

Linux学习 笔记

linux系统安装

系统分区

第一步系统分区

- 主分区:最多有四个
- 扩展分区:
 - 最多只能有一个
 - 主分区加扩展分区最多有四个
 - 不能写入数据，只能包含逻辑分区
- 逻辑分区

第二步格式化

- **格式化**(高级格式化)又称逻辑格式化，它是指根据用户选定的文件系统(如FAT16,FAT32,NTFS,EXT2,EXT3,EXT4等),在磁盘的特定区域写入特定数据，在分区中划分出一片用于存放文件分配表，目录表等用于文件管理的磁盘空间。

第三步设备文件名

- 硬件设备文件名:('/' 为根目录 dev为硬件目录)

| 硬 件 | 设备文件名 |
|-----------------|---------------------|
| IDE硬盘 | /dev/hd[a-d] |
| SCSI/SATA/USB硬盘 | /dev/sd[a-p] |
| 光驱 | /dev/cdrom或/dev/sr0 |
| 软盘 | /dev/fd[0-1] |
| 打印机 (25针) | /dev/lp[0-2] |
| 打印机 (USB) | /dev/usb/lp[0-15] |
| 鼠标 | /dev/mouse |

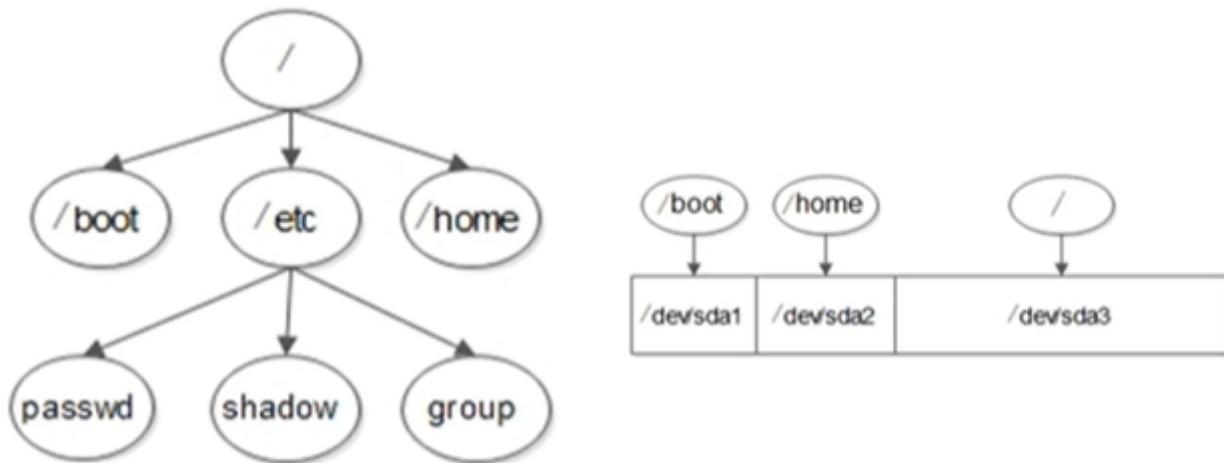
- 分区设备文件名字
 - 设备文件名
 - /dev/hda1 (IDE硬盘接口)

- /dev/sda1 (SCSI硬盘接口, SATA硬盘接口)
- 现在常用的是SATA硬盘接口

第四步挂载(只有空文件夹可以挂载)

- 必须分区
 - / (根分区)
 - swap分区 (交换分区, 内存2倍, 不超过2GB)
- 推荐分区
 - /boot(启动分区, 200M)

文件结构:



在linux里根目录(/)里的子目录可以和根目录不在同一个分区,如上图。

总结:

- 分区: 把大硬盘分为小的逻辑分区。
- 格式化: 写入文件系统
- 分区设备文件名: 给每一个分区定义设备文件名。
- 挂载: 给每个分区分配挂载点(这个挂载点必须是目录而且必须是空目录)。

linux系统安装

安装说明

安装时可以选择安装的类型有很多: Desktop(带有桌面), Minimal(最小化安装,服务器推荐)等等。 **安装的服务越少, 系统出错的概率越低**, 所以如果用来当服务器使用选择Minimal, 最小化安装可以少很多服务, 使计算机有跟多的资源为用户服务。

安装日志

- /root/install.log：存储了安装在系统中的软件包及其版本信息。
- /root/install.log.syslog：存储了安装过程中留下的事件记录。
- /root/anaconda-ks.cfg：以Kickstart配置文件的格式记录安装过程中设置的选项信息。

远程登陆管理工具

ip选项操作

```
ip [选项] 操作对象{link|addr|route...}

# 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

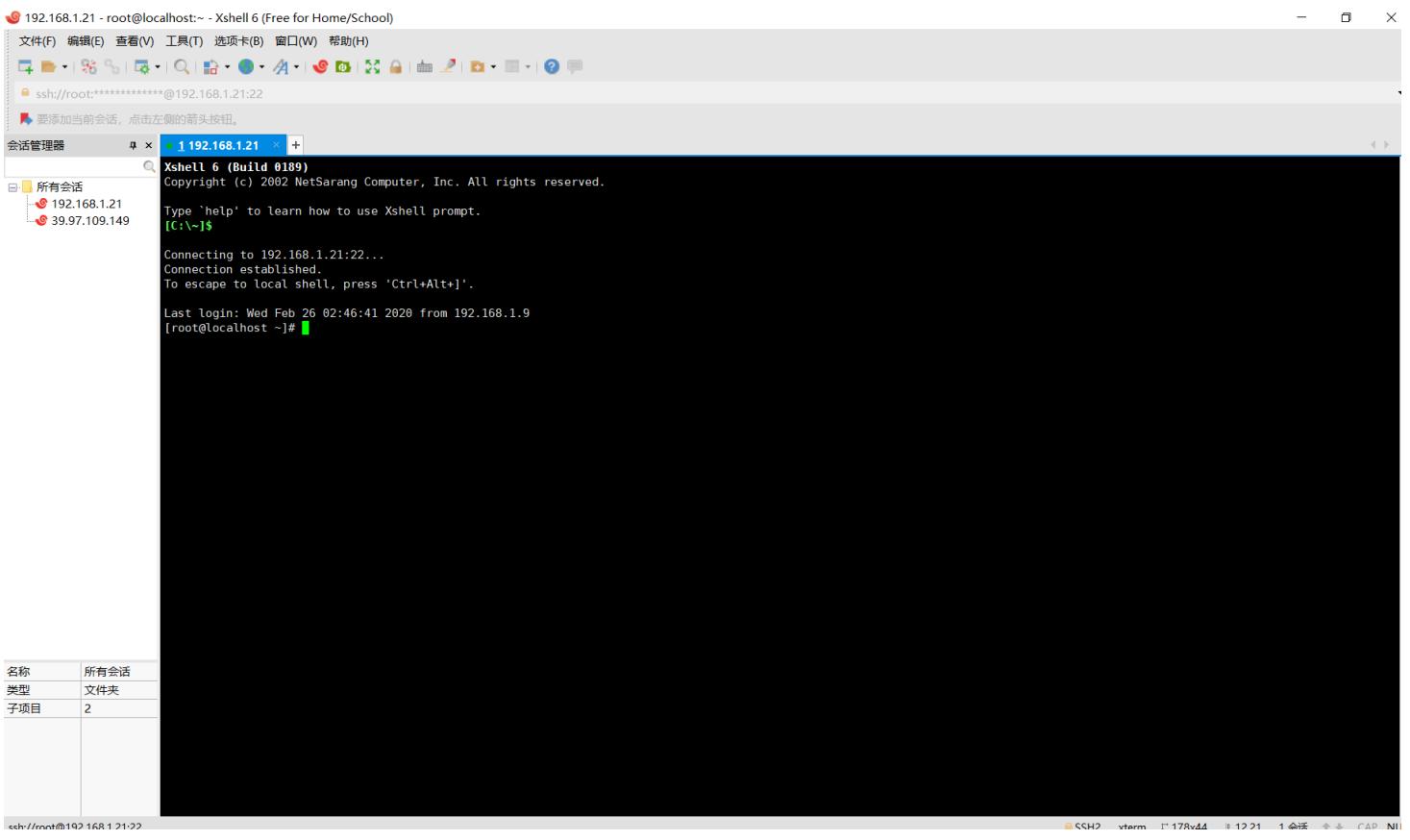
# ip link show           # 显示网络接口信息
# ip link set eth0 up    # 开启网卡
# ip link set eth0 down   # 关闭网卡
# ip link set eth0 promisc on # 开启网卡的混合模式
# ip link set eth0 promisc offi # 关闭网卡的混个模式
# ip link set eth0 txqueuelen 1200 # 设置网卡队列长度
# ip link set eth0 mtu 1400    # 设置网卡最大传输单元

#ip route show 或 ip route list 或 route -n # 查看路由(网关)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 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
```

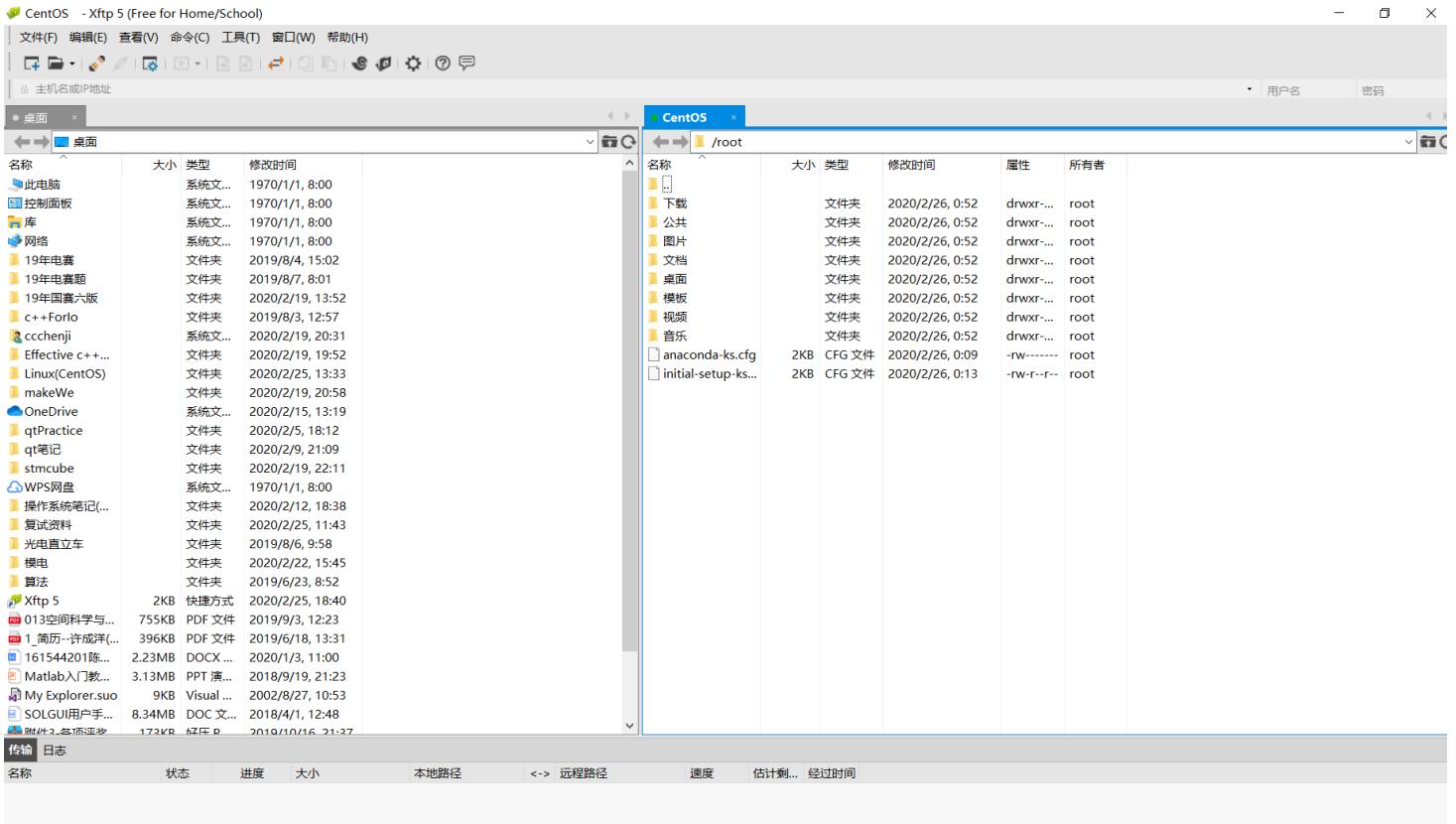
service network restart(重启网络服务)

利用xshell远程联接服务器

xshell工具可以远程联接服务器，可以自行下载。



使用xftp工具可以和linux服务器之间传输文件



linux的一些注意事项

- linux严格区分大小写
- linux中所有内容以文件形式保存，包括硬件
 - 硬盘文件是/dev/sd[a-p]
 - 光盘文件是/dev/sr0等
- linux不靠扩展名区分文件类型(后缀名只是为了方便管理员管理和区分)
 - 网页文件:".html", ".php"
 - 脚本文件:"*.sh"
 - 配置文件:"*.conf"
- linux所有的存储设备都必须**挂载**之后用户才能使用，包括硬盘，U盘和光盘
- Windows下的程序不能直接在Linux安装和运行

linux各目录的作用

| 目录名 | 目录作用 |
|-----------------|---|
| /bin/ | 存放系统命令的目录，普通用户和超级用户都可以执行。不过放在/bin下的命令在单用户模式下也可以执行 |
| /sbin/ | 保存和系统环境设置相关的命令，只有超级用户可以使用这些命令进行系统环境设置，但是有些命令可以允许普通用户查看 |
| /usr/bin/ | 存放系统命令的目录，普通用户和超级用户都可以执行。这些命令和系统启动无关，在单用户模式下不能执行 |
| /usr/sbin/ | 存放根文件系统不必要的系统管理命令，例如多数服务程序。只有超级用户可以使用。大家其实可以注意到Linux的系统，在所有“sbin”目录中保存的命令只有超级用户可以使用，“bin”目录中保存的命令所有用户都可以使用 |
| /boot/ | 系统启动目录，保存系统启动相关的文件，如内核文件和启动引导程序(grub)文件等 |
| /dev/ | 设备文件保存位置。我们已经说过Linux中所有内容以文件形式保存，包括硬件。那么这个目录就是用来保存所有硬件设备文件的 |
| /etc/ | 配置文件保存位置。系统内所有采用默认安装方式(rpm安装)的服务的配置文件全部都保存在这个目录当中，如用户账户和密码，服务的启动脚本，常用服务的配置文件等 |
| /home/ _____ | 普通用户的家目录。建立每个用户时，每个用户要有一个默认登录位置，这个位置就是这个用户的家目录，所有普通用户的家目录就是在/home下建立一个和用户名相同的目录。如用户user1的家目录就是/home/user1 |
| /lib/ | 系统调用的函数库保存位置 |
| /lost+found/ | 当系统意外崩溃或机器意外关机，而产生一些文件碎片放在这里。当系统启动的过程中fsck工具会检查这里，并修复已经损坏的文件系统。这个目录只在每个分区中出现，例如/lost+found就是根分区的备份恢复目录，/boot/lost+found就是/boot分区的备份恢复目录 |
| /media/ | 挂载目录。系统建议是用来挂载媒体设备的，例如软盘和光盘 |
| /mnt/ | 挂载目录，早期Linux中只有这一个挂载目录，并没有细分。现在这个目录系统建议挂载额外设备，如U盘，移动硬盘和其他操作系统的分区 |
| /misc/ | 挂载目录。系统建议用来挂载NFS服务的共享目录。我们在刚刚已经解释了挂载，童鞋们应该知道只要是一个已经建立的空目录就可以作为挂载点。那么系统虽然准备了三个默认挂载目录/media、/mnt、/misc，但是到底在哪个目录中挂载什么设备都可以由管理员自己决定。例如超哥接触Linux的时候，默认挂载目录只有/mnt一个，所以养成了在/mnt下建立不同目录挂载不同设备的习惯。如/mnt/cdrom挂载光盘，/mnt/usb挂载U盘，这都是可以的 |
| /opt/ | 第三方安装的软件保存位置。这个目录就是放置和安装其他软件的位置，我手工安装的源码包软件都可以安装到这个目录当中。不过我还是更加习惯把软件放置到/usr/local/目录当中，也就是说/usr/local/目录也可以用来安装软件 |

| | |
|--------|--|
| /proc/ | 虚拟文件系统，该目录中的数据并不保存到硬盘当中，而是保存到内存当中。主要保存系统的内核，进程，外部设备状态和网络状态等。如 /proc/cpuinfo 是保存 CPU 信息的，/proc/devices 是保存设备驱动的列表的，/proc/filesystems 是保存文件系统列表的，/proc/net/ 是保存网络协议信息的 |
| /sys/ | 虚拟文件系统。和 /proc 目录相似，都是保存在内存当中的，主要是保存于内核相关信息的 |
| /root/ | 超级用户的家目录。普通用户家目录在 “/home” 下，超级用户家目录直接在 “/” 下 |
| /srv/ | 服务数据目录。一些系统服务启动之后，可以在这个目录中保存所需要的数据 |
| /tmp/ | 临时目录。系统存放临时文件的目录，该目录下所有用户都可以访问和写入。我们建议此目录中不能保存重要数据，最好每次开机都把该目录清空 |
| /usr/ | 系统软件资源目录。注意 usr 不是 user 的缩写，而是 “ Unix Software Resource ” 的缩写，所以不是存放用户数据，而是存放系统软件资源的目录。系统中安装的软件大多数保存在这里，所以除了 /usr/bin/ 和 /usr/sbin/ 这两个目录，我在介绍几个 /usr/ 下的二级目录 |
| /var/ | 动态数据保存位置。主要保存缓存、日志以及软件运行所产生的文件 |

服务器注意事项

- 远程服务器不允许关机，只能重启(远程服务器一但关机，只能靠按电源键重启)
- 重启时应该关闭服务。(如果不关闭服务直接重启服务器那么导致服务器崩溃)
- 不要再服务器访问高峰运行高负载命令
- 远程配置防火墙时不要把自己提出服务器
- 指定合理的密码规范并定期更新
- 合理分配权限
- 定期备份重要数据和日志

linux常用命令

文件处理命令

命令格式与目录处理命令ls

命令格式：命令 [-选项] [参数] (中括号代表可选)

- 例：ls -la /etc

说明：

- 个别命令使用不遵循此格式
- 当有多个选型时，可以写在一起
- 简化选项与完整选项 -a 等于 --all

目录处理命令ls

命令名称:ls

命令英文原意:list

命令所在路径:/bin/ls

执行权限:所有用户

功能描述:显示目录文件

语法:ls 选项[-ald] [文件或目录]

-a显示所有文件，包括隐藏文件(以'.'开头的是隐藏文件，隐藏文件的目的是因为这个一般是系统文件，如果没有必要不要去操作它,有些病毒会把自己伪装成隐藏文件,想要将文件搞成隐藏文件加'.'

-l详细信息显示

```
[root@localhost 桌面]# ls -l
总用量 4
-rw-r--r--. 1 root root 72 2月 26 05:13 hello.c
[root@localhost 桌面]#
```

说明:这里的1代表文件引用的次数，第一个"root"代表文件的所有者(user),第二个root代表文件的所属组,72代表文件大小，后面的日期是文件的最后一次修改日期，然后是文件名。PS:命令可以连用如 -la

这里的**-rw-r--r--**

-文件类型(- 二进制文件，d 目录，l 软链接文件)

| r <u>w</u> - | r-- | r-- |
|--------------|------|------|
| u | g | o |
| u所有者 | g所属组 | o其他人 |
| 读写 | 读 | 读 |

注解:r读 w写 x执行

- **-d**查看目录属性("ls -ld" 查看当前目录的详细信息,d一般与l连用。 "ls -ld /root 查看root目录的详细信息")
- **-i**查看文件或者目录的i节点(i节点是操作系统用来查找文件的索引)

目录处理命令:mkdir

命令名称: mkdir

命令英文原意: make directories

命令所在路径: /bin/mkdir

执行权限: 所有用户

语法: mkdir -p [目录名]

功能描述: 创建新目录, -p 递归创建

范例:

- \$ mkdir /temp/
- \$ mkdir -p /temp/image/first

目录处理命令:cd

命令名称: cd

命令英文原意: change directity

命令所在路径: shell内置命令

执行权限: 所有用户

语法: cd[目录]

功能描述: 切换目录

范例:

- \$ cd /temp/image/first 切换到指定目录
- \$ cd .. 回到上一级目录

目录处理命令:pwd

命令名称: pwd

命令英文原意: print working directory

命令所在路径: /bin/pwd

执行权限: 所有用户

语法: pwd

功能描述: 显示当前目录

范例: \$pwd

/temp/image

文件处理命令:rmdir

命令名称: rmdir

命令英文原意: remove empty directories

命令所在路径: /bin/rmdir

执行权限: 所有用户

语法: rmdir[目录名称]

功能描述: 删除空目录

范例: \$ rmdir /temp/image/first

目录处理命令:cp

命令名称：cp
命令英文原意：copy
命令所在路径：/bin/cp
执行权限：所有用户
语法：cp -rp [原文件或目录] [目标目录] (-r 复制目录， -p 保留文件属性)
功能描述：复制文件或目录

目录处理命令:mv

命令名称：mv
命令英文原意：move
命令所在路径：/bin/mv
执行权限：所有用户
语法：mv [原文件或目录] [目标目录]
功能描述：剪切文件、改名

目录处理命令:rm

命令名称：rm
命令英文原意：remove
命令所在路径：/bin/rm
执行权限：所有用户
语法：rm -rf [文件或目录] (-r 删除目录， -f 强制执行)
功能描述：删除文件

文件处理命令:touch

命令名称：touch
命令所在路径：/bin/touch
执行权限：所有用户
语法：touch[文件名]
功能描述：创建空文件
范例：\$touch test.txt

文件处理命令:cat

命令名称：cat
命令所在路径：/bin/cat
执行权限：所有用户
语法：cat[文件名]

功能描述：显示文件内容 (-n 显示行号)

范例：

- \$ cat /etc/issue
- \$ cat -n /etc/services

文件处理命令:more

命令名称： more

命令所在路径： /bin/more

执行权限： 所有用户

语法： more[文件名]

- (空格)或f 翻页
- b向前翻页
- (Enter) 换行
- q或Q 退出

功能描述:分页显示文件内容

范例： \$more /etc/services

文件处理命令:less

命令名称： less

命令所在路径： /usr/bin/less

执行权限： 所有用户

语法： less[文件名]

- more的语法都适用
- '/' 后面输入关键字可以在文件中查找关键字 然后按n键可以向下查找

功能描述： 分页显示文件内容(可向上翻页)

范例： \$ less /etc/services

文件处理命令:head

命令名称： head

命令所在路径： /usr/bin/head

执行权限： 所有用户

语法： head[文件名]

功能描述： 显示文件前面几行 (-n 指定行数 默认10行)

范例： \$ head -n 20 /etc/services

文件处理命令:tail

命令名称: tail

命令所在路径: /usr/bin/tail

执行权限: 所有用户

语法: tail[文件名]

功能描述: 显示文件后面几行 (-n 指定行数,默认10行 -f 动态显示文件末尾内容(ctrl+c退出该状态))

范例: \$ tail -n 18 /etc/services

文件处理命令: ln

命令名称: ln

命令英文原意: link

命令所在路径: /bin/ln

执行权限: 所有用户

语法: ln -s [原文件] [目标文件] (-s 创建软连接)

功能描述: 生成链接文件(软连接文件类似window的快捷方式)

范例:

- \$ ln -s /etc/issue /tmp/issue.soft (创建文件/etc/issue的软连接/tmp/issue.soft)
- \$ ln /etc/issue /tmp/issue.hard(创建文件/etc/issue的硬链接/tmp/issue.hard)

硬链接特征:

1. 拷贝cp -p +同步更新 echo"[mcusorcerer.cn](#)">>/etc/issue
2. 通过i节点识别
3. 不能跨分区
4. 不能针对目录使用

权限管理命令

权限管理命令:chmod

命令名称: chmod

命令英文原意: change the permissions mode of a file

命令所在路径: /bin/chmod

执行权限: 所有用户

语法: chmod [{ugoa}{+-}{rwx}][文件或目录] [mode=421] [文件或目录] -R 递归修改

功能描述: 改变文件或目录权限

权限的数字表示

- r----4
 - w---2
 - x---1
- rw-rw-r-- 7 6 4

文件目录权限总结

| 代表字符 | 权限 | 对文件的含义 | 对目录的含义 |
|------|------|----------|---------------|
| r | 读权限 | 可以查看文件内容 | 可以列出目录中的内容 |
| w | 写权限 | 可以修改文件内容 | 可以在目录中创建，删除文件 |
| x | 执行权限 | 可以执行文件 | 可以进入目录 |

权限管理命令:chown

命令名称: chown
 命令英文原意: change file ownership
 命令所在路径: /bin/chown
 执行权限: 所有用户
 语法: chown[用户][文件或目录]
 功能描述: 改变文件或目录的所有者
 范例: \$ chown root hello.c (改变文件hello.c的所有者为root)

权限管理命令:chgrp

命令名称: chgrp
 命令英文原意: change file group ownership
 命令所在路径: /bin/chgrp
 执行权限: 所有用户
 语法: chgrp [用户组][文件或目录]
 功能描述: 改变文件或目录的所属组
 范例: \$ chgrp cchenji hello.c (改变文件hello.c的所属组为cchenji)

权限管理命令: umask

命令名称: umask
 命令英文原意: the user file-creation mask
 命令所在路径: Shell内置命令
 执行权限: 所有用户
 语法: umask[-S] (-S 以rwx形式显示新建文件缺省权限)

功能描述：显示、设置文件的缺省权限

范例：\$ umask -S

- 在Linux里缺省所创建的文件是不具备x的，但目录可以

文件搜索命令

文件搜索命令：find

命令名称：find

命令所在路径：/bin/find

执行权限：所有用户

语法：find [搜索范围] [匹配条件]

功能描述：文件搜索

\$ find /etc -name init (name 是精确查找必须所有字符都匹配才算找到 可以利用通配符 *init* 来查找含有init的文件)

在目录/etc中查找文件init

-iname 不区分大小写的搜索

init* 查找以init开头的文件

init??? 查找以init开头并且后面还有三个字符的文件(使用?来匹配单个字符)

\$ find / -size +204800 (这里的查找是以数据块为单位的查找 在linux里一个数据块的大小是512字节 也就是 0.5k)

在根目录下查找大于100MB(100*1024*2 是100M的数据块的大小)的文件

+n 大于 -n 小于 n等于

\$ find /home -user shenchao

在根目录下查找所有者为shenchao的文件

-group 根据所属组查找

\$ find /etc -cmin -5

在/etc下查找5分钟内被修改过属性的文件和目录

-amin 访问时间 access

-cmin 文件属性 change

-mmin 文件内容 modify

\$ find /etc -size +163840 -a size -204800

在/etc下查找大于80MB/小于100MB的文件

-a 两个条件同时满足

-o 两个条件满足任意一个即可

-type 根据文件类型查找

f文件 d目录 l软连接文件

-inum 根据i节点查找

\$ find /etc -name inittab -exec ls -l {} \;

在/etc下查找inittab文件并显示其详细信息

-exec/-ok 命令 {} \; 对搜索结果执行操作

文件搜索命令： locate

命令名称： locate

命令所在路径： /usr/bin/locate

执行权限： 所有用户

语法： locate 文件名

功能描述： 在文件资料库中查找文件(文件资料库中查找文件速度快,查找的原理相当于windows里的 everything软件)

范例： \$ locate inittab (locate -i 不区分大小写)

updatedb 更新文件资料库

文件搜索命令： which

命令名称： which

命令所在路径： /usr/bin/which

执行权限： 所有用户

语法： which命令

功能描述： 搜索命令所在目录及别名信息

范例： \$ which ls

文件搜索命令： whereis

命令名称： whereis

命令所在路径： /usr/bin/whereis

执行权限： 所有用户

语法： whereis [命令名称]

功能描述： 搜索命令所在目录及帮助文档路径

范例： \$ whereis ls

文件搜索命令： grep

命令名称： grep

命令所在路径： /bin/grep

执行权限： 所有用户

语法: grep -iv [指定字串] [文件]

功能描述: 在文件中搜寻字串匹配的行并输出

-i 不区分大小写

-v 排除指定字串(看代码 -v # 排除#所在的行, -v ^# 排除#所在的首行)

范例: \$ grep mysql /root/install.log

帮助命令

帮助命令: man(很重要的命令)

命令名称: man

命令英文原意: manual

命令所在路径: /usr/bin/man

执行权限: 所有用户

语法: man[命令或配置文件]

功能描述: 获得帮助信息

范例:

man ls

查看ls命令的帮助信息

man services(不要加文件的绝对路径, 这样显示的是文件内容, 只需要加文件名称就行)

查看配置文件services的帮助信息

linux里的帮助文档 (1代表命令的帮助),(5代表配置文件的帮助)

如果某个配置文件有一个命令和它重名(**如passws**)那么man会优先选择显示命令的帮助信息, 如果这时候想要获得配置文件的帮助信息则需要明确告诉man命令**man 5 passwd**

whatis: 如果只是想要看一下命令的作用可以使用whatis+命令 这样只是显示命令的作用(用man查看的NAME那一行)

apropos: 如果只是想要查看配置文件的作用那么可以使用apropos+命令

命令 --help: 这样直接可以查看命令的选项信息

info: 可以达到和man一样的效果

帮助命令: help

命令名称: help

命令所在路径: Shell内置命令

执行权限: 所有用户

语法: help命令

功能描述: 获得Shell内置命令的帮助信息

范例: \$help umask

查看umask命令的帮助信息

用户管理命令：useradd

命令名称：useradd

命令所在路径：/usr/sbin/useradd

执行权限：root

语法：useradd用户名

功能描述：添加新用户

范例：\$useradd ccchenji

用户管理命令：passwd

命令名称：passwd

命令所在路径：/usr/bin/passwd

执行权限：所有用户

语法：passwd用户名

功能描述：设置用户密码

范例：\$passwd ccchenji

普通用户只能用passwd更改自己的密码，root可以更改所有人的密码

用户管理命令：who

命令名称：who

命令所在路径：/usr/bin/who

执行权限：所有用户

语法：who

功能描述：查看登陆用户信息

范例：\$ who

用户管理命令：w

命令名称：w

命令所在路径：/usr/bin/w

执行权限：所有用户

语法：w

功能描述：查看登陆用户详细信息

范例：\$w

压缩解压命令

压缩解压命令：gzip(只能压缩文件)

命令名称： gzip
命令英文原意： GUNzip
命令所在路径： /bin/gzip
执行权限： 所有用户
语法： gzip[文件]
功能描述： 压缩文件
压缩后文件格式： .gz

解压缩命令： gunzip

命令名称： gunzip
命令英文原意： GUNunzip
命令所在路径： /bin/gunzip
执行权限： 所有用户
语法： gunzip[文件]
功能描述： 解压缩.gz的压缩文件
范例： \$ gunzip hello.gz

压缩解压命令： tar

命令名称： tar
命令所在路径： /bin/tar
执行权限： 所有用户
语法： tar 选项[-zcf][压缩后文件名][目录]
-c 打包
-v 显示详细信息
-f 指定文件名(命令连用时f要放到最后面)
-z 打包同时压缩
功能描述： 打包目录
压缩后文件格式： .tar.gz

tar命令解压缩语法：

-x 解包
-v 显示详细信息
-f 指定压缩文件
-z 解压缩
范例： \$ tar -zxvf hello.tar.gz

压缩解压命令： zip

命令名称：zip

命令所在路径：/usr/bin/zip

执行权限：所有用户

语法：

zip 选项[-r] [压缩后文件名] [文件或目录]

-r 压缩目录

功能描述：压缩文件或目录

压缩后文件格式：.zip

压缩解压命令：unzip

命令名称：unzip

命令所在路径：/usr/bin/unzip

执行权限：所有用户

语法：unzip [压缩文件]

功能描述：解压.zip的压缩文件

范例：\$ unzip hello.zip

压缩解压命令：bzip2

命令名称：bzip2

命令所在路径：/usr/bin/bzip2

执行权限：所有用户

语法：bzip2 选项 [-k] [文件]

-k 产生压缩文件后保留原文件

功能描述：压缩文件

压缩文件格式：.bz2

范例：

\$ bzip2 -k hello

\$ tar -cjf hello.tar.bz2 hello

-j 以bz2的格式压缩

解压缩命令：bunzip2

命令名称：bunzip2

命令所在路径：/usr/bin/bunzip2

执行权限：所有用户

语法：bunzip2 选项[-k] [压缩文件]

-k 解压缩后保留原文件

功能描述：解压缩

范例：

```
$ bunzip2 -k hello.bz2
```

```
$ tar -xjf hello.tar.bz2
```

网络命令

网络命令： write

指令名称： write

指令所在路径： /usr/bin/write

执行权限： 所有用户

语法： write <用户名>

功能描述： 给用户发信息， 以Ctrl+D保存结束

范例： \$ write ccchenji

网络命令： wall

指令名称： wall

命令英文原意： write all

指令所在路径： /usr/bin/wall

执行权限： 所有用户

语法： wall [message]

功能描述： 发广播信息

范例： \$ wall hello word!

网络命令： ping(主要看网络是否联通以及通过查看丢包率来确定网络状况)

命令名称： ping

命令所在路径： /bin/ping

执行权限： 所有用户

语法： ping 选项 IP地址

-c 指定发送次数

功能描述： 测试网络连通性

范例： \$ ping 192.168.1.156

网络命令： ifconfig

命令名称： ifconfig

命令英文原意： interface configure

命令所在路径： /sbin/ifconfig

执行权限： root

语法: ifconfig 网卡名称 IP地址

功能描述: 查看和设置网卡信息

范例: \$ ifconfig eth0 192.168.8.250

网络命令: mail

命令名称: mail

命令所在路径: /bin/mail

执行权限: 所有用户

语法: mail [用户名]

功能描述: 查看发送电子邮件

范例: \$ mail root

网络命令: last(很重要的日志查询命令)

命令名称: last

命令所在路径: /usr/bin/last

执行权限: 所有用户

语法: last

功能描述: 列出目前与过去登入系统的用户信息

范例: \$ last

网络命令: lastlog

命令名称: lastlog

命令所在路径: /usr/bin/lastlog

执行权限: 所有用户

语法: lastlog

功能描述: 检查某特定用户上次登陆的时间

范例:

\$ lastlog

\$ lastlog -u 502(这里的502是用户的ID)

网络命令: traceroute

命令名称: traceroute

命令所在路径: /bin/traceroute

执行权限: 所有用户

语法: traceroute

功能描述: 显示数据包到主机间的路径

范例: \$ traceroute www.4399.com

网络命令：netstat

命令名称：netstat

命令所在路径：/bin/netstat

执行权限：所有用户

语法：netstat [选项]

功能描述：显示网络相关信息

选项：

-t :TCP 协议

-u :UDP 协议

-l :监听

-r :路由

-n :显示IP地址和端口号

范例：

\$ netstat -tlun 查看本机监听的端口

\$ netstat -an 查看本机所有的网络联接

\$ netstat -rn 查看本机路由表(看网关)

网络命令：setup(redhat 专有命令,用setup配置的ip地址是永久生效的,ifconfig配置的是临时生效的)

命令名称：setup

命令所在路径：/usr/bin/setup

执行权限：root

语法：setup

功能描述：配置网络

范例：\$setup

挂载命令：mount

命令位置：/bin/mount

执行权限：所有用户

命令语法：mount [-t 文件系统] 设备文件名 挂载点

范例：\$ mount -t iso9660 /dev/sr0 /mnt/cdrom

要卸载则使用umount卸载时不可以直接在挂载目录卸载，必须在别的目录卸载

关机重启命令(推荐使用)

1. shutdown命令

\$ shutdown [选项] 时间

选项: -c:取消前一个关机命令 -h:关机 -r:重启

2. 其他关机命令

halt poweroff(相当于直接断电没事别再服务器使用) init 0

3. 其他重启命令

reboot init 6

4. 系统运行级别

0 关机;1 单用户(类似Win的安全模式);2 不完全多用户,不含NFS服务;3 完全多用户;4 未分配;5 图形界面;6 重启,**其中1, 2, 3均没有图形界面**

- \$ cat/etc/inittab(centos6,centos7是其他的方式)

修改系统默认运行级别 id:3:initdefault:

- \$ runlevel 查询系统运行级别

5. 退出登陆命令

\$ logout

软件包管理

软件包管理简介

软件包分类

- 源码包(可以看到源代码但是安装过程比二进制包慢, 因为要编译才能安装)
 - 脚本安装包(类似win里的安装方式, 其实就是给源码包开发一个安装界面)
- 二进制包(RPM包, 系统默认包)(linux发行版本的主要不同就是二进制包的格式不同, 其他基本相同)

源码包

源码包的优点:

- 开源, 如果有足够的能力, 可以修改源代码
- 可以自由选择所需的功能
- 软件是编译安装, 所以更加适合自己的系统, 更加稳定也效率更高。
- 卸载方便

源码包的缺点:

- 安装过程步骤较多, 尤其安装较大的软件集合时(如LAMP环境搭建), 容易出现拼写错误
- 编译过程时间较长, 安装比二进制安装时间长
- 因为是编译安装, 安装过程中一旦报错新手很难解决

RPM包

二进制包的优点：

- 包管理系统简单，只通过几个命令就可以实现包的安装、升级、查询和卸载
- 安装速度比源码包安装快的多

二进制包的缺点：

- 经过编译，不再可以看到源代码
- 功能选择不如源码包灵活
- 依赖性

软件包管理-rpm命令管理-包命令与与依赖性

RPM包命名原则

httpd-2.2.15-15.e16.centos.1.i686.rpm

- httpd 软件包名(加上后面的所有的是包全名，一定要区分**包名**和**包全名**)
- 2.2.15 软件版本
- 15 软件发布的次数
- e16.centos 适合的Linux平台
- i686 适合的硬件平台(noarch可以在各个不同的CPU上运行)
- rpm rpm包扩展名

RPM包依赖性

- 树形依赖：a->b->c(安装时先安装c再b再a,卸载时先a再b再c)
- 环形依赖：a->b->c->a(安装时一起安装)
- 模块依赖：模块依赖查询网站：www.rpmfind.net (有的时候安装依赖的不是rpm整个包而是包里的某个文件，那么我们需要安装对应的rpm包,这样里面的文件肯定也安装了，但是我们不知道这个文件在哪个包里，所以需要用这个网站查询)

软件包管理-rpm命令管理-安装升级与卸载

包全名与包名

- 包全名：操作的包是没有安装的软件包时，使用包全名。而且要注意路径
- 包名：操作已经安装的软件包时，使用包名。是搜索/var/lib/rpm中的数据库

RPM安装

rpm-ivh 包全名

选项:

- i (install) 安装
- v (verbose) 显示详细信息
- h (hash) 显示进度
- nodeps 不检测依赖性

RPM包升级

rpm -Uvh 包全名

选项:

- U (upgrade) 升级

卸载

rpm -e 包名

选项:

- e (erase) 卸载
- nodeps 不检查依赖性

软件包管理-rpm命令管理-查询

查询是否安装

\$ rpm -q 包名 (查询包是否安装)

选项: -q 查询 (query)

\$ rpm -qa (查询所有已经安装的RPM包)

选项: -a 所有(all)

查询软件包详细信息

\$ rpm -qi 包名

选项:

- i 查询软件信息(information)
- p 查询未安装包信息(package)

查询包中文件安装位置

\$ rpm -ql 包名

选项:

-l 列表 (list)

-p 查询未安装包信息(package)

查询系统文件属于哪个RPM包

\$ rpm -qf 系统文件名

选项： -f 查询系统文件属于哪个软件包(file)

查询软件包的依赖性

\$ rpm -qR 包名

选项：

-R 查询软件包的依赖性 (requires)

-p 查询未安装包信息(package)

软件包管理-rpm命令管理-校验和文件提取

RPM包校验

\$ rpm -V 已安装的包名

选项： -V 校验指定RPM包中的文件(verify)

```
[root@localhost ~]# rpm -V linuxqq
S.5....T.    /usr/share/applications/qq.desktop
```

验证内容中的8个信息的具体内容如下：

- S 文件大小是否改变
- M 文件的类型或文件的权限(rwx)是否被改变
- 5 文件MD5校验和是否改变(可以看成文件内容是否改变)
- D 设备的中，从代码是否改变
- L 文件路径是否改变
- U 文件的属主(所有者)是否改变
- G 文件的属组是否改变
- T 文件的修改时间是否改变

文件类型

- c 配置文件 (config file)
- d 普通文档 (documentation)
- g "鬼"文件(ghost file),很少见，就是该文件不应该被这个RPM包包含
- l 授权文件 (license file)
- r 描述文件 (read me)

RPM包中文件提取

\$ rpm2cpio 包全名 |cpio -idv .文件绝对路径

rpm2cpio (将rpm包转换为cpio格式的命令)

cpio (是一个标准工具，它用于创建软件档案文件和从档案文件中提取文件)

rpm包管理-yum在线管理-IP地址配置和网络yum源

网yum源

\$ vim /etc/yum.repos.d/CentOS-Base.repo

```
1 # CentOS-Base.repo
2 #
3 # The mirror system uses the connecting IP address of the client and the
4 # update status of each mirror to pick mirrors that are updated to and
5 # geographically close to the client. You should use this for CentOS updates
6 # unless you are manually picking other mirrors.
7 #
8 # If the mirrorlist= does not work for you, as a fall back you can try the
9 # remarked out baseurl= line instead.
10 #
11 #
12
13 [base]
14 name=CentOS-$releasever - Base
15 mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os&infra=$infra
16 #baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
17 gpgcheck=1
18 gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
19
20 #released updates
21 [updates]
22 name=CentOS-$releasever - Updates
23 mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=updates&infra=$infra
24 #baseurl=http://mirror.centos.org/centos/$releasever/updates/$basearch/
25 gpgcheck=1
26 gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
27
28 #additional packages that may be useful
29 [extras]
30 name=CentOS-$releasever - Extras
31 mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=extras&infra=$infra
32 #baseurl=http://mirror.centos.org/centos/$releasever/extras/$basearch/
33 gpgcheck=1
34 gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
35
36 #additional packages that extend functionality of existing packages
37 [centosplus]
38 name=CentOS-$releasever - Plus
39 mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=centosplus&infra=$infra
40 #baseurl=http://mirror.centos.org/centos/$releasever/centosplus/$basearch/
41 gpgcheck=1
42 enabled=0
43 gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```

- **[base]** 容器名称，一定要放在[]中

- **name** 容器说明，可以自己随便写

- **mirrorlist** 镜像站点，这个可以注释掉
- **baseurl** 我们的yum源服务器的地址。默认是Centos官方的yum源服务器，是可以使用的，如果你觉得慢可以改成你喜欢的yum源地址
- **enabled** 此容器是否生效，如果不写或写成enable=1都是生效，写成enable=0就是不生效
- **gpgcheck** 如果是1是指RPM的数字证书生效，如果是0则不生效
- **gpkey** 数字证书的公钥文件保存位置。不用修改

rpm包管理-yum在线管理-yum命令

常用yum命令

1. 查询
 - \$ yum list (查询所有可用软件包列表)
 - \$ yum search 关键字 (搜索服务器上所有和关键字相关的包)
2. 安装
 - \$ yum -y install 包名
 - -y 自动回答yes
 - install(安装)
3. 升级(核弹命令 小心后面一定要加包名 不然会更新所有的软件包包括linux内核,这时候需要做配置才能再次联接服务器)
 - yum -y update 包名
 - update 升级
 - -y 自动回答yes
4. 卸载(尽量别卸载软件， yum会自动卸载要卸载软件的依赖包，这些包可能被系统所用，写在会导致系统崩溃)
 - \$ yum -y remove 包名
 - remove 卸载
 - -y 自动回答yes

YUM软件组管理命令

\$ yum grouplist (列出所有可用的软件组列表)
\$ yum groupinstall 软件组名 (安装指定软件组，组名可以由grouplist查询出来)
\$ yum groupremove 软件组名 (卸载指定软件组)

rpm包管理-yum在线管理-光盘yum源

1. 挂载光盘
2. 让网络yum源文件失效
 - cd /etc/yum.repos.d/

- mv CentOS-Base.repo CentOS-Base.repo.bak
- mv CentOS-Debuginfo.repo CentOS-Debuginfo.repo.bak
- CentOS-Vault.repo CentOS-Vault.repo.bak

3. 修改光盘yum源文件

- vim CentOS-Media.repo(打开配置文件)
- baseurl=file:///mnt/cdrom(修改为光盘挂载点地址)
- enable =1;(让yum源配置文件生效)

重要:

- linux中的配置文件有严格的格式要求，有的甚至不允许一行后面有空格，或者前面有缩进

rpm包管理-源码包管理-源码包与RPM包的区别

区别

- 安装之前的区别：概念上的区别
- 安装之后的区别：安装位置不同

RPM包安装位置

- 安装在默认位置中

| RPM包默认安装路径 | 说明 |
|-----------------|---------------|
| /etc/ | 配置文件安装目录 |
| /usr/bin/ | 可执行的命令安装目录 |
| /usr/lib/ | 程序所使用的函数库保存位置 |
| /usr/share/doc/ | 基本的软件使用手册保存位置 |
| /usr/share/man/ | 帮助文件保存位置 |

源码包安装位置

- 安装在指定位置当中，一般是 /usr/local/软件名/

安装位置不同带来的影响

- RPM包安装的服务可以使用系统服务管理命令(service)来管理，例如RPM包安装的apache的启动方法是：
 - /etc/rc.d/init.d/httpd start

- service httpd start
- 而源码包安装的服务则不能被服务管理命令管理，因为没有安装到默认路径中。所以只能用绝对路径进行服务的管理。

rpm包管理-源码包管理-源码包安装过程

安装准备

- 安装c语言编译器
- 下载源码包

安装注意事项

- 源代码保存位置： /usr/local/src/
- 软件安装位置: /usr/local/
- 如何确定安装过程报错：
 - 安装过程停止
 - 并出现error,warning或no的提示

源码包安装过程

- 下载源码包
- 解压下载的源码包
- 进入解压缩目录
- 一个源码包安装首先看INSTALL这里面有安装说明，然后看README这个是使用说明

源码包的卸载

- 不需要卸载命令，直接删除安装目录即可，不会遗留任何垃圾文件。

用户和用户组管理

用户配置文件-用户信息文件

用户管理简介

- 越是对服务器安全性要求高的服务器，越需要建立合理的用户权限等级制度和服务器操作规范。
- 在Linux中主要是通过用户配置文件来查看和修改用户信息。

/etc/passwd

```

1 root:x:0:0:root:/root:/bin/bash
2 bin:x:1:1:bin:/bin:/sbin/nologin
3 daemon:x:2:2:daemon:/sbin:/sbin/nologin
4 adm:x:3:4:adm:/var/adm:/sbin/nologin
5 lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
6 sync:x:5:0:sync:/sbin:/bin/sync
7 shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
8 halt:x:7:0:halt:/sbin:/sbin/halt
9 mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
10 operator:x:11:0:operator:/root:/sbin/nologin
11 games:x:12:100:games:/usr/games:/sbin/nologin
12 ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
13 nobody:x:99:99:Nobody::/sbin/nologin
14 systemd-network:x:192:192:systemd Network Management:/sbin/nologin
15 dbus:x:81:81:System message bus:/sbin/nologin
16 polkitd:x:999:998:User for polkitd:/sbin/nologin
17 sssd:x:998:996:User for sssd:/sbin/nologin
18 libstoragemgmt:x:997:994:daemon account for libstoragemgmt:/var/run/lsm:/sbin/nologin
19 rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
20 colord:x:996:993:User for colord:/var/lib/colord:/sbin/nologin
21 gluster:x:995:992:GlusterFS daemons:/var/run/gluster:/sbin/nologin
22 saslauth:x:994:76:Saslauthd user:/run/saslauthd:/sbin/nologin
23 abrt:x:173:173::/etc/abrt:/sbin/nologin
24 setroubleshoot:x:993:990::/var/lib/setroubleshoot:/sbin/nologin
25 rtkit:x:172:172:RealtimeKit:/proc:/sbin/nologin
26 pulse:x:171:171:PulseAudio System Daemon:/var/run/pulse:/sbin/nologin
27 chrony:x:992:987:/var/lib/chrony:/sbin/nologin
28 rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
29 nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
30 unbound:x:991:986:Unbound DNS resolver:/etc/unbound:/sbin/nologin
31 tss:x:59:59:Account used by the trousers package to sandbox the tcscd daemon:/dev/null:/sbin/nologin
32 usbmuxd:x:113:113:usbmuxd user:/sbin/nologin
33 geoclue:x:990:984:User for geoclue:/var/lib/geoclue:/sbin/nologin
34 radvd:x:75:75:radvd user:/sbin/nologin
35 qemu:x:107:107:qemu user:/sbin/nologin
36 ntp:x:38:38::/etc/ntp:/sbin/nologin
37 gdm:x:42:42::/var/lib/gdm:/sbin/nologin
38 gnome-initial-setup:x:989:983::/run/gnome-initial-setup:/sbin/nologin
39 sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
40 avahi:x:70:70:Avahi mDNS/DNS-SD Stack:/var/run/avahi-daemon:/sbin/nologin
41 postfix:x:89:89::/var/spool/postfix:/sbin/nologin
42 tcpdump:x:72:72::/sbin/nologin
43 ccchenji:x:1000:1000:ccchenji:/home/ccchenji:/bin/bash

```

- 第一字段：用户名
- 第2字段：密码标志
- 第3字段：UID(用户ID)(如果想要将一个用户变为超级用户可以将它的UID变为0-->得root才能操作)
 - 0: 超级用户
 - 1-499: 系统用户(伪用户)
 - 500-65535: 普通用户
- 第4字段：GID(用户初始组ID)
- 第5字段：用户说明
- 第6字段：家目录(宿主目录)
 - 普通用户: /home/用户名/
 - 超级用户: /root/
- 第7字段：登陆之后的Shell

初始组和附加组

初始组:就是指用户一登陆就立刻拥有这个用户组的相关权限，每个用户的初始组只能有一个，一般就是和这个用户的用户名相同的组名作为这个用户的初始组。

附加组:指用户可以加入多个其他的用户组，并拥有这些组的权限，附加组可以有多个。

Shell是什么

- Shell就是Linux的命令解释器
- 在/etc/passwd当中，除了标准Shell是/bin/bash之外，还可以写如/sbin/nologin

用户配置文件-影子文件

/etc/shadow

```
1 root:$6$z7rb80vrGCrSm20$nE/DQ0HJ8g2CmouF1YLy.jkqQNlUl6q.IJXQ4kq9P9FVPG/Si2DvSXGYLVimui  
bbPgvUqpfbX0B8rBwmqCDyv0::0:99999:7:::  
2 bin:*:17632:0:99999:7:::  
3 daemon:*:17632:0:99999:7:::  
4 adm:*:17632:0:99999:7:::  
5 lp:*:17632:0:99999:7:::  
6 sync:*:17632:0:99999:7:::  
7 shutdown:*:17632:0:99999:7:::  
8 halt:*:17632:0:99999:7:::  
9 mail:*:17632:0:99999:7:::  
10 operator:*:17632:0:99999:7:::  
11 games:*:17632:0:99999:7:::  
12 ftp:*:17632:0:99999:7:::  
13 nobody:*:17632:0:99999:7:::  
14 systemd-network:!!:18317:::::::  
15 dbus:!!:18317:::::::  
16 polkitd:!!:18317:::::::  
17 sssd:!!:18317:::::::  
18 libstoragemgmt:!!:18317:::::::  
19 rpc:!!:18317:0:99999:7:::  
20 colord:!!:18317:::::::  
21 gluster:!!:18317:::::::  
22 saslauth:!!:18317:::::::  
23 abrt:!!:18317:::::::  
24 setroubleshoot:!!:18317:::::::  
25 rtkit:!!:18317:::::::  
26 pulse:!!:18317:::::::  
27 chrony:!!:18317:::::::  
28 rpcuser:!!:18317:::::::  
"/etc/shadow" [只读] 45L, 1293C
```

1,1

顶端

- 第1字段：用户名
- 第2字段：加密密码
 - 加密算法升级为SHA512散列加密算法
 - 如果密码位是"!!" 或 "*"代表没有密码，不能登陆
- 第3字段：密码最后一次修改日期

- 使用1970年1月1日作为标准时间，每过一天时间戳加1
- 第4字段：两次密码的修改间隔时间(和第3字段相比)
- 第5字段：密码有效期(和第3字段相比)
- 第6字段：密码修改到期前的警告天数(和第5字段相比)
- 第7字段：密码过期后的宽限天数(和第5字段相比)
 - 0:代表密码过期后立即失效
 - -1:则代表密码永远不会失效
- 第8字段：账号失效时间
 - 要用时间戳表示
- 第9字段：保留

时间戳换算

- 把时间戳换算为日期
 - date -d "1970-01-01 16066 days"
- 把日期换算为时间戳
 - echo ((date --date="2014/01/06" +%s)/86400)

用户配置文件-组信息文件

组信息文件/etc/group

```
1 root:x:0:
2 bin:x:1:
3 daemon:x:2:
4 sys:x:3:
5 adm:x:4:
6 tty:x:5:
7 disk:x:6:
8 lp:x:7:
9 mem:x:8:
10 kmem:x:9:
11 wheel:x:10:ccchenji
12 cdrom:x:11:
13 mail:x:12:postfix
14 man:x:15:
15 dialout:x:18:
16 floppy:x:19:
17 games:x:20:
18 tape:x:33:
19 video:x:39:
20 ftp:x:50:
21 lock:x:54:
22 audio:x:63:
23 nobody:x:99:
24 users:x:100:
25 utmp:x:22:
26 utempter:x:35:
27 input:x:999:
28 systemd-journal:x:190:
29 systemd-network:x:192:
"/etc/group" 73L, 1004C
```

1,1

顶端

第1字段：组名

第2字段：组密码标志

第3字段：GID

第4字段：组中附加用户

组密码文件 /etc/gshadow

第1字段：组名

第2字段：组密码

第3字段：组管理员用户名

第4字段：组中附加用户

用户管理相关文件

用户的家目录

普通用户：/home/用户名/,所有者和所属组都是此用户，权限是700

超级用户：/root/,所有者和所属组都是root用户，权限是550

用户的邮箱

/var/spool/mail/用户名/

用户模板目录

/etc/skel/ (创建新用户时会自动将该目录下的文件拷贝到用户的家目录下)

用户管理命令-useradd命令

useradd命令格式

```
$ useradd [选项] 用户名
-u UID: 手工指定用户的UID号
-d 家目录: 手工指定用户的家目录
-c 用户说明: 手工指定用户的说明
-g 组名: 手工指定用户的初始组
-G 组名: 指定用户的附加组
-s shell: 手工指定用户的登陆shell。默认是/bin/bash
```

用户默认值文件

/etc/default/useradd

- GROUP =100 #用户默认组
- HOME =/home #用户家目录
- INACTIVE =-1 #密码过期宽限天数(shadow文件7字段)
- EXPIRE = #密码失效时间(8)
- SHELL =/bin/bash #默认shell
- SKEL =/etc/skel #模板目录
- CREATE_MAIL_SPOOL=yes #是否建立邮箱

/etc/login.defs

- PASS_MAX_DAYS 99999 #密码有效期(5)
- PASS_MIN_DAYS 0 #密码修改间隔(4)
- PASS_MIN_LEN 5 #密码最小5位(PAM)(现在该这里不生效)
- PASS_WARN_AGE 7 # 密码到期警告(6)
- UID_MIN 500 #最小和最大UID范围
- GID_MAX 60000
- ENCRYPT_METHOD SHA512 #加密模式

用户管理命令-passwd命令

passwd命令

\$ passwd [选项] 用户名

-S 查询用户密码的密码状态。仅root用户可用

-l 暂时锁定用户。仅root用户可用

-u 解锁用户。仅root用户可用

--stdin 可以通过管道符输出的数据作为用户的密码(echo "123" | passwd --stdin lamp)

passwd 不加用户名可以直接修改当前用户密码

linux管道符"|"的作用

命令A|命令B，即命令1的正确输出作为命令B的操作对象

用户管理命令-usermod和change

修改用户信息usermod

usermod [选项] 用户名

-u UID：修改用户的UID号

-c 用户说明：修改用户的说明信息

-G 组名：修改用户的附加组 (-a -G 将用户添加到新的附加组而不删除原来的组)

-L：临时锁定用户 (Lock)

-U：解锁用户锁定(unlock)

修改用户密码状态chage

chage [选项] 用户名

-l：列出用户的详细密码状态

-d 日期：修改密码最后一次更改日期(shadow3字段)

-m 天数：两次密码修改间隔(4字段)

-M 天数：密码有效期(5字段)

-W 天数：密码过期前警告天数(6字段)

-I 天数：密码过后宽限天数(7字段)

-E 日期：账号失效时间(8字段)

用户管理命令-userdel和su

删除用户userdel

\$ userdel [-r] 用户名

选项： -r 删除用户的同时删除用户家目录

查看用户 ID

\$ id 用户名

切换用户身份su

\$ su[选项] 用户名

-: 选项只使用“-”代表连带用户的环境变量一起切换

-c命令：仅执行一次命令，而不切换用户身份

用户组管理命令

添加用户组

\$ groupadd [选项] 组名

-g GID: 指定组ID

修改用户组

\$ groupmod [选项] 组名

-g GID: 修改组ID

-n 新组名: 修改组名

eg. groupmod -n jixuan ccchenji

删除用户组

\$ groupdel 组名

把用户添加入组或从组中删除(操作的是附加组)

\$ gpasswd 选项 组名

-a 用户名: 把用户加入组

-d 用户名: 把用户从组中删除

权限管理

ACL权限-简介与开启

查看分区ACL权限是否开启

```
# dumpe2fs -h /dev/sda3
```

dumpe2fs命令是查询指定分区详细文件系统命令信息的命令

选项：-h(仅显示超级快中信息，而不显示磁盘块组额详细信息)

临时开启分区ACL权限

```
# mount -o remoun,acl /
```

重新挂载根分区，并挂载加入acl权限

永久开启分区ACL权限

```
# vi/etc/fstab
```

```
UUID=c2ca6f57-b15-43ea-bca0-f239083d8bd2 / ext4 defaults,acl 1 1
```

加入acl

```
1 #
2 #
3 # /etc/fstab
4 # Created by anaconda on Tue Feb 25 23:56:36 2020
5 #
6 # Accessible filesystems, by reference, are maintained under '/dev/disk'
7 # See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
8 #
9 UUID=feab886c-3bfa-4247-8c98-648176fb2917 /          xfs    defaults
   0 0
10 UUID=672bb0d3-b46a-4183-9038-fc4b0fb5af2f /boot        xfs    defaults
   0 0
11 UUID=946a4d39-63a5-45e9-972d-10eb46cad65a /home        xfs    defaults
   0 0
12 UUID=b333a737-9901-4665-961f-927ecfd25124 swap        swap    defaults
   0 0
```

~
~
~
~
~
~
~
~

1,0-1

全部

现在Linux系统默认选项里就有acl权限，所以一般不用修改

```
mount -o remount / (重新挂载根分区)
```

ACL权限-查看与设定

查看ACL命令

```
# getfacl 文件名 (查看ACL权限)
```

设定ACL权限的命令

```
# setfacl 选项 文件名
```

-m 设定ACL权限

-x 删除指定的ACL权限

-b 删除所有的ACL权限

-d 设定默认ACL权限

-k 删除默认ACL权限

-R 递归设定ACL权限

给用户st赋予r-x权限，使用"u:用户名:权限(rx)"格式 setfacl -m u:st:rx /project/

注意:有ACL权限的文件后面会有一个加号

```
[root@localhost 桌面]# ls -dl hello
drwxr-xr-x+ 2 root root 72 3月 26 05:14 hello
```

ACL权限-最大有效权限与删除ACL权限

最大有效权限mask

mask是用来指定最大有效权限的。如果我给用户赋予了ACL权限，是需要和mask的权限“相与”才能得到用户的真正权限。

```
# setfacl -m m:rx 文件名 (设定mask权限为r-x。使用"m:权限"格式)
```

注意:当设置了ACL权限的时候ls -l 命令显示的组的权限是mask的权限而不是组的权限

删除ACL权限

```
# setfacl -x u:用户名 文件名(删除指定用户的ACL权限)
```

```
# setfacl -x g:组名 文件名 (删除指定用户组的ACL权限)
```

ACL权限-默认ACL权限和递归ACL权限

递归ACL权限

- 递归是父目录在设定ACL权限时，所有的子文件和子目录也会拥有相同的ACL权限。
- setfacl -m u:用户名:权限 -R 文件名

注意: -R一定要放在后面，不然报错

默认ACL权限

- 默认ACL权限的作用是如果给父目录设定了默认ACL权限，那么父目录中所有新建的子文件都会继承父目录的ACL权限。
- setfacl -m d:u:用户名:权限 文件名

文件特殊权限-SetUID

SetUID的功能

- 只有可以执行的二进制程序才能设定SUID权限
- 命令执行者要对该程序拥有x(执行)权限
- 命令执行者在执行该程序时获得该程序文件属主的身份(在执行程序的过程中灵魂附体为文件的属主)
- SetUID权限只在该程序执行过程中有效，也就是说身份改变只在程序执行过程中有效
- passwd命令拥有SetUID权限，所以普通用户可以修改自己的密码(图片里的rws s代表SetUID权限大S代表报错说明权限不可用)

```
[root@localhost ~]# ll /usr/bin/passwd
-rwsr-xr-x. 1 root root 27832 6月 10 2014 /usr/bin/passwd
```

- cat命令没有SetUID权限，所以普通用户不能查看/etc/shadow文件内容

设定SetUID的方法

4代表SUID

- chmod 4755 文件名
- chmod u+s 文件名

取消SetUID的方法

- chmod 755 文件名
- chmod u-s 文件名

危险的SetUID

- 关键目录应该严格控制写权限。比如"/"、"usr" 等
- 用户的密码设置要严格遵守密码三原则
- 对系统中默认应该具有SetUID权限的文件作以列表，定时检查有没有这之外的文件被设置了SetUID权限。

文件特殊权限-SetGID

SetGID对文件的作用

- 只有可执行的二进制程序才能设置SGID权限
- 命令执行者要对该程序拥有x(执行)权限
- 命令执行在执行程序的时候，组身份升级为该程序文件的属组
- SetGID权限同样只在该程序执行过程中有效，也就是说组身份改变只在程序执行过程中有效

SetGID对目录的作用

- 普通用户必须对此目录拥有r和x权限，才能进入此目录
- 普通用户在此目录中的有效组会变成此目录的属组
- 若普通用户对此目录拥有w权限时，新建的文件的默认属组是这个目录的属组

设定SetGID

- 2代表SGID
 - chmod 2755 文件名
 - chmod g+s 文件名

文件特殊权限-Sticky BIT

SBIT黏着位作用

- 黏着位目前只对目录有效
- 普通用户对该目录拥有w和x权限，即普通用户可以在此目录拥有写权限
- 如果没有黏着位，因为普通用户拥有w权限，所以可以删除此目录下所有文件，包括其他用户建立的文件。一旦赋予了黏着位，除了root可以删除所有文件，普通用户就算拥有w权限，也只能删除自己建立的文件，但是不能删除其它用户建立的文件

设置与取消黏着位

- 设置黏着位
 - chmod 1755 目录名
 - chmod o+t 目录名
- 取消黏着位
 - chmod 777 目录名
 - chmod o-t 目录名

文件系统属性chattr权限

chattr命令格式(对root用户生效)

```
# chattr [+-=][选项] 文件或目录名
```

+:增加权限

-:删除权限

=:等于某权限

- 选项:

- i:如果对文件设置i属性，那么不允许对文件进行删除，改名，也不能添加和修改数据；如果对目录设置i属性，那么只能修改目录下文件的数据，但不允许建立和删除文件。
- a:如果对文件设置a属性，那么只能在文件中增加数据，但是不能删除也不能修改数据；如果对目录设置a属性，那么只允许在目录中建立和修改文件，但是不允许删除。

查看文件系统属性

```
# lsattr 选项 文件名
```

- 选项:

- -a 显示所有文件和目录
- -d 若目标是目录，仅列出目录本身的属性，而不是子文件的

系统命令sudo权限

sudo 权限

- root把本来只能超级用户执行的命令赋予普通用户执行。
- sudo的操作对象是系统命令

sudo使用

```
# visudo (实际修改的是/etc/sudoers文件)
```

```
root ALL=(ALL) ALL(用户名 被管理主机的地址= (可用的身份) 授权命令(绝对路径))
```

```
# %wheel ALL=(ALL) ALL (%组名 被管理主机的地址= (可使用的身份) 授权命令(绝对路径))
```

注意： 千万不要给普通用户vi的权限，否则普通用户可以直接用vim 操作他没有权限的文件如 shadow

普通用户执行sudo赋予的命令

```
$ sudo -l (查看可用的sudo命令)
```

```
$ sudo/sbin/shutdown -r now (普通用户执行sudo赋予的命令)
```

文件系统管理

文件系统常用命令

文件系统查看命令df

- # df [选项] [挂载点]
- 选项：
 - -a 显示所有的文件系统信息，包括特殊文件系统，如/proc、/sysfs
 - -h 使用习惯单位显示容量，如KB，MB或GB等
 - -T 显示文件系统类型
 - -m 以MB位单位显示容量
 - -k 以KB为单位显示容量。默认就是以KB为单位

统计目录或文件大小du

- # du [选项] [目录或文件名]
- 选项：
 - -a 显示每个子文件的磁盘占用量。默认只统计子目录的磁盘占用量
 - -h 使用习惯单位显示磁盘占用量
 - -s 统计总占用量，而不列出子目录或子文件的占用量

文件系统修复命令fsck(系统会自动执行)

- # fsck [选项] 分区设备文件名
- 选项：
 - -a：不用显示用户提示，自动修复文件系统
 - -y：自动修复。和-a作用一致，不过有些文件系统只支持-y

显示磁盘状态命令 dumpe2fs

- # dumpe2fs 分区设备文件名 (只对ext4格式有效)

挂载命令mount

查询与自动挂载

- # mount [-l] (查询系统中已经挂载的设备，——l会显示卷标名称)
- # mount -a (依据配置文件/etc/fstab的内容，自动挂载)

挂载格式(括号代表可选项，如果什么都没有就用默认)

- # mount [-t 文件系统] [-L 卷标名] [-o 特殊选项] 设备文件名 挂载点
- 选项：

- -t 文件系统：加入文件系统类型来指定挂载的类型，可以ext3、ext4、iso9660等文件系统
- -L 卷标名：挂载指定卷标的分区，而不是安装设备文件名挂载
- -o 特殊选项：可以指定挂载的额外选项 (mount -o remount,exec /home)

| 参数 | 说明 |
|----------------------|---|
| atime/noatime | 更新访问时间/不更新访问时间。访问分区文件时，是否更新文件的访问时间，默认为更新 |
| async/sync | 异步/同步，默认为异步 |
| auto/noauto | 自动/手动，mount -a命令执行时，是否会自动安装/etc/fstab文件内容挂载，默认为自动 |
| defaults | 定义默认值，相当于rw,suid,dev,exec,auto,nouser,async这七个选项 |
| exec/noexec | 执行/不执行，设定是否允许在文件系统中执行可执行文件，默认是exec允许 |
| remount | 重新挂载已经挂载的文件系统，一般用于指定修改特殊权限 |
| rw/ro | 读写/只读，文件系统挂载时，是否具有读写权限，默认是rw |
| suid/nosuid | 具有/不具有SUID权限，设定文件系统是否具有SUID和SGID的权限，默认是具有 |
| user/nouser | 允许/不允许普通用户挂载，设定文件系统是否允许普通用户挂载，默认是不允许，只有root可以挂载分区 |
| usrquota | 写入代表文件系统支持用户磁盘配额，默认不支持 |
| grpquota | 写入代表文件系统支持组磁盘配额，默认不支持 |

挂载光盘与u盘

挂载光盘

- # mkdir /mnt/cdrom/ (建立挂载点)
- # mount -t iso9660 /dev/cdrom/ /mnt/cdrom (挂载光盘)
- # mount /dev/sr0 /mnt/cdrom

卸载命令

- # umount 设备文件名或挂载点
- # umount /mnt/cdrom

挂载u盘

- # fdisk -l(查看u盘设备文件名)
- # mount -t vfat /dev/sdb1 /mnt/usb

*注意：Linux默认是不支持NTFS文件系统的

支持ntfs文件系统(移动硬盘)

fdisk分区-分区过程

添加新硬盘

查看新硬盘

```
# fdisk -l
```

使用fdisk命令分区

```
# fdisk /dev/sdb
```

| fdisk交互指令说明 | |
|-------------|--|
| 命令 | 说明 |
| a | 设置可引导标记 |
| b | 编辑bsd磁盘标签 |
| c | 设置DOS操作系统兼容标记 |
| d * | 删除一个分区 |
| l | 显示已知的文件系统类型。82为Linux swap分区，83为Linux分区 |
| m | 显示帮助菜单 |
| n | 新建分区 |
| o | 建立空白DOS分区表 |
| p | 显示分区列表 |
| q | 不保存退出 |
| s | 新建空白SUN磁盘标签 |
| t | 改变一个分区的系统ID |
| u | 改变显示记录单位 |
| v | 验证分区表 |
| w | 保存退出 |
| x | 附加功能（仅专家） |

重新读取分区表信息

```
# partprobe
```

格式化分区

```
# mkfs -t ext4 /dev/sdb1
```

建立挂载点并挂载

```
# mkdir /disk1  
# mount /dev/sdb1/ disk1/  
k
```

fdisk分区-分区自动挂载与fstab文件修复

/etc/fstab文件

查看UUID:blkid /dev/sdb1

- ◆ 第一字段: 分区设备文件名或UUID (硬盘通用唯一识别码)
- ◆ 第二字段: 挂载点
- ◆ 第三字段: 文件系统名称
- ◆ 第四字段: 挂载参数
- ◆ 第五字段: 指定分区是否被dump备份, 0代表不备份, 1代表每天备份, 2代表不定期备份
- ◆ 第六字段: 指定分区是否被fsck检测, 0代表不检测, 其他数字代表检测的优先级, 那么当然1的优先级比2高

分区自动挂载

```
# vi /etc/fstab  
  
/dev/sdb1 /disk1 ext4 defaults 1 2
```

写完执行 mount -a 命令 看有没有写错(一定不能写错不然开机系统会崩溃)

/etc/fstab 文件修复(只用在fstab文件写错时修复)

```
# mount -o remount,rw / (当重启出错时, 可以进行修复, 进入fstab文件, 但如果出错, /文件是以只读挂载的, 无法修改fstab文件, 此时要重新挂载/然后修改fstab文件)
```

Shell基础

概述

Shell是什么

- Shell是一个命令行解释器, 它为用户提供了一个向Linux内核发送请求以便运行程序的界面系统级程序, 用户可以用Shell来启动, 挂起, 停止甚至编写一些程序。
- Shell还是一个功能相当强大的编程语言, 易编写, 易调试, 灵活性较强。Shell是解释执行的脚本语言, 在Shell中可以直接调用Linux系统命令。

Shell的分类

- **Bourne Shell**:从1979年起Unix就开始使用Bourne Shell,Bourne Shell 的主文件名为sh。
- **C Shell**: C Shell 主要在BSD版的Unix系统中使用, 其语法和c语言相类似而得名

- Shell的两种主要语法类型有Bourne和C，这两种语法彼此不兼容。Bourne家族主要包括sh,ksh,Bash,psh,zsh; C家族主要包括:csh,tcsh
- Bash: Bash与sh兼容，现在使用的Linux就是使用Bash作为用户的基本Shell。

Linux支持的Shell

- /etc/shells (shells里面有所有可用的Shell)

Shell脚本的执行方式

echo输出命令

- # echo [选项] [输出内容]
- 选项:
 - -e 支持反斜线控制的字符转换

| 控制字符 | 作用 |
|-------|-------------------------------------|
| \\ | 输出\本身 |
| \a | 输出警告音 |
| \b | 退格键，也就是向左删除键 |
| \c | 取消输出行末的换行符。和“-n”选项一致 |
| \e | ESCAPE键 |
| \f | 换页符 |
| \n | 换行符 |
| \r | 回车键 |
| \t | 制表符，也就是Tab键 |
| \v | 垂直制表符 |
| \0nnn | 按照八进制ASCII码表输出字符。其中0为数字零，nnn是三位八进制数 |
| \xhh | 按照十六进制ASCII码表输出字符。其中hh是两位十六进制数 |

第一个脚本

```
# vi hello.sh
#!/bin/bash ->(这里标称下面为Shell脚本,不能省略)
#The first program (注释)
# Author: ccchenji (注释)
```

```
echo -e 'hello word!'
```

脚本运行

- 赋予执行权限，直接运行
 - chmod 755 hello.sh
 - ./hello.sh (必须为绝对路径或者相对路径)
- 通过Bash调用执行脚本
 - bash hello.sh

注意：

- 在win中编写的Shell脚本放到Linux中得转换格式因为win中有一些格式和Linux中不一样,如回车(可以用 cat -A 查看文件隐藏字符,如回车换行)可以使用dos2unix插件

Bash基本功能

历史命令与命令补全

历史命令

- # history [选项] [历史命令保存文件]
- 选项：
 - -c：清空历史命令
 - -w：把缓存中的历史命令写入历史命令保存文件 ~/.bash_history
- 历史命令默认会保存100条，可以在环境变量配置文件/etc/profile中进行修改

历史命令调用

- 使用上，下箭头调用以前的历史命令
- 使用"!n"重复执行第n条历史命令
- 使用"!!"重复执行上一条命令
- 使用"!字符串"重复执行最后一条以改字符串开头的命令

命令与文件补全

- 在Bash中，命令与文件补全是非常方便与常用的功能，我们只要在输入命令或文件时，按"Tab"键就会自动进行补全

命令别名与常用快捷键

命令别名

- # alias 别名="原命令" (设定命令别名)

- # alias (查询命令别名)

命令执行时顺序

1. 第一顺位执行用绝对路径或相对路径执行的命令。
2. 第二顺位执行别名。
3. 第三顺位执行Bash的内部命令
4. 第四顺位执行按照\$PATH环境变量定义的目录查找顺序找到的第一个命令

让别名永久生效

- # vim /root/.bashrc

删除别名

- # unalias 别名

Bash常用快捷键

| 快捷键 | 作用 |
|--------|--|
| ctrl+A | 把光标移动到命令行开头。如果我们输入的命令过长，想要把光标移动到命令行开头时使用。 |
| ctrl+E | 把光标移动到命令行结尾。 |
| ctrl+C | 强制终止当前的命令。 |
| ctrl+L | 清屏，相当于clear命令。 |
| ctrl+U | 删除或剪切光标之前的命令。我输入了一行很长的命令，不用使用退格键一个一个字符的删除，使用这个快捷键会更加方便 |
| ctrl+K | 删除或剪切光标之后的内容。 |
| ctrl+Y | 粘贴ctrl+U或ctrl+K剪切的内容。 |
| ctrl+R | 在历史命令中搜索，按下ctrl+R之后，就会出现搜索界面，只要输入搜索内容，就会从历史命令中搜索。 |
| ctrl+D | 退出当前终端。 |
| ctrl+Z | 暂停，并放入后台。这个快捷键牵扯工作管理的内容，我们在系统管理章节详细介绍。 |
| ctrl+S | 暂停屏幕输出。 |
| ctrl+Q | 恢复屏幕输出。 |

输入输出重定向

标准输入输出

| 设备 | 设备文件名 | 文件描述符 | 类型 |
|-----|-------------|-------|------|
| 键盘 | /dev/stdin | 0 | 标准输入 |
| 显示器 | /dev/stdout | 1 | 标准输出 |

| 设备 | 设备文件名 | 文件描述符 | 类型 |
|-----|-------------|-------|--------|
| 显示器 | /dev/sdterr | 2 | 标准错误输出 |

输出重定向

| 类型 | 符号 | 作用 |
|-----------|------------|-------------------------------|
| 标准输出重定向 | 命令 > 文件 | 以覆盖的方式，把命令的正确输出输出到指定的文件或设备当中。 |
| | 命令 >> 文件 | 以追加的方式，把命令的正确输出输出到指定的文件或设备当中。 |
| 标准错误输出重定向 | 错误命令 2>文件 | 以覆盖的方式，把命令的错误输出输出到指定的文件或设备当中。 |
| | 错误命令 2>>文件 | 以追加的方式，把命令的错误输出输出到指定的文件或设备当中。 |

| | | |
|---------------|-------------------|-------------------------------|
| 正确输出和错误输出同时保存 | 命令 > 文件 2>&1 | 以覆盖的方式，把正确输出和错误输出都保存到同一个文件当中。 |
| | 命令 >> 文件 2>&1 | 以追加的方式，把正确输出和错误输出都保存到同一个文件当中。 |
| | 命令 &> 文件 | 以覆盖的方式，把正确输出和错误输出都保存到同一个文件当中。 |
| | 命令 &>> 文件 | 以追加的方式，把正确输出和错误输出都保存到同一个文件当中。 |
| | 命令 >> 文件1 2>> 文件2 | 把正确的输出追加到文件1中，把错误的输出追加到文件2中。 |

输入定向

- # wc [选项] [文件名]
- 选项:
 - -c 统计字节数
 - -w 统计单次数
 - -l 统计行数
- 命令<文件 把文件作为命令的输入

多命令顺序执行和管道符

多命令顺序执行

| 多命令执行符 | 格式 | 作用 |
|--------|--------------|--|
| ; | 命令1; 命令2 | 多个命令顺序执行，命令之间没有任何逻辑联系 |
| && | 命令1&& 命令2 | 逻辑与。当命令1正确执行，则命令2才会执行。 当命令1执行不正确，则命令2不会执行 |
| | 命令1 命令2 | 逻辑或。当命令1执行不正确，则命令2才会执行。 当命令1正确执行，则命令2不会执行 |

dd(磁盘或文件复制)

- # dd if=输入文件 of=输出文件 bs=字节数 count=个数
- 选项:
 - if=输入文件 指定源文件或源设备
 - of=输出文件 指定目标文件或目标设备
 - bs=字节数 指定一次输出/输入多少字节，即把这些字节看作一个数据块
 - count=个数 指定输入/输出多少个数据块
- 例子：# date ; dd if=/dev/zero of=/root/testfile bs=1k count=100000; date

管道符

- 命令格式：# 命令1 | 命令2 (命令1的正确输出作为命令2的操作对象)

通配符与其他特殊符号

通配符

| 通配符 | 作用 |
|-----|----------|
| ? | 匹配一个任意字符 |

| 通配符 | 作用 |
|-----|---|
| * | 匹配0个或任意多个任意字符，也就是可以匹配任何内容 |
| [] | 匹配中括号中任意一个字符。例如：[abc]代表一定匹配一个字符，或者是a，或者是b，或者是c。 |
| [-] | 匹配中括号中任意一个字符，-代表一个范围。例如：[a-z]代表匹配一个小写字母。 |
| [^] | 逻辑非，表示匹配不是中括号内的一个字符。例如：[^0-9] 代表匹配一个不是数字的字符。 |

Bash中其他特殊符号

| 符号 | 作用 |
|------|---|
| ' | 单引号，在单引号中所有的特殊符号，如"\$"和"\'"（反引号）都没有特殊含义 |
| "" | 双引号，在双引号中特殊符号都没有特殊含义，但是"\$"、"\``"和"\\"是例外，拥有“调节变量的值”、“引用命令”和“转义符”的特殊含义。 |
| `` | 反引号。反引号括起来的内容是命令，在Bash中会先执行它，和\$()作用一样，不过推荐使用\$(),因为反引号非常容易看错。 |
| \$() | 和反引号作用一样，用来引用系统命令。 |
| # | 在Shell脚本中，#开头的行代表注释 |
| \$ | 用于调用变量的值，如需要调用变量name的值时，需要用\$name的方式得到变量的值 |
| \ | 转移符，跟在\之后的特殊符号将失去特殊含义，变为普通字符。如\\$将输出\$ |

Bash变量

用户自定义变量

什么是变量

- 变量是计算机内存的单元，其中存放的值可以改变。当Shell脚本需要保存一些信息时，如一个文件名或是一个数字，就把它存放在一个变量中。每一个变量有一个名字，所以很容易引用它。使变量可以保存有用信息，是系统获知用户相关设置，变量也可以用于保存暂时信息。

变量设置规则

- 变量名称可以有字母，数字和下划线组成，但是不能以数字开头。如果变量名是"2name"则是错误的。

- 在Bash中，变量的默认类型都是字符串型，如果要进行数值运算，则必须指定变量类型为数值型。
- 变量用等号连接值，等号左右两侧不能有空格。
- 变量的值如果有空格，需要使用单引号或双引号包括。
- 在变量的值中，可以使用""转移符。
- 如果需要增加变量的值，那么可以进行变量值的叠加。不过变量需要用双引号包含"变量名"或用{变量名}包含。
- 如果是把命令的结果作为变量值赋予变量，则需要使用反引号或\$()包含命令。
- 环境变量名建议大写，便于区分。

变量分类

- 用户自定义变量
- 环境变量：这种变量中主要保存的是和系统操作环境相关的数据。
- 位置参数变量：这种变量主要是用来在脚本当中传递参数或数据的，变量名不能自定义，变量作用是固定的。
- 预定义变量：是Bash中已经定义好的变量，变量名不能自定义，变量作用也是固定的。

本地变量(用户自定义变量)

- 变量定义：# name="ccchenji"
- 变量叠加：
 - # aa=123
 - aa="\$aa"456
 - aa=\${aa}789
- 变量调用：# echo \$name
- 变量查看：# set
- 变量删除：# unset name

环境变量

环境变量是什么

- 用户自定义变量只在当前的Shell中生效，而环境变量会在当前Shell和这个Shell的所有子Shell当中生效。如果把环境变量写入相应的配置文件，那么这个环境变量就会在所有的Shell中生效。

设置环境变量

- export 变量名=变量值(声明变量)
- env (查询环境变量)
- unset 变量名(删除变量)
- pstree 查看进程树

系统常见环境变量

- PATH: 系统查找命令的路径
 - # echo \$PATH -> /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin
 - PATH="\$PATH":/root/sh (#PATH变量添加)
- PS1: 定义系统提示符的变量

\d: 显示日期，格式为“星期 月 日”

\h: 显示简写主机名。如默认主机名“localhost”

\t: 显示24小时制时间，格式为“HH:MM:SS”

\T: 显示12小时制时间，格式为“HH:MM:SS”

\A: 显示24小时制时间，格式为“HH:MM”

\u: 显示当前用户名

\w: 显示当前所在目录的完整名称

\W: 显示当前所在目录的最后一个目录

\#: 执行的第几个命令

\\$: 提示符。如果是root用户会显示提示符为“#”，如果是普通用户会显示提示符为“\$”

位置参数变量

位置参数变量

| 位置参数变量 | 作用 |
|--------|---|
| \$n | n为数字，\$0代表命令本身，\$1-\$9代表第一到第九个参数，十以上的参数需要用大括号包含，如\${10} |
| \$* | 这个变量代表命令行中所有的参数，\$*把所有的参数看成一个整体 |
| \$@ | 这个变量也代表命令行中所有的参数，不过\$@把每个参数区分对待 |
| \$# | 这个变量代表命令行中所有参数的个数 |

预定义变量

预定义变量

| 预定义变量 | 作用 |
|-------|----|
|-------|----|

| 预定义变量 | 作用 |
|-------|---|
| \$? | 最后一次执行的命令的返回状态。如果这个变量的值为0,证明上一个命令正确执行;如果这个变量的值为非0(具体哪个数,由命令自己来决定),则证明上一个命令执行不正确了。 |
| \$\$ | 当前进程的进程号(PID) |
| \$! | 后台运行的最后一个进程的进程号(PID) |

```
#!/bin/bash

echo "The current process is $$"
#输出当前进程的PID
#这个PID就是当前脚本执行时,生成的PID

find /root/桌面 -name show.sh &
#使用find命令在桌面目录下查找show.sh文件
#符号&的意思是把命令放入后台执行。

echo "The last one Daemon process is $!"
```

接受键盘输入

- # read [选项] [变量名]
- 选项:
 - -p: "提示信息":在等待read输入时,输出提示信息
 - -t: 秒数: read命令会一直等待用户输入,使用此选项可以指定等待时间
 - -n: 字符数: read命令只接受指定的字符数,就会执行
 - -s: 隐藏输入的数据,使用与机密信息的输入

数值运算与运算符

注意:

- 在bash中所有变量的默认类型为**字符串型**

declare声明变量类型

- # declare [+/-] [选型] 变量名
- 选项:
 - -:给变量设定类型属性
 - +:取消变量的类型属性

- -i: 将变量声明为整数型(integer)
- -x: 将变量声明为环境变量
- -p: 显示指定变量的被声明的类型

数值运算-方法1

- # aa=11
- # bb=22 (给变量aa,bb赋值)
- # declare -i cc=aa+bb

数值运算-方法2: expr或let数值运算工具

- # aa=11
- # bb=22 (给变量aa和bb赋值)
- # dd=\$(expr aa+bb) (#dd的值是aa和bb的和。注意“+”号左右两侧必须有空格)

数值运算：方法3：“\$((运算式))”或“\${[运算式]}”

- # aa=11
- # bb=22
- # ff=\$((aa+bb))
- # gg=\${aa+\$bb};

运算符

| 优先级 | 运算符 | 说明 |
|-----|---|-------------------|
| 13 | -, + | 单目负、单目正 |
| 12 | !, ~ | 逻辑非、按位取反或补码 |
| 11 | * , / , % | 乘、除、取模 |
| 10 | +, - | 加、减 |
| 9 | << , >> | 按位左移、按位右移 |
| 8 | <=, >=, <, > | 小于或等于、大于或等于、小于、大于 |
| 7 | ==, != | 等于、不等于 |
| 6 | & | 按位与 |
| 5 | ^ | 按位异或 |
| 4 | | 按位或 |
| 3 | && | 逻辑与 |
| 2 | | 逻辑或 |
| 1 | =, +=, - =, *=, /=, %=, &=, ^= =, <<=, >>= | 赋值、运算且赋值 |

变量测试与内容替换

| 变量置換方式 | 变量y没有设置 | 变量y为空值 | 变量y设置值 |
|-------------|-------------------|-------------|------------|
| x=\${y-新值} | x=新值 | x为空 | x=\$y |
| x=\${y:-新值} | x=新值 | x=新值 | x=\$y |
| x=\${y+新值} | x为空 | x=新值 | x=新值 |
| x=\${y:+新值} | x为空 | x为空 | x=新值 |
| x=\${y=新值} | x=新值, y=新值 | x为空,y值不变 | x=\$y,y值不变 |
| x=\${y:=新值} | x=新值,y=新值 | x=新值,y=新值 | x=\$y,y值不变 |
| x=\${y?新值} | 新值输出到标准错误输出(就是屏幕) | x为空 | x=\$y |
| x=\${y:?新值} | 新值输出到标准错误输出 | 新值输出到标准错误输出 | x=\$y |

环境变量配置文件

环境变量配置文件简介

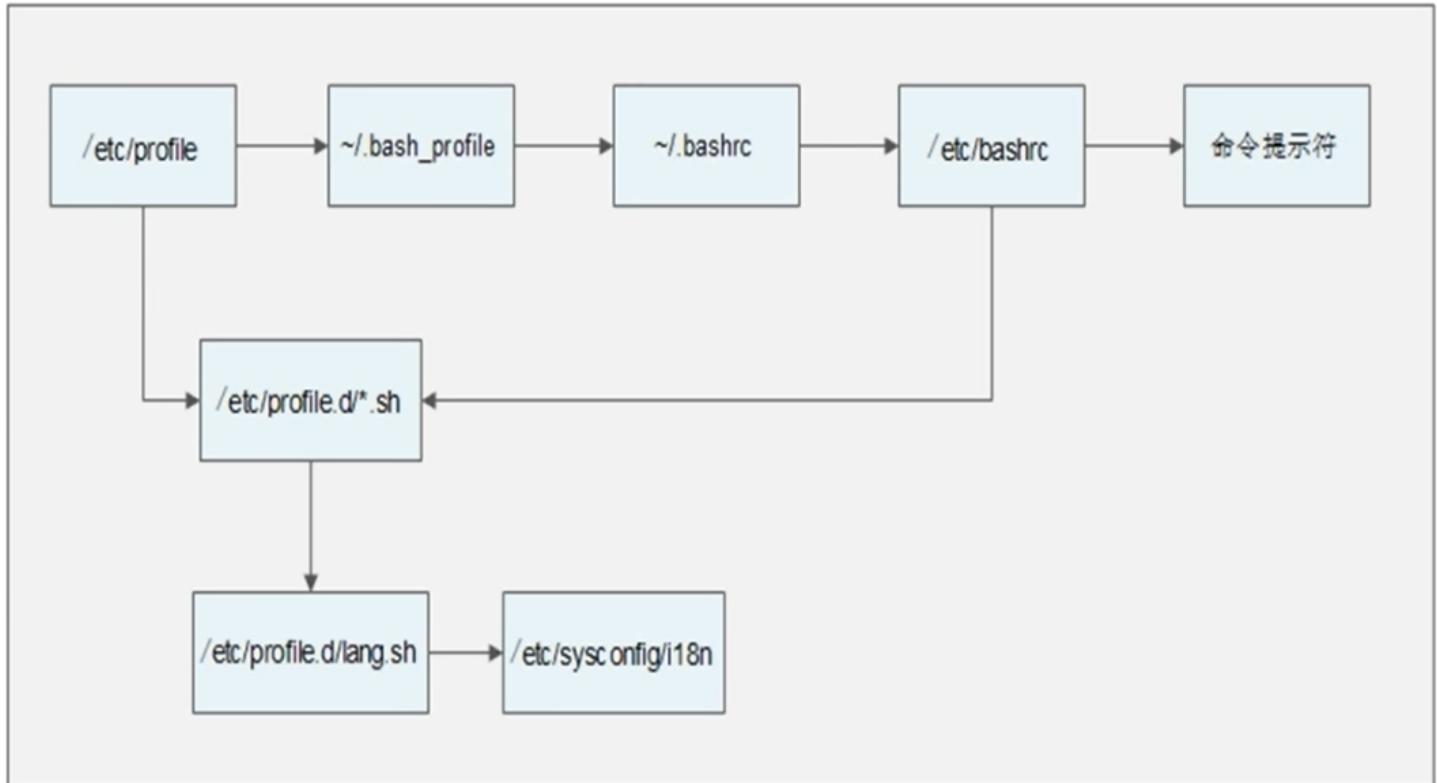
source命令(不用再重启服务器让配置文件直接生效)

- # source 配置文件 或 # . 配置文件

环境变量配置文件简介

- 环境变量配置文件中主要是定义对系统的操作环境生效的默认环境变量，比如 PATH,HISTSIZE,PS1,HOSTNAME等默认环境变量。
- /etc/profile
- /etc/profile.d/*.sh
- ~/.bash_profile
- ~/.bashrc
- /etc/bashrc

环境变量配置文件作用



/etc/profile的作用

- USER变量, LOGNAME变量, MAIL变量, PATH变量, HOSTNAME变量, umask, 调用/etc/profile.d/*.sh文件

~/.bash_profile的作用

- 调用了~/.bashrc文件
- 在PATH变量后面加入了“:\$HOME/bin”这个目录

其他配置文件和登陆信息

注销时生效的环境变量配置文件

- ~/.bash_logout(注销登陆执行)
- ~/bash_history

Shell登陆信息

- 本地终端欢迎信息:/etc/issue

| 转义符 | 作用 |
|-----|----------|
| \d | 显示当前系统日期 |
| \s | 显示操作系统名称 |

| 转义符 | 作用 |
|-----|------------------------|
| \l | 显示登陆的终端号,这个比较常用 |
| \m | 显示硬件体系结构, 如i386, i686等 |
| \n | 显示主机名 |
| \o | 显示域名 |
| \r | 显示内核版本 |
| \t | 显示当前系统时间 |
| \u | 显示当前登陆用户的序列号 |

远程终端欢迎信息：/etc/issue.net

- 转义符在/etc/issue.net文件中不能使用
- 是否显示此欢迎信息，有ssh的配置文件/etc/ssh/sshd_config决定，加入"Banner /etc/issue.net"行才能显示(记得重启SSH服务)

登陆后欢迎信息：/etc/motd

不管是本地登陆，还是远程登陆，都可以显示此欢迎信息。

Shell编程

基础正则表达式

正则表达式与通配符

- 正则表达式是用来在文件中匹配符合条件的字符串，正则是包含匹配。grep,awk,sed等命令可以支持正则表达式。
- 通配符用来匹配符合条件的文件名，通配符是完全匹配。ls,find,cp这些命令不支持正则表达式，所以只能使用shell自己的通配符来进行匹配。

基础正则表达式

| 元字符 | 作用 |
|-----|----------------|
| * | 前一个字符匹配0次或任意多次 |
| . | 匹配除换行符外任意一个字符 |

| 元字符 | 作用 |
|---------|---|
| ^ | 匹配行首。例如：^hello会匹配以hello开头的行 |
| \$ | 匹配行尾。例如：hello\$会匹配以hello结尾的行 |
| [] | 匹配中括号中指定的任意一个字符，只匹配一个字符。例如：[aoeiu] 匹配任意一个原音字母， [0-9]匹配任意一个位数字， [a-z][0-9] 匹配小写字母和一位数字构成的两位字符。 |
| [^] | 匹配除中括号以外的任意一个字符，例如：[^0-9]匹配任意一位非数字字符。[a-zA-Z] 表示任意一位非小写字母。 |
| \ | 转义符。用于取消特殊符号的含义 |
| \{n\} | 表示前面的字符恰好出现n次。例如：[0-9]\{4\}匹配4位数字， [1][3-8][0-9]\{9\} 匹配手机号码。(这里的\是为了消除{}的含义) |
| \{n,\} | 表示其前面的字符出现不小于n次。例如：[0-9]\{2,\}表示两位及以上的数字。 |
| \{n,m\} | 表示其前两位的字符至少出现n次，最多出现m次，例如：[a-zA-Z]\{6,8\} 匹配6到8位的小写字母 |

字符截取命令

cut命令

- # cut [选项] 文件名
- 选项：
 - -f 列号：提取第几列
 - -d 分隔符：按照指定分隔符分割列
 - # cut -f 2,3 student.txt
 - # cut -d ":" -f 1,3 student.txt

printf命令

- printf ‘输出类型输出格式’ 输出内容
- 输出类型：
 - %ns:输出字符串。n是数字代指输出几个字符
 - %ni:输出整数。n是数字指代输出几个数字。
 - %m.nf:输出浮点数。m和n是数字，指代输出的整数位数和小数位数。如%8.2f代表共输出8位数，其中2位是小数，6位是整数。
- 输出格式：

- \a:输出警告声音
- \b:输出退格键，也就是Backspace键
- \f:清除屏幕
- \n:换行
- \r:回车，也就是Enter键
- \t:水平输出推个键，也就是Tab键
- \v:垂直输出退格键，也就是Tab键
- 在awk命令的输出中支持print和printf命令
 - print: print会在每一个输出之后自动加一个换行符(Linux默认没有print命令)
 - printf: printf是标准格式输出命令，并不会自动加入换行符，如果需要换行，需要手工加入换行符

awk命令(究极牛皮的命令)

- # awk '条件1{动作1} 条件2{动作2}...' 文件名
- 条件(Pattern):
 - 一般使用关系表达式作为条件
 - x>10 判断变量x是否大于10
 - x>=10 大于等于
 - x<=10 小于等于10
- 动作(Action)
 - 格式化输出
 - 流程控制语句
- awk '{printf \$2 "\t" \$6 "\n"}' student.txt

sed命令

- sed是一种几乎包括在所有UNIX平台(包括Linux)的轻量级流编辑器。sed主要是用来将数据进行选取，替换，删除，新增的命令。
- # sed[选项] '[动作]' 文件名
- 选项:
 - -n: 一般sed命令会把所有数据都输出到屏幕，如果加入此选择，则只会把经过sed命令处理的行输出到屏幕。
 - -e:允许对输入数据应用多条sed命令编辑
 - -i:用sed的修改结果直接修改读取数据的文件，而不是由屏幕输出。
- 动作:
 - a \:追加，在当前行后添加一行或多行。添加多行时，除最后一行外，每行末尾需要\"代表数据未完结。
 - c \:行替换，用c后面的字符串替换原数据行，替换多行时，除最后一行外，每行末尾需要\"代表数据未完结。

- i \: 插入，在当期行前插入一行或多行。插入多行时，除最后一行外，每行末尾需要用\"代表数据未完结。
- d: 删除，删除指定的行。
- p: 打印，输出指定的行。
- s: 字串替换，用一个字符串替换另外一个字符串。格式为“行范围”s/旧字串/新字串/g(和vim中的替换格式类似)

字符处理命令

排序命令sort

- # sort [选项] 文件名
- 选项：
 - -f: 忽略大小写
 - -n: 以数值型进行排序，默认使用字符串型排序
 - -r: 反向排序
 - -t: 指定分隔符。默认分隔符是制表符
 - -k [n,m]: 按照指定的字段范围排序。从第n字段开始，m字段结束(默认到行尾)

统计命令wc

- # wc [选项] 文件名
- 选项：
 - -l: 只统计行数
 - -w: 只统计单词数
 - -m: 只统计字符数

条件判断

按照文件类型进行判断

| 测试选项 | 作用 |
|------|----------------------------------|
| -b文件 | 判断该文件是否存在，并且是否为块设备文件(是块设备文件为真) |
| -c文件 | 判断该文件是否存在，并且是否为字符设备文件(是字符设备文件为真) |
| -d文件 | 判断该文件是否存在，并且是否为目录文件(是目录为真) |
| -e文件 | 判断该文件是否存在(存在为真) |
| -f文件 | 判断该文件是否存在，并且是否为普通文件(是普通文件为真) |

| 测试选项 | 作用 |
|------|----------------------------------|
| -L文件 | 判断该文件是否存在，并且是否为符号链接文件(是符号链接文件为真) |
| -p文件 | 判断该文件是否存在，并且是否为管道文件(是管道文件为真) |
| -s文件 | 判断该文件是否存在，并且是否为空(非空为真) |
| -S文件 | 判断文件是否存在，并且是否为套接字文件(是套接字文件为真) |

两种判断格式

- # test -e /root/install.log
- # [-e /root/install.log]

按照文件权限进行判断

| 测试选项 | 作用 |
|-------|--------------------------------------|
| -r 文件 | 判断该文件是否存在，并且是否该文件拥有读权限(有读权限为真) |
| -w 文件 | 判断该文件是否存在，并且是否该文件拥有写权限(有写权限为真) |
| -x 文件 | 判断该文件是否存在，并且是否该文件拥有执行权限(有执行权限为真) |
| -u 文件 | 判断文件是否存在，并且是否该文件拥有SUID权限(有SUID权限为真) |
| -g 文件 | 判断该文件是否存在，并且是否该文件拥有SGID权限(有SGID权限为真) |
| -k 文件 | 判断文件是否存在，并且是否该文件拥有SBit权限(有SBit权限为真) |

两个文件之间进行比较

| 测试选项 | 作用 |
|----------------|---|
| 文件1 -nt 文件2 | 判断文件1的修改时间是否比文件2的新 (如果新则为真) |
| 文件1 -ot 文件2 | 判断文件1的修改时间是否比文件2的旧(如果旧则为真) |
| 文件1 -ef 文件2 | 判断文件1是否和文件2的inode号一致，可以理解为两个文件是否为同一文件， 这个判断用于判断硬链接是很好的方法 |

两个整数之间比较

| 测试选项 | 作用 |
|-------------|------------------------|
| 整数1 -eq 整数2 | 判断整数1是否和整数2相等(相等为真) |
| 整数1 -ne 整数2 | 判断整数1是否和整数2不相等(不相等为真) |
| 整数1 -gt 整数2 | 判断整数1是否大于整数2(大于为真) |
| 整数1 -lt 整数2 | 判断整数1是否小于整数2(小于为真) |
| 整数1 -ge 整数2 | 判断整数1是否大于等于整数2(大于等于为真) |
| 整数1 -le 整数2 | 判断整数1是否小于等于整数2(小于等于为真) |

字符串的判断

| 测试选项 | 作用 |
|--------------|--------------------------|
| -z字符串 | 判断字符串是否为空(为空返回真) |
| -n字符串 | 判断字符串是否为非空(非空返回真) |
| 字符串1==字符串2 | 判断字符串1是否和字符串2相等(相等返回真) |
| 字符串1 != 字符串2 | 判断字符串1是否和字符串2不相等(不相等放回真) |

注意：

- 一对方括号，不支持关系运算符；[] 支持关系运算符；[] 和 [] 判断的都是字符串，要进行数值判断用命令(如 -gt)或者(());

多重条件判断

| 测试选线 | 作用 |
|------------|---------------------------|
| 判断1 -a 判断2 | 逻辑与，判断1和2都成立，最终的结果才为真 |
| 判断1 -o 判断2 | 逻辑或，判断1和判断2有一个成立，最后的结果就为真 |
| ! 判断 | 逻辑非，使原始的判断式取反 |

流程控制

if语句

单分支if条件语句

```
if [ 条件判断式 ];then  
    程序  
fi  
或者  
if [ 条件判断式 ]  
    then  
        程序  
fi
```

单分支语句需要注意几个点

- if语句使用fi结尾，和一般语言使用大括号结尾不同
- [条件判断式] 就是使用test命令判断，所以中括号和条件判断之间必须有空格。
- then后面跟符合条件之后执行的程序，可以放再[]之后，用";"分割。也可以换行写入，就不需要";"。

双分支语句

```
if [ 条件判断式 ]  
    then  
        条件成立时，执行的程序  
    else  
        条件不成立时，执行的另一个程序  
fi
```

多分支if条件语句

```
if [ 条件判断式1 ]  
    then  
        当条件判断式1成立时，执行程序1  
elif [ 条件判断式2 ]  
    then  
        当条件判断式2成立时，执行程序2  
else  
    当所有条件都不成立时，最后执行此程序  
fi
```

case语句

多分支case条件语句

- case语句和if...elif...else语句一样都是多分支条件语句，不过和if多分支条件语句不同的是，case语句只能判断一种条件关系，而if语句可以判断多种条件关系。

```

case $变量名 in
    "值1")
        如果变量的值等于1，则执行程序1
        ;;
    "值2")
        如果变量的值等于值2，则执行程序2
        ;;
    ...
    *)
        如果变量的值都不是以上的值，则执行此程序
        ;;
esac

```

for循环

语法一

```

for 变量 in 值1 值2 值3
do
    程序
done

```

```

ls *.tar.gz >ls.log
for i in $(cat ls.log)
do
    tar -zxf $i &>dev/null #dev/null是回收站
done
rm -f ls.log

```

语法二

```

for((初始值;循环控制条件;变量变化))
do
    程序
done

```

while循环和until循环

while循环

```
while command
do
...
done
```

until循环

```
until [条件判断式]
do
    程序
done
```

break和continue

break

```
i=10
while ((i--))
do
    echo $i
    for((a=0;a<5;a++))
    do
        echo "a=$a"
        if((a==1));then
            break #跳出for循环
            #break 2 跳出两层循环,即while循环
        fi
    done
done
```

continue

- 和break用法一样，区别是continue是跳出当前的一次循环

函数

格式

```
[function] functionName(){
    ...
    [return value]
}
```

#function 可选，可加可不加

注意事项

- 函数返回值，可以显示增加return语句；如果不加，会将最后一条命令运行结果作为返回值。
- Shell函数返回值只能是整数，一般用来表示函数执行成功与否，0表示成功，其他值表示失败。如果return 其他数据会得到错误提示。
- 调用函数只需要给出函数名，不需要加括号
- 函数的返回值可以在函数执行以后通过\$?来获得
- 可以使用unset命令删除函数
 - unset .f functionName

函数参数

- 在Shell中，调用函数可以向其传递参数。在函数内部，通过调用 n 的形式来获取参数的值。 1 代表第一个参数。当参数值大于等于10时应该加大如 \${10}。

| 特殊变量 | 说明 |
|------|--------------------|
| \$# | 传递给函数的参数个数 |
| \$* | 显示所有传递给函数的参数 |
| \$@ | 与\$*相同，但是略有区别，上面讲过 |
| \$? | 函数的返回值 |

Linux服务管理

服务分类

- ### 服务分类
- Linux服务
 - RPM包默认服务
 - 独立的服务
 - 基于xinetd服务
 - 源码包安装的服务

启动与自启动

- 服务启动：就是在当前系统中让服务运行，并提供功能
- 服务自启动：自启动是指让服务在系统开机或重启之后，随着系统的启动而自动启动服务。

查询已安装的服务

- RPM包安装的服务
 - chkconfig --list (查看服务自启动状态，可以看到所有RPM包安装的服务)
- 源码包安装的服务
 - 查看服务安装的位置，一般是/usr/local下

RPM服务的管理

独立服务管理

RPM包安装服务的位置

- RPM安装服务和源码包安装服务的区别就是安装位置的不同
 - 源码包安装在指定位置，一般/usr/local
 - RPM包安装在默认位置中

独立服务的启动

- /etc/init.d/独立服务名 start|stop|status|restart (centos7不是这样,centos7是在/usr/lib/systemd/system)
- service 独立服务名 start|stop|restart|status (centos7用systemcal)
- centos7里的systemctl融合了service和chkconfig命令(两个命令有的功能它都有)

独立服务的自启动

1. 修改/etc/rc.d/rc.local文件 (标准方法,centos中该文件权限被降低需要手工添加执行权限)
2. 使用ntsysv命令管理自启动(redhat自己的命令)
3. 后面补充

基于xinetd服务的管理

安装xinetd与telne

- 现在基于xinetd的服务很少了所以了解一下就行

源码包安装服务的管理

源码包安装服务的启动

- 使用绝对路径，调用启动脚本来启动。不同的源码包的启动脚本不同。可以查看源码包的安装说明，查看启动脚本的方法。

源码包服务的自启动

- # vim /etc/rc.d/rc.local (修改这个文件)

Linux系统管理

进程管理

进程查看

进程简介

- 进程是正在执行的一个程序或者命令，每一个进程都是一个运行的实体，都有自己的地址空间，并占用一定的系统资源。

进程管理的作用

- 判断服务器健康状态
- 查看系统中所有进程
- 杀死进程

查看系统中所有进程

- # ps aux (查看系统中所有进程，使用BSD操作系统格式)
- # ps le (查看系统中所有进程，使用Linx标准命令格式)

| 名称 | 作用 |
|------|--|
| USER | 该进程是由哪个用户产生 |
| PID | 进程的ID号； |
| %CPU | 该进程占CPU资源的百分比，占用越高，进程越耗费资源； |
| %MEN | 该进程占用物理内存的百分比，占用越高，进程越耗费资源； |
| VSZ | 该进程占用虚拟内存的大小，单位为KB |
| RSS | 该进程占用实际物理内存的大小，单位为KB |
| TTY | 该进程是在哪个终端中运行的。其中tty1-tty7代表本地控制台终端，tty1-tty6是本地的字符界面终端，tty7是图形终端。pts/0-256代表虚拟终端 |

| 名称 | 作用 |
|---------|---|
| STAT | 进程状态。常见的状态有：R：运行、S：睡眠、T：停止、s：包含子进程、+：位于后台 |
| START | 该进程的启动时间 |
| TIME | 该进程占用CPU的运算时间，注意不是系统时间 |
| COMMAND | 产生此进程的命令名 |

查看系统健康状态

- # top [选项]
- 选项：
 - -d秒数：指定top命令每隔几秒更新，默认是3秒
- 在top命令的交互模式中可以执行的命令：
 - ?或h:显示交互模式的帮助
 - P:以CPU使用率排序，默认就是此选项
 - M:以内存的使用率排序
 - N:以PID排序
 - q:退出top

第一行信息为任务队列信息

| 内容 | 说明 |
|---------------------------------|--|
| 12:26:46 | 系统当前时间 |
| up 1 day , 13:32 | 系统的运行时间，本机已经运行1天13小时32分钟 |
| 2 users | 当前登陆了两个用户 |
| load average: 0.00,0.00,0.00 | 系统在之前1分钟，5分钟，15分钟的平均负载。一般认为小于1小时，负载较小。如果大于1，系统已经超出负荷 |

第二行为进程信息

| 内容 | 说明 |
|-----------------|----------|
| Tasks: 95 total | 系统中的进程总数 |
| 1 running | 正在运行的进程数 |
| 94 sleeping | 睡眠的进程 |

| 内容 | 说明 |
|-----------|-----------------------|
| 0 stopped | 正在终止的进程 |
| 0 zombie | 僵尸进程，如果不是0，需要手工检查僵尸进程 |

第三行为CPU信息

| 内容 | 说明 |
|-------------------|--|
| Cpu(s): 0.1%us | 用户模式占用的CPU百分比 |
| 0.1%sy | 系统模式占用的CPU百分比 |
| 0.0%ni | 改变过优先级的用户进程占用的CPU百分比 |
| 99.7%id | 空闲CPU的CPU百分比 |
| 0.1%wa | 等待输入/输出的进程的占用CPU百分比 |
| 0.0%hi | 硬中断请求服务占用的CPU百分比 |
| 0.1%si | 软中断请求服务占用的CPU百分比 |
| 0.0%st | st(Steal time)虚拟时间百分比，就是当前虚拟机时， 虚拟机CPU等待实际CPU的时间百分比 |

第四行为物理内存信息

| 内容 | 说明 |
|------------------|---------------|
| Mem: num k total | 物理内存的总量，单位为KB |
| num k used | 已经使用的物理内存数量 |
| num k free | 空闲的物理内存数量 |
| num k buffers | 作为缓冲的内存数量 |

第五行为交换分区(swap)信息

| 内容 | 说明 |
|-------------------|-----------------|
| Swap: num k total | 交换分区(虚拟内存) 的总大小 |
| 0K used | 已经使用的交换分区的大小 |

| 内容 | 说明 |
|--------------|--------------|
| num k free | 空闲交换分区的大小 |
| num k cached | 作为缓冲的交换分区的大小 |

查看进程树

- # pstree [选项]
- 选项：
 - -p：显示进程的PID
 - -u：显示进程的所属用户

终止进程

kill命令

- # kill -l (查看可用的进程信号)

常用信号说明

| 信号代号 | 信号名称 | 说明 |
|------|---------|--|
| 1 | SIGHUP | 该信号让进程立即关闭，然后重新读取配置文件之后重启 |
| 2 | SIGINT | 程序终止信号，用于终止前台程序。相当于输出ctrl+c快捷键 |
| 8 | SIGFPE | 在发生置命的算数运算错误时发生，不仅包括浮点运算错误，还包括溢出及除数为0等其他所有的算数的错误 |
| 9 | SIGKILL | 用来立即结束程序的运行。本信号不能被阻塞、处理和忽略。 一般用于强制终止进程 |
| 14 | SIGALRM | 时钟定时信号，计算的是实际时间或时钟时间。alarm函数使用信号 |
| 15 | SIGTERM | 正常结束进程的信号，kill命令的默认信号， 有时如果进程已经发生问题，这个信号是无法正常终止进程的。 我们才会尝试SIGKILL信号，也就是信号9 |
| 18 | SIGCONT | 该信号可以让暂停的进程回复执行，本信号不能被阻断 |
| 19 | SIGSTOP | 该信号可以暂停前台进程，相当于输入ctrl+z快捷键。本信号不能被阻断 |

使用命令

- # kill -l 22356 (重启进程)

- # kill -9 22357 (强制杀死进程)

killall命令

- # killall [选项] [信号] 进程名(按照进程名杀死进程)
- 选项：
 - -i: 交互式，询问是否要杀死某个进程
 - -l: 忽略进程名的大小写

pkill命令

- # pkill [选项] [信号] 进程名 (按照进程名终止进程)
- 选项：
 - -t 终端号：按照终端号提出用户
- 例子：
 - # w (使用w命令查询本机已经登陆的用户)
 - # pkill -t -9 pts/1 (强制杀死从pts/1虚拟终端登陆的进程)

工作管理

把进程放入后台

- # tar -zcf etc.tar.gz /etc &
- # top (在top命令执行的过程中，按下ctrl+z快捷键)

查看后台的工作

- # jobs [-l]
- 选项：
 - -l :显示工作的PID
 - “+”号代表最近一个放入后台的工作，也是工作恢复时，默认恢复的刚工作。 “-”号代表倒数第二个放入后台的工作。

将后台暂停的工作恢复到前台执行

- # fg %工作号
- 参数：
 - %工作号：%号可以省略，但是注意工作号和PID的区别

把后台暂停的工作恢复到后台执行

- # bg %工作号(后台恢复执行的命令，是不能和前台有交互的，否则不能恢复到后台执行)

系统资源查看

vmstat命令监控系统资源

- # vmstat [刷新延时 刷新次数]
- 例子：vmstat 2 3

dmesg开机时内核检测信息

- # dmesg
- # dmesg | grep CPU

free命令查看内存使用状态

- # free [-b|-k|-m|-g]
- 选项：
 - -b:以字节为单位显示
 - -k:以KB为单位显示， 默认就是以KB为单位显示
 - -m:以MB为单位显示
 - -g:以GB为单位显示

缓冲和缓存的区别

- 简单来说缓存(cache)就是用来加速数据从硬盘中"读取"的， 而缓冲(buffer)是用来加速数据"写入"硬盘的。

查看CPU信息

- # cat /proc/cpuinfo

uptime命令

- # uptime (显示系统的启动时间和平均负载， 也就是top命令的第一行。w命令也可以看到这个数据)

查看系统与内核相关信息

- # uname [选项]
- 选项：
 - -a:查看系统所有相关信息;
 - -r:查看内核版本;
 - -s:查看内核名称;

判断当前系统的位数

- # file /bin/ls

查看当前Linux系统的发行版本

- # lsb_release -a

列出进程打开或使用的文件信息

- # lsof [选项] (列出进程调用或打开的文件的信息)
- 选项：
 - -c 字符串：只列出以字符串开头的进程打开的文件
 - -u 用户名：只列出某个用户的进程打开的文件
 - -p pid: 列出某个PID进程打开的文件

系统定时任务(很重要)

crond服务管理与访问控制

- # service crond restart
- # chkconfig crond on
- centos7用systemctl status crond.service 查看进程状态,systemctl list-unit-files 相当于 chkconfig --list

用户的crontab设置

- # crontab [选项]
- 选项：
 - -e:编辑crontab定时任务
 - -l:查询crontab任务
 - -r:删除当前用户所有的crontab任务
- # crontab -e (进入crontab)编辑界面。会打开vim编辑你的工作。
 - * * * * * 执行的任务

| 项目 | 含义 | 范围 |
|---------|-------------|----------------|
| 第一个"**" | 一个小时当中的第几分钟 | 0-59 |
| 第二个"**" | 一天当中的第几小时 | 0-23 |
| 第三个"**" | 一个月当中的第几天 | 1-21 |
| 第四个"**" | 一年当中的第几月 | 1-21 |
| 第五个"**" | 一周当中的星期几 | 0-7(0和7都代表星期日) |

| 特殊符号 | 含义 |
| --- | --- |
| * | 代表任何时间。比如第一个"**"就代表一小时中每分钟都执行一次的意思。 |
| , | 代表不连续的时间，比如"0 8,12,16 * * *" 命令，就代表在每天的8点0分，12点0分，16点0分都执行一次命令 |
| - | 代表连续的时间范围。比如"0 5 * * 1-6命令"，代表在周一到周六的凌晨5点0分执行命令 |
| */n | 代表没隔多久执行一次，比如"*/10 * * * * 命令"，代表没隔10分钟就执行一次命令 |

日志管理

日志简介

确定服务启动

- # ps aux | grep rsyslogd (查看服务是否启动)
- chkconfig --list | grep rsyslog(查看服务是否自启动)(centos7用systemctl list-unit-file | grep rsyslog.service来查看)

常见日志的作用

| 日志文件 | 说明 |
|-----------------|---|
| /var/log/cron | 记录了系统定时任务相关的日志 |
| /var/log/cups | 记录了打印信息的日志 |
| /var/log/dmesg | 记录了系统在开机时内核自检的信息。 也可以使用dmesg命令直接查看内核自检信息。 |
| /var/log/btmp | 记录错误登录的日志。这个文件是二进制文件， 不能直接vi查看，而要使用lastb命令查看。 |
| /var/log/lastl | 记录系统中所有用户最后一次的登陆时间的日志。 这个文件也是二进制文件，不能直接vi， 而要使用lastlog命令查看。 |
| /var/log/mailog | 记录邮件信息。 |

| 日志文件 | 说明 |
|---------------------------|--|
| /var/log/message(一个重要的日志) | 记录系统重要信息的日志，这个日志文件中会记录Linux系统的绝大多数重要信息，如果系统出现问题时，首先要检查的就应该是这个日志文件 |
| /var/log/secure | 记录验证和授权方面的信息，只要涉及账户和密码的程序都会记录，比如说系统的登陆，ssh的登陆，su切换用户，sudo授权，甚至添加用户和修改用户密码都会记录在这个日志文件 |
| /var/log/wtmp | 永久记录多有用户的登陆，注销信息，同时记录系统的启动，重启，关机事件。同样这个文件也是一个二进制文件，不能直接vi，而要使用last命令来查看 |
| /var/run/utmp | 记录当期已经登陆的用户的信息。这个文件会随着用户的登陆和注销而不断变化，只记录当前登陆用户的信息，同样这个文件不能直接vi，而要使用w,who,users等命令来查询 |

- 除了系统默认的日志之外，采用RPM方式安装的系统服务也会默认把日志记录在/var/log/目录中(源码包安装的服务日志也是在源码包指定目录中)。不过这些日志不是有rsylogd服务来记录和管理的，而是由各个服务使用自己的日志管理文档来记录自身日志

rsyslog日志服务

日志文件格式

- 基本日志格式包含以下四列：
 - 事件产生的时间；
 - 发生事件的服务器的主机名；
 - 产生事件的服务名或程序名；
 - 事件的具体信息；

/etc/rsyslog.conf配置文件

| authpriv.* | /var/log/secure |
|---|-----------------|
| 服务名称[连接符]日志等级;认证相关服务，所有日志等级 /var/log/secure日志中 | 日志记录位置， 记录在 |

服务名称

| 服务名称 | 说明 |
|-------------------|---|
| auth | 安全和认证相关消息(不推荐使用authpriv替代) |
| authpriv | 安全和认证相关消息(私有的) |
| cron | 系统定时任务cron和at产生的日志 |
| daemon | 和各个守护进程相关的日志 |
| ftp | ftp守护进程产生的日志 |
| kern | 内核产生的日志(不是用户进程产生的) |
| local0- local7 | 为本地使用预留的服务 |
| lpr | 打印产生的日志 |
| mail | 邮件收发信息 |
| news | 与新闻服务相关的日志 |
| syslog | 有syslogd服务产生的日志信息 (虽然服务名称已经改为syslogd) ,但是很多配置都还是沿用了syslogd的, 这里并没有修改服务名 |
| uucp | uucp子系统的日志信息, uucp是早期linux系统仅从数据传递的协议,后来也经常用在新闻组服务中 |

连接符号

- 连接符号可以识别为：
 - ""代表所有日志等级，比如："authpriv."代表authpriv认证信息服务产生的日志，所有的日志等级都记录。
 - ". ."代表只要比后面的等级高的(包含该等级)日志都记录下来。比如："cron.info"代表cron服务产生的日志，只要日志等级大于等于info级别，就记录
 - ".=."代表只记录所需等级的日志，其他等级的都不记录。比如":*=emerg"代表人和日志服务产生的日志，只要等级是emerg等级就记录。这种用法极少见，了解就好
 - ".!"代表不等于，也就是除了该等级的日志外，其他等级的日志都记录。

日志等级

| 等级名称 | 说明 |
|------|----|
|------|----|

| 等级名称 | 说明 |
|---------|------------------------------------|
| debug | 一般的调试信息说明 |
| info | 基本的通知信息 |
| notice | 普通信息，但有一定的重要性 |
| warning | 警告信息，但是还不影响到服务或系统的运行 |
| err | 错误信息，一般达到err等级的信息以及可以影响到服务或系统的运行了。 |
| crit | 临界状况信息，比err等级还严重 |
| alert | 警告状态信息，比crit还严重。必须立即采取行动 |
| emerg | 疼痛等级信息，系统已经无法使用了 |

日志记录位置

- 日志文件的绝对路径。如"var/log/secure"
- 系统设备文件，如"/dev/lp0"
- 转发给远程主机，如"@192.168.0.210:500"(日志服务器)
- 用户名，如"root"
- 忽略或丢弃日志，"~"

日志轮替(对于访问量很多的服务器很重要)

日志文件的命名规则

- 如果配置文件拥有"dateext"参数，那么日志会用日期来作为日志文件的后缀，例如"secure-20140709"。这样的话日志文件就不会重叠，所以也就不需要日志文件的改名，只需要保存指定的日志个数，删除多余的日志文件即可。
- 如果配置文件没有"dateext"参数，那么日志文件就需要进行改名了。当第一次进行日志轮替时，当前的"secure"日志会自动该名为"secure.1"，然后新建"secure"日志，用来保存新的日志。

logrotate.conf配置文件

```

7
8 # create new (empty) log files after rotating old ones
9 create
10
11 # use date as a suffix of the rotated file
12 dateext
13
14 # uncomment this if you want your log files compressed
15 #compress
16
17 # RPM packages drop log rotation information into this directory
18 include /etc/logrotate.d
19
20 # no packages own wtmp and btmp -- we'll rotate them here
21 /var/log/wtmp {
22     monthly
23     create 0664 root utmp
24     minsize 1M
25     rotate 1
26 }
27
28 /var/log/btmp {
29     missingok
30     monthly
31     create 0600 root utmp
32     rotate 1
33 }
34
35 # system-specific logs may be also be configured here.

```

35,1

底端

| 参数 | 参数说明 |
|----------------------------|--|
| daily | 日志的轮替周期是天 |
| weekly | 日志的轮替周期是每周 |
| monthly | 日志的轮替周期是每月 |
| rotate 数字 | 保留的日志文件的个数。0指没有备份 |
| compress | 日志轮替时，旧的日志进行压缩 |
| create mode owner group | 建立新日志，同时指定新日志的权限与所有者和所属组。如creat 0600 root utmp |
| mail address | 当日志轮替时，输出内容通过邮件发送到指定的邮件地址。如mail + 邮箱 |
| missingok | 如果日志不存在，则忽略该日志的警告信息 |
| notifempty | 如果日志为空文件，则不进行日志轮替 |

| 参数 | 参数说明 |
|------------|---|
| minsize 大小 | 日志轮替的最小值，也就是日志一定要达到这个最小值才会轮替，否则就算时间达到也不轮替 |
| size 大小 | 日志只有大于指定大小才进行日志轮替，而不是按照时间轮替。如size 100k |
| dateext | 使用日期作为日志轮替文件的后置。如secure-20130605 |

logrotate命令

- # logrotate [选项] 配置文件名
- 选项：
 - 如果此命令没有选项，则会按照配置文件中的条件进行日志轮替
 - -v :显示日志轮替过程。加了-v选项，会显示日志轮替的过程
 - -f :强制进行日志轮替。不管日志轮替的条件是否已经符合，强制配置文件中所有的日志进行轮替

启动管理

启动流程

运行级别

运行级别

| 运行级别 | 含义 |
|------|--------------------|
| 0 | 关机 |
| 1 | 单用户模式，相当于win的安全模式 |
| 2 | 不完全的命令行模式，不含有NFS服务 |
| 3 | 完全的命令模式，就是标准字符界面 |
| 4 | 系统保留 |
| 5 | 图形模式 |
| 6 | 重启动 |

运行级别命令

- # runlevel (查看运行级别命令)
- # init 运行级别 (改变运行级别命令)

系统默认运行级别

- # vim /etc/inittab
- id:3:initdefault:(系统开机直接进入哪个运行级别,centos7不是这个)

运行级别centos7

运行级别对应表

| init级别 | systemctl target |
|--------|-------------------|
| 0 | shutdown.target |
| 1 | emergency.target |
| 2 | recure.target |
| 3 | multi-user.target |
| 4 | 无 |
| 5 | graphical.target |
| 6 | 无 |

设置运行级别

- # systemctl [command] [unit.target]
- command:
 - get-default:取得当前的target
 - set-default:设置指定大的target为默认的运行级别
 - isolate:切换到指定的运行级别
 - unit.target:为上表中列出的运行级别

启动过程



备份与恢复

概述

Linux系统需要备份的数据

- /root/目录;
- /home/目录;
- /var/spool/mail/目录;
- /etc/目录;
- 其他目录;

安装服务的数据

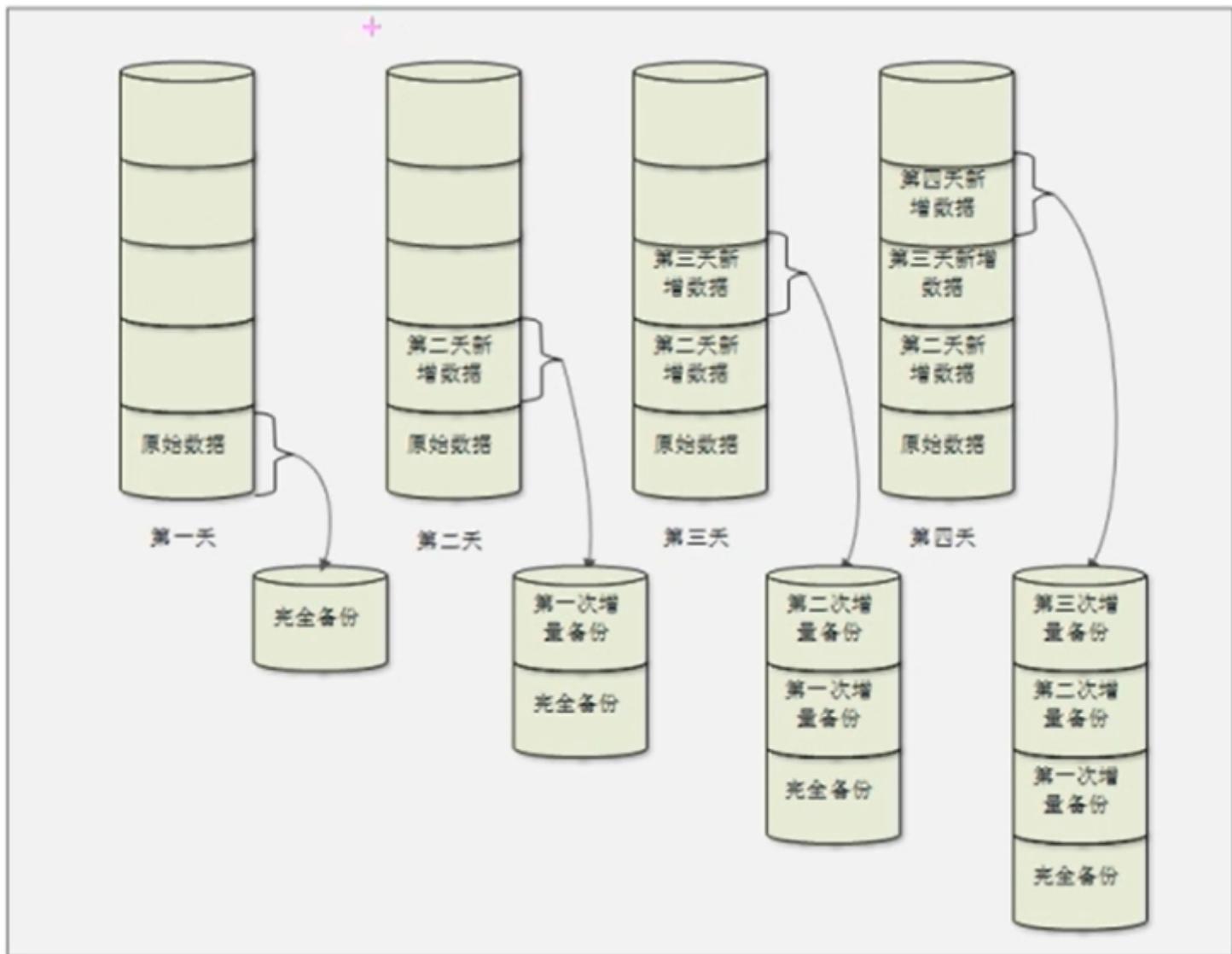
- apache 需要备份的数据
 - 配置文件
 - 网页主目录
 - 日志文件

- mysql需要备份的数据
 - 源码包安装的mysql: /usr/local/mysql/data
 - RPM包安装的mysql:/var/lib/mysql/

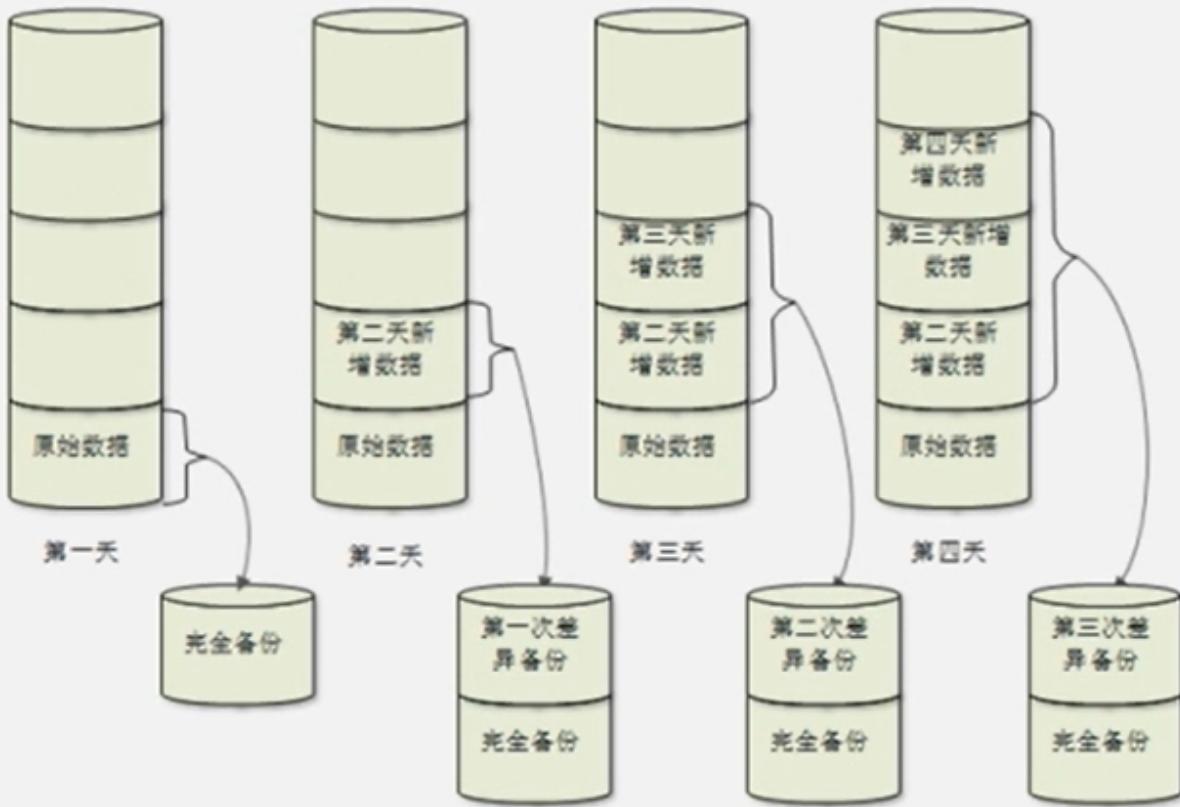
备份策略

- 完全备份: 完全备份就是指把所需要备份的数据全部备份, 当然完全备份可以备份整块硬盘, 整个分区或某个具体的目录

增量备份



差异备份



备份命令

dump命令

- # dump [选项] 备份之后的文件名 原文件或目录
- 选项:
 - -level: 就是我们说的0-9是个备份级别
 - -f 文件名: 指定备份之后的文件名
 - -u: 备份成功之后, 把备份时间记录在/etc/dumpdates文件
 - -j: 调用bzlib库压缩备份文件, 其实就是把备份文件压缩为.bz2格式
 - -W: 显示允许被dump的分区的备份等级及备份时间

备份分区

- dump -0uj -f /root/boot.bak.bz2 /boot (备份命令。先执行一次完全备份, 并压缩和更新备份时间)
- cat /etc/dumpdates (查看备份时间)
- cp install.log /boot (复制日志文件到boot分区)

- dump -1uj -f /root/boot.bak.bz2 /boot (增量备份/boot分区,并压缩 ,bak为备份文件)
- dump -W (查询分区的备份时间及备份级别)

备份文件或目录

- dump -0j -f /root/etc.dump.bz2 /etc/
- 完全备份/etc/目录，只能使用0级别进行完全备份，而不再支持增量备份

restore 命令

- # restore [模式选项] [选项]
- \模式选项：restore命令常用的模式有以下四种，者四个模式不能混用。
 - -C:比较备份数据和实际数据的变化
 - -i:进入交互模式，手工选择需要恢复的文件。
 - -t:查看模式，用于查看备份文件中拥有那些数据
 - -r:还原模式，用于数据还原
- 选项:
 - -f:指定备份文件的文件名