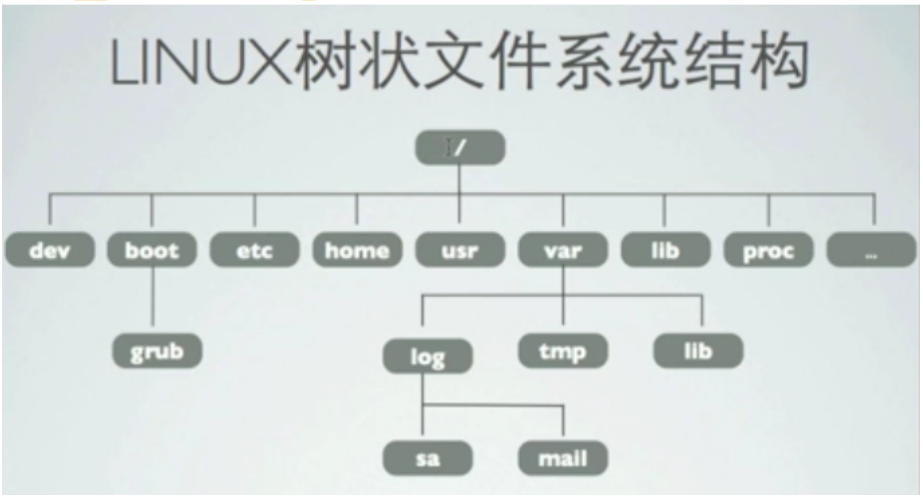
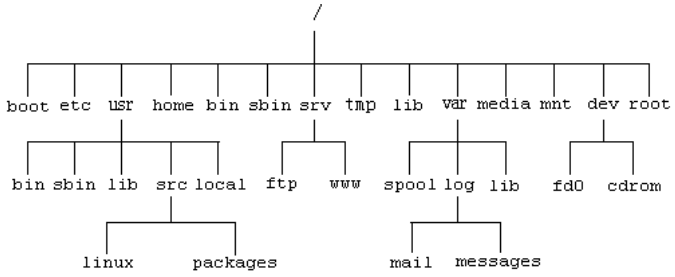
Linux

Linux文件系统结构

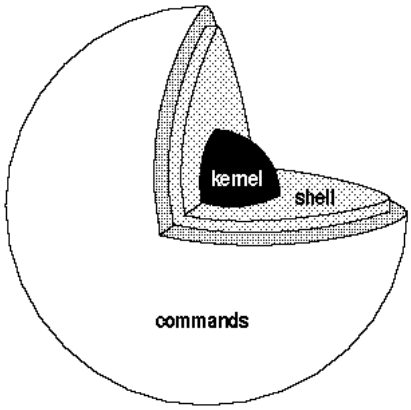
- Linux文件系统为一个倒转的单根树状结构
- 文件系统的根为`"/"`
- 文件系统严格区分大小写
- 路径使用`"/"`分割，（windows中使用`\`）





SHELL

```
用户<=>Shell | Kernel
```



- Shell 分为CLI与GUI两种
- CLI: Command Line Interface
- GUI: Grophical User Interface

操作系统的Shell

- GUI: GNOME
- CLI: BASH

BASH

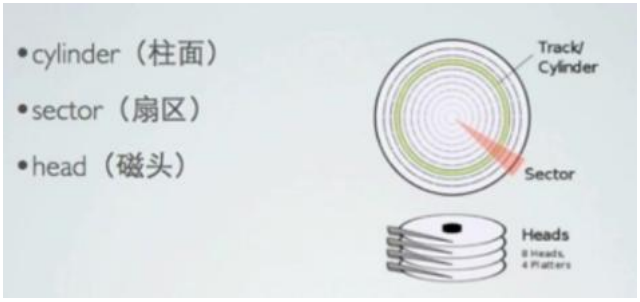
- 提示符：#，$
- 命令一般由三个部分组成：*命令*、*选项*、*参数*
- 使用Tab键来简化命令输入
    - 自动补全命令
    - 自动补全文件名
    - 无法自动补全参数

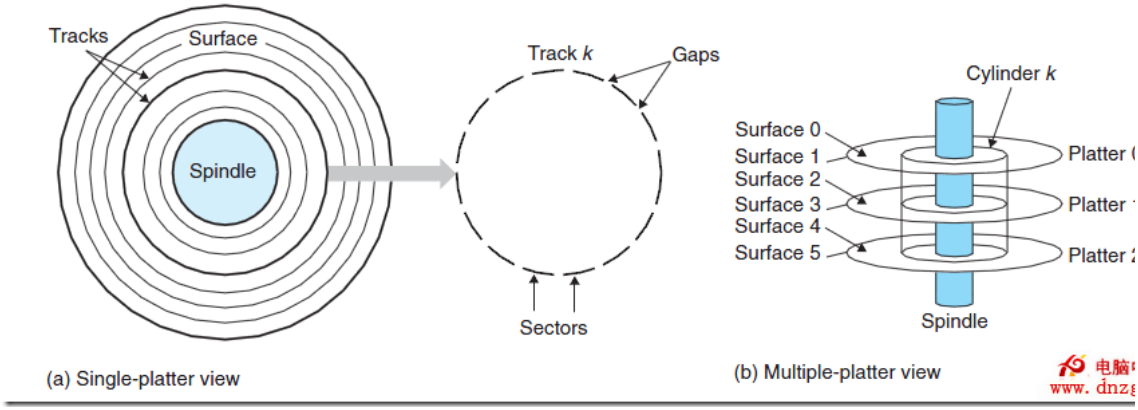
创建、删除文件

- 通过`touch`命令可以创建一个空文件或更新文件时间
- 通过`rm`命令可以删除文件或目录
- 常用参数：
    - -i 交互式
    - -r 递归的删除包括目录中的所有内容
    - -f 强制删除，没有警告提示（使用时需十分谨慎）

磁盘基本概念

- cylinder(柱面)
- sector (扇区)
- head (磁头)





磁盘在Linux中的表示

- Linux所有设备都被抽象为一个文件，保存在`/dev`目录下。
- 设备名称一般为hd[a-z] (a-z为区号),
- IDE设备的名称为hd[a-z],SATA, SCSI, SAS, USB等设备的名称为hd[a-z]

分区概念

> 将一个磁盘逻辑的分为几个区，每个区当作独立磁盘，以方便使用管理。

- 不同分区用：设备名称+分区号 方式表示，如：sda1,sda2.

- **主流的分区机制分为MBR和GPT两种**

MBR

- MBR(Master Boot Record)是传统的分区机制，（应用于绝大多数使用BIOS的PC设备）。
- MBR支持32bit和64bit系统
- MBR只支持分区数量有限
- MBR只支持不超过2T的硬盘，超过2T的硬盘将只能使用2T空间（有第三方解决方法）

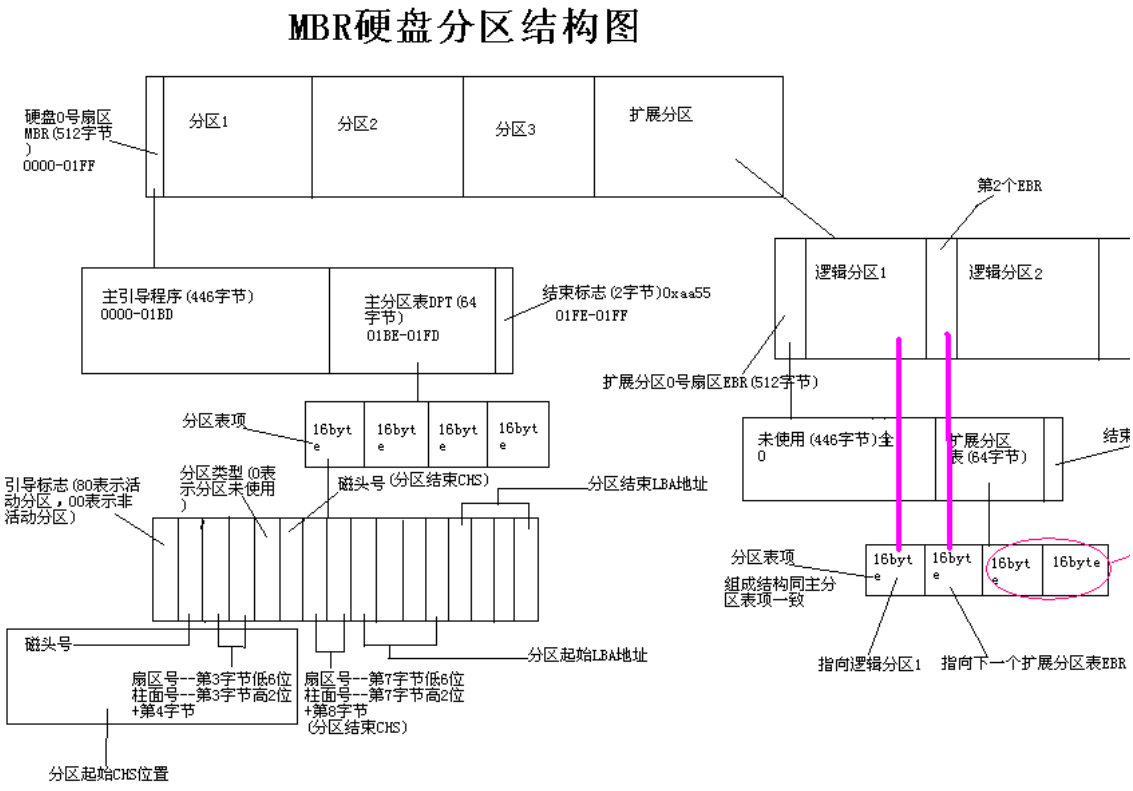
MBR结构

标准 MBR 结构

地址			描述		长度（字节）
Hex	Oct	Dec			
0000	0000	0	代码区		**440** (最大 446)
01B8	0670	440	选用软盘标志		4
01BC	0674	444	一般为空值; 0x0000		2
01BE	0676	446	**标准 MBR 分区表规划** (四个16 byte的主分区表入口)		**64**
01FE	0776	510	55h	MBR 有效标志: 0x55AA	**2**
01FF	0777	511	AAh		
MBR, 总大小: 446 + 64 + 2 =					**512**

MBR分区

- 主分区 最多只能创建4个主分区
- 扩展分区 一个扩展分区会占用一个主分区的位置
- 逻辑分区 Linux最多支持63个IDE分区和15个SCSI分区。



GPT

GPT (GUID Partition Table)是一个较新的分区机制，解决了MBR的很多缺点。

- 支持超过2T的磁盘
- 向后兼容
- 必须在支持UER的硬盘上才能使用
- 必须使用64bit系统
- Mac, Linux系统都能支持GPT分区格式
- Windows7 64bit、windowsServer 2008 64bit支持GPT

FDISK分区工具

fdisk是来自IBM的老牌分区工具，支持绝大多数操作系统，几乎所有的Linux的发行版本都装有fdisk,包括在Linux的`rescue`模式下的依然能够使用。

- fdisk是一个基于MBR的分区工具，所以如果需要使用GPT，则无法使用fdisk进行分区。
- fdisk命令只有具有超级用户权限才能够运行
- 使用fdisk -l可以列出所有安装的磁盘及其分区信息
- 使用`fdisk /dev/sda`可以对目标磁盘进行分区操作
- 分区之后需要使用partprabe命令让内核更新分区信息，否则需要重启才能识别新的分区
- /proc/partitions文件也可用来查看分区信息

文件系统

操作系统通过文件系统管理文件及数据，磁盘或分区需要创建文件系统之后才能够为操作系统使用，创建文件系统的过程又称为格式化。

- 没有文件系统的设备称之为裸（raw）设备
- 常见的文件系统有（fat32、ext2、ext3、ext4、xfs、HFS）等（看不清楚，待确认）
- 文件系统之间的区别：日志、支持的分区大小、支持的单个文件大小、性能等
- windows 下的主流文件系统是：NTFS Linux下的主流文件系统是：Ext3、Ext4

Linux支持的文件系统

- `ext2`, `ext3`, `ext4`, `fat(msdos)`, `vfat`, `nfs`, `iso9660`, `proc`, `gfs`, `jfs`

MKE2FS

命令`make2fs`用来创建文件系统`make2fs -t ext4 /dev/sda3`

常用命令

- `-b blocksize` 指定文件系统块大小
- `-c` 建立文件系统时检查坏损块
- `-L tabel` 指定卷标
- `-j` 建立文件系统日志

JOURNAL日志

带日志的文件系统(ext3、ext4)拥有较强的稳定性，在出现错误时可以进行恢复。

- 使用带日志的文件系统，文件系统会使用一个叫做`两阶段提交`的方式进行磁盘操作，当进行磁盘操作时，文件系统进行以下操作:
- 文件系统将准备执行的事务的具体内容写入日志
- 文件系统进行操作
- 操作成功后，将事务的具体内容从日志中删除
- 这样的好处是，当事务执行的时候如果出现意外（如断电或磁盘故障），可以通过查询日志进行恢复操作，缺点是会丧失一定的性能（额外的日志读写操作）。

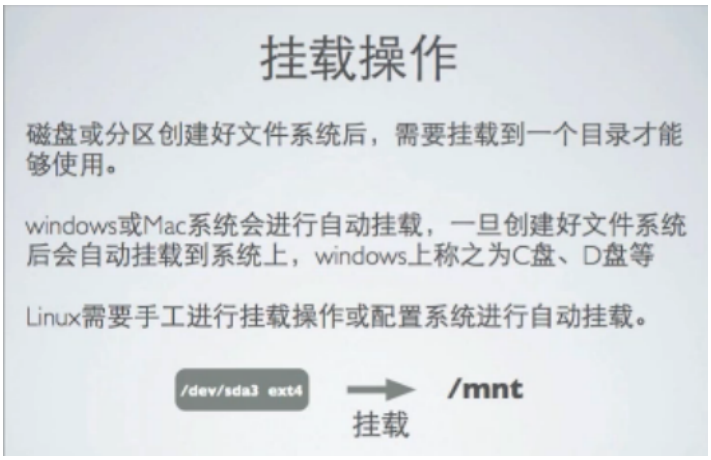
FSCK

命令fsck用来检查并修复损坏的文件系统`fsck /dev/sda2`

- 使用`-y`参数不提示而直接进行修复
- 默认fsck会自动判断文件系统类型，如果文件系统损坏较为严重，请使用`-t`参数指定文件系统类型。
- 对于识别为文件的损坏数据（文件系统无记录），fsck会将该文件放入`lost+found`目录
- 系统启动时会对磁盘进行fsck操作

挂载操作

磁盘或分区创建好文件系统后，需要挂载到一个目录才能够使用。windows或Mac会进行自动挂载，一旦创建好文件系统后会自动挂载到系统上，windows上称之为C盘、D盘等。Linux需要手工进行挂载操作或配置系统进行自动挂载。



MOUNT

在Linux中，我们通过`mount`命令将格式化好的磁盘或分区挂载到一个目录上。
eg: mount /dev/sda3(要挂载的分区) /mnt（挂载点）

常用参数

- 不带参数的`mount`命令会显示所有已挂载的文件系统
- `-t` 指定文件系统的类型
- `-o` 指定挂载选项
- `ro, rw` 以只读或读写塔式挂载，默认是rw
- `sync` 代表不使用缓存，而是对所有操作直接写入磁盘
- `async` 代表使用缓存，默认是`async`
- `noatime` 代表每次访问文件时不更新文件的访问时间
- `atime` 代表每次访问文件时更新文件的访问时间
- `remount` 重新挂载文件系统

UMOUNT

命令umount用来卸载已挂载的文件系统，相当于windows中的弹出
eg: umount 文件系统/挂载点
    umount /dev/sda3 == umount /mnt

- 如果出现device is busy报错，则表示该文件系统正在被使用，无法卸载。
- 可以使用以下命令查看使用文件系统的进程：`fuser -m /mnt`
- 也可以使用命令`lsof`查看正在使用的文件:`lsof /mnt`

自动挂载

配置文件`/etc/fstab`用来定义需要自动挂载的文件系统，fstab中每一行代表一个挂载配置，格式如下：

/dev/sda3	/mnt	ext4	defaults	o o
需要挂载的设备	挂载点	文件系统	挂载选项	dump、fsck相关选项

- 要挂载的设备也可以使用LABEL进行识别，使用LABEL=LINUXCAST取代/dev/sda3
- `mount -a`命令挂载所有`fstab`中定义的自动挂载项

获取帮助

没必要记住所有东西

Linux提供了极为详细的帮助工具及文档，一定要养成查看帮助查文档的习惯，可以大大减少需要记忆的东西并且提高效率。

MAN

- `man`命令是Linux中最为常用的帮助命令，将要获取帮助命令作为参数运行`man`命令就可以获取相应的文档帮助
- `man`文档分为很多类型：

部分	类型
1	用户命令
2	系统调用
3	库函数
4	设备文件
5	文件格式
6	游戏
7	帮助手册
8	系统管理
9	内核

- `man -k`关键字 可以用来查询包含该关键字的文档

INFO

- `info`与`man`类似，但是提供的信息更为详细深入，以类似网页的形式显示
- `man`与`info`都可以都过`/+关键字`方式进行搜索

DOC

> 很多程序、命令都带有详细的文档，以txt、HTML、PDF等方式保存在/usr/share/doc目录中，这些文档是相应程序最为详尽的文档

用户、组

> 当我们使用Linux时，需要以一个用户的身份登入，一个进程也需要以一个用户的身份运行，用户限制使用者或进程可以使用、不可以使用哪些资源。
> 组用来方便组织管理用户

- 每个用户拥有一个`UserID`,操作系统实际使用的是用户ID，而非用户名
- 每个用户属于一个主组，属于一个或多个附属组
- 每个组拥有一个GroupID
- 每个进程以一个用户身份运行，并受该用户可访问的资源限制
- 每个可登陆用户拥有一个指定的`shell`

用户

- 用户ID为32位，从0开始，但是为了和老式系统兼容，用户ID限制在60000以下。
- 用户分为以下三种：
  - root用户 （ID为0的用户为root用户）
  - 系统用户 （1~499）
  - 普通用户 （500以上）
- 系统中的文件都有一个所属用户及所属组
- 使用`id`命令可以显示当前用户的信息
- 使用`passwd`命令可以个性当前用户密码

相关文件

- `/etc/passwd` 保存用户信息
- `etc/shadow` 保存用户密码（加密的）
- `/etc/group` 保存组信息

查看登陆的用户

- 命令`whoami`显示当前用户
- 命令`who`显示有哪些用户已经登陆系统
- 命令`w`显示有哪些用户已经登陆并且在干什么

修改用户信息

- 命令`usermod`用来修改用户信息

    - `usermod` 参数 `username`
- 命令usermod支持以下参数
    - `-i` 新用户名
    - `-u` 新`userid`
    - `-d` 用户家目录位置
    - `-g` 用户所属主组
    - `-G` 用户所属附属组
    - `-L` 锁定用户例其不能登陆
    - `-U` 解除锁定

组

> 几乎所有操作系统都有组的概念，通过组，我们可以更加方便的归类、管理用户。一般来讲，我们使用部门、职能或地理区域的分类方式来创建使用组。

- 每个都有一个组ID
- 组信息保存在`/etc/group`中
- 每个用户都有一个主组，同时还可以拥有最多31个附属组

创建、修改、删除组

- 命令`groupadd`用以创建组：
    - `groupadd linux_xi`
- 命令`groupmod`用以修改组信息：
    - `groupmod -n newname oldname` 修改组名
    - `groupmod -g newGid oldGid` 修改组ID
- 命令`groupdel`用以删除组：
    - `groupdel linux_c`

文件权限

> Linux中，每个文件拥有三种权限

权限	对文件的影响	对目录的影响
r（读取)	可读取文件内容	可列出目录内容
w(写入)	可以修改文件内容	可在目录中创建删除文件
x(执行)	可以作为命令执行	可访问目录内容

- 目录必须有`x`权限，否则无法查看其内容

UGO

> Linux权限基于UGO模型进行控制：

- U代表`User`, G代表`Group`，O代表`Other`
- 每个文件的权限基于UGO进行设置
- 权限三个一组（`rwx`）,对应UGO分别设置
- 每一个文件拥有一个所属用户和所属组，对应UG，不属于该文件所属用户或所属组的使用`0`权限
- 命令`ls -l`可以查看当前目录下文件的详细信息

修改权限

- 命令`chmod`用以修改文件的权限 **`chmod` 模式 文件**
- 模式为如下格式：
- `u`,`g`,`o`分别代表用户、组和其他
- `a`可以代指`ugo`
- `+`、`-`代表加入或删除对应权限
- `r`、`w`、`x`代表权限
- **模式示例**：
- `chmod u+rw linux`
- `chmod g-x linux`
- `chmod go+r linux`
- `chmod o-x linux`
- 命令`chmod`也支持以数字方式修改权限，三个权限分别由三个数字表示：

    - `r = 4` (2^2)
    - `w = 2` (2^1)
    - `x = 1` (2^0)
- 使用数字表示权限时，每组权限分别为对应数字之和：
    - `rw = 4+2 = 6`
    - `rwx = 4+2+1 = 7`
    - `r+x = 4+1 = 5`
- 所以，使用数字表示 ugo 权限使用如下方式表示：
    - `chmod 660 linux == rw-rw----`
    - `chmod 775 linux == rwxrwxr-x`

默认权限

- 每一个终端都拥有一个`umask`属性，来确定新建文件、文件夹的默认权限
- `umask`使用数字权限方式表示，如：022
- 目录的默认权限是：777 -umask
- 文件的默认权限是：666 -umask
- 一般，普通用户的默认`umask`是002，root用户的默认`umask`是022
- 也就是说，对于普通用户来说：
- 新建文件的权限是：666-002 = 664
- 新建目录的权限是：777-002 = 775
- 命令`umask`用以查看设置`umask`值

特殊权限

权限	对文件的影响	对目录的影响
suid	以文件的所属用户身份执行，而非执行文件的用户	无
sgid	以文件所属组身份执行	在该目录中创建的任意新文件的所属组与该目录的所属组相同
sticky	无	对目录所有写入权限的用户仅可以删除其拥有的文件，无法删除其他用户所拥有的文件

设置特殊权限

- 设置suid:
    - `chmod u+s linux`
- 设置sgid:
    - `chmod g+s linux`
- 设置sticky:
    - `chmod o+t linux`
- 与普通权限一样，特殊权限也可以使用数字方式表示
    - `suid = 4`
    - `sgid = 2`
    - `sticky = 1`
- 所以，我们可以通过以下命令设置：
    - `chmod 4775 linux`

IP编址

- ip编址是一个双层编址方案，一个ip地址标识一个主机（或一个网卡接口）
- 现在应用最为广泛的是`ipv4`编址，已经开始逐渐向`ipv6`编址切换
- `ipv4`地址为32位长，`ipv6`地址为128位长
- 一个`ipv4`地址分为两个部分：网络部分和主机部分
- 网络部分用来标识所属区域、主机部分用来标识该区域中的哪个主机

32bit	网络部分|主机部分

IP地址

- IPV4地址共32位，通常使用点分十进制方式表示
- 整个IP地址分为4个部分，每个部分8位
- 例：
    - `192 . 168 . 1 . 1`
    - `11000000.10101000.00000001.00000001`

子网掩码

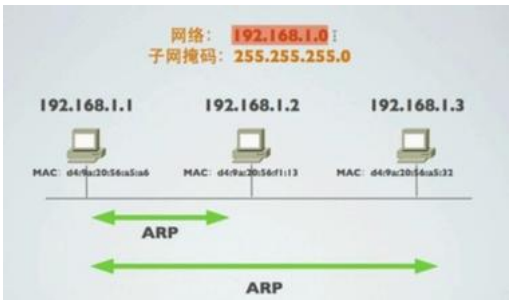
- IPv4地址的32bit分为网络部分和减速机部分
- 我们通过子网掩码来确定网络部分的位数
- 子网掩码与IP地址一样，拥有32bit,每一位与IP地址中的每一位一一对应
- IP地址中相对应子网掩码中为`1`的部分为网络部分
- 例:
    - `192 . 168 . 1 . 1`
    - `11000000.10101000.00000001.00000001`
    - `255 . 255 . 255 . 0`
    - `11111111.11111111.111111111.00000000`
- 证明此IP地址前24位网络部分，也就是说，与此IP地址处在同一个网络的其他主机的IP地址前24位相同，以证明他们在同一个网络。

IP编址

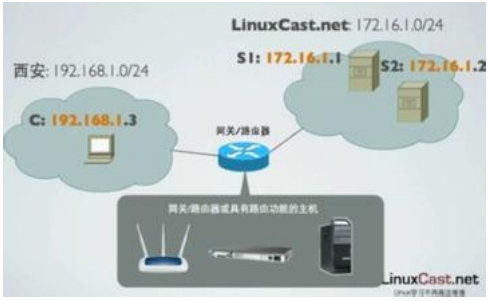
> 通过比较网络部分是否相同来判断是否处于同一网络

- eg:
    - 北京：192.168.1.0/24
    - A： 192.168.1.1
    - B： 192.168.1.2
    - C： 192.168.1.3
    - 西安：172.16.1.0/24
    - D： 172.16.1.1
    - E： 172.16.1.2
    - F： 172.16.1.3
    - 上海：10.0.0.0/8
    - H：10.0.0.1

同一个网络主机之间通信



不同网络之间通信



路由

域名

输出、查看命令

- 命令`echo`用以显示输入的内容
- 命令`cat`用以显示文件内容
- 命令`head`用以显示文件的头几行（默认10行）
  - `-n`指定显示的行数
- 命令`tail`用以显示文件的末尾几行(默认10行)
  - `-n`指定显示的行数
  - `-f` 显示文件更新（一般用于查看日志，循环读取）
- 命令`map`用于翻页显示文件内容（只能躺下翻页）
- 命令`less`用于翻页显示文件内容（带上下翻页）

管道与重定向

文件浏览

- `cat` 查看文件内容
- `more` 以翻页形式查看文件内容（只能向下翻页）
- `less` 以翻页形式查看文件内容（可上下翻页）
- `head` 查看文件的开始10行（或指定行数）
- `tail` 查看文件的结束10行（或指定行数）

基于关键词搜索

基于列处理文本

文体统计

- 命令`wc`用以统计文本信息
- `-i` 只统计行数
- `-w` 只统计单词
- `-c` 只统计字节数
- `-m` 只统计字符数

文本比较

```
命令`diff`用以比较两个文件的区别
```

- `-i` 忽略大小写
- `-b` 忽略空格数量的改变
- `-u` 统一显示比较信息（一般用以生成patch文件）

检查拼写

- 命令`aspell`用以显示检查英文拼写

处理文本内容

- 命令`tr`用以处理文本内容
- 删除关键字
  - `tr -d 'TMD' < fileName`
- 转换大小写
  - `tr 'a-z''A-Z' < fileName`

搜索替换

- 命令`sed`用以搜索并替换文本

VI、VIM

- vi是一个命令行界面下的文本编辑工具
- vim是基于vi进行了改进，加入了对GUI的支持
- emacs

VIM

- vim拥有三种模式
  - **命令模式**
  - **插入模式**
  - **ex模式**
    - 按":"进入

EX模式

- `:sh` 切换到命令行，使用`ctrl+d`切换到vim

系统启动流程

系统启动流程
BIOS
MBR:Boot Code
执行引导程序-GRUB
加载内核
执行init
runlevel

BIOS

```
BIOS(Basic Input Output System)我们称之为基本输入输出系统，一般保存在主板上的BIOS芯片中。
```

- 计算机启动时第一个运行的程序，负责检查硬件并且查找可启动设备
- 可启动设备在BIOS设置中进行定义，如：USB、CDROM、HD

MBR

- BIOS找到可启动设备后执行引导代码
- 引导代码为MBR的前446字节

GRUB

- `Grub`是现在Linux使用的主流引导程序
- 可以用来引导现在几乎所有的操作系统
- Grub的相关文件保存在`/boot/grub`目录中
- Grub配制文件为`/boot/grub/grub.conf`
- 配制格式：

KERNEL

- MBR的引导代码将负责找到并加载Linux内核
- Linux内核保存在`/boot/vmlinuz...`(模糊)
- 一般还会加载内核模块打包文件：`/boot/initramfs-... .img`
- Linux为何保持`kernel`的精简一些不常用的驱动、功能编译成为模块。在需要的时候动态加载，而这些模块被打包保存为一个`initramfs`文件。
- 早期版本Linux使用`initrd`文件，`initramfs`是`initrd`的替代优化版本，比`initrd`更加节省空间、更加灵活。
- 命令`dmesg`可以查看本次启动时内核的输出信息

INIT

- `init`是Linux系统中运行的第一个进程
- 调用`/etc/rc.d/rc.sysinit`负责对系统进行初始化。挂载文件系统，并且根据运行级别启动相应服务
- Linux运行级别：
    - **- 0 关机**
    - **- 1 单用户模式**
    - **- 2 不需网络的多用户模式**
    - **- 3 多用户模式**
    - **- 4 未使用**
    - **- 5 XII图形化模式**
    - **- 6 重新启动**
- 可以通过`/`