链滴

Linux 学习总结（一）常用基本命令

作者：wlgzs-sjl

原文链接：https://ld246.com/article/1604546960423

来源网站：链滴

关机操作

在linux领域内大多用在服务器上，很少遇到关机的操作。毕竟服务器上跑一个服务是永无止境的，除特殊情况下，不得已才会关机。

关机指令为：shutdown ；

sync # 将数据由内存同步到硬盘中。

shutdown # 关机指令，你可以man shutdown 来看一下帮助文档。例如你可以运行如下命令关机：

shutdown –h 10 # 这个命令告诉大家，计算机将在10分钟后关机

shutdown –h now # 立马关机

shutdown –h 20:25 # 系统会在今天20:25关机

shutdown –h +10 # 十分钟后关机

shutdown –r now # 系统立马重启

shutdown –r +10 # 系统十分钟后重启

reboot # 就是重启，等同于 shutdown –r now

halt # 关闭系统，等同于shutdown –h now 和 poweroff

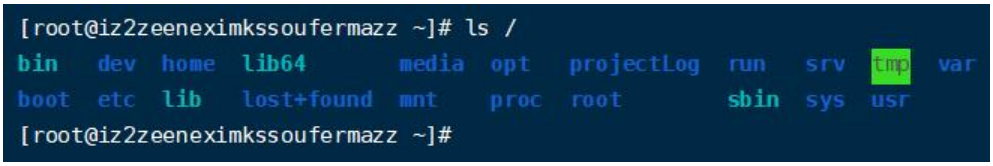
最后总结一下，不管是重启系统还是关闭系统，首先要运行 **sync** 命令，把内存中的数据写到磁盘中。

系统目录结构

登录系统后，在当前命令窗口下输入命令：

```
ls /
```

你会看到如下图所示：



以下是对这些目录的解释：

- **/bin**：bin是Binary的缩写, 这个目录存放着最经常使用的命令。
- **/boot：** 这里存放的是启动Linux时使用的一些核心文件，包括一些连接文件以及镜像文件。
- **/dev ：** dev是Device(设备)的缩写, 存放的是Linux的外部设备，在Linux中访问设备的方式和访问件的方式是相同的。
- **/etc：** 这个目录用来存放所有的系统管理所需要的配置文件和子目录。
- **/home**：用户的主目录，在Linux中，每个用户都有一个自己的目录，一般该目录名是以用户的账命名的。
- **/lib**：这个目录里存放着系统最基本的动态连接共享库，其作用类似于Windows里的DLL文件。
- **/lost+found**：这个目录一般情况下是空的，当系统非法关机后，这里就存放了一些文件。
- **/media**：linux系统会自动识别一些设备，例如U盘、光驱等等，当识别后，linux会把识别的设备载到这个目录下。
- **/mnt**：系统提供该目录是为了让用户临时挂载别的文件系统的，我们可以将光驱挂载在/mnt/上，后进入该目录就可以查看光驱里的内容了。
- **/opt**：这是给主机额外安装软件所摆放的目录。比如你安装一个ORACLE数据库则就可以放到这个录下。默认是空的。
- **/proc**：这个目录是一个虚拟的目录，它是系统内存的映射，我们可以通过直接访问这个目录来获系统信息。
- **/root**：该目录为系统管理员，也称作超级权限者的用户主目录。
- **/sbin**：s就是Super User的意思，这里存放的是系统管理员使用的系统管理程序。
- **/srv**：该目录存放一些服务启动之后需要提取的数据。
- **/sys**：这是linux2.6内核的一个很大的变化。该目录下安装了2.6内核中新出现的一个文件系统 sysfs 。
- **/tmp**：这个目录是用来存放一些临时文件的。
- **/usr**：这是一个非常重要的目录，用户的很多应用程序和文件都放在这个目录下，类似于windows的program files目录。
- **/usr/bin：** 系统用户使用的应用程序。
- **/usr/sbin：** 超级用户使用的比较高级的管理程序和系统守护程序。
- **/usr/src：** 内核源代码默认的放置目录。
- **/var**：这个目录中存放着在不断扩充着的东西，我们习惯将那些经常被修改的目录放在这个目录下包括各种日志文件。
- **/run**：是一个临时文件系统，存储系统启动以来的信息。当系统重启时，这个目录下的文件应该被掉或清除。

目录管理

绝对路径和相对路径

绝对路径：

路径的写法，由根目录 / 写起，例如：/usr/share/doc 这个目录。

相对路径：

路径的写法，不是由 / 写起，例如由 /usr/share/doc 要到 /usr/share/man 底下时，可以写成：cd .. man

处理目录的常用命令

- ls: 列出目录
- cd：切换目录
- pwd：显示目前的目录
- mkdir：创建一个新的目录
- rmdir：删除一个空的目录
- cp: 复制文件或目录
- rm: 移除文件或目录
- mv: 移动文件与目录，或修改文件与目录的名称

你可以使用 man [命令] 来查看各个命令的使用文档，如 ： man cp。

ls （列出目录）

在Linux系统当中， ls 命令可能是最常被运行的。

语法：

[root@iz2zeeneximkssoufermazz ~]# ls [-aAdfFhilnrRSt] 目录名称

选项与参数：

- -a ： 全部的文件，连同隐藏文件(开头为 . 的文件) 一起列出来(常用)
- -l ： 长数据串列出，包含文件的属性与权限等等数据；(常用)

将目录下的所有文件列出来(含属性与隐藏档)

[root@iz2zeeneximkssoufermazz ~]# ls -al ~

cd （切换目录）

cd是Change Directory的缩写，这是用来变换工作目录的命令。

语法：

cd [相对路径或绝对路径]

pwd (显示目前所在的目录)

pwd 是 **Print Working Directory** 的缩写，也就是显示目前所在目录的命令。

[root@iz2zeeneximkssoufermazz home]#pwd [-P]

选项与参数：**-P** ： 显示出确实的路径，而非使用连接(link) 路径。

测试：

```
# 如果是链接，要显示真实地址，可以使用 -P参数
[root@iz2zeeneximkssoufermazz bin]# pwd
/bin
[root@iz2zeeneximkssoufermazz bin]# pwd -P
/usr/bin
```

mkdir （创建新目录）

如果想要创建新的目录的话，那么就使用mkdir (make directory)

mkdir [-mp] 目录名称

选项与参数：

- -m ： 配置文件的权限,直接配置，不需要默认权限 (umask) 。
- -p ： 直接将所需要的目录(包含上一级目录)递归创建。

测试：

```
# 进入我们用户目录下
[root@iz2zeeneximkssoufermazz ~]# cd /home

# 创建一个 test 文件夹
[root@iz2zeeneximkssoufermazz home]# mkdir test

# 加了这个 -p 的选项，可以创建多层目录
[root@iz2zeeneximkssoufermazz home]# mkdir -p test1/test2/test3

# 创建权限为 rwx--x--x 的目录。
[root@iz2zeeneximkssoufermazz home]# mkdir -m 711 test2
[root@iz2zeeneximkssoufermazz home]# ls -l
drwxr-xr-x 2 root root  4096 Mar 12 21:55 test
drwxr-xr-x 3 root root  4096 Mar 12 21:56 test1
drwx--x--x 2 root root  4096 Mar 12 21:58 test2
```

rmdir (删除空的目录)

语法：

rmdir [-p] 目录名称

选项与参数：**-p ： **连同上一级『空的』目录也一起删除

测试：

```
# 看看有多少目录存在?
[root@iz2zeeneximkssoufermazz home]# ls -l
drwxr-xr-x 2 root root  4096 Mar 12 21:55 test
drwxr-xr-x 3 root root  4096 Mar 12 21:56 test1
drwx--x--x 2 root root  4096 Mar 12 21:58 test2

# 可直接删除掉，没问题
[root@iz2zeeneximkssoufermazz home]# rmdir test

# 因为尚有内容，所以无法删除!
[root@iz2zeeneximkssoufermazz home]# rmdir test1
rmdir: failed to remove ‘test1’: Directory not empty

# 利用 -p 这个选项，立刻就可以将 test1/test2/test3/test4 依次删除。
[root@iz2zeeneximkssoufermazz home]# rmdir -p test1/test2/test3
```

注意：这个 rmdir 仅能删除空的目录，可以使用 rm 命令来删除非空目录

cp (复制文件或目录)

语法：

```
[root@iz2zeeneximkssoufermazz ~]# cp [-adfilprsu] 来源档(source) 目标档(destination)
[root@iz2zeeneximkssoufermazz ~]# cp [options] source1 source2 source3 .... directory
```

选项与参数：

- -a：相当于 -pdr 的意思，至于 pdr 请参考下列说明；(常用)
- -p：连同文件的属性一起复制过去，而非使用默认属性(备份常用)；
- -d：若来源档为连结档的属性(link file)，则复制连结档属性而非文件本身；
- -r：递归持续复制，用于目录的复制行为；(常用)
- -f：为强制(force)的意思，若目标文件已经存在且无法开启，则移除后再尝试一次；
- -i：若目标档(destination)已经存在时，在覆盖时会先询问动作的进行(常用)
- -l：进行硬式连结(hard link)的连结档创建，而非复制文件本身。
- -s：复制成为符号连结档 (symbolic link)，亦即『捷径』文件；
- -u：若 destination 比 source 旧才升级 destination ！

测试：

```
# 找一个有文件的目录，我这里找到 root目录
[root@iz2zeeneximkssoufermazz home]# cd /root
[root@iz2zeeneximkssoufermazz ~]# ls
install.sh
[root@iz2zeeneximkssoufermazz ~]# cd /home

# 复制 root目录下的install.sh 到 home目录下
[root@iz2zeeneximkssoufermazz home]# cp /root/install.sh /home
[root@iz2zeeneximkssoufermazz home]# ls
install.sh

# 再次复制，加上-i参数，增加覆盖询问?
```

```
[root@iz2zeeneximkssoufermazz home]# cp -i /root/install.sh /home
cp: overwrite ‘/home/install.sh’ ? y # n不覆盖，y为覆盖
```

rm（移除文件或目录）

语法：

```
rm [-fir] 文件或目录
```

选项与参数：

- -f：就是 force 的意思，忽略不存在的文件，不会出现警告信息；
- -i：互动模式，在删除前会询问使用者是否动作
- -r：递归删除，最常用在目录的删除，这是非常危险的选项！！！

测试：

```
# 将刚刚在 cp 的实例中创建的 install.sh删除掉
[root@iz2zeeneximkssoufermazz home]# rm -i install.sh
rm: remove regular file ‘install.sh’ ? y
# 如果加上 -i 的选项就会主动询问，避免删除到错误的档名

# 尽量不要在服务器上使用 rm -rf /
```

mv（移动文件与目录，或修改名称）

语法：

```
[root@iz2zeeneximkssoufermazz ~]# mv [-fiu] source destination
[root@iz2zeeneximkssoufermazz ~]# mv [options] source1 source2 source3 .... directory
```

选项与参数：

- -f：force 强制的意思，如果目标文件已经存在，不会询问而直接覆盖；
- -i：若目标文件 (destination) 已经存在时，就会询问是否覆盖！
- -u：若目标文件已经存在，且 source 比较新，才会升级 (update)

测试：

```
# 复制一个文件到当前目录
[root@iz2zeeneximkssoufermazz home]# cp /root/install.sh /home

# 创建一个文件夹 test
[root@iz2zeeneximkssoufermazz home]# mkdir test

# 将复制过来的文件移动到我们创建的目录，并查看
[root@iz2zeeneximkssoufermazz home]# mv install.sh test
[root@iz2zeeneximkssoufermazz home]# ls
test
[root@iz2zeeneximkssoufermazz home]# cd test
[root@iz2zeeneximkssoufermazz test]# ls
install.sh
```

```
# 将文件夹重命名，然后再次查看！
[root@iz2zeeneximkssoufermazz test]# cd ..
[root@iz2zeeneximkssoufermazz home]# mv test mvtest
[root@iz2zeeneximkssoufermazz home]# ls
mvtest
```

基本属性

看懂文件属性

Linux系统是一种典型的多用户系统，不同的用户处于不同的地位，拥有不同的权限。为了保护系统安全性，Linux系统对不同的用户访问同一文件（包括目录文件）的权限做了不同的规定。

在Linux中我们可以使用 ll或者 ls –l命令来显示一个文件的属性以及文件所属的用户和组，如：

```
[root@iz2zeeneximkssoufermazz /]# ls -l
总用量 64
lrwxrwxrwx    1 root root     7 8月   18 2019 bin -> usr/bin
dr-xr-xr-x.   5 root root  4096 8月   18 2019 boot
drwxr-xr-x   19 root root  2980 5月   27 00:21 dev
drwxr-xr-x.  82 root root  4096 8月   19 2019 etc
drwxr-xr-x.   6 root root  4096 11月   5 10:58 home
lrwxrwxrwx    1 root root     7 8月   18 2019 lib -> usr/lib
lrwxrwxrwx    1 root root     9 8月   18 2019 lib64 -> usr/lib64
drwx------.   2 root root 16384 8月   18 2017 lost+found
drwxr-xr-x.   2 root root  4096 4月   11 2018 media
drwxr-xr-x.   2 root root  4096 4月   11 2018 mnt
drwxr-xr-x.   4 root root  4096 1月   26 2020 opt
dr-xr-xr-x  102 root root     0 8月   18 2019 proc
drwxr-x---    3 root root  4096 8月   19 2019 projectLog
dr-xr-x---.   8 root root  4096 11月   3 17:35 root
drwxr-xr-x   29 root root   900 12月 21 2019 run
lrwxrwxrwx    1 root root     8 8月   18 2019 sbin -> usr/sbin
drwxr-xr-x.   2 root root  4096 4月   11 2018 srv
dr-xr-xr-x   13 root root     0 8月   18 2019 sys
drwxrwxrwt.   9 root root  4096 11月   5 03:37 tmp
drwxr-xr-x.  14 root root  4096 8月   18 2019 usr
drwxr-xr-x.  19 root root  4096 8月   18 2019 var
```

实例中，boot文件的第一个属性用"d"表示。"d"在Linux中代表该文件是一个目录文件。

在Linux中第一个字符代表这个文件是目录、文件或链接文件等等：

- 当为[**d**]则是目录
- 当为[**-**]则是文件；
- 若是[**l**]则表示为链接文档 (link file)；
- 若是[**b**]则表示为装置文件里面的可供储存的接口设备 (可随机存取装置)；
- 若是[**c**]则表示为装置文件里面的串行端口设备，例如键盘、鼠标 (一次性读取装置)。

接下来的字符中，以三个为一组，且均为『rwx』 的三个参数的组合。

其中，[r]代表可读(read)、[w]代表可写(write)、[x]代表可执行(execute)。

要注意的是，这三个权限的位置不会改变，如果没有权限，就会出现减号[-]而已。

对于文件来说，它都有一个特定的所有者，也就是对该文件具有所有权的用户。

同时，在Linux系统中，用户是按组分类的，一个用户属于一个或多个组。

文件所有者以外的用户又可以分为文件所有者的同组用户和其他用户。

因此，Linux系统按文件所有者、文件所有者同组用户和其他用户来规定了不同的文件访问权限。

在以上实例中，boot 文件是一个目录文件，属主和属组都为 root。

修改文件属性

1、chgrp：更改文件属组

chgrp [-R] 属组名 文件名

-R：递归更改文件属组，就是在更改某个目录文件的属组时，如果加上-R的参数，那么该目录下的所文件的属组都会更改。

2、chown：更改文件属主，也可以同时更改文件属组

chown [–R] 属主名 文件名
chown [-R] 属主名：属组名 文件名

3、chmod：更改文件9个属性

chmod [-R] xyz 文件或目录

Linux文件属性有两种设置方法，一种是数字，一种是符号。

Linux文件的基本权限就有九个，分别是owner/group/others三种身份各有自己的read/write/execue权限。

先复习一下刚刚上面提到的数据：文件的权限字符为：『-rwxrwxrwx』， 这九个权限是三个三个一的！其中，我们可以使用数字来代表各个权限，各权限的分数对照表如下：

r:4　　w:2　　x:1

每种身份(owner/group/others)各自的三个权限(r/w/x)分数是需要累加的，例如当权限为：[-rwxrwx--] 分数则是：

- owner = rwx = 4+2+1 = 7
- group = rwx = 4+2+1 = 7
- others= --- = 0+0+0 = 0

chmod 770 filename

文件内容查看

概述

Linux系统中使用以下命令来查看文件的内容：

- cat 由第一行开始显示文件内容
- tac 从最后一行开始显示，可以看出 tac 是 cat 的倒着写!
- nl 显示的时候，顺道输出行号
- more 一页一页的显示文件内容
- less 与 more 类似，但是比 more 更好的是，他可以往前翻页!
- head 只看头几行
- tail 只看尾巴几行

你可以使用**man [命令]**来查看各个命令的使用文档，如 ： man cp。

cat 由第一行开始显示文件内容

语法:

```
cat [-AbEnTv]
```

选项与参数:

- -A ： 相当于 -vET 的整合选项，可列出一些特殊字符而不是空白而已;
- -b ： 列出行号，仅针对非空白行做行号显示，空白行不标行号;
- -E ： 将结尾的断行字节 $ 显示出来;
- -n ： 列印出行号，连同空白行也会有行号，与 -b 的选项不同;
- -T ： 将 [tab] 按键以 ^I 显示出来;
- -v ： 列出一些看不出来的特殊字符

nl 显示行号

语法:

```
nl [-bnw] 文件
```

选项与参数:

- -b ： 指定行号指定的方式，主要有两种：-b a ： 表示不论是否为空行，也同样列出行号(类似 cat -n ; -b t ： 如果有空行，空的那一行不要列出行号(默认值);
- -n ： 列出行号表示的方法，主要有三种：-n ln ： 行号在荧幕的最左方显示; -n rn ： 行号在自己位的最右方显示，且不加 0 ； -n rz ： 行号在自己栏位的最右方显示，且加 0 ;
- -w ： 行号栏位的占用的位数。

more 一页一页翻动

在 more 这个程序的运行过程中，你有几个按键可以按的:

- 空白键 (space)：代表向下翻一页;
- Enter   ： 代表向下翻『一行』;
- /字串   ： 代表在这个显示的内容当中，向下搜寻『字串』这个关键字;
- :f   ： 立刻显示出档名以及目前显示的行数;

- q　　：代表立刻离开 more，不再显示该文件内容。
- b 或 [ctrl]-b：代表往回翻页，不过这动作只对文件有用，对管线无用。

```
[root@iz2zeeneximkssoufermazz etc]# more /etc/csh.login
....(中间省略)....
--More--(28%) # 重点在这一行，你的光标也会在这里等待你的命令
```

less　一页一页翻动，以下实例输出/etc/man.config文件的内容：

less运行时可以输入的命令有：

- 空白键　：向下翻动一页；
- [pagedown]：向下翻动一页；
- [pageup]：向上翻动一页；
- /字串　：向下搜寻『字串』的功能；
- ?字串　：向上搜寻『字串』的功能；
- n　　：重复前一个搜寻 (与 / 或 ? 有关！)
- N　　：反向的重复前一个搜寻 (与 / 或 ? 有关！)
- q　　：离开 less 这个程序；

```
[root@iz2zeeneximkssoufermazz etc]# more /etc/csh.login
....(中间省略)....
:　# 这里可以等待你输入命令！
```

head　取出文件前面几行

语法：

```
head [-n number] 文件
```

选项与参数：**-n** 后面接数字，代表显示几行的意思！

tail　取出文件后面几行

语法：

```
tail [-n number] 文件
```

选项与参数：

- -n：后面接数字，代表显示几行的意思

拓展：Linux 链接概念

Linux 链接分两种，一种被称为硬链接（Hard Link），另一种被称为符号链接（Symbolic Link）。

情况下，**ln** 命令产生硬链接。

硬连接

硬连接指通过索引节点来进行连接。在 Linux 的文件系统中，保存在磁盘分区中的文件不管是什么类

都给它分配一个编号，称为索引节点号(Inode Index)。在 Linux 中，多个文件名指向同一索引节点存在的。比如：A 是 B 的硬链接（A 和 B 都是文件名），则 A 的目录项中的 inode 节点号与 B 的目项中的 inode 节点号相同，即一个 inode 节点对应两个不同的文件名，两个文件名指向同一个文件，和 B 对文件系统来说是完全平等的。删除其中任何一个都不会影响另外一个的访问。

硬连接的作用是允许一个文件拥有多个有效路径名，这样用户就可以建立硬连接到重要文件，以防止误删”的功能。其原因如上所述，因为对应该目录的索引节点有一个以上的连接。只删除一个连接并影响索引节点本身和其它的连接，只有当最后一个连接被删除后，文件的数据块及目录的连接才会被放。也就是说，文件真正删除的条件是与之相关的所有硬连接文件均被删除。

软连接

另外一种连接称之为符号连接（Symbolic Link），也叫软连接。软链接文件有类似于 Windows 的捷方式。它实际上是一个特殊的文件。在符号连接中，文件实际上是一个文本文件，其中包含的有另文件的位置信息。比如：A 是 B 的软链接（A 和 B 都是文件名），A 的目录项中的 inode 节点号与 B 的目录项中的 inode 节点号不相同，A 和 B 指向的是两个不同的 inode，继而指向两块不同的数据。但是 A 的数据块中存放的只是 B 的路径名（可以根据这个找到 B 的目录项）。A 和 B 之间是“主”关系，如果 B 被删除了，A 仍然存在（因为两个是不同的文件），但指向的是一个无效的链接。

测试：

```
[root@iz2zeeneximkssoufermazz /]# cd /home
[root@iz2zeeneximkssoufermazz home]# touch f1 # 创建一个测试文件f1
[root@iz2zeeneximkssoufermazz home]# ls
f1
[root@iz2zeeneximkssoufermazz home]# ln f1 f2     # 创建f1的一个硬连接文件f2
[root@iz2zeeneximkssoufermazz home]# ln -s f1 f3   # 创建f1的一个符号连接文件f3
[root@iz2zeeneximkssoufermazz home]# ls -li       # -i参数显示文件的inode节点信息
397247 -rw-r--r-- 2 root root     0 Mar 13 00:50 f1
397247 -rw-r--r-- 2 root root     0 Mar 13 00:50 f2
397248 lrwxrwxrwx 1 root root     2 Mar 13 00:50 f3 -> f1
```

从上面的结果中可以看出，硬连接文件 f2 与原文件 f1 的 inode 节点相同，均为 397247，然而符号接文件的 inode 节点不同。

```
# echo 字符串输出 >> f1 输出到 f1文件
[root@iz2zeeneximkssoufermazz home]# echo "I am f1 file" >>f1
[root@iz2zeeneximkssoufermazz home]# cat f1
I am f1 file
[root@iz2zeeneximkssoufermazz home]# cat f2
I am f1 file
[root@iz2zeeneximkssoufermazz home]# cat f3
I am f1 file
[root@iz2zeeneximkssoufermazz home]# rm -f f1
[root@iz2zeeneximkssoufermazz home]# cat f2
I am f1 file
[root@iz2zeeneximkssoufermazz home]# cat f3
cat: f3: No such file or directory
```

通过上面的测试可以看出：当删除原始文件 f1 后，硬连接 f2 不受影响，但是符号连接 f1 文件无效；

依此可以做一些相关的测试，可以得到以下全部结论：

- 删除符号连接f3,对f1,f2无影响；
- 删除硬连接f2，对f1,f3也无影响；

- 删除原文件f1，对硬连接f2没有影响，导致符号连接f3失效；
- 同时删除原文件f1,硬连接f2，整个文件会真正的被删除。

写在最后

学习内容以及总结参考B站狂神说Javahttps://www.bilibili.com/video/BV187411y7hF