## 老段Linux教程笔记

**输入技巧**

1. 引用上一个命令的参数：先按 esc 然后按 **.**

2. 在输入命令时，可以通过Ctrl+p/N 查看以前输入过的命令

3. Ctrl+a ：光标插入到行首

4. Ctrl+e ：光标插入到行尾

5. ctrl+u：删除所有光标之前的字符

6. ctrl+k：删除所有光标之后的字符

7. Ctrl+w：以单词为单位删除

8. Ctrl+L = clear

**第六章 Linux的目录结构**

> 各个目录分别是什么作用的！！！

bin:二进制文件，是大部分命令程序的存放路径

boot:系统的启动文件，内核...

dev:设备文件

etc:都是配置文件

home：用户的家目录，相当于 documents

libs：连接库文件，相当于 dll文件，动态链接库

media：系统的默认挂载点，比如插入U盘所弹出的目录

mnt：也是挂载点，手动挂载点，约定成俗的

opt：源码包安装一般指定安装到此目录，默认安装在usr/ local

proc：存放内核的参数，本身不占用空间

root：超级用户的家目录

sbin：和bin类似，存放的是系统管理员能使用的命令

usr：相当于program files也就是软件的默认安装路径

var：存放的都是一些缓存，日志，数据库信息

Windows下分区是不能直接使用的，必须分配盘符

如果分配不到盘符，就只能把这个分区装到一个空的文件夹里面，访问这个文件夹就是相当于访问这个分区

所以：Linux下就只有一个分区 / 其他磁盘分区都是装在/下的空文件夹

FHS：Linux目录配置标准，几乎所有的Linux发行版都遵循这个标准

tree:查看目录的树结构

查看内核版本：

uname -a

uname -i

uname -r

还有：lsb\_release -a

也是查看所有系统信息

cat /etc/redhat-release 同样可以查看

对于相对路径可以这样写：cd ../../../

有关cd的一些细节：

1. 直接cd，进入当前用户的家目录

2. cd - 上一次访问的目录，可以方便实现两个目录之间的来回切换

3. cd ~tom/ 进入tom的家目录

**目录操作**

- 创建一个目录：mkdir dir\_name

- 可以同时创建多个目录 ：mkdir x y z

- 删除空目录：rmdir dir\_name

- 递归强制删除目录下的所有文件，包括目录：rm -fr dir\_name

- 如何递归创建目录呢？比如 x/y/z

mkdir -p x/y/z 如果父文件夹不存在，则同时创建父文件夹

在dir目录下同时创建多个文件夹：mkdir -p dir/{xx,yy,zz}

只能删除这个空文件夹和空的父文件夹：rmdir -p dir/x/y/z/

-----------------------------------------------

echo $PATH 查看当前环境变量配置，dos 下直接 path

ls --time=atime/ctime 分别表示访问时间，创建时间

ls -R /etc 递归列出

**#### 复制和剪切**

- 复制：cp [选项] 原路径 新路径

- 复制整个目录:cp -rf 源目录 目标位置

- 如何连同复制它的所有属性？

cp -p xx xx

- cp -a 相当于 cp -rfp 也就是可以复制目录，完完全全复制- 文件的内容和属性（包括创建时间，权限...)

- cp -u xx xx 始终复制最新的，如果不是最新的则不会复制

- 如果不加-u，他是会询问是否覆盖原来的

> 注意：如果复制的是链接文件，那么复制出来的将会是这个链接文件上所指向的文件的内容，然后重命名

如果就想拷贝快捷方式，可以用 -d 选项

剪切：mv [options] xxx xxx

可以重命名

其用法和cp是一样的

----------------------------

文件名：basename /boot/grub/grub.conf

目录名：dirname /boot/grub/grub.conf

[root@localhost ~]# basename /etc/sysconfig/network-scripts/ifcfg-eth0

ifcfg-eth0

[root@localhost ~]# dirname /etc/sysconfig/network-scripts/ifcfg-eth0

/etc/sysconfig/network-scripts

----------------------------

**#### 查看文件内容：**

cat 文件名 从第一行一直显示到最后一行

所以：cat只适合查看小文件

cat -n 每一行加编号

tac 文件名 与cat相反

nl 文件名 = cat -n

more/less 文件名 分屏显示文本内容

空格：一屏一屏显示，

回车：一行一行显示

more不能往前翻

less可以使用pageup、pagedown来上下翻屏

查找关键字： /keywords

然后通过按 n N 可以进行上下翻页

head 文件名 默认查看前十行

head -n 文件名 查看前n行

tail -n 文件名 默认查看尾五行

tail -f xxx 实时查看日志

tailf 文件名 = tail -f xxx

查看二进制文件呢？

od -t c /bin/

strings /bin/ls 查看二进制内部是如何写的

**创建文件：**

touch 文件名

如果文件不存在，则创建；如果存在，则把这个文件的创建时间或者修改时间改为当前时间！！！

touch -t timestramp xxx.txt 修改时间

时间戳格式为 ： 年月日时分

**#### 默认权限**

目录的权限:默认是 755

文件的权限：默认是　644

why？

umask 查看过滤的权限

umask 0022

权限过滤符

文件 666 rw- rw- rw- 文件默认是不可执行的！！！

目录 777 rwx rwx rwx

用022来进行过滤 --- -w- -w-

于是：文件变为：644 rw- r-- r--

目录变为：755 rwx rx- rx-

umask是可以改的

umask -S 查看过滤之后的实际权限

*--------------------------------*

**#### chattr**

chattr +a xxx/ 只能添加不能删除

ls attr

chattr +i xxx/ 不能添加和删除，也就是不能更改！！！

减去权限 chattr -a xxx/

chattr -i xxx/

chmod u+s xxx

表示当其他人执行xxx这个文件时具有所有者的权限

ls -l `which passwd` 查看这个命令的权限

反引号的作用：可以解释包含在里面的命令

SUID:

chmod u+s `which passwd`

比如：echo `xxx hostname xxx`

除了反引号 $(命令)也是可以的

su 或者 su - 然后输入秘密，直接切换root用户

注意：密码文件存放在/etc/shadow下，任何人都没有任何直接修改的权限

passwd命令中有 s 权限，也就是当其他用户使用该命令时，具有管理员权限，所以其他用户也可以修改自己的密码！！！

SGID:在组权限位置上有s权限，

作用：这个文件夹下面的所有内容都具有与这个文件夹相同的权限属性，比如这个文件夹属于PHP组，那么这个文件夹下面的所有都属于PHP组！！！

SBIT：只能分配给一个目录的other的权限

chmod o+t /xxx

除了所有者和root 其他人都不能删除！！！

可以用 4 2 1 很方便的给一个文件分别添加SUID,SGID,SBIT权限

可以使用file来查看文件类型，后缀不是必须要有的

------------------------------

**第八章：磁盘和分区管理**

MBR：512B

分区组成：

1.bootsector 引导扇区

2.superblock 超级块

3.iNode

4.block 格式化时，分成很多格子，每个格子有一个编号，size=4K，是真正存放文件的地方

注意：如果存有１Ｋ的文件，其余3K都不能再使用了，可以在分区时指定block大小，存放大文件会占多个block

block数目少，读取效率就高，如果文件很小就设置较小的block，这样就不会浪费很多空间

iNode会记录文件的属性信息（所有者，所属组，权限），同时会记录这个文件存放在哪个block

也就是相当于一个索引

每个iNode也有编号，每个iNode大小为128字节

iNode相当于一本书的目录

在superblock中记录的是iNode和block的信息

可以将一个很大的block划分成多个blockgroup

在blockgroup中，有superblock，文件系统的描述（比如这个blockgroup从第几个block到第几个block），块位图，iNode位图，iNode表，block

块位图：记录的是block的使用情况

iNode位图：记录iNode的使用情况

superblock：记录 ...

一个iNode是128字节

iNode中每指向一个block消耗4B，所以一共可以指向128B/4B=32个block，每个块是4k，所以是128k

即是，磁盘中一个iNode最多能指向128K的文件

实际上呢？

系统并不是直接在iNode记录block信息，而是将一个iNode划分12个iNode直接区，12个间接区，二间接区，三间接区

间接区并不存储数据，记录的是block的信息，也就是使用block来存放原来的iNode指向，这样4k/4b=1024个指向，也就是1024\*4K=4M

二间区：存储的是间接区的信息，间接区再存放...这样就能存放很多的数据了

通过计算得知：每增加一级间接区，允许的单个文件容量增大1024倍，直接128K，间接128M，二间接128G

ls -i install.log 查看节点

filefrag -v install.log 查看这个文件具体在哪个block中

dumpe2fs /dev/sda1 查看第一个分区的信息

如果是文件夹：它的iNode指向一个block，这个block存储的是这个文件夹下其他文件的iNode信息，然后指向真正存储文件的block

写文件的过程：在blockgroup中，iNode位图中找到空的iNode表->把文件属性写到iNode表->查找block 位图，找到空余的block->将数据写入到空的block中 -> 更新iNode位图和block位图信息->更新superblock信息

如果位图还没有更新，突然停电了呢？

系统会重启，修复，很漫长

还好，有日志！！！

写文件的每一步操作都会记录到日志中，开机时，系统会自动读取日志，理智修复，节约时间

文件系统ext3就比ext2多了一个日志功能

查看日志：dumpe2fs /dev/sda1

与tune2fs -l /dev/sda1 功能类似，只是上一个命令更详细！！！

可以看到有junal iNode

系统同时存在多种文件系统

VFS屏蔽了文件系统的差异 虚拟文件系统

du

df 查看已挂载文件系统

df -h 更友好的查看

df -hT 显示文件系统类型

[root@localhost ~]# df -hT

文件系统 类型 容量 已用 可用 已用% 挂载点

/dev/sda2 ext3 1.9G 674M 1.2G 37% /

/dev/sda6 ext3 487M 97M 365M 21% /var

/dev/sda3 ext3 4.8G 3.3G 1.3G 73% /usr

/dev/sda1 ext3 99M 12M 83M 13% /boot

tmpfs tmpfs 506M 0 506M 0% /dev/shm

du 文件夹 查看文件夹大小以及所有子文件夹大小

du -sh /etc/ single单一的意思

[root@localhost ~]# du -sh /root

9.9M /root

--------------------------------

连接文件：ln

软连接：就是快捷方式 ln -s a a1,给a创建a1软连接

硬链接：Linux和unix专有的概念

ln passwd a 给passwd 加一个硬链接

diff a b 查看两个文件的差异

和cp的区别呢？

硬链接就是有一个新的iNode也是指向源文件的同一个block，也就是引用，就是一个文件有两个名字

但是，硬链接是不能垮分区的，因为每个分区都有自己的iNode

每创建一个硬链接，这个文件的节点数就+1

---------------------------------

磁盘分区

fdisk -l 指定硬盘

创建分区：fdisk /dev/sdb 注意我们是对硬盘来进行分区

一个硬盘只能划分最多四个主分区

最前面是MBR区，64字节，标记一个主分区消耗16字节

分区的最小组成单位是柱面，一个柱面是8M

可以直接写 +100M

每一个分区对应一个类型，再对应一个id

扩展分区上只能划分逻辑分区

逻辑分区只能从5开始

创建好分区后，用partprobe /dev/sda 来更新分区表

格式化分区才能使用：也就是要产生block才能使用

mkfs -t ext3 /dev/sda3

mkfs -t ext3 -b 4096 /dev/sda3

也可以使用 mkfs.ext3 /dev/sda3

挂载才能使用 mount /dev/sda3 /mnt

检查磁盘：fsck

badblocks -sv

挂载，卸载 mount umount

mount -o remount,ro /mnt 以只读方式重新挂载，不能写入文件

可以使用fuser -mv /mnt 查看哪个进程在占用磁盘，杀死，即可卸载磁盘

mount /dev/sda1 /mnt 这样只是临时挂载

可以把以上命令写入到 /etc/fstab 文件中实现自动挂载

格式为：

设备名 挂载点 文件系统 默认权限default 0（是否备分） 0（是否自检fsck）

查看磁盘的参数：

tune2fs -l /dev/sda3

tune2fs可以对分区进行各种操作！！！

每个磁盘都有保留的block，是为了方便管理！！！

当mount count > maximum mount count时，每次挂载都会进行自检

可以直接通过e2lable 设备 卷标

来设置卷标，这个卷标就是代表这个设备

挂载特殊文件，比如镜像.iso

mount -o loop xx.iso /mnt

对于光盘系统的文件类型是iso9660

交换分区：很老的技术 swap

相当于Windows中的虚拟内存

查看虚拟内存 cat /proc/swaps

创建交换分区：

先用fdisk创建一个swap分区

用mkswap格式化这个分区

使用swapon /dev/sda5 激活

关闭swapoff /dev/sda5

可以把以上设置写入到/etc/fstab让开机自动挂载

创建指定大小的文件：

dd if=/dev/zero of=file bs=1M count=100

意思是：创建一个file文件，每份1m，有100份，所以这个file就是100m

dd if=/dev/zero if=file bs=1M count=100

zero 表示没有指定大小，要多少有多少，由后面的bs和count决定！！！

可以把file所占用的空间来当做交换分区

步骤：还是先格式化mkswap ，swapon

------------------------------

**第九章：压缩与解包**

压缩：gzip 文件名

查看压缩文件内容：zcat 压缩文件名

解压：gzip -d 压缩文件名

保留源文件：gzip -c hosts>hosts.gz

bzip2 xxx

bzip2 -d xxx.bz2

bzip2 -c xxx>xxx.bz2

bzcat xxx 直接查看压缩内容

zip xxx.zip xxx

unzip xxx.zip

打包：没有压缩功能，会增大体积，因为多了个袋子

tar -fvc xxx.tar xxx --remove-files

表示不保留原来的文件

查看归档文件内容：tar tfv xxx.tar

指定解档的路径，-C 路径

tar fvx xxx.tar -C /tmp

可以使用cp同时拷贝多个文件

cp a b c d .

直接解压一部分内容：

tar fvx xxx.tar a 这样就直接解压了a文件

----------------------------------

备份：

全备份

增量备份：只备份比上一天新的

差异备份：会备份上一次全备份之后更新的数据

增量备份并不好，如果要恢复，需要恢复每一天的增量才能恢复所有

实际工作中，各种备份结合使用的

全备份工具：dump -0 指明备份级别

级别范围0-9

0：完全备份

注意：只会从备份级别比他小的开始备份

0 1 2 3 4这样就是增量备份

0 9 8 7 6这样就是差异备份

比如：dump -0u -f 备份到的目录/文件名.dump /dev/sda3

restore -t -f 备份的文件 表示在不恢复的情况下查看备份的内容

练习：卸载umount，格式化一个分区mkfs.ext3 /dev/sda3，然后挂载mount，恢复restore

restore -r -f 备份文件 ： 恢复所有的文件

restore -i -f 备份文件 ：询问要恢复的文件

-----------------------------

创建光盘镜像ISO镜像，

mkisofs -o xxx.iso file1 file2 file3

通过挂载这个ISO镜像可以查看里面的内容

可以通过cp命令直接将光盘刻录成ISO镜像

cp /dev/cdrom xxx.iso 是不用挂载的

dd if=/dev/zero of=file bs=1M count=10

-----------------------------

可以使用dd命令来进行硬盘或者分区的対拷

比如：dd if=/dev/sda1 of=/dev/sdb1

把分区备份到一个文件：

dd if=/dev/sda1 of=test

卸载，格式化分区后，从文件test恢复

dd if=test of=/dev/sda1

---------------------------

Cpio备份：

find -name "file\*" | cpio -o > xxx.cpio

其中-o表示打包，-i表示解包

解包：cpio -iu < xxx.cpio

-u表示如果较新，就覆盖旧的文件

---------------------------

vim编辑器：

进入插入模式的各种对比：

I：光标跳到本行开头

A：光标调到本行结尾

O：在上一行插入空行

可以通过 / 来进入查找模式

查找替换：s/旧内容/新内容/g

说明：如果不加g则只替换当前行的第一个，有g表示替换当前行的所有

2,5s :表示替换2-5行的、

%s : 表示替换全文

3,$s : 表示从第三行到结束

.,$s : . 表示当前行

如果新内容部分为空则表示删除

行首加注释：%s/^/#/g :尖括号表示行的开头

去掉行首的注释：%s/^#//g : ^#表示行首的#

注意：规定范围的放在s前面！！！

Ctrl+r ：前进 与 u 撤销 相反

在末行模式直接输入行号即可进入具体行

命令行模式：连按两下g即可切换到首行开头

3yy表示从当前行一直往下复制3行

p表示粘贴

3dd表示剪切或删除从当前行往下3行

替换字符：先按r，再按新的字符

块选择： Ctrl+v进入可视块，移动光标即可，按y复制，p粘贴

vim aaa bbb 同时打开多个文件

切换文件：在末行模式输入files查看有什么文件

next，first，last切换文件

n,N也是可以切换的！！！

sp分屏，将一个文件分成多屏，同时编辑，Ctrl+w 上下箭头切换或者ctrl+两次w 这样会自动切换的！！！

直接退出就会退出分屏

set nu；set nonu

设置每次都显示行号：vim ~/.vimrc,在这里面写 set nu即可

/etc/sysconfig/i18n规定了系统的编码

可以使用unix2dos将Linux的文件转成Windows能识别的文件，比如换行

unix2dos -k -n file.txt

注意：单个字符的粘贴是粘贴在光标后面，一行的粘贴是粘在下一行！！！

-------------------------------

bash shell入门

shell就是一个翻译器，将我们的命令转换成CPU能看懂的0,1，命令经过CPU处理后，返回结果给shell，shell再将CPU返回的结果转换成我们人能识别的内容

终端不是shell，当我们打开终端就自动连接到一个shell

cat /etc/shells查看系统支持的shell

不同的系统使用的shell是不一样的，Linux默认使用bash shell bash

有bash csh tcsh ksh

在终端输入不同的shell即可实现切换shell

Windows远程桌面的原理就是不停的抓图，然后通过网络发送过来，如果网络不好，我们看起来就会很卡的样子

但是如果使用命令行，传输的仅仅是字符，在网络中传输很快

在shell中有大量的快捷键，大大提高管理速度

如果用which查找不出的命令很可能是shell集成的，会出现在bin sbin目录下的命令是安装软件而生成的！！！

可以使用type 命令 来查看一个命令的类型

比如：type source

source is a shell builtin

如果在同一行输入多个命令可以用分号隔开，如果一个命令太长可以使用\回车，转到下一行

------------------------------

变量：

变量名=值 中间不能有空格，不能以数字开头

本地变量：只影响当前shell

可用echo $$ 查看shell id

xx=1

echo $xx

取消变量：unset xx

[root@localhost ~]# echo $$

4781

[root@localhost ~]# ksh

# echo $$

4805

---------------------------------

环境变量：会影响到所有子shell

export xx=2

echo $xx

直接 set 会显示所有变量

显示环境变量 ： env 或者 export

declare也可以用来定义环境变量

declare -x aa=2

取消：declare +x aa

------------------------------

位置变量：

在shell脚本中$1,$2,$3分别表示第一，第二，第三个参数，也就是运行.sh 后面传入来的参数

比如：

#!/bin/bash

echo "这个程序的名字是：$0"

echo "这个程序的第一个参数是：$1"

echo "这个程序的第2个参数是：$2"

echo "这个程序的第3个参数是：$3"

echo "这个程序的第4个参数是：$4"

echo "一共有$#个参数，分别是：$\*"

可以使用 set a b c d 设置变量

echo $1

echo ${10} 两位数以上的就要用{}包起来，否则shell会理解错！！！

运行结果：

[root@localhost ~]# ./xx.sh a b c d

这个程序的名字是:./xx.sh

这个程序的第一个参数是：a

这个程序的第2个参数是：b

这个程序的第3个参数是：c

这个程序的第4个参数是：d

一共有4个参数，分别是：a b c d

---------------------------------

系统变量：

$HOME

$PATH

$USER当前登录的用户

$HISTFILESIZE

$LANG当前系统使用的编码

系统在我们登录或者切换用户时，这些系统变量就自动赋值了！！！

[root@localhost ~]# echo $PS1

[\u@\h \W]\$

\w表示全路径 \W只显示最后一级路径！！！

分析：用户@短主机名 工作目录

hostname -s 短主机名

echo PS2 就是回车哪个符号

可以自定义 ： PS2="--->"

locale:查看系统支持的编码

[root@localhost ~]# locale

LANG=zh\_CN.UTF-8

LC\_CTYPE="zh\_CN.UTF-8"

LC\_NUMERIC="zh\_CN.UTF-8"

------------------------------

read：读取用户的输入，并赋给一个变量$REPLY

用法：一般用于脚本中

echo "请输入你的名字："

read name

echo "欢迎您$name"

也可以写成一行：

read -p "请输入你的名字：" read

echo "欢迎您$REPLY"

---------------------------

数组：下标从0开始！！！

array=(a b c d)注意这里是没有逗号的！！！

echo ${array[1]}

输入 echo ${array[@/\*]}

显示当前数组所有元素！！！

查看下标：${!array[@]}

查看有几个元素：${#array[@]}

--------------------------------

设置一个环境变量只读：

declare -x a=3

declare -r a

[root@localhost ~]# declare -x a=3

[root@localhost ~]# declare -r a

[root@localhost ~]# a=2

bash: a: readonly variable

或者使用 readonly a

问题：如何销毁只读变量？

双引号里面的变量会正常解释，单引号里面的则不会解释

建议用${var}来表示变量

文件大小的限制

ulimit -f 20480 这样限制了以后就不能用dd创建>20M的文件了！！！

查看 ： ulimit -a

**#### 第十五章 磁盘配额 Quota**

概念：就是限制某个用户可以使用多大的磁盘空间，超过了就会存不进去

软限制：达到指定大小时，有一个提醒，还可以继续存放一点

硬限制：达到指定大小，无论如何都存不进去了

默认系统是没有磁盘配额的

为了更好的学习，我们在安装 centos 时 ，可以指定分区大小，然后预留 2G 的空间用于以后的分区测试！！！

安装过程是否许需要联网？

答：好像不需要的哦。

i386 与 i686 有什么区别？

答：I686 只是i386的一个子集,

**### 第十七章 进程管理和SELinux初探**

进程（process）：

程序（program）：

进程是静态的，程序是动态的。

Linux是多用户和多任务的系统。

同时可以运行多个任务，可以多个用户同时登陆。Windows不算是真正的多用户操作系统！！！

也就是Linux有多个虚拟控制台，而Windows只有一个控制台，同一时间只允许一个用户登陆控制台。

多用户登陆的好处：

cat 、/dev/zero > /dev/null #相当于是一个死循环

如果执行以上任务，当前控制台就会挂掉，于是我们就可以进入其他的控制台关掉这个进程。

firefox # 打开火狐浏览器 ，此时终端被前台占用

按 Ctrl+z ：停止当前程序，把它放到后台，腾出控制台，再输入 jobs命令即可查看后台进程。

bg 数字 ：激活相应的后台进程。此时进程的 状态由 stopped变成running。

firefox & ：一开始就把进程放到后台运行并且不占用当前控制台。

kill -9 %1 : 强制删除后台的第一个进程，%表示后台进程，如果没有%则是杀死pid为1的进程。

远程管理：nohup ping localhost &

这样，即使关掉终端，ping这个程序还是会在后台运行

ps aux | grep ping : 查看ping进程

ps aux : 显示系统的所有进程，包括所有的shell

也可以使用 ps ex 来实现同样的功能。

PPID： 父进程 也就是进程可以产生其他的进程