## 根目录下的常见目录

/bin: 存放二进制可执行文件(系统命令)

/dev: 存放 硬件设备所对应的文件（鼠标之类）

/etc: 存放 系统配置文件

/home: 家目录

/peoc: 存放系统核心，内存中的数据

/tmp: 存放用户程序工作期间产生的临时文件

/var: 存放系统工作期间产生的 经常变化的文件。（日志、数据库文件）

/root: root用户 家目录

/lib: 存放系统库

usr: 存放用户程序 相关 资源目录

/usr/lib：用户库

/usr/include/ 头文件

## 常用命令

### man手册的使用

向下翻页：空格

上下翻页：PGUP、PGON

上下翻动：上下方向键

回到头部：Home

回到尾部：END

向下搜索：/

向上搜索：？

搜索上一个、下一个：n、N

退出：q

### ls

参数：

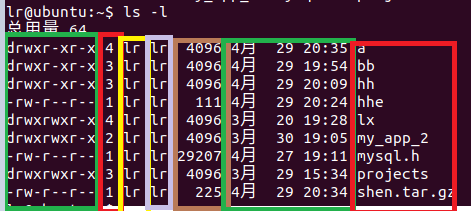
-l: 显示详细信息

-a: 显示所有文件，包括隐藏

-h: 显示人类可读信息，带有 单位 信息

-i: 显示文件的inode节点

命令 ls -l 解读



1、文件类型和权限

a.文件的类型，第一个字符表示

(1). 普通文件 - 占用磁盘存储。

(2). 目录文件 d 占用磁盘存储。

(3). 软链接文件 l 占用磁盘存储。

(4). 块设备文件 b 伪文件，不占用磁盘存储。

(5). 字符设备文件 c 伪文件，不占用磁盘存储。

(6). 管道文件 p 伪文件，不占用磁盘存储。

(7). 套接字 s 伪文件，不占用磁盘存储。

(8). 未知文件

b.文件的权限，后面的字符表示

(1).r表示可读、 w表示可写、 x表示可执行

(2).前三个所有者权限、中三同组权限、后三其他人权限

2、硬连接数

3、文件所有者

4、文件所有组

5、文件大小（单位字节）（4k 的整数倍）

6、时间 文件首次创建的、最后一次修改的时间

7、文件名

### shell 命令使用技巧

history：列出过往执行过的命令

使用 ！+ 命令编号，重新执行该命令

Ctrl+a： 将光标移至最前端。

Ctrl+e： 将光标移至最末端。

Ctrl+u： 清空已输入的命令。

### 创建删除文件、目录

创建文件语法：touch 文件名

创建目录语法：mkdir 目录名

参数：-p 创建多层目录

删除文件、目录语法：rm 文件、目录名

参数：-r 递归删除

-I 询问、交互式删除

### 软、硬链接

硬连接语法：ln 源文件 硬链接文件

硬连接实现原理：源文件和硬链接文件之间有相同的inode节点 指向同一块扇区。

注意：不能给目录创建硬连接。目录的硬连接来自”.”和”..”

软连接语法：ln -s 源文件 软链接文件

注意：

1、软连接可以使用相对路径和绝对路径

2、可以给目录创建软连接

### cp

拷贝语法：cp 源文件 新文件

参数：-r 递归拷贝

### mv

移动语法：mv 源文件 新位置、新文件名

### which命令所在的位置

查找命令语法：which 需要查询的命令

### | 管道

将前一个命令的输出，作为后一个命令的输入。

语法：ps aux | grep xxx 从 系统所有的进程中，过滤出 xxx 进程。

### >、>>重定向

重定向，将前一个命令的输出，重定向到其他地方

>: 文件不存在创建,文件存在覆盖

>>: 文件不存在创建,文件存在追加

### cat、tac、more、head、tail

查看文件语法：cat 文件名

逆序查看文件语法：tac 文件名

-n：每行加编号

查看大文件语法：less 文件名

显示前10行语法：head head -10

显示后10行语法：tail tail -10

### 压缩、解压缩

压缩语法： tar -zcvf 压缩包名.tar.gz 压缩源材料

tar -jcvf 压缩包名.tar.bz2 压缩源材料

解压缩语法：tar -zxvf 压缩包名.tar.gz -C 解压到什么路径

tar -jxvf 压缩包名.tar.bz2

参数含义：

z: 使用gzip格式压缩

j: 使用bzip2 格式压缩

c: 创建压缩包（多个文件、只生成一个压缩包）

x：解压缩

v: 显示过程

f: 指定压缩包的名称。参数通常放在后面

### terr树状态目录显示

语法：terr录名

### ps 命令

语法：ps -aux

参数a：显示终端上的所有进程，包括其他用户的进程

参数u：显示所有进程的详细状态

参数x：显示没有控制终端的进程

使用管道过滤需要的进程：

ps aux | grep xxx

top命令动态进程管理(了解)

### kill发送信号命令

语法：kill -9 进程PID

注意：

1、9号信号的名称SIGKILL

2、查看所有信号：kill -l

### 用户管理

添加用户

语法：adduser 用户名

修改用户密码：

语法：passwd 用户名

给用户管理员权限

chmod -v u+w /etc/sudoers

vim /etc/sudoers

找到 root ALL=(ALL) ALL

下面添加 用户名 ALL=(ALL) ALL

chmod -v u-w /etc/sudoers

### 权限管理

修改文件拥有者

语法：chown 用户名 需要修改的文件、目录名

修改文件所有组

语法：chgrp 组名 需要修改文件/目录名

修改文件权限

语法：chmod 777 文件名

7表示：读(r) 4

写(w) 2

执行(x) 1

第一个数字表示：所有者权限

第二个数字表示：所有组权限

第三个数字表示：其他人权限

权限对于目录和文件的不同作用

r对于文件：可以查看文件内容。cat、less、head

w对于文件：可以修改、删除文件内容。vim

x对于文件：可以被运行。./aout

r对于目录：可以被查看。ls、tree

w对于目录：可以被曾、删、改。mkdir、rm、mv、touch

x对于目录：可以被进入。cd

### find按文件属性查找

按名称找的语法：find 需要查找的目录 -name “查找内容”

通配符：

1、\* 表示零到多个任意字符

2、? 表示一个任意字符

按类型找的语法：find 需要查找的目录 -type “查找的类型”

类型：

(1). 普通文件 f

(2). 目录文件 d

(3). 软链接文件 l

(4). 块设备文件 b

(5). 字符设备文件 c

(6). 管道文件 p

(7). 套接字 s

按文件大小找的语法：find 需要查找的目录 -size “+100M” -size “-1000M”

注意：

1、+100M 表示查找大于100M的文件

2、-1000M 表示查找小于1000M的文件

3、不写单位单位的情况下默认是字节，其他单位k(小写)、M、G

指定搜索层级深度：-maxdepth 1，放在目录后面，数字表示层级

语法：find 需要查找的目录 -maxdepth 1 -name “查找内容”

对查询结果执行命令：-exec，放在结尾

语法：find 需要查找的目录 -name “查找内容” -exec 命令 {} \;

注意：

1、按类型、按文件、按大小、指定搜索深度、对查询结果执行命令，可以同时使用

了解，使用管道对查询结果执行命令

find ./ -maxdepth 1 -type f | xargs ls -l

【问题】：当有一个 含有空格的文件存在时，上述实现报错

find ./ -maxdepth 1 -type f -print0 | xargs -0 ls -l 解决 上述问题。

了解，可扩展：-ctime、-atime、-mtime -cmin -amin -mmin

### grep按文件内容查找

语法：grep -r “查找的内容” 查找的目录 -n

参数：

加参数-n可以显示查找内容的所在的行号，也可以不佳

-r，递归查找

注意：

1、find可以和grep结合使用 示例：find ./ -name "\*.c" -exec grep "main" -n {} \;

### ssh文件上传、下载

上传语法：

scp -r 本地文件 目标主机用户名@目标主机IP:绝对存储路径（不能使用相对）

下载语法：

scp -r 目标主机用户名@目标主机IP:待下载文件的绝对存储路径（不能使用相对） 本地存储位置（相对、绝对）

注意：需要输入密码

## vim简单使用

### 命令模式

复制 : yy 复制多行 行数加yy

粘贴：小p粘贴到光标前面，大P粘贴到光标后面

剪切：dd 剪切多行 行数加dd

多行选中：V 加 方向键

撤销：u

反撤销：Ctrl +r

搜索：/向下搜索 ?向上搜索 小写n搜索下一个

分屏显示：vsp 需要打开的文件名

还可以使用vs上下分屏显示

Ctrl+w切换

wqall 一次退出所有分屏

自动对齐：gg=G gg是跳转行首可以省略

### 末行模式

q：退出

q！：强制退出不保存

w：保存

wq：保存退出

1：数字几就跳转第几行

！：执行shell命令

### 基础配置

[src\linux常用命令\vim\.vimrc](file:///C:\Users\lr\AppData\Roaming\Microsoft\Word\src\linux常用命令\vim\.vimrc)

## gcc

### gcc编译的4个步骤

1、预处理：-E

* + - * 生成 .i 文件
* 头文件展开
* 宏替换
* 注释删除

2、C转汇编：-S

* 生成 .s 汇编代码

3、汇编：-c

* 生成 .o 二进制文件

4、链接：

* 链接所有 .o 文件
* 生成可执行程序

### gcc常用参数

| **gcc编译选项** | **选项的意义** |
| --- | --- |
| -E | 预处理指定的源文件，不进行编译 |
| -S | 编译指定的源文件，但是不进行汇编 |
| -c | 编译、汇编指定的源文件，但是不进行链接 |
| -o [file1] [file2] / [file2] -o [file1] | 将文件 file2 编译成可执行文件 file1 |
| -I directory | 指定 include 包含文件的搜索目录 |
| -g | 在编译的时候，生成调试信息，该程序可以被调试器调试 |
| -D | 在程序编译的时候，指定一个宏 |
| -w | 不生成任何警告信息 |
| -Wall | 生成所有警告信息 |
| -On | n的取值范围：0~3。编译器的优化选项的4个级别，-O0表示没有优化，-O1为缺省值，-O3优化级别最高 |
| -l | 在程序编译的时候，指定使用的库 |
| -L | 指定编译的时候，搜索的库的路径。 |
| -fPIC/fpic | 生成与位置无关的代码 |
| -shared | 生成共享目标文件。通常用在建立共享库时 |
| -std | 指定C方言，如:-std=c99，gcc默认的方言是GNU C |

## 库的制作和使用

### 静态库

* 静态库的特点
* 编译阶段加入程序
* 优点
* 静态库被打包到应用程序中加载速度快
* 发布程序无需提供静态库，移植方便
* 缺点
* 消耗系统资源，浪费内存
* 更新、部署、发布麻烦
* 静态库的制作：
* 通过gcc -c 编译源文件获得 .o 二进制文件
* 使用 ar 把 .o 文件打包，生成 .a 库文件
* 示例：ar rcs libcalc.a add.o div.o mult.o sub.o
* ar 命令
* rcs 参数
* libcalc.a
* lib前缀
* .a 后缀
* calc 静态库名称
* 静态库的使用
* 指定头文件位置
* -I 参数 （大写i）
* 指定静态库位置
* -L 参数
* 指定静态库的名称
* -l 参数 （小写L）
* 不要静态库文件的前后缀，只要名称

### 动态库/共享库

* 动态库的特点
* 程序运行时加入程序，多个程序共享一份动态库文件
* 优点
* 可实现进程间资源共享
* 程序升级简单
* 程序员可以控制何时加载动态库
* 缺点
* 加载速度比静态库慢
* 发布程序需要提供依赖的动态库
* 动态库的制作
* 使用gcc 得到 .o 文件
* 示例：gcc -c sub.c add.c div.c mult.c -fpic
* -fpic 动态库需要加-fpic参数，得到于位置无关的代码
* 使用gcc 得到 .so动态库文件
* 示例：gcc -shared add.o sub.o div.o mult.o -o libcalc.so
* lib前缀
* .so 后缀
* calc 动态库名称
* 动态库的使用
* 指定头文件位置
* -I （大写i）
* 指定动态库位置
* -L
* 指定动态库的名称
* -l （小写L）
* 不要动态库文件的前后缀，只要名称
* 找不到动态库文件解决
* ldd 程序名 ，检测程序是否链接到动态库
* 解决方法1：动态库的绝对路径添加到 LD\_LIBRARY\_PATH 中
* 临时生效（当前终端有效）
* 执行该命令：export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:动态库的绝对路径和名称
* 库文件名称不要前后缀
* 用户级配置
* 把该代码添加到 ~ 目录下的 .bashrc 配置文件尾部：export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:动态库的绝对路径和名称
* 库文件名称不要前后缀
* 系统级配置
* 把该代码添加到 /etc/ 目录下的 profile配置文件尾部：export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:动态库的绝对路径
* 库文件名称不要前后缀
* 解决方法2：在 /etc/ld.so.cache 添加动态库路径
* 无法直接修改 /etc/ld.so.cach 需要间接修改
* 在 /etc/ld.so.conf 配置文件中添加库所在的路径，不要名称
* 执行一个命令, 进行更新: sudo ldconfig
* 解决方法3：库文件放入 /use/lib/ 或者 /lib/ 目录中

## 简单的Makefile编写

### makefile中的规则语法：

目标 … : 依赖 …

命令(shell命令)

* 目标：最终要生成的文件，伪目标不算（.PHINY:）
* …：可以有一个或者多个生成文件或者依赖
* 依赖：生成目标的需要的材料
* 命令：通过命令对依赖操作生成目标，命令前面是tab缩进
* 一个makefile中可以有多条规则，后面的规则一般为前面的规则服务，如果前面的规则找不到依赖的文件，就会执行后面的规则，如果依赖文件足够就不会执行后面的依赖

### makefile中的变量

* 自动变量
* $@: 规则中的目标
* $<: 规则中的第一个依赖
* $^: 规则中的所有依赖

### makefile中的函数

* $(wildcard ./\*.c) 获取文件名，多个目录用空格分开
* $(patsubst %.c,%.o,$(src)) 字符串替换
* 参数：查找参数3字符串中的单词，(单词以“空格”、“tab”、”回车”、“换行”分割),是否符合参数1的单词格式，如果符合使用参数2的单词替换，结果使用变量接收，参数三的数据不会改变
* %号是通配符，表示任意长度的字符串，如果使用两个百分号，那第二个百分号中的内容来源就是第一个百分号中的内容

### 代码示例

[src\linux常用命令\makefile\makefile](file:///C:\Users\lr\AppData\Roaming\Microsoft\Word\src\linux常用命令\makefile\makefile)

### 课堂笔记：动态库、静态库、makefile的编写

[src\linux常用命令\makefile\3-gcc、动态库、静态库、makefile.md](file:///C:\Users\lr\AppData\Roaming\Microsoft\Word\src\linux常用命令\makefile\3-gcc、动态库、静态库、makefile.md)