动卸载模块;
-t, --type         # 指定模块类型;
-v, --verbose      # 执行时显示详细的信息;
-V, --version       # 显示版本信息;
-help              # 显示帮助。
```

sh

举例

查看modules的配置文件：

```
modprobe -c
```

sh

这里，可以查看modules的配置文件，比如模块的alias别名是什么等。会打印许多行信息，例如其中的一行会类似如下：

```
alias symbol:ip_conntrack_unregister_notifier ip_conntrack
```

sh

列出内核中所有已经或者未挂载的所有模块：

```
modprobe -l
```

sh

这里，我们能查看到我们所需要的模块，然后根据我们的需要来挂载；其实modprobe -l读取的模块列表就位于/lib/modules/ `uname -r` 目录中；其中uname -r是内核的版本，例如输出结果的其中一行是：

```
/lib/modules/2.6.18-348.6.1.el5/kernel/net/netfilter/xt_statistic.ko
```

sh

挂载vfat模块：

```
modprobe vfat
```

sh

这里，使用格式modprobe 模块名来挂载一个模块。挂载之后，用lsmod可以查看已经挂载的模块。模块名是不能带有后缀的，我们通过modprobe -l所看到的模块，都是带有.ko或.o后缀。

移除已经加载的模块：

```
modprobe -r 模块名
```

sh

这里，移除已加载的模块，和rmmod功能相同。

mount - 用于挂载Linux系统外的文件

mount命令 Linux mount命令是经常会使用到的命令，它用于挂载Linux系统外的文件。

将指定设备的文件系统挂载到系统目录下，设备挂载之后就可以访问了。在Unix系统中可访问的所有文件都排列在一个大树中，即文件层次结构，根植于/。这些文件可以分布在多个设备上。挂载命令用于将在某个设备上找到的文件系统附加到大文件树上。相反，umount(8)命令将再次分离它。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mount [选项] [src] [dst]
mount [-lhV]
mount -a [-fFnrsvw] [-t vfstype] [-O optlist]
mount [-fnrsvw] [-o option[,option]...] device|dir
mount [-fnrsvw] [-t vfstype] [-o options] device dir
```

sh

选项

命令行选项

调用mount所使用的全部挂载选项集是通过首先从fstab表中提取文件系统的挂载选项，然后应用"-o"参数指定的任何选项，然后在出现时应用"-r"或"-w"选项来确定的。

```

-a, --all          # 挂载/etc/fstab中的所有文件系统
-f, --fake         # 除了系统调用外, 执行挂载的所有过程, 一般和“-v”一起使用。这是一个模拟挂载过程
-F, --fork         # 和“-a”一起使用, 并行挂载多个设备, 加快挂载速度。一个缺点是挂载是以未定义的顺序完
-i, --internal-only # 即使存在/sbin/mount, 也不要调用/sbin/mount.<filesystem>Helper
-l
-n, --no-mtab      # 在mount输出中添加标签。mount必须具有读取磁盘设备(例如, suid root)的权限, 才能工
--no-canonicalize # 不要把路径规范化。挂载命令将所有路径(从命令行或fstab)规范化, 并存储到/etc/mtab
-p, --pass-fd num  # 在使用加密的循环挂载情况下, 从文件描述符num读取密码短语, 而不是从终端读取密码。
-s
-r, --read-only    # 容忍草率的mount选项而不是失败。这将忽略文件系统类型不支持的挂载选项。并非所有文件
# 只读的方式挂载, 同义词是“-o ro”。请注意, 根据文件系统类型、状态和内核行为, 系统可能无法正
# 以读写的方式挂载, 默认选项。同义词是“-o rw”。
-w l -rw
-L label          # 挂载指定label的分区
-U uuid            # 挂载指定UUID的分区, 这两个选项要求文件“/proc/partitions”(自Linux2.1.116以来)
-t type            # 指定挂载的文件系统类型。目前支持的系统有: adfs, affs, autofs, cifs, coda, c
# 程序mount和umount支持文件系统子类型。子类型由‘.subtype’后缀定义。例如‘fuse.s
# 对于大多数类型, 挂载程序所要做的就是发出一个简单的挂载(2)系统调用, 而不需要详细了解
# 如果没有给出“-t”选项, 或者指定了auto类型, 那么挂载将尝试猜测所需的类型。挂载使用
# auto类型对于用户安装的软盘可能很有用。创建一个文件“/etc/ filesystems”可以帮助
# 可以在逗号分隔的列表中指定多个类型。文件系统类型列表可以no作为前缀, 以指定不应该
# 与“-a”一起使用, 以限制应用“-a”的一组文件系统。除了在“-a”的上下文中, 它是无用的。
# 选项使用“-o”标志指定, 后面是逗号分隔的选项字符串。例如“mount LABEL=mydisk -o
# 在其他地方重新装入子树(以便其内容在两个地方都可用)
# 重新装入子树, 并将所有可能的子树放到其他地方(以便其内容在两个地方都可用)
-M, --move         # 将子树移到其他地方
-v                 # 显示详细执行过程

--help              # 显示帮助文档
--version           # 显示命令版本信息

```

文件系统独立安装选项

只有当这些选项出现在“/etc/fstab”文件中时, 这些选项才会有用。其中一些选项可以在默认情况下在系统内核中启用或禁用。要检查当前设置, 请参阅“/proc/mount”中的选项。以下选项适用于正在挂载的任何文件系统(但并不是每个文件系统实际上都符合它们-例如, 现在的sync选项只对ext 2、ext 3、fat、vfat和ufs有效)。

```

async          # 文件系统的所有I/O都应该异步完成
atime          # 不要使用noatime特性，那么inode访问时间由内核缺省值控制。
noatime        # 不要更新此文件系统上的inode访问时间(例如，为了更快地访问新闻线轴以加快新闻服务器的运行)
auto           # 使用“-a”选项挂载
noauto         # 只能显式挂载(即-a选项不会导致安装文件系统)
context=context #
fscontext=context #
defcontext=context #
rootcontext=context # 当安装不支持扩展属性的文件系统(如用VFAT格式化的软盘或硬盘)或通常不在SELinux下运行时
# “fstcontext=”选项适用于所有文件系统，而不管它们对xattr的支持如何，fstext选项将总被忽略
# 可以使用“defcontext= option”为未标记的文件设置默认的安全上下文。这将重写策略中未标记的文件
# “rootcontext= option”允许您显式地标记在FS或inode之前挂载的FS的根inode，因为可以
defaults        #
dev            # 在文件系统上解释字符或阻塞特殊设备
nodev          # 不在在文件系统上解释字符或阻塞特殊设备
diratime        # 更新此文件系统上的目录inode访问时间。这是默认的
nodiratime      # 不要更新此文件系统上的目录inode访问时间。
dirsync         # 文件系统中的所有目录更新都应该同步进行，这会影响以下系统调用：creat、link、unlink、rmdir
exec           # 允许执行二进制文件
noexec          # 不允许在安装的文件系统上直接执行任何二进制文件。
group          # 如果普通用户的组与设备组匹配，则允许普通用户(而非root用户)挂载文件系统。此选项意味着
iversion        # 每次修改inode时，i_version字段都会递增
noiversion      # 每次修改inode时，i_version字段不变
mand            # 允许此文件系统上的强制锁
nomand          # 不允许此文件系统上的强制锁
_netdev         # 文件系统驻留在需要网络访问的设备上(用于防止系统试图在系统上启用网络之前安装这些文件系统)
nofail          # 如果此设备不存在，请不要报告它的错误
relatime        # 更新inode访问时间相对于修改或更改时间。只有在上一次访问时间早于当前修改或更改时间时
# 由于Linux2.6.30，内核默认使用此选项提供的行为(除非指定了noatime)，因此需要严格的选择
norelatime      # 不使用relatime
strictatime     # 允许显式请求完整的atime更新。这使得内核可以默认为relatime或noatime，但仍然允许用户
nostrictatime   # 使用内核的默认行为进行inode访问时间更新
suid            # 允许设置用户标识符或组标识符位生效
nosuid          # 不允许设置用户标识符或组标识符位生效
owner           # 允许普通用户(而非root用户)在设备所有者的情况下挂载文件系统。
remount         # 试图重新挂载已经挂载的文件系统。这通常用于更改文件系统的挂载标志，特别是使只读文件系
ro              # 以只读方式挂载
_rnetdev        # 类似于“_netdev”，除了“fsck-a”在rc.sysinit期间检查这个文件系统。
rw              # 以读写的方式挂载
sync            # 对文件系统的I/O应该同步执行。如果媒体的写入周期有限(例如，一些闪存驱动器)，“同步”可
user            # 允许普通用户挂载文件系统。将挂载用户的名称写入mtab，以便他可以再次卸载文件系统。
nouser          # 禁止普通用户(而非根用户)挂载文件系统，这是默认的。
users           # 允许每个用户挂载和卸载文件系统。

```

adfs选项

```

uid=value
gid=value      # 设置文件系统中文件的所有者和组(默认值：uid=gid=0)

ownmask=value
othmask=value   # 分别为ADFS“所有者”权限和“其他”权限设置权限掩码(默认值分别为0700和0077)。

```

affs选项

```
sh
uid=value
gid=value          # 设置文件系统根目录的所有者和组(默认值: uid=gid=0, 但如果使用uid或gid选项而没有指
setuid=value        # 设置所有文件的所有者和组
setgid=value        # 将所有文件的模式设置为值&0777(不考虑原始权限)。向具有读取权限的目录添加搜索权限。该
mode=value          # 不允许对文件系统上的保护位进行任何更改。
protect             # 在第一次同步或umount时, 将文件系统根的uid和gid设置为挂载点的uid和gid, 然后清除此
usemp               # 打印每个成功安装的信息消息。
verbose              # 在跟随链接时, 在卷名之前使用的前缀
prefix=string        # 前缀(长度最多为30), 在符号链接后面时使用在'/'之前。
volume=string        # (默认: 2.)设备开始时未使用的块数
reserved=value       # 显式地给出根块的位置。
root=value           # 设置块大小。允许值为512, 1024, 2048, 4096。
bs=value             # grpquota|noquota|quota|usrquota # 这些选项被接受, 但被忽略。(但是, 配额实用程序可以对/etc/fSTAB中的此类

```

cifs选项

参考mount.cifs。

coherent选项

没有

debugfs选项

调试器文件系统是一个伪文件系统, 传统上安装在"/sys/kernel/debug"上。没有选项

devpts选项

devpt文件系统是一个伪文件系统, 传统上安装在"/dev/pts"上。为了获得伪终端, 进程打开"/dev/ptmx"; 然后将伪终端的数量提供给进程, 伪终端从站可以作为"/dev/pt/"访问。

```
sh
uid=value
gid=value          # 这将新创建的PTY的所有者或组设置为指定的值。当未指定任何内容时, 它们将被设置为创建过程
mode=value          # 将新创建的PTY的mode设置为指定的值。默认值为0600。mode=620和gid=5的值使“mesg”成为
newinstance         # 创建devpt文件系统的私有实例, 以便在这个新实例中分配的ptys索引独立于在其他devpt实例中
ptmxmode=value      # 在devpt文件系统中为新的ptmx设备节点设置mode。随着对devpt的多个实例的支持(请参阅上面
# 为了与较早版本的内核兼容, 新ptmx节点的默认模式是0000。 “ptmxmode=value”为ptmx节点指

```

ext选项

没有选项。请注意, "ext"文件系统已经过时。不要使用它, 因为Linux2.1.21版本的extfs不再是内核源代码的一部分。

ext2选项

'ext2'文件系统是标准的Linux文件系统, 从Linux2.5.46开始, 对于大多数挂载选项, 默认是由文件系统超级块决定的。

```
sh
acl, noacl          # 支持POSIX的访问权限，或者不支持
bsddf|minixdf       # 设置statfs系统调用的行为。minixdf行为是在f_block字段中返回文件系统的块总数，而bsdd
% mount /k -o minixdf; df /k; umount /k
Filesystem  1024-blocks  Used Available Capacity Mounted on
/dev/sda6      2630655   86954  2412169      3%   /k
% mount /k -o bsddf; df /k; umount /k
Filesystem  1024-blocks  Used Available Capacity Mounted on
/dev/sda6      2543714     13  2412169      0%   /k
check={none|nocheck} # 在挂载时不进行检查。这是默认的。这是快速的。明智的做法是时不时地调用e2fsck(8)
debug           # 在每个(重新)挂载上打印调试信息
errors={continue|remount-ro|panic} # 定义遇到错误时的行为。(要么忽略错误，标记文件系统错误并继续，要么立
grpgid|bsdgroups
nogrpuid|sysvgroups # 这些选项定义了新创建的文件所获得的组id。当设置grpuid时，它接受创建它的目录的组id
grpquota|noquota|quota|usrquota # 这些选项被接受，但被忽略
nobh             # 不要将buffer_heads附加到文件分页缓存
nouuid32        # 禁用32位UID和GID。这是为了与只存储和期望16位值的旧内核的互操作性。
oldalloc, orlov  # 对新节点使用旧分配器或Orlov分配器。Orlov是默认的。
resgid=n        # ext 2文件系统保留一定比例的可用空间(默认为5%，参见mke2fs(8)和tune2fs(8)，这些选项
resuid=n        # 使用块n作为超级块，而不是块1。当文件系统被损坏时，这可能很有用。(早些时候，超级块
sb=n            # 使用块n作为超级块，而不是块1。当文件系统被损坏时，这可能很有用。(早些时候，超级块
user_xattr|nouser_xattr # 支持“用户”扩展属性
```

ext3选项

ext3文件系统是ext2文件系统的一个版本，它通过日志记录得到了增强。它支持与ext 2相同的选项以及以下额外内容

```
sh
journal=update      # 显示帮助信息
journal=inode        # 将ext3文件系统日志更新为当前格式
journal_dev=devnum   # 当外部日志设备的主要/次要数字发生更改时，此选项允许用户指定新的日志位置。日志设
norecovery/noload   # 不要在挂载时加载日志。请注意，如果文件系统没有干净地卸载，跳过日志重播将导致文件
data={journal|ordered|writeback} # 指定文件数据的日志记录模式。元数据总是日志记录。若要使用根文件系统上持
                           # - journal，在写入主文件系统之前，所有数据都提交到日志中。
                           # - ordered，这是默认模式。在将所有数据的元数据提交到日志之前，所有数据都直接被强
                           # - writeback，数据排序没有被保留-在将数据元数据提交到日志之后，可以将数据写入主
barrier=0 / barrier=1 # 启用/禁用了障碍。barrier =0禁用了它，barrier =1启用了它。写屏障强制在磁盘上对
commit=nrsec         # 每nrsec秒同步所有数据和元数据。默认值为5秒。零代表默认。
user_xattr           # 启用扩展用户属性
acl                  # 启用POSIX访问控制列表
```

12) ext4选项

ext4文件系统是ext3文件系统的高级级别，它结合了支持大型文件系统的可伸缩性和可靠性增强功能。下面这些选项向后兼容ext3或ext2

```
journal_checksum          # 启用日记账事务的校验和。这将允许e2fsck中的恢复代码和内核检测内核中的损坏。这
journal_async_commit      # 提交块可以写入磁盘，而无需等待描述符块。如果启用，旧内核无法挂载设备。
journal=update             # 将ext4文件系统的日志更新为当前格式
barrier=0
barrier=1
barrier
nobarrier
inode_readahead_blocks=n   # 这允许/禁止在JBD代码中使用写屏障。barrier =0禁用，barrier =1启用。这还需要
                           # 此调优参数控制最大数目的inode表块，ext4的inode表ReadAhead算法将预读到缓冲区
                           # mballoc将尝试用于分配、大小和对齐的文件系统块的数目。对于RAID 5/6系统，这应该
                           # 将块分配推迟到写入时间。
delalloc
nodealloc
max_batch_time=usec        # ext 4应该等待额外的文件系统操作与同步写入操作一起批处理最大时间。由于同步写入
                           # 此参数将提交时间(如上所述)设置为至少min_batch_time。默认为零微秒。增加此参数
                           # I/O优先级(从0到7，其中0是最高优先级)，在提交操作期间，应该使用它来执行kJournal
abort
auto_da_alloc|noauto_da_alloc # 当noautoda_alloc通过以下模式替换现有文件时，许多损坏的应用程序不使用fsck
                           # 如果启用了auto_da_alloc，ext 4将检测replace-via-rename和replace-via-truncate
discard/nodiscard          # 控制在释放块时ext4是否应该向基础块设备发出discard/TRIM命令。这对于ssd设备和稀
                           # 疏散存储器很重要。
nouid32
resize
block_validity/noblock_validity # 该选项允许启用或禁用内核中的工具，用于跟踪内部数据结构中的文件系统元数据
dioread_lock/dioread_nolock   # 控制ext4是否应该使用DIO读取锁定。如果指定dioread_nolock选项，ext4将在
i_version
                           # 启用64位inode版本支持。默认情况下，此选项已关闭。
```

fat选项

```
blocksize={512|1024|2048}      # 设置块大小(默认512)。这一选择已经过时。
uid=value
gid=value
umask=value
dmask=value
fmask=value
allow_utime=value
check=value
codepage=value
conv={b[inary]|t[ext]|a[uto]} # FAT文件系统可以在内核中执行“CRLF<->NL”(MS-DOS文本格式到UNIX文本格式)的转换
                           # - binary，不执行转换。这是默认的。
                           # - text，对所有文件执行“CRLF<->NL”。
                           # - auto，CRLF<->NL翻译是对所有没有“众所周知的二进制”扩展名的文件执行的。
cvf_format=module
cvf_option=option
debug
fat={12|16|32}
iocharset=value
tz=UTC
quiet
showexec
sys_immutable
flush
usefree
dots, nodots, dotsOK=[yes|no] # 将Unix或DOS约定强加于FAT文件系统的各种错误尝试
```

hfs选项

```
creator=cccc          # 设置创建者/类型值，如用于创建新文件的MacOS查找器所示。默认值：“? ? ? ?”  
type=cccc  
uid=n, gid=n        # 设置所有文件的所有者和组。(默认值：当前进程的uid和gid。)  
dir_umask=n,  
file_umask=n,  
umask=n              # 设置用于所有目录、所有常规文件或所有文件和目录的umask。默认为当前进程的umask。  
session=n  
part=n               # 从设备中选择分区号n。只对CDROMS有意义。默认情况下根本不解析分区表  
quiet                # 静默模式，不要抱怨挂载选项无效
```

hpfs选项

```
uid=value  
gid=value            # 设置所有文件的所有者和组。(默认值：当前进程的uid和gid。)  
umask=value          # 设置umask(不存在的权限的位掩码)。默认的是当前进程的umask。这个值是以八进制表示的。  
case={lower|asis}    # 将所有文件名转换为小写，或保留它们。(默认值：case=lower。)  
conv={binary|text|auto} # 对于conv=text，在读取文件时删除一些随机CRS(特别是所有后面跟着NL)。对于binary，忽略。  
nocheck              # 当某些一致性检查失败时，不要中止安装。
```

iso9660选项

ISO 9660是描述用于CD-ROM的文件系统结构的标准。(在一些DVD上也可以看到这种文件系统类型。还请参阅UDF文件系统。)正常的iso 9660文件名以8.3格式出现(即，对文件名长度的类似DOS的限制)，此外，所有字符都是大写的。此外，也没有文件所有权、保护、链接数量、块/字符设备的设置等字段。

RockRidge是对iso 9660的扩展，它提供了所有类似Unix的特性。基本上，每个目录记录都有提供所有附加信息的扩展，而且当Rock Ridge在使用时，文件系统与普通UNIX文件系统是无法区分的(当然，它是只读的)。

```
norock                # 显示帮助信息  
nojoliet              # 禁用Rock Ridge扩展的使用，即使可用  
check={r[e]laxed}|s[trict] # 使用check=relaxed，文件名在进行查找之前首先被转换为小写。这可能只有与normal卷一起使用。  
uid=value  
gid=value              # 为文件系统中的所有文件提供指定的用户或组id，可能会覆盖RockRidge扩展中的信息。  
map={n[ormal]|o[ff]|a[corn]} # 对于非Rock Ridge卷，普通名称转换为从大写到小写的ASCII，删除一个尾随‘；’。  
mode=value             # 对于非Rock Ridge卷，给所有文件指定模式。(默认情况下：每个人都有读取权限。)  
unhide                # 还显示隐藏的和相关的文件。(如果普通文件和关联的或隐藏的文件具有相同的文件名，则显示关联的文件。)  
block={512|1024|2048} # 将块大小设置为指示值。(默认值：Block=1024。)  
conv={a[uto]|b[inary]|m[text]|t[ext]} # (默认值：conv=binary。)由于Linux1.3.54，此选项不再起作用。(未实现)  
cruft                 # 如果文件长度的高字节包含其他垃圾，请设置此挂载选项以忽略文件长度的高阶位。  
session=x  
sbsector=xxx           # 回话从xxx区开始。  
iocharset=value         # 用于将CD上的16位Unicode字符转换为8位字符的字符集。默认值为iso_8859-1。  
utf8                  # 将cd上的16位Unicode字符转换为utf-8
```

jfs选项

```
iocharset=name          # 用于将Unicode转换为ASCII的字符集。默认情况是不进行转换。使用iocharset=lu
resize=value            # 调整卷的块数。JFS只支持增长卷，而不支持缩小卷。此选项仅在重装入时有效，当
nointegrity            # 不要写日志。此选项的主要用途是在从备份媒体还原卷时允许更高的性能。如果系统
integrity               # 默认。将元数据更改提交到日志。使用此选项可重新装入以前指定no完整性选项的卷
errors={continue|remount-ro|panic} # 定义遇到错误时的行为。(要么忽略错误，标记文件系统错误并继续，要么重
noquota|quota|usrquota|grpquota # 这些选项有效，但是被忽略
```

minix选项

没有

msdos选项

请参阅FAT的安装选项。如果MSDOS文件系统检测到不一致性，它会报告一个错误并设置文件系统只读。通过重新安装文件系统，可以再次使其可写。

ncpfs选项

就像nfs一样，ncpfs实现需要一个二进制参数(Structncp_Mad_Data)到挂载系统调用。这个参数是由ncpmount(8)构造的，并且当前版本的挂载(2.12)不知道任何关于ncpfs的信息。

nfs和nfs4选项

请参阅nfs(5)手册页的选项部分(必须安装nfs-utils包)。nfs和nfs4实现需要对挂载系统调用使用二进制参数(Structnfs_Armad_Data)。这个参数是由mount.nfs(8)构造的，而且当前版本的挂载(2.13)不知道任何关于nfs和nfs4的信息。

ntfs选项

```
iocharset=name          # 当返回文件名时要使用的字符集。与VFAT不同，NTFS禁止包含不可转换字符的名称。不
nls=name                # 前面名为iocharset的选项的新名称
utf8                     # 使用UTF-8转换文件名。
uni_xlate={0|1|2}        # 如果是0(或“no”或“false”)，不要对未知Unicode字符使用转义序列。如果是1(或“
posix=[0|1]              # 如果启用(POSIX=1)，则文件系统区分大小写。8.3别名被表示为硬链接，而不是被抑
uid=value                # 在文件系统上设置文件权限。umask值以八进制为单位。默认情况下，这些文件由root
gid=value
umask=value
```

proc选项

```
uid=value
gid=value          # 目前没有效果
```

ramfs选项

Ramfs是一个基于内存的文件系统。从Linux2.3.99pre4开始。没有挂载选项

reiserfs选项

Reiserfs是一个日志文件系统。

```
conv          # 指示3.6ReiserFS软件使用新创建的对象的3.6格式挂载3.5版本的文件系统。此文件系统将  
hash={rupasov|tea|r5|detect} # 选择ReiserFS将用于查找目录中文件的散列函数  
                           # rupasov, 一种由Yury Yu. Rupasov发明的哈希。它快速并保持局部性, 将按字典顺序关闭  
                           # tea, 由Jeremy Fitzhardinge实现的Davis-Meyer函数。它在名称中使用散列置换位。它  
                           # r5, 一个修改版本的rupasov散列。默认情况下, 它是最好的选择, 除非文件系统有巨大的目  
                           # detect, 通过检查正在挂载的文件系统, 指示挂载以检测正在使用的哈希函数, 并将这些信息  
hashed_relocation # 调块分配器。这可能在某些情况下提供性能改进  
no_unhashed_relocation # 调块分配器。这可能在某些情况下提供性能改进  
noborder          # 禁用Yury Yu发明的边界分配器算法  
nolog             # 禁用日志记录。这将在某些情况下提供轻微的性能改进, 代价是失去ReiserFS在崩溃中的快  
notail            # 默认情况下, ReiserFS将小文件和“文件尾”直接存储到其树中。这就混淆了一些实用程序,  
replayonly        # 重播日志中的事务, 但不要实际挂载文件系统。主要用于reiserfsck。  
resize=number     # 重新装入选项, 它允许在线扩展ReiserFS分区。指示ReiserFS假定设备有数字块。此选项设  
user_xattr        # 启用扩展用户属性  
acl               # 启用POSIX访问控制列表  
barrier=none / barrier=flush # 这允许/禁用在日记代码中使用写屏障。barrier=none用它, barrier=flush启用
```

romfs选项

无

squashfs选项

无

smbfs选项

与nfs一样, smbfs实现需要一个二进制参数(smb_mount_data)到挂载系统调用。这个参数是由smbmount(8)构造的, 而当前版本的挂载(2.12)不知道任何有关smbfs的信息。

sysv选项

无

ubifs选项

UBIFS是一个在UBI卷之上工作的闪存文件系统。请注意, atime不受支持, 并且总是被关闭。设备名称可以指定为

ubiX_Y, UBI设备号X, 卷标Y。

ubiY, UBI设备号0, 卷标Y

ubiX:NAME, UBI设备号X, 卷标NAME。

ubi:NAME, UBI设备号0, 卷标NAME。

可以使用“! ”代替分隔符：“”。

```
bulk_read      # 启用大容量读取。VFS预读是禁用的, 因为它减慢了文件系统的速度.批量读取是一种内部优化  
no_bulk_read   # 不要启用bulk_read。这是默认的  
chk_data_crc  # 检查数据CRC-32校验和。这是默认的  
no_chk_data_crc. # 不要检查数据CRC-32校验和。使用此选项, 文件系统不会检查crc-32校验和中的数据, 但  
compr={none|lzo|zlib} # 选择写入新文件时使用的默认压缩程序。如果使用None选项挂载, 仍然可以读取压缩文件。
```

udf选项

Udf是光存储技术协会定义的“通用磁盘格式”文件系统，经常用于dvd-rom。

```
sh
gid=          # 设置默认组
umask=         # 设置默认umask，八进制
uid=          # 设置默认用户
unhide        # 显示其他隐藏文件
undelete      # 在列表中显示已删除的文件
nostrict      # 未设定严格一致性
iocharset     # 设置NLS字符集
bs=           # 设置块大小，如果不是2048，可能不工作
novrs         # 忽略卷标序列识别
session=       # 从0设置CDROM会话计数。默认值：最后一次会话
anchor=        # 覆盖标准锚位置。缺省值：256
volume=        # 覆盖VolumeDesc位置
partition=    # 覆盖PartitionDesc位置
lastblock=    # 设置系统的最后一个块
fileset=       # 重写文件集块位置
rootdir=      # 重写根目录位置
```

ufs选项

```
sh
ufstype=value      # UFS是一个在不同操作系统中广泛使用的文件系统。问题在于实现之间的差异。一些实现的
                     # old，旧格式的UFS，这是默认的，只读的。（别忘了给出-r选项。）
                     # 44bsd，用于BSD类系统(NetBSD、FreeBSD、OpenBSD)创建的文件系统。
                     # sun，对于Sparc上由SunOS或Solaris创建的文件系统。
                     # sunx86，对于Solaris在x86上创建的文件系统
                     # hp，对于hp-ux创建的文件系统，只读
                     # nextstep，对于由NeXTStep创建的文件系统(在下一个站点上)(当前为只读)
                     # nextstep-cd，对于NextStepCDROM(块_size=2048)，只读。
                     # openstep，对于OpenStep创建的文件系统(当前仅读)。MacOSX也使用相同的文件系统类
                     # 设置错误行为：
                     # panic，如果遇到错误，则引发内核恐慌。
                     # [lock|umount|repair]，这些挂载选项目前不执行任何操作；当遇到错误时，只打印控
```

umsdos选项

查看msdos的选项

vfat选项

首先，FAT的安装选项被识别出来。dotsOK选项由vFAT显式地终止。

```
sh
uni_xlate        # 将未处理的Unicode字符转换为特殊转义序列。这使您可以备份和还原用任何Unicode字
posix            # 允许两个文件的名字相同，但是大小写不同
nonumtail        # 在尝试name~num.ext之前，先尝试使用没有序列号的短名称。
utf8             # utf8是控制台使用的Unicode的文件系统安全8位编码。可以为具有此选项的文件系统启
shortname={lower|win95|winnt|mixed} # 定义用于创建和显示符合8.3个字符的文件名的行为。如果文件的长名称存
                     # lower，在显示时强制将短名显示为小写；在短名并非全部大写时存储长名称。此模式是默
                     # win95，在显示时将短名强制显示为大写；在短名并非全部大写时存储长名称。
                     # winnt，按原样显示短名；当短名不是全部小写或全部大写时，存储一个长名称。
                     # mixed，按原样显示短名；在短名不全大写时存储长名
```

usbfs选项

```
devuid=uid          # 在usbfs文件系统中设置设备文件的所有者、组和模式(默认值: uid=gid=0, mode=0644)
devgid=gid
devmode=mode
busuid=uid          # 在usbfs文件系统中设置总线目录的所有者、组和模式(默认值: uid=gid=0, mode=0555)
busgid=gid
busmode=mode
listuid=uid          # 设置文件设备的所有者、组和模式(默认值: UID=gid=0, mode=0444)。该模式以八进制表示
listgid=gid
listmode=mode
```

sh

xenix选项

无

xfs选项

```
allocsize=size        # 在执行延迟分配写入时设置缓冲的I/O文件结束预分配大小(默认大小为64 KiB)。此选项
attr2|noattr2
barrier
dmapi
grp|bsdgroups
nogrpid|sysvgroups
ihashsize=value
ikeep|noikeep
inode64
largeio|nolargeio
logbufs=value
logbsize=value
logdev=device
rtdev=device
mtpt=mountpoint
noalign
noatime
norecovery
nouuid
osynciosync
uquota,usrquota|uqnoenforce, quota # 启用用户磁盘配额记帐，并强制限制(可选)。
gquota, grpquota, gqnoenforce # 已启用磁盘配额记帐组，并强制执行限制(可选)。
pquota|prjquota|pqnoenforce # 启用项目磁盘配额会计并强制执行限制(可选)
sunit=value
swidth=value          # 用于指定RAID设备或条带卷的条带单元和宽度。值必须以512字节块单元指定.如果没有
swalloc              # 当扩展文件的当前结束且文件大小大于条带宽度大小时，数据分配将被舍入到条带宽度边
```

sh

xiafs选项

无

回环设备

另一种可能的类型是通过循环装置安装，例如指令“mount /tmp/fdimage /mnt -t vfat -o loop=/dev/loop3”，设置循环设备/dev/loop 3，使其与文件/tmp/fdigage相对应，然后在/mnt上挂载此设备。这种类型的挂载知道四个选项，即循环、偏移、大小消除和加密，它们实际上是losetup (8)的选项。

(除了特定于文件系统类型的选项之外，还可以使用这些选项。) 如果没有提到显式循环设备(但只提供了一个选项’-o loop’)，那么挂载将尝试找到一些未使用的循环设备并使用它。

由于Linux2.6.25支持自动销毁循环设备，因此由挂载分配的任何循环设备都将由umount在/etc/mtab上独立释放。您也可以使用’loset-d’或’umount-d’手动释放循环设备。

tmpfs选项

```
size=nbytes          # 覆盖文件系统的默认最大大小。大小以字节为单位，并舍入到整页。默认值是内存的一半。  
nr_blocks=          # 与size相同，但在PAGE_CACHE_SIZE的块中  
nr_inodes=          # 此实例的最大节点数。默认为物理RAM页数的一半，或(在具有高内存的计算机上)低内存页  
mode=               # 设置根目录的初始权限  
uid=                # 用户id  
gid=                # 组id  
mpol=[default|prefer:Node|bind:NodeList|interleave|interleave:NodeList] # 为该实例中的所有文件设置N  
                      # - default, 更倾向于从本地节点分配内存。  
                      # - prefer: Node, 更愿意从给定的节点分配内存。  
                      # - bind: NodeList, 仅从NodeList中的节点分配内存。  
                      # - interleave, 依次从每个节点分配。  
                      # - interleave: NodeList, 依次从NodeList的每个节点分配。  
                      # - NodeList格式是以逗号分隔的小数和范围列表，范围为两个连字符分隔的十进制数，是
```

说明

mount的标准格式是“mount -t type device dir”。这告诉内核将在设备(type类型)上找到的文件系统附加到目录dir上。以前的内容(如果有的话)以及dir的所有者和模式变得不可见，只要这个文件系统仍然挂载，路径名dir就会引用设备上的文件系统的根。

- 三种形式的调用实际上不会挂载任何东西：

```
mount -h          # 显示帮助信息  
mount -V          # 显示命令版本信息  
mount [-l] [-t type] # 列出所有挂载的文件系统，选项“-l”在这个清单中添加标签
```

- 设备指示

大多数设备都由文件名(块特殊设备的文件名)表示，如“/dev/sda1”，但还有其他可能性。例如，在NFS安装的情况下，设备看起来可能类似于“knuth.cwi.nl:/dir”。可以使用它的卷标签或UUID来指示块特殊设备(请参阅下面的-L和-U选项)

建议的设置是使用LABEL= 或UUID= 标记，而不是使用udev符号链接“/etc/disk/by-{LABEL, UUID}”。这些标签更具可读性、鲁棒性和可移植性。mount(8)命令在内部使用udev符号链接，因此使用“/etc/fstab”中的符号链接并不比Label=或者UUID=占优势。

proc文件系统不与特定设备相关联，在挂载它时，可以使用任意关键字(如proc)代替设备规范。

- /etc/fstab, /etc/mtab, /proc/mounts

文件/etc/fstab(参见fstab(5)可能包含描述哪些设备通常安装在何处，使用哪些选项安装的行。命令“mount -a [-t type] [-O optlist]” 将按所示的方式(具有或没有正确的选项)安装fstab中提到的所有文件系统，但行中包含“noauto”关键字的文件系统除外。添加-F选项将使用挂载子进程，以便文件系统同时挂载。

当安装fstat或mtab中提到的文件系统时，只给设备或仅给挂载点就足够了。mount和umount程序维护文件"/etc/mtab"中当前挂载的文件系统列表。如果没有给出挂载参数，则打印此列表。

如果指定了设备(或LABEL/UUID)和dir，则挂载程序不会读取"/etc/fstab"文件。例如"mount /dev/foo /dir"。如果您想要覆盖"/etc/fstab"中的挂载选项，则必须使用"mount device|dir -o"，然后，命令行中的挂载选项将被追加到"/etc/fstab"的选项列表中。通常的做法是，如果有更多重复的选项，最后一个选项就会获胜。

当安装proc文件系统(例如/proc)时，文件"/etc/mtab"和"/proc/mount"具有非常相似的内容。前者有更多的信息，例如所使用的挂载选项，但不一定是最新的(参见下面的-n选项)。可以将"/etc/mtab"替换为指向"/proc/mount"的符号链接，特别是当您有大量的挂载时，使用该符号链接的速度会快得多，但有些信息会以这种方式丢失，特别是使用"user"选项会失败

- 非超级用户挂载

通常，只有超级用户才能挂载文件系统。然而，当fstab文件中的行中包含"让任何人都可以挂载系统"的这种内容之后，例如"/dev/cdrom /cd iso9660 ro,user,noauto,unhide"，任何人都可以使用以下命令挂载在CDROM中找到的iso9660系统："mount /dev/cdrom"或者"mount /cd"

只有挂载文件系统的用户才能卸载它。如果任何用户应该能够卸载，那么在fstab行中使用users而不是user。owner选项类似于user选项，其限制是用户必须是特殊文件的所有者。这可能是有用的，例如对于"/dev/fd"如果登录脚本使控制台用户拥有此设备。

- 绑定挂载

从Linux2.4.0开始，可以将文件层次结构的一部分重新装入其他地方。使用命令"mount --bind olddir newdir"或者"mount -B olddir newdir"或者是在fstab中使用"/olddir /newdir none bind"。调用这个命令之后之后，可以在两个地方访问相同的内容，还可以重新装入单个文件。

此调用仅关系到(部分)单个文件系统，而不附加可能的子挂载。整个文件层次结构(包括子挂载)将使用："mount --rbind olddir newdir"或者"mount -R olddir newdir"

请注意，文件系统挂载选项将与原始挂载点上的选项相同，不能通过传递"-o"选项和"-bind/-rbind"来更改。挂载选项可以通过单独的重新装入命令进行更改，例如："mount --bind olddir newdir"，"mount -o remount,ro newdir"。

- 移动操作

从Linux2.5.1开始，可以原子地将挂载的树移动到另一个地方。使用如下命令："mount --move olddir newdir"或者"mount -M olddir newdir"。这将导致在newdir下可以访问以前出现在olddir下的内容。

- 共享子树操作

由于Linux2.6.15可以将挂载及其子挂载标记为共享、私有、从或不可绑定的。共享挂载提供了创建该挂载的镜像的能力，以便在任何镜像中挂载和汇总到另一个镜像。从装入接收来自其主的传播，反之亦然。私有挂载不具有传播能力。不可绑定的挂载是不能通过绑定操作克隆的私有挂载。详细的语义记录在内核源代码树中的文档/sharedsubtree.txt文件中。

```
mount --make-shared mountpoint
mount --make-slave mountpoint
mount --make-private mountpoint
mount --make-unbindable mountpoint
```

sh

以下命令允许递归更改给定挂载点下所有挂载的类型

```
mount --make-rshared mountpoint
mount --make-rslave mountpoint
mount --make-rprivate mountpoint
mount --make-runbindable mountpoint
```

sh

返回值

- 0，成功。
- 1，不正确的唤醒或者权限。
- 2，系统错误（内存溢出，不能fork，没有回环设备）。
- 4，内部mount出现bug。
- 8，用户中断。
- 16，在写或者锁定“/etc/mtab”时出错。
- 32，挂载失败。
- 64，一些挂载成功。

外部安装

外部安装帮助程序的语法是“/sbin/mount. spec dir [-sfnv] [-o options] [-t type.subtype]”，是文件系统类型，“-sfnvo”选项的含义与标准挂载选项相同。“-t”选项用于支持子类型的文件系统(例如/sbin/mount.fuse.sshfs)。

文件

- /etc/fstab，系统列表。
- /etc/mtab，已经挂载的系统。
- /etc/mtab~，锁定的文件。
- /etc/mtab.tmp，临时文件。
- /etc/filesystems，尝试的文件系统列表。

举例

创建ext2文件系统，并将其挂载到/media/sf_data

```
[root@localhost ~]$ mknod /dev/sdb4 b 1 1 #创建一个设备
[root@localhost ~]$ mke2fs /dev/sdb4 #创建文件系统，没有指定类型，默认是ext2
[root@localhost ~]$ mount /dev/sdb4 /media/disk #将文件系统挂载
[root@localhost ~]$ mount #不适用任何选项，输出已经挂载的文件系统
/dev/mapper/VolGroup-lv_root on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs (rw)
/dev/sda1 on /boot type ext4 (rw)
/dev/sdb4 on /media/sf_data type ext2 (rw)
```

sh

挂载iso文件

```
sh

[root@localhost ~]$ mount -t iso9660 -o loop /sogrey/my.iso /media/sf_data/ #挂载iso文件，使用-o参数指定挂载类型
[root@localhost ~]$ mount  #查看已经挂载的文件系统
/dev/mapper/VolGroup-lv_root on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
/sogrey/my.iso on /media/sf_data type iso9660 (rw,loop=/dev/loop0)
```

mpstat - 显示各个可用CPU的状态

mpstat命令 指令主要用于多CPU环境下，它显示各个可用CPU的状态系你想。这些信息存放在/proc/stat文件中。在多CPUs系统里，其不但能查看所有CPU的平均状况信息，而且能够查看特定CPU的信息。

mpstat指令用来显示cpu的使用状况，将内容显示到标准输出。处理器0是第一个。还报告了所有处理器之间的全局平均活动。mpstat命令既可以在SMP机器上使用，也可以在UP机器上使用，但是在后者中，只会打印全局平均活动。如果未选择活动，则默认报告是CPU利用率报告。

Interval参数指定每个报表之间以秒为单位的时间量。值为0(或根本没有参数)表示自系统启动(启动)以来将报告处理器统计信息。如果未将count参数设置为零，则可以与Interval参数一起指定Count参数。计数值决定间隔秒生成的报表数。如果未使用count参数指定Interval参数，则mpstat命令将连续生成报告。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mpstat [ -A ] [ -I { SUM | CPU | ALL } ] [ -u ] [ -P { cpu [...] | ON | ALL} ] [ -V ] [ sh ]
```

选项

```
-A # 等价于“-I ALL -u -P ALL”  
-I {SUM | CPU | ALL} # 显示中断信息: SUM, 显示每个cpu的中断次数; CPU, 显示每秒收到的中断次数; ALL,  
-P {cpu [...] | ON | ALL} # 指定CPU编号, 从0开始。ON代表, 每行显示一个CPU信息。ALL代表显示所有的cpu信息  
-u # 显示cpu的使用状态:  
-V # 显示版本信息并且推出
```

cpu状态

“-u”选项可以显示的cpu状态有

```
CPU          # 处理器号码。关键字ALL表示统计数据是以所有处理器之间的平均值计算的。  
%usr         # 显示在用户级别(应用程序)执行时出现的CPU利用率百分比。  
%nice        # 以良好的优先级在用户级别执行时显示CPU利用率的百分比。  
%sys         # 显示在系统级(内核)执行时CPU利用率的百分比。请注意，这不包括用于服务硬件和软件  
%iowait      # 显示CPU或CPU空闲的时间百分比，在此期间，系统有未执行的磁盘I/O请求。  
%irq         # 显示cpu或cpu用于服务硬件中断的时间百分比。  
%soft        # 显示CPU或CPU用于服务软件中断的时间百分比。  
%steal       # 显示虚拟机管理程序为另一个虚拟处理器服务时，虚拟CPU或CPU在非自愿等待中花费的  
%guest       # 显示CPU或cpu运行虚拟处理器所花费的时间百分比。  
%idle        # 显示CPU或CPU空闲的时间百分比，并且系统没有未执行的磁盘I/O请求。
```

举例

显示cpu使用情况

```
[sogrey@bogon ~]$ mpstat -u  
Linux 3.10.0-862.14.1.0.h209.eulerosv2r7.x86_64 (bogon) 2021年06月26日 _x86_64_ (1 CPU)  
  
01时00分11秒  CPU    %usr    %nice    %sys %iowait    %irq    %soft    %steal    %guest    %gnice    %idle  
01时00分11秒  all    51.71    0.05   14.88     0.06    0.00    0.16    0.00    0.00    0.00    33.14  
[sogrey@bogon ~]$
```

mput - 登录ftp服务器之后将文件上传到服务器

使用lftp登录ftp服务器之后，可以使用put指令将文件上传到服务器。mput指令可以使用通配符，而put指令则不可以。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux		Arch Linux

语法

```
mput [-c] [-d] [-a] [-E] [-O base] files
```

sh

选项

-d	# 穿件与文件名字一样的目录，存放文件
-c	# 如果失败，持续获取
-E	# 获取之后，删除源文件
-a	# 使用ascii模式
-O	# 指定输出文件存放的目录

sh

举例

上传文件

```
[root@localhost ~]$ lftp 192.168.1.8      #登录服务器
lftp 192.168.1.8:~> cd pub                  #切换工作目录
lftp 192.168.1.8:/pub> mput *.c            #上传所有c文件
mput: Access failed: 553 Could not create file. (3.c)
155 bytes transferred
Transfer of 1 of 4 files failed
lftp 192.168.1.8:/pub> ls                  #查看内容，以上传成功
-rwxrwxrwx    1 0      0      2375494044 Aug 14 06:38 1.zip
-rw-----    1 14     50      0 Oct 02 01:52 11.c
-rw-r--r--    1 0      0      0 Oct 02 01:19 11c
-rw-r--r--    1 0      0      0 Oct 02 01:19 22c
-rw-----    1 14     50      65 Oct 02 01:48 3.c
-rw-----    1 14     50      52 Oct 02 01:52 4.c
-rw-----    1 14     50      38 Oct 02 01:52 5.c
drwxr-xr-x    2 0      0      4096 Oct 02 01:12 testftp
```

sh

mv - 用来对文件或目录重新命名

将文件或者目录移动到另一个地方，或者重命名。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
mv [选项] [-T] src dst
mv [选项] src directory
mv [选项] -t directory src
```

sh

选项

```
--backup=[control]          # 为每一个存在的文件创建备份
-b                          # 和“--backup”一样，但是没有参数
-f, --force                  # 强制移动
-i, --interactive            # 使用交互的方式移动
-n, --no-clobber             # 不覆盖已经存在的文件。如果同时制定了“-i, -n, -f”中的多个选项，那么只有最近的一个起作用。
--strip-trailing-slashes    # 从每个源参数中移除任何尾随斜线。
-S, --suffix=SUFFIX          # 重写通常的备份后缀
-t, --target-directory       # 将所有源文件移动到目标文件夹
-T, --no-target-directory    # 将目标视为正常文件
-u, --update                  # 当目的文件不存在，或者源文件比目的文件新的时候才移动
-v, --verbose                 # 显示详细执行过程

--help                       # 显示帮助文档
--version                     # 显示命令版本信息
```

sh

备份后缀为‘~’，除非设置为“--”后缀或者\$SIMPLE_BACKUP_SUFFIX。版本控制方法可以通过“--backup”的选项或通过VERSION_CONTROL环境变量来选择。以下是这些值：

- none, off: 从不备份，即使给出了“--backup”选项。
- numbered, t: 创建编号备份。
- existing, nil: 如果有编号备份，则为编号，否则为简单。
- simple, over: 总是创建简单备份。

举例

将目录/usr/men中的所有文件移到当前目录（用.表示）中：

```
mv /usr/men/* .
```

sh

移动文件

```
mv file_1.txt /home/office/
```

sh

移动多个文件

```
mv file_2.txt file_3.txt file_4.txt /home/office/
mv *.txt /home/office/
```

sh

移动目录

```
mv directory_1/ /home/office/
```

sh

重命名文件或目录

```
mv file_1.txt file_2.txt # 将文件file_1.txt改名为file_2.txt
```

sh

重命名目录

```
mv directory_1/ directory_2/
```

sh

打印移动信息

```
mv -v *.txt /home/office
```

sh

提示是否覆盖文件

```
mv -i file_1.txt /home/office
```

sh

源文件比目标文件新时才执行更新

```
mv -uv *.txt /home/office
```

sh

不要覆盖任何已存在的文件

```
mv -vn *.txt /home/office
```

sh

复制时创建备份

```
mv -bv *.txt /home/office
```

sh

无条件覆盖已经存在的文件

```
mv -f *.txt /home/office
```

sh

在同一个目录下移动，即重命名。

```
[sogrey@bogon DirTest]$ ls  
Dir1  Dir2  
[sogrey@bogon DirTest]$ mv Dir1 DirTmp  
[sogrey@bogon DirTest]$ ls  
Dir2  DirTmp  
[sogrey@bogon DirTest]$
```

sh

移动到其他地方

```
[sogrey@bogon DirTest]$ ls  
Dir2  DirTmp  
[sogrey@bogon DirTest]$ mv DirTmp Dir2/Dir1 #DirTmp移动到Dir2目录下并重命名为Dir1  
[sogrey@bogon DirTest]$ ls  
Dir2  
[sogrey@bogon DirTest]$ cd Dir2  
[sogrey@bogon Dir2]$ ls  
Dir1  run.sh
```

sh

mysql - 一个简单的sql shell

mysql是一个简单的sql shell，它可以用来管理mysql数据库。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mysql [options] db
```

sh

选项

```
-? , --help          # 显示帮助信息
--auto-rehash       # 激活自动rehash功能
--bind-address=ip   # 绑定ip, 当电脑有多个网卡的时候, 可以指定mysql连接时的网卡
--character-sets-dir = path # 指定字符集所在的目录
--column-names      # 在结果中显示列名
-C, --comments     # 在发送给服务器的状态中显示注释
-c, --compress      # 在服务器和客户端之间的数据进行压缩
-D db, --database=db # 指定数据库名
--default-character-set=charset # 默认的字符集
-e statement, --execute=statement # 指定要使用的指令
-f, --force          # 强制执行
-H, --html            # 输出html格式
-i, --ignore-spaces  # 忽略空格
--line-numbers       # 为错误信息显示行号
--local-infile=0|1    # 关闭或者开启LOAD DATA INFILE功能
-A, --no-autp-rehash # 关闭自动rehash功能
-b, --no-beep         # 关闭出错提醒
-p password, --password= # 连接数据库使用的密码
-W, --pipe             # 使用有名管道连接数据库
-P port, --port=        # 连接数据库使用的端口
--protocol=TCP|SOCKET|PIPI|MEMORY # 连接数据库使用的协议
-q, --quick           # 不缓存查询结果
-s, --silent          # 输出简短的内容
-v, --verbose         # 显示详细执行过程
-V, --version         # 显示版本信息
-w, --wait             # 等待时间
-X, --xml              # 产生xml输出
-u user, --user=        # 连接数据库的用户名, 默认是rootq
```

sh

举例

登录

```
[root@localhost ~]$ mysql -u root -p #使用用户root连接数据库
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.1.71 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> quit      #退出
Bye
```

查看权限

```
mysql> show privileges;
+-----+-----+-----+
| Privilege      | Context          | Comment           |
+-----+-----+-----+
| Alter          | Tables           | To alter the table |
| Alter routine  | Functions,Procedures | To alter or drop stored functions |
| Create         | Databases,Tables,Indexes | To create new databases and tables |
| Create routine | Databases        | To use CREATE FUNCTION/PROCEDURE |
| Create temporary tables | Databases | To use CREATE TEMPORARY TABLE |
| Create view    | Tables           | To create new views |
| Create user   | Server Admin     | To create new users |
| Delete         | Tables           | To delete existing rows |
| Drop           | Databases,Tables | To drop databases, tables, and indexes |
| Event          | Server Admin     | To create, alter, drop and enable events |
| Execute        | Functions,Procedures | To execute stored routines |
| File           | File access on server | To read and write files on the file system |
| Grant option   | Databases,Tables,Functions,Procedures | To give to other users those privileges |
| Index          | Tables           | To create or drop indexes |
| Insert          | Tables           | To insert data into tables |
| Lock tables    | Databases        | To use LOCK TABLES (together with GRANT) |
| Process         | Server Admin     | To view the plain text of current process |
| References     | Databases,Tables | To have references on tables |
| Reload          | Server Admin     | To reload or refresh tables, triggers, and views |
| Replication client | Server Admin | To ask where the slave or master is |
| Replication slave | Server Admin     | To read binary log events from the replication slave |
| Select          | Tables           | To retrieve rows from table |
| Show databases  | Server Admin     | To see all databases with SHOW DATABASES |
| Show view       | Tables           | To see views with SHOW CREATE VIEW |
| Shutdown        | Server Admin     | To shut down the server |
| Super           | Server Admin     | To use KILL thread, SET GLOBAL, and other superuser privileges |
| Trigger          | Tables           | To use triggers |
| Update          | Tables           | To update existing rows |
| Usage            | Server Admin     | No privileges - allow connection |
+-----+-----+-----+
29 rows in set (0.00 sec)
```


mysqladmin - mysql数据库的管理工具

mysqladmin是mysql数据库的管理工具，可以控制、查看、修改数据库服务器的配置和状态。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mysqladmin [options] command [command-options] [command [command-options]]
```

sh

选项

```
-? , --help          # 显示帮助信息
--bind-address=ip    # 绑定ip, 当电脑有多个网卡的时候, 可以指定mysql连接时的网卡
--character-sets-dir = path # 指定字符集所在的目录
-c, --compress       # 在服务器和客户端之间的数据进行压缩
--default-character-set=charset # 默认的字符集
-f, --force          # 强制执行
-b, --no-beep         # 关闭出错提醒
-W, --pipe            # 使用有名管道连接数据库
-P port, --port=      # 连接数据库使用的端口
--protocol=TCP|SOCKET|PIPE|MEMORY # 连接数据库使用的协议
-s, --silent          # 输出简短的内容
-v, --verbose         # 显示详细执行过程
-V, --version         # 显示版本信息
-w, --wait             # 等待时间
-u                   # 指定用户名
-p                   # 指定密码
```

sh

命令

mysqladmin支持的命令如下

sh

```

create          # 创建数据库
debug           # 开启调试模式
drop            # 删除数据库
extend-status   # 显示mysql扩展状态信息
flush-hosts    # 刷新mysql缓冲的主机
flush-logs     # 刷新日志
flush-status   # 刷新状态变量
flush-privileges # 刷新权限
flush-table    # 刷新所有表格
flush-threads  # 刷新线程
kill            # 杀死指定的线程
password        # 修改密码
ping            # 测试服务器是否可连接
processlist    # 显示活动的线程
reload          # 重新加载授权表
refresh         # 刷新所有的表，并且关闭和打开日志文件
shutdown        # 关闭服务器
status          # 显示服务器状态
start-slave    # 启动slave
stop-slave     # 关闭slave
variables       # 显示可用的变量和值
version         # 显示版本信息

```

举例

显示服务器状态

```

[root@localhost ~]$ mysqladmin -u root -p status
Enter password:
Uptime: 1501  Threads: 1  Questions: 32  Slow queries: 0  Opens: 15  Flush tables: 1  Open tables:

```

查看活动线程

```

[root@localhost ~]$ mysqladmin -u root -p processlist
Enter password:
+-----+-----+-----+-----+-----+-----+
| Id, User, Host      , db, Command, Time, State, Info          ,
+-----+-----+-----+-----+-----+-----+
| 7 , root, localhost,  , Query , 0 ,      , show processlist,
+-----+-----+-----+-----+-----+-----+

```

显示服务器版本信息

```
[root@localhost ~]$ mysqladmin -u root -p version
Enter password:
mysqladmin Ver 8.42 Distrib 5.1.71, for redhat-linux-gnu on i386
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Server version      5.1.71
Protocol version   10
Connection          Localhost via UNIX socket
UNIX socket        /var/lib/mysql/mysql.sock
Uptime:              26 min 13 sec

Threads: 1  Questions: 33  Slow queries: 0  Opens: 15  Flush tables: 1  Open tables: 8  Queries
```

mysqldump - 一个mysql客户端的备份程序

mysqldump是一个客户端的备份程序，他可以备份数据库，或者将数据库传输到另外一个服务器。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
mysqldump [options] [db_name [tbl_name ...]]
```

sh

选项

```
-? , --help          # 显示帮助信息
--add-drop-database # 在创建数据库之前，增加删除数据库语句
--add-drop-table    # 在创建表之前，增加删除表语句
--add-locks         # 在输出insert语句的时候增加锁表语句
-A, --all-database # 备份所有数据库的所有表
--allow-keywords   # 允许列名使用关键字
--bind-address=ip   # 绑定ip
--character-set-dir # 默认的字符集目录
-i, --comments      # 在备份的时候添加注释
-c, --complete-insert # 使用完整的insert语句
-C, --compress       # 在服务器和客户端之间使用压缩语句
-B, --database        # 指定要备份的数据库
--dump-date          # 如果使用了--comments选项，那么就可以追加日期
-F, --flush-logs     # 刷新日志
--flush-privileges  # 刷新权限
-f, --force           # 强制执行
--ignore-table        # 备份的时候忽略表
--lock-all-tables | -x # 备份的时候锁定所有的数据库
-l, --lock-tables    # 锁定指定的表
--log-error          # 错误日志
-t, --no-create-db   # 备份数据库的时候，不输出创建表语句
-d, --no-data         # 备份数据库的时候，只备份数据结构，不备份数据
-P, --port            # 指定端口
--protocol           # 指定协议
-q, --quick           # 静默模式
-v, --verbose         # 显示详细过程
-V, --version         # 显示版本信息
-u, --user             # 指定用户
-p, --password        # 指定密码
```

sh

举例

备份指定的数据库中的指定表

```
[root@localhost ~]$ mysqldump -v -u root -p wordpress wp_links #默认情况下，备份内容到标准输出
Enter password:
-- Connecting to localhost...
-- MySQL dump 10.13 Distrib 5.1.71, for redhat-linux-gnu (i386)
--
-- Host: localhost      Database: wordpress
-- -----
-- Server version 5.1.71

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
-- Retrieving table structure for table wp_links...

--
-- Table structure for table `wp_links`
--

DROP TABLE IF EXISTS `wp_links`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `wp_links` (
  `link_id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
  `link_url` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_name` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_image` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_target` varchar(25) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_description` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_visible` varchar(20) COLLATE utf8_unicode_ci NOT NULL DEFAULT 'Y',
  `link_owner` bigint(20) unsigned NOT NULL DEFAULT '1',
  `link_rating` int(11) NOT NULL DEFAULT '0',
  `link_updated` datetime NOT NULL DEFAULT '0000-00-00 00:00:00',
  `link_rel` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_notes` mediumtext COLLATE utf8_unicode_ci NOT NULL,
  `link_rss` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  PRIMARY KEY (`link_id`),
  KEY `link_visible` (`link_visible`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
-- Sending SELECT query...

--
-- Dumping data for table `wp_links`
--

-- Retrieving rows...
```

```

LOCK TABLES `wp_links` WRITE;
/*!40000 ALTER TABLE `wp_links` DISABLE KEYS */;
/*!40000 ALTER TABLE `wp_links` ENABLE KEYS */;
UNLOCK TABLES;
-- Disconnecting from localhost...
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2018-10-05 20:34:17
You have new mail in /var/spool/mail/root

```

查看活动线程

```

[root@localhost ~]$ mysqldump -v -u root -p wordpress wp_links > sql.bak #使用重定向功能，备份到sql.bak
sh
Enter password:
-- Connecting to localhost...
-- Retrieving table structure for table wp_links...
-- Sending SELECT query...
-- Retrieving rows...
-- Disconnecting from localhost...
You have new mail in /var/spool/mail/root
[root@localhost ~]$ cat sql.bak    #查看备份内容
-- MySQL dump 10.13 Distrib 5.1.71, for redhat-linux-gnu (i386)
--
-- Host: localhost    Database: wordpress
-- -----
-- Server version 5.1.71

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `wp_links`
--

DROP TABLE IF EXISTS `wp_links`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `wp_links` (
  `link_id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
  `link_url` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_name` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_description` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_visible` tinyint(1) NOT NULL DEFAULT '1',
  `link_status` tinyint(1) NOT NULL DEFAULT '1',
  `link_type` tinyint(1) NOT NULL DEFAULT '0',
  `link_rss` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_rel` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `link_l�

```

```
`link_name` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
`link_image` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
`link_target` varchar(25) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
`link_description` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
`link_visible` varchar(20) COLLATE utf8_unicode_ci NOT NULL DEFAULT 'Y',
`link_owner` bigint(20) unsigned NOT NULL DEFAULT '1',
`link_rating` int(11) NOT NULL DEFAULT '0',
`link_updated` datetime NOT NULL DEFAULT '0000-00-00 00:00:00',
`link_rel` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
`link_notes` mediumtext COLLATE utf8_unicode_ci NOT NULL,
`link_rss` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
PRIMARY KEY (`link_id`),
KEY `link_visible` (`link_visible`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

-- 
-- Dumping data for table `wp_links`
-- 

LOCK TABLES `wp_links` WRITE;
/*!40000 ALTER TABLE `wp_links` DISABLE KEYS */;
/*!40000 ALTER TABLE `wp_links` ENABLE KEYS */;
UNLOCK TABLES;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2018-10-05 20:36:27
```

mysqlimport - 可以用来将文本文件中的数据导入到数据库

mysqlimport指令可以用来将文本文件中的数据导入到数据库。在导入文本文件的时候，必须确保数据库中有一张表，而且他的名字和文本文件的名字是一样的。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mysqlimport [options] dbname textfile1 ...
```

sh

选项

```
-? , --help          # 显示帮助信息
--bind-address=ip   # 绑定ip
--character-sets-dir # 默认的字符集目录
-i, --ignore        # 与replace选项相同
-c, --columns       # 设置字段列表
-C, --compress      # 在服务器和客户端之间使用压缩语句
-D, --delete        # 导入数据之前，删除数据库表
-f, --force         # 强制执行
-ignore-lines       # 忽略前n行
-L, --local          # 从客户端主机读取内容
-l, --local-tables  # 执行写操作之前，锁定表
-P, --port           # 指定端口
--protocol          # 指定协议
-r, --replace        # 如果有相同的行，那么久覆盖
-v, --verbose        # 显示详细过程
-V, --version        # 显示版本信息
-u, --users          # 指定用户
-p, --password       # 指定密码
```

sh

举例

导入数据库文件

```
[root@localhost ~]$ cat wj.txt          #查看文本内容
1           zhangsan
2           lisi
3           wangwu
4           zhangliu

[root@localhost ~]$ mysqlimport -v -u root -p test wj.txt #导入文本到数据库。前提是必须有一个表，而且
Enter password:
Connecting to localhost
Selecting database test
Loading data from SERVER file: wj.txt into wj
test.wj: Records: 4 Deleted: 0 Skipped: 0 Warnings: 0
Disconnecting from localhost

[root@localhost ~]$ mysql -u root -p -e 'select * from wj' test #查看数据库中表wj的内容
Enter password:
+---+-----+
| id | name   |
+---+-----+
| 1  | zhangsan |
| 2  | lisi    |
| 3  | wangwu  |
| 4  | zhangliu |
+---+-----+
```

mysqlshow - 一个mysql客户端显示数据库的信息、表信息、字段信息备份程序

mysqlshow是一个客户端的程序，它可以显示数据库的信息、表信息、字段信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
mysqlshow [options] [db_name [tbl_name [col_name]]]
```

sh

选项

```
-? , --help          # 显示帮助信息
--bind-address=ip   # 绑定ip
--character-sets-dir # 默认的字符集目录
-i, --status        # 显示表格的额外信息
-count             # 显示表中的行数
-C, --compress      # 在服务器和客户端之间使用压缩语句
-P, --port          # 指定端口
--protocol         # 指定协议
-v, --verbose       # 显示详细过程
-V, --version       # 显示版本信息
-u, --user          # 指定用户
-p, --password      # 指定密码
```

sh

举例

查看有哪些数据库

```
[root@localhost ~]$ mysqlshow -u root -p      #显示有哪些数据库
Enter password:
+-----+
|   Databases   |
+-----+
| information_schema |
| david           |
| discuz          |
| drupal          |
| mediawiki       |
| mysql           |
| phpmyvisites   |
| test            |
| test01          |
| test02          |
| wordpress       |
+-----+
```

sh

查看数据库中有哪些表

```
[root@localhost ~]$ mysqlshow -u root -p test    #显示数据库test的信息
Enter password:
Database: test
+-----+
| Tables |
+-----+
| wj     |
+-----+
```

sh

查看表中有哪些字段

```
[root@localhost ~]$ mysqlshow -u root -p test wj  #显示数据库test中的表wj的信息
Enter password:
Database: test  Table: wj
+-----+-----+-----+-----+-----+-----+-----+
| Field | Type   | Collation        | Null | Key  | Default | Extra | Privileges
+-----+-----+-----+-----+-----+-----+-----+
| id   | int(11) |                 | NO   |       |         |       | select,insert,update,refer
| name | text    | latin1_swedish_ci | NO   |       |         |       | select,insert,update,refer
+-----+-----+-----+-----+-----+-----+-----+
```

sh

netstat - 查看Linux中网络系统状态信息

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。

netstat指令可以显示当前的网络连接、路由表、接口统计信息、伪装连接和多播成员资格等信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
netstat [address_family_options] [--tcp|-t] [--udp|-u] [--raw|-w] [--listening|-l] [--all|-a]
netstat {--route|-r} [address_family_options] [--extend|-e[--extend|-e]] [--verbose|-v] [--version|-V]
netstat {--interfaces|-I|-i} [iface] [--all|-a] [--extend|-e] [--verbose|-v] [--program|-p]
netstat {--groups|-g} [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports]
netstat {--masquerade|-M} [--extend|-e] [--numeric|-n] [--numeric-hosts] [--numeric-ports]
netstat {--statistics|-s} [--tcp|-t] [--udp|-u] [--raw|-w] [delay]
netstat {--version|-V}
netstat {--help|-h}
```

```
address_family_options:
[--protocol={inet,inet6,unix,ipx,ax25,netrom,ddp, ... }] [--unix|-x] [--inet|--ip] [--ax25]
```

选项

```

-v, --verbose          # 显示执行过程
-n, --numeric          # 直接显示数字ip
--numeric-hosts       # 显示主机的数字地址, 不影响端口和用户名
--numeric-ports       # 显示端口, 不影响主机和用户名
--numeric-users        # 显示用户id, 不影响主机和端口
-A, --protocol        # 指定网络类型
-c, --continuous      # 持续显示
-e, --extend           # 显示其他附加信息
-o, --timers           # 显示计时器
-p, --program          # 显示正在使用的socket程序pid和名字
-l, --listen            # 只显示监听的socket信息
-a, -all               # 显示所有连接中的socket信息
-F                   # 显示FIB
-C                   # 显示路由的缓存
-Z, --context          # 如果打开了SELinux, 那么就打印SELinux的上下文
-T, --notrim           # 停止修剪长地址

--help                 # 显示帮助文档
--version              # 显示命令版本信息

```

说明

netstat打印有关Linux网络子系统的信息。打印的信息类型由第一个参数控制

参数	说明
(none)	默认情况下, netstat显示打开的套接字列表。如果不指定任何地址族, 则将打印所有已配置地址家族的活动套接字。
-r --route	打印内核路由表
-g --groups	显示IPv4和IPv6的多播组成员信息
-i --interfaces =iface , -l =iface	显示所有网络接口的表, 或指定的ifaces。
-M --masquerade	显示假连接列表。
-s --statistics	显示每个协议的汇总统计信息。

输出

Active Internet connections (TCP, UDP, raw)		说明
Proto	socket使用的协议, tcp, udp, raw。	
Recv-Q	连接到此套接字的用户程序未复制的字节数。	
Send-Q	远程主机未确认的字节数。	
Local Address	套接字的本地端的地址和端口号。除非指定了"--numeric (-n)"选项, 否则套接字地址将解析为其规范主机名(FQDN), 端口号将被转换为相应的服务名称。	
Foreign Address	套接字的远程端的地址和端口号。类似于"本地地址"。	
State	socket的状态。由于在原始模式中没有状态, 通常在UDP中也没有使用状态, 因此这一列可以保留为空白。通常, 这可以是几个值之一: ESTABLISHED, 套接字有一个已建立的连接。SYN_SENT, 套接字正在积极尝试建立连接。SYN_RECV, 已从网络接收到连接请求。FIN_WAIT1, 套接字关闭, 连接正在关闭。FIN_WAIT2, 连接被关闭, 套接字正在等待来自远程端的关闭。TIME_WAIT, 套接字在关闭后等待处理仍在网络中的数据包。CLOSED, 没有使用套接字。CLOSE_WAIT, 远程终端已关闭, 等待套接字关闭。LAST_ACK, 远程终端已关闭, 套接字已关闭。等待确认LISTEN, 套接字正在监听传入的连接。除非您指定"--listening"或"--all(-a)"选项, 否则输出中不包含此类套接字。CLOSING, 两个套接字都已关闭, 但我们仍然没有发送所有数据。UNKNOWN, 套接字的状态未知。	
User		
PID/Program name		
Timer		

Active UNIX domain Sockets		说明
Proto	套接字使用的协议(通常是Unix)。	
RefCnt	参考计数(即通过这个套接字附加的进程)。	
Flags	所显示的标志是SO_ACCEPTON (显示为ACC)、SO_WAITDATA(W)或SO_NOSPACE(N)。如果未连接套接字的相应进程正在等待连接请求, 则在未连接套接字上使用SO_ACCEPTON。其他的标志不正常。	
Type	可能的几种值: SOCK_DGRAM, 套接字以数据报(无连接)模式使用。SOCK_STREAM, 这是一个流(连接)套接字。SOCK_RAW, 套接字用作原始套接字。SOCK_RDM, 这个服务提供可靠传递的消息。SOCK_SEQPACKET, 这是一个顺序的数据包套接字。SOCK_PACKET, 原始接口访问套接字。UNKNOWN, 不知道的状态	
State	FREE, 套接字还没有分配LISTENING, 套接字正在监听请求。CONNECTING, 套接字正在尝试连接。CONNECTED, 套接字已经连接。DISCONNECTING, 套接字断开连接。(empty), 套接字没有连接到其他地方。UNKNOWN, 未知的状态	
PID/Program name	打开套接字的进程ID(PID)和进程名。更多信息可在上面写的活动互联网连接部分获得。	
Path	这是连接到套接字上的相应进程的路径名。	

相关文件

- /etc/services, 服务翻译文件
- /proc, proc文件系统的挂载点, 它允许访问内核状态。
- /proc/net/dev, 设备信息。
- /proc/net/raw, raw套接字信息。
- /proc/net/tcp, tcp套接字信息。
- /proc/net/udp, udp套接字信息。
- /proc/net/igmp, IGMP多播信息。
- /proc/net/unix, Unix域名套接字信息。
- /proc/net/ipx, IPX套接字信息。
- /proc/net/ax25, AX25套接字信息。
- /proc/net/appletalk, DDP (appletalk) 套接字信息。
- /proc/net/nr, NET/ROM套接字信息
- /proc/net/route, IP路由信息。
- /proc/net/ax25_route, AX25路由信息。
- /proc/net/px_route, IPX路由信息。
- /proc/net/nr_nodes, NET/ROM 节点列表。
- /proc/net/nr_neigh, NET/ROM邻居。
- /proc/net/ip_masquerade, 伪装连接。
- /proc/net/snmp, 静态。

举例

显示指定类型的网络信息

```
[root@localhost ~]$ netstat -A inet      #指定网络类型inet
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
udp      0      0 192.168.0.113:33423    192.168.0.1:domain  ESTABLISHED
udp      0      0 192.168.0.113:40242    192.168.1.1:domain  ESTABLISHED
udp      0      0 192.168.0.113:50786    192.168.0.1:domain  ESTABLISHED
```

显示路由表

```
[root@localhost ~]$ netstat -r      #显示路由表
Kernel IP routing table
Destination     Gateway         Genmask        Flags   MSS Window irtt Iface
255.255.255.255 -           255.255.255.255  IH       - -       - -
192.168.1.0     *           255.255.255.0   U        0 0       0 eth0
224.0.0.0       -           255.255.255.0   I        - -       - -
default         192.168.1.1   0.0.0.0       UG       0 0       0 eth0
```

显示网卡状态

```
[root@localhost ~]$ netstat -i #显示网卡状态
Kernel Interface table
Iface      MTU Met     RX-OK RX-ERR RX-DRP RX-OVR     TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500   0    121817      0      0      0    79998      0      0      0 BMRU
lo       16436   0     8374      0      0      0     8374      0      0      0 LRU
```

sh

nfsstat - 列出NFS客户端和服务器的工作状态

nfsstat命令 用于列出NFS客户端和服务器的工作状态。

nfsstat指令用来显示nfs客户端和服务器的活动信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
nfsstat [选项]
```

sh

选项

```
-s, --server      # 只显示服务器信息, 默认还要显示客户端信息  
-c, --client      # 只显示客户端信息  
-n, --nfs       # 只显示nfs信息, 默认还要显示rpc信息  
-m, --mounts     # 显示被加载的nfs文件信息信息  
-r, --rpc        # 只显示rpc信息  
-l, --list       # 显示列表的nfs信息  
-s, --since      # 从文件显示nfs信息  
-o              # 显示自定义信息: nfs, 显示nfs协议信息; rpc, 显示rpc常规信息; net, 显示网络层状态; fh,  
-v, --verbose    # 等价“-o all”
```

sh

举例

要显示关于客户机发送和拒绝的RPC和NFS调用数目的信息，输入：

```
nfsstat -c
```

sh

要显示和打印与客户机NFS调用相关的信息，输入如下命令：

```
nfsstat -cn
```

sh

要显示和打印客户机和服务器的与RPC调用相关的信息，输入如下命令：

```
nfsstat -r
```

sh

要显示关于服务器接收和拒绝的RPC和NFS调用数目的信息，输入如下命令：

```
nfsstat -s
```

sh

显示客户端nfs信息

```
[root@localhost ~]$ nfsstat -cn
Client nfs v4:
  null      read      write      commit      open      open_conf
  0       0% 0       0% 0       0% 0       0% 1       0% 0       0% 0
  open_noat   open_dgrd    close     setattr     fsinfo      renew
  0       0% 0       0% 0       0% 0       0% 3       2% 20      18%
  setclntid   confirm     lock     lockt     locku      access
  1       0% 1       0% 0       0% 0       0% 0       0% 7       6%
  getattr     lookup   lookup_root   remove     rename      link
  58      52% 7       6% 1       0% 0       0% 0       0% 0       0% 0
  symlink     create   pathconf   statfs     readlink    readdir
  0       0% 0       0% 2       1% 0       0% 0       0% 4       3%
  server_caps  delegreturn  getacl    setacl     fs_locations  rel_lkowner
  5       4% 0       0% 0       0% 0       0% 0       0% 0       0%
  seinfo      exchange_id  create_ses  destroy_ses sequence  get_lease_t
  0       0% 0       0% 0       0% 0       0% 0       0% 0       0%
  reclaim_comp  layoutget  getdevinfo  layoutcommit layoutreturn  getdevlist
  0       0% 0       0% 0       0% 0       0% 0       0% 0       0%
  (null)
  0       0%
```

sh

显示已经挂载的系统信息

```
[root@localhost ~]$ nfsstat -m
/media/test from 192.168.1.8:/wj/
Flags:rw,relatime,vers=4,rsize=262144,wsize=262144,namlen=255,hard,proto=tcp,port=0,timeo=600,r
clientaddr=192.168.1.8,minorversion=0,local_lock=none,addr=192.168.1.8
[root@localhost ~]$
```

sh

nice - 改变程序执行的优先权等级

nice命令 用于以指定的进程调度优先级启动其他的程序。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
nice [OPTION] [参数]
```

选项

-n, --adjustment=N	# 指定进程的优先级（整数）。
--help	# 显示帮助文档
--version	# 显示命令版本

举例

新建一个进程并设置优先级，将当前目录下的documents目录打包，但不希望tar占用太多CPU：

```
sh
nice -19 tar zcf pack.tar.gz documents
```

方法非常简单，即在原命令前加上nice -19。很多人可能有疑问了，最低优先级不是19么？那是因为这个"-19"中的"-"仅表示参数前缀；所以，如果希望将当前目录下的documents目录打包，并且赋予tar进程最高的优先级：

```
sh
nice --19 tar zcf pack.tar.gz documents
```

设置指令wc优先级

```
sh
[root@localhost ~]$ nice -n 19 wc # 设置wc的优先级最低
^Z
[2]+  Stopped                  nice -n 19 wc
[root@localhost ~]$
```

查看进程优先级

sh

```
[root@localhost ~]$ ps -ao "%p%y%x%c%n" # 查看进程等级, wc的等级是19
 PID TTY      TIME COMMAND      NI
 8321 pts/0    00:00:00 wc        19
 8370 pts/0    00:00:00 ps        0
```

nisdomainname - 显示主机NIS的域名

ypdomainname命令 显示主机的NIS的域名。

nisdomainname指令显示由函数“getdomainname”返回的主机域名，使用这个指令也可以设置一个主机NIS/YP域名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
nisdomainname [ -v ]
```

sh

选项

-v	# 显示详细执行过程
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

显示主机域名

```
[root@localhost ~]$ nisdomainname          #显示域名  
www.david.com  
[root@localhost ~]$ nisdomainname www.weijie.com    #设置域名  
[root@localhost ~]$ nisdomainname          #显示域名  
www.weijie.com
```

sh

nohup - 将程序以忽略挂起信号的方式运行起来

nohup命令可以将程序以忽略挂起信号的方式运行起来，被运行的程序的输出信息将不会显示到终端。

无论是否将 nohup 命令的输出重定向到终端，输出都将附加到当前目录的 nohup.out 文件中。如果当前目录的 nohup.out 文件不可写，输出重定向到\$HOME/nohup.out文件中。如果没有文件能创建或打开以用于追加，那么 command 参数指定的命令不可调用。如果标准错误是一个终端，那么把指定的命令写给标准错误的所有输出作为标准输出重定向到相同的文件描述符。

nohup可以使程序能够忽略挂起信号，继续运行。用户退出时会挂载，而nohup可以保证用户退出后程序继续运行。如果标准输入是终端，请将其从/dev/null重定向。如果标准输出是终端，则将输出附加到“nohup.out”(如果可能的话)，否则追加到“\$home/nohup.out”。如果标准错误是终端，请将其重定向到标准输出。若要将输出保存到文件中，请使用“nohup COMMAND > FILE”。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
nohup cmd
```

sh

选项

```
--help          # 显示帮助文档  
--version       # 显示命令版本信息
```

sh

举例

以nohup形式运行find指令

```
[root@localhost ~]$ nohup find /weijie/ -name *.c      #运行find  
nohup: 忽略输入并把输出追加到"nohup.out"          #结果会保存到nohup.out文件中  
[root@localhost ~]$ cat nohup.out                   #查看结果  
/weijie/11.c  
/weijie/4.c  
/weijie/2.c  
/weijie/3.c  
/weijie/5.c  
/weijie/1.c
```

sh

使用nohup命令提交作业，如果使用nohup命令提交作业，那么在缺省情况下该作业的所有输出都被重定向到一个名为nohup.out的文件中，除非另外指定了输出文件：

```
nohup command > myout.file 2>&1 &
```

sh

在上面的例子中，输出被重定向到myout.file文件中。

该指令表示不做挂断操作，后台下载

```
nohup wget site.com/file.zip
```

sh

下面命令，会在同一个目录下生成一个名称为 nohup.out 的文件，其中包含了正在运行的程序的输出内容

```
nohup ping -c 10 baidu.com
```

sh

nslookup - 查询域名DNS信息的工具

nslookup是一个查询DNS域名的工具，它有交互和非交互两种工作模式。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
nslookup [-option] [name | -] [server]
```

sh

选项

```
host          # 查询host的信息
server domain # 改变服务器
exit          # 退出
set keyword=value
              # - all, 打印所有的属性
              # - domain=name 设置查询的名字
              # - port=value 改变服务器端口
              # - type=value 改变查询的类型
              # - timeout=number 设置等待超时
              # - class={IN | CH | HS | ANY}
```

sh

举例

非交互模式查询

```
[sogrey@bogon ~]$ nslookup www.baidu.com
Server:  192.168.0.1
Address: 192.168.0.1#53
```

sh

```
Name: www.baidu.com
Address: 14.215.177.38
```

```
[sogrey@bogon ~]$
```

交互模式查询

```
[sogrey@bogon ~]$ nslookup # 交互模式
> www.baidu.com # 输入查询的域名
Server: 192.168.0.1
Address: 192.168.0.1#53

Name: www.baidu.com
Address: 14.215.177.38
> set all # 打印当前所有的配置信息
Default server: 192.168.0.1
Address: 192.168.0.1#53
Default server: 8.8.8.8
Address: 8.8.8.8#53

Set options:
  novc  nodebug  nod2
  search  recurse
  timeout = 0  retry = 3  port = 53
  querytype = A      class = IN
  srchlist =
>
```

ntpdate - 通过轮询指定服务器设置本地日期和时间从而确定正确的时间

ntpdate指令通过轮询指定为服务器参数的网络时间协议(NTP)服务器来设置本地日期和时间,从而确定正确的时间。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
ntpdate [选项] server
```

sh

选项

```
-a      # 启用身份验证功能并指定要用于身份验证的密钥标识符。密钥和密钥标识符必须 在客户端密钥文件和服务器 sh  
-B      # 强制始终使用adjtime系统调用来微调时间(即使测量到的偏移量大于±128ms)。 默认设置时在偏移量大于±  
-b      # 强制使用clock_settime系统调用来步进时间,而不是使用adjtime系统调用来微 调时间 (默认值)。 如果  
-d      # 启用调试模式,在该模式下ntpdate将经历所有步骤,而不仅仅是调整本地时钟。 另外还将输出可用于一般性  
-e      # 将执行身份验证功能的处理延迟指定为值 authdelay (以秒及其分为单位,有关详细信息请参阅xntpd(1N  
-k      # 将身份验证密钥文件的路径指定为字符串keyfile。 默认值为/etc/ntp.keys。 该文件应该采用xntpd中用  
-o      # 将外发数据包的NTP版本指定为整数版本(可以是1或2)。 默认值是3。 它允许将ntpdate与早期NTP版本一起  
-p      # 将要从每个服务器中获取的示例数指定为整数示例,其值的范围是 1~8(包括这 两个数)。默认值为4  
-q      # 输出偏移量测量结果、服务器层次以及延迟测量结果,但不调整本地时钟。它类似于“-d”选项,后者提供更为详  
-s      # 将日志记录输出从标准输出(默认)转移到系统 syslog (请参阅 syslog(3C))工具。它主要是为方便使用c  
-t      # 将等待服务器响应的最长时间指定为超时值,以秒及其分为单位。该值将四舍五入成0.2秒的倍数。默认值是  
-u      # 指示ntpdate将无特权的端口用于外发的数据包。在防火墙后,如果阻塞向特权端口的传入流量,并且希望与防  
-v      # 输出NTP版本号和偏移量测量信息
```

举例

同步时间

```
[root@localhost ~]$ ntpdate 202.112.29.82  
16 Aug 10:13:21 ntpdate[20212]: adjust time server 202.112.29.82 offset 0.006454 sec
```

sh

ntpq - 使用NTP模式6数据包与NTP服务器通信

ntpq指令使用NTP模式6数据包与NTP服务器通信,能够在允许的网络上查询的兼容的服务器。它以交互模式运行,或者通过命令行参数运行。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux		Arch Linux

语法

```
ntpq [-46dinp] [-c command] [host] [...]
```

sh

选项

-4	# 使用ipv4解析
-6	# 使用ipv6解析
-c	# 添加执行的命令到指定主机的命令列表
-d	# 打开调试模式
-i	# 使用交互模式
-n	# 以十进制格式显示主机地址
-p	# 显示服务器同级设备的列表

sh

举例

查看服务器同级设备列表

```
[root@localhost ~]$ ntpq -p          #查看同级服务器列表
      remote           refid      st t when poll reach   delay    offset  jitter
=====
 120.25.115.20  10.137.53.7    2 u     9   64      1  184.815  279.126   0.000
 203.107.6.88  10.137.55.181   2 u    12   64      1   71.254  230.184   0.000
[root@localhost ~]$
```

sh

ntpstat - 用于显示本机上一次和服务器同步时间的情况

ntpstat指令用于显示本机上一次和服务器同步时间的情况。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ntpstat
```

sh

选项

无

举例

```
[root@localhost ~]$ ntpstat
unsynchronised
 time server re-starting
 polling server every 8 s
```

sh

od - 输出文件的八进制、十六进制等格式编码的字节

将指定文件的内容以八进制、十进制、十六进制等编码方式显示。

od命令 用于输出文件的八进制、十六进制或其它格式编码的字节，通常用于显示或查看文件中不能直接显示在终端的字符。

常见的文件为文本文件和二进制文件。此命令主要用来查看保存在二进制文件中的值。比如，程序可能输出大量的数据记录，每个数据是一个单精度浮点数。这些数据记录存放在一个文件中，如果想查看下这个数据，这时候od命令就派上用场了。在我看来，od命令主要用来格式化输出文件数据，即对文件中的数据进行无二义性的解释。不管是IEEE754格式的浮点数还是ASCII码，od命令都能按照需求输出它们的值。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
od [选项] file
od [-abcdfilosx]... [FILE] [[+]OFFSET[.][b]]
od --traditional [OPTION]... [FILE] [[+]OFFSET[.][b]] [+][LABEL][.][b]
```

sh

选项

```
-A, --address-radix=RADIX # 设置偏移量的编码单位
-j, --skip-bytes=BYTES # 跳过指定书目的字符
-N, --read-bytes=BYTES # 输出指定字符数
-S, --strings[=BYTES] # 输出至少BYTES个图形字符的字符串
-t, --format=TYPE # 指定输出格式
-w, --width[=BYTES] # 设置每一行的最大字数
-v, --output-duplicates # 显示重复的数据
--traditional # 接受传统形式的参数

--help # 显示帮助文档
--version # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ cat test2.txt
123
23
212
[sogrey@bogon newDir3]$ od test2.txt # 以八进制显示
0000000 031061 005063 031462 031012 031061 000012
0000013
[sogrey@bogon newDir3]$ od -t c test2.txt # 以字符方式显示
```

sh

```
[sogrey@bogon newDir3]$ od -t c test2.txt # 以字节方式显示  
0000000 1 2 3 \n 2 3 \n 2 1 2 \n  
0000013  
[sogrey@bogon newDir3]$ od -b test2.txt # 使用单字节八进制解释进行输出, 注意左侧的默认地址格式为八字节  
0000000 061 062 063 012 062 063 012 062 061 062 012  
0000013  
[sogrey@bogon newDir3]$ od -c test2.txt # 使用ASCII码进行输出, 注意其中包括转义字符  
0000000 1 2 3 \n 2 3 \n 2 1 2 \n  
0000013  
[sogrey@bogon newDir3]$ od -t d1 test2.txt # 使用单字节十进制进行解释  
0000000 49 50 51 10 50 51 10 50 49 50 10  
0000013  
[sogrey@bogon newDir3]$ od -A d -c test2.txt # 设置地址格式为十进制  
0000000 1 2 3 \n 2 3 \n 2 1 2 \n  
0000011  
[sogrey@bogon newDir3]$ od -A x -c test2.txt # 设置地址格式为十六进制  
0000000 1 2 3 \n 2 3 \n 2 1 2 \n  
00000b  
[sogrey@bogon newDir3]$ od -j 2 -c test2.txt # 跳过开始的两个字节  
0000002 3 \n 2 3 \n 2 1 2 \n  
0000013  
[sogrey@bogon newDir3]$ od -N 2 -j 2 -c test2.txt # 跳过开始的两个字节, 并且仅输出两个字节  
0000002 3 \n  
0000004  
[sogrey@bogon newDir3]$ od -w1 -c test2.txt # 每行仅输出1个字节  
0000000 1  
0000001 2  
0000002 3  
0000003 \n  
0000004 2  
0000005 3  
0000006 \n  
0000007 2  
0000010 1  
0000011 2  
0000012 \n  
0000013  
[sogrey@bogon newDir3]$ od -w2 -c test2.txt # 每行仅输出2个字节  
0000000 1 2  
0000002 3 \n  
0000004 2 3  
0000006 \n 2  
0000010 1 2  
0000012 \n  
0000013  
[sogrey@bogon newDir3]$ od -w3 -c test2.txt # 每行仅输出3个字节  
0000000 1 2 3  
0000003 \n 2 3  
0000006 \n 2 1  
0000011 2 \n  
0000013  
[sogrey@bogon newDir3]$
```

passwd - 用于让用户可以更改自己的密码

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。

更改用户密码，超级用户可以修改所有用户密码，普通用户只能修改自己的密码。这个任务是通过调用LinuxPAM和LibuserAPI来完成的。本质上，它使用LinuxPAM将自己初始化为一个“passwd”服务，并利用配置的密码模块对用户的密码进行身份验证和更新。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
passwd [选项] user  
  
passwd [-k] [-l] [-u [-f]] [-d]  
       [-e] [-n mindays] [-x maxdays]  
       [-w warndays] [-i inactivedays]  
       [-S] [--stdin] [username]
```

sh

选项

```
-d          # 删除用户的密码，只有root用户才能使用  
-e          # 使用用户密码失效，强制用户下次登录改变密码，只有root用户才能使用  
-n          # 设置密码的最短有效时间，只有root用户才能使用  
-x          # 设置密码最大有效时间，只有root用户才能使用  
-S          # 显示简短的密码信息，只有root用户才能使用  
-l          # 锁定用户，只有root用户才能使用  
-u          # 解锁用户，只有root用户才能使用  
-k          # 选项-k用于指示更新只适用于过期的身份验证令牌(密码)；用户希望像以前一样保留其未过期的令牌  
-i          # 这将设置此帐户的过期密码将被视为不活动的天数，如果用户帐户支持密码生存期，则应禁用该帐户  
-w          # 这将设置用户将开始收到警告，如果用户帐户支持密码生存期，其密码将过期的天数。只对根用户有效  
--stdin     # 此选项用于指示passwd应从标准输入中读取新密码，该输入可以是管道。  
  
--help      # 显示帮助文档  
--version   # 显示命令版本信息
```

sh

举例

锁定用户，锁定之后不能登录

sh

```
[root@bogon 文档]$ passwd -l sogrey #锁定用户  
锁定用户 sogrey 的密码 。  
passwd: 操作成功  
[root@bogon 文档]$ passwd -u sogrey #解锁用户  
解锁用户 sogrey 的密码 。  
passwd: 操作成功
```

删除用户密码

sh

```
[root@bogon 文档]$ passwd -d sogrey #清除用户sogrey密码  
清除用户的密码 sogrey  
passwd: 操作成功  
[root@bogon 文档]$ su user01      #切换到普通用户user01  
[user01@bogon 文档]$ su sogrey    #从普通用户user01切换到普通用户sogrey, 不需要密码  
[sogrey@bogon 文档]$
```

paste - 将多个文件按列队列合并

paste命令 用于将多个文件按照列队列进行合并。

将指定的文件按照列的方式合并，将结果显示到标准输出设备上，相当于两个并列的cat命令。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
paste [选项] files
```

sh

选项

```
-d list, --delimiters=list      # 以指定的分隔符区取代tab  
-s                               # 合并同一个文件的多行  
  
--help                            # 显示帮助文档  
--version                         # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir]$ ls
new000  new001  test.txt  xx00  xx01
[sogrey@bogon newDir]$ cat new000
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon newDir]$ cat new001
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon newDir]$ paste new000 new001 # 将文件合并
石家庄今日新增16例确诊病例 特朗普夫人发文谴责国会暴乱
中国留美博士遇害 美驻华使馆慰问 理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon newDir]$ cat test.txt
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon newDir]$ paste -s test.txt # 将同一个文件的多行合并成多列
石家庄今日新增16例确诊病例 中国留美博士遇害 美驻华使馆慰问 特朗普夫人发文谴责国会暴乱 理塘文旅公司回应丁真抽
[sogrey@bogon newDir]$
```

patch - 为开放源代码软件安装补丁程序

patch命令 被用于为开放源代码软件安装补丁程序。让用户利用设置修补文件的方式，修改，更新原始文件。如果一次仅修改一个文件，可直接在命令列中下达指令依序执行。如果配合修补文件的方式则能一次修补大批文件，这也是Linux系统核心的升级方法之一。

使用diff指令比较两个文件的差异，将差异结果保存到一个文件patchfile，patch指令可以将这个patchfile更新到原始文件中。这个命令的实际作用就是修补补丁，Linux内核用的就是这种更新方式。

patch程序接受包含由diff程序生成的差异列表的补丁文件，并将这些差异应用于一个或多个原始文件，生成修补版本。通常，修补版本被放在正本上。可以进行备份；请参阅-b或-备份选项。要修补的文件的名称通常取自修补程序文件，但如果只有一个要修补的文件，则可以在命令行中将其指定为原始文件。启动时，patch试图确定diff列表的类型，除非被"-c(--context)","-e (--ed)","-n (--normal)","-u (-unified)"选项所否决。Context diffs和normal diffs是由补丁程序本身应用的，而ed diffs只是通过管道提供给ed(1)编辑器。patch尝试跳过任何前导垃圾，应用差异，然后跳过任何跟踪垃圾。因此，您可以将包含diff列表的文章或消息提供给patch。

使用上下文差异，并在较小程度上与正常的差异，补丁可以检测什么时候，在补丁中提到的行号是不正确的，并试图找到正确的位置应用每一个补丁。作为第一猜测，它采用了前面提到的行号，加或减任何偏移量，用于应用以前的主存。如果这不是正确的位置，补丁会前后扫描一组与WORK中给出的上下文相匹配的行。第一个补丁程序寻找上下文中所有行匹配的位置。如果找不到这样的位置，并且是上下文差异，并且最大模糊因子被设置为1或更多，则会忽略第一行和最后一行上下文进行另一次扫描。如果失败，并且最大模糊因子设置为2或更多，则忽略前两行和最后两行上下文，并进行另一次扫描。(默认的最大Fuzz因子为2。)

前缀上下文少于后缀上下文的块(在应用Fuzz之后)必须在文件的开头应用，如果它们的第一行号是1。前缀上下文多于后缀上下文的块(在应用Fuzz之后)必须在文件末尾应用。

如果修补程序找不到安装该修补程序的位置，它会将该块放到一个拒绝文件中，该文件通常是输出文件的名称加上.rej后缀，或者#if.rej将生成一个过长的文件名(即使在单个字符#后面加上文件名)，然后#替换文件名的最后一个字符)。被拒绝的块以统一的或上下文的diff格式出现。如果输入是一个正常的差异，那么许多上下文都是空的。拒绝文件中的块上的行号可能与修补程序文件中的行号不同：它们反映了大致位置补丁，认为失败的块属于新文件而不是旧文件。

当每个模块都完成时，您会被告知，如果这个块失败了，如果是的话，那么哪一行(在新的文件中)补丁认为应该继续执行。如果主程序安装在与diff中指定的行号不同的行，则会告诉您偏移量。一个单一的大偏移量可能表明一个大块头安装在错误的地方。你还会被告知，如果使用了一个模糊因素来进行匹配，那么你也应该稍微有点怀疑。如果给出了-详细的选项，你也会被告知与之完全匹配的大块头。

如果命令行中没有指定原始文件源文件，则修补程序试图使用以下规则从前面的垃圾中找出要编辑的文件的名称。

首先，patch获取候选文件名的有序列表，如下所示：

- 1) 如果头是context diff，则patch在标题中使用旧的和新的文件名。如果名称没有足够的斜线来满足"-pnum"或"-strik=num"选项，则会忽略它。名称"/dev/null"也被忽略。
- 2) 如果前面的垃圾中有Index: line，如果旧的和新的名称都不存在，或者如果修补程序符合POSIX，则修补程序在Index: line中使用名称。

3) 为了下列规则的目的，候选人文件名被认为是按顺序(旧的、新的、索引的)，而不管它们在头中出现的顺序是什么。

然后，修补程序从候选列表中选择一个文件名，如下所示

- 1) 如果存在一些已命名的文件，则修补程序选择符合POSIX的名称，否则选择最佳名称。
- 2) 如果patch没有忽略RCS、ClearCase、Perforce和SCCS(请参阅-g num或-get=num选项)，并且除了找到RCS、ClearCase、Perforce或SCCS母版之外，不存在任何命名文件，则patch将选择具有RCS、ClearCase、Perforce或SCCS母版的第一个命名文件。
- 3) 如果不存在命名文件，没有找到RCS、ClearCase、Perforce或SCCS母版，给出了一些名称，patch不符合POSIX，patch似乎创建了一个文件，选择了需要创建最少目录的最佳名称。
- 4) 如果上面的启发式方法没有产生任何文件名，则会要求您提供要修补的文件的名称，而修补程序将选择该名称。

要确定非空文件名列表中的最佳名称，patch首先取所有具有最少路径名称组件的名称；然后，它以最短的basename获取所有名称；在这些名称中，它取所有最短的名称；最后，它取第一个剩余的名称。

所有这些的结果是，在新闻界面中，您应该能够说出如下所示

```
| patch -d /usr/src/local/blurfl
```

sh

直接从包含修补程序的文章中修补blurfl目录中的文件。

如果修补程序文件包含多个修补程序，则patch试图应用每个修补程序，就好像它们来自不同的修补程序文件一样。这意味着，除其他外，假定要修补的文件的名称必须为每个diff列表确定，而每个diff列表之前的垃圾包含有趣的内容，如前面提到的文件名和修订级别。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
patch [选项] originfile patchfile
patch -pnum <patchfile
```

sh

选项

```

-b                                         # 产生备份文件。在修补文件时，重命名或复制原始文件，而不是删除它。当备份不存
--backup-if-mismatch                      # 在修补数据不完全吻合，而且没有可以要求备份文件的时候才备份。这是默认的，除
--no-backup-if-mismatch                   # 如果修补程序与文件不完全匹配，如果不请求备份，则不要备份文件。这是默认的，sh
-B pref, --prefix=pref                    # 使用简单方法确定备份文件名(请参阅-V方法或--version-control方法选项)，并
-binary                                     # 以二进制模式写入所有文件，但标准输出和/dev/tty除外。读取时，禁用将CRLF线
-c, --context                                # 将修补程序文件解释为普通上下文差异。
-d dir , --directory=dir                  # 设置工作目录
-D flag , --ifdef=define                 # 用指定的标志“#ifdef...#endif”去标记修改过的地方
--dry-run                                    # 打印应用修补程序的结果，而不实际更改任何文件。
-e, --ed                                      # 将修补数据解释性ed命令可以识别的数据
-E                                         # 如果修补后输出的文件是空白，那么删除文件。通常，此选项是不必要的，因为修补
-f, --force                                    # 假设用户确切地知道他或她正在做什么，并且不问任何问题。跳过标题不说明要修补
-F num , --fuzz=num                         # 设置最大fuzz因子。此选项仅适用于具有上下文的差异，并导致修补程序忽略多达那
-g num , --get=num                          # 当文件处于RCS或SCCS控制下，且不存在或只读，并与默认版本匹配时，或当文件处
-i file , --input=patchfile                # 读取指定的修补文件
-l, --ignore-whitespace                     # 松散匹配模式，以防tabs或空格在文件中被咀嚼。修补程序文件中的一个或多个空白
-n, normal                                    # 将修补数据解释成一般性的差异
-N, --forward                                 # 忽略似乎已反转或已应用的修补程序
-o outfile , --output=outfile               # 设置输出文件。如果outfile是要修补的文件之一，请不要使用此选项。当outfi
-pnum, --strip=num                           # 设置要剥离的路径层数。例如，假设修补程序文件中的文件名是“/u/howard/src/b
--posix                                       # 更严格地遵守POSIX标准：当从diff头直观文件名时，从列表(旧的、新的、索引)获
--quoting-style=word                        # 使用样式字引用输出名称。这个词应该是下列之一：
                                              # - literal, 不改变输出
                                              # - shell, 如果shell包含shell元字符或将导致不明确的输出，则引用shell的名
                                              # - shell-always, 引用shell的名称，即使它们通常不需要引用。
                                              # - c, 引用C语言字符串的名称。
                                              # - escape, 除省略周围的双引号字符外，引用与c相同。
                                              # - 你可以使用环境变量QUOTING_STYLE来设置这个选项的默认值，如果没有环境变
-r rejectfile , --reject-file=rejectfile   # 将拒绝放入拒绝文件，而不是默认的.rej文件。当拒绝文件为‘-’时
-R, --reverse                                  # 假设此修补程序是用交换的旧文件和新文件创建的。修补程序尝试在应用之前交换每
--reject-format=format                       # 以指定的格式(上下文或统一格式)生成拒绝文件。如果没有此选项，如果输入补丁是
-s, --silent, --quite                        # 不显示执行过程，除非发生错误
-t, --batch                                     # 自动跳过错误。
-u, --unified                                 # 将修补数据解释成一致化的差异
--verbose                                     # 显示详细执行过程
-V method, --version-control=method          # 确定备份文件名。method还可以由PATCH_VERSION_CONTROL、VERSION_
                                              # - existing, nil, 对已经有编号的文件进行编号备份，否则进行简单备份。这是默
                                              # - numbered, t, 进行编号备份。例如F的编号备份文件名为F~N~其中N是版本号。
                                              # - simple, never, 做简单的备份。-B或--prefix、-Y或--basename-prefix、-
-x num , --debug=num                         # 只将感兴趣的内部调试标志设置为修补程序
-Y preg , --basename-prefix=pref             # 使用简单方法确定备份文件名(请参阅-V)，并在生成备份文件名时将pref前缀
-z suffix , --suffix=suffix                  # 使用简单方法确定备份文件名(请参阅-V)，并使用suffix作为后缀。例如，使用“-d
-Z, --set-utc                                  # 根据上下文diff头中给出的时间戳设置修补文件的修改和访问时间，假设上下文di
--help                                         # 显示帮助文档
--version                                     # 显示命令版本信息

```

环境变量

- PATCH_GET，这指定默认情况下修补程序是否会丢失rcs、ClearCase、Perforce或scs中的只读文件；请参阅-g或-get选项。
- POSIXLY_CORRECT，如果已设置，则默认情况下补丁程序更严格地符合POSIX标准；请参见-POSIX选项。
- QUOTING_STYLE，--quoting-style选项的默认值。

- SIMPLE_BACKUP_SUFFIX, 扩展名用于简单备份文件名, 而不是.orig
- TMPDIR, TMP, TEM, 放置临时文件的目录; 修补程序使用此列表中设置的第一个环境变量。如果没有设置, 则默认值依赖于系统; 在unix主机上通常是/tmp。
- VERSION_CONTROL or PATCH_VERSION_CONTROL, 选择版本控制样式。

patch发送者注意事项

如果你要发送补丁, 有几件事你应该记住:

系统地创建补丁。一个很好的方法是命令“diff -Naur old new”。旧名和新名不应包含任何斜杠。diff命令的标题应该使用传统的Unix格式在通用时间中有日期和时间, 这样补丁接收者就可以使用-Z或--set-utc选项。下面是一个示例命令, 使用Bourneshell语法: “LC_ALL=C TZ=UTC0 diff -Naur gcc-2.7 gcc-2.8”。

告诉收件人如何应用修补程序, 方法是告诉收件人要到哪个目录, 以及要使用哪个修补程序选项。建议使用String-Np1选项。通过冒充收件人并将修补程序应用于原始文件的副本来测试您的过程。

您可以通过保存patchlevel.h文件来减轻人们的痛苦, 该文件修补程序可以增加补丁级别, 作为您发送的补丁文件中的第一个差异。如果您在补丁中放了一个prereq: line, 它不会让它们在没有警告的情况下按顺序应用补丁。

您可以通过发送一个比较/dev/null的diff或一个日期为Epoch(1970-01-01: 00: 00: 00 UTC)的空文件来创建一个文件。只有当要创建的文件在目标目录中不存在时, 这才有效。相反, 您可以通过发送上下文diff来删除文件, 将要删除的文件与日期为Epoch的空文件进行比较。该文件将被删除, 除非补丁符合POSIX和-E或-删除-空-文件选项没有提供。生成创建和删除文件的修补程序的一种简单方法是使用GNU diff的-N或-new-file选项。

如果收件人应该使用-pn选项, 则不要发送如下所示的输出:

```
diff -Naur v2.0.29/prog/README prog/README
--- v2.0.29/prog/README Mon Mar 10 15:13:12 1997
+++ prog/README Mon Mar 17 14:58:22 1997
```

sh

因为这两个文件名有不同的斜杠数, 不同版本的修补程序对文件名的解释也不同。为了避免混淆, 请发送如下所示的输出:

```
diff -Naur v2.0.29/prog/README v2.0.30/prog/README
--- v2.0.29/prog/README Mon Mar 10 15:13:12 1997
+++ v2.0.30/prog/README Mon Mar 17 14:58:22 1997
```

sh

避免发送比较备份文件名(如README.orig)的修补程序, 因为这可能会将补丁混淆为修补备份文件而不是真正的文件。相反, 发送比较不同目录中相同的基本文件名的修补程序, 例如, old/README和New/README

注意不要发出反向补丁, 因为它使人们怀疑他们是否已经应用了补丁。

尽量不要让修补程序修改派生文件(例如, 您的Makefile文件配置中有一行configure: configure.in), 因为接收方无论如何都应该能够重新生成派生文件。如果必须发送派生文件的差异, 则使用UTC生成差异, 让接收方使用-Z或--set-utc选项应用修补程序, 并让它们删除任何依赖于修补文件的未修补文件(例如, 使用make干净)。

虽然您可以通过将582个不同的列表放到一个文件中而不受影响, 但最好将相关的补丁分组到单独的文件

中，以防出现混乱。

警告

上下文差异不能可靠地表示空文件、空目录或符号链接等特殊文件的创建或删除。它们也不能表示对文件元数据的更改，如所有权、权限或一个文件是否是另一个文件的硬链接。如果这样的更改也是必需的，那么单独的指令(例如shell脚本)应该伴随补丁。

patch无法判断在ed脚本中行号是否关闭，只有在发现更改或删除时，才能在普通差异中检测到不良行号。使用模糊因子3的上下文差异可能存在同样的问题。在这些情况下，您可能应该做一个上下文差异，看看这些更改是否有意义。当然，没有错误的编译是一个很好的指示，表明补丁可以工作，但并不总是这样。

patch通常会产生正确的结果，即使需要进行大量的猜测。但是，只有当修补程序应用于与修补程序生成的文件完全相同的版本时，才能保证结果是正确的。

举例

```
[sogrey@bogon demos]$ diff test.txt test2.txt > diff.txt
[sogrey@bogon demos]$ cat diff.txt
1,2d0
< 石家庄今日新增16例确诊病例
< 中国留美博士遇害 美驻华使馆慰问
4,6c2,3
< 理塘文旅公司回应丁真抽烟
< 北京一确诊者隐瞒行程不配合流调
< 山西晋中新增2例无症状感染者
---
>
> 1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。
[sogrey@bogon demos]$ patch -p0 test.txt diff.txt
patching file test.txt
[sogrey@bogon demos]$ cat test.txt
特朗普夫人发文谴责国会暴乱

1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。
[sogrey@bogon demos]$
```

pgrep - 根据用户给出的信息在当前运行进程中查找并列出符合条件的进程ID（PID）

pgrep命令 以名称为依据从运行进程队列中查找进程，并显示查找到的进程id。每一个进程ID以一个十进制数表示，通过一个分割字符串和下一个ID分开，默认的分割字符串是一个新行。对于每个属性选项，用户可以在命令行上指定一个以逗号分割的可能值的集合。

pgrep指令可以按名字或者其他属性搜索指定的进程，显示出进程的id到标准输出。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
pgrep [-flvx] [-d delimiter] [-n|-o] [-P ppid,...] [-g pgrep,...] [-s sid,...] [-u euid,..]
```

选项

-d delimiter	# 设置用于分隔输出中每个进程ID的字符串(默认为换行符)。(仅限于pgrep)
-f	# 查找完成的文件名
-g pgrep	# 只匹配列出的进程组ID中的进程。进程组0被转换为pgrep或pkill自己的进程组。
-G gid	# 只匹配实际组ID列出的进程。可以使用数值或符号值。
-l	# 列出进程的名字和id
-n	# 选择最近执行的进程
-o	# 选择最早的进程
-P ppid	# 选择父pid匹配的进程
-s sid	# 只匹配进程会话ID列出的进程。会话ID 0被转换为pgrep或pkill自己的会话ID。
-t term	# 查找符合终端号的进程
-u euid	# 只匹配其有效用户ID列出的进程。
-U uid	# 只匹配实际用户ID列出的进程。可以使用数值或符号值。
-v	# 查找不符合条件的进程
-x	# 只匹配其名称(如果-f指定了命令行)与模式完全匹配的进程。
--help	# 显示帮助文档
--version	# 显示命令版本信息

举例

查看指定进程信息

sh

```
[sogrey@bogon ~]$ pgrep -l bash
10101 bash
[sogrey@bogon ~]$
```

查看某个终端的进程

sh

```
[sogrey@bogon 文档]$ pgrep -l -t tty1    #显示tty1下的进程id和进程名字
7734 Xorg
```

pico - 功能强大全屏幕的文本编辑器

pico命令 是功能强大全屏幕的文本编辑器。pico的操作简单，提供了丰富的快捷键。常用的快捷键如下：

```
Ctrl+G: 获得pico的帮助信息;  
Ctrl+O: 保存文件内容, 如果是新文件, 需要输入文件名;  
Ctrl+R: 在当前光标位置插入一个指定的文本文件内容;  
Ctrl+Y: 向前翻页;  
Ctrl+V: 向后翻页;  
Ctrl+w: 对文件进行搜索;  
Ctrl+K: 剪切当前文件行到粘贴缓冲区;  
Ctrl+U: 粘贴缓冲区中的内容到当前光标所在位置;  
Ctrl+C: 显示当前光标位置;  
Ctrl+T: 调用拼写检查功能, 对文档进行拼写检查;  
Ctrl+J: 段落重排;  
Ctrl+X: 退出, 当文件内容发生改变时, 提供是否保存修改。
```

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
pico [OPTION] [参数]
```

sh

选项

-b: 开启置换的功能;
-d: 开启删除的功能;
-e: 使用完整的文件名称;
-f: 支持键盘上F1、F2...功能键;
-g: 显示光标;
-h: 在线帮助;
-j: 开启切换的功能;
-k: 预设pico在使用剪下命令时, 会把光标所在的列的内容全部删除;
-m: 开启鼠标支持的功能, 您可用鼠标点选命令列表;
-n<间隔秒数>: 设置多久检查一次新邮件;
-o<工作目录>: 设置工作目录;
-q: 忽略预设值;
-r<编辑页宽>: 设置编辑文件的页宽;
-s<拼字检查器>: 另外指定拼字检查器;
-t: 启动工具模式;
-v: 启动阅读模式, 用户只能观看, 无法编辑文件的内容;
-w: 关闭自动换行, 通过这个参数可以编辑内容很长的列;
-x: 关闭页面下方的命令列表;
-z: 让pico可被Ctrl+z中断, 暂存在后台作业里;
+<列表编号>: 执行pico指令进入编辑模式时, 从指定的列数开始编辑。

pidof - 查找指定名称的进程的进程号ID号

pidof命令 用于查找指定名称的进程的进程号id号。

pidof可以查找指定名称的进程的pid，将结果送到标准输出。pidof有两种返回值：0，找到至少一个进程；1，没有找到进程。pidof实际上与killall5相同；程序根据调用它的名称进行操作。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
pidof [OPTION] [name]
```

sh

选项

```
-s      # 一次只响应一个进程号  
-c      # 只显示运行在root目录下的进程，这个选项只对root用户有效  
-o      # 忽略指定进程号的进程  
-x      # 同时显示在shell脚本运行中的相同名称
```

sh

举例

查看指定进程的id

```
[sogrey@bogon ~]$ pidof bash # 显示bash进程的id，有3个  
10101 3141 3021  
[sogrey@bogon ~]$
```

sh

一次只显示一个进程

```
[sogrey@bogon ~]$ pidof -s bash  
10101  
[sogrey@bogon ~]$
```

sh

ping - 测试主机之间网络的连通性

ping命令用来测试主机之间网络的连通性。执行ping指令会使用ICMP传输协议，发出要求回应的信息，若远端主机的网络功能没有问题，就会回应该信息，因而得知该主机运作正常。

ping指令可以发送ICMP请求到目标地址，如果网络功能正常，目标主机会给出回应信息。ping使用ICMP协议强制发送ECHO_REQUEST报文到目标主机，从主机或网关获取ICMP ECHO_RESPONSE。

ECHO_REQUEST数据报('pings')有一个IP和ICMP报头，后面跟着一个timeval结构体，然后是用于填充数据包的任意数量的“pad”字节。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
ping [选项] destination
```

sh

选项

```

-a          # 可听的ping
-A          # 自适应平包间隔适应往返时间，因此有效地不超过一个(或多个，如果设置了预加载)未回答的报文
-b          # 允许ping广播地址
-B          # 不允许ping更改探针的源地址。当ping启动时，该地址绑定到选定的地址。
-c count    # 指定ping的次数
-d          # 在所使用的套接字上设置SO_DEBUG选项。实际上，linux内核不使用这个套接字选项。
-F flow label # 在回送请求数据包上分配和设置20位flow label。(只有平6)。如果值为零，内核将分配随机数
-f          # 极限检测，不等收到回复就发送下一个请求，只有超级管理员才可以使用
-i interval # 指定发送的时间间隔
-I interface address # 将源地址设置为指定的接口地址。参数可以是数字IP地址或设备名称。当选择ipv 6链路本地地址时，将忽略此标志
-l preload   # 如果指定了预加载preload，ping将发送许多未等待回复的数据包。只有超级用户才能选择超长的报文
-L          # 抑制组播数据包的回送。此标志仅适用于ping目标为多播地址的情况
-n          # 用数字方式显示
-p pattern   # 您可以指定多达16个“PAD”字节来填充您发送的数据包。这对于诊断网络中与数据相关的问题很有用
-Q tos       # 在ICMP数据报中设置与服务相关的比特的质量。TOS可以是十进制数，也可以是十六进制数。值为零表示尽力而为
-q          # 不显示执行过程
-r          # 忽略正常的路由表
-R          # 记录路由
-s packetsize # 指定数据包的大小
-S sndbuf    # 设置套接字发送缓冲区sndbuf。如果没有指定，则选择它来缓冲不超过一个数据包
-t ttl       # 指定数据包的生存期TTL
-T timestamp option # 设置特殊的ip时间戳选项。可以是tsonly (only timestamps)，tsandaddr (timestamp and source address)
-M hint      # 选择路径MTU发现策略，可以是do (禁止碎片，即使是本地的)、want (当数据包大小较大时，希望接收方不要进行分片)或never (从不进行MTU发现)
-U          # 打印完整的user-to-user延迟
-w deadline  # 在ping退出之前指定一个超时(以秒为单位)，而不管发送或接收了多少数据包。
-W timeout   # 等待响应的时间，以秒为单位。该选项只影响任何响应的超时，否则ping将等待两个RTT
-v          # 显示详细执行过程
-V          # 显示版本

```

ICMP报文

没有选项的IP报头是20个字节。ICMP echo_Request数据包包含另外8字节的ICMP报头，后面跟着任意数量的数据。当给定一个数据包大小时，这表明了这个额外数据块的大小(缺省值为56)。因此，在ICMP ECHO_REPLY类型的IP数据包中接收的数据量总是比请求的数据空间(ICMP报头)多8个字节。

如果数据空间至少是结构体timeval的大小，Timeval ping使用这个空间的起始字节来包含它在计算往返时间时使用的时间戳。如果数据空间较短，则不提供往返时间。

重复和损坏的数据包

ping将报告重复和损坏的数据包。重复的数据包不应该发生，似乎是由不适当的链路级重传引起的。重复可能在许多情况下发生，很少(如果有的话)是一个好的迹象，尽管低水平的重复可能并不总是引起恐慌。损坏的数据包显然是引起警报的严重原因，并且经常表示ping数据包路径(网络中或主机中)的某个硬件出现故障。

尝试不同的数据模式

(内部)网络层不应根据数据部分中包含的数据对分组进行不同的处理。不幸的是，依赖于数据的问题已经被人们知道潜入网络，并且在很长一段时间内没有被发现。在许多情况下，会出现问题的特定模式是没有足够的“转换”的东西，例如所有的1或所有的零，或者在边缘的一个模式，例如几乎所有的零。仅仅指定

命令行上所有零的数据模式(例如)并不一定足够，因为感兴趣的模式是在数据链接级别上，而且您键入的内容与控制器发送的内容之间的关系可能很复杂。

这意味着，如果您有一个数据依赖的问题，您可能需要做大量的测试才能找到它。如果幸运的话，您可能会设法找到一个文件，该文件要么无法通过您的网络发送，要么需要比其他类似长度的文件更长的传输时间。然后，您可以检查这个文件是否有重复的模式，可以使用ping的-p选项进行测试。

TTL

IP数据包的TTL值表示数据包在被丢弃之前可以通过的最大IP路由器数。在当前的实践中，您可以期望Internet中的每个路由器将TTL字段减少一个。TCP/IP规范规定，TCP数据包的TTL字段应该设置为60，但许多系统使用较小的值(4.3BSD使用30，4.2使用15)。该字段的最大可能值为255，大多数Unix系统将ICMP ECHO_REQUEST数据包的TTL字段设置为255。这就是为什么您会发现您可以“ping”一些主机，但不能通过telnet(1)或ftp(1)到达它们。

在正常操作中，ping从它接收的数据包打印ttl值。当远程系统收到ping数据包时，它可以在响应中使用ttl字段执行以下三项任务之一。

1. 不改变它；这是Berkeley Unix系统在4.3BSD Tahoe发布之前所做的事情。在这种情况下，接收到的数据包中的TTL值将为255减去往返路径中的路由器数量。
2. 将其设置为255；这是目前Berkeley Unix系统所做的。在这种情况下，接收到的数据包中的TTL值将为255减去从远程系统到ping主机的路径中的路由器数量。
3. 将其设置为其他值。有些机器对ICMP数据包使用的值与它们对TCP数据包使用的值相同，例如30或60。其他人可能会使用完全狂野的价值观。

举例

ping广播地址

```
[root@localhost ~]$ ping 192.168.1.255
Do you want to ping broadcast? Then -b
[root@localhost ~]$ ping -b 192.168.1.255 #只有使用-b选项才能ping广播地址
WARNING: pinging broadcast address
PING 192.168.1.255 (192.168.1.255) 56(84) bytes of data.
64 bytes from 192.168.1.8: icmp_seq=1 ttl=64 time=0.219 ms
64 bytes from 192.168.1.8: icmp_seq=2 ttl=64 time=0.282 ms
64 bytes from 192.168.1.8: icmp_seq=3 ttl=64 time=0.215 ms
64 bytes from 192.168.1.8: icmp_seq=4 ttl=64 time=0.268 ms
^C
--- 192.168.1.255 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3684ms
rtt min/avg/max/mdev = 0.215/0.246/0.282/0.029 ms
```

测试目标地址是否畅通

sh

```
[root@localhost ~]$ ping -c 4 192.168.1.8 #指定发送数据包的次数
PING 192.168.1.8 (192.168.1.8) 56(84) bytes of data.
64 bytes from 192.168.1.8: icmp_seq=1 ttl=64 time=0.427 ms
64 bytes from 192.168.1.8: icmp_seq=2 ttl=64 time=0.196 ms
64 bytes from 192.168.1.8: icmp_seq=3 ttl=64 time=0.220 ms
64 bytes from 192.168.1.8: icmp_seq=4 ttl=64 time=0.329 ms

--- 192.168.1.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 0.196/0.293/0.427/0.092 ms
```

pkill - 可以按照进程名杀死进程

pkill命令 可以按照进程名杀死进程。pkill和killall应用方法差不多，也是直接杀死运行中的程序；如果您想杀掉单个进程，请用kill来杀掉。

pkill可以给指定的进程发送信息，它可以结束某个执行的进程或者目录登录的用户。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
pkill [-signal] [-fvx] [-n|-o] [-P ppid,...]  
      [-g pgrep,...] [-s sid,...] [-u euid,...]  
      [-U uid,...] [-G gid,...] [-t term,...] [pattern]
```

sh

选项

```
-f          # 查找完成的文件名  
-g pgrep   # 只匹配列出的进程组ID中的进程。进程组0被转换为pgrep或pkill自己的进程组。  
-G gid    # 只匹配实际组ID列出的进程。可以使用数值或符号值。  
-n          # 选择最近执行的进程  
-o          # 选择最早的进程  
-P ppid    # 选择父pid匹配的进程  
-s sid     # 只匹配进程会话ID列出的进程。会话ID 0被转换为pgrep或pkill自己的会话ID。  
-t term    # 查找符合终端号的进程  
-u euid    # 只匹配其有效用户ID列出的进程。  
-U uid     # 只匹配实际用户ID列出的进程。可以使用数值或符号值。  
-v          # 查找不符合条件的进程  
-x          # 只匹配其名称(如果-f指定了命令行)与模式完全匹配的进程。  
-signal    # 要发送的信号  
  
--help      # 显示帮助文档  
--version   # 显示命令版本信息
```

sh

返回值

- 0 一个或多个进程符合。
- 1 没有进程符合
- 2 命令的语法错误
- 3 致命错误：内存不足等。

举例

杀死wc进程

```
[root@localhost ~]$ pkill -SIGKILL wc          #向wc发送KILL信号, 杀死进程
[1]-  已杀死                  nice -n 19 wc
You have new mail in /var/spool/mail/root
[root@localhost ~]$ ps          #查看进程, wc已经杀死
  PID TTY      TIME CMD
 8266 pts/0    00:00:00 bash
 8554 pts/0    00:00:00 ps
```

sh

popd - 从目录堆栈中删除目录

popd指令用来从shell目录堆栈中删除记录，如果没有选项，那么先删除栈顶记录，然后将当前目录切换到新的栈项目录中。

主要用途

- 从目录堆栈中删除目录，如果是顶部目录被删除，那么当前工作目录会切换到新的顶部目录。
- 没有参数时，删除目录堆栈顶部。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
popd [-n] [+num] [-num]
```

sh

在没有参数的情况下，从堆栈中移除顶层目录，并对新的顶层目录执行CD。如果popd命令成功，也会执行dir，返回状态为0。如果遇到无效选项、目录堆栈为空、指定不存在的目录堆栈条目或目录更改失败，则popd返回false。

选项

+num	# 将从左起第num个目录删除，从0开始计数
-num	# 将从右起第num个目录删除，从0开始计数
-n	# 在从堆栈中删除目录时，取消目录的正常更改，以便只操作堆栈。

sh

举例

删除最右面的目录

```
[root@localhost doc]$ dirs.          #查看目录  
/home/sogrey/app/doc  
[root@localhost doc]$ popd -0        #删除最右面的，成功  
/home/sogrey/app
```

sh

删除最左面的目录

sh

```
[root@localhost doc]$ dirs.          #查看目录  
/home/sogrey/app/doc  
[root@localhost doc]$ popd +0        #删除最左面的, 成功  
/sogrey/app/doc
```

pr - 将文本文件转换成适合打印的格式

pr命令 用来将文本文件转换成适合打印的格式，它可以把较大的文件分割成多个页面进行打印，并为每个页面添加标题。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
pr [OPTION] [参数]
```

sh

选项

```
-h<标题> # 为页指定标题;  
-l<行数> # 指定每页的行数。
```

sh

文件：需要转换格式的文件。

ps - 报告当前系统的进程状态

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。

ps指令可以显示系统中当前进程的信息，它的输出结果是高度可定制的。如果您希望重复更新所选内容和显示的信息，请使用top(1)代替。

请注意，“ps-aux”与“ps aux”不同。POSIX和UNIX标准要求“ps-aux”打印名为“x”的用户拥有的所有进程，以及打印由-a选项选择的所有进程。如果名为“x”的用户不存在，此ps可以将命令解释为“ps aux”，并打印警告。此行为旨在帮助转换旧脚本和习惯。它是脆弱的，随时可能发生变化，因此不应依赖它。

默认情况下，ps选择所有具有相同有效用户ID(EUID=EUID)的进程作为当前用户，并与调用方相关联的终端。它显示进程ID(PID=PID)、与进程关联的终端(tname=TTY)、[dd-]hh: mm: SS格式的累计CPU时间(time=TIME)和可执行名称(ucmd=CMD)。默认情况下输出未排序。

使用BSD样式的选项将向默认显示中添加进程状态(stat=STAT)，并显示命令args(args=命令)而不是可执行名称。您可以使用PS_FORMAT环境变量重写此操作。使用BSD样式的选项还将更改流程选择，以包括您拥有的其他终端(TTY)上的进程；或者，这可以描述为将选择设置为筛选的所有进程的集合，以排除其他用户拥有的进程或终端上的进程。当选项被描述为“相同”时，不考虑这些影响，所以-M将被认为与Z相等。

这个ps通过读取“/proc”中的虚拟文件来工作。这个ps不需要是setuid kmem，也不需要有任何特权来运行。不要给这个ps任何特殊的权限。此ps需要访问名称列表数据以获得正确的WCHAN显示。对于2.6之前的内核，必须安装System.map文件。

CPU使用率当前表示为进程在整个生命周期中运行的时间百分比。CPU使用率不太可能达到100%。

SIZE和RSS字段不计算进程的某些部分，包括页表、内核堆栈、结构体thread_info和task_struct。这通常是至少20kb的内存大小。SIZE是进程的虚拟大小(代码数据堆栈)。

标记为“defunct”的进程是死进程(所谓的“僵尸”)，因为它们的父进程没有正确地销毁它们。如果父进程退出，则init(8)将销毁这些进程。

如果用户名的长度大于显示列的长度，则将显示数字用户ID。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
ps [OPTION]
```

sh

选项

```
--info          # 显示调试信息  
L              # 列出所有的格式  
  
--help          # 显示帮助文档  
--version       # 显示命令版本信息
```

sh

```
-A, -e          # 选择所有进程  
-N, --deselect # 反向选择不符合条件的  
T              # 选择符合的终端  
-a              # 选择除会话进程和与终端无关的进程以外的所有进程。  
a              # 此选项导致ps列出带有终端(TTY)的所有进程，或在与x选项一起使用时列出所  
-d              # 选择所有进程，忽略进程所有者  
g              # 全部的进程，包含会话领导  
r              # 将选择限制为仅运行进程。  
x              # 此选项导致ps列出您拥有的所有进程(与ps相同的EUID)，或者在与a选项一起使
```

sh

```
-C cmdlist      # 按命令名选择。这将选择在cmdlist中给出可执行名称的进程。  
-G grplist     # 选择进程所属的组id与给定list匹配的进程  
U userlist      # 按有效用户ID(EUID)或名称选择。有效用户ID描述进程使用其文件访问权限的  
-U userlist     # 按实际用户ID(RUID)或名称进行选择。真正的用户ID标识创建进程的用户，  
-g grplist      # 按会话或有效组名选择。  
p pidlist       # 按进程ID选择。  
-p pidlist      # 按进程ID选择。  
-s sesslist     # 按照会话ID选择  
t ttylist        # 通过tty选择。与-t和--tty几乎相同，但也可以与空ttylist一起使用，以指  
-t ttylist      # 选择指定的tty  
-u userlist      # 按有效用户ID(EUID)或名称选择，等价于“U”和“--user”  
--Group grplist # 等价于“-G”  
--User userlist # 等价于“-U”  
--group grplist # 按有效组ID(EGID)或名称选择。有效组ID描述进程使用其文件访问权限的组。  
--pid pidlist    # 等价于“-p”和“p”  
-ppid ppidlist   # 选择父进程id匹配的进程  
--sid sesslist   # 等价于“-s”  
--tty ttylist    # 等价于“-t”和“t”  
--user userlist  # 等价于“-u”和“U”  
-123            # 等价于“--sid 123”  
123             # 等价于“-pid 123”
```

sh

输出格式控制

```
sh
-f, -F          # 输出完整的格式。它还会导致输出命令参数。当与-L一起使用时，将添加NLWP(线程)
-O format       # 类似-o，但预加载了一些默认列。等价于“-o pid,format,state,tname,time,
O format       # 预加载的o。当用作格式设置选项时，它与-O完全相同，具有BSD个性。
-M              # 添加一列安全数据。
X              # 寄存器格式
Z              # 等价于“-M”
-c              # 显示-l选项的不同调度程序信息。
j              # BSD作业控制格式
-j              # 作业格式
l              # BSD长格式
-l              # 长格式，经常和“-y”一起使用
-o format       # 等价于“-o”和“--format”
-o format       # 用户定义格式。format是以空格分隔或逗号分隔的列表形式的单个参数，它提供了
s              # 显示信号格式
u              # 以用户为主的模式
v              # 以虚拟内存为主的模式
-y              # 不显示标志位
-Z              # 显示安全上下文格式
--format format # 和“-o”一样
--context        # 显示安全上下文格式
```

输出修饰符

```
sh
-H              # 显示进程层次结构
O order         # 排序，过时的操作
S              # 总结一些信息，如CPU使用情况，从死子进程到父进程。
c              # 显示真正的命令名。这是从可执行文件的名称派生出来的，而不是从argv值派生的。因此，没有显
e              # 在命令之后显示环境
f, --forest     # ASCII格式的进程层次结构
h              # 没有头
k spec         # 指定排序顺序，等价于“--sort”
-n namelist, N namelist # 正确的WCHAN显示需要名称列表文件，并且必须与当前Linux内核完全匹配才能得到正确的
                      # $PS_SYSPMAP
                      # $PS_SYSTEM_MAP
                      # /proc/*/wchan
                      # /boot/System.map-`uname -r`-
                      # /boot/System.map
                      # /lib/modules/`uname -r`/System.map
                      # /usr/src/linux/System.map
                      # /System.map
n              # WCHAN和User的数字输出。(包括所有类型的UID和GID)
-w, w           # 宽输出，对无限宽度使用此选项两次
--cols n, --columns n, --width n # 设置屏幕宽度
--cumulative    # 包括一些死子进程数据(作为父进程的和)
--headers       # 重复头行，每页输出一行
--no-headers    # 根本不打印标题行
--lines n, --rows n # 设置屏幕高度
--sort spec     # 排序。语法是“[+|-]key[, [+|-]key[,...]]”，例如，ps jax --sort=uid,-ppid,+pid
```

线程模式

```
H      # 把线程当做进程显示  
-L      # 显示线程的LWP NLWP  
-T      # 显示线程的SPID  
m, -m    # 在进程之后显示线程
```

sh

进程标志

这些值的和显示在“F”列中，该列由标志输出说明符提供。

```
1  已经fork, 但是没有执行。  
4  使用超级用户权限。
```

sh

进程状态码

下面是s、stat和state输出说明符(标头“stat”或“S”)将显示的用于描述进程状态的不同值：

```
D  不间断睡眠(通常为IO)。  
R  正在运行或可运行(在运行队列上)。  
S  可中断睡眠(等待事件完成)。  
T  停止, 要么是被作业控制信号阻止, 要么是因为它正在被跟踪。  
W  分页(自2.6.xx内核以来无效)。  
X  死了(不应该被看见)。  
Z  已停止(“僵尸”)进程, 终止但未由其父进程收获。
```

sh

对于bsd格式和当使用stat关键字时，可能会显示其他字符：

```
<  高优先级(对其他用户不好)。  
N  低优先级(对其他用户很好)。  
L  将页面锁定在内存中(用于实时和自定义IO)。  
s  是会话。  
l  是多线程的。  
+  在前台进程组中。
```

sh

AIX格式描述符

这个ps支持AIX格式描述符，它们的工作方式有点像printf(1)和printf(3)的格式代码。例如，正常的默认输出可以这样产生： ps -eo "%p %y %x %c

CODE	NORMAL	HEADER
%C	pcpu	%CPU
%G	group	GROUP
%P	ppid	PPID
%U	user	USER
%a	args	COMMAND
%c	comm	COMMAND
%g	rgroup	RGROUP
%n	nice	NI
%p	pid	PID
%r	pgid	PGID
%t	etime	ELAPSED
%u	ruser	RUSER
%x	time	TIME
%y	tty	TTY
%z	vsz	VSZ

标准格式说明符

以下是用于控制输出格式(例如， 使用选项-o)或使用GNU样式的“--sort”序选项对所选进程进行排序的不同关键字。例如，“ps -eo pid,user,args --sort user”。这个版本的ps试图识别大多数在ps的其他实现中使用的关键字。以下用户定义的格式说明符可能包含空格： args, cmd, comm, command, fname, ucmd, ucomm, lstart, bsdstart, start。某些关键字可能无法用于排序。

CODE	HEADER	说明
%cpu	%CPU	进程的CPU利用率为“#.#”格式。当前，它是CPU时间除以进程运行的时间(cputime/realtim比率)，表示为百分比。除非你是幸运的，否则它不会达到100%。(别名 pcpu)
%mem	%MEM	进程的驻留集大小与机器上物理内存的比率，以百分比表示。(别名 PMEM)
args	COMMAND	命令，它的所有参数都是字符串。可以显示对参数的修改。该列中的输出可能包含空格。标记为“已失效”的进程部分死亡，等待其父进程完全销毁。有时进程args将不可用；当发生这种情况时，ps将可执行文件的名称打印在括号中。(别名cmd，命令)。当最后指定该列时，该列将扩展到显示的边缘。如果ps不能确定显示宽度，例如当输出被重定向(管道)到一个文件或另一个命令时，输出宽度是未定义的。(它可以是80，无限，TERM等决定)环境变量COLUMNS或-cols选项可以用于精确地确定这种情况下的宽度。 w 或**-w**选项也可用于调整宽度。
blocked	BLOCKED	blocked信号掩码。根据字段的宽度，以十六进制格式显示32位或64位掩码。(别名sig_block, sigmask)。
bsdstart	START	命令开始的时间。如果进程在24小时前启动，则输出格式为“hh:mm”，否则为“mmm dd”(其中mmm是月份的三个字母)。
bsdtime	TIME	用户和系统的累积CPU时间，。显示格式通常为“mmm: ss”，但如果进程占用的cpu时间超过999分钟，则可以移到右边。
c	C	处理器利用率当前，这是进程生存期内使用百分比的整数值。(见%cpu)。
caught	CAUGHT	捕获信号的掩码，见信号(7)。根据字段的宽度，以十六进制格式显示32或64位掩码。(别名 sig_catch , sigcatch)

cgroup	header	显示进程所属的控制组。说明
class	CLS	进程的调度类。(别名 policy , cls)。字段的可能值是: - not reported TS SCHED_OTHER FF SCHED_FIFO RR SCHED_BATCH ISO SCHED_ISIDL SCHED_IDLE? unknown value
cls	CLS	同 class
cmd	CMD	同 args
comm	COMMAND	命令名(只有可执行的名称)。将不会显示对命令名的修改。标记为"已失效"的进程部分死亡, 等待其父进程完全销毁。该列中的输出可能包含空格。(别名 ucmd , ucomm)。当最后指定该列时, 该列将扩展到显示的边缘。如果ps不能确定显示宽度, 例如当输出被重定向(管道)到一个文件或另一个命令时, 输出宽度是未定义的(它可以是80, 无限, TERM)。 COLUMNS 环境变量或**--cols 选项可以用于精确地确定这种情况下的宽度。 w 或 -w** 选项也可用于调整宽度。
command	COMMAND	同 args
cp	CP	CPU使用率/ms
cpu_time	TIME	累计CPU时间, "[DD-]HH:MM:SS"格式。(别名 time)。
egid	EGID	进程的有效组ID数为十进制整数。(别名 gid)。
egroup	EFROUP	进程的有效组ID。如果可以获得并且字段宽度允许, 这将是文本组ID, 否则将是十进制表示。(别名 group)。
eip	EIP	指令指针
esp	ESP	栈指针
etime	ELAPSED	自进程启动以来, 以[dd-]hh:]mm: SS形式运行的时间。
euid	EUID	有效用户ID, 别名 uid
euser	EUSER	有效用户名。如果可以获得并且字段宽度允许, 这将是文本用户ID, 否则将是十进制表示。 n 选项可用于强制十进制表示。(别名 uname , user)。
f	F	与进程关联的标志, 请参阅流程标志部分。(别名 flag , flags)。
fgid	FGID	文件系统访问组ID。(别名 fsgid)。
fgroup	FGROUP	文件系统访问组ID。如果可以获得并且字段宽度允许, 这将是文本用户ID, 否则将是十进制表示。(别名 fsgroup)
flag	F	同 f
flags	F	同 f
fname	COMMAND	进程可执行文件的基名的前8个字节。该列中的输出可能包含空格。
fuid	FUID	文件系统访问用户ID。(别名 fsuid)。
fuser	FUSER	文件系统访问用户ID。如果可以获得并且字段宽度允许, 这将是文本用户ID, 否则将是十进制表示。
gid	GID	同 egid
group	GROUP	同 egroup

ignored_CODE	IGNORED_HEADER	忽略命令行参数的掩码，根据子权限的定义，以下八进制掩码或十六进制掩码或64位掩码。(别名 sig_ignore , sigignore)
label	LABEL	安全标签，最常用于SELinux上下文数据。这是针对在高安全系统上发现的强制访问控制("MAC")。
lstart	STARTED	命令开始的时间。
lwp	LWP	正在报告的LWP(轻量过程或线程)ID。(别名 spid , tid)
ni	NI	nice值，范围从19(最好)到-20(对他人不友好)。(别名 nice)。
nice	NI	同 ni
nlwp	NLWP	进程中的lwps(线程)数。(别名 thcount)。
nwchan	WCHAN	进程处于休眠状态的内核函数的地址(如果需要内核函数名称，请使用wchan)。正在运行的任务将在本列中显示一个破折号('')。
pcpu	%CPU	同**%cpu**
pending	PENDING	挂起信号的掩码。进程上挂起的信号不同于单个线程上的待决信号。使用 m 选项或**-m 选项查看两者。根据字段的宽度，以十六进制格式显示32位或64位掩码。(别名 sig *)。
pgid	PGID	进程组ID或相应的流程组领导的进程ID。(别名 pgrp)。
pgrp	PGRP	同 pgid
pid	PID	进程的进程ID号
pmem	%MEM	同**%mem**
policy	POL	同 cls
ppid	PPID	父进程id
psr	PSR	进程当前分配给的处理器。
rgid	RGID	真实的组id
rgroup	RGROUP	真正的组名。如果可以获得并且字段宽度允许，这将是文本组ID，否则将是十进制表示。
rip	RIP	64位指令指针。
rsp	RSP	64位栈指针。
rss	RSS	驻留集大小，任务使用的非交换物理内存(以千字节为单位)。(别名 rssize , rsz)。
rssize	RSS	同rss
rsz	RSZ	同rss
rtprio	RTPRIO	实时优先级
ruid	RUID	实际用户ID
ruser	RUSER	真实的用户ID。如果可以获得并且字段宽度允许，这将是文本用户ID，否则将是十进制表示。
s	S	最小状态显示(一个字符)。
	SCN	进程的调度策略。策略SCHED_OTHER(SCHED_Normal)、 SCHED_FIFO、SCHED_RR、SCHED_BATCH、SCHED_IDLE和

scnead	SCH HEADER	SCHED_FIFO、SCHED_RR、SCHED_BATCH、SCHED_ISO和SCHED_IDLE分别显示为0、1、2、3、4和5。
sess	SESS	会话ID或等效的会话领导的进程ID。(别名 session ， sid)。
sgi_p	P	进程当前正在执行的处理器。如果进程当前未运行或无法运行，则显示“*”。
sgid	SGID	保存的组ID。(别名 svgid)
sgroup	SGROUP	保存的组名。如果可以获得并且字段宽度允许，这将是文本组ID，否则将是十进制表示。
sid	SID	同 sess
sig	PENDING	同 pending
sigcatch	CAUGHT	同 caught
sigignore	IGNORED	同 ignored
sigmask	BLOCKED	同 blocked
size	SZ	如果进程要脏所有可写页，然后交换掉，则需要交换大约的交换空间。这个数字很粗糙！
spid	SPID	同lwp
stackp	STACKP	进程堆栈的底部(开始)地址
start	STARTED	命令开始的时候。如果进程在24小时前启动，则输出格式为“hh: mm: ss”，否则为“mmm dd”(其中mmm是三个字母的月份名称)。
start_time	START	进程的开始时间或日期。只有进程未启动的年份(即调用ps的年份)或“mmdd”(如果进程未在同一天启动)或“hh: mm”将显示。
stat	STAT	多字符进程状态。有关不同值的含义，请参见处理状态代码一节。如果只希望显示第一个字符，请参见 s 和 state 。
state	S	同s
suid	SUID	保存的用户ID。(别名 svuid)。
suser	SUSER	保存的用户名。如果可以获得并且字段宽度允许，这将是文本用户ID，否则将是十进制表示。(别名 svuser)
svgid	SVGID	同 sgid
svuid	SVUID	同 suid
sz	SZ	进程核心图像的物理页面大小。这包括文本、数据和堆栈空间。当前排除了设备映射；这可能会发生更改。参见 vsz 和 rss 。
thcount	THCNT	同 nlwp
tid	TID	同 lwp
time	TIME	统计CPU时间, "[DD-]HH:MM:SS"格式。(别名 cputime)。
tname	TTY	控制TY(终端)(别名 tt ， tty)。
tpgid	TPGID	进程连接到的TTY(终端)上的前台进程组的ID，如果进程没有连接到TTY，则为-1。
tt	TT	同 tname 。

ttyCODE	THEADER	同 tname 。	说明
ucmd	CMD	同 comm 。	
ucomm	COMMAND	同 comm 。	
uid	UID	同 euid 。	
uname	USER	同 euser 。	
user	USER	同 euser 。	
vsize	VSZ	同 vsz 。	
vsz	VSZ	进程的虚拟内存大小(1024字节单位)。当前排除了设备映射；这可能会发生更改。(别名 vszie)。	
wchan	WHAN	进程处于休眠状态的内核函数的名称，如果进程正在运行，则为“-”，如果进程是多线程且ps不显示线程，则为“*”。	

环境变量

下面的环境变量会影响ps的行为：

- COLUMNS，覆盖默认的宽度。
- LINES，覆盖默认的高度。
- PS_PERSONALITY，设置为POSIX中的一个，old, linux, bsd, sun, digital。
- CMD_ENV，设置为POSIX中的一个，old, linux, bsd, sun, digital。
- I_WANT_A_BROKEN_PS，解释过时命令。
- LC_TIME，日期格式。
- PS_COLORS，现在还不支持。
- PS_FORMAT，默认输出格式覆盖。
- PS_SYSMAP，默认名称列表(System.map)位置。
- PS_SYSTEM_MAP，默认名称列表(System.map)位置。
- POSIXLY_CORRECT，不要找借口忽视不好的“特性”。
- POSIX2，当设置为“on”时，充当POSIXLY_TRIDER。
- UNIX95，不要找借口忽视不好的“特性”。
- _XPG，取消CMD_ENV=irix非标准行为。

一般来说，设置这些变量是个坏主意。一个例外是CMD_ENV或PS_PERSONALITY，对于正常系统，可以将它们设置为Linux。如果没有这种设置，PS就会遵循Unix 98标准中无用的和坏的部分。

举例

sh

```
ps axo pid,comm,pcpu # 查看进程的PID、名称以及CPU 占用率
ps aux | sort -rnk 4 # 按内存资源的使用量对进程进行排序
ps aux | sort -nk 3 # 按 CPU 资源的使用量对进程进行排序
ps -A # 显示所有进程信息
ps -u root # 显示指定用户信息
ps -efL # 查看线程数
ps -e -o "%C : %p : %z : %a" | sort -k5 -nr # 查看进程并按内存使用大小排列
ps -ef # 显示所有进程信息，连同命令行
ps -ef | grep ssh # ps 与grep 常用组合用法，查找特定进程
ps -C nginx # 通过名字或命令搜索进程
ps aux --sort=-pcpu,+pmem # CPU或者内存进行排序,-降序, +升序
ps -f --forest -C nginx # 用树的风格显示进程的层次关系
ps -o pid,uname,comm -C nginx # 显示一个父进程的子进程
ps -e -o pid,uname=USERNAME,pcpu=CPU_USAGE,pmem,comm # 重定义标签
ps -e -o pid,comm,etime # 显示进程运行的时间
ps -aux | grep named # 查看named进程详细信息
ps -o command -p 91730 | sed -n 2p # 通过进程id获取服务名称
```

若要使用标准语法查看系统上的每个进程，请执行以下操作：

sh

```
ps -e
ps -ef
ps -eF
ps -ely
```

若要使用BSD语法查看系统上的每个进程，请执行以下操作：

sh

```
ps ax
ps ax
```

打印进程树

sh

```
ps -ejH
ps axjf
```

获取线程信息

sh

```
ps -eLf
ps axm
```

获取安全信息

sh

```
ps -eo euser,ruser,suser,fuser,f,comm,label
ps axZ
ps -eM
```

若要以用户格式将每个进程作为根进程运行(实际有效ID)，请执行以下操作：

```
ps -U root -u root
```

sh

要使用用户定义的格式查看每个进程:

```
ps -eo pid,tid,class,rtprio,ni,pri,psr,pcpu,stat,wchan:14,comm  
ps axo stat,euid,ruid,tty,tpgid,ses,pgid,ppid,pid,pcpu,comm  
ps -eopid,tt,user, fname,tmout,f,wchan
```

sh

只打印进程syslogd的PID

```
ps -C syslogd -o pid=
```

sh

只打印PID为42的进程名字

```
ps -p 42 -o comm=
```

sh

将目前属于您自己这次登入的 PID 与相关信息列示出来

```
[sogrey@bogon 文档]$ ps -l  
F S   UID    PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD  
0 S  1000  10957 10939  1 80   0 - 29205 do_wai pts/0    00:00:00 bash  
0 R  1000  11131 10957  0 80   0 - 37793 -        pts/0    00:00:00 ps  
[sogrey@bogon 文档]$
```

sh

- F 代表这个程序的旗标 (flag), 4 代表使用者为 super user
- S 代表这个程序的状态 (STAT), 关于各 STAT 的意义将在内文介绍
- UID 程序被该 UID 所拥有
- PID 就是这个程序的 ID !
- PPID 则是其上级父程序的ID
- C CPU 使用的资源百分比
- PRI 这个是 Priority (优先执行序) 的缩写, 详细后面介绍
- NI 这个是 Nice 值, 在下一小节我们会持续介绍
- ADDR 这个是 kernel function, 指出该程序在内存的那个部分。如果是个 running的程序, 一般就是 "-"
- SZ 使用掉的内存大小
- WCHAN 目前这个程序是否正在运作当中, 若为 - 表示正在运作
- TTY 登入者的终端机位置
- TIME 使用掉的 CPU 时间。
- CMD 所下达的指令为何

在预设的情况下, ps 仅会列出与目前所在的 bash shell 有关的 PID 而已, 所以, 当我使用 ps -l 的时候, 只有三个 PID。

列出目前所有的正在内存当中的程序

```
[sogrey@bogon 文档]$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root        1  1.4  0.1 128244  6860 ?      Ss  23:02  0:03 /usr/lib/systemd/syste
root        2  0.0  0.0      0     0 ?      S   23:02  0:00 [kthreadd]
root        3  0.0  0.0      0     0 ?      S   23:02  0:00 [ksoftirqd/0]
root        4  0.0  0.0      0     0 ?      S   23:02  0:00 [kworker/0:0]
...
sogrey    10084  0.0  0.2 477752 10524 ?
sogrey    10088  0.0  0.2 661556 12020 ?
sogrey    10091  0.0  0.2 557860 10104 ?
sogrey    10106  0.0  0.2 702096 10728 ?
sogrey    10121  0.1  0.3 525492 14708 ?
sogrey    10126  0.0  0.3 656656 13616 ?
sogrey    10158  0.5  0.7 1046924 32948 ?
sogrey    10269  0.0  0.0 313740  3416 ?
sogrey    10557  0.0  0.1 526932 5588 ?
sogrey    10562  0.0  0.1 376060  5756 ?
sogrey    10564  0.0  0.4 580512 19332 ?
sogrey    10568  0.0  0.3 482280 13228 ?
sogrey    10576  0.0  0.0 376028  3444 ?
sogrey    10586  0.0  0.0 302388  3572 ?
sogrey    10939  0.5  0.7 869080 30724 ?
sogrey    10955  0.0  0.0  8548   720 ?
sogrey    10957  0.0  0.0 116820  3396 pts/0
sogrey    11679  0.0  0.1 320008  4656 ?
root     12250  0.0  0.0 107968  620 ?
sogrey    13773  0.0  0.0 153296 1908 pts/0
R+  23:06  0:00 ps aux
[sogrey@bogon 文档]$
```

- USER: 该 process 属于那个使用者账号的
- PID : 该 process 的号码
- %CPU: 该 process 使用掉的 CPU 资源百分比
- %MEM: 该 process 所占用的物理内存百分比
- VSZ : 该 process 使用掉的虚拟内存量 (Kbytes)
- RSS : 该 process 占用的固定的内存量 (Kbytes)
- TTY : 该 process 是在那个终端机上面运作, 若与终端机无关, 则显示 ?, 另外, tty1-tty6 是本机上面的登入者程序, 若为 pts/0 等等的, 则表示为由网络连接进主机的程序。
- STAT: 该程序目前的状态, 主要的状态有
- R : 该程序目前正在运作, 或者是可被运作
- S : 该程序目前正在睡眠当中 (可说是 idle 状态), 但可被某些讯号 (signal) 唤醒。
- T : 该程序目前正在侦测或者是停止了
- Z : 该程序应该已经终止, 但是其父程序却无法正常的终止他, 造成 zombie (僵尸) 程序的状态
- START: 该 process 被触发启动的时间
- TIME : 该 process 实际使用 CPU 运作的时间
- COMMAND: 该程序的实际指令

列出类似程序树的程序显示

```

[sogrey@bogon 文档]$ ps -axjf
  PPID   PID  PGID   SID TTY      TPGID STAT   UID    TIME COMMAND
    0     2     0     0 ?          -1 S      0  0:00 [kthreadd]
    2     3     0     0 ?          -1 S      0  0:00 \_ [ksoftirqd/0]
    2     4     0     0 ?          -1 S      0  0:00 \_ [kworker/0:0]
    2     5     0     0 ?          -1 S<    0  0:00 \_ [kworker/0:0H]
    2     6     0     0 ?          -1 S      0  0:00 \_ [kworker/u2:0]
    2     7     0     0 ?          -1 S      0  0:00 \_ [migration/0]
...
    1  6416  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/at-spi2-registryd --use
    1  6456  6455  6455 ?          -1 S<1  1000 0:00 /usr/bin/pulseaudio --start --log-ta
    1  6581  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gnome-shell-calendar-se
    1  6604  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/evolution-source-regist
    1  6606  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/mission-control-5
    1  6609  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfs-udisks2-volume-mon
    1  6626  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/goa-daemon
    1  6627  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfs-gphoto2-volume-mon
    1  6642  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfs-goa-volume-monitor
    1  6654  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/goa-identity-service
    1  6691  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfs-afc-volume-monitor
    1  6710  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfs-mtp-volume-monitor
    1  6885  6159  6159 ?          -1 S1    1000 0:00 /usr/libexec/gsd-printer
    1  6919  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/evolution-calendar-fact
  6919  7010  6174  6174 ?          -1 S1    1000 0:00 \_ /usr/libexec/evolution-calendar-
  6919  7062  6174  6174 ?          -1 S1    1000 0:00 \_ /usr/libexec/evolution-calendar-
    1  7061  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/dconf-service
    1  7110  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/evolution-addressbook-f
  7110  7135  6174  6174 ?          -1 S1    1000 0:00 \_ /usr/libexec/evolution-addressbo
    1 10077  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfsd-trash --spawner :
    1 10121  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/tracker-store
    1 10158  6174  6174 ?          -1 S1    1000 0:00 /usr/bin/nautilus --gapplication-ser
    1 10269  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/gvfsd-metadata
    1 10568 10557  6174 ?          -1 S1    1000 0:00 /usr/libexec/ibus-x11 --kill-daemon
    1 10576  6174  6174 ?          -1 S1    1000 0:00 /usr/libexec/ibus-portal
    1 10939  6174  6174 ?          -1 S1    1000 0:01 /usr/libexec/gnome-terminal-server
  10939 10955  6174  6174 ?          -1 S    1000 0:00 \_ gnome-pty-helper
  10939 10957 10957 10957 pts/0    17771 Ss    1000 0:00 \_ bash
  10957 17771 17771 10957 pts/0    17771 R+    1000 0:00 \_ ps -axjf
    1 14389  2863  2863 ?          -1 S1      0  0:00 /usr/sbin/abrt-dbus -t133
[sogrey@bogon 文档]$

```

找出与 cron 与 syslog 这两个服务有关的 PID 号码

```

[sogrey@bogon 文档]$ ps aux | egrep '(cron|syslog)'
root    2989  0.0  0.0  7140  288 ?          Ss    23:02  0:00 /usr/sbin/mcelog --ignorenodev
root    5006  0.0  0.1 218596  8520 ?          Ssl   23:02  0:00 /usr/sbin/rsyslogd -n
root    5020  0.1  0.0 126300  1704 ?          Ss    23:02  0:00 /usr/sbin/crond -n
sogrey  6456  0.0  0.1 1281356  7160 ?          S<1  23:03  0:00 /usr/bin/pulseaudio --start --
root    20412  0.0  0.0 115380  1592 ?          S    23:09  0:00 /bin/bash /usr/libexec/sysmonit
sogrey  20417  0.0  0.0 112744  1032 pts/0    S+   23:09  0:00 grep -E --color=auto (cron|sys
[sogrey@bogon 文档]$

```

把所有进程显示出来，并输出到ps001.txt文件

```
ps -aux > ps001.txt
```

sh

显示用户Sogrey的进程信息

```
[sogrey@bogon 文档]$ ps -u sogrey
  PID TTY      TIME CMD
 6062 ?        00:00:00 gnome-keyring-d
 6159 ?        00:00:00 gnome-session-b
 6173 ?        00:00:00 dbus-launch
 6174 ?        00:00:00 dbus-daemon
 6234 ?        00:00:00 imsettings-daem
 6238 ?        00:00:00 gvfsd
...
10557 ?        00:00:00 ibus-daemon
10562 ?        00:00:00 ibus-dconf
10564 ?        00:00:00 ibus-ui-gtk3
10568 ?        00:00:00 ibus-x11
10576 ?        00:00:00 ibus-portal
10586 ?        00:00:00 ibus-engine-sim
10939 ?        00:00:02 gnome-terminal-
10955 ?        00:00:00 gnome-pty-help
10957 pts/0    00:00:00 bash
11679 ?        00:00:00 ibus-engine-lib
14145 ?        00:00:17 gnome-shell
23460 pts/0    00:00:00 ps
[sogrey@bogon 文档]$
```

sh

pstree - 以树状图的方式展现进程之间的派生关系

pstree命令 以树状图的方式展现进程之间的派生关系，显示效果比较直观。

pstree显示正在运行的进程的树形结构，树以PID为根；如果省略了pid则以init为根。如果指定了用户名，则显示根植于该用户拥有的进程的所有进程树。如果**pstree**被调用为**pstree.x11**，那么它将提示行尾的用户按RETURE，并且在这种情况下不会返回。这对于在x终端中运行**pstree**非常有用。

pstree通过将相同的分支放在方括号中并以重复计数作为前缀，在视觉上合并它们。例如：

```
init---getty
  |-getty
  |-getty
  '-getty
# 变成下面的样子
init---4*[getty]
```

进程的子线程在父进程中找到，并以大括号显示进程名，例如：

```
icecast2---13*[{icecast2}]
```

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
pstree [OPTION]
```

选项

```
-a          # 显示每个进程的完整指令，包括路径、参数
-A          # 使用ascii码显示树形
-c          # 关闭精简表示法
-G          # 使用VT 100线条绘制字符
-h          # 高亮显示正在执行的程序
-H          # 类似“-h”，但是突出显示指定的进程。与-h不同，如果高亮显示不可用，pstree在使用-H时会忽略
-l          # 长格式显示
-n          # 以进程号排序，默认以名字排序
-p          # 显示pid
-u          # 显示用户
-U          # 以utf-8显示字符
-v          # 显示命令版本信息
-z          # 每个SELinux的上下文
```

举例

显示当前所有进程的进程号和进程id

```
pstree -p
```

sh

显示所有进程的所有详细信息，遇到相同的进程名可以压缩显示。

```
pstree -a
```

sh

获取 SSH 会话的 PID

```
pstree -p | grep ssh
```

sh

```
#  |-sshd(1221)-+sshd(2768)-+bash(2770)-+grep(2810)
#  |           `sshd(2807)-+sshd(2808)
```

从上方的输出中，你可以看到 sshd 进程与分支的树形图。sshd 的主进程是 sshd (1221)，另两个分支分别为 sshd (2768) 和 sshd (2807)

显示完成的树形结构

```
[root@localhost ~]$ pstree -a
init
|---NetworkManager --pid-file=/var/run/NetworkManager/NetworkManager.pid
|   |---dhclient -d -4 -sf /usr/libexec/nm-dhcp-client.action -pf /var/run/dhclient-eth0.pid ...
|   |   `---{NetworkManager}
|---VBoxClient --clipboard
|   |---VBoxClient --clipboard
```

sh

显示进程号

```
[root@localhost ~]$ pstree -p
init(1)---NetworkManager(6362)---dhclient(6377)
|           |---{NetworkManager}(6379)
|           |---VBoxClient(7869)---VBoxClient(7870)---{VBoxClient}(7872)
|           |---VBoxClient(7882)---VBoxClient(7883)
|           |---VBoxClient(7890)---VBoxClient(7891)---{VBoxClient}(7894)
|           |---VBoxClient(7898)---VBoxClient(7899)---{VBoxClient}(7901)
|           |           `---{VBoxClient}(7903)
|           |---VBoxClient(7306)---VBoxClient(7308)
|           |---VBoxClient(7312)---VBoxClient(7314)---{VBoxClient}(7317)
|           |---VBoxClient(7318)---VBoxClient(7320)---{VBoxClient}(7323)
|           |           `---{VBoxClient}(7325)
```

sh

pushd - 将目录添加到目录堆栈顶部

pushd指令用来将目录加入堆栈的顶部，并且切换到该目录。如果没有任何参数，那么将栈最上面的两个记录切换位置。

主要用途

- 将目录添加到目录堆栈顶部，切换当前工作目录到该目录。
- 旋转目录堆栈，使堆栈的新顶部成为当前工作目录。
- 没有参数时，交换目录堆栈的前两个目录。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
pushd [-n] [+num] [-num]
pushd [-n] [dir]
```

sh

如果没有参数，则交换前两个目录并返回0，除非目录堆栈为空。如果pushd命令成功，那么还会执行dir。如果使用第一种形式，除非cd到dir失败，否则pushd返回0。对于第二种表单，除非目录堆栈为空，指定了一个不存在的目录堆栈元素，或者将目录更改为指定的新的当前目录，否则PUSD返回0。

选项

```
+num            # 将从左起第num个目录移动到栈顶，从0开始计数
-num            # 将从右起第num个目录移动到栈顶，从0开始计数
-n              # 添加记录的时候，不切换目录
dir            # 将dir添加到顶部的目录堆栈，使其成为新的当前工作目录。
```

sh

举例

不使用任何参数

```
[root@localhost dev]$ dirs.                #查看目录
/dev/so/sogrey
You have new mail in /var/spool/mail/root
[root@localhost dev]$ pushd                #直接调用pushd，栈顶的两个目录位置切换
/so/dev/sogrey
```

sh

删除最左面的目录

```
[root@localhost wj]$ pushd /etc      #增加目录  
/etc/so/dev/sogrey  
[root@localhost etc]$          #位置已经切换
```

sh

移动目录的位置

```
[root@localhost wj]$ dirs.      #查看目录  
/etc/so/dev/sogrey  
[root@localhost etc]$ pushd -0      #将最右面的移动到栈顶  
/sogrey/etc/so/dev  
[root@localhost sogrey]$      #成功移动，并且切换位置
```

sh

put - 登录ftp服务器之后将文件上传到服务器

使用lftp登录ftp服务器之后，可以使用put指令将文件上传到服务器。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
put [-E] [-a] [-c] [-O base] lfile [-o rfile]
```

sh

选项

```
-o      # 指定输出文件的名字，不指定则使用原来的名字  
-c      # 如果失败，持续获取  
-E      # 获取之后，删除源文件  
-a      # 使用ascii模式  
-O      # 指定输出文件存放的目录
```

sh

举例

上传文件

```
[root@localhost ~]$ lftp 192.168.1.8      #登录服务器  
lftp 192.168.1.8:> cd pub                 #切换目录  
lftp 192.168.1.8:/pub> put 3.c             #上传文件  
65 bytes transferred  
lftp 192.168.1.8:/pub> ls                  #查看内容，已经上传成功  
-rwxrwxrwx  1 0      0  2375494044 Aug 14 06:38 1.zip  
-rw-r--r--  1 0      0  0 Oct 02 01:19 11c  
-rw-r--r--  1 0      0  0 Oct 02 01:19 22c  
-rw-----  1 14     50  65 Oct 02 01:48 3.c  
drwxr-xr-x  2 0      0  4096 Oct 02 01:12 testftp  
lftp 192.168.1.8:/pub>
```

sh

pwck - 用来验证系统认证文件内容和格式的完整性

pwck命令 用来验证系统认证文件/etc/passwd和/etc/shadow的内容和格式的完整性。

检查用户密码文件"/etc/passwd"和"/etc/shadow"的完整性，将验证结果送到标准输出。提示用户删除格式不正确或有其他不可更正错误的条目。检查以验证每个条目是否具有：正确的字段数、唯一有效的用户名、有效的用户和组标识符、有效的主组、有效的家目录、有效的登录shell。

当指定了第二个文件参数或系统上存在"/etc/shadow"时，就启用了shadow检查。它会检查一下信息：每个passwd条目都有一个匹配的shadow条目，每个shadow条目都有一个匹配的passwd条目，在shadow文件中指定了密码，shadow条目有正确的字段数，shadow条目在shadow中是唯一的，最近的密码更改不会在将来发生。

检查正确的字段数和唯一用户名是致命的。如果条目有错误的字段数，则会提示用户删除整行。如果用户没有肯定地回答，所有进一步的检查都会被绕过。提示删除具有重复用户名的条目，但仍将进行其余检查。所有其他错误都是警告，并鼓励用户运行usermod命令来更正错误。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
pwck [选项]
```

sh

选项

-q	# 仅显示报错信息
-r	# 以只读模式执行
-s	# 使用UID作为文件的排序依据
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

配置

下面"/etc/login.defs"中的配置变量更改了该工具的行为：

PASS_MAX_DAYS，可以使用密码的最大天数。如果密码早于此，则将强制进行密码更改。如果没有指定，将假定-1(这将禁用限制)。

PASS_MIN_DAYS，密码更改之间允许的最短天数。任何密码更改尝试比这更早将被拒绝。如果未指定，则假定为-1(这将禁用限制)

PASS_WARN_AGE，在密码过期前发出警告的天数。零表示警告只在到期之日发出，负值表示没有发出

警告。如未指定，则不会提供警告。

相关文件

- `/etc/group` 组账户信息。
- `/etc/passwd` 用户账户信息。
- `/etc/shadow` 安全用户帐户信息。

返回值

- 0 成功
- 1 无效的命令
- 2 1个或多个密码出错
- 3 无法打开密码文件
- 4 不能锁定密码文件
- 5 不能更新密码文件
- 6 无法排序密码文件

举例

普通用户调用pwck

```
[sogrey@bogon ~]$ sudo pwck # 查看密码文件
用户“ftp”: 目录 /var/ftp 不存在
用户“saslauth”: 目录 /run/saslauthd 不存在
用户“gluster”: 目录 /var/run/gluster 不存在
用户“pulse”: 目录 /var/run/pulse 不存在
用户“gnome-initial-setup”: 目录 /run/gnome-initial-setup/ 不存在
用户“avahi”: 目录 /var/run/avahi-daemon 不存在
用户“vboxadd”: 目录 /var/run/vboxadd 不存在
pwck: 无改变
[sogrey@bogon ~]$ echo $? #打印返回值, 这个返回值在shell变量“$?”中
2
[sogrey@bogon ~]$
```

sh

root调用pwck

```
[sogrey@bogon ~]$ sudo -i
[root@bogon ~]# pwck
用户“ftp”: 目录 /var/ftp 不存在
用户“saslauth”: 目录 /run/saslauthd 不存在
用户“gluster”: 目录 /var/run/gluster 不存在
用户“pulse”: 目录 /var/run/pulse 不存在
用户“gnome-initial-setup”: 目录 /run/gnome-initial-setup/ 不存在
用户“avahi”: 目录 /var/run/avahi-daemon 不存在
用户“vboxadd”: 目录 /var/run/vboxadd 不存在
pwck: 无改变
[root@bogon ~]# pwck /etc/passwd
用户“ftp”: 目录 /var/ftp 不存在
用户“saslauth”: 目录 /run/saslauthd 不存在
用户“gluster”: 目录 /var/run/gluster 不存在
用户“pulse”: 目录 /var/run/pulse 不存在
用户“gnome-initial-setup”: 目录 /run/gnome-initial-setup/ 不存在
用户“avahi”: 目录 /var/run/avahi-daemon 不存在
用户“vboxadd”: 目录 /var/run/vboxadd 不存在
pwck: 无改变
[root@bogon ~]#
```

pwd - 显示当前工作目录

查看用户当前的工作目录，输出完整路径。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
pwd [OPTION]
```

sh

选项

-L	# 从环境变量中使用PWD
-P	# 跳过所有的符号链接
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

```
[sogrey@bogon 文档]$ pwd  
/home/sogrey/文档
```

sh

quit - 退出ftp服务器

使用quit指令可以退出ftp服务器，另外还有一些指令可以退出：bye、exit、close。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
quit
```

sh

reject - 指示打印系统拒绝发往指定目标打印机的打印任务

reject命令 属于CUPS套件，用于指示打印系统拒绝发往指定目标打印机的打印任务。

reject指令用来设置拒绝向目标打印机发送打印任务。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
reject [ -E ] [ -U username ] [ -h hostname[:port] ] [ -r reason ] destination(s) sh
```

选项

-E	# 强制加密
-U	# 连接打印机的时候，发送用户名
-h	# 选择目标打印机ip和端口
-r	# 设置拒绝的原因

sh

举例

拒绝向目标打印机发送打印任务

```
[root@localhost /]$ reject printer01      #拒绝发送打印请求  
You have new mail in /var/spool/mail/root  
[root@localhost /]$ lpr /weijie/5.c      #打印文件，可以看到结果失败了  
lpr: Destination "printer01" is not accepting jobs. sh
```

renice - 修改正在运行的进程的调度优先级

renice命令 可以修改正在运行的进程的调度优先级。预设是以程序识别码指定程序调整其优先权，您亦可以指定程序群组或用户名称调整优先权等级，并修改所有隶属于该程序群组或用户的程序的优先权。只有系统管理者可以改变其他用户程序的优先权，也仅有系统管理者可以设置负数等级。

renice指令可以重新调整程序运行的优先级，可以通过进程id、用户id、组id来修改优先级。修改组的等级，影响组内所有用户的所有进程优先级；修改用户等级，影响该用户的所有进程优先级。除了超级用户之外，其他用户只能改变他们拥有的进程的优先级，并且只能在0到PRIO_MAX(20)范围内单调地增加他们的“nice value”。(这防止了凌驾于行政法规之上。)超级用户可以更改任何进程的优先级，并将优先级设置为PRIO_MIN(-20)~PRIO_MAX。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
renice -n N -u username  
renice -n N -g gid  
renice -n N -p pid
```

sh

选项

```
-h, --help          # 显示帮助文档  
-v, --version      # 显示命令版本  
-n, --priority    # 优先级  
-u, --user=name    # 设置指定用户的优先级  
-g, --pgrp=gid     # 设置组的优先级  
-p, --pid=PID       # 设置指定进程的优先级
```

sh

举例

通过pid修改指令wc优先级

```
[root@localhost ~]$ ps -ao "%p%y%x%c%n"          # 查看进程优先级  
 PID TTY      TIME COMMAND      NI  
8321 pts/0    00:00:00 wc        19  
8451 pts/0    00:00:00 ps        0  
[root@localhost ~]$ renice -n 15 -p 8364          # 重设wc进程的优先级，变为15  
8364: old priority 19, new priority 15
```

sh

修改用户root的优先级

```
[root@localhost ~]$ renice -n 0 -u root          # 修改用户root的优先级为0
0: old priority -11, new priority 0
[root@localhost ~]$ ps -ao "%p%y%x%c%n"        # 查看进程优先级, 当前进程都是root用户的, 优先级都是0
 PID TTY      TIME COMMAND      NI
 8321 pts/0    00:00:00 wc          0
 8364 pts/0    00:00:00 wc          0
 8458 pts/0    00:00:00 ps          0
```

sh

resize2fs - 调整ext2\ext3\ext4文件系统的大小

调整ext2\ext3\ext4文件系统的大小，它可以放大或者缩小没有挂载的文件系统的大小。如果文件系统已经挂载，它可以扩大文件系统的大小，前提是内核支持在线调整大小。

size参数指定所请求的文件系统的新大小。如果没有指定任何单元，那么size参数的单位应该是文件系统的文件系统块大小。size参数可以由下列单位编号之一后缀：“s”、“K”、“M”或“G”，分别用于512字节扇区、千字节、兆字节或千兆字节。文件系统的大小可能永远不会大于分区的大小。如果未指定Size参数，则它将默认为分区的大小。

resize2fs程序不操作分区的大小。如果希望扩大文件系统，必须首先确保可以扩展基础分区的大小。如果您使用逻辑卷管理器LVM(8)，可以使用fdisk(8)删除分区并以更大的大小重新创建它，或者使用lrexport(8)。在重新创建分区时，请确保使用与以前相同的启动磁盘圆柱来创建分区！否则，调整大小操作肯定无法工作，您可能会丢失整个文件系统。运行fdisk(8)后，运行resize2fs来调整ext 2文件系统的大小，以使用新扩大的分区中的所有空间。

如果希望缩小ext2分区，请首先使用resize2fs缩小文件系统的大小。然后可以使用fdisk(8)缩小分区的大小。缩小分区大小时，请确保不使其小于ext2文件系统的新大小。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
resize2fs [选项] device [size]
resize2fs [ -fFpPM ] [ -d debug-flags ] [ -S RAID-stride ] device [ size ]
```

sh

选项

```
-d debug-flags      # 打开各种resize2fs调试特性，如果它们已经编译成二进制文件的话。调试标志应该通过从
                      # 2，调试块重定位。
                      # 4，调试iNode重定位。
                      # 8，调试移动inode表。
-f                  # 强制执行，覆盖一些通常强制执行的安全检查。
-F                  # 执行之前，刷新文件系统的缓冲区
-M                  # 将文件系统缩小到最小值
-p                  # 显示已经完成任务的百分比
-P                  # 显示文件系统的最小值
-S RAID-stride     # resize2fs程序将启发式地确定在创建文件系统时指定的RAID步长。此选项允许用户显式地
```

举例

显示sda1最小值

```
[root@localhost ~]$ resize2fs -P /dev/sda1
resize2fs 1.41.12 (17-May-2010)
Estimated minimum size of the filesystem: 37540
```

sh

设置sdb4为1k

```
[root@localhost ~]$ resize2fs /dev/sdb4 1k
resize2fs 1.41.12 (17-May-2010)
resize2fs: New size smaller than minimum (373) #小于最小值，失败
```

sh

restore - 所进行的操作和dump指令相反

restore命令是dump命令的逆过程，用于还原dump命令生成的备份文件。倾倒操作可用来备份文件，而还原操作则是写回这些已备份的文件。

RESTORE命令执行dump(8)的逆功能。文件系统的完全备份可以被恢复，随后的增量备份可以在其之上分层。可以通过全部或部分备份恢复单个文件和目录子树。还原可以跨网络工作；要做到这一点，请参阅下面描述的-f标志。命令的其他参数是指定要还原的文件名或目录名。除非指定了-h标志(见下文)，目录名的外观是指该目录的文件和(递归)子目录。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
restore -C [-cdHklMvVy] [-b blocksize] [-D filesystem] [-f file] [-F script] [-L limit] sh
restore -i [-acdHklmMNouvVy] [-A file] [-b blocksize] [-f file] [-F script] [-Q file]
restore -P file [-acdHklmMNuvVy] [-b blocksize] [-f file] [-F script] [-s fileno] [-T di
restore -R [-cdHklMNuvVy] [-b blocksize] [-f file] [-F script] [-s fileno] [-T director
restore -r [-cdHklMNuvVy] [-b blocksize] [-f file] [-F script] [-s fileno] [-T directory]
restore -t [-cdhHklMNuvVy] [-A file] [-b blocksize] [-f file] [-F script] [-Q file] [-s
restore -x [-adchHklmMNouvVy] [-A file] [-b blocksize] [-f file] [-F script] [-Q file] [
```

模式

```
sh
-C          # 此模式允许比较转储中的文件。还原读取备份并将其内容与磁盘上的文件进行比较。它首先将其
-i          # 此模式允许从转储中交互式地恢复文件。在从转储中读取目录信息之后，RESTORE提供了一个类
# add [arg]，将当前目录或指定的参数添加到要提取的文件列表中。如果指定了目录，则将其
# cd arg, Arg将当前工作目录更改为指定的参数。
# delete [arg]，从要提取的文件列表中删除当前目录或指定的参数。如果指定了目录，则从
# extract，提取列表中的所有文件都是从dump中提取的。还原将询问用户希望挂载哪个卷。提
# help，列出可用命令的摘要。
# list [arg]，列出当前或指定的目录。作为目录的条目附加了“/”。标记为提取的条目在前面
# pwd，打印当前工作目录的完整路径名。
# quit，RESTORE立即退出，即使提取列表不是空的。
# setmodes，添加到提取列表中的所有目录都有它们的所有者、模式和时间设置；没有从转储中
# verbose，切换-v标志的感觉。设置时，详细标志将导致ls命令列出所有条目的inode编号，这
# 还原从现有转储文件中创建新的快速文件访问文件，而不还原其内容
-P file    # RESTORE请求一个多卷集的特定磁带，在该磁带上重新启动完全恢复(请参阅下面的-r标志)。
-R          # 恢复(重建)文件系统。目标文件系统应该以mke2fs(8)作为原始文件系统，并在恢复初始0级备
-r          # - mke2fs /dev/sda1
# - mount /dev/sda1 /mnt
# - cd /mnt
# - restore rf /dev/st0
# 注意，restore会在根目录中留下一个文件restoresymable，以便在增量恢复传递之间传递信
-t          # 如果指定文件发生在备份上，则列出指定文件的名称。如果没有给出文件参数，则列出根目录
-x          # 命名文件从给定媒体中读取。如果指定的文件匹配内容位于备份上且未指定-h标志的目录，则
```

选项

```
sh
-a          # 在-i或-x模式下, RESTORE要求用户提供要提取的文件的卷号(目的是通过只读取有趣的卷来
-A archive_file    # 从ARCHIVE_FILE而不是媒体读取目录。这个选项可以与-t、-i或-x选项结合使用, 这样就可
-b blocksize       # 每个转储记录的千字节数。如果未指定-b选项, 还原将尝试动态确定媒体块大小
-c          # 通常, RESTORE将尝试动态地确定转储是从旧的(预-4.4)文件系统还是从新的格式文件系统生
-d          # 打印调试信息
-D filesystem     # -D标志允许用户在使用RESTORE和-C选项检查备份时指定文件系统名称。
-f file        # 从文件中读取备份; 文件可能是一个特殊的设备文件, 如/dev/st0(磁带驱动器)、/dev/sd
-F script      # 在每个磁带的开头运行脚本。设备名称和当前卷号将在命令行上传递。如果恢复应该继续而不
-h          # 提取实际目录, 而不是它引用的文件。这将防止从转储中恢复完整子树的层次结构。
-H hashsize     # 使用具有指定数目的条目的哈希表来存储目录条目, 而不是链接列表。这个哈希表将大大加快
-k          # 与远程磁带服务器联系时使用Kerberos身份验证。(只有在编译还原时启用此选项时才可用。
-l          # 在进行远程还原时, 假设远程文件是常规文件(而不是磁带设备)。如果要还原远程压缩文件,
-L limit       # -L标志允许用户在使用RESTORE和-C选项检查备份时指定最大错误比较数。如果达到此限制,
-m          # 按inode编号而不是按文件名提取。如果只提取几个文件, 并且希望避免重新生成文件的完整
-M          # 启用多卷功能(用于读取使用-M选项转储的转储)。使用-f指定的名称被视为前缀, 并尝试按
-N          # -N标志使RESTORE按照-i, -R, -r, t或x命令之一的请求执行完全执行, 而不实际在磁盘上
-o          # -o标志使RESTORE自动恢复当前目录权限, 而无需询问操作符是在-i或-x模式中这样做。
-Q file        # 使用文件, 以便在-i, -x或-t模式中使用转储快速文件访问模式读取磁带位置。建议将st驱动
-s fileno      # 从多文件磁带上指定的文件读取。文件编号从1开始
-T dir         # -T标志允许用户指定用于存储临时文件的目录。默认值为/tmp。此标志在从软盘启动后恢复文
-u          # 在创建某些类型的文件时, 如果目标目录中已经存在RESTORE, 则RESTORE可能会生成警告诊
-v          # 正常情况下, RESTORE会默默地完成它的工作。-v(详细)标志导致它键入它所处理的每个文件
-V          # 启用读取多卷非磁带介质(如cdrom)。
-X filelist    # 读取除命令行中指定的文件外, 还要从文本文件列表中列出或提取的文件列表。这可以与-t或
-y          # 不要问用户是否在发生错误时中止还原。始终尝试跳过坏块并继续
--tmpdir=[dir]  # 指定临时文件的路径, 如果tmpdir后面没有路径, 那么临时文件创建在/tmp目录下
TEMPLATE      # 临时文件名, 名字中必须包含至少3个字母X。如果没有指定, 那么默认是tmp. XXXXXXXXXX

--help          # 显示帮助文档
--version       # 显示命令版本信息
```

说明

如果得到读取错误, 则进行抱怨。如果指定了y, 或者用户响应y, RESTORE将尝试继续还原。如果使用多个磁带卷进行备份, 还原将在安装下一个卷时通知用户。如果指定了-x或-i标志, RESTORE还将询问用户希望挂载哪个卷。提取几个文件的最快方法是从最后一个卷开始, 然后开始第一个卷。

有许多一致性检查可以通过RESTORE列出。大多数检查都是不言自明的, 或者“永远不会发生”。常见错误如下:

1. Converting to new file system format, 从旧文件系统创建的转储磁带已经加载。它将自动转换为新的文件系统格式。
2. : not found on tape, 指定的文件名列在磁带目录中, 但没有在磁带上找到。这是由于查找文件时磁带读取错误以及使用在活动文件系统上创建的转储磁带造成的。
3. expected next file , got , 出现了一个未在目录中列出的文件。这可能发生在使用在活动文件系统上创建的转储时。
4. Incremental dump too low, 在执行增量恢复时, 加载了在上一次增量转储之前写入的转储, 或者加载了增量级别过低的转储。
5. Incremental dump too high, 当执行增量恢复时, 当先前的增量转储停止时, 或者加载的增量级别过高时, 不开始覆盖的转储。
6. Tape read error while restoring
7. Tape read error while skipping over inode

8. Tape read error while trying to resynchronize, 已发生磁带(或其他媒体)读取错误。如果指定了文件名, 其内容可能是部分错误。如果正在跳过inode或磁带试图重新同步, 则未将提取的文件损坏, 但可能无法在磁带上找到文件。
9. resync restore, skipped blocks, 转储读取错误后, 还原可能必须重新同步自身。此消息列出跳过的块数。

退出码

"restore"退出, 成功时状态为零。磁带错误用退出代码1表示。在对转储文件进行比较时, 退出代码为2表示自转储后某些文件已被修改或删除。

环境变量

如果存在以下环境变量, 则restore将使用该变量:

1. TAPE, 如果没有指定"-f"选项, restore将使用通过磁带指定的设备作为转储设备。TAPE的形式可以是 tapename、host: tapename或user@host: tapename。
2. TMPDIR, TMPDIR中给出的目录将用于存储临时文件, 而不是"/tmp"。
3. RMT, 环境变量RMT将用于确定远程RMT(8)程序的路径名。
4. RSH, RESTORE使用这个变量的内容来确定在进行网络恢复时要使用的远程shell命令的名称(rsh、ssh等)。如果未设置此变量, 则将使用rcmd(3), 但只有root用户才能进行网络恢复。

文件

- /dev/st0 默认磁带设备。
- /tmp/rstdir 包含磁带上目录的文件。
- /tmp/rstmode* 目录的所有者、模式和时间戳。
- ./restoresymtable 增量还原之间传递的信息。

举例

```
dump -9 -u -f /dev/hda3 /home/frank/
```

sh

用restore命令来恢复备份:

```
restore rf /dev/hda3 /home/frank
```

sh

用restore命令来查看备份文件里的文件列表:

```
restore ft /dev/hda3
```

sh

rm - 删除文件和目录

默认情况下不会删除目录。

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。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
rm [选项] file
```

sh

选项

```
-f, --force          # 强制执行, 不交互
-i                  # 交互模式, 每删除一个文件都要询问
-I                  # 在删除三个以上的文件之前, 或者在递归删除之前,
                   # 提示一次。“-I”的侵扰性不如“-i”，
                   # 但仍能免受大多数错误的侵扰。
--interactive[=WHEN]
--one-file-system
--no-preserve-root
--preserve-root
-r, -R, --recursive
-v, --verbose        # 显示详细执行过程

--help               # 显示帮助文档
--version            # 显示命令版本信息
```

sh

补充说明

默认情况下，`rm` 不删除目录。使用 `--recursive` (`-r` or `-R`) 选项也可以删除每个列出的目录及其所有内容。要删除名称以 `-` 开头的文件，例如 `-foo`，请使用以下命令之一：

```
rm -- -foo  
rm ./-foo
```

sh

注意，如果使用rm删除文件，通常可以恢复该文件的内容。如果您想要更多的保证内容是真正不可恢复的，请考虑使用shred。

举例

```
[sogrey@bogon 文档]$ ls  
backup demos test test2.txt test3.txt test4.txt test.txt  
[sogrey@bogon 文档]$ rm test*  
rm: 无法删除"test": 是一个目录  
[sogrey@bogon 文档]$ ls  
backup demos  
[sogrey@bogon 文档]$ rm -rf test/  
[sogrey@bogon 文档]$ ls  
backup demos  
[sogrey@bogon 文档]$
```

sh

rmdir - 用来删除空目录

删除一个空目录，可以同时删除途经的父目录，但是要确保父目录中没有其他内容。

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佬表示目录名。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命令全部，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。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
rmdir [选项] dir
```

sh

选项

```
--ignore-fail-on-non-empty # 忽略由非空目录造成错误信息  
-p, --parent               # 删除目录，以及途经的父目录  
-v, --verbose              # 显示详细信息  
  
--help                      # 显示帮助文档  
--version                  # 显示命令版本信息
```

sh

举例

将工作目录下，名为 www 的子目录删除：

```
rmdir www
```

sh

在工作目录下的 www 目录中，删除名为 Test 的子目录。若 Test 删除后，www 目录成为空目录，则 www 亦予删除。

```
rmrdir -p www/Test
```

sh

下面命令等价于 `rmrdir a/b/c` , `rmrdir a/b` , `rmrdir a`

```
rmrdir -p a/b/c
```

sh

简单删除一个目录

```
[sogrey@bogon DirTest]$ ls -l # 查看当前目录
总用量 8
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
[sogrey@bogon DirTest]$ ls -l Dir1 # 查看已有目录Dir1子文件和子目录, 不为空
总用量 0
-rw----- 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw----- 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw----- 1 sogrey sogrey 0 3月 9 00:44 3.c
[sogrey@bogon DirTest]$ rmrdir Dir1 # 尝试删除非空目录失败
rmrdir: 删除 "Dir1" 失败: 目录非空
[sogrey@bogon DirTest]$ mkdir DirTmp #创建临时空目录
[sogrey@bogon DirTest]$ ls -l #查看临时空目录DirTmp已创建
总用量 12
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
drwx----- 2 sogrey sogrey 4096 3月 28 23:33 DirTmp
[sogrey@bogon DirTest]$ rmrdir DirTmp #删除空目录
[sogrey@bogon DirTest]$ ls -l #查看删除成功
总用量 8
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
[sogrey@bogon DirTest]$
```

sh

使用-p选项, 删除子目录以及途经的父目录, 父目录只有当前的一个子目录

```
[sogrey@bogon DirTest]$ mkdir -p pDir/cDir #使用-p选项创建目录pDir以及其子目录cDir
[sogrey@bogon DirTest]$ ll
总用量 12
drwx----- 2 sogrey sogrey 4096 3月 28 23:38 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
drwx----- 3 sogrey sogrey 4096 3月 28 23:39 pDir
[sogrey@bogon DirTest]$ ll pDir/
总用量 4
drwx----- 2 sogrey sogrey 4096 3月 28 23:39 cDir
[sogrey@bogon DirTest]$ ll pDir/cDir/
总用量 0
[sogrey@bogon DirTest]$ rmrdir -p pDir/cDir/ #使用-p选项删除目录cDir, 这样会导致pDir一起被删除
[sogrey@bogon DirTest]$ ll
总用量 8
drwx----- 2 sogrey sogrey 4096 3月 28 23:38 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
[sogrey@bogon DirTest]$
```

sh

使用-p选项, 删除子目录以及途经的父目录, 父目录中还有其他内容

sh

```
[sogrey@bogon DirTest]$ ll #先查看当前目录
总用量 8
drwx----- 2 sogrey sogrey 4096 3月 28 23:38 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
[sogrey@bogon DirTest]$ mkdir -p Dir1/cDir #为目录Dir1创建子目录cDir
[sogrey@bogon DirTest]$ ll
总用量 8
drwx----- 3 sogrey sogrey 4096 3月 28 23:43 Dir1
drwx----- 2 sogrey sogrey 4096 3月 28 23:30 Dir2
[sogrey@bogon DirTest]$ ll Dir1 #查看子目录cDir已创建
总用量 4
-rw----- 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw----- 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw----- 1 sogrey sogrey 0 3月 9 00:44 3.c
drwx----- 2 sogrey sogrey 4096 3月 28 23:43 cDir
[sogrey@bogon DirTest]$ rmdir -p Dir1/cDir/ #使用-p选项删除目录Dir1/cDir/, 父目录Dir1不为空, 删除失败
rmdir: 删除目录 "Dir1" 失败: 目录非空
[sogrey@bogon DirTest]$ ll Dir1 #但cDir成功删除
总用量 0
-rw----- 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw----- 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw----- 1 sogrey sogrey 0 3月 9 00:44 3.c
[sogrey@bogon DirTest]$
```

同时改变所有者和组

sh

```
[root@bogon DirTest]$ ls -l 1.c #当前的组是sogrey
-rwxr--r-- 1 root sogrey 0 9月 7 09:11 1.c
[root@bogon DirTest]$ chown 500: 500 1.c #把组和所有者都改为500, 注意语法, 中间有个冒号
[root@bogon DirTest]$ ls -l 1.c
-rwxr--r-- 1 user01 user01 0 9月 7 09:11 1.c
```

rmmmod - 从运行的内核中移除指定的内核模块

rmmmod命令 用于从当前运行的内核中移除指定的内核模块。执行rmmmod指令，可删除不需要的模块。Linux操作系统的内核具有模块化的特性，应此在编译核心时，务须把全部的功能都放如核心。你可以将这些功能编译成一个个单独的模块，待有需要时再分别载入它们。

rmmod指令用来卸载不需要使用的内核模块。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
rmmod [ -f ] [ -w ] [ -s ] [ -v ] [ modulename ]
```

sh

选项

-v, --verbose	# 显示详细执行过程
-f, --force	# 轻质卸载
-w, --wait	# 如果拒绝卸载，那么久等待知道模块不在使用，然后卸载
-s, --syslog	# 将错误送到syslog， 默认送到标准输出
-V, --version	# 显示命令版本信息，并且退出

sh

举例

```
[sogrey@bogon ~]$ lsmod # 查看模块, nfs没人使用
Module           Size  Used by
fuse              91880  3
xt_CHECKSUM       12549  1
ipt_MASQUERADE   12678  3
nf_nat_masquerade_ipv4    13412  1 ipt_MASQUERADE
tun               31665  1
devlink           42368  0
ip6t_rpfilter    12595  1
ipt_REJECT        12541  4
nf_reject_ipv4   13373  1 ipt_REJECT
ip6t_REJECT      12625  2
nf_reject_ipv6   13717  1 ip6t_REJECT
xt_conntrack     12760  12
ip_set            40898  0
nfnetlink         14519  1 ip_set
...
drm              397980  5 ttm,drm_kms_helper,vmwgfx
i2c_core          63151  3 drm,i2c_piix4,drm_kms_helper
libata            247431  3 pata_acpi,ata_generic,ata_piix
dm_mirror         22289  0
dm_region_hash   20813  1 dm_mirror
dm_log            18411  2 dm_region_hash,dm_mirror
dm_mod            124191  11 dm_log,dm_mirror
[sogrey@bogon ~]$
```

route - 显示并设置Linux中静态路由表

route命令 用来显示并设置Linux内核中的网络路由表，route命令设置的路由主要是静态路由。要实现两个不同的子网之间的通信，需要一台连接两个网络的路由器，或者同时位于两个网络的网关来实现。

在Linux系统中设置路由通常是为了解决以下问题：该Linux系统在一个局域网中，局域网中有一个网关，能够让机器访问Internet，那么就需要将这台机器的ip地址设置为Linux机器的默认路由。要注意的是，直接在命令行下执行route命令来添加路由，不会永久保存，当网卡重启或者机器重启之后，该路由就失效了；可以在/etc/rc.local中添加route命令来保证该路由设置永久有效。

route指令用于显示或者修改IP路由表。它的主要用途是在使用ifconfig(8)程序配置接口后，通过接口设置到特定主机或网络的静态路由。当使用add或del选项时，路由将修改路由表。如果没有这些选项，路由将显示路由表的当前内容。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
route [-CFvnee]
route [-v] [-A family] add [-net|-host] target [netmask Nm]
    [gw Gw] [metric N] [mssM] [window W]
    [irtt I] [reject] [mod] [dyn] [reinstate] [[dev] If]
route [-v] [-A family] del [-net|-host] target [gw Gw]
    [netmask Nm] [metric N] [[dev] If]
route [-V] [--version] [-h] [--help]
```

sh

选项

```
-A family          # 指定使用的地址类型
-F                # 操作内核FIB
-C                # 管理内核路由缓存
-v                # 显示详细过程
-n                # 以数字的方式显示路由表中的主机
-e                # 使用netstat的输出格式显示路由表
-net              # 指定一个网络路由
-host             # 指定一个主机路由
add              # 增加一个路由
del              # 删除一个路由
target           # 指定目标网络或者主机
netmask mask    # 添加一个路由时，使用这个子网掩码
gw ip            # 指定数据包通过的网关IP地址
metric           # 指定路由表的metric字段
window           # 指定路由表的TCP连接串口
irtt I            # 将此路由上的TCP连接的初始往返时间(Irtt)设置为I毫秒(1-12000)。这通常只在AX.25上有效。
reject           # 安装阻塞路由，这将迫使路由查找失败。例如，这用于在使用默认路由之前屏蔽网络。这通常只在AX.25上有效。
mod, dyn, reinstate # 安装动态或修改的路由。这些标志用于诊断，通常仅由路由守护进程设置。
dev eth          # 指定路由的网络接口

--help             # 显示帮助文档
--version          # 显示命令版本信息
```

输出

内核路由表的输出组织在以下列中

输出列	说明
Destination	目标网络或目标主机。
Gateway	网关地址或“*”(如果没有设置)
Genmask	目标网络的网络掩码； 主机目的地为‘255.255.255.255’， 默认路由为‘0.0.0.0’。
Flags	可能的flag有 U ， 路由已经起来 H ， 目标是一个主机 G ， 使用网关 R ， 用于动态路由的恢复路由 D ， 由守护进程动态安装或重定向 M ， 从路由守护进程或重定向中修改 A ， 由addrconf安装 C ， 缓存项**!**， 拒绝路由
Metric	与目标的距离(通常以啤酒花计)。它不是最近的内核使用的，而是路由守护进程所需要的。
Ref	引用此路由的次数。(不在Linux内核中使用。)
Use	查找路线的数量。根据-F和-C的使用情况，这将是路由缓存丢失(-F)或命中(-C)。
Iface	用于此路由的数据包将发送到该接口。
MSS	此路由上TCP连接的默认最大分段大小
Window	此路由上TCP连接的默认窗口大小
irtt	初始RTT(往返时间)。内核使用它来猜测最佳的TCP协议参数，而无需等待(可能是缓慢的)答案。
HH	引用缓存路由的硬件头缓存的ARP条目和缓存路由的数量。如果缓存路由的接口不需要硬件地址(例如lo)，则为-1。
Arp	缓存路由的硬件地址是否最新。

举例

```
route add -net 127.0.0.0
# 添加正常的回送条目，使用net掩码255.0.0.0(A级Net，从目标地址确定)并与“lo”设备相关联(假设此设备使用ifconfig)
# 10.x.x.x.通过“eth0”将路由添加到网络192.56.76.x。这里没有必要使用C类网络掩码修饰符，因为192.*是一个C类I

route add default gw mango-gw
# 添加默认路由(如果没有其他路由匹配，则使用该路由)。使用此路径的所有数据包将通过“芒果-GW”传送。实际用于该路由器

route add ipx4 sl0
# 通过SIP接口将路由添加到“IPX 4”主机(假设“IPX 4”是SILIP主机)。

route add -net 192.57.66.0 netmask 255.255.255.0 gw ipx4
# 此命令添加网络“192.57.66.x”，通过前一条路由传送到SILIP接口。

route add -net 224.0.0.0 netmask 240.0.0.0 dev eth0
# 。这将所有D类(多播)IP路由设置为通过“eth0”。这是具有多播内核的正确的正常配置行。

route add -net 10.0.0.0 netmask 255.0.0.0 reject
# 这为专用网络“10.x.x.x.”安装了一个拒绝路由。
```

显示本机路由表

```
[root@localhost ~]$ route
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref    Use Iface
255.255.255.255 -           255.255.255.255 !H      0      -          0 -
224.0.0.0       -           255.255.255.0   !       0      -          0 -
```

sh

以netstat格式显示

```
[root@localhost ~]$ route -e
Kernel IP routing table
Destination     Gateway         Genmask        Flags     MSS Window irtt Iface
255.255.255.255 -           255.255.255.255 !H      - -      - -
224.0.0.0       -           255.255.255.0   !       - -      - -
```

sh

添加一个路由

```
[root@localhost ~]$ route add -net 111.13.0.0/24 dev eth0    #指定网段使用设备eth0访问
[root@localhost ~]$ route      #查看路由表
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref    Use Iface
255.255.255.255 -           255.255.255.255 !H      0      -          0 -
111.13.0.0      *            255.255.255.0   U       0      0          0 eth0
224.0.0.0       -           255.255.255.0   !       0      -          0 -
172.16.0.0      *            255.255.0.0    U       0      0          0 eth0
```

sh

删除一个路由

```
[root@localhost ~]$ route del -net 111.13.0.0/24 #删除指定地址id路由
[root@localhost ~]$ route      #查看路由
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref    Use Iface
255.255.255.255 -           255.255.255.255 !H      0      -          0 -
224.0.0.0       -           255.255.255.0   !       0      -          0 -
172.16.0.0      *            255.255.0.0    U       0      0          0 eth0
```

sh

sar - 系统运行状态统计工具

sar命令是Linux下系统运行状态统计工具，它将指定的操作系统状态计数器显示到标准输出设备。sar工具将对系统当前的状态进行取样，然后通过计算数据和比例来表达系统的当前运行状态。它的特点是可以连续对系统取样，获得大量的取样数据。取样数据和分析的结果都可以存入文件，使用它时消耗的系统资源很小。

sar指令用来收集、报告、保存系统的活动信息。sar命令将操作系统中选定的累积活动计数器的内容写入标准输出。会计系统根据参数“interval”、“count”中的值，写入以秒为单位的指定间隔的指定次数的信息。如果参数“interval”设置为零，sar命令将显示自系统启动以来的平均统计信息。如果指定“count”参数而未指定“Interval”参数，则会连续生成报告。除了显示在屏幕上之外，还可以将收集到的数据保存在“-o”标志指定的文件中。如果省略了文件名，sar将使用标准的系统活动日数据文件“/var/log/sa/sadd”文件，其中dd参数指示当前日期。默认情况下，内核中的所有可用数据都保存在数据文件中。

sar命令提取并写入以前保存在文件中的标准输出记录。该文件可以是“-f”标志指定的文件，也可以是默认的标准系统活动日数据文件。

如果没有“-P”标志，sar命令将报告系统范围内(所有处理器中的全局统计)统计数据，这些统计数据是以百分比表示的值的平均值计算的，而以其他方式表示的和。如果给定“-P”标志，sar命令报告与指定处理器相关的活动。如果给出了“-P ALL”，sar命令会报告每个处理器的统计信息和所有处理器之间的全局统计信息。

可以使用标志选择有关特定系统活动的信息。没有指定任何标志，只选择CPU活动。指定-A标志等同于指定“-bBdqrRSvwWy -l SUM -i XALL -n ALL -u ALL -P ALL”。

sar命令的默认版本(CPU利用率报告)可能是用户开始系统活动调查的首批工具之一，因为它监视主要的系统资源。如果CPU利用率接近100%(使用“+ nice + system”)，则采样的工作负载是CPU限制的。

如果需要多个示例和多个报告，则可以方便地为sar命令指定一个输出文件。将sar命令作为后台进程运行。这方面的语法是：

```
sar -o datafile interval count >/dev/null 2>&1 &
```

所有数据以二进制形式捕获并保存到文件(数据文件)中。然后，可以使用sar命令使用-f选项选择性地显示数据。设置间隔和计数参数，以选择间隔秒间隔的计数记录。如果未设置Count参数，则将选择保存在文件中的所有记录。以这种方式收集数据对于描述一段时间内的系统使用情况和确定峰值使用时间非常有用。

注意：sar命令只报告本地活动。

适用范围

RedHat openSUSE	RHEL Fedora	Ubuntu Linux Mint	CentOS Alpine Linux	Debian Arch Linux	Deepin	SUSE
--------------------	----------------	----------------------	------------------------	----------------------	--------	------

语法

```
sar [OPTION]
```

sh

选项

选项	说明
-A	显示所有报告
-b	显示IO状态以及传输速率，有一下值可以显示： tps ，每秒发送给物理设备的传输总数以组合成对设备的单个I/O请求。转移是不确定的大小。 rtps ，每秒向物理设备发出的输入请求总数。 bread/s ，每秒从设备读取的数据总量(以块为单位)，块大小512字节。位)。
-B	显示页信息： pgpgin/s ，每秒从磁盘呼入系统的总字节数。注意：对于旧内核(2.2.x)， pgpgout/s ，系统每秒呼出到磁盘的千字节总数。注意：对于旧内核(2.2.x)，这个值是生的页面错误数(主要小错误)。这不是生成I/O的页面错误的计数，因为一些页面错误可秒发生的主要故障数，这些故障需要从磁盘加载内存页。 pgfree/s ，系统每秒放置在守护进程扫描的页数。 pgscan/s ，每秒直接扫描的页数。 pgsteal/s ，每秒从缓存(分)求。 %vmeff ，这是一个页面回收效率的度量(pgsteal / pgscan)。如果它接近100%被捕获。如果它太低(例如，不足30%)，那么虚拟内存就会有一些困难。如果在时间间隔
-C	读取文件的时候，显示备注信息
-d	显示块设备信息，有以下值可以显示： tps ，指示每秒发送给设备的传输次数。多个逻不确定的大小。 rd_sec/s ，从设备读取的扇区数。扇区的大小为512字节。 wr_sec/s ， avgrq-sz ，向设备发出的请求的平均大小(按扇区)。 avgqu-sz ，向设备发出的请求的请求的平均时间(毫秒)。这包括请求在队列中花费的时间和服务它们的时间。 svctm ， %util ，向设备发出I/O请求的CPU时间百分比(设备的带宽利用率)。当此值接近100%时
-e [hh:mm:ss]	设置报告的结束时间
-f	从文件获取信息
-h	显示简短的帮助信息
-i	在秒内选择数据记录，以尽可能接近由Interval参数指定的数字。
-I { int [...] SUM ALL XALL }	报告给定中断的统计信息。 int 是中断号。在命令行中指定多个INT参数将查看多个独立总数。 ALL 关键字表示将报告来自前16个中断的统计信息，而 XALL 关键字表示将报告源。注意，中断统计数据依赖于要收集的南共体选项“-S INT”。
-j { ID LABEL PATH UUID ... }	显示持久设备名称。结合选项-d使用此选项。选项ID、LABEL等。指定持久性名称的类是“/dev/disk”中存在具有所需持久名称的目录。如果没有为设备找到持久名称，则设备
--legacy	启用读取旧的“/var/log/sa/sadd”数据文件
-m	报告电源管理信息
-n	报告网络信息，可能的关键字有 DEV, EDEV, NFS, NFSD, SOCK, IP, EIP, ICMP, EICMP, TCP, ETCP, UDP 。

选项	使用 DEV 关键字，报告不同内核缓冲的统计信息。显示下列值: IFACE ， 报告统计数据包总数。 txpck/s ， 每秒发送的数据包总数。 rxkB/s ， 每秒接收的字节总数 txkB 的压缩数据包数量(用于跳频等) txcmp/s ， 每秒传输的压缩数据包数。 rxmcst/s ， 每秒接收的坏数据包总数。 txerr/s ， 发送数据包时每秒发生的错误总数。 coll/s ， 传输 Linux 缓冲区中缺少空间，接收时每秒丢弃的数据包数量。 txdrop/s ， 由于Linux缓冲区 txcarr/s ， 在传输数据包时每秒发生的carrier错误。 rxfram/s ， 每秒在接收到的数据包接收数据包上的FIFO溢出错误数 txfifo/s ， 每秒发生在传输数据包上的FIFO溢出错误数。
	使用 EDEV 关键字，将报告来自网络设备的故障(错误)统计信息。显示下列值: IFACE 秒接收的坏数据包总数。 txerr/s ，发送数据包时每秒发生的错误总数。 coll/s ，传输 Linux 缓冲区中缺少空间，接收时每秒丢弃的数据包数量。 txdrop/s ，由于Linux缓冲区 txcarr/s ，在传输数据包时每秒发生的carrier错误。 rxfram/s ，每秒在接收到的数据包接收数据包上的FIFO溢出错误数 txfifo/s ，每秒发生在传输数据包上的FIFO溢出错误数。
	使用 NFS 关键字，将报告有关NFS客户端活动的统计信息。显示下列值: call/s ，每秒数量，需要重新传输的请求(例如，由于服务器超时)。 read/s ，每秒进行的“read”RPC调用数。 access/s ，每秒进行的“access”RPC调用数。 getatt/s ，每秒进行的“getattr”RPC调用数。
	使用 NFSD 关键字，将报告有关NFS服务器活动的统计信息。显示下列值: scall/s ，全局错误rpc请求数，这些请求的处理会产生错误。 packet/s ，每秒接收的网络数据包数。接收的tcp数据包数。 hit/s ，每秒应答缓存命中次数。 miss/s ，每秒未命中应答缓存的 swrite/s ，每秒接收的“write”RPC调用数。 saccess/s ，每秒接收的“access”RPC调用数。
	使用 Sock 关键字，将报告正在使用的套接字的统计信息(IPv 4)。显示下列值: totsock 套接字数目 udpsck ，当前使用的UDP套接字数目 rawsck ，当前使用的RAW套接字数目 time 等待状态下TCP套接字的数目。
	使用 IP 关键字，报告有关IPv 4网络流量的统计信息。请注意，IPv 4统计信息依赖于要报告的实体(方括号内有正式SNMP名称): irec/s ，每秒从接口接收的输入数据报总数，包括错误接收的数据报的数量，该实体不是该实体的最终IP目的地，因此试图找到将其转发到该最终目的地的IP用户协议(包括ICMP)的输入数据报总数[ipInDelivers]。 orq/s ，本地IP用户协议(包括ICMP)的输出数据报总数[ipOutRequest]。请注意，此计数器不包括以fwddgm/s计算的任何数据报。 asmrq/s ，成功重新组装的IP数据报的数量[ipReasmReqds]。 asmok/s ，每秒成功重新组装的IP数据报的数量[ipReasmOKs]。 frag/s ，由于IP数据报被丢弃(例如，由于缺乏缓冲区空间)[ipFragOKs]。 fragcrt/s ，由于该实体[ipFragCreates]的碎片而每秒生成的IP数据报。
	使用 EIP 关键字，将报告有关IPv 4网络错误的统计信息。请注意，IPv 4统计信息依赖于要报告的实体(方括号内有正式SNMP名称): ihdrerr/s ，由于IP报头中的错误而每秒丢弃的输入数据报的错误、超时时间、在处理IP选项时发现的错误等。 [ipinHdrError] iadrerr/s ，每秒丢弃的IP地址不是要在此实体接收的有效地址。此计数包括无效地址(例如，0.0.0.0)和不支持的协议而成功接收到但每秒丢弃的本地寻址数据报的数目[ipInUnKnownProtos]。 iperr/s ，每秒遇到任何问题来阻止其继续处理，但这些数据报被丢弃(例如，由于缺乏缓冲区空间)[ipInDiscards]。 odisc/s ，每秒输出IP数据报的数量，没有遇到任何问题来阻止它们传输到目的地(例如，由于缺乏缓冲区空间)[ipOutDiscards]。请注意，如果任何此类数据包符合此(任意)丢弃标准，则此计数器不包括它们。 onorm/s ，每秒丢弃的IP数据报的数量，因为找不到将它们传输到目的地[ipOutNoRoutes]的路由。 otm/s ，每秒丢弃的IP数据报的数量，因为它们不符合“无路由”标准的任何数据包。请注意，这包括主机无法路由的任何数据报，因为其所报告的路由不可达。 frag/s ，每秒丢弃的IP数据报的数量，因为它们在目的地中被碎片化，但不能这样做。
	使用 ICMP 关键字，报告关于ICMPv 4网络流量的统计信息。请注意，ICMPv 4的统计信息依赖于要报告的实体(方括号内有正式SNMP名称): imsg/s ，实体每秒收到的ICMP消息总数[icmpInMsgs]。 iech/s ，每秒接收的ICMP Echo(请求)消息的数量[icmpInEchos]。 oech/s ，每秒发送的ICMP回声(请求)消息数量[icmpOutEchos]。 oecho/s ，每秒发送的ICMP回声(响应)消息数量[icmpOutEchoRep]。 itm/s ，每秒收到的ICMP时间戳(请求)消息数目[icmpInTimestampReq]。 otm/s ，每秒发送的ICMP时间戳(请求)消息数量[icmpOutTimestampReq]。 odrmk/s ，每秒收到的ICMP地址掩码请求消息的数量[icmpInAddrMaskReq]。 oadrmk/s ，每秒发送的ICMP地址掩码请求消息的数量[icmpOutAddrMaskReq]。 oadrmkr/s ，每秒发送的ICMP地址掩码应答消息的数量[icmpOutAddrMaskRep]。
	使用 EICMP 关键字，将报告有关ICMPv 4错误消息的统计信息。请注意，ICMPv 4的统计信息依赖于要报告的实体(方括号内有正式SNMP名称): ierr/s ，实体每秒收到但被确定为具有ICMP特性的错误消息的数量[icmpinError] oerr/s ，该实体由于在ICMP中发现的问题而没有发送的每秒ICMP消息的次数[icmpOutError]。 odstunr/s ，每秒发送的ICMP目标不可达消息的数量[icmpOutDestUnreachs]。 odtunr/s ，每秒发送的ICMP时间戳超时消息的数量[icmpOutTimeExcds]。 itmex/s ，ICMP时间超过每秒收到的消息的次数[icmpInTimeExcds]。 iparmpb/s ，每秒收到的ICMP参数问题消息数量[icmpInParmProbs]。 isrcq/s ，每秒收到的ICMP源Quench消息的数量[icmpInSrcQuenbergs]。 iredir/s ，每秒接收到的ICMP重定向消息的数量[icmpInRedirects]。 icmd/s ，ICMP重定向消息的数目[icmpOutRedirects]。

选项	ICMP里走向待办的数里[icmpOutRecreations]。	说明
	使用 TCP 关键字，将报告有关TCPv 4网络流量的统计信息。请注意，TCPv 4统计信息值(方括号内有正式SNMP名称): active/s ，tcp连接每秒从关闭状态直接转换到SYN发送;从每秒侦听状态直接转换到SYN-RCVD状态的次数[tcpPassiveOpens] iseg/s ，每秒接收数包括在当前建立的连接上接收的段。 oseg/s ，每秒发送的段数，包括当前连接上的段数。	
	使用 ETCP 关键字，将报告有关TCPv 4网络错误的统计信息。请注意，TCPv 4统计信息值(方括号内有正式SNMP名称): atmptf/s ，每秒TCP连接的次数已从SYN发送状态或SYN连接的次数已从SYN-RCVD状态直接转换到侦听状态[tcpAttemptFails]。 estres/s **，**等待状态[tcpEstabResets]直接转换到关闭状态。 retrans/s ，每秒重传的段总数，即包含TCP段数。 isegerr/s ，每秒接收的错误段总数(例如，错误的TCP校验和)[tcpInErrs] ors1 段数。	
	使用 UDP 关键字，报告有关UDPV 4网络流量的统计信息。注意，UDPV 4统计信息依赖(方括号内有正式SNMP名称): idgm/s ，每秒发送给UDP用户的UDP数据报总数[udpln 报总数[udpOutDatagram] noport/s ，在目标端口[udpNoport]上没有应用程序的每秒接收到的UDP数据报数量，由于目的地端口缺少应用程序[udplnError]而无法传递。	
	使用 SOCK6 关键字，将报告正在使用的套接字的统计信息(IPv 6)。请注意，IPv 6统计数据依赖于列值: tcp6sck ，目前正在使用的TCPv6套接字的数量。 udp6sck ，目前正在使用的UDRawv6套接字的数量 ip6-frag ，目前正在使用ipv6碎片的数量	
	使用 IP6 关键字，报告有关IPv 6网络流量的统计信息。请注意，IPv 6统计数据依赖于要内有正式SNMP名称): irec6/s ，每秒从接口接收的输入数据报总数，包括错误接收的实体接收并转发到其最终目的地的每秒输出数据报数[ipv6fStatsOutForwDatagram]。 icm6/s ，本地ICMP的数据报总数[ipv6fStatsInDelivers]。 orq6/s ，本地IPv 6用户协议(包括ICMP)每秒[ipv6fStatsOutRequest]。 asmrq6/s ，每秒接收到的需要在此接口重新组装的IPv 6片断数。 imcpck6/s ，接口每秒接收到的多播数据包数。 omcpck6/s ，通过接口[ipv6fStatsOutMcastPkts]每秒发送的多播数据包数。 fragok6/s ，本地生成的IP数据报的数量[ipv6fStatsOutFragOKs] fragcr6/s ，由于在此输出接口[ipv6fStatsOutFragCreates]	
	使用 EIP6 关键字，报告有关IPv 6网络错误的统计信息。请注意，IPv 6统计数据依赖于号之间的形式SNMP名称) ihdrer6/s ，由于IPv 6标头中的错误而每秒丢弃的输入数据报出、在处理IPv 6选项时发现的错误等。[ipv6fStatsInHdrError]。 iadrer6/s ，每秒丢弃的段中的IPv 6地址不是要在此实体接收的有效地址。此计数包括无效地址(例如: 0)和不支持IPv 6路由器，因此不转发数据报的实体，此计数包括丢弃的数据报，因为目标地址不可达，由于未知或不受支持的协议而成功接收但每秒丢弃的本地寻址数据报的数目[ipv6fStatsInTruncatedPkts]。 idi6/s ，没有遇到任何问题来阻止它们继续处理，但这些数据报被丢弃(例如，由于缺乏缓冲区空间)的输入数据报的数量，因为找不到将它们传输到其目的地[ipv6fStatsInNoRoutes]的路由。 if6/s ，本地生成的IP数据报的数量，因为找不到将它们传输到目的地的路由[unknown form]。 if6/s ，每秒检测到的故障数(无论出于什么原因：超时、错误等)。[ipv6fStatsReasmFails]。 frag16/s ，要在输出接口上被碎片化，但不能是[ipv6fStatsOutFragFails]。 itrpck6/s ，由于数据报数[ipv6fStatsInTruncatedPkts]	
	使用 ICMP6 关键字，报告了有关ICMPv 6网络流量的统计信息。请注意，ICMPv 6的统计以下值(方括号内有正式SNMP名称): imsg6/s ，接口每秒接收的ICMP消息总数，其中 omsg6/s ，此接口每秒试图发送的ICMP消息总数[ipv6fIcmpOutMsgs] iech6/s ，接口每秒[ipv6fIcmpInEchos] iechr6/s ，接口每秒接收的ICMP回波回复消息的数量[ipv6fIcmpInEchoes]。 igmbq6/s ，接口每秒接收到的ICMP组成员资格查询[ipv6fIcmpInGroupMembQueries] igmr6/s ，接口每秒接收到的ICMPv 6组成员资格响应[ipv6fIcmpInGroupMembResponse]。 ogmr6/s ，每秒发送的ICMPv 6组成员资格响应[igmr6/s]，接口每秒接收到的ICMPv 6组成员减少消息的数量[ipv6fIcmpOutGroupMembReductions] irtsol6/s ，ICMP路由器每秒[ipv6fIcmpInRouterSolrice]。 ortsol6/s ，每秒由接口发送的ICMP路由器请求消息的数量[ipv6fIcmpInRouterAdvertisements]。 inbsol6/s ，每秒接收到的ICMP邻居广告消息数量[ipv6fIcmpInNeighborSolrice]。 onbsol6/s ，接口每秒发送的ICMP邻居请求消息的数量[ipv6fIcmpInNeighborAdvertisements]。 oabsol6/s ，接口每秒接收到的ICMP邻居广告消息的数量[ipv6fIcmpOutNeighborAdvertisements]。	
	使用 EICMP6 关键字，报告有关ICMPv 6错误消息的统计信息。请注意，ICMPv 6的统计以下值(方括号内有正式SNMP名称)。 errs6/s 接口每秒收到但被确定为来自ICMPv 6的错误消息数。	

选项	以下值(方括号内有正负号的为可选): errors , 接口每秒收到的ICMP错误数量[ipv6ICMPInErrors]或ICMP目标不可达消息的数量[ipv6ICMPInError] idtunr6/s , 接口每秒接收到的ICMP目标不可达消息的数量[ipv6ICMPInDestUnreachs]。 itmex6/s , ICMP时间超过[ipv6ICMPInTimeExcds]。 otmex6/s , ICMP时间超过接口每秒发送的消息的次数[ipv6ICMPInParmProblems]。 oprmpb6/s , 接口每秒接收到的ICMP参数问题消息的数量[ipv6ICMPInParmProblems]。 iredir6/s , 每秒由接口接收的重定向消息的数量[ipv6ICMPOutRedirections]。 ipck2b6/s , 接口每秒接收到的ICMP数据包的数量[ipv6ICMPOutPktTooBigs]。
	使用 UDP6 关键字, 报告有关UDPV6网络流量的统计信息。请注意, UDPV6的统计数值(方括号之间的形式SNMP名称): idgm6/s , 每秒传递给UDP用户的UDP数据报总数[udpOutDatagram]。 noport6/s , 在目标端口上没有应用程序的每秒接收到的UDP数据报数量, 由于目的地端口缺少应用程序而无法传递[udplnError]
-o	保存内容
-P	报告cpu使用情况
-p	漂亮的打印设备名称。结合选项-d使用此选项。默认情况下, 名称被打印为dev m-n, 其中m和n是设备名。
-q	报告队列长度和负载平均值: runq-sz , 运行队列长度(等待运行时的任务数) plist-sz , 系统负载平均值。负载平均值计算为可运行或正在运行的任务的平均数量(R状态), 以及在数。 ldavg-5 , 在过去5分钟系统平均负荷。 ldavg-15 , 在过去15分钟系统平均负荷。
-r	报告内存使用情况: kbmemfree , 可用内存的数量(千字节)。 kbmemused , 使用内存的内核使用的内存。 %memused , 使用内存的百分比。 kbbuffers , 内核用作缓冲区的数据的内存量(以千字节为单位) kbcommit , 当前工作负载所需的以千字节为单位的内存耗尽的估计。 %commit , 当前工作负载所需内存占内存总量(RAM交换)的百分比。这表示内存。
-R	报告内存统计。显示下列值 frmpg/s , 系统每秒释放的内存页数。负值表示系统分配的或8kB。 bufpg/s , 系统每秒用作缓冲区的附加内存页数。负值意味着系统用作缓冲区存页数。负值意味着缓存中的页面减少。
-s [hh:mm:ss]	设置数据起始时间
-S	报告交换空间利用率统计数据: kbswpfree , 自由交换空间的数量(以千字节为单位) kbswpused , Percentage of used swap space. kbswpcad , 缓存交换内存的数量(以被交换回来, 但仍然在交换区(如果需要内存, 就不需要再交换了, 因为它已经在交换区)相对于使用交换空间数量的百分比)
-t	从每日数据文件读取数据时, 指示sar应在数据文件创建者的原始区域设置时间内显示时间戳。
-u [ALL]	报告CPU利用率。ALL关键字指示应该显示所有CPU字段。报告可以显示以下字段: %user , 用户利用率百分比。请注意, 此字段包括运行虚拟处理器所花费的时间。 %usr , 在用户级别, 此字段不包括运行虚拟处理器所花费的时间。 %nice , 在具有良好优先级的用户级别, 此字段不包括运行虚拟处理器所花费的时间。 %system , 在系统级(内核)执行时出现的CPU利用率百分比。请注意, 此字段包括用于服务硬件和软件中断的CPU利用率百分比。 %idle , CPU或CPU用于服务硬件或软件中断的时间百分比。 %io , 系统有未执行的磁盘I/O请求。 %steal , 在虚拟机管理程序为另一个虚拟处理器服务时比**%irq**, CPU或CPU用于服务硬件中断的时间百分比。 %soft , CPU或CPU用于服务软件中断的时间百分比。 %idle , CPU或CPU空闲且系统没有未执行磁盘I/O请求的时间百分比。
-v	报告inode、file和其他内核表的状态。显示下列值: dentunusd , 目录缓存中未使用的inode数。 inode-nr , 系统使用的inode处理程序的数量。 pty-nr , 系统使用的伪终端数量。
-V	打印命令版本信息, 并且退出
-w	报告任务创建和切换情况
-W	报告交换空间情况: pswpin/s , 系统每秒输入的交换页总数 pswpout/s , 系统每秒输出的交换页数。
	报告tty设备情况, 显示以下值: rcvin/s , 当前串行线路每秒接收中断的次数。在TTY3上显示: rcvif/s , 从串行线接收的字符数。在TTY4上显示: txif/s , 从串行线发送的字符数。

-y 选项	反达中断的次数。 tramerr/s ， 当前串行线路每秒帧错次数。 pntyerr/s ， 当前串行线每秒中断次数。 ovrun/s ， 当前串行线路每秒溢出错误数	说明

文件

- /var/log/sa/sadd，指示每日数据文件，其中“dd”参数是表示月份中的某一天的数字。
- /proc，包含具有系统统计信息的各种文件。

举例

```
sh
sar -u 2 5 # 每2秒报告CPU利用率。显示5行。
sar -I 14 -o int14.file 2 10 # 每2秒报告IRQ 14的统计数据。显示10行。数据存储在一个名为“int14.file”的文
sar -r -n DEV -f /var/log/sa/sa16 # 显示存储在每日数据文件“sa16”中的内存和网络统计数据
sar -A # 显示当前每日数据文件中保存的所有统计信息。
```

显示cpu使用情况

```
sh
[root@localhost ntop-4.0.1]$ sar -P ALL          # 显示所有cpu使用情况
Linux 2.6.32-431.el6.i686 (localhost.localdomain)  2018年10月10日 _i686_ (1 CPU)
09时00分01秒    CPU    %user    %nice   %system   %iowait   %steal   %idle
09时10分01秒    all    0.16    0.00     0.07     0.02     0.00    99.75
09时10分01秒      0    0.16    0.00     0.07     0.02     0.00    99.75

09时10分01秒    CPU    %user    %nice   %system   %iowait   %steal   %idle
09时20分01秒    all    0.13    0.00     0.09     0.03     0.00    99.75
09时20分01秒      0    0.13    0.00
```

显示网络使用情况

```
[root@localhost ntop-4.0.1]$ sar -n SOCK -s 15:00:00 -e 16:20:00      # 显示网路中socket使用状态, 设 sh
Linux 2.6.32-431.el6.i686 (localhost.localdomain) 2018年08月10日 _i686_ (1 CPU)

15时00分01秒 totsck tcpsck udpsck rawsck ip-frag tcp-tw
15时10分01秒 707      5       6       0       0       0
15时20分01秒 723      5       6       0       0       0
平均时间:    715      5       6       0       0       0

15时24分21秒      LINUX RESTART

15时30分01秒 totsck tcpsck udpsck rawsck ip-frag tcp-tw
15时40分01秒 702      5       6       0       0       0
15时50分01秒 698      5       6       0       0       0
16时00分01秒 698      5       6       0       0       0
16时10分01秒 717      5       6       0       0       0
平均时间:    704      5       6       0       0       0

15时00分01秒 totsck tcpsck udpsck rawsck ip-frag tcp-tw
15时10分01秒 797      19      11      0       0       0
15时20分01秒 801      19      11      0       0       0
15时30分01秒 797      19      11      0       0       0
15时40分01秒 775      16      11      0       0       0
15时50分02秒 775      16      11      0       0       0
16时00分01秒 775      16      11      0       0       0
16时10分01秒 775      16      11      0       0       0
平均时间:    785      17      11      0       0       0

[root@localhost ntop-4.0.1]$
```

sed - 功能强大的流式文本编辑器

sed是一种流编辑器，用来从输入流中读取内容并完成转换，输入流可以来自一个文件，也可以来自一个管道。

sed 是一种流编辑器，它是文本处理中非常重要的工具，能够完美的配合正则表达式使用，功能不同凡响。处理时，把当前处理的行存储在临时缓冲区中，称为“模式空间”（pattern space），接着用sed命令处理缓冲区中的内容，处理完成后，把缓冲区的内容送往屏幕。接着处理下一行，这样不断重复，直到文件末尾。文件内容并没有改变，除非你使用重定向存储输出。Sed主要来自动编辑一个或多个文件；简化对文件的反复操作；编写转换程序等。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sed [options] 'command' file(s)
sed [options] -f scriptfile file(s)
```

sh

选项

```
-n, --quite, --client          # 静默模式
-e, --expression=script        # 给指令添加脚本
-f, --file=script-file         # 将文件内容作为脚本，追加给指令
--follow-symlinks              # 处理到位时遵循符号链接；硬链接仍将被破坏。
-i[SUFFIX], --in-place[=SUFFIX] # 编辑文件到位(如果提供了扩展名，则进行备份)
                                # 认的操作模式是中断符号链接和硬链接。
                                # 这可以通过跟随符号链接和复制来改变。
-c, --copy                     # 当在-i模式下对文件进行洗牌时，请使用复制而不是重命名。
                                # 虽然这将避免断开链接(符号或硬链接)，但结果的编辑操作并不是原子
                                # 这很少是想要的模式：-遵循符号链接通常就足够了，而且它更快、更安
                                # 指定每一行最大字符数，超过就自动换行
-l, --line-length=N            # 禁用所有的GNU表达式
--posix                         # 在脚本中使用扩展正则表达式。
-s                             # 将文件看作是分离的，而不是单独连续的长字符串
-u, --unbuffered                # 从输入文件中加载最少数量的数据，并更频繁地刷新输出缓冲区。

--help                          # 显示帮助文档
--version                       # 显示命令版本信息
```

sh

如果没有给出-e、--expression、-f或--file选项，那么第一个非选项参数将作为sed脚本进行解释。其余的参数都是输入文件的名称；如果没有指定输入文件，则读取标准输入。

没有地址的命令

```
:label          b和t命令的标签
#comment       注释将扩展到下一行(或-e脚本片段的末尾)。
}
```

没有地址或者一个地址的命令

```
=           打印当前的行号

a \
text        追加文本，在换行符之前有一个嵌入的反斜杠

i \
text        插入文本，在换行符之前有一个嵌入的反斜杠

q [exit-code] 立即退出sed脚本，而不处理任何更多的输入，除非自动打印没有被禁用，当前的模式空间将被打印出来。

Q [exit-code] 立即退出sed脚本，而不处理任何更多的输入。这是一个GNU扩展

r filename   附加从文件中读取的文本。

R filename   附加从文件读取的一行。命令的每次调用都从文件中读取一行。
```

接受地址范围的命令

```

{
    开始一个命令块(以)结尾)

b label    分支到标签; 如果省略标签, 则分支到脚本的末尾。

t label    如果“s///”自读取上一个输入行以来以及从最后一个t或T命令开始已成功地进行了替换, 则从“分支到标签”

T label    如果自读取上一个输入行以来, 以及自最后一个t或T命令以来, 没有“s///”已成功地进行了替换, 则从“分支

c \
text      用文本替换徐那种的行, 在换行符之前有一个嵌入的反斜杠

d          删除模式空间。开始下一个周期。

D          删除模式空间中的第一个嵌入换行符。开始下一个周期, 但如果模式空格中仍然有数据, 则跳过从输入中读取

h H        复制/追加模式空间到保持空间

g G        复制/追加保持空间到模式空间

x          交换持有空格和模式空格的内容

l          以“视觉清晰”的形式列出当前行。

l width   以“视觉清晰”的形式列出当前行, 在宽度width处将其拆分。这是一个GNU扩展。

n N        在模式空间中读取/追加下一行输入

p          打印到当前模式空间

P          打印到当前模式空间的第一个嵌入换行符

s/regexp/replacement/    尝试将regexp与模式空间匹配。如果成功, 则将该部分替换为替换部分。替换可以包含特

w filename    将当前模式空间写入文件名

W filename   将当前模式空间的第一行写入文件名。这是一个GNU扩展。

y/source/dest/    将在源中出现的模式空间中的字符音译为dst中相应的字符。

```

地址

SED命令可以在没有地址的情况下给出, 在这种情况下, 命令将对所有输入行执行; 使用一个地址, 则只对与该地址匹配的输入行执行该命令; 或者使用两个地址, 在这种情况下, 将对所有与从第一个地址开始并继续到第二个地址的包含行范围匹配的输入行执行命令。关于地址范围, 需要注意三件事: 语法是addr 1, addr 2(即地址用逗号分隔); addr 1匹配的行将始终被接受, 即使addr 2选择了前面的行; 如果addr 2是regexp, 则不会根据addr 1匹配的行对其进行测试。

在地址(或地址范围)之后, 在命令之前, 可以插入一个“!”, 这指定只有当地址(或地址范围)不匹配时才执行命令。支持以下的地址类型

- number, 只匹配指定的行号。
- first~step, 从指定的行first开始, 每step行匹配一次。
- \$, 匹配最后一行。

- `/regexp/`, 匹配正则表达式`regexp`的行。
- `\c{regexp}c`, 匹配正则表达式`regexp`的行。c可以是任何字符
- 0,addr2, 从“匹配的第一个地址”状态开始, 直到找到addr 2为止。这类似于1, addr 2, 但如果addr 2匹配输入的第一行0, addr 2表单将位于其范围的末尾, 而1, addr 2窗体仍将位于其范围的开头。这只在addr 2是正则表达式时才起作用。
- `addr1,+N`, 将匹配addr 1和addr 1后面的N行。
- `addr1,~N`, 将匹配addr 1和addr 1后面的行, 直到输入行号为N的倍数的下一行为止。

举例

```
[sogrey@bogon demo4]$ cat test.txt
eeeeee eeeee
hello world!
[sogrey@bogon demo4]$ sed r test.txt >> test22.txt # 将test.txt读取, 写入到test22.txt
[sogrey@bogon demo4]$ cat test22.txt
eeeeee eeeee
hello world!
[sogrey@bogon demo4]$
```

sendmail - 一个发送邮件的代理程序

sendmail是postfix中的一个发送邮件的代理程序，它负责发送邮件到远程服务器，并且可以接收邮件。sendmail在发送邮件的时候，默认从标准输入读取内容，以“.”为结束。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
sendmail [option ...] [recipient ...]
```

sh

选项

-bd	# 进入daemon模式
-bi	# 初始化别名数据库
-bm	# 从标准输入读取邮件
-bp	# 列出邮件列表
-bs	# 独立的smtp模式，从标准输入读取，输出到标准输出
-C	# main.cf的位置
-F	# 指定发送者的全名
-f	# 指定发送者
-i	# 忽略只有单独点的行
-q	# 以给定的时间间隔处理队列中的邮件

sh

举例

给某人发送邮件

```
[root@localhost ~]$ sendmail david      #发送邮件给david，默认的发送者是root
123.
.
[root@localhost ~]$ tail /var/spool/mail/david  #查看david邮箱，收到邮件
Delivered-To: david@david.cn
Received: by mailsrv.david.cn (Postfix, from userid 0)
          id 5B3A7143211; Fri,  5 Oct 2018 21:33:34 +0800 (CST)
Message-Id: <20181005133334.5B3A7143211@mailsrv.david.cn>
Date: Fri,  5 Oct 2018 21:33:29 +0800 (CST)
From: root@david.cn (wejje)
To: undisclosed-recipients:;

123.
```

指定发送者

```
[root@localhost ~]$ sendmail -f weijie david      #weijie发送邮件给david
hehe
.
You have new mail in /var/spool/mail/root
[root@localhost ~]$ tail /var/spool/mail/david      #查看david邮箱, 发送者是魏杰

Delivered-To: david@david.cn
Received: by mailsrv.david.cn (Postfix, from userid 0)
           id 3544314308F; Fri,  5 Oct 2018 21:37:11 +0800 (CST)
Message-Id: <20181005133711.3544314308F@mailsrv.david.cn>
Date: Fri,  5 Oct 2018 21:37:07 +0800 (CST)
From: weijie@david.cn (weijie)
To: undisclosed-recipients:;

hehe
```

查看表中有哪些字段

```
[root@localhost ~]$ mysqlshow -u root -p test wj      #显示数据库test中的表wj的信息
Enter password:
Database: test  Table: wj
+-----+-----+-----+-----+-----+
| Field | Type   | Collation        | Null | Key | Default | Extra | Privileges
+-----+-----+-----+-----+-----+
| id    | int(11) |                | NO   |     |          |       | select,insert,update,refer
| name  | text    | latin1_swedish_ci | NO   |     |          |       | select,insert,update,refer
+-----+-----+-----+-----+-----+
```

service - 控制系统服务的实用工具

service命令 是Redhat Linux兼容的发行版中用来控制系统服务的实用工具，它以启动、停止、重新启动和关闭系统服务，还可以显示所有系统服务的当前状态。

service可以控制系统服务（打开、关闭、重启）。service在尽可能可预测的环境中运行SystemV init脚本，删除大多数环境变量并将当前工作目录设置为根目录。脚本参数位于“/etc/init.d/script”中的System V init脚本。受支持的命令值取决于调用的脚本，服务将命令和选项传递给init脚本。

所有脚本至少应该支持start命令和stop命令。作为特例，如果命令是“--full-restart”，脚本将运行两次，首先使用stop命令，然后使用start命令。“service --status-all”按照字母顺序运行所有的init脚本，执行status命令。用户可以在/etc/init.d/目录下找到服务的脚本文件。

只有环境变量LANG和TERM传递给init脚本文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
service SCRIPT COMMAND [OPTIONS]
service --status-all
service --help | -h | --version
```

sh

选项

```
-h, --help          # 帮助信息
-V, --version       # 显示命令版本信息
--status-all        # 显示所有的服务状态
--full-restart      # 重启服务，运行两次，先停止后开启
[service_name cmd] # 控制服务。例如service vsftpd start。cmd可以是start、stop、restart
```

sh

举例

查看所有服务当前的运行状态

```
[sogrey@bogon ~]$ service --status-all
未加载 netconsole 模块
已配置设备:
lo enp0s3
当前活跃设备:
lo enp0s3 virbr0
[sogrey@bogon ~]$
```

sh

查看指定服务（vsftpd）的运行状态

```
[sogrey@bogon ~]$ service vsftpd status  
vsftpd (pid 30818) 正在运行...  
[sogrey@bogon ~]$
```

sh

停止指定服务（vsftpd）

```
[sogrey@bogon ~]$ service vsftpd stop  
关闭 vsftpd:  
[ 确定 ]  
[sogrey@bogon ~]$ service vsftpd status  
vsftpd 已停  
[sogrey@bogon ~]$
```

sh

set - 显示或设置shell特性及shell变量

set指令用来修改shell变量的显示和执行方式，没有任何选项和参数的时候，显示所有的变量名字和值。

set命令作用主要是显示系统中已经存在的shell变量，以及设置shell变量的新变量值。使用set更改shell特性时，符号"+ "和"- "的作用分别是打开和关闭指定的模式。set命令不能够定义新的shell变量。如果要定义新的变量，可以使用declare命令以变量名=值的格式进行定义即可。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
set [OPTION] [参数]
```

sh

选项

```
-a          # 自动标记已经创建或者修改的变量，以供输出到环境变量
-b          # 使被终止的后台程序立刻汇报状态
-e          # 如果指令执行的退出值不是0，那么立刻退出
-f          # 取消使用通配符
-h          # 自动记录函数所在的位置
-k          # 指令的参数都会被看作是指令的环境变量
-m          # 使用监视模式
-n          # 只读模式，不执行
-p          # 启动优先顺序模式
-t          # 当读取执行命令之后退出
-u          #
-v          # 显示shell读取的输入值
-x          # 执行指令后显示该指令和其参数
-o option   # 打开指定的特性。如果没有指定option，那么就打印当前的选项值
+o          # 关闭指定的特性。如果没有指定option，则在标准输出中显示一系列用于重新创建当前选项设
-B          # shell执行支撑扩展(请参阅上面的Braces扩展)。默认情况下这是打开的。
-C          # 如果设置，bash不会用'>'、'>&'和'<>'重定向操作符覆盖现有文件。当使用重定向运算符">
-E          # 如果设置了，Err上的任何陷阱都由shell函数、命令替换和在子shell环境中执行的命令继承
-H          # 启用！风格历史替代。默认情况下，当shell是交互式的时，此选项是打开的。
-P          # 如果已设置，则在执行更改当前工作目录的命令(如CD)时，shell不会遵循符号链接。它使用
-T          # 如果设置，任何调试和返回的陷阱都由shell函数、命令替换和在子shell环境中执行的命令继
--          # 如果在此选项之后没有参数，那么位置参数将被取消设置。否则，位置参数将被设置为args,
-            # 向选项的结束发出信号，使所有剩余的args被分配到位置参数。关闭-x和-v选项。如果没有ar
```

其中选项o的特征：

sh

```

allelexport          # 等价于“-a”
braceexpand          # 等价于“-B”
emacs               # 使用emacs风格的命令行编辑界面。默认情况下，当shell是交互式的时，这是启用的，除非sh
errexit              # 等价于“-e”
errtrace             # 等价于“-E”
functrace            # 等价于“-T”
hashall              # 等价于“-h”
histexpand           # 等价于“-H”
history              # 启用命令历史记录，如上文在“历史记录”下所述。默认情况下，此选项在交互式shell中打开。
ignoreeof            # 其效果似乎是执行了shell命令‘IGNOREEOF=10’(参见上面的Shell变量)。
keyword              # 等价于“-k”
monitor              # 等价于“-m”
noclobber            # 等价于“-C”
noexec               # 等价于“-n”
noglob               # 等价于“-f”
nolog                # 忽略
notify               # 等价于“-b”
nounset              # 等价于“-u”
onecmd               # 等价于“-t”
physical              # 等价于“-P”
pipefail             # 如果已设置，则管道的返回值是在非零状态下退出的最后一个(最右边)命令的值，如果管道中
posix                 # 将默认操作与POSIX标准不同的bash行为更改为与标准(POSIX模式)匹配。
privileged           # 等价于“-p”
verbose              # 等价于“-v”
vi                   # 使用vi样式的命令行编辑界面。这也将影响用于读-e的编辑界面。
xtrace               # 等价于“-x”

```

举例

显示所有的变量

sh

```

[root@localhost ~]$ set
BASH=/bin/bash
BASHOPTS=cdspell:checkwinsize:cmdhist:expand_aliases:extquote:force_fignore:hostcomplete:
interactive_comments:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_LINENO=()
...

```

显示shell的输出值

sh

```

[root@localhost ~]$ set -v
printf "\033[0;%s@%s:%s\007" "${USER}" "${HOSTNAME%%.*}" "${PWD/#$HOME/~}"

```

使用declare命令定义一个新的环境变量"mylove"，并且将其值设置为"Visual C++"，输入如下命令：

sh

```

declare mylove='Visual C++'    #定义新环境变量

```

再使用set命令将新定义的变量输出为环境变量，输入如下命令：

```
set -a mylove          #设置为环境变量
```

sh

执行该命令后，将会新添加对应的环境变量。用户可以使用env命令和grep命令分别显示和搜索环境变量"mylove"，输入命令如下：

```
env | grep mylove      #显示环境变量值
```

sh

此时，该命令执行后，将输出查询到的环境变量值。

shopt - 显示和设置shell操作选项

shopt命令 用于显示和设置shell中的行为选项，通过这些选项以增强shell易用性。shopt命令若不带任何参数选项，则可以显示所有可以设置的shell操作选项。

shopt指令可以设置控制shell行为特性的变量的开关值。没有任何选项的时候，显示由shopt控制的变量以及值。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
shopt [-pqsu] [-o] [optname ...]
```

sh

如果使用-s或-u中没有optname参数，则显示仅限于分别设置或未设置的选项。除非另有说明，否则Shop选项在默认情况下将被禁用(未设置)。如果启用了所有选项，则列表选项时的返回状态为零，否则为非零。设置或取消设置选项时，返回状态为零，除非optname不是有效的shell选项。

选项

-s	# 不带参数时，显示所有打开的变量；带参数时，打开变量
-u	# 不带参数时，显示所有关闭的变量；带参数时，关闭变量
-p	# 显示所有变量
-q	# 静默模式
-o	# 限制变量的值必须是使用-o选项定义的set内置变量

sh

举例

显示shpot控制的变量

```
[root@localhost ~]$ shopt
autocd      off
cdable_vars off
cdspell     on
checkhash   off
checkjobs   off
```

sh

打开变量

sh

```
[root@localhost ~]$ shopt -s autocd      #打开变量  
[root@localhost ~]$ shopt autocd          #查看变量状态, 已经打开  
autocd          on
```

关闭变量

sh

```
[root@localhost ~]$ shopt -u autocd      #关闭变量  
[root@localhost ~]$ shopt autocd          #查看变量, 已经关闭  
autocd          off
```

showmount - 显示NFS服务器加载的信息

showmount命令 查询“mountd”守护进程，以显示NFS服务器加载的信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
showmount [OPTION] [参数]
```

sh

选项

```
-d          # 仅显示已被NFS客户端加载的目录;  
-e          # 显示NFS服务器上所有的共享目录。
```

sh

NFS服务器：指定NFS服务器的ip地址或者主机名。

shutdown - 用来执行系统关机的命令

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。

shutdown指令以安全的方式来关闭系统，所有已经登录的用户都会被告知系统将要关闭。并且在最后五分钟内，新的登录将被阻止。过了指定的time后，关机会向init(8)守护进程发送一个请求，以便将系统降至适当的运行级别。这是通过发出runlevel(7)事件来执行的，该事件包括RUNLEVEL环境变量中的新运行级以及PREVLEVEL变量中的前一个运行级(从环境或/var/run/utmp获得)。可以设置一个额外的INIT_HALT变量，它将包含使用halt指令关机的HAL值，或者使用power off指令关机的POWEROFF值。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
shutdown [选项] [时间] [警告信息]
```

sh

选项

-c	# 取消正在执行的关机，这个选项没有时间参数
-h	# 关闭计算机，等于halt或者power off
-P	# 等价power off
-H	# 等价halt
-k	# 只是发出警告信息，注销登录，并没有关机
-r	# 重启，等于reboot
时间	# now代表立刻关机； +m代表m分钟后关闭； 23: 00代表在晚上11点关机

sh

说明

如果设置为首选从/var/run/utmp读取RUNLEVEL，关机将从此环境变量读取当前运行级。

"/var/run/utmp"文件，读取当前运行级的位置，该文件还将使用新的运行级别进行更新。

"/var/log/wtmp"文件，新运行级记录将追加到此文件中。Upstart init(8)守护进程不跟踪运行级别本身，而是完全由其用户空间工具实现。

举例

5分钟后关闭机器，并发出警告"I am downing"

```
[sogrey@bogon /]$ shutdown -h +5 I am downing # 当前操作有root用户发出，5分钟后关机，并且有提示信息  
Broadcast message from root@192.168.0.113  
(/dev/pts/1) at 10:40 ...  
The system is going down for halt in 5 minutes! # 5分钟  
I am downing # 自定义的提示信息
```

取消关机

```
[sogrey@bogon /]$ shutdown -c # 需要打开另一个终端，输入取消命令  
[sogrey@bogon /]$ shutdown -h +5 I am downing # 在之前的关机命令窗口，最后可以看到取消的信息  
...  
The system is going down for halt in 4 minutes!  
I am downing  
shutdown: Shutdown cancelled  
[sogrey@bogon /]$
```

skill - 向选定的进程发送信号冻结进程

skill命令 用于向选定的进程发送信号，冻结进程。这个命令初学者并不常用，深入之后牵涉到系统服务优化之后可能会用到。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
skill [OPTION]
```

sh

选项

-f	# 快速模式；
-i	# 交互模式，每一步操作都需要确认；
-v	# 冗余模式；
-w	# 激活模式；
-V	# 显示版本号；
-t	# 指定开启进程的终端号；
-u	# 指定开启进程的用户；
-p	# 指定进程的id号；
-c	# 指定开启进程的指令名称。

sh

举例

如果您发现了一个占用大量CPU和内存的进程，但又不想停止它，该怎么办？考虑下面的top命令输出：

```
top -c -p 16514
23:00:44 up 12 days, 2:04, 4 users, load average: 0.47, 0.35, 0.31
1 processes: 1 sleeping, 0 running, 0 zombie, 0 stopped
CPU states: cpu user nice system irq softirq iowait idle
      total 0.0% 0.6% 8.7% 2.2% 0.0% 88.3% 0.0%
Mem: 1026912k av, 1010476k used, 16436k free, 0k shrd, 52128k buff
      766724k actv, 143128k in_d, 14264k in_c
Swap: 2041192k av, 83160k used, 1958032k free          799432k cached

 PID USER      PRI  NI   SIZE  RSS SHARE stat %CPU %MEM   time CPU command
 16514 oracle    19   4 28796  26M 20252 D N    7.0  2.5  0:03  0 oraclePRODB2...
```

sh

既然您确认进程16514占用了大量内存，您就可以使用skill命令“冻结”它，而不是停止它。

```
skill -STOP 1
```

sh

之后，检查top输出：

```
23:01:11  up 12 days,  2:05,  4 users,  load average: 1.20, 0.54, 0.38
1 processes: 0 sleeping, 0 running, 0 zombie, 1 stopped
CPU states:  cpu      user      nice      system      irq      softirq      iowait      idle
              total     2.3%    0.0%     0.3%     0.0%     0.0%     2.3%    94.8%
Mem: 1026912k av, 1008756k used,   18156k free,       0k shrd,   3976k buff
                  770024k actv, 143496k in_d, 12876k in_c
Swap: 2041192k av,   83152k used, 1958040k free           851200k cached

PID USER      PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM      TIME CPU COMMAND
16514 oracle    19   4 28796  26M 20252 T N    0.0  2.5    0:04  0 oraclePRODB2...
```

现在，CPU从0%空闲变为94%空闲。该进程被有效冻结。过一段时间之后，您可能希望唤醒该进程：

```
skill -CONT 16514
```

sh

如果希望暂时冻结进程以便为完成更重要的进程腾出空间，该方法非常有用。

此命令用途很广。如果您要停止"oracle"用户的所有进程，只需要一个命令即可实现：

```
skill -STOP oracle
```

sh

可以使用用户、PID、命令或终端id作为参数。以下命令可停止所有rman命令。

```
skill -STOP rman
```

sh

如您所见，skill决定您输入的参数（进程ID、用户ID或命令）并进行相应操作。这可能会导致在某些情况下出现这样的问题：您可能具有同名的用户和命令。最好的示例是"oracle"进程，通常由用户"oracle"运行。因此，当您希望停止名为"oracle"的进程时，可执行以下命令：

```
skill -STOP oracle
```

sh

用户"oracle"的所有进程都停止，包括您可能要使用的会话。要非常明确地执行命令，您可以选择使用一个新参数指定参数的类型。要停止一个名为oracle的命令，可执行以下命令：

```
skill -STOP -c oracle
```

sh

stime命令的功能与skill类似。但它用于降低进程的优先级，而不是停止进程。首先，检查top输出：

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	%CPU	%MEM	TIME	CPU	COMMAND
3	root	15	0	0	0	0	RW	0.0	0.0	0:00	0	kapmd
13680	oracle	15	0	11336	10M	8820	T	0.0	1.0	0:00	0	oracle
13683	oracle	15	0	9972	9608	7788	T	0.0	0.9	0:00	0	oracle
13686	oracle	15	0	9860	9496	7676	T	0.0	0.9	0:00	0	oracle
13689	oracle	15	0	10004	9640	7820	T	0.0	0.9	0:00	0	oracle
13695	oracle	15	0	9984	9620	7800	T	0.0	0.9	0:00	0	oracle
13698	oracle	15	0	10064	9700	7884	T	0.0	0.9	0:00	0	oracle
13701	oracle	15	0	22204	21M	16940	T	0.0	2.1	0:00	0	oracle

现在，将 "oracle" 进程的优先级降低四个点。注意，该值越高，优先级越低。

```
sh
snice +4 -u oracle
PID USER      PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM    TIME CPU COMMAND
16894 oracle    20   4 38904  32M 26248 D N    5.5  3.2  0:01  0 oracle
```

注意，NI 列（nice 值）现在是 4，优先级现在设置为 20，而不是 15。这对于降低优先级非常有帮助。

slabtop - 实时显示内核slab内存缓存信息

slabtop命令 以实时的方式显示内核“slab”缓冲区的细节信息。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
slabtop [OPTION]
```

sh

选项

```
-d n, --delay=n          # 设置显示的时间间隔
-s S, --sort=S          # 设置排序规则。a, 以活动对象数目排序; b, 以每个slab对象数目排序; c, 以
-o, --once               # 只显示一次, 之后退出

--help                   # 显示帮助文档
-V, --version            # 显示命令版本信息
```

sh

扩展

内核的模块在分配资源的时候，为了提高效率和资源的利用率，都是透过slab来分配的。通过slab的信息，再配合源码能粗粗了解系统的运行情况，比如说什么资源有没有不正常的多，或者什么资源有没有泄漏。linux系统透过/proc/slabinf来向用户暴露slab的使用情况。

Linux 所使用的 slab 分配器的基础是 Jeff Bonwick 为 SunOS 操作系统首次引入的一种算法。Jeff 的分配器是围绕对象缓存进行的。在内核中，会为有限的对象集（例如文件描述符和其他常见结构）分配大量内存。Jeff 发现对内核中普通对象进行初始化所需的时间超过了对其进行分配和释放所需的时间。因此他的结论是不应该将内存释放回一个全局的内存池，而是将内存保持为针对特定目而初始化的状态。Linux slab 分配器使用了这种思想和其他一些思想来构建一个在空间和时间上都具有高效性的内存分配器。

保存着监视系统中所有活动的 slab 缓存的信息的文件为/proc/slabinf。

举例

显示slab信息

sh

```
[root@localhost ntop-4.0.1]$ slabtop -d 5          # 每5s显示一次
Active / Total Objects (% used)      : 434059 / 438446 (99.0%)
Active / Total Slabs (% used)       : 23317 / 23318 (100.0%)
Active / Total Caches (% used)      : 104 / 200 (52.0%)
Active / Total Size (% used)        : 87844.50K / 88275.36K (99.5%)
Minimum / Average / Maximum Object : 0.01K / 0.20K / 4096.00K
```

OBJS	ACTIVE	USE	OBJ	SIZE	SLABS	OBJ/SLAB	CACHE	SIZE	NAME
120321	120297	99%	0.13K	4149	29		16596K	dentry	
64722	64605	99%	0.05K	966	67	3864K	buffer_head		
50178	50170	99%	0.63K	8363		6	33452K	ext4_inode_cache	
47870	47846	99%	0.38K	4787	10		19148K	proc_inode_cache	

...

显示slab信息，并且排序

sh

```
[root@localhost ntop-4.0.1]$ slabtop -d 10 -s n      # 按名字排序，每10s显示一次
Active / Total Objects (% used)      : 433100 / 438274 (98.8%)
Active / Total Slabs (% used)       : 23307 / 23307 (100.0%)
Active / Total Caches (% used)      : 104 / 200 (52.0%)
Active / Total Size (% used)        : 87750.19K / 88220.52K (99.5%)
Minimum / Average / Maximum Object : 0.01K / 0.20K / 4096.00K
```

OBJS	ACTIVE	USE	OBJ	SIZE	SLABS	OBJ/SLAB	CACHE	SIZE	NAME
290	261	90%	0.02K	2	145		8K	Acpi-Namespace	
1564	1501	95%	0.04K		17 92		68K	Acpi-Operand	
0	0	0%	0.03K	0	113		0K	Acpi-Parse	
0	0	0%	0.05K	0	78		0K	Acpi-ParseExt	
0	0	0%	0.04K	0	84		0K	Acpi-State	
0	0	0%	0.56K	0		7	0K	PING	
7	2	28%	0.56K	1		7	4K	RAW	
5	4	80%	0.75K	1		5	4K	RAWv6	

...

smbclient - 交互式的访问samba服务器的客户端的管理程序

smbclient是一个smb服务器的客户端的管理程序，可以交互式的访问samba服务器。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
smbclient servername [选项]
```

sh

选项

```
-?, --help          # 显示帮助文档
-V, --version       # 显示命令版本信息
-R, --name-resolve # 将NetBIOS名称解析成对应的IP地址顺序
-M, --message        # 使用winpopup协议发送消息
-p, --port          # 指定连接端口，默认TCP的139
-m, --max-protocol # 协议的最大版本
-I, --ip-address    # 指定连接的ip地址
-E, --stderr         # 将信息送到标准出错设备
-L, --list           # 显示服务器的资源列表
-b, --send-buffer   # 设置传输过程的缓冲区大小
-e, --encrypt        # 要求服务器采用加密方式
-d, --debuglevel    # 设置调试模式级别，级别越高显示的日志就月详细
-l, ---log-basename # 日志文件的存放目录
-N, --no-pass        # 不使用密码
-A                  # 从指定文件读取用户名和密码，文件格式如下：
                    # username=<value>
                    # password=<value>
                    # domain=<value>
-U, --user=username[%password] # 指定用户名和密码
-n                  # 指定NetBIOS名称
-W, --workgroup      # 指定用户的smb域
-T, --tar            # 将服务器共享的文件打包成tar格式
-s                  # 指定smb.conf目录
```

sh

命令

登录samba服务器后支持的命令有

```

-? [cmd]          # 显示命令的说明文档, 如果不指定命令, 列出所有的命令文档
! [shell]         # 运行shell
allinfo file      # 要求服务器返回所有文件或者目录的信息
altname file      # 要求服务器返回文件或者目录的别名
archive <num>     # 设置归档级别
blocksize <size>   # 设置打包的块大小, 默认20。块的单位是521B
case_sensitive    # 设置文件大小写敏感
cd                # 切换目录
chmod              # 修改权限
chown              # 修改uid和gid
close <fd>         # 关闭使用open打开的文件
del <mask>        # 删除当前目录下符合mask的文件
du                # 列出目录信息和磁盘信息
echo <number> <data> # 该指令的作用是向服务器发送ping的测试信息
exit              # 退出
get                # 获取文件
hardlink <src> <dst> # 创建硬连接
iosize <bytes>     # 设置传输文件时使用内存缓冲区的大小, 默认64512字节, 可以设置的范围16384~1677
lowercase          # 将受到的文件中字母都改成小写
ls                # 查看目录信息
md                # 创建目录
mget <mask>       # 获取所有匹配mask的文件
mkdir              # 创建目录
mput <mask>       # 发送当前目录下所有匹配mask的文件到服务器
put <local file> [<remote file>] # 发送文件
queue              # 显示打印队列
quit              # 退出
rd                # 删除目录
rmdir              # 删除目录
recurse            # 改变递归选项的开关
rename             # 重命名
rm <mask>         # 删除当前目录下所有匹配mask的文件
showconnect        # 显示当前连接
stat file          # 显示文件信息
tar                # 压缩文件

```

举例

连接服务器

```

[root@localhost ~]$ cat wj.txt          #查看文件内容
username=david
password=543092
[root@localhost ~]$ smbclient //192.168.1.8/wj -A wj.txt    #登录服务器, 从指定文件读取用户名和密码
Domain=[MYGROUP] OS=[Unix] Server=[Samba 3.6.23-51.el6]
smb: \>

```

查看命令

sh

```
smb: \> ?          #列出支持的命令
?
allinfo      altnname      archive      blocksize
cancel       case_sensitive cd           chmod        chown
close        del           dir           du           echo
exit         get           getfacl      geteas      hardlink
help         history       iosize       lcd          link
lock         lowercase    ls            l             mask
md           mget          mkdir        more        mput
newer        open          posix        posix_encrypt posix_open
posix_mkdir  posix_rmdir   posix_unlink print      prompt
put          pwd           q             queue      quit
readlink     rd            recurse     reget      rename
reput        rm            rmdir       showacls   setea
setmode      stat          symlink     tar         tarmode
timeout     translate    unlock      volume     vuid
wdel        logon        listconnect showconnect ..
!
smb: \> ? allinfo      #查看指定命令的帮助信息
HELP allinfo:
<file> show all available info

smb: \>
```

上传文件

sh

```
smb: \> put wj.txt      #上传文件
putting file wj.txt as \wj.txt (10.1 kb/s) (average 10.1 kb/s)
smb: \> ls wj.txt      #查看文件, 已经上传
wj.txt                                A          31  Tue Oct  9 09:15:22 2018

49907 blocks of size 524288. 2282 blocks available
smb: \>
```

smbpasswd - 可以用来修改samba用户的密码

smbpasswd指令可以用来修改samba用户的密码，该指令不仅可以修改本地samba服务器的用户密码，还可以修改远程samba服务器的用户密码。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
smbpasswd [选项] username
```

sh

选项

-a	# 添加用户到samba服务器
-c	# 指定配置文件smb.conf的位置
-x	# 删除用户
-d	# 停止使用指定的用户
-e	# 激活暂停的用户
-D	# 设置调试级别0~10
-n	# 指定用户名为空密码
-r	# 指定远程smb服务器上的用户密码
-U	# 指定用户名，只和-r配合使用
-h	# 显示帮助信息

sh

举例

添加用户

```
[root@localhost ~]$ smbpasswd -a sogrey      #添加用户，设置密码
New SMB password:
Retype new SMB password:
Added user sogrey.
```

sh

修改用户密码

```
[root@localhost ~]$ smbpasswd sogrey      #修改用户密码
New SMB password:
Retype new SMB password:
```

sh

sort - 对文本文件中所有行进行排序

以行为单位，对文本文件进行排，并输出排序结果。默认情况下，以每一行为一个单位，从首字符开始按照ASCII码向后逐个比较。

主要用途：

- 将所有输入文件的内容排序后并输出。
- 当没有文件或文件为-时，读取标准输入。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sort [选项] file
```

```
sort [OPTION]... --files0-from=F
```

sh

选项

```

# 排序
-b, --ignore-leading-blanks      # 忽略开头的空白。
-d, --dictionary-order          # 仅考虑空白、字母、数字。
-f, --ignore-case                # 将小写字母作为大写字母考虑。
-g, --general-numeric-sort     # 根据数字排序。
-i, --ignore-nonprinting        # 排除不可打印字符。
-M, --month-sort                 # 按照非月份、一月、十二月的顺序排序。
-h, --human-numeric-sort        # 根据存储容量排序(注意使用大写字母，例如：2K 1G)。
-n, --numeric-sort               # 根据数字排序。
-R, --random-sort                # 随机排序，但分组相同的行。
--random-source=FILE            # 从FILE中获取随机长度的字节。
-r, --reverse                     # 将结果倒序排列。
--sort=WORD                      # 根据WORD排序，其中：general-numeric 等价于 -g,
# human-numeric 等价于 -h, month 等价于 -M, numeric
# 等价于 -n, random 等价于 -R, version 等价于 -V。
-V, --version-sort               # 文本中(版本)数字的自然排序。

# 其他

--batch-size=NMERGE              # 一次合并最多NMERGE个输入;
# 超过部分使用临时文件。
-c, --check, --check=diagnose-first # 检查输入是否已排序，该操作不会执行排序。
-C, --check=quiet, --check=silent   # 类似于 -c 选项，但不输出第一个未排序的行。
--compress-program=PROG           # 使用PROG压缩临时文件；使用PROG -d解压缩。
--debug                           # 注释用于排序的行，发送可疑用法的警报到stderr。
--files0-from=F                   # 从文件F中读取以NUL结尾的所有文件名称;
# 如果F是 -，那么从标准输入中读取名字。
-k, --key=KEYDEF                 # 通过一个key排序; KEYDEF给出位置和类型。
-m, --merge                         # 合并已排序文件，之后不再排序。
-o, --output=FILE                  # 将结果写入FILE而不是标准输出。
-s, --stable                        # 通过禁用最后的比较来稳定排序。
-S, --buffer-size=SIZE             # 使用SIZE作为内存缓存大小。
-t, --field-separator=SEP         # 使用SEP作为列的分隔符。
-T, --temporary-directory=DIR      # 使用DIR作为临时目录，而不是 $TMPDIR
# 或 /tmp; 多次使用该选项指定多个临时目录。
--parallel=N                       # 将并发运行的排序数更改为N。
-u, --unique                        # 同时使用-c，严格检查排序；不同时使用-c，
# 输出排序后去重的结果。
-z, --zero-terminated               # 设置行终止符为NUL（空），而不是换行符。

--help                            # 显示帮助信息并退出。
--version                         # 显示版本信息并退出。

```

KEYDEF的格式为：F[C][OPTS][,F[C][OPTS]]，表示开始到结束的位置。

F表示列的编号

C表示

OPTS为[bdfgiMhnRrV]中的一到多个字符，用于覆盖当前排序选项。

使用--debug选项可诊断出错误的用法。

返回值

返回0表示成功，返回非0值表示失败。

举例

```
[sogrey@bogon 文档]$ ls  
backup demos  
[sogrey@bogon 文档]$ cd demos  
[sogrey@bogon demos]$ ls  
test2.txt test3.txt test4.txt test.txt  
[sogrey@bogon demos]$ cat test.txt  
石家庄今日新增16例确诊病例  
中国留美博士遇害 美驻华使馆慰问  
特朗普夫人发文谴责国会暴乱  
理塘文旅公司回应丁真抽烟  
北京一确诊者隐瞒行程不配合流调  
山西晋中新增2例无症状感染者  
[sogrey@bogon demos]$ sort test.txt  
北京一确诊者隐瞒行程不配合流调  
理塘文旅公司回应丁真抽烟  
山西晋中新增2例无症状感染者  
石家庄今日新增16例确诊病例  
特朗普夫人发文谴责国会暴乱  
中国留美博士遇害 美驻华使馆慰问  
[sogrey@bogon demos]$
```

spell - 对文件进行拼写检查

spell命令 对文件进行拼写检查，并把拼写错误的单词输出。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
spell file
```

sh

举例

```
$ cat test.txt
Holle world
$ spell test.txt
Holle # 检查出 Holle 拼写错误
```

sh

split - 分割任意大小的文件

将一个大文件切割成较小的文件， 默认情况下每1000行就会切割一次。分割后的文件， 默认以xaa、xab、xac等命名。用户亦可以指定名字的前缀， 例如指定前缀test， 那么分割后的文件是testaa、testab、testac等。

split命令 可以将一个大文件分割成很多个小文件， 有时需要将文件分割成更小的片段， 比如为提高可读性， 生成日志等。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
split [OPTION]... [INPUT [PREFIX]]
```

sh

选项

```
-a, --suffix-length=N      # 使用长度为N的后缀(默认为2)
-b, --bytes=SIZE          # 设置多少个字节分割一次
-C size, --line-size=size # 设置每行最多size个字节
-d, --numeric-suffixes    # 用数字后缀代替字母
-l num, --line=num, -num   # 设置每多少行切割一次
--verbose                 # 在打开每个输出文件之前打印一个诊断文件

--help                     # 显示帮助文档
--version                  # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon demos]$ ls
test2.txt test3.txt test4.txt test.c test.txt
[sogrey@bogon demos]$ cat test.txt
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon demos]$ split -2 test.txt # 每2行分割一次
[sogrey@bogon demos]$ ls
test2.txt test3.txt test4.txt test.c test.txt xaa xab xac
[sogrey@bogon demos]$ cat xaa
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon demos]$ cat xab
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
[sogrey@bogon demos]$ cat xac
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon demos]$ cat test2.txt
特朗普夫人发文谴责国会暴乱
```

1月11日，美国第一夫人梅拉尼娅·特朗普通过白宫发表声明，谴责上周发生在美国国会的暴乱。

```
[sogrey@bogon demos]$ split -b 10 test2.txt SPLIT # 10个字节分割，前缀名是split
[sogrey@bogon demos]$ ls
SPLITaa SPLITad SPLITag SPLITaj SPLITam SPLITap    test3.txt test.txt xac
SPLITab SPLITae SPLITah SPLITak SPLITan SPLITaq    test4.txt xaa
SPLITac SPLITaf SPLITai SPLITal SPLITao test2.txt test.c      xab
[sogrey@bogon demos]$ cat SPLITaa
特朗普 [sogrey@bogon demos]$
```

squid - squid服务器守护进程

squid命令 高性能的Web客户端代理缓存服务器套件“squid”的服务器守护进程。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
squid [OPTION]
```

sh

选项

```
-d      # 将指定调试等级的信息发送到标准错误设备;  
-f      # 使用指定的配置文件。而不使用默认配置文件;  
-k      # 向squid服务器发送指令;  
-s      # 启用syslog日志;  
-z      # 创建缓存目录;  
-C      # 不捕获致命信号;  
-D      # 不进行DNS参数测试;  
-N      # 以非守护进程模式运行;  
-X      # 强制进入完全调试模式。
```

sh

squidclient - squid服务器的客户端管理工具

squidclient命令 使用squid服务器的客户端管理工具，它可以查看squid服务器的详细运行信息和管理squid服务器。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
squidclient [OPTION] [参数]
```

sh

参数：

URL：指定操作缓存中的URL。

选项

```
-a      # 不包含“accept:header”;  
-r      # 强制缓存重新加载URL;  
-s      # 安静模式，不输出信息到标准输出设备;  
-h      # 从指定主机获取url  
-l      # 指定一个本地ip地址进行绑定;  
-p      # 端口号， 默认为3128;  
-m      # 指定发送请求的方法;  
-u      # 代理认证用户名。
```

sh

sshd - openssh软件套件中的服务器守护进程

sshd命令 是openssh软件套件中的服务器守护进程。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sshd [OPTION]
```

sh

选项

```
-4      # 强制使用IPv4地址;  
-6      # 强制使用IPv6地址;  
-D      # 以后台守护进程方式运行服务器;  
-d      # 调试模式;  
-e      # 将错误发送到标准错误设备, 而不是将其发送到系统日志;  
-f      # 指定服务器的配置文件;  
-g      # 指定客户端登录时的过期时间, 如果在此期限内, 用户没有正确认证, 则服务器断开次客户端的连接;  
-h      # 指定读取主机key文件;  
-i      # ssh以inetd方式运行;  
-o      # 指定ssh的配置选项;  
-p      # 静默模式, 没有任何信息写入日志;  
-t      # 测试模式。
```

sh

stat - 显示文件或者文件系统的状态信息

显示文件或者文件系统的状态信息，可以输出文件名、大小、文件类型、访问权限、uid、gid。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
stat [选项] file
```

sh

选项

```
-f, --file-system          # 显示文件系统状态
-c, --format=FORMAT       # 使用指定的格式输出，每一个输出后打印换行
--printf=FORMAT           # 和“--format”一样，但是解释反斜杠转义,
                           # 不要输出强制性的尾随换行符。如果您想要换行符,
                           # 请在格式中包括\n
-t, --terse               # 用简明的格式显示
-L                         # 允许符号链接
-Z                         # 如果selinux可用，那么显示安全上下文

--help                      # 显示帮助文档
--version                   # 显示命令版本信息
```

sh

针对文件的格式

```
%a      # 以八进制为单位的访问权
%A      # 人类可读的访问权
%b      # block 的大小
%B      # %b报告的每个块的大小(以字节为单位)
%C      # SELinux上下文的字符串
%d      # 十进制设备号
%D      # 十六进制的设备号
%f      # 十六进制中的原始模式
%F      # 文件类型
%g      # 拥有者的gid
%G      # 拥有者的组名
%h      # 硬连接数量
%i      # inode
%n      # 文件名
%N      # 如果是符号链接, 那么文件名用引号扩起来
%o      # IO块的大小
%s      # 全部大小, 字节
%t      # 十六进制的主设备类型
%T      # 十六进制的次设备类型
%u      # 拥有者的uid
%U      # 拥有者的名字
%x      # 最后访问时间
%X      # 上一次访问时间为自纪元以来的秒数
%y      # 最后修改时间
%Y      # 上一次修改时间为自纪元以来的秒数
%z      # 上一次变更的时间
%Z      # 上一次变更时间为自纪元以来的秒数
```

针对文件系统的格式

```
%a      # 非超级用户可用的空闲块
%b      # 文件系统中的数据块总量
%c      # 文件系统中的节点总量
%d      # 文件系统中的空闲节点
%f      # 文件系统中的空闲块
%C      # SELinux的上下文
%i      # 文件系统的ID, 十六进制
%l      # 文件名字的最大长度
%n      # 文件名
%s      # 块大小 (用于快速传输)
%S      # 基本块大小(用于块计数)
%t      # 十六进制的类型
%T      # 人类可读的形式
```

举例

```
[sogrey@bogon 文档]$ cd demos
[sogrey@bogon demos]$ ls
test2.txt test3.txt test4.txt test.c test.txt
[sogrey@bogon demos]$ stat test.txt
文件: "test.txt"
大小: 250      块: 8          IO 块: 4096    普通文件
设备: fd02h/64770d Inode: 9568824    硬链接: 1
权限: (0700/-rwx-----) Uid: ( 1000/  sogrey)  Gid: ( 1000/  sogrey)
环境: unconfined_u:object_r:user_home_t:s0
最近访问: 2021-01-15 23:25:26.061982500 +0800
最近更改: 2021-01-12 00:54:12.868311798 +0800
最近改动: 2021-01-12 00:54:12.868311798 +0800
创建时间: -
[sogrey@bogon demos]$ stat -t test.txt
test.txt 250 8 81c0 1000 1000 fd02 9568824 1 0 0 1610724326 1610384052 1610384052 0 4096 unconfi
[sogrey@bogon demos]$ stat test.txt -c %u,%g,%x
1000,1000,2021-01-15 23:25:26.061982500 +0800
[sogrey@bogon demos]$
```

su - 用于切换当前用户身份到其他用户身份

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。

临时切换身份到另外一个用户，使用su切换用户之后，不会改变当前的工作目录，但是会改变一些环境变量。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
su [选项] [参数]
```

选项

- , -l, --login	# 切换用户时启动一个新的shell，可以改变工作目录以及环境变量
-c, --command	# 执行案指令后，立刻恢复原来的身份
--session-command=COMMAND	# 使用-c将单个命令传递给shell，而不创建新会话。
-f, --fast	# 使shell不读取启动文件
-m, -p, --preserve-environment	# 保留原来的环境变量
-s, --shell	# 指定切换用户后使用的shell
--help	# 显示帮助文档
--version	# 显示命令版本信息

举例

从其他用户切换到root

```
sh
[sogrey@bogon ~]$ su
密码:
[root@bogon sogrey]#
```

切换用户,环境变量没有发生改变

```
sh
[root@bogon sogrey]# su sogrey
[sogrey@bogon ~]$
```

使用选项"-切换,工作目录发生变化

sh

```
[root@bogon sogrey]# su - sogrey
上一次登录: 2021年 3月 24日 00:23:52 星期三 CSTpts/0 上
[sogrey@bogon ~]$
```

sudo - 以其他身份来执行命令

sudo命令 用来以其他身份来执行命令，预设的身份为root。在/etc/sudoers中设置了可执行sudo指令的用户。若其未经授权的用户企图使用sudo，则会发出警告的邮件给管理员。用户使用sudo时，必须先输入密码，之后有5分钟的有效期限，超过期限则必须重新输入密码。

sudo允许用户以超级用户或安全策略指定的另一个用户的身份执行命令。Sudo支持安全策略插件和输入/输出日志的插件。第三方可以开发和分发自己的策略和I/O日志插件，以便与sudo前端无缝地工作。默认的安全策略是sudoers，它是通过文件/etc/sudoers或通过LDAP配置的。

安全策略确定用户在需要什么权利的时候需要运行sudo。该策略可能要求用户使用密码或其他身份验证机制进行身份验证。如果需要身份验证，如果用户的密码未在可配置的时限内输入，sudo将退出。此限制是特定于策略的；sudoers安全策略的默认密码提示超时为5分钟。

安全策略可能支持凭据缓存，允许用户在不需要身份验证的情况下再次运行sudo。sudoers策略将凭据缓存5分钟，除非在sudoers(5)中重写。通过使用"-v"选项运行sudo，用户可以在不运行命令的情况下更新缓存的凭据。

安全策略可能记录使用sudo的成功和失败情况。如果配置了I/O插件，运行中的命令的输入和输出也可能被记录下来。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
sudo -h | -K | -k | -V
sudo -v [-AknS] [-g group name | #gid] [-p prompt] [-u user name | #uid]
sudo -l[1] [-AknS] [-g group name | #gid] [-p prompt] [-U user name] [-u user name | #uid]
sudo [-AbEHnPS] [-C fd] [-g group name | #gid] [-p prompt] [-r role] [-t type] [-u user r
sudoedit [-AnS] [-C fd] [-g group name | #gid] [-p prompt] [-u user name | #uid] file ...
```

选项

```

-b          # 在后台执行指令;
-E          # 继承当前环境变量
-H          # 将HOME环境变量设为新身份的HOME环境变量;
-k          # 结束密码的有效期限, 也就是下次再执行sudo时便需要输入密码; .
-l          # 列出目前用户可执行与无法执行的指令;
-p          # 改变询问密码的提示符号;
-s<shell>  # 执行指定的shell;
-u<用户>   # 以指定的用户作为新的身份。若不加上此参数, 则预设以root作为新的身
-v          # 延长密码有效期限5分钟;

-h, --help      # 显示帮助文档
-V, --version   # 显示命令版本信息

```

执行命令

当sudo执行命令时, 安全策略指定命令的执行环境。通常, 将实际有效的uid和gid设置为与密码数据库中指定的目标用户相匹配, 并根据组数据库初始化组向量(除非指定了-p选项)。安全策略可能会指定一些参数: 真实有效用户ID、真实有效组ID、补充组ID、环境列表、当前工作目录、文件创建掩码、SELinux的角色和类型、调度级别。

1. 进程模型

当sudo运行一个命令时, 它调用fork(2), 设置上面描述的执行环境, 并在子进程中调用execve系统调用。主sudo进程等待命令完成, 然后将命令的退出状态传递给安全策略的close方法并退出。如果配置了I/O日志插件, 则将一个新的伪终端("pty")被创建, 第二个sudo进程用于在用户现有的pty和正在运行的新pty之间传递作业控制信号。这个额外的进程使挂起并恢复命令成为可能。如果没有它, 命令将使用POSIX语中的“孤立进程组”。也不会收到任何作业控制信号

2. 信号处理

因为命令是作为sudo进程的子进程运行的, 所以sudo会将接收到的信号中继到命令。除非该命令在新的pty中运行, 否则SIGHUP、SIGINT和SIGQUIT信号将不会被中继, 除非它们是由用户进程而不是内核发送的。否则, 该命令将在用户按下“ctrl+c”时接收到两次SIGINT信号。由于SIGSTOP和SIGKILL不能被捕获, 因此不会被中继到命令。作为一般规则, 当您希望挂起sudo运行的命令时, 应该使用SIGTSTP而不是SIGSTOP。

作为特例, sudo将不会中继它正在运行的命令发送的信号。这可以防止命令意外地杀死自己。在某些系统上, reboot(8)命令在重新启动系统之前将SIGTERM发送到所有非系统进程, 而不是它自己。这防止sudo将接收到的SIGTERM信号中继回reboot(8), 然后该信号可能会在系统实际启动之前退出, 使其处于类似于单用户模式的半死状态。但是, 请注意, 此检查只适用于sudo运行的命令, 而不适用于命令可能创建的任何其他进程。因此, 通过sudo运行调用重reboot(8)或shutdown(8)的脚本可能导致系统处于这种未定义状态, 除非使用exec()函数系列而不是system()运行reboot(8)或shutdown(8)(后者在命令和调用进程之间插入一个shell)。

插件

插件根据“/etc/sudo.conf”文件的内容动态加载。如果没有“/etc/sudo.conf”文件, 或者它不包含插件行, sudo将使用传统sudoers安全策略和I/O日志记录, 这相当于以下“/etc/sudo.conf”文件

```
#  
# Default /etc/sudo.conf file  
#  
# Format:  
#   Plugin plugin_name plugin_path plugin_options ...  
#   Path askpass /path/to/askpass  
#   Path noexec /path/to/sudo_noexec.so  
#   Debug sudo /var/log/sudo_debug all@warn  
#   Set disable_coredump true  
#  
# The plugin_path is relative to /usr/libexec unless  
#   fully qualified.  
# The plugin_name corresponds to a global symbol in the plugin  
#   that contains the plugin interface structure.  
# The plugin_options are optional.  
#  
Plugin policy_plugin sudoers.so  
Plugin io_plugin sudoers.so
```

插件行由插件关键字组成，后面跟着符号名字和包含插件的共享对象的路径。符号名字是插件共享对象中 struct policy_plugin 或 struct io_plugin 的名称。路径可以是完全限定的或相对的。如果不完全限定，则相对于“/usr/libexec”目录。路径之后的任何附加参数都是作为参数传递给插件的开放函数。不以Plugin、Path、Debug、Set开头的行将被默认忽略。

路径

路径行由Path关键字组成，后面跟着要设置的路径的名称及其值。例：

```
Path noexec /usr/libexec/sudo_noexec.so  
Path askpass /usr/X11R6/bin/ssh-askpass
```

以下与插件无关的路径可以在“/etc/sudo.conf”文件中设置

1. askpass，辅助程序的完全限定路径，用于在没有终端可用时读取用户的密码。当sudo从图形应用程序执行时，情况可能是这样。由askpass指定的程序应该将传递给它的参数显示为提示符，并将用户的密码写入标准输出。askpass可能被环境变量SUDO_ASKPASS覆盖。
2. noexec，共享库的完全限定路径，包含仅返回错误的execv()、execve()和fexecve()库函数的虚拟版本，用于在支持LD_PRELOAD或其等效的系统上实现noexec功能。默认值为“/usr/libexec/sudo_noexec.so”。

调试标志

Sudo版本1.8.4及更高版本支持一种灵活的调试框架，如果存在问题，可以帮助跟踪sudo在内部做什么。

Debug行由Debug关键字组成，后面跟着要调试的程序名称(sudo、visudo、sudoreplay)、调试文件名和以逗号分隔的调试标志列表。sudo和sudoers插件使用的调试标志语法是subsystem@priority，但是插件可以自由使用不同的格式，只要它不包括逗号。例如语句“Debug sudo /var/log/sudo_debug all@warn,plugin@info”将会在警告级别和更高级别记录插件子系统的所有调试语句以及信息级别的语句。

目前，每个程序只支持一个调试条目。sudo调试条目由sudo前端、sudodit和plugins共享。将来的版本可

能会增加对每个插件调试行的支持和/或对单个程序的多个调试文件的支持。

sudo前端使用的优先级依次为: crit、err、warn、notice、diag、info、trace、debug.。当指定每个优先级时, 还包括所有高于此优先级的优先级。例如, 通知的优先级将包括记录在通知中的调试消息以及更高的优先级。

sudo前端可以使用一下子系统:

子系统	说明
all	所有的子系统。
args	命令行参数进程。
conv	用户回会话
edit	sudoedit。
exec	命令执行过程。
main	sudo的主函数。
netif	网络接口处理。
pcomm	插件会话
plugin	插件配置。
pty	为tty相关代码。
selinux	SELinux专用处理。
util	实用函数。
utmp	utmp处理

退出值

当程序成功执行时, sudo的退出状态将只是被执行的程序的退出状态。否则, 如果存在配置/权限问题或sudo无法执行给定的命令, sudo将以1退出。在后一种情况下, 错误字符串将打印到标准错误。如果sudo无法在用户路径中调用stat函数统计一个或多个条目, 则在stderr上打印错误。(如果该目录不存在, 或者它实际上不是一个目录, 则忽略该条目, 并且不打印错误。)在正常情况下不应该发生这种情况。stat(2)返回“拒绝权限”的最常见原因是, 如果您正在运行一个自动侦听器, 并且您的路径中的一个目录位于当前无法访问的计算机上。

安全说明

当执行外部命令时, sudo试图保持安全。为了防止命令欺骗, sudo在用户路径中搜索命令时, 最后检查".和""。但是请注意, 实际的path环境变量没有被修改, 而是不改变地传递给sudo执行的程序。

请注意, sudo通常只记录它显式运行的命令。如果用户运行“sudo su”或“sudo sh”之类的命令, 则从该shell运行的后续命令不受sudo的安全策略的约束, 提供shell转义的命令也是如此。如果启用了I/O日志记录, 随后的命令将有它们的输入和输出记录, 但这些命令不会有传统的日志。因此, 当用户通过sudo访问命令时, 必须小心, 以验证该命令不会无意中给用户一个有效的root shell。

为了防止泄露潜在的敏感信息, sudo在执行时默认禁用核心转储。为了帮助调试sudo崩溃, 您可能希望通

过在“/etc/sudo.conf”文件中将“disable_coredump”设置为false来重新启用核心转储，如下所示

```
Set disable_coredump false
```

请注意，默认情况下，大多数操作系统从setuid程序(包括sudo)禁用核心转储。要实际获得sudo核心文件，您可能需要为setuid进程启用核心转储。在BSD和Linux系统上，这是通过“sysctl”命令完成的，在Solaris上可以使用“coreadm”命令。

环境变量

sudo使用一下环境变量，安全策略控制命令环境的实际内容。

环境变量	说明
EDITOR	如果没有设置SUDO_EDITOR或VISUAL，则默认编辑器使用“-e”模式。
MAIL	在“-i”模式中或在sudoers中启用env_reset时，将其设置为目标用户的邮件线轴。
HOME	如果指定了“-i”或“-H”，在sudoers中赋值给目标用户的home目录，或者赋值给env_reset或all_set_home。或者当指定“-s”选项时，sudoers中赋值给set_home。
PATH	可能被安全策略覆盖
SEHLL	使用“-s”选项，执行运行的shell
SUDO_ASKPASS	如果没有可用的终端，或者指定了“-A”选项，则指定用于读取密码的辅助程序的路径。
SUDO_COMMAND	赋值给sudo运行的命令
SUDO_EDITOR	“-e”模式下的默认编辑器
SUDO_GID	赋值给调用sudo的用户的组ID
SUDO_PROMPT	作为默认的密码提示语句
SUDO_PS1	如果设置，PS1将被设置为正在运行的程序的值。
SUDO_UID	赋值给调用sudo的用户的ID
SUDO_USER	赋值给调用sudo的用户登录名
USER	赋值给目标用户（默认是root，除非指定“-u”选项）
VISUAL	如果在“-e”模式下没有指定“SUDO_EDITOR”，那么这个就是默认编辑器

举例

```
[sogrey@bogon ~]$ sudo su -
[sudo] sogrey 的密码:
上一次登录: 2021年 3月 24日 00:24:36 星期三 CSTpts/0 上
[root@bogon ~]#
```

sh

这个命令相当于使用root超级用户重新登录一次shell，只不过密码是使用的当前用户的密码。而且重要

是，该命令会重新加载/etc/profile文件以及/etc/bashrc文件等系统配置文件，并且还会重新加载root用户的\$SHELL环境变量所对应的配置文件，比如：root超级用户的\$SHELL是/bin/bash，则会加载/root/.bashrc等配置。如果是/bin/zsh，则会加载/root/.zshrc等配置，执行后是完全的root环境。

```
[sogrey@bogon ~]$ sudo -i  
[root@bogon ~]#
```

sh

这个命令基本与 sudo su - 相同，执行后也是root超级用户的环境，只不过是多了一些当前用户的信息。

```
$ sudo -u yaz ls ~yaz          //查看用户yaz的家目录  
$ sudo -u www vi ~www/htdocs/index.html //以用户www的身份编辑文件  
$ sudo -g adm view /var/log/syslog //以组adm的身份去查看日志文件，  
$ sudo -u jim -g audio vi ~jim/sound.txt //要使用不同的主组以Jim的身份运行编辑器  
$ sudo shutdown -r +15 "quick reboot" //关机  
$ sudo sh -c "cd /home ; du -s * | sort -rn > USAGE" //若要对/home分区中的目录进行使用列表，请注意
```

sh

sum - 计算文件的校验码和块数

计算文件的校验码和块数，可以采用BSD和System V两种算法。如果没有指定文件，或者文件名是“-”，那么文件就是标准输入。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
sum [选项] file
```

sh

选项

-r	# 使用System V算法，使用1K字节。
-s	# 使用BSD算法，使用512字节
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

```
[sogrey@bogon demos]$ sum -r test.txt
19632 1
[sogrey@bogon demos]$ sum -s test.txt
44089 1 test.txt
[sogrey@bogon demos]$
```

sh

swapoff - 关闭指定的交换空间

swapoff命令 用于关闭指定的交换空间（包括交换文件和交换分区）。swapoff实际上为swapon的符号连接，可用来关闭系统的交换区。

关闭交换分区，同时可以刷新交换分区的缓存。关闭交换分区后，使用free指令查看内存，swap数值会减少。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
swapoff [-v] specialfile...
swapoff -a [-v]
```

sh

选项

```
-v, --verbose          # 执行的时候显示详细信息
-a, --all              # 关闭所有的交换分区

-V, --version          # 显示版本信息
-h, --help              # 显示帮助文档
```

sh

举例

关闭所有的交换分区

```
[root@bogon ~]$ swapoff -a    #关闭所有交换分区
[root@bogon ~]$ free      #查看内存使用状态
              total        used         free       shared    buffers     cached
Mem:   1659316     678908     980408       0       85608     369308
-/+ buffers/cache:    223992     1435324
Swap:            0           0           0           #swap分区不使用
```

sh

swapon - 激活Linux系统中交换空间

swapon命令 用于激活Linux系统中交换空间，Linux系统的内存管理必须使用交换区来建立虚拟内存。

在指定的设备上启用交换分区，使用的设备或文件由专用文件参数提供。它可以是“-L label”或“-U UUID”，以指示一个设备的标签或UUID。对swapon的调用通常发生在系统引导脚本中，使所有交换设备都可用，因此分页和交换活动交叉在多个设备和文件之间。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
swapon -s [-h] [-V]
swapon [-f] [-p priority] [-v] specialfile...
swapon -a [-e] [-f] [-v]
```

选项

```
-v, --verbose          # 执行的时候显示详细信息
-a, --all              # 打开所有的交换分区。除具有“noauto”选项的设备外，所有标记为“/etc/fstab”中“
-e, --ifexists         # 跳过不存在的设备
-f, --fixpgsz          # 如果交换空间的页大小与当前运行的内核不匹配，则重新初始化(exec/sbin/mkswap)
-L label               # 启动指定label的交换分区，为此，需要访问/proc/分区。
-p                     # 设置优先权0~32767，数字越大，优先权越高。将“pri=value”添加到与“swapon -a
-s, --summary          # 按设备显示交换用途摘要，等价于“cat /proc/swaps “
-U uuid                # 启动指定uuid的交换分区

--help                 # 显示帮助文档
--version              # 显示命令版本信息
```

说明

您不应该在有漏洞的文件上使用swapon。交换NFS可能无法工作。swapon会自动检测并用旧的软件挂起数据重写交换空间签名(例如S1SUSPEND, S2SUSPEND,)。问题是，如果我们不这样做，那么我们将在下一次尝试取消挂起时数据损坏。

举例

启动sdb4

sh

```
[root@localhost ~]$ swap      #创建交换分区
Setting up swap space version 1, size = 16380 KiB
LABEL=wj, UUID=aec14728-0f33-4676-8bef-612ea1bdf985
[root@localhost ~]$ swapon /dev/sdb4      #启用交换分区
[root@localhost ~]$ swapon -s /dev/sdb4    #查看信息
Filename          Type      Size   Used  Priority
/dev/sdb4        partition 16376  0       -1
[root@localhost ~]$
```

以标签的形式启动

sh

```
[root@localhost ~]$ blkid -c /dev/null    #找到交换分区的UUID
/dev/sda1: UUID="059facc9-c58e-42d0-b8f5-7644c4574888" TYPE="ext4"
/dev/sda2: UUID="z3WRza-EIUL-dib9-7CGq-zRYt-DMdL-jT2ld9" TYPE="LVM2_member"
/dev/mapper/VolGroup-lv_root: UUID="9ad51e8e-3700-45a8-a195-531a95ff717d" TYPE="ext4"
/dev/mapper/VolGroup-lv_swap: UUID="2ebcaf57-3c6a-45be-8f34-3326d1fa1762" TYPE="swap"
[root@localhost ~]$ free      #查看
              total        used        free      shared  buffers  cached
Mem:      1659316     1601620      57696          0     261664   1075780
-/+ buffers/cache:     264176     1395140
Swap:          0          0          0
[root@localhost ~]$ swapon -U 2ebcaf57-3c6a-45be-8f34-3326d1fa1762    #启动交换分区
[root@localhost ~]$ free      #查看
              total        used        free      shared  buffers  cached
Mem:      1659316     1604132      55184          0     261684   1075780
-/+ buffers/cache:     266668     1392648
Swap:     2940920          0     2940920
[root@localhost ~]$
```

sync - 用于强制被改变的内容立刻写入磁盘

sync命令 用于强制被改变的内容立刻写入磁盘，更新超块信息。

在Linux/Unix系统中，在文件或数据处理过程中一般先放到内存缓冲区中，等到适当的时候再写入磁盘，以提高系统的运行效率。sync命令则可用来强制将内存缓冲区中的数据立即写入磁盘中。用户通常不需执行sync命令，系统会自动执行update或bdflush操作，将缓冲区的数据写入磁盘。只有在update或bdflush无法执行或用户需要非正常关机时，才需手动执行sync命令。

将内存缓冲区的数据强制写入到磁盘，通常Linux系统会将数据写入到内存缓冲区，然后一次性将缓冲区的数据写入到磁盘。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
sync [选项]
```

sh

选项

```
-d, --data          # 只同步文件数据，不同步不必要的元数据  
-f, --file-system # 同步包含这些文件的文件系统  
--help             # 显示帮助文档  
--version          # 显示命令版本信息
```

sh

buffer与cache

- buffer：为了解决写磁盘的效率
- cache：为了解决读磁盘的效率

linux系统为了提高读写磁盘的效率，会先将数据放在一块buffer中。在写磁盘时并不是立即将数据写到磁盘中，而是先写入这块buffer中了。此时如果重启系统，就可能造成数据丢失。

sync命令用来flush文件系统buffer，这样数据才会真正的写到磁盘中，并且buffer才能够释放出来，flush就是用来清空buffer。sync命令会强制将数据写入磁盘中，并释放该数据对应的buffer，所以常常会在写磁盘后输入sync命令来将数据真正的写入磁盘。

如果不去手动的输入sync命令来真正的去写磁盘，linux系统也会周期性的去sync数据。

sysctl - 时动态地修改内核的运行参数

sysctl命令 被用于在内核运行时动态地修改内核的运行参数，可用的内核参数在目录/proc/sys中。它包含一些TCP/ip堆栈和虚拟内存系统的高级选项，这可以让有经验的管理员提高引人注目的系统性能。用sysctl可以读取设置超过五百个系统变量。

sysctl指令用来修改正在运行的内核参数，可以修改的参数都保存在/proc/sys/目录中，修改会立即生效。Linux中的sysctl支持需要Procs。您可以使用sysctl来读取和写入sysctl数据。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sysctl [-n] [-e] variable ...
sysctl [-n] [-e] [-q] -w variable=value ...
sysctl [-n] [-e] [-q] -p <filename>
sysctl [-n] [-e] -a
sysctl [-n] [-e] -A
```

sh

选项

```
-n          # 显示内核参数的值，不显示其名称
-e          # 忽略错误
-N          # 只显示名字
-q          # 不在标准输出显示值
-w          # 复制的时候需要
-p          # 加载sysctl的设置情况
-a, -A, --all # 显示所有内核参数的值和名字
variable    # 读取的变量名，可以用“/”做分隔符
variable=value # 如果value包含由shell解析的引号或字符，则可能需要将该值括在双引号中。这需要使用-w参数。
```

sh

举例

```
[sogrey@bogon ~]$ sysctl
Usage:
  sysctl [options] [variable[=value] ...]

Options:
  -a, --all           display all variables
  -A                 alias of -a
  -X                 alias of -a
  --deprecated       include deprecated parameters to listing
```

sh

```
-b, --binary      print value without new line
-e, --ignore     ignore unknown variables errors
-N, --names       print variable names without values
-n, --values      print only values of the given variable(s)
-p, --load[=<file>] read values from file
-f               alias of -p
    --system     read values from all system directories
-r, --pattern <expression>
                select setting that match expression
-q, --quiet      do not echo variable set
-w, --write       enable writing a value to variable
-o               does nothing
-x               does nothing
-d               alias of -h

-h, --help        display this help and exit
-V, --version     output version information and exit
```

For **more** details see **sysctl(8)**.

```
[sogrey@bogon ~]$ sysctl --all # 显示所有的内核参数名字和值
abi.vsyscall32 = 1
crypto.fips_enabled = 0
debug.exception-trace = 1
debug.kprobes-optimization = 1
debug.panic_on_rcu_stall = 0
dev.cdrom.autoclose = 1
dev.cdrom.autoeject = 0
dev.cdrom.check_media = 0
dev.cdrom.debug = 0
dev.cdrom.info = CD-ROM information, Id: cdrom.c 3.20 2003/12/17
net.ipv6.conf.lo.use_optimistic = 0
net.ipv6.conf.lo.use_tempaddr = -1
net.ipv6.conf.virbr0.accept_dad = 1
net.ipv6.conf.virbr0.accept_ra = 0
net.ipv6.conf.virbr0.accept_ra_defrtr = 1
net.ipv6.conf.virbr0.accept_ra_pinfo = 1
net.ipv6.conf.virbr0.accept_ra_rt_info_max_plen = 0
net.ipv6.conf.virbr0.accept_ra_rtr_pref = 1
net.ipv6.conf.virbr0.accept_redirects = 0
net.ipv6.conf.virbr0.accept_source_route = 0
net.ipv6.conf.virbr0.autoconf = 0
net.ipv6.conf.virbr0.dad_transmits = 1
net.ipv6.conf.virbr0.disable_ipv6 = 1
net.ipv6.conf.virbr0.force_mld_version = 0
net.ipv6.conf.virbr0.force_tllao = 0
net.ipv6.conf.virbr0.forwarding = 0
net.ipv6.conf.virbr0.hop_limit = 64
net.ipv6.conf.virbr0.max_addresses = 16
net.ipv6.conf.virbr0.max_desync_factor = 600
net.ipv6.conf.virbr0.mc_forwarding = 0
...
[sogrey@bogon ~]$
```

tac - 连接多个文件并以行为单位反向打印到标准输出

将指定文件中的行，按照反序方式显示。

主要用途

- 按行为单位反向显示文件内容，如果没有文件或文件为-则读取标准输入。
- 处理多个文件时，依次将每个文件反向显示，而不是将所有文件连在一起再反向显示。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
tac [选项] file
```

sh

选项

```
-b, --before          # 在之前而不是之后连接分隔符。  
-r, --regex           # 将分隔符作为基础正则表达式（BRE）处理。  
-s, --separator=STRING # 使用STRING作为分隔符代替默认的换行符。  
  
--help                # 显示帮助文档  
--version             # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ cat test2.txt  
123  
23  
212  
[sogrey@bogon newDir3]$ tac test2.txt # 反序显示  
212  
23  
123  
[sogrey@bogon newDir3]$ tac -b test2.txt # 将分隔符“回车”放在文件最前面，导致最前面是两个空行，最后一个空行是“回车”  
  
212  
23123[sogrey@bogon newDir3]$
```

注意

1. 该命令是GNU coreutils包中的命令，相关的帮助信息请查看man -s 1 tac或info coreutils 'tac invocation'。
2. 关于基础正则表达式（BRE）的内容，详见man -s 1 grep的REGULAR EXPRESSIONS段落。

tail - 在屏幕上显示指定文件的末尾若干行

显示文本文件尾部的部分内容， 默认显示最后10行。

tail命令 用于输入文件中的尾部内容。

- 默认在屏幕上显示指定文件的末尾10行。
- 处理多个文件时会在各个文件之前附加含有文件名的行。
- 如果没有指定文件或者文件名为-， 则读取标准输入。
- 如果表示字节或行数的NUM值之前有一个+号， 则从文件开头的第NUM项开始显示， 而不是显示文件的最后NUM项。
- NUM值后面可以有后缀：
 - b : 512
 - kB : 1000
 - k : 1024
 - MB : 1000 * 1000
 - M : 1024 * 1024
 - GB : 1000 * 1000 * 1000
 - G : 1024 * 1024 * 1024
 - T、P、E、Z、Y等以此类推。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
tail [选项] file
```

sh

选项

```
-c, --bytes=NUM          # 输出文件尾部的NUM (NUM为整数) 个字节内容。
-f, --follow[={name|descriptor}] # 显示文件最新追加的内容。“name”表示以文件名的方式监视文件的变化。
-F                         # 与“--follow=name --retry” 功能相同。
-n, --line=NUM            # 输出文件的尾部NUM (NUM位数字) 行内容。
--pid=<进程号>           # 与“-f”选项连用，当指定的进程号的进程终止后，自动退出tail命令。
-q, --quiet, --silent      # 当有多个文件参数时，不输出各个文件名。
--retry                    # 即是在tail命令启动时，文件不可访问或者文件稍后变得不可访问，都始终尝试
-s, --sleep-interval=<秒数> # 与“-f”选项连用，指定监视文件变化时间间隔的秒数。
-v, --verbose              # 当有多个文件参数时，总是输出各个文件名。

--help                     # 显示帮助文档
--version                  # 显示命令版本信息
```

举例

```
[sogrey@bogon newDir3]$ ls  
1.txt students.txt test2.txt test.txt  
[sogrey@bogon newDir3]$ cat test2.txt  
123  
23  
212  
[sogrey@bogon newDir3]$ tail -n 1 test2.txt # 显示最后一行  
212  
[sogrey@bogon newDir3]$ tail -c 6 test2.txt # 显示最后6个字符  
3  
212  
[sogrey@bogon newDir3]$
```

sh

tailf - 在屏幕上显示指定文件的末尾若干行内容，通常用于日志文件的跟踪输出

tailf命令几乎等同于tail -f，严格说来应该与tail --follow=name更相似些。当文件改名之后它也能继续跟踪，特别适合于日志文件的跟踪（follow the growth of a log file）。与tail -f不同的是，如果文件不增长，它不会去访问磁盘文件。tailf特别适合那些便携机上跟踪日志文件，因为它能省电，因为减少了磁盘访问。tailf命令不是个脚本，而是一个用C代码编译后的二进制执行文件，某些Linux安装之后没有这个命令。

tailf和tail -f的区别

- tailf 总是从文件开头一点一点的读，而tail -f 则是从文件尾部开始读
- tailf 检查文件增长时，使用的是文件名，用stat系统调用；而tail -f 则使用的是已打开的文件描述符；
注：tail 也可以做到类似跟踪文件名的效果；但是tail总是使用fstat系统调用，而不是stat系统调用；结果就是：默认情况下，当tail的文件被偷偷删除时，tail是不知道的，而tailf是知道的。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
tailf logfile # 动态跟踪日志文件logfile，最初的时候打印文件的最后10行内容。 sh
```

选项

```
-n, --lines NUMBER # 输出最后数行  
-NUMBER           # 与NUMBER相同 ` -n NUMBER '  
  
--help            # 显示帮助文档  
--version         # 显示命令版本信息 sh
```

举例

```
[sogrey@bogon newDir3]$ tailf test2.txt  
123  
23  
212  
  
^C  
[sogrey@bogon newDir3]$ tailf -n 1 test2.txt  
212 sh
```


tar - 将许多文件一起保存至一个单独的磁带或磁盘归档，并能从归档中单独还原所需文件

创建归档、解压或者压缩文件。

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）。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
tar [选项] 文件
```

sh

选项

sh

```

-A, --catenate, --concatenate    # 在归档最后追加tar文件
-c, --create                      # 创建归档
-d, --diff, --compare             # 查找归档和文件系统之间的差异
--delete                           # 从档案中删除(不是在磁带上! )
-f file                            # 指定归档名字
-r, --append                        # 在归档的最后追加文件
-t, --list                          # 列出归档的文件列表
--telst-label                      # 测试存档卷标签并退出
-u, --update                         # 只有文件比归档中的内容新的时候, 才会追加
-x, --extract, --get                # 从归档中释放文件
-C dir , --directory=DIR           # 更改工作目录
-j, --bzip2                          # 从bzip2中解压或者压缩
-J, -xz                             # 通过xz过滤存档
-p, --preserve-permissions         # 提取有关文件权限的信息(超级用户默认)
-z, --gzip                            # 从gzip中解压或者压缩
-v, --verbose                         # 显示详细信息

--help                                # 显示帮助文档
--version                            # 显示命令版本信息

```

操作修饰符

sh

```

--check-device                      # 创建增量档案时检查设备编号(默认)
-g, --listed-incremental=FILE      # 处理新的GNU格式增量备份
-G, --incremental                   # 处理旧的GNU格式增量备份
--ignore-failed-read                # 不要在不可读文件上使用非零退出
--level=NUMBER                      # 创建列表的转储级别-增量存档
-n, --seek                           # 档案是可查找的
--no-check-device                   # 创建增量档案时不检查设备编号
--no-seek                           # 档案是不可寻的
--occurrence[=NUMBER]               # 只处理存档中每个文件的第NUMBER次出现的文件; 此选项仅与--delete、--diff
--sparse-version=MAJOR[.MINOR]       # 将稀疏格式的版本
-S, --sparse                          # 有效处理稀疏文件

```

覆盖控制

sh

```

-k, --keep-old-files                # 解压缩时不要替换现有文件, 将它们视为错误
--skip-old-files                   # 解压缩时不要替换现有文件, 请悄悄跳过它们
--keep-newer-files                 # 不要替换比其存档副本更新的现有文件
--no-overwrite-dir                  # 保存现有目录的元数据
--overwrite                          # 解压时替换现有的文件
--overwrite-dir                     # 提取时覆盖现有目录的元数据(默认)
--recursive-unlink                  # 提取目录之前的空层次结构
--remove-files                      # 将文件添加到存档后删除它们
-U, --unlink-first                 # 在提取每个文件之前删除它。
-W, --verify                         # 在编写存档后尝试验证它

```

选择输出流

sh

```
--ignore-command-error          # 忽略子级退出代码
--no-ignore-command-error      # 将子级的非零退出码视为错误
-O, --to-stdout                # 将文件解压缩到标准输出
--to-command=COMMAND           # 将文件解压缩到另一个处理文件属性的程序。
--acls                         # 将ACLs保存到存档中
--atime-preserve[=METHOD]       # 保留转储文件的访问时间，方法是在读取后恢复时间(METHOD='replace'；默认值
--delay-directory-restore      # 将已提取目录的设置修改时间和权限延迟到提取结束
--group=NAME                    # 强制NAME作为添加文件的组
--mode=CHANGES                  # 强制(符号)添加的文件的模式为CHANGES
--mtime=DATE-OR-FILE            # 设置文件的mtime
-m, --touch                     # 不要提取文件修改时间
--no-acls                      # 不要从存档中提取ACLs
--no-delay-directory-restore   # 取消选项"--delay-directory-restore"
--no-same-owner                 # 自解压缩文件(普通用户默认)
--no-same-permissions           # 从存档提取权限时应用用户的umask(普通用户默认)
--no-selinux                    # 不要提取SELinux上下文
--no-xattrs                     # 不要从归档文件中提取用户/root的xattrs
--numeric-owner                 # 始终使用数字作为用户名/组名。
--owner=NAME                    # 强制NAME作为添加文件的所有者
-p, --preserve-permissions, --same-permissions # 提取有关文件权限的信息(超级用户默认)
--preserve                       # 同时具备“-p”和“-s”选项
--same-owner                     # 尝试提取与存档中存在的所有权相同的文件(默认为超级用户)。
-s, --preserve-order, --same-order # 排序要提取的名称以匹配存档
--selinux                        # 保存SELinux上下文到存档
-xattrs                          # 保存用户/rootd xattrs到存档
```

设备选择和切换

sh

```
-f, --file=ARCHIVE             # 使用存档文件或设备ARCHIVE
--force-local                   # 存档文件是本地的，即使它有冒号。
-F
--info-script=NAME              # 在每个磁带的末尾运行脚本
--new-volume-script=NAME         # 在写入NUMBERx1024字节后更改磁带
L, --tape-length=NUMBER         # 在写入NUMBERx1024字节后更改磁带
-M, --multi-volume               # 创建/列表/提取多卷存档
--rmt-command=COMMAND            # 使用给定的rmt命令代替rmt
--rsh-command=COMMAND            # 使用给定的远程命令代替rsh
--volno-file=FILE                # 使用/更新文件中的卷号
```

设备块

sh

```
-b, --blocking-factor=BLOCKS    # 每个记录BLOCKSx512个字节
-B, --read-full-records          # 重新定义block大小
-i, --ignore-zeros                # 忽略归档中的零块(意为EOF)
--record-size=NUMBER               # 每个记录NUMBER字节，倍数为512
```

归档格式选择

sh

```

-H, --format=FORMAT          # 使用给定的格式创建归档, 格式可以是:
                             # gnu, GNU tar 1.13.x格式
                             # oldgnu, 按tar<=1.12表示的GNU格式
                             # pax, POSIX 1003.1-2001 (pax)格式
                             # posix, 和pax一样
                             # ustar, POSIX 1003.1-1988 (ustar)格式
                             # v7, 旧的v7 tar格式
--old-archive, --portability # 等价于“--format=v7”
--pax-option=
keyword[[:]=value][,keyword[[:]=value]]...
                             # 控制pax关键字
--posix                      # 等价于“--format=posix”
V, --label=TEXT              # 使用卷名TEXT创建存档; 在列表/解压缩时, 使用TEXT作为卷名的全局模式

```

压缩选项

sh

```

-a, --auto-compress          # 使用归档后缀确定压缩程序
-I, --use-compress-program=PROG # 通过PROG过滤(必须接受-d)
-j, --bzip2                   # 通过bzip2过滤存档
-J, --xz                      # 通过xz过滤存档
--lzip                       # 通过lzip过滤存档
--lzma                        # 通过lzma过滤存档
--lzop, --no-auto-compress    # 不要使用归档后缀来确定压缩程序。
-z, --gzip, --gunzip, --ungzip # 通过gzip过滤存档
-Z, --compress, --uncompress   # 通过compress过滤存档

```

本地文件选择

sh

```
--add-file=FILE          # 将给定的文件添加到归档文件中(如果其名称以‘-’开头, 则有用)
--backup[=CONTROL]       # 删除前的备份, 选择版本控制
-C, --directory=DIR      # 切换到目录dir
--exclude=PATTERN        # 排除指定文件
--exclude-backups        # 排除备份和锁定的文件
--exclude-caches          # 排除包含CACHEDIR.TAG的目录的内容, 标记文件本身除外
--exclude-caches-all      # CACHEDIR.TAG的目录
--exclude-caches-under    # 排除包含CACHEDIR.TAG目录下的所有内容
--exclude-tag=FILE        # 排除包含FILE的目录的内容, 但文件本身除外
--exclude-tag-all=FILE     # 排除包含FILE的目录
--exclude-tag-under=FILE   # 排除包含FILE的目录下的所有内容
--exclude-vcs             # 排除版本控制的系统目录
-h, --dereference        # 遵循符号链接; 归档和转储它们指向的文件
--hard-dereference        # 遵循硬链接; 归档和转储它们指向的文件
-K, --starting-file=MEMBER-NAME # 从存档中的成员MEMBER-NAME开始
--newer-mtime=DATE         # 比较仅更改数据的日期和时间
--no-null                  # 关闭“--null”选项
--no-recursion             # 避免在目录中自动递归
--no-unquote                # 不要取消引用-T读取的文件名
--null, -T                  # 读取以空结尾的名称, 禁用-C
-N, --newer=DATE-OR-FILE    # 只保存比DATE-OR-FILE新的文件
--after-date=DATE-OR-FILE    # 创建存档时保持本地文件系统
--one-file-system           # 不要从文件名中去掉前导‘/’
--P, --absolute-names        # 递归操作, 默认的
--suffix=STRING              # 删除之前备份, 重写通常的后缀(‘~’除非被环境变量SIMPLE_BACKUP_SUFFIX覆盖)
-T, --files-from=FILE        # 获取要从FILE中提取或创建的名称
--unquote                   # 取消引用-T读取的文件名(默认)
-X, --exclude-from=FILE      # 排除FILE中列出的模式
```

文件名字转换

sh

```
--strip-components=NUMBER    # 提取文件名中的带NUM前导组件
--transform=EXPRESSION,        # 使用sed替换EXPRESSION转换文件名
--xform=EXPRESSION           # 模式匹配文件名开始
--anchored                    # 忽略大小写
--ignore-case                 # 任何“/”之后的模式匹配(exclude的默认)
--no-anchored                 # 大小写敏感, 默认。
--no-wildcards                # 逐字字符串匹配
--no-wildcards-match-slash    # 通配符不匹配“/”
--wildcards                   # 使用通配符(默认)
--wildcards-match-slash       # 通配符匹配“/”, 默认
```

多信息输出

```
--checkpoint[=NUMBER]          # 每NUMBERth个记录显示一次进度
--checkpoint-action=ACTION      # 对每个检查点执行操作ACTION
--index-file=FILE               # 将详细信息输出到FILE
-l, --check-links              # 如果不是所有链接都已转储，则打印一条消息
--no-quote-chars=STRING        # 禁止引用STRING中的字符
--quote-chars=STRING           # 引用STRING中的字符
--quoting-style=STYLE           # 设置名称引用样式
-R, --block-number              # 在存档中显示每条消息的块号
--show-defaults                 # 显示tar默认值
--show-omitted-dirs             # 列出或提取时，列出不匹配搜索条件的每个目录
--show-transformed-names         # 转换后显示文件名或存档名称
--show-stored-names              # 在处理归档文件后使用参数“-”打印总字节；在发送此信号时使用参数打印总字节；
--totals[=SIGNAL]                # 使用UTC格式打印文件修改日期
--utc                           # 详细列出已处理的文件
--warning=KEYWORD                # 警告控制
-w, --interactive, --confirmation # 要求确认每一项行动
```

备份后缀为‘~’，除非设置为“--suffix”或“SIMPLE_BACKUP_SUFFIX”。版本控制可以用“--backup”或“VERSION_CONTROL”设置，值为：

- 1) none, off, 从不备份。
- 2) t, numbered, 数字备份。
- 3) nil, existing, 如果存在数字备份，那么使用数字备份；否则，使用普通备份。
- 4) never, simple, 简单备份。

举例

```
- z: 有gzip属性的
- j: 有bz2属性的
- Z: 有compress属性的
- v: 显示所有过程
- O: 将文件解开到标准输出

tar -cf archive.tar foo bar    # 从文件 foo 和 bar 创建归档文件 archive.tar。
tar -tvf archive.tar           # 详细列举归档文件 archive.tar 中的所有文件。
tar -xf archive.tar            # 展开归档文件 archive.tar 中的所有文件。
```

下面的参数-f是必须的

- -f: 使用档案名字，切记，这个参数是最后一个参数，后面只能接档案名。

```

tar -cf all.tar *.jpg
# 这条命令是将所有.jpg的文件打成一个名为all.tar的包。-c是表示产生新的包，-f指定包的文件名。

tar -rf all.tar *.gif
# 这条命令是将所有.gif的文件增加到all.tar的包里面去。-r是表示增加文件的意思。

tar -uf all.tar logo.gif
# 这条命令是更新原来tar包all.tar中logo.gif文件，-u是表示更新文件的意思。

tar -tf all.tar
# 这条命令是列出all.tar包中所有文件，-t是列出文件的意思
tar -cvf archive.tar foo bar # 从文件foo和bar创建archive.tar。
tar -tvf archive.tar          # 详细列出archive.tar中的所有文件。
tar -xf archive.tar          # 从archive.tar提取所有文件。

```

zip格式

- 压缩: zip -r [目标文件名].zip [原文件/目录名]
- 解压: unzip [原文件名].zip

注: -r参数代表递归

tar格式（该格式仅仅打包，不压缩）

- 打包: tar -cvf [目标文件名].tar [原文件名/目录名]
- 解包: tar -xvf [原文件名].tar

注: c参数代表create（创建），x参数代表extract（解包），v参数代表verbose（详细信息），f参数代表filename（文件名），所以f后必须接文件名。

tar.gz格式

- 方式一: 利用前面已经打包好的tar文件，直接用压缩命令。
 - 压缩: gzip [原文件名].tar
 - 解压: gunzip [原文件名].tar.gz
- 方式二: 一次性打包并压缩、解压并解包
 - 打包并压缩: tar -zcvf [目标文件名].tar.gz [原文件名/目录名]
 - 解压并解包: tar -zxvf [原文件名].tar.gz

注: z代表用gzip算法来压缩/解压。

tar.bz2格式

- 方式一: 利用已经打包好的tar文件，直接执行压缩命令：
 - 压缩: bzip2 [原文件名].tar
 - 解压: bunzip2 [原文件名].tar.bz2
- 方式二: 一次性打包并压缩、解压并解包
 - 打包并压缩: tar -jcvf [目标文件名].tar.bz2 [原文件名/目录名]
 - 解压并解包: tar -jxvf [原文件名].tar.bz2

注：小写j代表用bzip2算法来压缩/解压。

tar.xz格式

- 方式一：利用已经打包好的tar文件，直接用压缩命令：
 - 压缩：xz [原文件名].tar
 - 解压：unxz [原文件名].tar.xz
- 方式二：一次性打包并压缩、解压并解包
 - 打包并压缩：tar -Jcvf [目标文件名].tar.xz [原文件名/目录名]
 - 解压并解包：tar -Jxvf [原文件名].tar.xz

注：大写J代表用xz算法来压缩/解压。

tar.Z格式（已过时）

- 方式一：利用已经打包好的tar文件，直接用压缩命令：
 - 压缩：compress [原文件名].tar
 - 解压：uncompress [原文件名].tar.Z
- 方式二：一次性打包并压缩、解压并解包
 - 打包并压缩：tar -Zcvf [目标文件名].tar.Z [原文件名/目录名]
 - 解压并解包：tar -Zxvf [原文件名].tar.Z

注：大写Z代表用ncompress算法来压缩/解压。另，ncompress是早期Unix系统的压缩格式，但由于ncompress的压缩率太低，现已过时。

jar格式

- 压缩：jar -cvf [目标文件名].jar [原文件名/目录名]
- 解压：jar -xvf [原文件名].jar

注：如果是打包的是Java类库，并且该类库中存在主类，那么需要写一个META-INF/MANIFEST.MF配置文件，内容如下：

```
Manifest-Version: 1.0
Created-By: 1.6.0_27 (Sun Microsystems Inc.)
Main-class: the_name_of_the_main_class_should_be_put_here
```

然后用如下命令打包：

```
jar -cvfm [目标文件名].jar META-INF/MANIFEST.MF [原文件名/目录名]
```

这样以后就能用“java -jar [文件名].jar”命令直接运行主类中的public static void main方法了。

7z格式

- 压缩：7z a [目标文件名].7z [原文件名/目录名]
- 解压：7z x [原文件名].7z

注：这个7z解压命令支持rar格式，即：

```
7z x [原文件名].rar
```

sh

其它例子

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
```

sh

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。

解压目录

去掉第一层目录结构，要出除第二层，--strip-components 2

```
tar -xvf portal-web-v2.0.0.tar --strip-components 1 -C 指定目录
```

sh

查阅上述tar包内有哪些文件：

```
tar -ztvf log.tar.gz
```

sh

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
```

sh

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
```

sh

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
```

sh

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
```

sh

备份文件夹内容是排除部分文件:

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

sh

打包文件之后删除源文件:

```
tar -cvf test.tar test --remove-files
```

sh

其实最简单的使用 tar 就只要记忆底下的方式即可:

- 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

tee - 从标准输入读取数据并重定向到标准输出和文件

将标准输入的内容复制到指定的文件中，同时在标准输出中显示。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
tee [选项] [files]
```

sh

选项

```
-a, --append          # 追加模式，并不覆盖  
-i, --ignore-interrupts # 忽略终端信号  
  
--help               # 显示帮助文档  
--version            # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon demos]$ cat test5.txt
石家庄今日新增16例确诊病例
北京一确诊者隐瞒行程不配合流调
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
特朗普夫人发文谴责国会暴乱
山西晋中新增2例无症状感染者
特朗普夫人发文谴责国会暴乱
[sogrey@bogon demos]$ sort test5.txt | tee test50.txt # 排序之后保存到test50.c，并且送到标准输出
北京一确诊者隐瞒行程不配合流调
北京一确诊者隐瞒行程不配合流调
理塘文旅公司回应丁真抽烟
山西晋中新增2例无症状感染者
石家庄今日新增16例确诊病例
特朗普夫人发文谴责国会暴乱
特朗普夫人发文谴责国会暴乱
特朗普夫人发文谴责国会暴乱
中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon demos]$ cat test50.txt # 查看内容
北京一确诊者隐瞒行程不配合流调
北京一确诊者隐瞒行程不配合流调
理塘文旅公司回应丁真抽烟
山西晋中新增2例无症状感染者
石家庄今日新增16例确诊病例
特朗普夫人发文谴责国会暴乱
特朗普夫人发文谴责国会暴乱
特朗普夫人发文谴责国会暴乱
中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon demos]$
```

tload - 显示系统负载状况

tload命令 以图形化的方式输出当前系统的平均负载到指定的终端。假设不给予终端机编号，则会在执行tload指令的终端机显示负载情形。

tload指令以字符的方式显示当前系统的平均负载情况。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
tload [-V] [-s scale] [-d delay] [tty]
```

sh

选项

```
-d          # 显示更新时间间隔  
-s          # 显示图表的垂直刻度单位  
-V          # 显示版本信息
```

sh

举例

```
[sogrey@bogon ~]$ tload -d 5 -s 1  
2.63, 1.79, 1.24  
2.60, 1.82, 1.25  
-----
```

sh

tmpwatch - 删 除 最 近 一 段 时 间 没 有 访 问 的 文 件

删除最近一段时间没有访问的文件，时间以小时为单位，节省磁盘空间。tmpwatch递归删除给定时间未被访问的文件。通常，它用于清理用于临时保存空间(如/tmp)的目录。当更改目录时，tmpwatch对可能的争用条件非常敏感，如果检测到错误，就会退出。它不遵循它正在清理的目录中的符号链接(即使给出一个符号链接作为它的参数)，它不会切换文件系统，跳过根用户的lost+found目录，只删除空目录、常规文件和符号链接。

默认情况下，tmpwatch根据文件的`atime`(访问时间)，而不是它们的`Mtime`(修改时间)来确定文件的日期。如果文件在 `ls -l` 暗示应该删除时没有被移除，请使用 `ls -u` 检查它们的数据，以确定这是否解释了问题的原因。

如果指定了 `--atime`，`--ctime` 或者 `--mtime` 选项，那么删除文件的时间由他们中的最大值决定。如果 `--dirmtime` 选项意味着忽略目录的`atime`，即使使用了 `-atime` 选项。

我在 CentOS Linux 8 上测试 需要单独安装。

适 用 范 围

RedHat	RHEL	Ubuntu	Debian	Deepin	SUSE	openSUSE
Fedora	Linux Mint	Alpine Linux	Arch Linux			

语 法

```
tmpwatch [选项] time file
tmpwatch [-u|-m|-c] [-MUadfqstvx] [--verbose] [--force] [--all] [--nodirs]
          [--nosymlinks] [--test] [--fuser] [--quiet] [--atime|--mtime|--ctime]
          [--dirmtime] [--exclude path] [--exclude-user user] time dirs
```

sh

选 项

```
-u, --atime          # 根据文件的atime(访问时间)做出删除文件的决定。  
-m, --mtime          # 请注意, 定期更新的文件系统扫描使目录保持最近的状态。  
-c, --ctime          # 根据文件的ctime(Inode Changing Time)而不是  
# atime来决定删除文件; 对于目录, 根据mtime做出决定  
-M, --dirmtime        # 根据目录的Mtime(修改时间)而不是atime作出删除  
# 目录的决定; 完全忽略目录的atime  
-a, -all             # 删除所有文件类型, 而不仅仅是常规文件、  
# 符号链接和目录。  
-d, --nodirs          # 不要尝试删除目录, 即使它们是空的。  
-f, --force            # 强制删除, 即使root用户没有写的权利  
-l, --nosymlinks       # 不删除符号链接  
-q, --quite            # 只报告致命错误  
-s, --fuser             # 尝试在删除文件之前使用“fuser”命令查看文件是否已打开。  
# 默认情况下未启用。在某些情况下确实有帮助, 但不是全部。  
# 依赖于/sbin中安装的fuser。不支持HPUX或Solaris  
-t, --test              # 不删除, 只是演示要做什么  
-U, --exclude-user=user  # 不删除指定所有者的文件, 可以指定用户名, 也可以指定用户ID  
-v, --verbose           # 显示详细信息  
-x, --exclude            # 跳过目录及其内部文件, 如果路径不存在,  
# 则它必须是不包含符号链接的绝对路径。  
-X --exclude-pattern=pattern # 跳过路径匹配模式; 如果目录匹配模式,  
# 则其中包含的所有文件也将被跳过。  
# 模式必须匹配不包含符号链接的绝对路径。
```

举例

```
[sogrey@bogon 文档]$ ls  
backup demos test test2.txt test3.txt test4.txt test.txt  
[sogrey@bogon 文档]$ tmpwatch 1 .          # 删除当前目录1小时内没有访问的文件  
  
[sogrey@bogon 文档]$ tmpwatch -U root 1 .  # 不删除root用户的文件
```

top - 显示或管理执行中的程序

top命令 可以实时动态地查看系统的整体运行情况，是一个综合了多方信息监测系统性能和运行信息的实用工具。通过top命令所提供的互动式界面，用热键可以管理。

top指令用来显示Linux的进程信息，这是一个动态显示的过程。top提供运行系统的动态实时视图。它可以显示系统摘要信息以及当前由Linux内核管理的任务列表。所显示的系统摘要信息的类型以及为任务显示的信息的类型、顺序和大小都是用户可配置的，并且可以在重新启动期间使配置持久。

该程序为进程操作提供了有限的交互界面，也为个人配置提供了更广泛的界面-包括其操作的各个方面。虽然TOP是在整个文档中引用的，但您可以随意命名该程序。这个新的名称(可能是别名)将反映在top的显示器上，并在读取和写入配置文件时使用。

当操作top时，最重要的两个键是Help("h"或"?")并退出('q')键。或者，您可以简单地使用传统的中断键('^C')。当您第一次启动top时，您将看到传统的屏幕元素：1)摘要区域；2)消息/提示行；3)列标题；4)任务区域。然而，与之前的top相比，会有一些不同之处。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
top -hv | -abcHimMsS -d delay -n iterations -p pid [, pid ...]
```

选项

```
-a          # 按照内存使用排序
-b          # 批处理模式操作。以“批处理模式”启动top，这对于将输出从top发送到其他程序或文件可能很有用
-c          # 以最后一个记忆中的‘c’状态反转开始。因此，如果top正在显示命令行，那么现在该字段将显示
-d ss.tt   # 设置top监视的时间间隔，默认5s。可以使用小数秒，但是负数却不行
-h          # 显示帮助信息
-H          # 线程取反。从最后一个记忆中的“H”状态开始。当此切换打开时，将显示所有单独的线程。否则，将显示所有共享线程
-i          # 不显示僵尸进程。从最后一个记忆中的“I”状态开始。当此切换关闭时，将不会显示闲置或僵尸进程
-m          # 使用的报告(进程RSS和交换总计数之和)，而不是VIRT
-M          # 显示内存单元
-n          # 设置监控更新次数
-p          # 仅监视指定pid的信息。这个选项可以被给予最多20次，或者您可以提供一个逗号分隔列表和最短前缀
-s          # 安全模式。以强制的安全模式启动top，即使对于root用户也是如此。通过系统配置文件更好地控制安全模式
-S          # 累积时间模式切换。从最后一个记忆中的‘S’状态反转开始。当“累积模式”打开时，每个进程都累加其运行时间
-u          # 只监视具有有效UID或用户名匹配的进程
-U          # 只监视具有给定UID或用户名匹配的进程。这与实际的、有效的、保存的和文件系统UID匹配。
-v          # 显示库版本和使用提示，然后退出。
```

字段/列

字段描述

下面列出了top的可用字段。它们总是与所显示的字母相关联，无论您为它们设置的位置是'o'(Order字段)交互命令。任何字段都可以选择为排序字段，您可以控制它们是按高低排序还是从低到高排序。

字段	说明
PID	任务的唯一进程ID，它定期包装，但从不在零重新启动
PPID	进程的父ID
RUSER	任务所有者的真实用户名
UID	任务所有者的有效用户ID
USER	任务所有者的有效用户名
GROUP	任务所有者的有效组名称
TTY	控制终端的名称。这通常是设备(串口，pty等)。从其中启动进程，并将其用于输入或输出。但是，任务不需要与终端相关联，在这种情况下，您会看到'?'显示
PR	任务优先级
NI	任务的nice值。负的好值意味着更高的优先级，而正的好值则意味着较低的优先级。该字段中的零只意味着在确定任务的可调度性时不会调整优先级。
P	表示最后一次使用的处理器的数字。在真正的SMP环境中，由于内核有意使用弱亲和力，这很可能经常发生变化。此外，运行top的行为可能会打破这种微弱的亲和力，导致更多进程更频繁地更改CPU(因为对CPU时间的额外需求)。
%CPU	自上次屏幕更新以来，任务在经过的CPU时间中所占的份额，表示为总CPU时间的百分比。在真正的SMP环境中，如果"Irix模式"关闭，top将在"Solaris模式"中操作，其中任务的CPU使用量将除以CPU总数。使用"i"交互命令切换"Irix/Solaris"模式
TIME	任务自启动以来使用的总CPU时间。当"累积模式"打开时，每个进程都会列出它及其死子进程使用的CPU时间。使用"S"切换"累积模式"，这是命令行选项和交互式命令。有关此模式的其他信息，请参见"S"交互式命令
TIME+	与"TIME"相同，但通过百分之一秒反映出更多的粒度。
%MEME	任务当前使用的可用物理内存共享
VIRT	任务使用的虚拟内存总量。它包括所有代码、数据和共享库以及已被交换的页面。 (注意：您可以定义STATSIZE=1环境变量，并且VIRT将从/proc/#/state VmSize字段中计算。)
SWAP	每个进程交换值现在从/proc/#/Status VmABP字段中获取。
RES	任务使用的非交换物理内存。
CODE	用于可执行代码的物理内存量，也称为"文本驻留集"大小或TRS。
DATA	用于可执行代码以外的物理内存量，也称为"数据驻留集"大小或DRS。
SHR	任务使用的共享内存量。它只是反映了可能与其他进程共享的内存。
nFLT	任务发生的主要页面错误数。当进程试图读取或写入当前不在其地址空间中的虚拟页时，会发生页错误。一个主要的页面错误是当磁盘访问涉及到使该页可用时。
	自上次写入磁盘以来已修改的页数。页面必须写入磁盘，才能将相应的物理内存位置

nDRT 字段	说明
S	任务的状态，可以是 D ，不间断睡眠 R ，运行 S ，睡眠 T ，追踪或停止 Z ，僵尸
Command	显示用于启动任务的命令行或关联程序的名称。使用‘c’在命令行和名称之间切换，这既是命令行选项，也是交互式命令。当您选择显示命令行时，没有命令行的进程(如内核线程)将仅以括号中的程序名显示，如本例所示(mdrecoveryd)如果显示的长度太长，无法适应该字段的当前宽度，则这两种显示形式都会受到潜在截断的影响。该宽度取决于所选的其他字段、其顺序和当前屏幕宽度。
WCHAN	根据内核链接映射('System.map')的可用性，该字段将显示任务当前处于休眠状态的内核函数的名称或地址。正在运行的任务将在本列中显示一个'-'。
Flags	此列表示任务的当前调度标志，这些标志以十六进制表示法表示，零被抑制。这些标志正式记录在<linux/disk.h>中。

选择和排序列

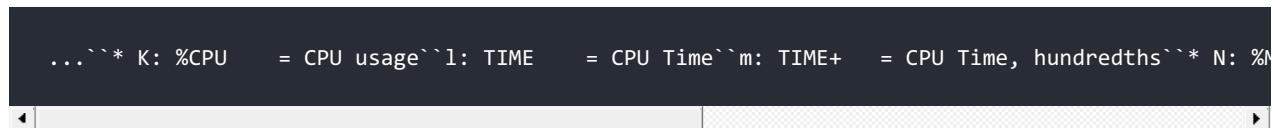
在按下交互命令‘f’(字段选择)或‘o’(顺序字段)后，将显示一个屏幕，其中包含当前字段字符串，后面跟着所有字段的名称和说明。下面是来自TOP的四个窗口/字段组之一的示例字段字符串，以及对所使用的约定的解释：

示例字段字符串：

ANOPQRSTUWXbcdefgilmxyzWHIK

显示字段的顺序对应于该字符串中字母的顺序。

如果字母大写，则相应字段本身将显示为任务显示的一部分(屏幕宽度允许)。这也将用一个领先的星号(*)来表示，如以下摘录所示：



字段选择屏幕---‘f’交互式命令：只需按相应的字母，就可以切换字段的显示。

命令字段屏幕---“o”交互命令：通过按相应的大写字母向左移动一个字段，用小写字母向右移动一个字段。

汇总区域字段

描述CPU统计信息的汇总区域字段被缩写。它们提供了关于在下列方面花费的时间的信息：

us， 用户模式。

sy， 系统模式。

ni， 低级别用户模式

id， 空闲任务。

wa， IO等待。

hi， 服务中断。

si， 服务软中断。

st，偷窃(给其他Domu实例的时间。

交互命令

下面列出的是类别中命令的简短索引。有些命令不止一次出现-它们的含义或范围可能因发出命令的上下文而异。

全局命令

全局交互命令始终可以在全屏模式和交替显示模式下使用.但是，在“安全模式”下运行时，这些交互命令中有些是不可用的。如果您希望预先知道您的顶部是否已被保护，只需请求帮助并查看第二行的系统摘要。

命令	说明
<Enter> or <Space>	刷新显示 这些命令什么也不做，它们只是被忽略了。但是，它们将唤醒顶部，在接收到任何输入后，整个显示将被重新绘制。如果您有较大的延迟间隔并希望看到当前状态，请使用这些键中的任何一个。
? or h	帮助 有两个帮助级别可用。第一个命令将提醒您注意所有基本的交互命令。如果顶部是安全的，屏幕就会缩写。输入“h”或“?”在“帮助”屏幕上，将为那些适用于交替显示模式的交互式命令提供帮助。
=	退出任务限制 移除显示哪些任务的限制。此命令将反转所有可能处于活动状态的‘i’(空闲任务)和‘n’(最大任务)命令。它还提供了从PID监控中的“退出”。有关PID监视的讨论，请参见‘-p’命令行选项。在交替显示模式下操作时，此命令的含义稍宽一些。
A	交替显示模式开关 此命令将在全屏模式和交替显示模式之间切换。交替显示模式和“G”交互命令，以洞察“当前”窗口和字段组。
B	粗体禁用/启用切换 此命令将影响“粗体”终端功能的使用，并更改当前窗口的摘要区域和任务区域。虽然它主要是用于哑巴终端，但它可以在任何时候应用。注意：当这个按钮在单色模式下运行时，整个显示将显示为正常文本。因此，除非‘x’和/或‘y’切换是用相反的强调，就不会有视觉确认他们是均匀的。
d or s	改变延迟时间间隔 系统将提示您在显示更新之间输入延迟时间(以秒为单位)。不允许使用小数秒，但不允许使用负数。输入0导致(几乎)不断更新，显示不令人满意，因为系统和TTY驱动程序试图跟上TOP的要求。延迟值与系统负载成反比，因此要小心设置。如果您想知道当前的延迟时间，只需请求帮助并查看第二行的系统摘要。
G	选择另一个窗口/字段组 您将被提示输入一个介于1到4之间的数字，指定应该成为“当前”窗口的窗口/字段组。你很快就会对这4个窗口感到舒服，特别是在尝试了交替显示模式之后。
I	Irix/Solaris模式切换 当在“Solaris”模式下操作(“I”切换关闭)时，任务的CPU使用量将除以CPU总数。发出此命令后，您将被告知此切换的新状态
u	选择用户 将提示您输入UID或用户名。只显示属于选定用户的进程。此选项与有效UID匹配。
U	选择用户 将提示您输入UID或用户名。只显示属于选定用户的进程。此选项与实际的、有效的、保存的和文件系统UID匹配。
k	杀死任务 系统会提示您输入PID，然后再发送信号。在提示符中反映的默认信号是SIGTERM。但是，您可以通过号码或名称发送任何信号。如果希望中止终止进程，请根据进度执行以下操作之一：1) 在PID提示符下，只需按2) 在信号提示处，键入0
q	退出
r	重新设置任务优先级 系统会提示您输入PID，然后将值设置为NICE。输入一个正值将导致进程失去优先级。相反，负值将导致内核更好地查看进程。
w	写入配置文件 这将节省您的所有选项和切换加上当前显示模式和延迟时间。通过在退出top之前发出此命令，您将能够在以后完全相同的状态下重新启动。
Z	改变颜色映射 这个键将带你到一个单独的屏幕，在那里你可以改变“当前”窗口的颜色，或者所有窗口的颜色。有关此交互式命令的详细信息。
*	以星号(*)显示的命令在“安全模式”中不可用，也不会显示在第1级帮助屏幕上。

摘要区域命令

摘要区域交互命令始终可以在全屏模式和交替显示模式下使用。它们会影响显示的起始行，并将决定消息和提示的位置。这些命令总是只影响“当前”窗口/字段组。如果整个摘要区域已切换到任何窗口，则只剩下消

息行。这样，您就可以最大限度地使用可用的任务行，但是(暂时)在全屏模式下牺牲了程序名，或者在交替显示模式下牺牲了当前的窗口名。

命令	说明
'T'	切换负载平均/正常运行时间 这也是在全屏模式下操作时包含程序名称(可能是别名)的行，或者在交替显示模式下操作时包含“当前”窗口名称的行。
'm'	切换内存/交换使用 此命令影响两个摘要区域行。
't'	切换任务/CPU状态 此命令会影响从2到多个摘要区域行，这取决于“1”切换的状态以及top是否在真正的SMP下运行。
'1'	切换单个/分离CPU状态 此命令影响“t”命令的CPU状态部分的显示方式。虽然这种切换主要是为了服务大规模并行的SMP机器，但它并不仅限于SMP环境。

任务区域命令

任务区域交互命令总是在全屏模式下可用。如果“当前”窗口的任务显示已关闭，则任务区域交互命令在交替显示模式下是不可用的。

I) 任务窗口的外观

以下命令也将受到全局'b'(粗体禁用)切换状态的影响。

命令	说明
'b'	粗体/反向切换 此命令将影响“x”和“y”切换的显示方式。此外，它将只有在这些开关中至少有一个是可用的。
'x'	列高亮切换 当前排序字段的高亮显示更改。您可能不需要一个持续的视觉提醒，排序字段和顶希望，您总是运行‘列高亮’关闭，因为成本在路径长度。如果忘记正在排序的字段，则此命令可用作快速的可视化提醒。
'y'	行高亮切换 为“运行”任务突出显示的更改。有关此任务状态的更多信息，请参见主题2a。字段描述，进程状态。使用这一规定为您的系统健康提供了重要的洞察。唯一的成本将是一些额外的TTY转义序列。
'z'	多色/单色切换 切换“当前”窗口之间的最后一次使用的配色方案和旧形式的黑白或白色对黑色。此命令将同时更改摘要区域和任务区域，但不影响“x”、“y”或“b”切换的状态。

II) 任务窗口内容

命令	说明
'c'	命令行/程序名称切换 无论“命令”列当前是否可见，此命令都将得到响应。稍后，如果出现该字段，则将看到您应用的更改。
'f' 'o'	字段选择或排序字段 这些键显示单独的屏幕，您可以在其中更改显示的字段及其顺序。
'H' ' '	线程切换 当此切换打开时，将显示所有单独的线程。否则，top将显示进程中所有线程的总和。
'S'	累积时间模式开关 当“累积模式”打开时，每个进程都会列出它及其死子进程使用的CPU时间。当关闭时，分得多个任务的程序看起来就不那么苛刻了。对于像‘init’或shell这样的程序来说，这是合适的，但是对于其他程序，比如编译器，也许不是。尝试使用两个任务窗口共享相同的排序字段，但使用不同的‘S’状态，并查看您喜欢哪种表示形式。发出此命令后，您将被告知此切换的新状态。如果您希望预先知道“累积模式”是否有效，只需请求帮助并查看第二行的窗口摘要即可。
'u'	只显示特定用户 将提示您输入要显示的用户的名称。此后，在该任务窗口中，只会显示匹配的用户ID，或者可能不会显示任务。稍后，如果希望再次监视所有任务，请重新发出此命令，但只需在提示符处按，而不提供名称。

III) 任务窗口大小

命令	说明
'T'	空闲进程切换 显示所有任务或仅显示活动任务。当此切换关闭时，将不会显示闲置或僵尸进程。如果在交替显示模式下将此命令应用于最后一个任务显示，那么它将不会影响窗口的大小，因为所有以前的任务显示都已经绘制过了。
'n' '#'	设置最大任务系统将提示您输入要显示的任务数。您的编号和可用屏幕行的出租人将被使用。当在交替显示模式中使用时，这是一个命令，它使您能够精确地控制每个当前可见任务显示的大小，但最后一个任务显示除外。它不会影响最后一个窗口的大小，因为以前的所有任务显示都已经绘制过了如果您希望在交替显示模式下增加最后一个可见任务显示的大小，只需缩小上面任务显示的大小。

IV) 任务窗口排序

为了兼容性，此顶支持大多数以前的顶级排序键。由于这主要是为前顶级用户提供的服务，因此这些命令不会出现在任何帮助屏幕上。

command sorted field supported

A start time (non-display) No

M %MEM Yes

N PID Yes

P %CPU Yes

T TIME+ Yes

在使用以下任何排序条款之前，top建议您使用“x”交互式命令暂时打开突出显示列。这将有助于确保实际的排序环境与您的意图相匹配。只有在当前排序字段可见时，才会执行以下交互命令。排序字段可能不可见，因为：屏幕宽度不足；“f”交互命令将其关闭。

命令	说明
'<'	左移排序字段 将排序列移到左侧，除非当前排序字段是要显示的第一个字段。
'>'	右移排序字段 将排序列移到右侧，除非当前排序字段是显示的最后一个字段。

无论当前排序字段是否可见，都将始终执行以下交互命令

命令	说明
'F'	选择排序字段 这些键显示一个单独的屏幕，您可以在其中更改使用哪个字段作为排序列。如果选择了以前未显示的字段，则将在返回到顶部显示时强制打开该字段。但是，根据屏幕宽度和字段的顺序，此排序字段可能无法显示。当在关闭列高亮显示的情况下运行top时，这个交互式命令可以方便地简单地验证当前的排序字段。
'R'	反向/正常排序字段切换 使用这个交互式命令，您可以在高到低和低到高的排序之间进行交替。

注意：字段排序使用的是内部值，而不是列显示中的值。因此，TTY和WCHAN字段将违反严格的ASCII排序序列。

颜色映射

当您发出“Z”交互命令时，将显示一个单独的屏幕。该屏幕可以用来在“当前”窗口或所有四个窗口中更改颜色，然后再返回到顶部显示。可用交互命令：

4个大写字母选择目标

8个数字选择颜色

正常切换到可用规则

'b'，运行任务“粗体”/反转

'B'，禁用/启用粗体

'z'，颜色/单色

其他可用命令

'a'/'w'，应用，然后转到下一个/优先

，应用并退出。

'q'，放弃当前的变化并退出。

如果您使用'a'或'w'循环目标窗口，您将应用离开该窗口时显示的配色方案。当然，您可以轻松地返回到任何窗口并重新应用不同的颜色，或者使用"z"按钮完全关闭颜色。颜色映射屏幕还可以用于在全屏模式或交替显示模式中更改“当前”窗口/字段组。当'q'或 被按下时，任何目标都将在返回到顶部显示时被设置为当前。

交替显示模式

Windows概述

Groups/Windows字段

在全屏模式下，只有一个窗口由整个屏幕表示。该单一窗口仍然可以更改为显示4个不同字段组中的一个(请参见‘G’交互式命令，重复如下)。四个字段组中的每一个都有一个独特的可单独配置的摘要区域和它自己的可配置任务区域。在交替显示模式下，这4个底层字段组现在可以同时显示，也可以在您的命令下单独关闭。摘要区域将始终存在，即使它只是消息行。在任何给定时间，只能显示一个摘要区域。但是，根据您的命令，屏幕上可能会显示从零到四个单独的任务显示。

当前窗口

“当前”窗口是与摘要区域相关联的窗口，也是任务相关命令始终指向的窗口。由于在交替显示模式下，您可以将任务显示关闭，因此某些命令可能被限制在“当前”窗口中。一个更复杂的情况是，当你已经切换了第一个摘要区域线。随着窗口名称(“l”切换行)的丢失，您将很难知道“当前”窗口是哪个窗口。

窗口命令

命令	说明
‘-’ 或者 ‘-’	显示或者隐藏窗口。 ***-“**键打开和关闭“当前”窗口的任务显示。打开时，该任务区域将显示使用“f”和“o”命令建立的列标题的最小值。它还将反映您应用的任何其他任务区域选项/切换，从而产生零或多个任务。 ***_“**键对所有任务显示都执行相同的操作。换句话说，它在当前可见的任务显示和您切换掉的任何任务显示之间切换。如果当前所有4个任务显示都是可见的，则此交互式命令将摘要区域保留为唯一的显示元素。
*** ‘=’** 或者 *** ‘+’**	均衡化_(再平衡)窗口***=”键强制“当前”窗口的任务显示为可见的。它还会逆转任何可能处于活动状态的‘i’(空闲任务)和‘n’(最大任务)命令。 ‘+’**键对所有窗口都是一样的。四项任务显示将重新显示，均衡平衡。它们还保留了以前应用过的任何自定义，除了‘i’(空闲任务)和‘n’(最大任务)命令。
* ‘A’	交替显示模式开关。此命令将在全屏模式和交替显示模式之间切换。第一次发出此命令时，将显示所有四个任务显示。此后，当您切换模式时，您将只看到您选择的任务显示以使其可见。
*** ‘a’** 或者 *** ‘w’**	下一个窗口向前/向后。这将改变“当前”窗口，而“当前”窗口又会更改命令指向的窗口。这些键以循环的方式工作，这样你就可以使用任意一个键达到任何想要的“当前”窗口。假设窗口名称是可见的(没有切换“l”OFF)，每当“当前”窗口名称失去其强调/颜色时，这就提醒任务显示关闭，许多命令将受到限制。
*** ‘G’**	选择另一个窗口/字段组 系统将提示您输入介于1到4之间的数字，指定应将其设置为“当前”窗口的窗口/字段组。在全屏模式下，此命令是更改“当前”窗口所必需的。在交替显示模式下，它只是“a”和“w”命令的一种不太方便的替代方式。
*** ‘g’**	更改窗口/字段组名称 系统将提示您将一个新名称应用于“当前”窗口。它不要求窗口名称是可见的(“l”切换到打开)。
*****	使用星号(*)显示的交互式命令已经超出了交替显示模式。‘=’， ‘A’， ‘G’总是可用的。‘a’， ‘w’在颜色映射时作用相同。

文件

系统配置文件

该文件的存在将影响“帮助”屏幕的哪个版本显示给普通用户。更重要的是，它将限制普通用户在运行top时

可以做的事情。他们将无法发出以下命令:

k，杀死任务

r，重新安排任务优先级。

d **, **s** **, 改变睡眠或者延迟时间。

系统配置文件不是由top创建的。相反，您可以手动创建这个文件，并将其放在/etc目录中。它的名字必须是“toprc”，不能有前导‘.’。(期间)它肯定只有两行。下面提供一个“/etc/toprc”文件的实例:

```
s      # line 1: 'secure' mode switch`^5.0    # line 2: 'delay' interval in seconds
```

个人配置文件

这个文件被写成‘\$HOME/.your-name-4-top’+‘rc’。使用‘W’交互式命令创建或更新它。如果\$HOME变量不存在，top将根据权限尝试将个人配置文件写入当前目录。

```
global  # line 1: the program name/alias notation``"    # line 2: id,altscr,irixps,delay,curwin`
```

愚蠢的诡计采样器

内核魔术

-*-，用户界面通过提示和帮助有意地暗示延迟间隔限制在十分之一秒。但是，您可以随意设置任何期望的延迟。如果您想在他的日程安排中最好地看到Linux，请尝试延迟0.09秒或更短的时间。对于这个实验，在x-windows下打开一个xTerm并使其最大化。然后执行以下操作:

通过以下方式提供调度助推和微小延迟：“**nice -n -10 top -d.09**”

保持排序列高亮显示以最小化路径长度

打开反行突出显示以强调

尝试各种排序列(time/mem运行良好)，并进行常规或反向排序，以使最活跃的进程进入视图。

您将看到一个非常繁忙的Linux在做他一直为您做的事情，但是没有可用的程序来说明这一点。

-*-，在使用“白色对黑色”颜色的xTerm下，尝试将顶部的任务颜色设置为黑色，并确保任务高亮设置为粗体，而不是相反。然后将延迟间隔设置为大约3秒，在将最活跃的进程引入视图之后，您将看到当前正在运行的任务的幽灵图像。

-*-，删除现有的rcfile，或创建一个新的符号链接。启动这个新版本，然后键入‘T’(一个秘密密钥，参见主题3c)。任务区域命令，排序)后面跟着‘W’和‘Q’。最后，用-d0(零延迟)重新启动程序。您的显示将刷新三倍于前顶部的速度，一个300%的速度优势。当顶端爬上时间阶梯的时候，你要尽可能地耐心，同时猜测顶端是否会到达顶端。

弹跳窗口

-*-，在显示3或4个任务时，选择除最后一个窗口之外的任何窗口，然后关闭空闲进程。根据应用“l”的位

置，有时有几个任务显示在弹跳，有时就像手风琴，因为top尽力分配空间。

-*-，以不同的方式设置每个窗口的摘要行：一个没有内存；另一个没有状态；可能一个没有任何信息，只有消息行。然后按住“a”或“w”，观察跳窗风的变化。

-*-，显示所有4个窗口，然后依次将空闲进程设置为OFF。你刚刚进入了“极限反弹”区域

大鸟窗

显示所有4个窗口，并确保1: def是“当前”窗口。然后，继续增加窗口大小，直到所有其他任务显示被“推出巢”为止。当它们都被移动时，在所有可见的/不可见的窗口之间切换。

举例

显示所有进程信息

```
[sogrey@bogon ~]$ top
top - 00:00:24 up 14 min,  2 users,  load average: 1.85, 1.73, 1.26
Tasks: 190 total,   1 running, 189 sleeping,   0 stopped,   0 zombie
%Cpu(s): 77.0 us, 17.0 sy,  0.0 ni,  5.7 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0 st
KiB Mem : 4380724 total, 2835072 free, 832132 used, 713520 buff/cache
KiB Swap: 4587516 total, 4587516 free,      0 used. 3307992 avail Mem

 PID USER      PR  NI      VIRT      RES      SHR S %CPU %MEM     TIME+ COMMAND
 7042 sogrey    20   0 3024960 197116  57720 S 15.6  4.5  0:13.61 gnome-shell
 5932 setroub+  20   0 372296  63124  11580 S 10.6  1.4  1:21.78 setroubles+
 5582 root     20   0 321420  37864  13600 S  4.3  0.9  0:03.71 X
13287 sogrey    20   0 868620  32968  19116 S  1.7  0.8  0:01.64 gnome-terminal
  1 root     20   0 128380   6852   4064 S  0.3  0.2  0:07.67 systemd
 1488 root     20   0  39424   8608   7984 S  0.3  0.2  0:02.46 systemd-journal
 1536 sogrey    20   0 159920   2308   1588 R  0.3  0.1  0:00.08 top
  2 root     20   0      0     0      0 S  0.0  0.0  0:00.00 kthreadd
  3 root     20   0      0     0      0 S  0.0  0.0  0:00.26 ksoftirqd/0
  5 root     20 -20      0     0      0 S  0.0  0.0  0:00.00 kworker/0:0H
  6 root     20   0      0     0      0 S  0.0  0.0  0:00.01 kworker/u
...

```

监视指定进程

```
[sogrey@bogon ~]$ top -p 7042 # 指定监视的进程id(7042)，其他进程不监视
top - 00:02:03 up 16 min,  2 users,  load average: 2.23, 1.86, 1.35
Tasks:  1 total,   0 running,   1 sleeping,   0 stopped,   0 zombie
%Cpu(s): 66.7 us, 14.5 sy,  0.0 ni, 18.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 4380724 total, 2832772 free, 832692 used, 715260 buff/cache
KiB Swap: 4587516 total, 4587516 free,      0 used. 3307360 avail Mem

 PID USER      PR  NI      VIRT      RES      SHR S %CPU %MEM     TIME+ COMMAND
 7042 sogrey    20   0 3024960 197288  57720 S 12.0  4.5  0:16.61 gnome-shell

```

touch - 创建新的空文件

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。

将文件的访问时间和修改时间修改为当前时间。如果指定的文件不存在，那么将会创造空文件，除非指定-c或-h选项。文件参数字符串'-'被专门处理，并导致touch更改与标准输出相关联的文件的时间。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
touch [选项] file
```

sh

选项

```
-a          # 只改变访问时间  
-c, --no-create      # 不创建文件  
-d, --date=time    # 设置为指定的时间，而不是当前的时间  
-f          # 忽略  
-h          # 只改变符号链接  
-m          # 只改变修改时间  
-r, --reference=file # 使用指定文件的时间  
-t          # 使用CCYYMMDDhhmmss时间  
--time=WORD       # 改指定的时间：Word为access、atime或use  
  
--help        # 显示帮助文档  
--version     # 显示命令版本信息
```

sh

--date=STRING是一种主要自由格式的人类可读的日期字符串，例如“Sun, 2月29日16: 21: 42-0800”或“2004-02-29 16: 21: 42”，甚至“下星期四”。日期字符串可能包含指示日历日期、日时间、时区、周中日、相对时间、相对日期和数字的项。空字符串表示一天的开始。

举例

```
[sogrey@bogon newDir2]$ ls
demo test test.c
[sogrey@bogon newDir2]$ touch -c 1.txt # 使用选项-c, 不创建文件
[sogrey@bogon newDir2]$ ls
demo test test.c
[sogrey@bogon newDir2]$ touch 1.txt # 不使用任何选项, 创建文件
[sogrey@bogon newDir2]$ ls
1.txt demo test test.c
[sogrey@bogon newDir2]$ ll 1.txt
-rw-----. 1 sogrey sogrey 0 1月 18 23:23 1.txt
[sogrey@bogon newDir2]$ touch -r 1.txt test.c # 将test.c的时间修改和1.txt一样
[sogrey@bogon newDir2]$ ll 1.txt test.c
-rw-----. 1 sogrey sogrey 0 1月 18 23:23 1.txt
-rw-----. 1 sogrey sogrey 88 1月 18 23:23 test.c
[sogrey@bogon newDir2]$
```

tr - 将字符进行替换压缩和删除

删除或者更改文件中的字符串，这个指令一般需要两个字符集。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
tr [选项] set1 set2
```

sh

选项

```
-c, -C, --complement      # 使用SET1的补码
-d, --delete               # 删除字符集1中指定的内容
-s, --squeeze-repeats     # 将set1中重复出现的内容，替换成单次出现的内容
-t, --truncate-set1       # 首先将SET1按照SET2的长度截断

--help                     # 显示帮助文档
--version                  # 显示命令版本信息
```

sh

```

\NNN      # 具有八进制值nnn的字符(1到3位八进制数字)
\\        # 反斜线符号
\a        # 可听BEL
\b        # (键盘的)退格键
\f        # 换页
\n        # 换行
\r        # 返回
\t        # 水平tab
\v        # 垂直tab
CHAR1-CHAR2 # 从CHAR 1到CHAR 2的所有字符按升序排列
[CHAR*]    # 拷贝set2中的字符，长度为set1的长度
[CHAR*REPEAT] # 重复拷贝
[:alnum:]   # 所有字母和数字
[:alpha:]   # 所有字母
[:blank:]   # 所有的水平空白
[:cntrl:]   # 所有的控制字符
[:digit:]   # 所有的数字
[:graph:]   # 所有的可打印的字符，不包括空格
[:lower:]   # 所有的小写字母
[:print:]   # 所有的可打印字符，包括空格
[:punct:]   # 所有标点符号
[:space:]   # 所有的水平和垂直空格
[:upper:]   # 所有的大写字母
[:xdigit:]  # 所有的十六进制数字
[=CHAR=]    # 所有等价于CHAR的字符

```

如果没有给出'-d'，同时出现SET1和SET2，则会发生翻译。'-t'只能在翻译时使用。通过在必要时重复SET1的最后一个字符，Set2被扩展到SET1的长度。Set2的多余字符将被忽略。只有[: lower:]和[: upper:]保证按升序展开；在set2翻译时使用，它们只能成对使用以指定大小写转换。'-s'在不翻译或删除时使用SET1；压缩使用SET2，并在翻译或删除后发生。

举例

```

[sogrey@bogon newDir2]$ cat test.c
You no longer send out energy of desperation or the need to be filled from the outside.
[sogrey@bogon newDir2]$ tr -s a-z A-Z < test.c # 将文件中的小写字母替换成大写
YOU NO LONGER SEND OUT ENERGY OF DESPERATION OR THE NED TO BE FILED FROM THE OUTSIDE.
[sogrey@bogon newDir2]$ tr -d send < test.c # 内容输出，然后删除出现的字符
You o logr  out rgy of pratio or th  to b fill from th outi.
[sogrey@bogon newDir2]$

```

traceroute - 显示数据包到主机间的路径

tracepath指令可以追踪数据到达目标主机的路由信息，同时还能够发现MTU值。它跟踪路径到目的地，沿着这条路径发现MTU。它使用UDP端口或一些随机端口。它类似于Traceroute，只是不需要超级用户特权，并且没有花哨的选项。tracepath 6很好地替代了tracerout 6和Linux错误队列应用程序的典型示例。tracepath的情况更糟，因为商用IP路由器在ICMP错误消息中没有返回足够的信息。很可能，当它们被更新的时候，它会改变

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
tracepath [ -n ] [ -l pktlen ] destination [ port ]
```

sh

选项

```
-n      # 不查看主机名字  
-l      # 设置初始化的数据包长度，默认65535
```

sh

举例

```
root@mops:~ # tracepath6 3ffe:2400:0:109::2  
1?: [LOCALHOST]                                pmtu 1500  
1: dust.inr.ac.ru                            0.411ms  
2: dust.inr.ac.ru      asymm 1    0.390ms pmtu 1480  
2: 3ffe:2400:0:109::2          463.514ms reached  
Resume: pmtu 1480 hops 2 back 2
```

sh

第一列显示探针的TTL，后面是冒号。通常TTL的值是从网络中得到的，但有时回复并不包含必要的信息，我们不得不猜测它。在这种情况下，数字后面跟着?。

第二列显示网络跳，对探测作出答复。如果探测未发送到网络，则为路由器地址或者[localhost]地址。

行的其余部分显示了有关到达相关工作跳的路径的各种信息。作为规则，它包含RTT的值。此外，它可以显示路径MTU，当它改变。如果路径是不对称的，或者探测在到达指定跳之前完成，则显示前向和后向跳数之间的差异。这一信息不可靠。F.E.第三行显示1的不对称性，这是因为第一次TTL为2的探针在第一跳时由于路径MTU发现而被拒绝。

最后一行总结了到达目的地的所有路径的信息，显示了检测到的路径MTU、到达目的地的跳数以及我们对从目的地到我们的跳数的猜测，这在路径不对称时可能有所不同。

追踪到www.qq.com的路由

sh

```
[root@localhost ~]$ ping -c 2 www.qq.com          #ping目标地址, 可以看到目标ip
PING www.qq.com (111.30.132.101) 56(84) bytes of data.
64 bytes from 111.30.132.101: icmp_seq=1 ttl=53 time=23.9 ms
64 bytes from 111.30.132.101: icmp_seq=2 ttl=53 time=33.0 ms

[root@localhost ~]$ tracepath www.qq.com          #追踪路由
 1: 192.168.1.9 (192.168.1.9)                  0.067ms pmtu 1500
 1: 192.168.1.1 (192.168.1.1)                  3.569ms
 1: 192.168.1.1 (192.168.1.1)                  4.055ms
 2: 192.168.1.1 (192.168.1.1)                  17.651ms pmtu 1492
 2: 10.46.80.1 (10.46.80.1)                     13.434ms
 3: 183.203.226.201 (183.203.226.201)         9.547ms
 4: 211.138.99.57 (211.138.99.57)             70.194ms asymm 5
 5: 221.183.14.5 (221.183.14.5)               17.023ms
 6: 221.176.19.237 (221.176.19.237)           206.968ms
 7: 221.183.8.149 (221.183.8.149)             29.488ms asymm 8
 8: 221.183.27.106 (221.183.27.106)           57.434ms
 9: 111.30.145.34 (111.30.145.34)             27.426ms
```

traceroute - 显示数据包到主机间的路径

traceroute命令 用于追踪数据包在网络上的传输时的全部路径，它默认发送的数据包大小是40字节。

通过traceroute我们可以知道信息从你的计算机到互联网另一端的主机是走的什么路径。当然每次数据包由某一同样的出发点（source）到达某一同样的目的地(destination)走的路径可能会不一样，但基本上来说大部分时候所走的路由是相同的。

traceroute通过发送小的数据包到目的设备直到其返回，来测量其需要多长时间。一条路径上的每个设备traceroute要测3次。输出结果中包括每次测试的时间(ms)和设备的名称（如有的话）及其ip地址。

traceroute指令输出到目标主机的路由包。Traceroute跟踪从IP网络到给定主机的路由数据包。它利用IP协议的生存时间(TTL)字段，并试图在通往主机的路径上从每个网关激发ICMP TIME_STAMPS响应。

traceroute6等价于“traceroute -6”

唯一需要的参数是目标主机的名称或IP地址。探测数据包的总大小(IPv 4默认为60字节，IPv 6为80字节)是一个可选参数。在某些情况下，可以忽略指定的大小或将其增加到最小值。

该程序试图跟踪IP数据包将遵循的路由到某些Internet主机，方法是使用一个小的ttl(生命时间)启动探测包，然后从网关侦听ICMP“时间超过”的答复。我们以1开头，然后增加1，直到我们得到一个ICMP“端口不可达”(或TCP重置)，这意味着我们到达了“主机”，或者达到了最大值(默认为30跳)。在每个ttl设置处发送三个探针(默认情况下)，并打印一行，显示每个探针的ttl、网关地址和往返时间。在请求时，可以在地址之后添加其他信息。如果探测答案来自不同的网关，则将打印每个响应系统的地址。如果在5.0秒(默认)内没有响应，则会为该探针打印一个“*”(星号)。

追踪结束后，可以打印一些附加注释：！ h、！ n或！ P(主机、网络或协议不可达)、！ s(源路由失败)、！ F(所需碎片化)、！ X(管理上禁止通信)、！ v(主机优先级冲突)、！ C(有效的优先截止)，或！ (ICMP不可达代码)。如果几乎所有的探测器都导致某种无法到达的情况，Traceroute就会放弃并退出。

我们不希望目标主机处理UDP探测包，因此目标端口被设置为一个不太可能的值(您可以使用-p标志更改它)。ICMP或TCP跟踪不存在这样的问题(对于TCP，我们使用半开放技术，这样可以防止目标主机上的应用程序看到我们的探测)。

在现代网络环境下，由于防火墙的广泛应用，传统的traceroute方法并不总是适用的。这样的防火墙过滤“不太可能”的UDP端口，甚至ICMP回音。为了解决这个问题，还实现了一些额外的跟踪方法

适用范围

RedHat openSUSE	RHEL Fedora	Ubuntu Linux Mint	CentOS Alpine Linux	Debian	Deepin Arch Linux	SUSE
--------------------	----------------	----------------------	------------------------	--------	----------------------	------

语法

```

traceroute [-46dFITUnreAV]
    [-f first_ttl]
    [-g gate,...]
    [-i device]
    [-m max_ttl]
    [-p port]
    [-s src_addr]
    [-q nqueries]
    [-N squeries]
    [-t tos]
    [-l flow_label]
    [-w waittime]
    [-z sendwait]
    [-UL]
    [-P proto]
    [--sport=port]
    [-M method]
    [-O mod_options]
    [--mtu]
    [--back]
    host
    [packet_len]

```

选项

```

-4, -6      # 显式强制IPv4或IPv6跟踪。默认情况下，程序将尝试解析给定的名称，并自动选择适当的协议。如果解析失败，程序将使用IPv4。
-I          # 使用ICMP进行路由探测
-T          # 使用TCP协议的SYN进行路由探测
-d          # 是能socket调试功能
-f first_ttl # 指定第一个数据包的TTL，默认是1
-F          # 不使用碎片
-g gateway   # 告诉Traceroute将IP源路由选项添加到传出数据包，该数据包通知网络通过指定网关路由数据包(大多数情况下)
-i interface # 指定网络接口
-m max_ttl   # 指定最大ttl，默认30
-N squeries  # 指定同时发送的探测数据包的数量。同时发送几个探针可以大大加快示踪速度。默认值为16。
-n          # 使用ip地址，不使用hostname
-p port      # 指定UDP端口
-t tos       # 对于IPv4，设置服务类型(TOS)和优先级值。有用的值是16(低延迟)和8(高吞吐量)。注意，为了使用更高的优先级，必须使用大写TOS
-w waittime  # 指定等待应答的时间，默认5s
-q nqueries  # 设置每个跳的探测数据包数。默认为3
-r          # 忽略正常的路由表
-s          # 指定发送数据包的ip地址
-z          # 探测之间的最长时间间隔(默认为0)。如果值大于10，则它指定一个以毫秒为单位的数字，否则为秒数(浮点数)
-e          # 显示ICMP扩展(Rfc 4884)。一般形式是类/类型：后面是十六进制转储。MPLS(Rfc 4950)以一种形式显示
-A          # 在路由注册表中执行路径查找，并在相应地址之后直接打印结果。

```

高级选项

```
--sport=port      # 选择要使用的源端口
-M method        # 对traceroute操作使用指定的方法。默认的传统UDP方法有名称Default, ICMP(-I)和TCP(-T)分
-O option        # 指定一些特定于方法的选项。几个选项用逗号分隔(或在cmdline上使用多个-O)。每种方法都可能有
-U               # 使用UDP对特定的目标端口进行跟踪(而不是增加每个探针的端口)。默认端口为53(DNS)
-UL              # 使用UDPLITE跟踪
-P protocol      # 使用指定协议的原始数据包进行跟踪。默认协议为253(Rfc 3692)。
--mtu            # 沿着被追踪的路径发现MTU
--back           # 打印后跳数时, 它似乎与前进方向不同。在假定远程跳发送初始ttl设置为64、128或255(这似乎是-
--help            # 显示帮助文档
-V, --version     # 显示命令版本信息
```

可用的方法method

通常, 特定的traceroute方法可能必须由-M名称来选择, 但是大多数方法都有它们简单的命令行开关(如果存在, 您可以在方法名称之后看到它们)。

```
default          # 传统的、古老的追踪方法。默认使用。
                 # 探测包是具有所谓“不可能”目标端口的UDP数据报。第一个探针的“不可能”端口是33434, 然后每个
                 # 这个方法普通用户就可以使用。
icmp             -I # 目前最常用的方法是使用ICMP回波数据包作为探针。如果您可以ping(8)目标主机, 则icmp跟踪也
tcp              -T # 众所周知的现代方法, 旨在绕过防火墙。使用常量目标端口(默认为80, http)。
                 # 这种方法使用了众所周知的“半开放技术”, 它可以防止目标主机上的应用程序看到我们的探测。通常
                 # 这个方法有以下的一些选项, 默认的是syn、sysctl
                 # syn,ack,fin,rst,psh,urg,ece,cwr, 在任意组合中为探测包设置指定的tcp标志。
                 # flags=num, 将TCP标头中的标志字段设置为num。
                 # ecn, 发送带有TCP标志ECA和CWR的syn数据包(用于显式拥塞通知, rfc 3168)
                 # sack,timestamps,window_scaling, 在传出探测包中使用相应的tcp标头选项。
                 # sysctl, 对上面的TCP头选项和ecn使用当前sysctl("/proc/sys/net/*")设置。默认情况下始终
                 # mss=num, 对maxseg tcp报头选项使用num值(当syn)
tcpconn          # TCP方法的初始实现, 简单使用CONNECT(2)调用, 完成TCP会话的完全打开
udp               -U # 使用带有常量目标端口的UDP数据报(默认为53, DNS)。也打算绕过防火墙。
                 # 注意, 与TCP方法不同的是, 目标主机上的相应应用程序总是接收我们的探测(带有随机数据)
udplite          -UL # 对探针使用udplite数据报(具有固定的目标端口, 默认为53), 此方法不需要特权。选项:
coverage=num, 设置udplite范围num。
raw              -P proto # 发送协议原始数据包。选项: protocol=proto, 使用IP协议Proto(默认253)
```

说明

为了加速工作, 通常同时发送几个探测器。另一方面, 它制造了一个“包裹风暴”, 特别是在回复方向。路由器可以节流ICMP响应的速率, 有些应答可能会丢失。为了避免这种情况, 减少同步探测的数量, 甚至将其设置为1(类似于最初的traceroute实现), 即-N1。

最终(目标)主机可以丢弃一些同时进行的探测, 甚至可能只回答最新的探测。它可以导致额外的“看上去像过期”啤酒花接近最后一跳。我们使用智能算法来自动检测这种情况, 但如果在您的情况下它无法帮助, 只需使用-N1。

为了获得更好的稳定性, 您可以通过-z选项来减缓程序的工作速度, 例如, 在探测之间使用“-z 0.5”进行半

秒暂停。

如果有些跳对每种方法都没有任何报告，那么获得某些信息的最后机会是使用“ping -R”命令(ipv4，并且仅对最近的8跳)。

举例

追踪到baidu的路由信息

```
[root@localhost ~]$ ping www.baidu.com -c 1          #ping目标, 得到ip地址
PING www.a.shifen.com (111.13.100.91) 56(84) bytes of data.

[root@localhost ~]$ traceroute -n www.baidu.com      #追踪路由
traceroute to www.baidu.com (111.13.100.92), 30 hops max, 60 byte packets
 1  192.168.1.1  4.124 ms  3.936 ms  3.882 ms
 2  10.46.80.1  8.917 ms  9.238 ms  9.233 ms
 3  183.203.226.201  12.855 ms  12.788 ms  12.802 ms
 4  221.180.30.197  12.792 ms  221.180.30.45  12.776 ms  12.762 ms
 5  221.183.47.225  13.526 ms  13.363 ms  13.259 ms
 6  221.183.37.249  26.798 ms  23.556 ms  26.832 ms
 7  * * *
 8  111.13.98.101  20.569 ms  20.460 ms  111.13.98.93  24.463 ms
 9  111.13.98.93  27.215 ms  111.13.98.101  20.895 ms  111.13.112.53  26.946 ms
10  111.13.108.5  24.136 ms  111.13.112.57  23.754 ms  111.13.112.61  23.712 ms
```

tune2fs - tune2fs允许系统管理员在Linux ext2、ext3或ext4文件系统上调整各种可调的文件系统参数

tune2fs允许系统管理员在Linux ext2、ext3或ext4文件系统上调整各种可调的文件系统参数。这些选项的当前值可以使用-l选项显示，也可以通过使用dump2fs (8)程序显示。

pwd - 显示当前工作目录

查看用户当前的工作目录，输出完整路径。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
tune2fs [选项] device
tune2fs [ -l ] [ -c max-mount-counts ] [ -e errors-behavior ] [ -f ] [ -i interval-between-
[ -J journal-options ] [ -m reserved-blocks-percentage ] [ -o [^]mount-options[,...]
[ -s sparse-super-flag ] [ -u user ] [ -g group ] [ -C mount-count ] [ -E extended-
[ -M last-mounted-directory ] [ -O [^]feature[,...]] [ -T time-last-checked ] [
```

选项

```
-l          # 显示设备的详细信息
-c max-mount-counts # 检查文件系统之前，调整最大挂载次数。如果最大挂载计数为0或-1，e2fsck(8)和内核将忽略此参数。
-C mount-count # 设置文件系统的挂载次数，如果设置为比-c选项设置的max-mount-counts参数更大的值，e2fsck(8)将忽略此参数。
-e error-behavior # 当检测到错误时，更改内核代码的行为。在任何情况下，文件系统错误都会导致e2fsck(8)停止运行。
# - continue，继续执行正常执行。
# - remount-ro，重新以只读的方式挂载。
# - panic，产生kernel panic。
-E extended-options # 为文件系统设置扩展选项。扩展选项是逗号分隔的，可以使用相等号('=')符号进行参数化。
# - stride=stride-size，为RAID数组配置具有步长文件系统块的文件系统。这是在移动到RAID时必需的。
# - stripe_width=stripe-width，为RAID数组配置文件系统，每条带宽的文件系统块。这在RAID上是必需的。
# - hash_alg=hash-alg，设置用于具有散列b树目录的文件系统的默认哈希算法。接受的有'md5'、'sha1'、'sha256'、'sha512'。
# - mount_opts=mount_option_string，设置一组默认的挂载选项，这些选项将在挂载文件系统时应用。
# - test_fs，在文件系统超级块中设置一个标志，指示可以使用试验性内核代码（如ext4dev工具）来挂载文件系统。
# - ^test_fs，清除test_fs标志，指示仅使用production-level文件系统代码挂载文件系统。
-f          # 强制执行。当从具有外部日志的文件系统（或损坏到似乎有外部日志）而外部日志不可用时，e2fsck将忽略此参数。
-g group    # 设置可以使用保留文件系统块的组。group参数可以是数值gid或组名。如果给定组名，则在该组的所有块上启用保留。
-i interval-between-checks[d|m|w] # 调整两个文件系统检查之间的最大时间。没有后缀或d将数字间隔解释为分钟。
-j          # 向文件系统添加ext3日志，如果没有指定-J选项，则将使用默认日志参数来创建存储在文件系统中的日志。
-J journal-options # 如果此选项用于在已安装的文件系统上创建日志，则将在文件系统的顶层目录中创建一个名为journal的子目录。
# 重写默认ext 3日志参数。日记选项是逗号分隔的，可以使用相等号('=')符号进行参数化。
# - size=journal-size，创建一个存储在文件系统中的日志，大小journal-size。日志的大小由size参数指定。
# - device=external-journal，将文件系统附加到位于外部日志上的日志块设备上。外部日志块设备必须是块设备，且必须与文件系统的块设备具有相同的大小。
# - size和device只能有一个。
-L volume-label # 设置文件系统卷标。ext2文件系统标签最多可以长达16个字符；如果卷标签超过16个字符，将截断并忽略多余的字符。
-m reserved-blocks-percentage # 设置只能由特权进程分配的文件系统百分比。保留一些文件系统块以供特权进程使用。
-M last-mounted-directory # 设置最后的挂载目录。
-o [^]mount-option[,,...] # 在文件系统中设置或清除指定的默认挂载选项。默认的挂载选项可以被“/etc/fstab”文件覆盖。
-O [^]feature[,,...]      # 设置或清除文件系统中指定的文件系统特性（选项）。多个文件系统特性可以通过用逗号分隔它们来指定。
-r reserved-blocks-count # 设置保留文件系统块的数目。
-T time-last-checked      # 使用e2fsck设置上次检查文件系统的时间。时间使用当前（本地）时区进行解释。这在使用UTC时区时特别有用。
-u user                  # 设置可以使用保留文件系统块的用户。用户可以是uid或用户名。如果给定用户名，则将其转换为uid。
-U UUID                  # 将文件系统的通用唯一标识符（UUID）设置UUID。UUID的格式是由连字符分隔的一系列十六进制数字。
# - clear，清除UUID。
# - random，产生一个随机的UUID。
# - time，产生一个基于时间的UUID。
```

举例

查看sdb4的详细信息

```
[root@localhost ~]$ tune2fs -l /dev/sdb4
tune2fs 1.41.12 (17-May-2010)
Filesystem volume name:   hello
Last mounted on:          <not available>
Filesystem UUID:          e2a0cb30-f3ca-47de-92b8-780296960d93
...
First inode:               11
Inode size:                128
Default directory hash:    half_md4
Directory Hash Seed:       4930bf0f-771e-4940-9255-bee40d138079
```

设置最大挂载次数

sh

```
[root@localhost ~]$ tune2fs -c 30 /dev/sdb4    #设置最大挂载次数
tune2fs 1.41.12 (17-May-2010)
Setting maximal mount count to 30
You have new mail in /var/spool/mail/root
[root@localhost ~]$ tune2fs -l /dev/sdb4      #查看详细信息
tune2fs 1.41.12 (17-May-2010)
Filesystem volume name:   hello
Maximum mount count:     30                  #最大挂载次数已经修改为30
Directory Hash Seed:     4930bf0f-771e-4940-9255-bee40d138079
[root@localhost ~]$
```

ulimit - 控制shell程序的资源

ulimit命令 用来限制系统用户对shell资源的访问。如果不懂什么意思，下面一段内容可以帮助你理解：

假设有这样一种情况，当一台 Linux 主机上同时登陆了 10 个人，在系统资源无限制的情况下，这 10 个用户同时打开了 500 个文档，而假设每个文档的大小有 10M，这时系统的内存资源就会受到巨大的挑战。

而实际应用的环境要比这种假设复杂的多，例如在一个嵌入式开发环境中，各方面的资源都是非常紧缺的，对于开启文件描述符的数量，分配堆栈的大小，CPU 时间，虚拟内存大小，等等，都有非常严格的要求。资源的合理限制和分配，不仅仅是保证系统可用性的必要条件，也与系统上软件运行的性能有着密不可分的联系。这时，ulimit 可以起到很大的作用，它是一种简单并且有效的实现资源限制的方式。

ulimit 用于限制 shell 启动进程所占用的资源，支持以下各种类型的限制：所创建的内核文件的大小、进程数据块的大小、Shell 进程创建文件的大小、内存锁住的大小、常驻内存集的大小、打开文件描述符的数量、分配堆栈的最大大小、CPU 时间、单个用户的最大线程数、Shell 进程所能使用的最大虚拟内存。同时，它支持硬资源和软资源的限制。

作为临时限制，ulimit 可以作用于通过使用其命令登录的 shell 会话，在会话终止时便结束限制，并不影响其他 shell 会话。而对于长期的固定限制，ulimit 命令语句又可以被添加到由登录 shell 读取的文件中，作用于特定的 shell 用户。

ulimit指令用来设置shell或者shell启动的进程可用资源的限制，当选项后面没有参数的时候，显示当前选项的限制。如果给定限制，则为指定资源的新值(-a选项仅显示)。如果没有提供任何选项，则假定为"-f"。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux	Arch Linux		

语法

```
ulimit [-HSTabcdefilmnpqrstuvx [limit]]
```

sh

选项

sh

```

-a          # 显示当前资源显示情况
-b          # 最大socket缓冲区
-c          # core文件最大值, 单位为区块
-d          # 程序数据段最大值, 单位kb
-e          # 最大调度级别
-f          # shell创建文件最大值, 单位区块
-i          # 悬挂信号的最大值
-l          # 内存锁定的最大值
-m          # 可使用内存最大值, 单位kb
-n          # 打开文件最大数目
-p          # 管道缓冲区大小, 单位512B
-q          # posix队列最大值
-s          # 栈的最大值
-r          # 实际调度时间
-R          # 使用CPU上限, 单位s
-u          # 一个用户最大运行程序数目
-v          # 虚拟内存上限, 单位kb
-x          # 文件锁最大值
-T          # 最大线程数

```

举例

显示当前所有的限制

sh

```

[sogrey@bogon ~]$ ulimit -a
core file size      (blocks, -c) 0
data seg size        (kbytes, -d) unlimited
scheduling priority (-e) 0
file size            (blocks, -f) unlimited
pending signals      (-i) 16969
max locked memory   (kbytes, -l) 64
max memory size     (kbytes, -m) unlimited
open files           (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues (bytes, -q) 819200
real-time priority   (-r) 0
stack size            (kbytes, -s) 8192
cpu time              (seconds, -t) unlimited
max user processes    (-u) 4096
virtual memory        (kbytes, -v) unlimited
file locks            (-x) unlimited
[sogrey@bogon ~]$

```

修改设置

sh

```

[root@localhost ~]$ ulimit -s      #查看栈的限制
10240
[root@localhost ~]$ ulimit -s 10241  #设置栈的限制
[root@localhost ~]$ ulimit -s      #查看栈的限制, 已经修改
10241

```

umask - 显示或设置创建文件的权限掩码

指定创建文件时所需要的权限掩码，掩码的执行权限对于文件没有效果。如果模式以数字开头，则解释为八进制数字；否则解释为符号模式掩码，类似于chmod(1)所接受的模式掩码。如果省略模式，则打印掩码的当前值。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
umask [选项] mask
```

sh

选项

-S # 以字符方式表示权限掩码
-P # 显示当前权限掩码

sh

举例

```
[sogrey@bogon newDir2]$ umask #不用任何参数，可以查看掩码  
0077  
[sogrey@bogon newDir2]$ mkdir demo # 创建目录  
[sogrey@bogon newDir2]$ ll -d demo  
drwx----- 2 sogrey sogrey 4096 1月 18 23:06 demo # 权限700  
[sogrey@bogon newDir2]$ touch test.c  
[sogrey@bogon newDir2]$ ll test.c  
-rw----- 1 sogrey sogrey 0 1月 18 23:06 test.c # 权限600  
[sogrey@bogon newDir2]$ umask 0055 # 修改掩码  
[sogrey@bogon newDir2]$ mkdir test  
[sogrey@bogon newDir2]$ ll -d test  
drwx-w--w- 2 sogrey sogrey 4096 1月 18 23:07 test # 权限744  
[sogrey@bogon newDir2]$
```

sh

umount - 用于卸载已经加载的文件系统

umount命令 用于卸载已经加载的文件系统。利用设备名或挂载点都能umount文件系统，不过最好还是通过挂载点卸载，以免使用绑定挂载（一个设备，多个挂载点）时产生混乱。

移除已经挂载到系统中的文件系统，可以是挂载点，也可以使挂载设备。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
umount [选项] src|dst  
umount [-hV]  
umount -a [-dflnrv] [-t vfstype] [-O options]  
umount [-dflnrv] {dir|device}...
```

sh

选项

```
-a          # 卸载/etc/fstab中的所有文件系统。使用umountVersion2.7及更高版本，未卸载proc文件系统  
-f          # 强制卸载  
--fake      # 模拟卸载过程。它可用于从/etc/mtab中删除前面使用-n选项卸载的条目。  
-n          # 不在/etc/mtab中记录卸载信息，在/etc只读的系统中，这个选项很重要  
-d          # 如果是个回环设备，同时释放掉这回环设备  
-r          # 如果卸载失败，尝试以读写的方式挂载  
-l          # 懒散的umount。现在将文件系统从文件系统层次结构中分离出来，并在文件系统不再繁忙时立即清除对该文  
-O option   # 指示这些操作只应在/etc/fstab中指定选项的文件系统上执行。可以在逗号分隔列表中指定多个选项类型。  
-t type     # 指定挂载的文件系统类型。目前支持的系统有：adfs, affs, autofs, cifs, coda, coherent, cram  
-v          # 显示详细执行过程  
--no-canonicalize # 不要把路径规范化  
  
-h          # 显示帮助文档  
-V          # 显示命令版本信息
```

sh

说明

umount命令将释放与挂载相关联的循环设备(如果有的话)，在/etc/mtab中，找到“loop=”或在给出-d选项时。任何挂起的循环设备都可以使用“losetup -d”释放。

外部umount帮助程序的语法是：“/sbin/umount. {dir|device} [-nlfv] [-t type.subtype]”， 是文件系统类型或来自“uheler=”mtab选项的值。“-t”选项用于支持子类型的文件系统(例如/sbin/mount.fut-tfuse.sshfs)。

“/etc/mtab”显示已经挂载文件系统表。

举例

卸载文件

```
[root@localhost ~]$ mount          # 查看已经挂载的设备  
/dev/mapper/VolGroup-lv_root on / type ext4 (rw)  
proc on /proc type proc (rw)  
/dev/sr0 on /media/VBox_GAs_5.2.18 type iso9660 (ro,nosuid,nodev,uhelper=udisks,uid=0,gid=0,ioch  
/weijie/my.iso on /media/sf_data type iso9660 (rw,loop=/dev/loop0)  
/dev/sdb4 on /media/test type ext2 (rw)  
  
[root@localhost ~]$ umount /dev/sdb4      # 卸载, 参数是设备  
You have new mail in /var/spool/mail/root  
[root@localhost ~]$ umount /media/sf_data/  # 卸载, 参数是挂载点  
  
[root@localhost ~]$ mount          # 再次查看, 是否卸载成功  
/dev/mapper/VolGroup-lv_root on / type ext4 (rw)  
proc on /proc type proc (rw)  
gvfs-fuse-daemon on /root/.gvfs type fuse.gvfs-fuse-daemon (rw,nosuid,nodev)  
/dev/sr0 on /media/VBox_GAs_5.2.18 type iso9660 (ro,nosuid,nodev,uhelper=udisks,uid=0,gid=0,ioch
```

unalias - 删 除由alias设置的别名

unalias指令用来取消已经定义的别名。

主要用途

- 删除一个或多个别名。
- 删除全部已定义的别名。

适 用 范 围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语 法

```
unalias [-a] [name ...]
```

sh

选 项

```
-a # 取消所有别名
```

sh

举 例

删除已经定义的别名

```
[root@localhost ~]$ unalias mytail      # 删除别名mytail
You have new mail in /var/spool/mail/root
[root@localhost ~]$ alias                  # 查看已经定义别名, mytail已经不存在
alias cp='cp -i'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mv='mv -i'
alias rm='rm -i'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

sh

注 意

1. 执行脚本时请注意：

使用source命令执行的bash脚本如果执行了alias或unalias命令，那么有可能会对终端环境的别名设置产生影响；终端环境的别名设置也可能改变运行结果；
通过sh方式调用的bash脚本或直接运行当前用户有执行权限的脚本不受终端环境的别名影响。

2. 查看及设置别名，请查看alias命令。
3. 该命令是bash内建命令，相关的帮助信息请查看help命令。

uname - 打印系统信息

主要用途

- 打印机器和操作系统的各种信息。
- 当没有选项时， 默认启用 -s 选项。
- 如果给出多个选项或 -a 选项时， 输出信息按以下字段排序：内核名称 主机名称 内核release 内核版本 机器名称 处理器 硬件平台 操作系统。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
uname [OPTION]
```

sh

选项

-a, --all	# 显示所有的信息
-s, --kernel-name	# 显示内核名字
-n, --nodename	# 显示主机名
-r, --kernel-release	# 显示内核发型版本号
-v, --kernel-version	# 显示内核版本
-m, --machine	# 显示计算机硬件架构名字
-p, --processor	# 显示cpu类型
-i, --hardware-platform	# 显示硬件平台
-o, --operating-system	# 显示操作系统
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

显示系统所有信息

```
[sogrey@bogon ~]$ uname --all
Linux bogon 3.10.0-862.14.1.0.h209.eulerosv2r7.x86_64 #1 SMP Tue Feb 12 00:00:00 UTC 2019 x86_64
[sogrey@bogon ~]$ uname -s
Linux
[sogrey@bogon ~]$ uname -n
bogon
[sogrey@bogon ~]$
```

sh

unarj - 解压缩由arj命令创建的压缩包

unarj命令 用来解压缩由arj命令创建的压缩包。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
unarj [OPTION] [参数]
```

sh

选项

```
e                # 解压缩.arj文件;  
l                # 显示压缩文件内所包含的文件;  
t                # 检查压缩文件是否正确;  
x                # 解压缩时保留原有的路径。
```

sh

uncompress - 用来解压.Z文件

uncompress命令 用来解压缩由compress命令压缩后产生的".Z"压缩包。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
uncompress [选项] [参数]
```

sh

选项

```
-f            # 不提示用户，强制覆盖掉目标文件；  
-c            # 将结果送到标准输出，无文件被改变；  
-r            # 递归的操作方式。
```

sh

举例

先创建一个.Z压缩文件

```
compress FileName
```

sh

解压：

```
uncompress FileName.Z
```

sh

unexpand - 将文件的空白字符转换为制表符

将文件中的空白字符转换为控制字符tab，将结果送到标准输出。空格数大于8才能被替换。如果没有指定文件，那么从标准输入读取。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
unexpand [选项] file
```

sh

选项

```
-a                            # 转换所有的空格  
--first-only                # 仅转换开头的空格  
-t, --tabs                 # 指定tab代表的字符数，默认是8  
  
--help                      # 显示帮助文档  
--version                   # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon newDir3]$ ll  
总用量 16  
-rw----- 1 sogrey sogrey 27 1月 27 23:37 1.txt  
-rw----- 1 sogrey sogrey 62 1月 24 01:03 students.txt  
-rw----- 1 sogrey sogrey 11 1月 24 01:08 test2.txt  
-rw----- 1 sogrey sogrey 135 1月 24 01:05 test.txt  
[sogrey@bogon newDir3]$ cat 1.txt # 查看内容，有16个空格  
nihao                        linux  
[sogrey@bogon newDir3]$ unexpand -a -t 16 1.txt # 替换16个空格，变为tab  
nihao                        linux  
[sogrey@bogon newDir3]$
```

sh

uniq - 显示或忽略重复的行

将文件中重复出现的行删除，结果送到标准输出或者指定文件。在使用uniq指令之前，必须使用sort对内容进行排序，否则没有效果。如果没有选项，则将匹配的行合并到第一个匹配项。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
uniq [选项] [input] [output]
```

sh

将输入文件（或标准输入）中邻近的重复行写入到输出文件（或标准输出）中。

当没有选项时，邻近的重复行将合并为一个。

- INPUT（可选）： 输入文件，不提供时为标准输入。
- OUTPUT（可选）： 输出文件，不提供时为标准输出。

选项

```
-c,--count                                                 # 显示行重复出现的次数
-d, --repeated                                          # 仅显示重复出现的行
-D, --all-repeated[=delimit-method]              # 打印所有重复行
-f, --skip-fields=N                                  # 忽略前n个字段
-i, --ignore-case                                      # 比较时忽略大小写
-s, --skip-chars=N                                   # 忽略前n个字符
-u, --unique                                           # 只显示不重复的行
-z, --zero-terminated                                  # 以0字节为结束符，而不是换行
-w, --check-chars=N                                  # 比较不超过指定次数

--help                                                  # 显示帮助文档
--version                                              # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon demos]$ cat test5.txt
石家庄今日新增16例确诊病例
北京一确诊者隐瞒行程不配合流调
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
特朗普夫人发文谴责国会暴乱
山西晋中新增2例无症状感染者
特朗普夫人发文谴责国会暴乱
[sogrey@bogon demos]$ sort test5.txt | uniq -c # 先排序, 然后再删除重复行, 显示重复行出现的次数
 2 北京一确诊者隐瞒行程不配合流调
 1 理塘文旅公司回应丁真抽烟
 1 山西晋中新增2例无症状感染者
 1 石家庄今日新增16例确诊病例
 3 特朗普夫人发文谴责国会暴乱
 1 中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon demos]$ sort test5.txt | uniq -c -u # 只显示不重复的行
 1 理塘文旅公司回应丁真抽烟
 1 山西晋中新增2例无症状感染者
 1 石家庄今日新增16例确诊病例
 1 中国留美博士遇害 美驻华使馆慰问
[sogrey@bogon demos]$
```

unset - 删 除 指 定 的 shell 变 量 或 函 数

unset是和set相反的一个指令，用来删除shell变量或者函数。

主 要 用 途

- 删 除 一 到 多 个 shell 变 量 (不 包 括 只 读 变 量) 。
- 删 除 一 到 多 个 shell 函 数。
- 删 除 一 到 多 个 具 有 引 用 属性 的 变 量 (如 果 -n 选 项 存 在) 。

适 用 范 围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语 法

```
unset [-fv] [name ...]
```

sh

对于每个名称，删除相应的变量或函数。如果没有提供任何选项，或者给出了"-v"选项，那么每个名称都引用一个shell变量。只读变量可能不会被取消设置。如果指定了"-f"，则每个名称都引用一个shell函数，并删除函数定义。从传递给后续命令的环境中删除每个未设置变量或函数。如果任何COMP_WORDBREAKS, RANDOM, SECONDS, LINENO, HISTCMD, FUNCNAME, GROUPS, DIRSTACK都未设置，则它们将失去它们的特殊属性，即使它们随后被重置。除非名称是只读的，否则退出状态为真。

选 项

```
-f      # 删 除 函 数  
-v      # 删 除 变 量
```

sh

举 例

删 除 变 量

```
[root@localhost ~]$ set | grep USERS      #查 看 变 量  
_=USERS=weijie  
[root@localhost ~]$ unset -v USERS          #删 除 变 量  
[root@localhost ~]$ set | grep USERS          #查 看 变 量， 已 经 删 除  
_=
```

sh

unzip - 用于解压缩由zip命令压缩的压缩包

unzip命令 用于解压缩由zip命令压缩的".zip"压缩包。

解压zip指令压缩过的文件。unzip将列出、测试或从ZIP存档中提取文件，这些文件通常在MS-DOS系统中找到。默认行为(没有选项)是将指定ZIP存档中的所有文件提取到当前目录(及其下面的子目录)中。一个配套程序zip(1L)创建ZIP档案；这两个程序都与PKWare的PKZIP和PKUNZIP为MS-DOS创建的档案兼容，但在许多情况下，程序选项或默认行为有所不同。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
unzip [-Z] [-cflptTuvz[abjnoqsCDKLMUVWX$/:^]] file[.zip] [file(s) ...] [-x xfile(s) ...] [sh]
```

选项

```
file[.zip]      # ZIP存档的路径。如果文件规范是通配符，则按操作系统(或文件系统)确定的顺序处理每个匹配的文件。  
# *，匹配0或多个字符的序列。  
# ?，匹配一个字符。  
# [...]，匹配括号内的任何单个字符；范围由开始字符、连字符和结束字符指定。如果是感叹号或插入符号！  
# (确保引用任何可能被操作系统解释或修改的字符，特别是在Unix和VMS下。)如果没有找到匹配项，将忽略该参数。  
[file(s)]       # 要处理的归档成员的可选列表，用空格分隔。(用VMSCLI定义的VMS版本必须用逗号分隔文件。见“-c”选项)  
[-x xfile(s)]   # 要排除在处理之外的归档成员的可选列表。由于通配符通常匹配(‘/’ )目录分隔符(异常请参见选项“-Z”)  
[-d exdir]      # 提取文件的可选目录。默认情况下，所有文件和子目录都在当前目录中重新创建；-d选项允许在任
```

请注意，为了支持过时的硬件，Unzip的使用屏幕仅限于22或23行，因此只应被视为基本解压缩语法的提醒，而不是所有可能的标志的详尽列表。详尽的清单如下：

```

-c          # 将解压结果送到标准输出
-f          # 解压时更新现有的文件，即仅解压磁盘上已经存在且比磁盘副本更新的文件。默认情况下，在覆盖之前询问。
-l          # 显示压缩包内文件的详细信息。将打印指定文件的名称、未压缩文件大小、修改日期和时间，以及文件的MD5校验和。
-p          # 将解压结果送到标准输出，不对字符转换。只有文件数据被发送到stdout，文件总是以二进制格式输出。
-t          # 检查压缩文件的正确性。该选项提取内存中的每个指定文件，并将扩展文件的CRC(循环冗余校验)与文件头中的值进行比较。
-T          # 将存档上的时间戳设置为每个归档中最新文件的时间戳。这对应于zip的--set-date选项，但它可以用于通过zip命令创建的存档。
-u          # 更新现有文件，并在需要时创建新文件。该选项执行与-f选项相同的功能。
-v          # 列出存档文件(详细格式)或显示诊断版本信息。
-z          # 仅显示压缩文件的备注信息

-a          # 解压时，对文本文件做字符转换
-b          # 不对文本文件字符转换，把所有文件当做二进制文件。
-B          # 保存每个覆盖文件的备份副本，‘foo’的旧拷贝改名为‘foo~’。
-C          # 解压时，设置文件名大小写敏感
-D          # 跳过提取项的时间戳恢复。
-E          # [仅限MacOS]在恢复操作期间显示MacOS额外字段的内容。
-F          # [仅限Acorn]禁止从存储的文件名中删除NFS文件类型扩展。
-i          # [仅限MacOS]忽略存储在MacOS额外字段中的文件名。相反，使用存储在条目标题的泛型部分中的文件名。
-j          # 垃圾路。存档的目录结构不被重新创建；所有文件都存放在提取目录中(默认情况下是当前的)。
-J          # [仅限BeOS]垃圾文件属性。文件的BeOS文件属性没有恢复，只是文件的数据。
-K          # 保留SUID/SGID/duy文件属性。如果没有此标志，则出于安全原因，将清除这些属性位。
-L          # 解压时，将文件名改为小写字母
-M          # 把解压结果送给more分屏显示
-n          # 解压时，不覆盖原有的文件
-N          # 将文件注释解压缩为Amiga文件。
-o          # 不提示用户，覆盖原有文件
-P          # 解压时，输入密码
-q          # 静默模式
-s          # [OS/2, NT, MS-DOS]将文件名中的空格转换为下划线
-S          # [VMS]将文本文件(-a, -aa)转换为Stream_LF记录格式，而不是文本文件默认的可变长度记录格式。
-U          # [UNICODE_SUPPORT only] 修改或禁用UTF-8处理。当Unicode_Support可用时，选项-U强制启用。
-V          # 保留(VMS)文件版本号。VMS文件可以用版本号存储，格式为file.ext; #。默认情况下，‘; #’版本号表示不存在。
-W          # [仅当WILD_STOP_AT_DIRR编译时选项启用] 修改模式匹配例程，以便“?”和“*”与目录分隔符‘/’匹配。
-X          # [VMS, Unix, OS/2, NT, Tandem] 在Unix下恢复用户和组信息(UID/GID)
-Y          # [VMS] 将存档的文件名结尾“.nn”(其中‘nnn’是一个十进制数)视为VMS版本号(‘; nnn’)。(默认情况下，‘; 0’表示不存在)
-$          # [MS-DOS, OS/2, NT]如果提取介质是可移动的(例如磁盘)，则恢复卷标签。加倍的选项($-$)允许多个卷标签。
-extensions # [Acorn] 重写Unzip$Ext环境变量提供的扩展列表。在提取过程中，与此扩展列表中的项匹配的文件将被解压缩。
-:          # 允许将存档成员解压缩到当前“提取根文件夹”之外的位置。
-^          # [Unix only] 允许提取的ZIP存档条目的名称中包含控制字符。
-2          # [vms] 强制无条件地将文件名转换为ODS 2-兼容名称

--help       # 显示帮助文档
--version    # 显示命令版本信息

```

环境选项

unzip的默认行为可以通过放置在环境变量中的选项来修改。这可以用任何选项来完成，但是它可能对-a, -L, -C, -q, -o或-n修饰符最有用：默认情况下使unzip自动转换文本文件，使文件名从大写系统转换为小写，使其不敏感地匹配名称，使其更安静，或者让它在解压文件时始终覆盖或不覆盖它们。例如，要使解压缩尽可能安静，只报告错误，可以使用以下命令之一：

```
sh
Unix Bourne shell:      UNZIP=-qq; export UNZIP
Unix C shell:setenv      UNZIP -qq
OS/2 or MS-DOS:set      UNZIP=-qq
VMS (quotes for lowercase):    define UNZIP_OPTS "-qq"
```

实际上，环境选项被认为与任何其他命令行选项一样，只是它们实际上是命令行中的第一个选项。要覆盖环境选项，可以使用“减号运算符”来删除它。例如，若要覆盖上面示例中的一个静音标志，请使用以下命令

```
sh
unzip --q[other options] zipfile
```

第一个连字符是正常的开关字符，第二个是负号，作用于Q选项。因此，这里的效果是取消一个安静的量子。若要取消这两个安静的标志，可以使用两个(或更多)最小值：

```
sh
unzip -t--q zipfile
unzip ---qt zipfile      # 这两个是等价的
```

正如上面的例子所建议的，默认变量名是UNZIP_OPTS，用于VMS，以及所有其他操作系统的解压缩。为了与zip(1L)兼容，UNZIP_OPTS也被接受(不要问)。但是，如果同时定义了UNZIP和UNZIP_OPTS，那么UNZIP优先。Unzip的诊断选项(没有zip文件名的-v)可以用于检查所有四个可能的解压缩和zipinfo环境变量的值

说明

unzip的某些编译版本可能不支持解密。若要检查加密支持版本，可以尝试测试或提取加密存档，或者检查unzip的诊断屏幕(请参阅上面的-v选项)以“[解密]”作为特殊的编译选项之一。

如上所述，可以使用-P选项在命令行上提供密码，但代价是安全性。首选的解密方法是正常提取；如果zip文件成员被加密，解压缩将提示输入密码，而不回显所键入的内容。解压缩继续使用相同的密码，只要它看起来是有效的，通过在每个文件上测试一个12字节的头。正确的密码将始终签出与标题，但有1/256的机会，一个不正确的密码也会。(这是PKWare zipfile格式的一个安全特性；它有助于防止暴力攻击，否则只通过测试报头就可以获得很大的速度优势。)如果给出了错误的密码，但它还是通过了头测试，则将为提取的数据生成不正确的CRC，或者在提取过程中解压缩失败，因为‘解密’字节并不构成有效的压缩数据流。

如果第一个密码没有通过头检查某个文件，解压缩将提示输入另一个密码，以此类推，直到提取所有文件。如果不知道密码，则输入空密码(即只返回一个回车或‘Enter’)作为跳过所有进一步提示的信号。随后将只提取存档中未加密的文件。(事实上，这并不完全正确；早期版本的zip(1L)和zipcloak(1L)允许空密码，因此解压缩检查每个加密文件以查看空密码是否有效。这可能会导致“假阳性”和提取错误，如上所述。)

用8位密码加密的档案(例如，带有重音的欧洲字符的密码)可能无法跨系统和/或其他存档器移植。这个问题源于对这些字符使用多种编码方法，包括拉丁文1(ISO 8859-1)和OEM代码页850。DoS PKZIP 2.04g使用OEM代码页；Windows PKZIP 2.50使用拉丁文-1(因此与DOS PKZIP不兼容)；Info-ZIP使用DOS、OS/2和Win3.x端口上的OEM代码页，但使用ISO编码(拉丁文-1等)。在其他地方，NicoMak的WinZip6.x根本不允许8位密码。解压缩5.3(或更新)尝试首先使用默认字符集(例如拉丁语-1)，然后使用备用字符集(例如OEM代码页)测试密码。

举例

```
sh
unzip letters
# 将letters.zip解压到当前的目录以及其子目录下

unzip -j letters
# 将letters.zip解压到当前目录下

unzip -tq letters
# 测试letters.zip, 打印出测试信息, 确定压缩包是否完整

unzip -tq *.zip
# 测试当前目录下的所有zip文件, 打印总和信息

unzip -ca letters *.tex | more
# 要将名称以.tex结尾的letters.zip的所有成员提取为标准输出, 自动转换为本地行尾约定, 并将输出传递给more程序

unzip -p articles paper1.dvi | dvips
# 将二进制文件解压到标准输出, 并且通过管道送到另一个打印程序

unzip source.zip "*.[fch]" Makefile -d /tmp
# 将所有的c语言和FORTRAN语言的源文件, 以及Makefile解压到/tmp目录

unzip -C source.zip "*.[fch]" makefile -d /tmp
# 将所有的c语言和FORTRAN语言的源文件, 以及Makefile解压到/tmp目录(忽略所有的大小写)

unzip -aaCL source.zip "*.[fch]" makefile -d /tmp
# 提取任何此类文件, 但将ms-dos或vms中任何大写名称转换为小写, 并将所有文件的行尾转换为本地标准。

unzip -fo sources
# 仅提取当前目录中已经存在的较新版本的文件, 而不进行查询

unzip -uo sources
# 提取当前目录中已存在的较新版本的文件, 并创建尚未存在的任何文件。

unzip -v
# 要显示诊断屏幕, 显示哪些解压缩和zipinfo选项存储在环境变量中, 是否在环境变量中编译解密支持, 以及编译解压缩的

unzip -l file.zip
# 只是列出内容

unzip -ql file.zip
# 双倍安静的列表

unzip --ql file.zip
unzip -l-q file.zip
unzip -l--q file.zip
# 标准的列表
```

显示备注信息

```
[sogrey@bogon 文档]$ zip 1.zip 1.c          # 压缩  
adding: 1.c (deflated 9%)  
[sogrey@bogon 文档]$ unzip -z 1.zip        # 显示备注  
Archive: 1.zip
```

sh

显示压缩包内的文件信息

```
[sogrey@bogon 文档]$ unzip -l 1.zip  
Archive: 1.zip  
Length      Date    Time     Name  
-----      -----  
      53  09-19-2018 16:08  1.c  
-----  
      53                      1 file
```

sh

解压

```
[sogrey@bogon 文档]$ unzip -n -v 1.zip      # 显示解压过程  
Archive: 1.zip  
Length  Method   Size  Cmpr    Date    Time    CRC-32   Name  
-----  -----  
      53  Defl:N    48    9%  09-19-2018 16:08  1c3e46be  1.c  
-----  
      53                  48    9%  
                                1 file
```

sh

诊断信息

退出码说明

- 0 正常，没有错误信息
- 1 遇到一个或多个警告错误，但仍然成功地完成了处理。这包括由于不支持的压缩方法或密码未知的加密而跳过一个或多个文件的zip文件。
- 2 检测到zip文件格式中的通用错误。无论如何，处理可能已经成功完成；一些由其他归档程序创建的损坏的zip文件具有简单的工作环境。
- 3 检测到zip文件格式出现严重错误。处理可能立即失败
- 4 在程序初始化期间，unzip无法为一个或多个缓冲区分配内存。
- 5 unzip无法分配内存，也无法获取TTY来读取解密密码。
- 6 unzip过程中无法将内存分配给磁盘。
- 7 unzip无法在内存解压缩期间分配内存。
- 8 当前还没有用到这个退出码
- 9 指定的压缩文件没有找到
- 10 非法的选项
- 11 找不到匹配的文件
- 50 磁盘已经满了
- 51 ZIP存档的结尾是过早地遇到的。
- 80 用户使用ctrl+c终止了解压过程
- 81 由于不支持的压缩方法或不支持的解密，一个或多个文件的测试或提取失败
- 82 由于解密密码错误，没有找到任何文件。(但是，即使成功处理了一个文件，退出状态也是1。)

uptime - 查看Linux系统负载信息

uptime命令能够打印系统总共运行了多长时间和系统的平均负载。uptime命令可以显示的信息显示依次为：现在时间、系统已经运行了多长时间、目前有多少登陆用户、系统在过去的1分钟、5分钟和15分钟内的平均负载。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
uptime
```

sh

选项

```
-V # 显示版本信息并且退出
```

sh

举例

显示系统运行时间

```
[sogrey@bogon ~]$ uptime  
23:54:13 up 8 min, 2 users, load average: 1.83, 1.73, 1.00  
[sogrey@bogon ~]$
```

sh

useradd - 创建的新的系统用户

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。

创建新的系统用户，useradd指令只能以管理员的身份运行，创建的用户都在"/etc/passwd"文件中。当不加-D参数，useradd指令使用命令列来指定新帐号的设定值and使用系统上的预设值.新使用者帐号将产生一些系统档案，使用者目录建立，拷备起始档案等，这些均可以利用命令列选项指定。此版本为RedHat Linux提供，可帮每个新加入的使用者建立个别的group,毋须添加-n选项。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
useradd [-D] [选项] [参数]

useradd [-c comment] [-d home_dir] [-e expire_date] [-f inactive_time] [-g initial_group]

useradd -D [-g default_group] [-b default_home] [-f default_inactive] [-e default_expire_da
```

选项

```
--help # 显示帮助文档
--version # 显示命令版本信息

# 使用-D选项

-b dir
指定用户的基目录，默认是home

-e date
使用者账号有效日期

-f days
指定密码过期后多少天关闭账号

-g group
```

指定账户的组，这个组必须存在

-s shell

指定账户使用的shell

不使用 -D选项

-c comment

备注信息

-d home

账户每次登陆使用的目录

-e date

账号终止日期，MM/DD/YY

-f days

账号过期后多久停用

-g group

指定账户的组，这个组必须存在

-G group

指定用户的附加组

-m

使用者目录如果不存在，自动创建

-n

取消自动创建于账号同名的组

-p password

设置账户的密码

-r

创建系统账号

-s

指定账户登录后使用的shell

-u uid

指定用户ID

相关文件

- `/etc/passwd` 使用者帐号资讯。
- `/etc/shadow` 使用者帐号资讯加密。
- `/etc/group` 群组资讯。
- `/etc/default/useradd` 定义资讯。
- `/etc/login.defs` 系统广义设定。
- `/etc/skel` 内含定义档的目录。

举例

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 caojh -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。

创建用户

```
[sogrey@bogon ~]$ sudo useradd -p 123456 userTmp # 创建用户userTmp, 密码123456
[sudo] sogrey 的密码:
[sogrey@bogon ~]$ tail -n 2 /etc/passwd # 查看是否创建成功
vboxadd:x:989:1::/var/run/vboxadd:/bin/false
userTmp:x:1001:1001::/home/userTmp:/bin/bash
[sogrey@bogon ~]$ ls /home/ # 查看在home下创建家目录
lost+found sogrey userTmp
[sogrey@bogon ~]$
```

设定用户的家目录、uid、备注

```
[sogrey@bogon ~]$ sudo useradd -d /home/other -u 600 -c "test user" user02 # 创建用户
[sogrey@bogon ~]$ tail -n 2 /etc/passwd # 查看用户信息
userTmp:x:1001:1001::/home/userTmp:/bin/bash
user02:x:600:1002:test user:/home/other:/bin/bash
[sogrey@bogon ~]$ ls /home/ # 家目录other
lost+found other sogrey userTmp
[sogrey@bogon ~]$
```

userdel - 用于删除给定的用户以及与用户相关的文件

userdel命令 用于删除给定的用户，以及与用户相关的文件。若不加选项，则仅删除用户帐号，而不删除相关文件。

删除用户，如果没有附加选项，仅删除用户，不删除相关文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
userdel [-r] user
```

sh

选项

-r	# 删除用户的同时，删除其相关文件
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

userdel命令很简单，比如我们现在有个用户linuxde，其家目录位于/var目录中，现在我们来删除这个用户：

```
userdel linuxde      # 删除用户linuxde，但不删除其家目录及文件;  
userdel -r linuxde  # 删除用户linuxde，其家目录及文件一并删除;
```

sh

请不要轻易用-r选项；他会删除用户的同时删除用户所有的文件和目录，切记如果用户目录下有重要的文件，在删除前请备份。

其实也有最简单的办法，但这种办法有点不安全，也就是直接在/etc/passwd中删除您想要删除用户的记录；但最好不要这样做，/etc/passwd是极为重要的文件，可能您一不小心会操作失误。

不使用选项，删除用户

sh

```
[sogrey@bogon ~]$ tail -n 2 /etc/passwd
userTmp:x:1001:1001::/home/userTmp:/bin/bash
user02:x:600:1002:test user:/home/other:/bin/bash
[sogrey@bogon ~]$ sudo userdel userTmp #删除用户
[sogrey@bogon ~]$ ls /home/ #相关文件还存在
lost+found  other  sogrey  userTmp
[sogrey@bogon ~]$
```

删除用户所有信息

sh

```
[sogrey@bogon ~]$ sudo userdel -r userTmp #删除用户，使用-r
[sogrey@bogon ~]$ ls /home/ #相关文件也删除
lost+found  other  sogrey
```

usermod - 用于修改用户的基本信息

usermod命令 用于修改用户的基本信息。usermod 命令不允许你改变正在线上的使用者帐号名称。当 usermod 命令用来改变user id，必须确认这名user没在电脑上执行任何程序。你需手动更改使用者的 crontab 档。也需手动更改使用者的 at 工作档。采用 NIS server 须在server上更动相关的NIS设定。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
pwd [OPTION]
```

sh

选项

```
-c comment          # 修改备注信息  
-d home_dir        # 修改账户每次登陆使用的目录  
-e expire_date    # 修改账号终止日期, MM/DD/YY  
-f inactive_days  # 账号过期后多久停用  
-g initial_group   # 修改账户的组, 这个组必须存在  
-G group,[...]     # 修改用户的附加组  
-l login_name      # 变更使用者login时的名称为login_name, 其余不变。特别是, 使用者目录  
-s shell           # 修改账户登录后使用的shell  
-u uid             # 修改用户ID  
  
--help              # 显示帮助文档  
--version          # 显示命令版本信息
```

相关文件

- [/etc/passwd](#) 使用者帐号资讯。
- [/etc/shadow](#) 使用者帐号资讯加密。
- [/etc/group](#) 群组资讯。

举例

将 newuser2 添加到组 staff 中：

```
usermod -G staff newuser2
```

sh

修改newuser的用户名为newuser1：

```
usermod -l newuser1 newuser
```

sh

锁定账号newuser1:

```
usermod -L newuser1
```

sh

解除对newuser1的锁定:

```
usermod -U newuser1
```

sh

增加用户到用户组中:

```
apk add shadow # 安装 shadow 包, usermod 命令包含在 usermod 中  
usermod -aG group user # 添加用户到用户组中
```

sh

-a 参数表示附加, 只和 -G 参数一同使用, 表示将用户增加到组中。

修改用户uid:

```
[sogrey@bogon ~]$ tail -n 2 /etc/passwd      #查看用户信息  
webalizer:x:67:67:Webalizer:/var/www/usage:/sbin/nologin  
user01:x:502:502::/home/user01:/bin/bash  
[sogrey@bogon ~]$ usermod -u 503 user01      #修改uid为503  
[sogrey@bogon ~]$ tail -n 2 /etc/passwd      #查看用户信息, uid已经改变  
webalizer:x:67:67:Webalizer:/var/www/usage:/sbin/nologin  
user01:x:503:502::/home/user01:/bin/bash
```

sh

修改备注信息

```
[sogrey@bogon ~]$ usermod -c "user01 test" user01    #修改备注信息为“user01 test”  
[sogrey@bogon ~]$ tail -n 2 /etc/passwd    #查看用户信息  
webalizer:x:67:67:Webalizer:/var/www/usage:/sbin/nologin  
user01:x:503:502:user01 test:/home/user01:/bin/bash
```

sh

修改用户家目录

```
[sogrey@bogon ~]$ useradd sogrey  
[sogrey@bogon ~]$ ls /home  
sogrey  
[sogrey@bogon ~]$ usermod -md /data/new_home sogrey  
[sogrey@bogon ~]$ ls /home/  
[sogrey@bogon ~]$ ls /data/  
new_home
```

sh

usernetctl - 被允许时操作指定的网络接口

usernetctl命令 在用于被允许时操作指定的网络接口。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
usernetctl [param]
```

sh

选项

```
网络接口      # 被操纵的网络接口;  
up           # 激活网络接口;  
down         # 禁用网络接口;  
report       # 报告网络接口状态。
```

sh

users - 打印当前主机所有登陆用户的名称

每个显示的用户名对应一个登录会话；如果一个用户有不止一个登录会话，那他的用户名将显示相同的次数。

该命令是GNU coreutils包中的命令，相关的帮助信息请查看 `man -s 1 users` , `info coreutils 'users invocation'`

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
users [OPTION]... [FILE]
```

sh

FILE（可选）：记录用户当前登录情况的文件；默认使用 `/var/run/utmp` 、 `/var/log/wtmp` 。

选项

```
--help                          # 显示帮助文档  
--version                        # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon ~]$ users  
sogrey sogrey  
[sogrey@bogon ~]$
```

sh

uupick - 命令处理传送进来的文件

uupick命令 处理传送进来的文件。 当其他主机通过UUCP将文件传送进来时，可利用uupick指令取出这些文件。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
uupick [-v][-I<配置文件>][-s<主机>][-x<层级>][--help]
```

sh

选项

```
-I<配置文件>, --config<配置文件> # 指定配置文件。  
-s<主机>, --system<主机> # 处理由指定主机传送过来的文件。  
-v, --version # 显示版本信息。  
--help # 显示帮助。
```

sh

举例

处理由主机localhost传送过来的文件。在命令行直接输入如下命令：

```
uupick-s localhost
```

sh

该命令通常没有输出。

vi - 功能强大的纯文本编辑器

vi命令是UNIX操作系统和类UNIX操作系统中最通用的全屏幕纯文本编辑器。Linux中的vi编辑器叫vim，它是vi的增强版（vi Improved），与vi编辑器完全兼容，而且实现了很多增强功能。

vi编辑器支持编辑模式和命令模式，编辑模式下可以完成文本的编辑功能，命令模式下可以完成对文件的操作命令，要正确使用vi编辑器就必须熟练掌握着两种模式的切换。默认情况下，打开vi编辑器后自动进入命令模式。从编辑模式切换到命令模式使用“esc”键，从命令模式切换到编辑模式使用“A”、“a”、“O”、“o”、“I”、“i”键。

vi编辑器提供了丰富的内置命令，有些内置命令使用键盘组合键即可完成，有些内置命令则需要以冒号：“开头输入。常用内置命令如下：

```
Ctrl+u: 向文件首翻半屏;
Ctrl+d: 向文件尾翻半屏;
Ctrl+f: 向文件尾翻一屏;
Ctrl+b: 向文件首翻一屏;
Esc: 从编辑模式切换到命令模式;
ZZ: 命令模式下保存当前文件所做的修改后退出vi;
:行号: 光标跳转到指定行的行首;
:$: 光标跳转到最后一行的行首;
x或X: 删除一个字符, x删除光标后的, 而X删除光标前的;
D: 删除从当前光标到光标所在行尾的全部字符;
dd: 删除光标行正行内容;
n dd: 删除当前行及其后n-1行;
nyy: 将当前行及其下n行的内容保存到寄存器?中, 其中?为一个字母, n为一个数字;
p: 粘贴文本操作, 用于将缓存区的内容粘贴到当前光标所在位置的下方;
P: 粘贴文本操作, 用于将缓存区的内容粘贴到当前光标所在位置的上方;
/字符串: 文本查找操作, 用于从当前光标所在位置开始向文件尾部查找指定字符串的内容, 查找的字符串会被加亮显示;
?字符串: 文本查找操作, 用于从当前光标所在位置开始向文件头部查找指定字符串的内容, 查找的字符串会被加亮显示;
a, bs/F/T: 替换文本操作, 用于在第a行到第b行之间, 将F字符串换成T字符串。其中, “s/”表示进行替换操作;
a: 在当前字符后添加文本;
A: 在行末添加文本;
i: 在当前字符前插入文本;
I: 在行首插入文本;
o: 在当前行后面插入一空行;
O: 在当前行前面插入一空行;
:wq: 在命令模式下, 执行存盘退出操作;
:w: 在命令模式下, 执行存盘操作;
:w!: 在命令模式下, 执行强制存盘操作;
:q: 在命令模式下, 执行退出vi操作;
:q!: 在命令模式下, 执行强制退出vi操作;
:e文件名: 在命令模式下, 打开并编辑指定名称的文件;
:n: 在命令模式下, 如果同时打开多个文件, 则继续编辑下一个文件;
:f: 在命令模式下, 用于显示当前的文件名、光标所在行的行号以及显示比例;
:set number: 在命令模式下, 用于在最左端显示行号;
:set nonumber: 在命令模式下, 用于在最左端不显示行号;
```

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
pwd [OPTION] [参数]
```

sh

选项

+<行号>：从指定行号的行开始显示文本内容；
-b：以二进制模式打开文件，用于编辑二进制文件和可执行文件；
-c<指令>：在完成对第一个文件编辑任务后，执行给出的指令；
-d：以diff模式打开文件，当多个文件编辑时，显示文件差异部分；
-l：使用lisp模式，打开“lisp”和“showmatch”；
-m：取消写文件功能，重设“write”选项；
-M：关闭修改功能；
-n：不实用缓存功能；
-o<文件数目>：指定同时打开指定数目的文件；
-R：以只读方式打开文件；
-s：安静模式，不现实指令的任何错误信息。

sh

扩展

vi编辑器有三种工作方式：命令方式、输入方式和ex转义方式。通过相应的命令或操作，在这三种工作方式之间可以进行转换。

命令方式

在Shell提示符后输入命令vi，进入vi编辑器，并处于vi的命令方式。此时，从键盘上输入的任何字符都被作为编辑命令来解释，例如，a(append) 表示附加命令，i(insert) 表示插入命令，x表示删除字符命令等。如果输入的字符不是vi的合法命令，则机器发出“报警声”，光标不移动。另外，在命令方式下输入的字符（即vi命令）并不在屏幕上显示出来，例如，输入i，屏幕上并无变化，但通过执行i命令，编辑器的工作方式却发生变化：由命令方式变为输入方式。

输入方式

通过输入vi的插入命令 (i)、附加命令 (a)、打开命令 (o)、替换命令 (s)、修改命令(c) 或取代命令 (r) 可以从命令方式进入输入方式。在输入方式下，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，被当做该文件的正文。进入输入方式后，输入的可见字符都在屏幕上显示出来，而编辑命令不再起作用，仅作为普通字母出现。例如，在命令方式下输入字母i，进到输入方式，然后再输入i，就在屏幕上相应光标处添加一个字母i。

由输入方式回到命令办法是按下Esc键。如果已在命令方式下，那么按下Esc键就会发出“嘟嘟”声。为了确保用户想执行的vi命令是在命令方式下输入的，不妨多按几下Esc键，听到嘟声后再输入命令。

ex转义方式

vi和ex编辑器的功能是相同的，二者的主要区别是用户界面。在vi中，命令通常是单个字母，如a,x,r等。而在ex中，命令是以Enter键结束的命令行。vi有一个专门的“转义”命令，可访问很多面向行的ex命令。为使用ex转义方式，可输入一个冒号（:）。作为ex命令提示符，冒号出现在状态行（通常在屏幕最下行）。按下中断键（通常是Del键），可终止正在执行的命令。多数文件管理命令都是在ex转义方式下执行的（例如，读取文件，把编辑缓冲区的内容写到文件中等）。转义命令执行后，自动回到命令方式。例如：

```
:1,$s/I/i/g 按Enter键
```

则从文件第一行至文件末尾 (\$) 将大写I全部替换成小写i。vi编辑器的三种工作方式之间的转换如图所示。

```
!vi
```

vmstat - 显示虚拟内存状态

vmstat命令 的含义为显示虚拟内存状态（“Viryual Memor Statics”），但是它可以报告关于进程、内存、I/O等系统整体运行状态。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
vmstat [-a] [-n] [-t] [-S unit] [delay [ count]]
vmstat [-s] [-n] [-S unit]
vmstat [-m] [-n] [delay [ count]]
vmstat [-d] [-n] [delay [ count]]
vmstat [-p disk partition] [-n] [delay [ count]]
vmstat [-f]
vmstat [-V]
```

选项

```
sh
-a      # 显示使用和非使用的虚拟内存状态
-f      # 显示开机之后fork的使用情况
-t      # 增加时间戳到输出
-m      # 显示slab信息
-n      # 显示一次头信息，而不是间隔性的显示
-d      # 显示磁盘使用情况
-w      # -w增大了大内存的字段宽度。
-p      # -p后面跟着一些分区名称以进行详细统计(所需的2.5.70或更高内核)
-S      # 设置容量单位。可是k、K、m、M，分别代表1000, 1024, 1000000, 1048576字节
-V      # 显示版本信息
```

字段说明

Procs (进程)

- r: 运行队列中进程数量，这个值也可以判断是否需要增加CPU。（长期大于1）
- b: 等待IO的进程数量。

Memory (内存)

- swpd: 使用虚拟内存大小，如果swpd的值不为0，但是SI, SO的值长期为0，这种情况不会影响系统性能。
- free: 空闲物理内存大小。
- buff: 用作缓冲的内存大小。

- cache: 用作缓存的内存大小，如果cache的值大的时候，说明cache处的文件数多，如果频繁访问到的文件都能被cache处，那么磁盘的读IO bi会非常小。

Swap

- si: 每秒从交换区写到内存的大小，由磁盘调入内存。
- so: 每秒写入交换区的内存大小，由内存调入磁盘。

注意：内存够用的时候，这2个值都是0，如果这2个值长期大于0时，系统性能会受到影响，磁盘IO和CPU资源都会被消耗。有些朋友看到空闲内存（free）很少的或接近于0时，就认为内存不够用了，不能光看这一点，还要结合si和so，如果free很少，但是si和so也很少（大多时候是0），那么不用担心，系统性能这时不会受到影响的。

IO (现在的Linux版本块的大小为1kb)

- bi: 每秒读取的块数
- bo: 每秒写入的块数

注意：随机磁盘读写的时候，这2个值越大（如超出1024k），能看到CPU在IO等待的值也会越大。

system (系统)

- in: 每秒中断数，包括时钟中断。
- cs: 每秒上下文切换数。

注意：上面2个值越大，会看到由内核消耗的CPU时间会越大。

CPU (以百分比表示)

- us: 用户进程执行时间百分比(user time)

us的值比较高时，说明用户进程消耗的CPU时间多，但是如果长期超50%的使用，那么我们就该考虑优化程序算法或者进行加速。

- sy: 内核系统进程执行时间百分比(system time)

sy的值高时，说明系统内核消耗的CPU资源多，这并不是良性表现，我们应该检查原因。

- wa: IO等待时间百分比

wa的值高时，说明IO等待比较严重，这可能由于磁盘大量作随机访问造成，也有可能磁盘出现瓶颈（块操作）。

- id: 空闲时间百分比

说明

vmstat不需要特殊权限。这些报告旨在帮助识别系统瓶颈。Linux vmstat并不将自己视为一个正在运行的进程。所有Linux块当前为1024字节。旧内核可以将块报告为512字节、2048字节或4096字节。因为prps 3.1.9，vmstat允许您在默认模式下选择单位(k, K, m, M)默认为K(1024字节)。vmstat使用slabinfo 1.1修
补程序

举例

显示虚拟内存使用情况

```
[sogrey@bogon ~]$ vmstat -a
procs -----memory----- swap-- io--- system-- cpu-----
 r b swpd free inact active si so bi bo in cs us sy id wa st
 4 0     0 2840808 514380 818456   0   0 1246 179 976 1025 56 16 28 0 0
[sogrey@bogon ~]$
```

显示开机后fork使用状况

```
[sogrey@bogon ~]$ vmstat -f
24517 forks
[sogrey@bogon ~]$
```

显示磁盘信息

```
[sogrey@bogon ~]$ vmstat -d
disk- -----reads----- writes----- IO -----
      total merged sectors      ms  total merged sectors      ms  cur  sec
sda    16869    5501 1319134  38545  3545 11458 192064 33931    0   33
sr0      0      0      0      0      0      0      0      0      0      0      0
dm-0   20997      0 1227338  27533 13413      0 180064 435921    0   15
dm-1      94      0   4456      27      0      0      0      0      0      0      0
dm-2     884      0 27970  24174 1424      0 11856   3590    0   23
[sogrey@bogon ~]$
```

volname - 显示指定的ISO-9660格式的设备的卷名称

volname命令 用于显示指定的“ISO-9660”格式的设备的卷名称，通常这种格式的设备为光驱。

显示iso9660格式设备的卷标，一般情况下是CD-ROM。它还适用于包含iso-9660文件系统的普通文件。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
volname [device] sh
```

选项

设备名	# 要显示卷标的设备，如果不指定，默认显示/dev/cdrom的卷标q	sh
-----	-------------------------------------	----

举例

不指定设备，显示默认设备的卷标

```
[root@localhost ~]$ volname  
VBOXADDITIONS_5.2.18_124319 sh
```

显示指定设备的卷标

```
[root@localhost ~]$ volname /dev/sdb4 # 这个设备不是iso9660，因此没有结果  
[root@localhost ~]$ volname /weijie/my.iso # 这个设备是iso9660格式  
CDROM sh
```

w - 显示目前登入系统的用户信息

w命令 用于显示已经登陆系统的用户列表，并显示用户正在执行的指令。执行这个命令可得知目前登入系统的用户有那些人，以及他们正在执行的程序。单独执行w命令会显示所有的用户，您也可指定用户名称，仅显示某位用户的相关信息。

显示哪些用户登录，并且显示用户在干什么。报头按此顺序显示当前时间、系统运行时间、当前登录用户数以及过去1、5和15分钟的系统平均负载。接着为每个用户显示以下条目：登录名、TTY名称、远程主机、登录时间、空闲时间、JCPU、PCPU和当前进程的命令行。JCPU时间是附加到TTY的所有进程使用的时间。它不包括过去的后台作业，但也包括当前正在运行的后台作业。PCPU时间是当前进程使用的时间，在“what”字段中命名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
w [OPTION] [user]
```

sh

选项

```
-h, --no-header      # 不显示标题栏
-u, --no-current    # 忽略执行程序的名称和cpu时间
-s, --short          # 使用短格式，不显示登录时间、JCPU 和PCPU time
-f, --from            # 显示用户从哪里登录
-o, --old-style       # 老式输出
-i, --ip-addr         # 显示IP地址而不是主机名（如果可能）

--help                # 显示此帮助并退出
-V, --version          # 显示版本信息。
```

sh

相关文件

- /var/run/utmp, 正在登陆的用户信息。
- /proc process information, 进程信息。

举例

显示登录的用户信息

sh

```
[sogrey@bogon ~]$ w
23:19:05 up 2 min, 2 users, load average: 2.21, 1.16, 0.46
USER      TTY      LOGIN@    IDLE   JCPU   PCPU WHAT
sogrey    :0      23:17 ?xdm?  30.90s 0.22s /usr/libexec/gnome-session-bina
sogrey    pts/0    23:19    1.00s  0.06s  0.02s w
[sogrey@bogon ~]$
```

watch - 可以将命令的输出结果输出到标准输出设备，多用于周期性执行命令/定时执行命令

watch命令 以周期性的方式执行给定的指令，指令输出以全屏方式显示。watch是一个非常实用的命令，基本所有的Linux发行版都带有这个小工具，如同名字一样，watch可以帮你监测一个命令的运行结果，省得你一遍遍的手动运行。

watch指令可以间歇性的执行程序，将输出结果以全屏的方式显示，默认是2s执行一次。watch将一直运行，直到被中断。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
watch [-dhvt] [-n <seconds>] [--differences[=cumulative]]  
[--help] [--interval=<seconds>] [--no-title]  
[--version] <command>
```

sh

选项

```
-d, --differences          # 高亮显示差异部分  
--cumulative               # 高亮显示“sticky”  
-n                         # 指定时间间隔  
-t, --no-title              # 不显示日期时间以及间隔秒数  
  
--help                      # 显示帮助文档  
--version                   # 显示命令版本信息
```

sh

举例

```
watch -n 1 -d netstat -ant      # 命令：每隔一秒高亮显示网络链接数的变化情况
watch -n 1 -d 'pstree|grep http' # 每隔一秒高亮显示http链接数的变化情况。后面接的命令若带有管道符，需要
watch 'netstat -an | grep:21 | \ grep<模拟攻击客户机的IP>| wc -l' # 实时查看模拟攻击客户机建立起来的连
watch -d 'ls -l|grep scf'        # 监测当前目录中 'scf' 的文件的变化
watch -n 10 'cat /proc/loadavg' # 10秒一次输出系统的平均负载
watch uptime
watch -t uptime
watch -d -n 1 netstat -ntlp
watch -d 'ls -l | fgrep goface'    # 监测goface的文件
watch -t -differences=cumulative uptime
watch -n 60 from                 # 监控mail
watch -n 1 "df -i;df"            # 监测磁盘inode和block数目变化情况

#查看邮件
watch -n 60 from

#查看目录内容的变化
watch -d ls -l

#如果您只对用户joe拥有的文件感兴趣，可以使用
watch -d 'ls -l | fgrep joe'

#要想看到引号的效果，请试一试
watch echo $$ 
watch echo '$$'
watch echo ",$$$,"

#您可以监视管理员安装最新的内核。
watch uname -r
```

每3s执行一次ls指令

```
[root@localhost ~]$ watch -n 3 ls #每3s执行一次ls
Every 3.0s: ls                                         Sun Sep 23 09:04:40 2018

1
1.c~
anaconda-ks.cfg
icmp_echo_ignore_all~
icmp_echo_ignore_alv~
icmp_echo_ignore_alw~
icmp_echo_ignore_alex~
icmp_echo_ignore_aly~
icmp_echo_ignore_alz~
install.log
install.log.syslog
mail
nohup.out
公共的
模板
视频
图片
文档
下载
音乐
桌面
```

wc - 统计文件的字节数、字数、行数

wc命令 统计指定文件中的字节数、字数、行数，并将统计结果显示输出。利用wc指令我们可以计算文件的Byte数、字数或是列数，若不指定文件名称，或是所给予的文件名为“-”，则wc指令会从标准输入设备读取数据。wc同时也给出所指定文件的总统计数。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
sh
wc [选项] file
```

选项

-c, --bytes # 仅显示字节数
-m, --chars # 仅显示字符数
-l, --lines # 仅显示行数
-L, --max-line-length # 显示文件中最长行的字符数
-w, words # 显示单词数

--help # 显示帮助文档
--version # 显示命令版本信息

举例

```
sh
[sogrey@bogon demos]$ ls
diff.txt test2.txt test3.txt test4.txt test.c test.txt
[sogrey@bogon demos]$ cat test.txt
石家庄今日新增16例确诊病例
中国留美博士遇害 美驻华使馆慰问
特朗普夫人发文谴责国会暴乱
理塘文旅公司回应丁真抽烟
北京一确诊者隐瞒行程不配合流调
山西晋中新增2例无症状感染者
[sogrey@bogon demos]$ wc -c test.txt # 显示字节数
250 test.txt
[sogrey@bogon demos]$ wc -l test.txt # 显示行数
6 test.txt
```

wget - Linux系统下载文件工具

wget是一个免费的文件下载工具，可以从指定的URL下载文件到本地主机。它支持HTTP和FTP协议，经常用来抓取大量的网页文件。

wget命令 用来从指定的URL下载文件。wget非常稳定，它在带宽很窄的情况下和不稳定网络中有很强的适应性，如果是由于网络的原因下载失败，wget会不断的尝试，直到整个文件下载完毕。如果是服务器打断下载过程，它会再次联到服务器上从停止的地方继续下载。这对从那些限定了链接时间的服务器上下载大文件非常有用。

wget支持HTTP，HTTPS和FTP协议，可以使用HTTP代理。所谓的自动下载是指，wget可以在用户退出系统的之后在后台执行。这意味这你可以登录系统，启动一个wget下载任务，然后退出系统，wget将在后台执行直到任务完成，相对于其它大部分浏览器在下载大量数据时需要用户一直的参与，这省去了极大的麻烦。

用于从网络上下载资源，没有指定目录，下载资源回默认为当前目录。wget虽然功能强大，但是使用起来还是比较简单：

1. 支持断点下传功能 这一点，也是网络蚂蚁和FlashGet当年最大的卖点，现在，Wget也可以使用此功能，那些网络不是太好的用户可以放心了；
2. 同时支持FTP和HTTP下载方式 尽管现在大部分软件可以使用HTTP方式下载，但是，有些时候，仍然需要使用FTP方式下载软件；
3. 支持代理服务器 对安全强度很高的系统而言，一般不会将自己的系统直接暴露在互联网上，所以，支持代理是下载软件必须有的功能；
4. 设置方便简单 可能，习惯图形界面的用户已经不太习惯命令行了，但是，命令行在设置上其实有更多的优点，最少，鼠标可以少点很多次，也不要担心是否错点鼠标；
5. 程序小，完全免费 程序小可以考虑不计，因为现在的硬盘实在太大了；完全免费就不得不考虑了，即使网络上有很多所谓的免费软件，但是，这些软件的广告却不是我们喜欢的。

适用范围

RedHat RHEL Ubuntu CentOS Fedora

语法

```
 wget [options] [path or URL]
```

sh

wget有价格返回值：0，正常；1，通用错误；2，参数错误；3，IO错误；4，网络错误；5，SSL错误；6，用户名密码错误；7，协议错误；8，服务器错误。

选项

```

-a          # 将指令运行过程记录到指定文件
-A          # 设置要下载的文件的扩展名，多个扩展名使用，分割
-b          # 将下载任务放到后台运行
-B          # 设置基本参考的链接地址
-c          # 从上次中断的地方继续运行
-C          # 打开或者关闭服务器的数据快取功能，默认on
-d          # 调试模式
-D          # 设置接受的域名，多个域名使用，分开
-e          # 接货wget后就执行的指令
-F          # 将输入的文件作为HTML格式
-h          # 显示帮助信息
-i          # 从指定文件获取URL
-l          # 设置接受的目录
-L          # 下载有关联的连接
-P          # 指定文件存放目录
-r          # 递归下载指定目录下的所有文件
-R          # 设置排除下载的文件类型
-nc         # 文件存在时，不覆盖
-nd         # 所有文件都下载到当前目录
-nv         # 下载时，只显示更新和出错信息
-q          # 静默模式
-nh         # 不查询主机名称
-v          # 显示详细执行过程
-V          # 显示版本信息

```

举例

下载

```

[sogrey@localhost ~]$ wget -v www.baidu.com
--2021-07-15 00:07:11-- http://www.baidu.com/
正在解析主机 www.baidu.com (www.baidu.com)... 14.215.177.38, 14.215.177.39
正在连接 www.baidu.com (www.baidu.com)|14.215.177.38|:80... 已连接。
已发出 HTTP 请求，正在等待回应... 200 OK
长度: 2381 (2.3K) [text/html]
正在保存至: “index.html”

100%[=====] 2,381      --.-K/s 用时 0s

2021-07-15 00:07:11 (397 MB/s) - 已保存 “index.html” [2381/2381])

[sogrey@localhost ~]$

```

下载，指定存放目录

```
[sogrey@localhost 文档]$ sudo -i
[sudo] sogrey 的密码:
[root@localhost ~]# cd /home/sogrey/文档/
[root@localhost 文档]# wget -P /baidu/ -r -l 2 www.baidu.com
--2021-07-15 00:13:53--  http://www.baidu.com/
正在解析主机 www.baidu.com (www.baidu.com)... 14.215.177.38, 14.215.177.39
正在连接 www.baidu.com (www.baidu.com)|14.215.177.38|:80... 已连接。
已发出 HTTP 请求, 正在等待回应... 200 OK
长度: 2381 (2.3K) [text/html]
正在保存至: “/baidu/www.baidu.com/index.html”

100%[=====] 2,381          --.-K/s 用时 0s

2021-07-15 00:13:53 (586 MB/s) - 已保存 “/baidu/www.baidu.com/index.html” [2381/2381])

正在载入 robots.txt; 请忽略错误消息。
--2021-07-15 00:13:53--  http://www.baidu.com/robots.txt
再次使用存在的到 www.baidu.com:80 的连接。
已发出 HTTP 请求, 正在等待回应... 200 OK
长度: 2814 (2.7K) [text/plain]
正在保存至: “/baidu/www.baidu.com/robots.txt”

100%[=====] 2,814          --.-K/s 用时 0s

2021-07-15 00:13:54 (461 MB/s) - 已保存 “/baidu/www.baidu.com/robots.txt” [2814/2814])

FINISHED --2021-07-15 00:13:54--
Total wall clock time: 0.1s
Downloaded: 2 files, 5.1K in 0s (511 MB/s)
[root@localhost 文档]#
```

whereis - 查找二进制程序、代码等相关文件路径

whereis命令 用来定位指令的二进制程序、源代码文件和man手册页等相关文件的路径。

whereis命令只能用于程序名的搜索，而且只搜索二进制文件（参数-b）、man说明文件（参数-m）和源代码文件（参数-s）。如果省略参数，则返回所有信息。

和find相比，whereis查找的速度非常快，这是因为linux系统会将 系统内的所有文件都记录在一个数据库文件中，当使用whereis和下面即将介绍的locate时，会从数据库中查找数据，而不是像find命令那样，通过遍历硬盘来查找，效率自然会很高。但是该数据库文件并不是实时更新，默认情况下时一星期更新一次，因此，我们在用whereis和locate 查找文件时，有时会找到已经被删除的数据，或者刚刚建立文件，却无法查找到，原因就是因为数据库文件没有被更新。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
sh
whereis [选项] cmd
whereis [-bmsu] [-BMS directory... -f] filename...
```

选项

```
sh
-b          # 只搜索二进制文件
-m          # 只搜索手册文件
-s          # 只搜索源代码
-u          # 寻找不寻常的条目。如果文件没有每个请求类型的一个条目,
           # 则该文件被认为是不寻常的。因此, “Whereis -m -u *”
           # 请求当前目录中没有文档的文件。
-B          # 在指定目录下搜索二进制文件
-M          # 在指定目录下搜索手册文件
-S          # 在指定目录下搜索源代码文件
-f          # 不显示文件名前的路径，在是使用-S 、 -M、 -B选项时,
           # 必须使用这个选项
```

举例

```
sh
[sogrey@bogon 文档]$ whereis -b ls
ls: /usr/bin/ls
[sogrey@bogon 文档]$ whereis -m ls
ls: /usr/share/man/man1p/ls.1p.gz /usr/share/man/man1/ls.1.gz
[sogrey@bogon 文档]$
```


which - 查找并显示给定命令的绝对路径

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。

在环境变量PATH中搜索某个命令，返回命令的执行文件或者脚本位置，默认只显示第一个结果。这需要一个或多个参数。对于它的每个参数，它会打印出当在shell提示符下输入该参数时将执行的可执行文件的完整路径。它通过使用与bash(1)相同的算法在环境变量路径中列出的目录中搜索可执行文件或脚本来做到这一点。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
which [options] [--] programname [...]
```

sh

选项

-a, --all	# 输出所有的结果，而不是第一个
-i, --read-alias	# 从stdin中读取别名，在stdout上报告匹配的别名。
--skip-alias	# 忽略“--read-alias”
--read-function	# 从标准输入读取函数，在stdout输出
--skip-functions	# 忽略“--skip-functions”
--skip-dot	# 忽略PATH中以点开头的目录
--skip-tilde	# 跳过路径中以主目录中的波浪线和可执行文件开头的目录。
--show-dot	# 如果路径中的目录以点开始，并为该路径找到匹配的
--show-tilde	# 可执行文件，打印“./Programname”而不是完整路径。
--tty-only	# 当目录与主目录匹配时，输出一个波浪线。
--help	# 如果将该选项作以root身份调用，则忽略此选项
-v, --version	# 如果不在TTY上，则停止右边的处理选项。

返回值

它返回失败参数的数量，或者当未指定“程序名”时返回-1。

举例

```
[sogrey@bogon 文档]$ which pwd
/usr/bin/pwd
[sogrey@bogon 文档]$ which cd
/usr/bin/cd
[sogrey@bogon 文档]$ which ls
alias ls='ls --color=auto'
/usr/bin/ls
[sogrey@bogon 文档]$ which -a ls
alias ls='ls --color=auto'
/usr/bin/ls
/bin/ls
[sogrey@bogon 文档]$ which -a ll
alias ll='ls -l --color=auto'
/usr/bin/ls
/bin/ls
/usr/bin/which: no ll in (/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin:/home/sog
[sogrey@bogon 文档]$
```

xinit - Linux下X-Window系统的初始化程序

xinit命令 是Linux下X-Window系统的初始化程序，主要完成X服务器的初始化设置。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
xinit [OPTION]
```

sh

参数

- 客户端选项：客户端指令及选项；
- --：用于区分客户端选项和服务器端选项；
- 服务器端选项：服务器端选项指令及选项。

yes - 重复打印指定字符串

yes命令 在命令行中输出指定的字符串，直到yes进程被杀死。不带任何参数输入yes命令默认的字符串就是y。

反复的输出指定的字符串，直到手动停止。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
yes [STRING]...
yes OPTION
```

sh

如果不指定字符串，那么输出字符"y"

选项

```
--help          # 显示帮助文档
--version       # 显示命令版本信息
```

sh

举例

输出hello world

```
[sogrey@bogon ~]$ yes hello world
hello world
hello world
hello world
hello world
hello world
hello world^C      //使用ctrl+c强制停止
```

sh

输出字符y

sh

```
[sogrey@bogon ~]$ yes hello world
y
y
y
y
y
y^C      //使用ctrl+c强制停止
```

ypdomainname - 显示主机的NIS的域名

ypdomainname命令 显示主机的NIS的域名。

ypdomainname指令显示由函数“getdomainname”返回的主机域名，使用这个指令也可以设置一个主机NIS/YP域名。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint		Alpine Linux	Arch Linux	

语法

```
ypdomainname [-v]
```

sh

选项

-v	# 显示详细执行过程
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

举例

显示主机域名

```
[root@localhost ~]$ ypdomainname          #显示域名  
www.weijie.com  
[root@localhost ~]$ ypdomainname www.david.com #设置域名  
[root@localhost ~]$ ypdomainname          #显示域名  
www.david.com
```

sh

zcat - 显示压缩包中文件的内容

zcat命令 用于不真正解压缩文件，就能显示压缩包中文件的内容的场合。

解压有gzip压缩的文件，将解压结果送到标准输出。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
zcat [-fhVL] 文件
```

sh

选项

```
-S          # 指定gzip格式的压缩包的后缀。当后缀不是标准压缩包后缀时使用此选项;  
-c          # 将文件内容写到标注输出;  
-d          # 执行解压缩操作;  
-l          # 显示压缩包中文件的列表;  
-L          # 显示软件许可信息;  
-q          # 禁用警告信息;  
-r          # 在目录上执行递归操作;  
-t          # 测试压缩文件的完整性;  
-v          # 显示指令的版本信息;  
-1          # 更快的压缩速度;  
-9          # 更高的压缩比。  
  
-f, --force      # 强制执行  
-L, --licence    # 显示gzip的版本并且退出  
  
--help          # 显示帮助文档  
--version        # 显示命令版本信息
```

举例

```
[sogrey@bogon 文档]$ zcat -l 1.c.gz # 查看压缩包信息  
compressed      uncompressed  ratio  uncompressed_name  
    70              53   9.4%  1.c
```

sh

sh

```
[sogrey@bogon 文档]$ zcat 1.c.gz # 解压文件到标准输出
hello world,
i love linux,
love code.
```

zforce - 强制gzip格式的文件使用gz后缀名

这样可以防止gzip被压缩两次。

适用范围

RedHat RHEL Ubuntu CentOS Debian Deepin SUSE
openSUSE Fedora Linux Mint Alpine Linux Arch Linux

语法

```
zforce  gzip格式文件
```

sh

选项

无

举例

```
[sogrey@bogon 文档]$ ll
总用量 16
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 3.c
-rw-----. 1 sogrey sogrey 415 3月 8 23:50 file1.txt
-rw-----. 1 sogrey sogrey 576 3月 8 23:51 file2.txt
-rw-----. 1 sogrey sogrey 12 3月 9 00:43 list.txt
-rwx-----. 1 sogrey sogrey 90 3月 9 00:23 run.sh

[sogrey@bogon 文档]$ gzip file1.txt
[sogrey@bogon 文档]$ ll
总用量 16
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 3.c
-rw-----. 1 sogrey sogrey 337 3月 8 23:50 file1.txt.gz
-rw-----. 1 sogrey sogrey 576 3月 8 23:51 file2.txt
-rw-----. 1 sogrey sogrey 12 3月 9 00:43 list.txt
-rwx-----. 1 sogrey sogrey 90 3月 9 00:23 run.sh

[sogrey@bogon 文档]$ zforce file1.txt.gz
[sogrey@bogon 文档]$ ll
总用量 16
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 1.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 2.c
-rw-----. 1 sogrey sogrey 0 3月 9 00:44 3.c
-rw-----. 1 sogrey sogrey 337 3月 8 23:50 file1.txt.gz
-rw-----. 1 sogrey sogrey 576 3月 8 23:51 file2.txt
-rw-----. 1 sogrey sogrey 12 3月 9 00:43 list.txt
-rwx-----. 1 sogrey sogrey 90 3月 9 00:23 run.sh

[sogrey@bogon 文档]$
```

zip - 可以用来解压缩文件

zip是一种最通用的文件压缩方式，使用于unix、msdos、windows、OS等系统。如果在编译zip时包含bzip2库，zip现在也支持bzip 2压缩。当将大于4GB的文件添加到存档中时，zip会自动使用Zip 64扩展名，包含Zip 64条目的归档将被更新(如果结果的归档仍然需要Zip 64)，归档的大小将超过4GB，或者当归档中的条目数超过64K时。Zip 64也用于从标准输入中传输的档案，因为事先不知道这些档案的大小，但是选项fz可以用来强制zip创建与PKZIP 2兼容的档案(只要不需要Zip 64扩展)。必须使用PKZIP4.5兼容解压缩，例如解压缩6.0或更高版本，才能使用Zip 64扩展名提取文件。

zip程序将一个或多个压缩文件与有关文件的信息(名称、路径、日期、上次修改的时间、保护和检查信息以验证文件完整性)一起放入一个压缩存档中。可以使用一个命令将整个目录结构打包到zip存档中。对于文本文件来说，压缩比为2: 1和3: 1是常见的。zip只有一种压缩方法(通缩)，并且可以在不压缩的情况下存储文件。(如果添加了bzip 2支持，zip也可以使用bzip 2压缩，但这些条目需要一个合理的现代解压缩来解压缩。当选择bzip 2压缩时，它将通货紧缩替换为默认方法。)zip会自动为每个要压缩的文件选择更好的两个文件(通缩或存储，如果选择bzip2，则选择bzip2或Store)。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
zip [选项] zipfile list
```

sh

选项

		sh
-a, --ascii	# 将系统使用的EBCDIC文件转换为ascii	
-A, --adjust-sfx	# 调整自解压可执行档案。自解压缩的可执行存档是通过将SFX存根放在现有存档	
-b path , --temp-path path	# 为临时zip归档使用指定的路径	
-B, --binary	# 使用二进制格式，默认是文本格式	
-c, --entry-comments	# 给被压缩的文件加上注释信息	
-d, --delete	# 将指定的文件从压缩文件中删除	
-db, --display-bytes	# 显示正在运行的字节计数，显示压缩的字节和要运行的字节。	
-dc, --display-counts	# 显示已压缩的条目的运行计数和要运行的条目	
-dd, --display-dots	# 在每个条目被拉链时显示点(除了在具有自己的进度指示器的端口上)。见下面的	
-dg, --display-globaldots	# 显示存档的进度点，而不是为每个文件显示进度点。	
-ds size, --dot-size size	# 设置为显示的每个点处理的输入文件的数量。尺寸为nm格式，其中n是一个数字	
-du, --display-usize	# 显示每个条目的未压缩大小。	
-dv, --display-volume	# 显示正在读取的每个条目的卷(磁盘)号，如果读取现有存档，并将其写入	
-D, --no-dir-entries	# 不要在zip存档中为目录创建条目。	
-DF, --difference-archive	# 创建一个包含自原始存档创建以来所有新的和已更改的文件的归档文件。	
-e, --encrypt	# 压缩文件加密	
-f, --freshen	# 替代压缩文件中的旧文件，如果文件不存在，那么不会追加文件	
-F, --fix, -FF, --fixfix	# 尝试修复已经损坏的压缩文件	
-FI, --fifo	# 通常zip会跳过读取遇到的任何FIFO(命名管道)，因为如果FIFO没有被喂入，	
-FS, --filesync	# 将存档的内容与操作系统上的文件同步	
-g, --grow	# 将文件追加到已经存在的压缩文件结尾	

```

-i, --include          # 仅包含指定的文件
-j, --junk-paths      # 压缩的时候，忽略文件名中的路径，只保存文件名和内容
-J, --junk-sfx         # 从存档中删除任何预置的数据
-i, --include          # 只包含指定的文件
-la, --log-append     # 附加到现有日志文件。默认值是覆盖
-lf, --logfile-path   # 在给定路径上打开日志文件。默认情况下，该位置的任何现有文件都会被覆盖
-li, --log-info        # 在日志中包含信息消息，例如正在压缩的文件名。默认情况下，只包含命令行、
-L, --license          # 显示zip的许可证
-m, --move              # 将指定的文件移动到压缩文件
-MM, --must-match      # 所有输入模式必须至少匹配一个文件，所有找到的输入文件必须是可读的。
-n, --suffixes          # 不要尝试压缩以给定后缀命名的文件。
-nw, --no-wild          # 不要执行内部通配符处理。
-o, --latest-time       # 将zip归档的“最后修改”时间设置为在zip归档中的条目中找到的最新(最老的)
-O, --output-file        # 像往常一样处理归档更改，但不是更新现有存档，而是将新存档输出到输出文件
-p, --path                # 将相对文件路径作为存储在存档中的文件名称的一部分
-P, --password           # 加密
-q, --quite              # 静默模式，不显示执行过程
-r, --recurse-paths      # 压缩的时候，递归处理目录
-R, --recurse-patterns   # 递归遍历目录结构，从当前目录开始。
-s, --split-size          # 启用创建拆分存档并设置拆分大小
-sb, --split-bell         # 如果拆分并使用拆分暂停模式，则在zip对每个拆分目标暂停时按下铃声。
-sc, --show-command       # 显示已处理并退出的启动zip的命令行
-sf, --show-files          # 显示要操作的文件，然后退出。
-so, --show-options       # 显示在当前系统上编译的zip支持的所有可用选项
-sp, --split-pause         # 如果使用-s启用拆分，则启用拆分暂停模式
-su, --show-unicode        # 和-sf一样，如果存在，还会显示路径的unicode版本
-sU, --show-just-unicode    # 和-sf一样，仅显示路径的Unicode版本(如果存在)，否则显示路径的标准版本
-sv, --split-verbose        # 在拆分时启用各种详细的消息，显示拆分是如何进行的。
-t mmddyyyy, --from-date mmddyyyy # 不要对在指定日期之前修改的文件进行操作，其中mm是月份(00-12)，dd是月
-T, --test                  # 测试新zip文件的完整性。如果检查失败，旧的zip文件将保持不变，并且(使用
-TT, --unzip-command       # 当使用-T选项时，使用命令cmd而不是“unzip-tqq”来测试存档。
-U, --copy-entries          # 将条目从一个存档复制到另一个存档。
-u, --update                 # 更新文件，不存在的时候，直接追加。仅当zip存档中的现有条目比zip存档中
-UN, --unicode               # 确定zip应该如何处理unicode文件名
-x, --exclude                 # 压缩时不包含指定的文件
-y                           # 压缩时直接保存符号链接
-v, --verbose                  # 显示执行过程
-ws, --wild-stop-dirs        # 通配符仅在目录级别匹配。
-#                          # 使用指定的数字#调整压缩速度，其中-0表示没有压缩(存储所有文件)，-1表示
-@, --names-stdin            # 从标准输入中获取输入文件列表。每行只有一个文件名。
-h2, --more-help              # 显示扩展帮助，包括更多关于命令行格式、模式匹配和更模糊的选项。
-?, -h, --help                  # 显示帮助文档
-V, --version                  # 显示命令版本信息

```

模式匹配

- `?`，匹配任意单个字符
- `*`，匹配任意多个字符
- `[]` 匹配括号内所示范围内的任何字符(例如: [a-f], [0-9])。

退出码

- `0` 正常，没有错误。

- 2 压缩文件的意外结束。
- 3 检测到zip文件格式中的通用错误。
- 4 在程序初始化期间，zip无法为一个或多个缓冲区分配内存。
- 5 检测到zip文件格式出现严重错误。
- 6 条目太大，无法处理(例如，不使用Zip 64或试图读取现有存档太大时，输入文件大于2GB)或条目太大，不能用zip拆分。
- 7 无效注释格式。
- 8 zip-T失败或内存不足。
- 9 用户使用Control-C(或类似的)过早中止zip。
- 10 使用临时文件时，zip遇到了一个错误。
- 11 读取或查找错误。
- 12 zip无事可做。
- 13 丢失或空压缩文件。
- 14 写入文件时出错。
- 15 zip无法创建要写入的文件。
- 16 坏命令行参数。
- 18 zip无法打开要读取的指定文件。
- 19 在此系统中不支持的选项编译了zip。

例子代码

创建存档stuff.zip(假设它不存在)，并以压缩形式将所有文件放在当前目录中(zip后缀自动添加，除非存档名称已经包含一个点；这允许明确说明其他后缀)。

```
zip stuff *
zip stuff .* *
```

sh

压缩整个目录。创建存档foo.zip，其中包含当前目录中包含的目录foo中的所有文件和目录。

```
zip -r foo foo
```

sh

您可能希望创建一个包含foo中文件的zip存档，而不记录目录名foo。可以使用-j选项关闭路径，如

```
zip -j foo foo/*
```

sh

如果您缺少磁盘空间，您可能没有足够的空间来保存原始目录和相应的压缩归档文件。在这种情况下，您可以使用-m选项分步骤创建归档文件。如果foo包含子目录Tom、Dick和Harry。其中，第一个命令创建foo.zip，下两个命令添加到其中。在完成每个zip命令后，最后创建的归档文件将被删除，为下一个zip命令提供工作空间。

```
zip -rm foo foo/tom
zip -rm foo foo/dick
zip -rm foo foo/harry
```

sh

使用-s设置拆分大小并创建拆分存档。大小可选地被k(KB)、m(MB)、g(Gb)或t(Tb)之一跟随。创建目录foo的拆分存档，每个分区不大于2GB。如果foo包含5 GB的内容，并且内容未经压缩就存储在拆分存档中(为

了使这个示例变得简单), 这将创建三个分块, 分别是2GB的split.z01、2GB的split.z02和略高于1GB的split.zip。

```
zip -s 2g -r split.zip foo
```

sh

虽然zip不更新拆分档案, 但zip提供了新的选项-O(-Output-file), 允许更新拆分档案并将其保存在新的存档中。读取归档文件, 即使拆分, 也会添加foo.c和bar.c文件, 并将结果存档写入outArchive.zip。如果inArchive.zip是拆分的, 那么outArchive.zip默认为相同的拆分大小。请注意, outArchive.zip和使用它创建的任何拆分文件都会在没有警告的情况下被覆盖。这种情况将来可能会改变。

```
zip inarchive.zip foo.c bar.c --out outarchive.zip
```

sh

将/home/Blinux/html/这个目录下所有文件和文件夹打包为当前目录下的html.zip:

```
zip -q -r html.zip /home/Blinux/html
```

sh

上面的命令操作是将绝对地址的文件及文件夹进行压缩, 以下给出压缩相对路径目录, 比如目前在Bliux这个目录下, 执行以下操作可以达到以上同样的效果:

```
zip -q -r html.zip html
```

sh

比如现在我的html目录下, 我操作的zip压缩命令是:

```
zip -q -r html.zip *
```

sh

压缩 example/basic/ 目录内容到 basic.zip 压缩包中 -x 指定排除目录, 注意没有双引号将不起作用。

```
zip -r basic.zip example/basic/ -x "example/basic/node_modules/*" -x "example/basic/build/*" -x
```

sh

上面压缩解压出来, 内容存放在 example/basic/, 如果想存放到根目录, 进入目录进行压缩, 目前没有找到一个合适的参数来解决此问题。

```
cd example/basic/ && zip -r basic.zip . -x "node_modules/*" -x "build/*" -x "coverage/*"
```

sh

压缩效率选择:

```
zip -9 # 1-9 faster->better
```

sh

创建 public_html 目录下忽略所有文件和文件夹, 排除包括文本 backup 的所有文件。

```
$ zip -r public_html.zip public_html -x *backup*
```

sh

htdocs 目录忽略 .svn 文件或 git 的文件和目录下创建所有文件的归档。

```
$ zip -r httpdocs.zip httpdocs --exclude *.svn* --exclude *.git*
```

sh

httpdocs 目录忽略的所有文件，并与 .log 结尾的目录下创建所有文件的归档。

```
$ zip -r httpdocs.zip httpdocs --exclude "*.log"
```

sh

举例

压缩文件

```
[sogrey@bogon 文档]$ ls  
1.c 2.c 3.c  file1.txt.gz  file2.txt  list.txt  run.sh  
[sogrey@bogon 文档]$ zip -r res . -i *.c #压缩当前目录下的所有.c文件  
  adding: 2.c (stored 0%)  
  adding: 3.c (stored 0%)  
  adding: 1.c (stored 0%)  
[sogrey@bogon 文档]$ ls  
1.c 2.c 3.c  file1.txt.gz  file2.txt  list.txt  res.zip  run.sh  
[sogrey@bogon 文档]$
```

sh

2) 给压缩文件添加、删除内容

```
[sogrey@bogon 文档]$ zip res.zip -g file2.txt #追加file2.txt  
  adding: file2.txt (deflated 28%)  
[sogrey@bogon 文档]$ ls  
1.c 2.c 3.c  file1.txt.gz  file2.txt  list.txt  res.zip  run.sh  
[sogrey@bogon 文档]$ zip res.zip -d file2.txt #删除file2.txt  
  deleting: file2.txt  
[sogrey@bogon 文档]$
```

sh

3) 压缩时设置密码

```
[sogrey@bogon 文档]$ zip -er res2 . -i *.c #设置密码，需要输入两次密码  
Enter password:  
Verify password:  
  adding: 2.c (stored 0%)  
  adding: 3.c (stored 0%)  
  adding: 1.c (stored 0%)  
[sogrey@bogon 文档]$
```

sh

安装

CentOS:

```
yum install -y unzip zip
```

sh

zipinfo - 用来列出压缩文件信息

在不解压的情况下，获取zip压缩文件的的详细信息。zipinfo列出了ZIP档案中有关文件的技术信息，最常见的就是在MS-DOS系统上。这些信息包括文件访问权限、加密状态、压缩类型、版本和操作系统或压缩程序的文件系统等。默认的行为(没有选项)是列出存档中每个文件的单线条目，标题和拖车行为整个归档提供摘要信息。格式是Unix'ls-l'和'unzip-v'输出之间的交叉。见下文的详细说明。请注意，zipinfo与解压缩程序相同(在Unix下，链接到它)；然而，在某些系统上，在解压缩编译时可能忽略了zipinfo支持。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
zipinfo [-12smlvhMtTz] file[.zip] [file(s) ...] [-x xfile(s) ...]
```

sh

选项

file[.zip]	# ZIP存档的路径。如果文件规范是通配符，则按操作系统(或文件系统)确定的顺序处理每个匹配的文件。
[file(s)]	# 要处理的归档成员的可选列表，用空格分隔。(用VMSCLI定义的VMS版本必须用逗号分隔文件。) 正确的格式是“file1 file2 ...”。
[-x xfile(s)]	# 要排除在处理之外的归档成员的可选列表。

sh

-1	# 只列出文件名称
-2	# 和“-1”类似，但是可以单配其他选项“-h”、“-i”、“-z”
-s	# 类似“ls -l”的短格式
-m	# 类似“ls -l”的medium格式
-l	# 类似“ls -l”的长格式
-v	# 显示zip文件的详细信息
-h	# 只列出压缩包的文件名、大小、包内文件数目
-M	# 类似more命令，分屏显示
-t	# 列出压缩文件内的文件数目、压缩前后文件大小、压缩率
-T	# 将压缩包内文件的日期以年、月、日、时、分、秒的顺序列出
-z	# 显示压缩文件的注释信息
--help	# 显示帮助文档
--version	# 显示命令版本信息

sh

用例

要获得包含头行和总计行的ZIP存档Storage.zip的完整内容的基本的、简短的列表，只使用归档名称作为zipinfo的参数。

```
zipinfo storage
```

sh

要生成一个基本的、长格式的列表(而不是冗长的), 包括标题和总计行, 请使用-l:

```
zipinfo -l storage
```

sh

要列出没有标题和总计行的存档的完整内容, 可以忽略-h和-t选项, 或者显式地指定内容:

```
zipinfo --h-t storage  
zipinfo storage \*
```

sh

默认情况下, 若要关闭汇总行, 请使用环境变量(此处假定为C shell):

```
setenv ZIPINFO --t  
zipinfo storage
```

sh

为了再次获得第一个示例的完整、简短的列表, 考虑到前面示例中设置了环境变量, 有必要显式地指定-s选项, 因为-t选项本身意味着只打印页脚行

```
setenv ZIPINFO --t  
zipinfo -t storage [only totals line]  
zipinfo -st storage [full listing]
```

sh

若要以中等格式列出存档中单个文件的信息, 请显式指定文件名。

```
zipinfo -m storage unshrink.c
```

sh

任何成员文件的规范都将覆盖默认的标头和总计行; 将只打印有关请求文件的一行信息。这是在请求有关单个文件的信息时所期望的直观结果。对于多个文件, 了解总的压缩和未压缩大小通常是有用的; 在这种情况下, 可以显式地指定-t:

```
zipinfo -mt storage "*.[ch]" Mak\*
```

sh

要获取有关ZIP存档的最大信息, 请使用详细选项。如果操作系统允许, 通常明智的做法是将输出输送到Unix More(1)这样的过滤器中:

```
zipinfo -v storage | more
```

sh

要查看存档中最近修改的文件, 请结合外部排序实用程序(如Unix Sort(1)和sed(1)使用-T选项

```
zipinfo -T storage | sort -nr -k 7 | sed 15q
```

sh

举例

sh

```
[sogrey@bogon 文档]$ ll  
总用量 24  
-rw----- 1 sogrey sogrey 0 3月 9 00:44 1.c  
-rw----- 1 sogrey sogrey 0 3月 9 00:44 2.c  
-rw----- 1 sogrey sogrey 0 3月 9 00:44 3.c  
-rw----- 1 sogrey sogrey 337 3月 8 23:50 file1.txt.gz  
-rw----- 1 sogrey sogrey 576 3月 8 23:51 file2.txt  
-rw----- 1 sogrey sogrey 12 3月 9 00:43 list.txt  
-rw----- 1 sogrey sogrey 508 3月 11 00:05 res2.zip  
-rw----- 1 sogrey sogrey 424 3月 11 00:04 res.zip  
-rwx----- 1 sogrey sogrey 90 3月 9 00:23 run.sh  
[sogrey@bogon 文档]$ zipinfo res.zip # 查看压缩包内文件信息  
Archive: res.zip  
Zip file size: 424 bytes, number of entries: 3  
-rw----- 3.0 unx 0 bx stor 21-Mar-09 00:44 2.c  
-rw----- 3.0 unx 0 bx stor 21-Mar-09 00:44 3.c  
-rw----- 3.0 unx 0 bx stor 21-Mar-09 00:44 1.c  
3 files, 0 bytes uncompressed, 0 bytes compressed: 0.0%
```

sh

```
[sogrey@bogon 文档]$ zipinfo -h res.zip # 只显示压缩包大小、文件数目  
Archive: res.zip  
Zip file size: 424 bytes, number of entries: 3  
[sogrey@bogon 文档]$
```

znew - 将.Z压缩包重新转化为gzip命令压缩的.gz压缩包

znew命令 用于将使用compress命令压缩的".Z"压缩包重新转化为使用gzip命令压缩的".gz"压缩包。

将compress压缩成的".Z"文件，转换成".gz"格式的文件。ZNew将文件从.z(压缩)格式重新压缩到.gz(Gzip)格式。如果要重新压缩已以gzip格式的文件，请重命名该文件以强制.z扩展名，然后应用znew。

适用范围

RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin	SUSE
openSUSE	Fedora	Linux Mint	Alpine Linux		Arch Linux	

语法

```
znew [选项] name.Z
```

sh

选项

```
-f          # 强制转换  
-t          # 在删除原始的.Z文件之前，测试新生成的.gz文件  
-v          # 显示详细信息  
-9          # 使用最高压缩率  
-P          # 使用管道功能  
-K          # 如果.Z文件比新生成的.gz文件小，那么保留.Z文件  
  
--help      # 显示帮助文档  
--version   # 显示命令版本信息
```

sh

举例

```
[sogrey@bogon 文档]$ ls  
1.c 2.c 3.c file1.txt.gz  file2.txt  list.txt  res2.zip  res.zip  run.sh  
[sogrey@bogon 文档]$ compress file2.txt # compress压缩  
[sogrey@bogon 文档]$ ls  
1.c 2.c 3.c file1.txt.gz  file2.txt.Z  list.txt  res2.zip  res.zip  run.sh  
[sogrey@bogon 文档]$ znew file2.txt.Z # 将.Z压缩包重新转化为gzip命令压缩的.gz压缩包  
[sogrey@bogon 文档]$ ls  
1.c 2.c 3.c file1.txt.gz  file2.txt.gz  list.txt  res2.zip  res.zip  run.sh  
[sogrey@bogon 文档]$
```

sh

Linux 上安装Java&JDK

Ubuntu

环境

- Ubuntu 20.04.2 LTS

在线安装JDK(适用于可连接互联网的用户)

1. 先检查是否已安装JDK

输入以下命令，如果输出jdk版本号，则已安装，不用再往下看了。

```
java -version
```

sh

如果没有输出版本号，而是输出类似于以下提示，`java` 未找到，则还未安装。同时也看到下面给出几个jdk在线安装的命令：

```
sogrey@sogrey-VirtualBox:~/桌面$ java -version

Command 'java' not found, but can be installed with:

sudo apt install openjdk-11-jre-headless # version 11.0.11+9-0ubuntu2~20.04, or
sudo apt install default-jre           # version 2:1.11-72
sudo apt install openjdk-13-jre-headless # version 13.0.7+5-0ubuntu1~20.04
sudo apt install openjdk-16-jre-headless # version 16.0.1+9-1~20.04
sudo apt install openjdk-8-jre-headless  # version 8u292-b10-0ubuntu1~20.04
```

sh

我们试着安装jdk8。

2. 在线安装

根据提示我们输入以下命令，安装jdk8：

```
sudo apt install openjdk-8-jre-headless
```

sh

注意：此处需要输入sudo密码

3. 验证安装是否成功

输入以下命令，输出版本号，安装成功。

```
sogrey@sogrey-VirtualBox:~/桌面$ java -version
openjdk version "1.8.0_292"
OpenJDK Runtime Environment (build 1.8.0_292-8u292-b10-0ubuntu1~20.04-b10)
OpenJDK 64-Bit Server VM (build 25.292-b10, mixed mode)
sogrey@sogrey-VirtualBox:~/桌面$
```

sh

4. 卸载JDK

```
sogrey@sogrey-VirtualBox:~/桌面$ whereis java
java: /usr/bin/java /usr/share/java /usr/share/man/man1/java.1.gz
sogrey@sogrey-VirtualBox:~/桌面$
```

sh

卸载

```
sudo apt-get autoremove default-jdk
```

sh

CentOS

Linux命令

- Linux命令大全搜索工具，内容包含Linux命令手册、详解、学习、搜集。<https://git.io/linux> [github](#)
- https://linuxtools-rst.readthedocs.io/zh_CN/latest/index.html https://github.com/me115/linux-tools_rst
- <https://linuxgsm.com/> <https://github.com/GameServerManagers/LinuxGSM>

Linux

Linux

<https://www.linux.org/>

下载地址：

- <https://www.linux.org/pages/download/>

多样的Linux发行版

Linux的发行版多种多样：

Linux	RedHat	RHEL	Ubuntu	CentOS	Debian	Deepin
SUSE	openSUSE	Fedora	Linux Mint	MX Linux	Alpine Linux	
Arch Linux	FreeBSD	Gentoo	Linux Oracle			

RedHat RHEL



官网：<https://www.redhat.com/zh> [RHEL](#)

下载地址：

- <https://access.redhat.com/downloads/>

Ubuntu



官网：<https://ubuntu.com/> [Ubuntu Server](#)

下载地址：

- <https://ubuntu.com/download>
- <https://ubuntu.com/download/server>

CentOS



官网: <https://www.centos.org/>

下载地址:

- <https://wiki.centos.org/Download>

Rocky Linux (CentOS 替代品)

□

官网: <https://rockylinux.org/>

下载地址:

- <https://rockylinux.org/download>

AlmaLinux (另一个 CentOS 替代品)

□

官网: <https://almalinux.org/>

下载地址:

- <https://mirrors.almalinux.org/isos.html>

Huawei EulerOS (基于 CentOS)

□

官网: <https://developer.huaweicloud.com/ict/en/site-euleros/euleros>

下载地址:

- <https://developer.huaweicloud.com/en-us/euleros/download.html?developplan=Other>

openEuler

官网: <https://openeuler.org/>

下载地址:

- <https://openeuler.org/zh/mirror/list/>

OpenEuler安装UKUI

Debian

□

官网: <https://www.debian.org/>

下载地址:

- <https://www.debian.org/distrib/>
- <https://www.debian.org/CD/>

Deepin

□

官网: <https://www.deepin.org/>

下载地址:

- <https://www.deepin.org/zh/download/>

SUSE

官网: <https://www.suse.com/>

下载地址:

- <https://www.suse.com/download/>

openSUSE

□

官网: <https://www.opensuse.org/>

下载地址:

- <https://software.opensuse.org/distributions/tumbleweed>

Fedora

□

官网: <https://getfedora.org/>

下载地址:

- <https://getfedora.org/en/workstation/download/>
- <https://getfedora.org/en/server/download/>
- [Fedora CoreOS] <https://getfedora.org/en/coreos/download>

Linux Mint

□

官网: <https://linuxmint.com/>

下载地址:

- <https://linuxmint.com/download.php>

Alpine Linux

官网: <https://www.alpinelinux.org/>

下载地址:

- <https://alpinelinux.org/downloads/>
- <https://mirrors.alpinelinux.org/>

Arch Linux

官网: <https://archlinux.org/>

下载地址:

- <https://archlinux.org/download/>

FreeBSD

官网: <https://www.freebsd.org/>

下载地址:

- <https://www.freebsd.org/where.html>

MX Linux

□

官网: <https://mxlinux.org/>

下载地址:

- <http://mirror.umd.edu/mxlinux-iso/MX/Final/>
- <http://mirror.cogentco.com/pub/linux/mxlinux-iso/MX/Final/>

Kali Linux

□

官网: <https://www.kali.org/>

下载地址:

- <https://www.kali.org/downloads/>

Gentoo

□

官网: <https://www.gentoo.org/>

下载地址:

- <https://www.gentoo.org/downloads/>

Mandriva

□

官网: <https://www.openmandriva.org/>

下载地址:

- <https://www.openmandriva.org/en/download>

Linux | Oracle

□

官网: <https://www.oracle.com/linux/>

下载地址:

- <https://www.oracle.com/linux/technologies/oracle-linux-downloads.html>

Turbolinux

官网: <http://www.turbolinux.org/>

优麒麟 UbuntuKylin

□

官网: <https://www.ubuntukylin.com/>

下载地址:

- <https://www.ubuntukylin.com/downloads/show.php?id=451&lang=en>

银河麒麟 KylinOS (国产)

□

官网: <https://eco.kylinos.cn/>

下载地址:

- <https://eco.kylinos.cn/partners/mirror.html>

MaboxLinux

□

□

官网: <https://maboxlinux.org/>

下载地址:

- <https://maboxlinux.org/users-guide/download-and-installation/>
 - <https://sourceforge.net/projects/mabox-linux/files/latest/download>
-

Bodhi Linux

□

官网: <https://www.bodhilinux.com/>

下载地址:

<https://www.bodhilinux.com/download/>

Puppy Linux

□

官网: <https://puppylinux.com/>

下载地址:

<https://puppylinux.com/index.html#download>

Silverblue

□

官网: <https://silverblue.fedoraproject.org>

下载地址:

<https://silverblue.fedoraproject.org/download>

Elementary OS

□

官网: <https://elementary.io/>

Manjaro

□

官网: <https://manjaro.org>

下载地址:

<https://manjaro.org/download/>

Zorin OS

□

官网: <https://zorinos.com/>

下载地址:

<https://zorinos.com/download/>

eXtern OS

□

官网: <https://externos.io/>

下载地址:

<https://externos.io/latest/>

Nitrx

□

官网: <https://nxos.org/>

下载地址:

<https://fosstorrents.com/distributions/nitrx/>

Robolinux

官网: <https://www.roboLinux.org/>

Solus

□

官网: <https://getsol.us/>

下载地址:

<https://getsol.us/download/>

CutefishOS [国产]

□

官网: <https://cn.cutefishos.com/>

论坛:

- <https://bbs.cutefishos.com/>

- <https://forum.cutefishos.com/>

下载地址:

- 预览版 [预览版](#)
-

ClearLinux (Intel Corporation)

□

官网: <https://clearlinux.org>

下载地址:

- <https://clearlinux.org/downloads>
-

UbuntuDDE

□

官网: <https://ubuntudde.com/>

下载地址:

- <https://ubuntudde.com/download/>
-

Parrot OS(基于 Debian)

□

官网: <https://www.parrotsec.org/>

下载地址:

- <https://www.parrotsec.org/download/>
-

Linux Lite(基于 Ubuntu) 开始向 Windows 看齐

□

官网: <https://www.linuxliteos.com>

下载地址:

- <https://www.linuxliteos.com/download.php>
-

Linuxfx(基于 Ubuntu) 巴西人开发酷似Windows 11

□

官网: <https://linuxfx.org/>

下载地址:

- <https://linuxfx.org/index.php/downloads/x86-64-bit-pc>
-

欢迎补充。

资料

- [2021 年最佳 Linux 发行版推荐](#)