**第一部分**

**零：计算机概论**

计算机定义：只要有输入设备与输出设备的机器。

计算机官方定义：接受用户输入的命令与数据，经由中央处理器的算术与逻辑单元运算处理后，产生或存储成有用的信息。

电脑的五大单元：

输入单元、输出单元、算术逻辑单元、控制单元、记忆单元。

CPU内可以分为两个主要单元：算术逻辑单元、控制单元。

记忆单元包含内存和辅助存储。

注意：1. 所有单元都是由CPU内部的控制单元来负责协调的。

1. CPU的数据都来自于内存，就算从硬盘拿数据，也要先将数据移到内存中，然后在交于CPU判断计算。

比拟：

电脑：人

控制单元：大脑平衡部分

算术逻辑单元：大脑逻辑思考部分

记忆单元：记忆、经验储存/暂存区

输入单元：眼睛、鼻子、耳朵、皮肤、嘴巴

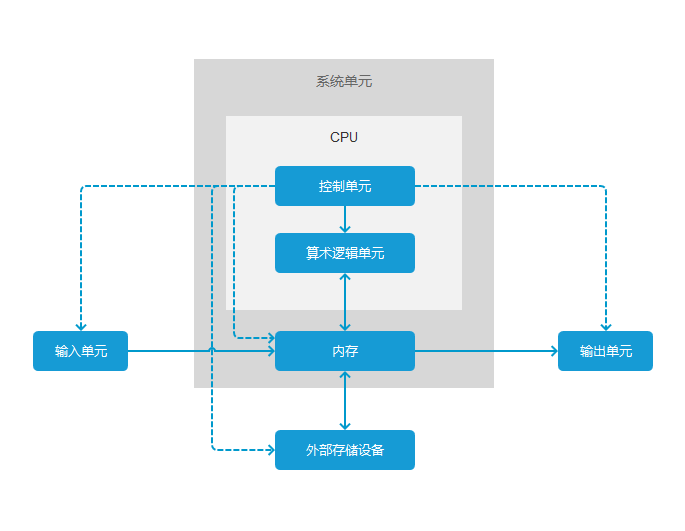
输出单元：嘴巴说、手脚做、排泄系统

CPU：脑袋

内存：信息暂存区

辅助存储：信息长久存储区

图示：



电脑常用计算单位：容量、速度等

二进制单位：位（bit）

光纤传输速度HZ，数据容量bit。

1bit=8位 bit（字节）

HZ：次数/秒

速度转为容量时除以8就行。

主板：

早期：

南桥：负责连接较慢的设备接口，硬盘、USB设备、网卡等。

北桥：负责连接速度较快的CPU、内存和显卡等组件。

现在：将内存控制器整合到了CPU中。

外频：CPU与外部组件进行数据传输时的速度。

倍频：CPU内部用来加速工作性能的一个倍数。

CPU的频率速度：外频 X 倍频

固件：固定在硬件上面的控制软件（一段程序）。

只读存储器：ROM（Read Only Memory），一种存储芯片。没有通电也可以记录数据。

一般硬件上面的固件都存储在ROM中。

硬盘与存储设备：

硬盘的物理组成：由许许多多的圆形碟片、机械手臂、磁头与主轴马达组成。

扇区：磁盘的最小存储单位。

磁道：同一个同心圆的扇区组合成的圆就是所谓的磁道。（像圆环）

柱面：由于磁盘可能会有多个碟片，因此在所有碟片上面的同一个磁道可以组

成所谓的柱面。

热插拔：即带电插拔。

描述：指的是支持在不关闭系统电源的情况下，将“模块、板卡”插入或

拔出系统，而不影响系统的正常工作的一种功能。

SSD（Solid State Drive）固态硬盘

HDD（Hard Disk Drive）机械硬盘

软件程序运行：

操作系统：其实也是一组程序。只是这组程序的重点在于管理电脑的所有活动以

及驱动系统中的所有硬件。

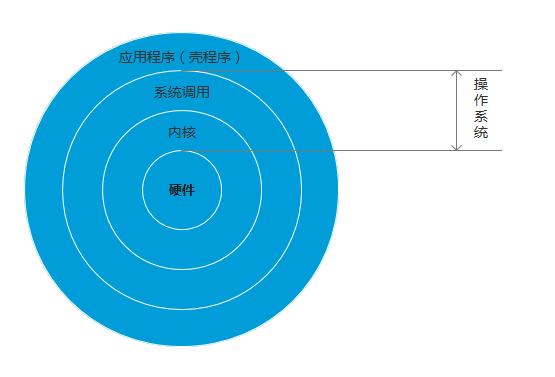
由内核及其提供的接口工具组成。

内核：属于操作系统中的一部分，也是一组程序。内核程序放置到内存当中的区

块是受保护的，并且启动后就一直常驻内存当中。

硬件：都是受到内核管理的。

关系图：



内核功能：

系统调用接口：方便程序员可以轻易的通过与内核的沟通，将硬件资源进一

步地利用。

进程管理：控制所有任务，让CPU的资源作有效的分配。

内存管理：控制整个系统的内存管理。内核通常会提供虚拟内存的功能，当

内存不足时可以提供交换分区的功能。

文件系统管理：数据的输入与输出等，还要不同文件格式的支持。

设备驱动：设备驱动就是内核需要做好的事情。

驱动程序：全称“设备驱动程序”，是一种可以使计算机和设备通信的特殊程序。

相当于硬件的接口，操作系统只有通过这个接口，才能控制硬件设备

的工作。它添加到操作系统中，包含有关硬件设备的信息。

**一：Linux是什么与如何学习**

1. 每种操作系统都是在它专门的硬件架构上面运行的。
2. 操作系统的核心是内核，而内核又是针对并根据硬件架构来开发出来的一组代码

程序。所以同一款操作系统不可能在不同的硬件架构下运行。

1. Linux版本指的应该是内核版本。
2. Linux发行版：内核+软件+工具+可完全安装程序。
3. Linux发行版分为商业、社区2类或者分为RPM、DPKG 2类。
4. 在Linux系统下所有的程序和系统设备都是文件。

源代码：程序员写出的并没有编译的程序。

开源：软件发布的同时，将源代码一起公布。

操作系统：

1. 能管理并驱动系统中的所有硬件
2. 能提供接口工具供程序使用

在服务器应用上：

1. 文件的安全性
2. 人员账号的管理
3. 软件的安装、修改、设置
4. 日志文件的分析及计划任务
5. 程序的编写

学习Linux：

1. 计算机概论与硬件相关知识
2. 先从Linux的安装与命令学起
3. Linux操作系统的基础技能
4. 务必学会vi文本编辑器
5. Shell与shell脚本的学习
6. 一定要学会软件管理
7. 网络基础的建立
8. 如果上面都做好了，那么网站的搭建对你来说，简直就是太简单了！

有系统的设计文件目录

养成一个做记录的习惯：

1. 发现问题的原因
2. 问题的引发状况
3. 错误信息
4. 解决方法

学习Linux最好从头由基础开始学习。

**二：主机规划与磁盘分区**

MBR磁盘分区表：Master Boot Record 主引导记录

格式：

第一个扇区主要会有两个东西：（这个扇区的大小通常是512字节）

1. 主引导记录：可以安装启动引导程序的地方，有446字节
2. 分区表：记录整块硬盘分区状态，有64字节。
3. 其实所谓的分区只是针对那64字节的分区表进行设置而已。
4. 硬盘默认的分区表仅能写入四组分区信息。
5. 这四组划分信息我们称为主要或者扩展分区。
6. 分区的最小单位通常称之为组面。
7. 当系统要写入磁盘时，一定会参考磁盘分区表，才能针对某个分区进行数据的处理。

MBR主要分区、扩展分区、逻辑分区的特性：

1. 主要分区与扩展分区最多可以有4个（硬盘限制）。
2. 扩展分区最多只能有一个（操作系统限制）。
3. 逻辑分区是由扩展分区持续划分出来的分区。
4. 能够被格式化后作为数据存取的分区是主要分区与逻辑分区，扩展分区无法格式化。
5. 逻辑分区的数量依操作系统而不同，在Linux系统中SATA硬盘已经可以突破63个以上的分区限制。
6. 如果扩展分区被破坏，所有的逻辑分区都会被删除。
7. 考虑到磁盘的连续性，一般建议将扩展区的柱面号码分配在最后面的柱面内。

限制：（过去的MBR）

1. 操作系统无法使用2.2TB以上的磁盘容量。（因为每组分区表仅有16字节）
2. MBR仅有一个区块，若被破坏后，经常无法或很难恢复。
3. MBR内的存放启动引导程序的区块进446字节，无法存储较多的程序代码。

GPT磁盘分区表：（GUID partition table）

结构：

1. 前面使用34个LBA来记录分区信息。
2. 最后面的34个LBA来作为另一个备份。
3. 中间的才是可用分区。

LBA0：MBR兼容区块

与MBR模式相似，这个区块也分为2部分。

1：和MBR446字节相似，存储了第一阶段的启动引导程序。

2：仅仅放入了一个特殊标识符，用来表示此盘为GPT格式之意。

LBA1：GPT表头记录

1. 记录了分区表本身的位置与大小。
2. 记录了备份用的GPT分区（最后34个LBA区块）放置的位置。
3. 放置了分区表的校验码。

LBA2~33：记录分区信息处

1. 从LBA2开始，每个LBA都可以记录4组分区记录，所以在默认情况下，总共可以有4X32=128组分区记录。
2. GPT分区表对于单一分区来说，它的最大容量限制就会在8ZB。
3. 现在GPT分区默认可以提供多达128组记录。
4. 如果磁盘容量大于2TB，系统自动使用GPT分区方式来处理磁盘分区。
5. Linux本身的内核设备记录中，针对单一磁盘来说，过去最多只能达到15个分区；现在由于Linux内核通过udev等方式处理，已经没有这个限制了。
6. GPT已经没有了所谓的主要分区、扩展分区、逻辑分区的概念了，既然每组记录都可以独立存在，当然每个都可以视为主要分区，每一个分区都可以拿来格式化使用。
7. 能否读写GPT格式与启动的检测程序有关。
   1. 启动的检测程序：BIOS与UEFI。
8. CMOS：记录各项硬件参数且嵌入在主板上面的存储器。
9. BIOS：在主机启动时，计算机系统会主动执行的第一个程序。
10. BIOS执行后，会认识第一个可启动设备。（会去分析计算机里面有哪些存储设备）
11. MBR：第一个可启动设备的第一个扇区的主引导记录块，内含启动引导代码。
12. 启动引导程序（boot loader）：一个可读取内核文件来执行的软件。（它是操作系统在安装的时候提供的）
13. 内核文件：开始启动操作系统。

注：

1. BIOS与MBR都是硬件本身会支持的功能。
2. 一个硬盘仅由于一个MBR。
3. 各个分区有各自的启动扇区。
4. 启动电脑需要启动引导程序（Boot loader）。
5. 启动引导程序除了可以安装在MBR之外，还可以安装在各个扇区的启动扇区。
   1. Boot loader（启动引导程序）：操作系统安装在MBR上面的一个软件

任务：

1. 提供选项：用户可以选择不同的启动项，这也是多重引导的重要

功能。

1. 加载内核文件：直接指向可以使用的程序区段来启动操作系统。
2. 转交其它启动引导程序：将启动管理功能转交给其它启动引导程序负责。

注：

1. 实际可启动的内核文件是放置到各分区中的。
2. Boot loader 只会认识自己的系统分区内的可启动的内核文件，以及其它Boot loader而已。
3. Boot loader可直接指向或是间接将管理权转交给另一个管理程序。
4. Linux在安装时，你可以选择将Boot loader安装在MBR或各别分区的启动扇区，而且Linux的Boot loader可以手动设置选项，所以你可以在Linux的Boot loader里面加入Windows启动的选项。
5. Windows在安装时，它的安装程序会主动的覆盖掉MBR以及自己所在分区的启动扇区，你没有选择安装在哪里的机会，而且它没有让我们自己选择选项的功能。
   1. 磁盘分区的选择（Linux）
6. Linux系统使用的是目录树架构。
7. 文件系统与目录树的关系（挂载）：

挂载：利用一个目录当成进入点，将磁盘分区的数据放置在该目录

下；也就是说进入该目录就可读取该分区的意思。

挂载点：当成进入点的那个目录。

1. .ios镜像文件不能以数据格式录成为 CD/DVD，你必须要使用刻录程序的功能将它以“镜像文件格式”刻录成为CD或者DVD才行，切记不要使用刻录数据文件格式来刻录。
2. NAT（完成IP分享器的功能）：用于Linux系统。
3. SAMBA（加入Windows网络邻居）：用于Linux系统完成系统之间数据分享功能。（没有客户端数量限制）
4. Mail（邮件服务器）：适用于需要私下传输邮件的情况。
5. Web（WWW服务器）：除了提供互联网的WWW连接之外，很多在网络主机上面的软件功能也都使用WWW作为显示接口。
6. DHCP（提供客户端自动获取IP的功能）：让计算机在连上互联网时让它自动的获取IP。
7. FTP：文件传输。（容易被查到传输记录）
8. Linux交换分区：就是Linux下的虚拟内存分区，它的作用是在物理内存使用完之后，将磁盘空间（swap分区）虚拟成内存来使用。它和Windows系统的交换文件作用类似，但是它是一段连续的磁盘空间，并且对用户不可见。
9. 开机流程：BIOS→MBR→引导启动程序→内核文件。
10. Boot loader可以安装的地点：MBR和启动扇区。

**三：安装CentOS 7.x**

1. 不论你安装什么样的系统，你都应该需要事先规划，例如磁盘分区、启动引导程序。
2. 安装Linux系统的模式至少有两种，分别是图形用户界面模式和命令行模式。
3. 在安装的过程中，可以建立逻辑卷管理器（LVM）。
4. 一般要求交换分区应该是物理内存的1.5~2倍，但即使没有交换分区一九能够安装与运行Linux操作系统。
5. Centos7默认使用xfs作为文件系统。
6. 没有连上互联网时，可以尝试关闭防火墙，但SELinux最好选择“强制”状态。
7. 设置时不要选择启动kdump，因为那是给内核开发者查看宕机数据之用。

安装步骤：

1. 调整BIOS：使用CD/DVD光盘启动，通常需要调整BIOS。（可使用U盘）
2. 选择安装模式并启动：包括图形用户界面模式或命令行模式。
3. 选择语言：由于不同地区的键盘按键不同，此时需要调整语系、键盘、鼠标等。
4. 软件选择：需要什么样的软件？全部安装还是默认安装即可。
5. 磁盘分区：最重要的地方之一。
6. 启动引导程序、网络、时区设置与root密码：一些需要的系统基础设置。
7. 安装后的首次设置：比如用户、SELinux、防火墙等等。

注：如果使用U盘启动的话，需要注意U盘的型号，2.0会被主板识别成便携设备，3.0则可能会被主板识别成磁盘。

**四：首次登录与在线求助**

1. 为了避免瞬间断电造成的linux系统危害，如果您的Linux主机是作为服务器的话，建议加上UPS来持续提供稳定的电源。
2. UPS－Uninterrupted Power System；利用电池化学能作为后备能量，在市电断电等电网故障时，不间断地为用户设备提供（交流）电能的一种能量转换装置。
3. 在X Window下，快速重启界面模式快捷键： CTRL+ALT+Backspace。
4. 默认情况下，Linux提供tty1~tty6 的终端界面。
5. Date显示日期；cal显示日历；bc可以作为计算器。
6. man page 说明后面的数字中，如DATE（1）中的1，代表一般用户可使用的命令。
   1. 1 -> 用户在shell 环境中可以操作的命令或可执行文件。
   2. 2 -> 系统内核可调用的函数与工具等。
   3. 3 -> 一些常用的函数与函数库，大部分为C的函数库。
   4. 4 -> 设备文件的说明，通常在 /dev 下的文件。
   5. 5 -> 配置文件或是某些文件的格式。
   6. 6 -> 游戏（games）
   7. 7 -> 惯例与协议等，列入Linux文件系统、网络协议、ASCLL代码等的说明。
   8. 8 -> 系统管理员可用的命令。
   9. 9 -> 跟内核有关的文件。
7. man page 一口气输出一堆信息。如：man date。
8. info page则是将文件数据拆成一个个的段落，每个段落用自己的页面来编写，并且在每个页面还有类似与网页的超链接来跳转到不同的页面；每个独立的页面也被称为一个节点。
9. 使用info page命令查询命令相关信息的前提是该命令的在线求助文件的格式有info page这类格式的。如果没有，则其查询结果和man 相同。
10. info page使用方式。 如：info date。

**五：Linux的文件权限与目录配置。**

1. 在Linux里面，任何一个文件都具有用户、所属群组、其他人三种身份的个别权限。
2. dr-xr-x--- 5 root root 4096 May 19 16:48 abc

[ 1 ] [2] [ 3 ] [ 4 ] [ 5 ] [ 6 ] [ 7 ]

权限 链接 拥有者 用户组 文件容量 修改日期 文件名

第一栏有10个字符：

第一个字符表示这个文件是目录/文件/链接文件。

【d】 --> 表示目录

【-】 --> 表示文件

【l】 --> 表示链接文件

【b】 --> 表示设备文件里面的可供存储的周边设备。

【c】 --> 表示设备文件里面的串行端口设备，比如键盘、鼠标（一次性读取设备）

第2，3，4 表示的是拥有者的权限。

第5，6，7表示的是加入此用户组之账号的权限。

第8，9，10表示的是其他人的权限。

1. Export LC\_ALL=en\_US.utf8修改语系。
2. 修改文件属性与权限：
   1. chgrp（change group）：修改文件所属用户组。
      1. chgrp [-R] groupname filename/dirname ...
      2. 例如：chgrp g\_one abc.txt ccc.txt；将abc.txt与ccc.txt的所属用户组改为g\_one。
      3. 例如：chgrp g\_one -R adc；将文件夹adc和其目录下的所有文件及其子目录下的所有文件的用户组都更改为g\_one。
   2. chown（change own）：修改文件拥有者。
      1. chown [-R] username filename/dirname
      2. 例如：chown u\_one a.txt；将文件a.txt 的拥有者修改为u\_one。
      3. 例如：chown -R u\_one adc；将文件夹adc和其目录下的所有文件以及其子目录下的所有文件的拥有者都修改为u\_one。
      4. chown uname:gname [-R] filename/dirname
      5. 例如：chown u\_one:g\_one a.txt；将文件a.txt 的拥有者修改为u\_one，所属用户组修改为g\_one。
      6. 例如：chown -R u\_one:g\_one adc；将文件夹adc和其目录下的所有文件以及其子目录下的所有文件的拥有者都修改为u\_one，所属用户组修改为g\_one。
      7. chown user.group file 和 chown user:group file等价。
   3. chmod（change modify）：修改文件的权限，SUID、SGID、SBIT等的特性。
      1. Linux文件的基本权限有9个，分别是拥有者，所属用户组，其他人三种身份各有自己的 读（read）、写（write）、执行（execute）权限。
      2. 这9个权限3个分为一组。
      3. r=4；w=2；x=1。
      4. 如果我们要给a.txt 文件设置 rwxr-xr-x的权限。如果a.txt的原本权限是r-xr--rwx
         1. chmod u=rwx,g=r-x,o=r-x a.txt
         2. chmod 755 a.txt （7=r+w+x；5=r+x；5=r+x）
         3. chmod u+w,g+x,o-w a.txt
      5. 如果要给a.txt 的拥有者，所属用户组，其他人的权限都加上x权限
         1. chmod a+x a.txt
      6. 如果要给a.txt 的拥有者，所属用户组，其他人的权限都删除x权限
         1. chmod a-x a.txt
      7. chmod [-R] 权限 filename/dirname
3. 如果文件名前面多一个【.】，则代表该文件是隐藏文件。
4. 如果需要root的权限时，可以使用 su root 这个命令来切换身份。完成之后exit即可离开su环境。
5. 权限：
   1. 对文件来说：
      1. 【r】：可读取此文件的实际内容，如a.txt 的文本内容。
      2. 【w】：可以编辑/新增或是修改该文件的内容（但是不含删除此文件）
      3. 【x】：该文件具有可以被系统执行的权限。
   2. 对目录来说：
      1. 【r】：可读取目录中的内容。
      2. 【w】：修改目录中的内容（含删除该目录下的文件）
      3. 【x】：访问目录。
6. Linux文件名的限制为：单一文件或目录的最大容许文件名为255个英文字符或128个中文字符。
7. FHS 制定出来的四种目录分别为：shareable、unshareable、static、variable等4类。
   1. FHS：可分享、不可分享、不变、可变动。
8. FHS所定义的三层主目录为：/、/var、/usr三层。
   1. /（root, 根目录）：与启动系统有关。
   2. /var（variable）：与系统运行过程有关。
   3. /usr（unix software resource）：与软件安装/执行有关。
9. 绝对路径为从根目录 / 开始写起，否则就是相对路径。

**六：Linux文件与目录管理**

1. cd（change directory）命令：
   * 1. 参数【~】：cd ~ 可以回到自己的家目录。（默认，即cd不带参数也是cd ~）
     2. 参数【~username】：cd ~dove 进入dove这个使用者的家目录。
     3. 参数【..】：cd .. 代表进入当前目录的上一层目录。
     4. 参数【-】：cd - 代表返回你前一次进入的目录。
     5. 参数【path】：cd /home/dove/game 通过绝对（相对）路径进入目标目录。
2. pwd（print working directory）命令：
   * 1. 显示当前所在目录：pwd，如我在home/dove/game 目录中使用pwd，返回的字符串：/home/dove/game
     2. 显示出真正的路径，而非使用链接路径：pwd -P
3. mkdir（make directory）命令：
   * 1. 参数【m】：mkdir -m 771 test ; 创建一个权限为rwxrwx--x 的空目录test。
     2. 参数【p】：mkdir -p test/test1/test2; 若test、test1目录不存在，将会依次创建test、test1、test2目录；也就是说 p 这个参数会自动帮你递归创建你所需要的目录。
     3. 无参数：mkdir tt; 创建tt空目录。
4. rmdir（remove directory）命令：
   * 1. 参数【p】：rmdir -p test/test2/test3; 递归删除test、test2、test3空目录。
     2. 无参数：rmdir tt; 删除tt这个空目录。
     3. 注意： rmdir该命令仅能删除空目录。
5. 关于执行文件路径的变量： $PATH
   * 1. 每一个路径之间用【:】隔开。
6. ls（list）命令：（常用一次向下）
   * 1. 参数【a】：ls -a; 列出全部的文件，包括隐藏文件（.开头的文件）。
     2. 参数【A】：ls -A; 列车全部的文件，包括隐藏文件，但不包括 【.】与【..】这两个目录。
     3. 参数【d】：ls -d; 仅列出目录本身，而不是列出目录内的文件数据。
     4. 参数【l】：ls -l; 详细信息显示，包含文件的属性与权限等数据；
     5. 参数【r】：ls -r; 将排序结果反向输出，例如：原本文件名由小到大，反向则为由大到小。
     6. 参数【R】：ls -R; 连同子目录内容一起列出来。等于该目录下所有的文件都会显示出来。
     7. 参数【S】：以文件容量大小来排序，而不是用文件名排序。
     8. 参数【t】：ls -t; 依照时间排序，而不是依照文件名。
     9. 参数【--full-time】：ls --full-timel; 以完整时间模式（包含年、月、日、时、分）输出。
     10. 参数【n】：ls -n; 列出UID与GID，而非使用者与用户组的名称。
     11. 参数【f】：ls -f; 直接列出结果，而不进行排序（ls 默认文件名排序）。
     12. 参数【F】：ls -F; 根据文件、目录等信息，给与附加数据结构。例如：

\*：代表可执行文件；/：代表目录；=：代表socket文件；|：代表FIFO文件

* + 1. 参数【h】：ls -h; 将文件容量以人类较易读的方式列出来。如GB/KB等
    2. 参数【i】：ls -i; 列出inode号码。
    3. 参数【--color】：ls --color=never/always/auto; 不显示颜色/显示颜色/系统判断。
    4. 参数【--time】：ls --time=atime/ctime; 输出文件读取时间/改变权限属性时间。默认是mtime（内容修改时间）

1. 获取路径的文件名与目录名称：
   * 1. basename etc/sysconfig/network; 返回network，也就是取得最后的文件名。
     2. dirname etc/sysconfig/network; 返回etc/sysconfit，也就是取得目录路径。
2. 文件内容查看：
   * 1. 命令cat：由第一行开始显示文件内容。（concatenate）
     2. 命令tac：从最后一行开始显示，可以看出tac是cat的倒着写。
     3. 命令nl：显示的时候，同时输出行号。
     4. 命令more：一页一页的显示文件内容。
     5. 命令less：和more类似，一页一页的显示文件内容，比more好的是可以往前翻页。
     6. 命令head：只看前面几行。
     7. 命令tail：只看后面几行。
     8. 命令od：以二进制的方式读取文件内容。
3. 命令详解：一下命令后面直接接文件（如：cat -A a.txt）
   * 1. cat -A：将文件中一些看不出来的字符以特殊符号显示出来，而不是显示空白。
     2. cat -v：列出一些看不出来的特殊字符。
     3. cat -T：将[tab]按键以【^I】的形式显示出来。
     4. cat -E：将结尾的换行符以【$】的形式显示出来。
     5. cat -b：列出行号，仅针对非空行做行号表示，空白行不标行号。
     6. cat -n：列出行号，不论空行还是非空行都标识行号。
     7. cat：没有参数，就没有任何格式的打印出文件内容。
     8. tac：将文件内容从最后一行开始显示，知道显示到第一行内容为止。
     9. nl -b a：表示不论是否空行，也同样列出行号，和cat -n类似。
     10. nl -b t：仅针对非空行列出行号，空行不列出行号，和cat -b类似。（默认值）
     11. nl -w：行号栏位的占用的字符数，默认是6。
     12. nl -n ln：行号在屏幕最左方显示。
     13. nl -n rn：行号在自己栏位的最右方显示，且不加0。
     14. nl -n rz：行号在自己栏位的最右方显示，且加0。（如：000001）
     15. head -n 20：显示文件内容前20行，包括空行。
     16. head：显示文件内容前10行。（不带参数的默认值）
     17. head -n -20：显示到文件倒数第20行为止（不包含倒数第20行）。可以理解为【20或者-20】都是截止位置，唯一不同的是，-20不包含-20，20包含20。
     18. tail -n 20：显示文件内容最后20行，包含空行。
     19. tail：显示文件内容最后10行。（不带参数的默认值）
     20. tail -n +20：显示文件内容第20行（包含第20行）之后的内容。可以理解为【20或者+20】都是开始位置，都包含第20行的内容。
     21. head -n 20 a.txt | tail -n 10：显示文件a.txt的第11行~第20行的内容。
     22. 管道命令【|】：前面命令所输出的信息，通过管道交由后续的命令继续使用。
     23. od -t a：利用默认的字符来输出文件内容。
     24. od -t c：利用ASCII字符来输出文件内容。
     25. od -t d[size]：利用十进制来输出文件内容，每个整数占用size Bytes。
     26. od -t f[size]：利用浮点数值来输出文件内容，每个整数占用size Bytes。
     27. od -t o[size]：利用八进制来输出文件内容，每个整数占用size Bytes。
     28. od -t x[size]：利用十六进制来输出文件内容，每个整数占用size Bytes。
     29. od -t oCc a.txt：将a.txt文件的内容以八进制列出存储值与ASCII的对照表。
4. 修改文件时间或创建新文件：touch
   * 1. 文件的主要的变动时间有3个：
        1. 修改时间mtime(modification time)：当该文件的【内容数据】变更时，就会更新这个时间。
        2. 状态时间ctime(status time)：当该文件的【状态】改变时，就会更新这个时间，比如他的权限与属性被更改了，都会更新这个时间。
        3. 读取时间atime(access time)：当该文件的【内容】被读取时，就会更新该事件，比如我们用cat 去读取a.txt这个文件时，就会更新a.txt 的atime。
     2. touch -a：仅定义 atime。
     3. touch -m：仅定义mtime。
     4. touch -t：后面可接欲自定义的时间而不是用目前的时间。【格式：YYYYMMDDhhmm】
     5. touch -c：仅修改文件的时间，若该文件不存在则创建该文件。
     6. touch -d：后面可以接欲自定义的时间而不是用目前的时间，也可以使用 --date=”日期或时间”。
     7. 复制文件的时候，会更新ctime为目前时间，而不会更新atime与mtime。atime与ctime的值与源文件一致。
     8. touch命令常用于：
        1. 新建一个空文件。
        2. 将某个文件日期自定义为目前（atime和mtime）
5. 文件默认权限：umask
   * 1. umask就是指定目前用户在建立文件或目录时候的权限默认值。
     2. 读取umask：
        1. umask ---> 0022（与一般权限有关的是最后三个数字，第一个数字是特殊权限）
        2. umask -S ---> u=rwx,g=rx,o=rx
     3. umask的数字指的是【该默认值需要减掉的权限】
     4. 建立文件时初始权限：-rw-rw-rw-
     5. 建立目录时初始权限：drwxrwxrwx
6. 文件隐藏属性：
   * 1. chattr [+-=] [ASacdistu] 文件或目录名称
     2. 【+】：增加一个或多个特殊参数，其他原本存在的参数则不动。
     3. 【-】：删除一个或多个特殊参数，其他原本存在的参数则不动。
     4. 【=】：直接设置特殊参数，且仅有后面接的参数，原有的参数全部删除。
     5. 【A】：当设置了该属性时，若你在存取此文件或目录时，它的存取时间atime将不会被修改，可避免I/O较慢的机器过度的读写磁盘。
     6. 【a】：当设置这个属性时，这个文件将只能增加数据，而不能删除也不能修改数据，只有root才能设置该属性。
     7. 【S】：一般文件是非同步写入磁盘的，如果加上该属性，则当你进行任何文件的修改，该修改会【同步】写入磁盘中。
     8. 【c】：当设置该属性时，将会自动的将此文件【压缩】，在读取的时候将会自动解压，但是在存储的时候，将会先压缩在存储。
     9. 【d】：当dump程序被执行的时候，设置该属性将可使该文件/或目录不会被dump备份。
     10. 【i】：当设置该属性时，这个文件或目录不能被删除、改名、设置链接也无法写入或新增数据。对系统安全性有相当大的助益，只有root可设置该属性。
     11. 【s】：当设置该属性时，如果该文件被删除，它将会被完全的从硬盘产出，所以如果误删，完全无法恢复。
     12. 【u】：与s属性相反，如果设置了该属性，如果文件被删除了，然则数据内容还在磁盘中，可以使用方法来恢复该文件。
     13. chattr 命令只能在ext2、ext3、ext4、的Linux传统文件系统上面完整生效，其他文件系统可能就无法完整支持该命令了。
     14. xft文件系统仅支持Aadis而已。
     15. 由于这些属性时隐藏的性质，所以需要以lsattr命令才能看到这些属性。
     16. lsattr [-adR] 文件或目录“
         1. 【a】：将隐藏的属性也显示出来。
         2. 【d】：如果接的时目录，仅列出目录本身的属性而非目录内的文件名。
         3. 【R】：连同子目录的数据也一并列出来。
7. 文件特殊权限：SUID、SGID、SBIT
   * 1. SUID
        1. 这个权限仅对二进制程序有效。
        2. 执行者对于该程序需要具有【x】的可执行权限。
        3. 本权限尽在执行该程序的过程中有效。
        4. 执行者将具有该程序拥有者的权限。
        5. 该权限仅可用在二进制程序上，不能够用在shell脚本上面。
     2. SGID
        1. 该权限可以针对文件或者目录来设置。
           1. 对文件：

SGID对二进制程序有用。

程序执行者对于该程序来说，具有【x】的权限。

执行者在执行该程序的过程中将获得该程序用户组的支持。

* + - * 1. 对目录：

用户若对于此目录具有r与x的权限时，该用户能进入该目录。

用户在此目录下的有效用户组，将会变成该目录的用户组。（user的用户组暂时变成该目录的用户组。例如：user的属性的值被暂时修改成目录的属性值）

用途：若用户在此目录下具有w的权限（可以新建文件），则用户所建立的新文件的用户组与此目录的用户组相同。

* + 1. SBIT
       1. 该权限目前只针对目录有效，对文件已经没有效果了。
       2. 当用户对于此目录具有w、x的权限时，即具有写入的权限。
       3. 当用户在此目录下建立文件或目录时，仅有自己与root才有权限删除该文件或目录。
    2. SUID/SGID/SBIT权限设置：
       1. 他们对应的数字分别是：SUID--->4，SGID--->2，SBIT--->1
       2. 例如：我们将a 二进制程序设置成SUID，chmod 4755 a，那么a的权限则为：-rwsr-xr-x
       3. 注意：这三个特殊权限都是修改的执行【x】所在位置。若文件没有x权限的话，那么就会使用大写字母来代替，如 chmod 7666 a，则a的权限就是-rwSrwSrwT

1. 观察文件类型：file
   * 1. 通过该命令，我们可以简单的先判断这个文件的格式是什么，包括你也可以用来判断tar包使用的是哪一种压缩方式。
2. 命令与文件的查找：
   * 1. 在命令行模式下，连续按2次tab键就能够知道用户有多少命令可以执行。
     2. 命令which：脚本文件（命令）的查找
        1. which：查找执行文件
        2. 这个命令是根据【PATH】这个环境变量所规范的路径，去查找执行文件的文件名，所以，重点是找出执行文件而已，且which后面接的是完整的文件名。
        3. which -a ls：列出所有的可以找到的同名执行文件，而非只显示第一个而已。
     3. 文件的查找：
        1. Whereis [-bmsu] 文件或目录名
           1. whereis -b a：只找二进制格式的且名字为a的文件。
           2. whereis -m a：只找在说明文件manual路径下的名字问a的文件。
           3. whereis -s a：只找名字为a的source源文件。
           4. Whereis -u a：查找不再上述3个项目当中的其他特殊文件（名字为a）。
           5. whereis -l：可以列出whereis命令会去查询的几个主要目录。
        2. locate [-ir] keyword
           1. 【i】：忽略大小写的差异。
           2. 【c】：不输出文件名，仅计算找到的文件数量。
           3. 【l】：仅输出几行的意思，例如：输出5行则是 -l 5。
           4. 【S】：输出locate所使用的数据库文件的相关信息，包括该数据库记录的文件/目录数量等。（locate -S）
           5. 【r】：后面可接正则表达式的显示方式。
           6. locate该命令的关键字可以是部分名称。
           7. 限制：

因为该命令是查找已建立的数据库的数据，而数据库是每天更新一次（centos7），所以说如果你是今天创建的文件数据，在今天是不能够通过该命令查找到的。（不手动更新）

直接在命令行输入 updatedb就可以更新locate数据库了。

* + - * 1. updatedb：根据/etc/updatedb.conf的设置去查找系统硬盘内的文件，并更新/var/lib/mlocate内的数据库文件。
        2. locate：依据/var/lig/mlocate内的数据库记录，找出用户所输入关键词的文件名。
      1. find [PATH] [option] [action]
         1. 与时间有关的选项：-atime、-ctime、-mtime，以 -mtime说明：

-mtime 5：意义为在5天之前（一天之内）的被修改过内容的文件。

-mtime +5：列出在5天之前（不含第5天本身）被修改过内容的文件。

-mtime -5：列出在5天之内（不包含第5天那一天）被修改过内容的文件。

-newer file：file为一个存在的文件，列出比file还要新的文件。

* + - * 1. 与使用者或用户组名称有关的参数：

-uid n：n为数字，这个数字是使用者的账号ID，亦即UID，这个UID是记录在/etc/passwd里面与账号名称相对应的数字。

-gid n：n为数字，这个数字是用户组名称的ID，亦即GID，这个GID记录在/etc/group。

-user name：name为使用者账号名称。例如dove

-group name：name为用户组名称。例如dove

-nouser：查找文件的拥有者不在/etc/passwd中的文件。

-nogroup：查找文件的拥有用户组不再/etc/group中的文件。

* + - * 1. 与文件权限及名称有关的参数：

-name filename：查找文件名为filename的文件。

-size [+-] SIZE：查找比SIZE还要大【+】或小【-】的文件。没有+/-就是等于。

-type TYPE：查找类型为TYPE的文件。

TYPE：

f：一般正规文件。

b/c：设备文件。

d：目录。

l：链接文件。

s：socket。

p：FIFL。

-perm mode：查找文件权限【刚好等于】mode的文件。

-perm -mode：查找文件权限【必须要全部囊括mode的权限】的文件。例如：find -perm -0744 那么权限为-rwsr-xr-x的文件也会被列出来，因为该文件已经囊括了-rwxr--r--权限了。

-perm /mode：查找文件权限【包含任一mode的权限】的文件。例如：find -perm +0744 那么权限为-rw-------的文件也会被列出来，因为它有-rw....的权限存在。

* + - * 1. 额外可进行的操作:

-exec command：command为其它命令，exec 后面可再接额外的命令来处理查找到的结果。

-print：将结果打印到屏幕上，这个操作是默认操作。

* + - * 1. 例子：

find /usr/bin -perm /7000 -exec ls -l {} \;

{} 代表的是find命令找到的内容，find找到的结果会被放置到{}位置中。

-exec 一直到【\;】是关键词，代表find额外操作的开始（-exec）到结束（\;），在这之间的就是find命令内的额外操作。在本例中是【ls -l {}】。

因为【;】在bash环境下是由特殊意义的，因此用反斜杠【\】来转义。

1. 重点回顾：
   * 1. 用户能使用的命令是依据PATH变量所规定的目录去查找的。
     2. 查找文件的完整文件名可以使用whereis找特定目录或locate到数据库去查找，而不实际查找文件系统。
     3. 查找命令的完整名可以使用which或type，这两个命令都是通过PATH变量来查找文件名。

**七：Linux磁盘与文件系统管理**

1. 文件系统特性：
   * 1. inode：
        1. 存放文件权限与文件属性等数据。
        2. 一个文件占用一个inode。
        3. 同时记录此文件的内容数据所在的数据区块号码。
        4. 注：
           1. 有inode存在的话，操作系统读取磁盘数据时，可以一口气将多个区块的内容读取出来。（相关联的数据区块之间的关系就像并联）
           2. 没有inode存在的话，比如U盘，它使用的文件系统格式一般时FAT格式，FAT没有inode存在；操作系统读取数据时，就不能一口气将多个区块的内容读取出来了，只能一个接一个去读取，因为下一个区块所在位置记录在上一个区块中。（相关联的数据区块之间的关系就像串联）
     2. 数据区块：
        1. 存放文件的实际内容数据。
        2. 若文件太大时，会占用多个区块。
     3. 超级区块：记录整个文件系统的整体信息。
        1. 包括inode与数据区块的总量、使用量、剩余量等。
        2. 以及文件系统的格式与相关信息。
     4. 如果文件系统的容量大小高达几百GB的话，就会有区块群组这个概念。每个区块群组都有独立的inode、数据区块、超级区块系统。
     5. 文件系统一开始就将inode与数据区块规划好了，除非重新格式化（或者利用resize2fs等命令修改其大小），否则inode与数据区块固定后就不再变动。
     6. 区块群组的6个主要内容：
        1. 数据区块：是用来放置文件数据的地方，在文件系统格式化的时候区块大小就固定了，且每个区块都有编号，方便inode记录。
           1. 在ext2文件系统的区块的限制：

区块的大小与数量在文件系统格式化完就不能够再修改（除格式化外）

每个区块内最多只能够放置一个文件的数据。

承上，如果文件大于区块大小，则一个文件会占用多个区块数量。

承上，若文件小于区块，则该区块的剩余容量就不能够再被其他文件使用了（磁盘空间会浪费）

* + - 1. Inode table（inode表），记录内容如下：
         1. 该文件的读写属性。
         2. 该文件的拥有者与用户组。
         3. 该文件的大小。
         4. 该文件建立或状态改变的时间（ctime）。
         5. 最近一次的读取时间（atime）。
         6. 最近的修改时间（mtime）。
         7. 定义文件特性的标识（flag），如SetUID。
         8. 特性如下：

inode的数量与大小也是在文件系统格式化完成时就已经固定了。

每个inode大小均固定为128B（新的ext4与xfs可设置到256B）

每个文件都仅会占用一个inode而已。

承上，因此文件系统能够建立的文件数量与inode的数量有关。

系统读取文件时需要先找到inode，并分析inode所记录的权限与用户是否相符，若符合才能读取区块的内容。

* + - 1. Superblock（超级区块）：
         1. 它是记录整个文件系统相关信息的地方，若没有超级区块，就没有这个文件系统。
         2. 记录的信息主要如下：

数据区块与inode的总量。

未使用与已使用的inode与数据区块数量。

数据区块与inode的大小（block为1k/2k/4k；inode为128B/256B）

文件系统的挂载时间、最近一次写入数据的时间、最近一次检验磁盘的时间等文件系统的相关信息。

一个有效位数值，若此文件系统已被挂载，则有效位为0；若未被挂载，则有效位为1。

* + - 1. Filesystem Description（文件系统描述说明）：
         1. 这个区段可以描述每个区块群组的开始与结束的区块，以及说明每个区段（超级区块、区块对照表、inode对照表、数据区块）分别介于哪一个区块之间，这部分也能够用【dumpe2fs】来观察。
      2. 区块对照表（block bitmap）：
         1. 记录使用与未使用的区块号码。
      3. inode对照表（inode bitmap）：
         1. 记录使用与未使用的inode号码。
      4. dumpe2fs：查询ext系列超级区块信息的命令。

1. 注意：每个区块群组都可能含有超级区块，但是我们也说一个文件系统应该仅有一个超级区块而已，怎么回事呢？事实上除了第一个区块群组内会含有超级区块之外，后续的区块群组不一定含有超级区块，而若含有超级区块则该超级区块主要是作为第一个区块群组内超级区块的备份，这样可以进行超级区块的恢复。
2. inode与目录树的关系：
   * 1. 目录：
        1. inode记录该目录的相关权限和属性，并可记录分配到的区块的号码
        2. 数据区块记录在这个目录下的文件名以及该文件名占用的inode号码数据。
     2. 目录树读取：
        1. 比如读取/etc/passwd这个文件
           1. 通过挂载点的信息找到inode号码为xxx的根目录inode，通过该inode可获取根目录 / 的数据区块的号码。
           2. 通过读取a获取的区块号码所对应的数据区块，可找到 /etc的 inode号码。
           3. 读取由b步骤获得的inode号码所对应的inode可以获取/etc的数据区块号码。
           4. 读取由c步骤获取的数据区块号码所对应的数据区块 ，可找到passwd文件所对应的inode号码。
           5. 通过读取由d步骤获取到的inode号码所对应的inode，可以获取passwd文件的数据区块号码。
           6. 通过e步骤获取的数据区块号码，可以找到对应的数据区块读取数据。
3. 新建文件/目录：
   * 1. 检查用户对于欲新建文件/目录的的目录是由具有w与x的权限，有的话才能新增。
     2. 对照inode表找到没有使用的inode号码，并将新文件/目录的权限和属性写入。
     3. 对照区块表找到没有使用的区块号码，并将实际数据写入区块中，且更新inode的区块指向数据。
     4. 将刚刚写入的inode与区块数据同步更新inode对照表和区块对照表，并且更新超级区块的内容。
4. 将文件系统与目录树结合的操作我们称为【挂载】。
5. 挂载点一定是目录，该目录为进入该文件系统的入口。
6. xfs 文件系统的配置
   * 1. 基本上xfs就是一个日志式文件系统。
     2. xfs 文件系统在数据的分布上，主要规划为三个部分：
        1. 数据区（data section）
           1. inode、数据区块、超级区块等数据都放在这个区块。
           2. 这个数据区也分为了多个存储区群组（和区块群组类似）来分别放置文件系统所需要的数据。每个存储区群组都包含了：

整个文件系统的超级区块

剩余空间的管理机制

inode的分配与追踪

* + - * 1. inode与区块都是系统需要用到时才动态配置产生，所以格式化操作超级快。
      1. 文件系统活动登陆区（log seciton）
         1. 这个区域主要被用来记录文件系统的变化，有点像是日志区。
         2. 文件的变化会在这里记录下来，知道该变化完整的写入到整个数据区后，该条记录才会被结束。
         3. 在这个区域中，你可以指定外部磁盘来作为xfs文件系统的日志区块。比如将一个SSD磁盘作为xfs的登陆区。
      2. 实时运行区（realtime section）
         1. 当有文件要被建立时，xfs会在这个区段里面找一个到数个的extent区块，将文件放置在这个区块内，等到分配完毕后，再写入到data section 的inode与区块中。
         2. extent 区块的大小要在格式化的时候就先指定，最小在为4k，最大可达1G。
    1. 可以使用【xfs\_info】命令去观察超级区块的内容。

1. 文件系统的简单操作