Linux文件系统

# 文件系统的基本概念

文件：具有名字的一组相关信息的有序集合

文件系统：是操作系统的一部分。负责管理系统中的文件，为用户提供使用文件的操作接口。文件系统以磁盘分区划分，每个分区对应一个独立的文件系统。Linux的文件系统结构。



挂载点：磁盘分区挂载在文件树的某一个目录。

Inode：文件储存在硬盘上，硬盘的最小存储单位叫做"扇区"（Sector）。每个扇区储存512字节（相当于0.5KB）。操作系统读取硬盘的时候，不会一个个扇区地读取，这样效率太低，而是一次性连续读取多个扇区，即一次性读取一个"块"（block）。这种由多个扇区组成的"块"，是文件存取的最小单位。"块"的大小，最常见的是4KB，即连续八个 sector组成一个 block。这种储存文件元信息的区域就叫做inode，中文译名为"索引节点"。Inode包含的内容：

* 文件的字节数
* 文件拥有者的User ID
* 文件的Group ID
* 文件的读、写、执行权限
* 文件的时间戳，共有三个：ctime指inode上一次变动的时间，mtime指文件内容上一次变动的时间，atime指文件上一次打开的时间。
* 链接数，即有多少文件名指向这个inode
* 文件数据block的位置

inode也会消耗硬盘空间，所以硬盘格式化的时候，操作系统自动将硬盘分成两个区域。一个是数据区，存放文件数据；另一个是inode区（inode table），存放inode所包含的信息。每个inode节点的大小，一般是128字节或256字节。inode节点的总数，在格式化时就给定，一般是每1KB或每2KB就设置一个inode。使用df –i查看使用情况。Unix/Linux系统内部不使用文件名，而使用inode号码来识别文件。

用户通过文件名，打开文件的实际过程：首先，系统找到这个文件名对应的inode号码；其次，通过inode号码，获取inode信息；最后，根据inode信息，找到文件数据所在的block，读出数据。

# 文件系统的类型

## 磁盘文件系统

ext2：是一个ext文件系统的升级版本，Redhat Linux 7.2版本以前系统默认的文件系统。1993年发布，最大支持16TB分区和最大2TB文件。

ext3：是ext2的升级版本，最大区别就使附带日志功能，以在系统突然停止时候提高文件系统的可靠性。最大支持16TB分区和最大2TB文件。

ext4：是ext3文件系统的升级版。在性能上，伸展性和可靠性方面进行了大量改进。EXT4的变化可以说是翻天覆地的，比如向下兼容EXT3，最大1EB文件系统和16TB文件，无线数量的子目录，Extends连续数据块概念，多块分配，延迟分配，持久预分配，快速FSCK，日志校验，无日志默认，在线碎片整理，inode增强默认开启barrier等。（1EB=1024\*1024TB（PB））

## 网络文件系统（NFS,SMB,GFS）

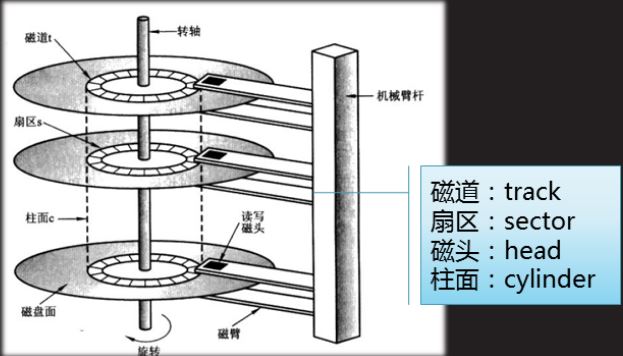
## 数据库文件系统

## 特殊文件系统

# Linux磁盘的基本概念

## 磁盘的物理组织

|  |
| --- |
| 磁盘：又叫硬盘，由一片或几片圆形盘片组成。  磁片：圆形盘片，表面度有磁性物质，磁化后可以进行存储数据。  磁头：每个盘片有两个面，每个面都有一个读写磁头，用于寻址。  磁道：每个磁片的若干个同心圆。最外面的称为0磁道。  扇区：每个磁道的螺杆个扇区，1号扇区,2号扇区，，，一个扇区存储了 512B 的  二进制信息。是CPU进行读写的最小单位。  柱面：由每个磁盘上同号磁道组成。 |



## Linux硬盘的基本概念

有关硬盘类型的识别，linux中根据设备的类型可以对存储设备进行识别。对于IDE存储设备，linux会识别为hd.第一个IDE设备会被识别为hda,第二个IDE设备会被识别为hdb。如果是SATA,USB,SCSI设备，会被识别为sd。第一个此类设备会被识别为sda,第二个此类设备会被识别为sdb。

数据在硬盘上存储：文件系统是建立在分区上的，它规定了文件存取的规则。创建分区是，就已经设置好了硬盘的各项物理参数，指定了硬盘主引导记录（MBR）和引导记录备份的存放位置。

硬盘分区格式的种类：

* 存取分区格式：FAT16，FAT32，NTFS,Ext2,和Ext3格式。
* 磁盘分区格式：linux Native，linux Swap

Linux磁盘的交换分区：在物理内存充满时使用。当物理内存充满时，不活跃的页就会被移到交换区。一般设置交换区的大小为计算机的两倍且不超过2048M

# 常用磁盘管理命令

## df命令

|  |
| --- |
| 命令格式：df [参数] [文件或磁盘设备]  英文缩写：Disk Free  功能：查看文件系统磁盘使用情况  常用参数：   * -i：显示磁盘inode使用量信息 * -h：人性化显示容量信息 * -T：显示文件系统类型。   常用命令：  df –ih 报告索引节点inode的使用信息 |

## du命令

|  |
| --- |
| 命令格式：du [参数][目录或文件]  英文缩写：disk usage  功能：查看文件目录占磁盘的大小  常用命令：  du –sh 查看当前目录占用磁盘的大小 |

## dd命令的使用

dd：用指定大小的块拷贝一个文件，并在拷贝的同时进行指定的转换

参数：

* if=文件名：输入文件名，缺省为标准输入。即指定源文件。< if=input file >
* of=文件名：输出文件名，缺省为标准输出。即指定目的文件。< of=output file >
* ibs=input\_bytes：一次读入input\_bytes个字节，即指定一个块大小为bytes个字节。obs=output\_bytes：一次输出output\_bytes个字节，即指定一个块大小为bytes个字节。bs=bytes：同时设置读入/输出的块大小为bytes个字节。
* cbs=bytes：一次转换bytes个字节，即指定转换缓冲区大小。
* skip=blocks：从输入文件开头跳过blocks个块后再开始复制。
* seek=blocks：从输出文件开头跳过blocks个块后再开始复制。

注意：通常只用当输出文件是磁盘或磁带时才有效，即备份到磁盘或磁带时才有效。

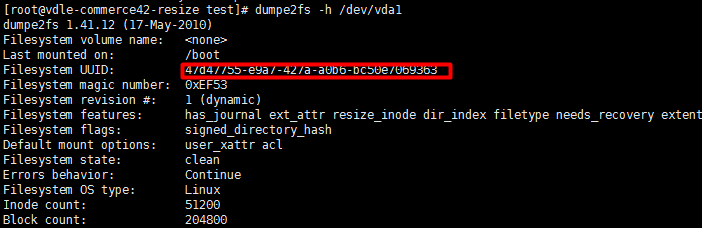
* count=blocks：仅拷贝blocks个块，块大小等于ibs指定的字节数。
* conv=conversion：用指定的参数转换文件。

|  |
| --- |
| #创建一个1024M的空文件  dd if=/dev/zero of=/home/test/hello.txt bs=1024M count=1  #测试磁盘写能力  time dd if=/dev/zero of=/test.dbf bs=8k count=300000  因为/dev/zero是一个伪设备，它只产生空字符流，对它不会产生IO，所以，IO都会集中在of文件中，of文件只用于写，所以这个命令相当于测试磁盘的写能力。  #测试磁盘读能力  time dd if=/dev/sdb1 of=/dev/null bs=8k  因为/dev/sdb1是一个物理分区，对它的读取会产生IO，/dev/null是伪设备，相当于黑洞，of到该设备不会产生IO，所以，这个命令的IO只发生在/dev/sdb1上，也相当于测试磁盘的读能力。  #文件复制功能  dd if=file1 of=file2 |
| /dev/null——它是空设备，也称为位桶（bit bucket）。任何写入它的输出都会被抛  弃。如果不想让消息以标准输出显示或写入文件，那么可以将消息重定向到位桶。  /dev/zero，是一个输入设备，你可你用它来初始化文件。该设备无穷尽地提供ASCII 0(就是NULL)，可以使用任何你需要的数目——设备提供的要多的多。他可以用于向设备或文件写入字符串0。 |

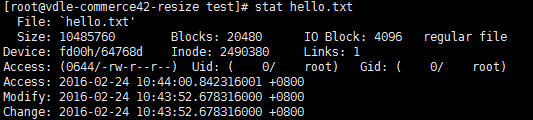
## dumpe2fs命令

#检查磁盘状态命令

dumpe2fs -h 分区设备文件名



## 查看文件详细时间参数：stat



## 校验文件md5值：md5sum 文件名

## 检测修复文件系统：fsck、e2fsck

单用户模式卸载文件系统后执行（一般不需要用，系统会自动修复）

fsck常用选项：

* -a :不用显示用户提示，自动修复文件系统
* -y：自动修复。和-a一致，不过有些文件系统只支持——y

注意：磁盘要处于未挂载状态。

# Linux磁盘分区

硬盘限制，分区表可以分一到四个主分区（Primary），或者包含一个扩展分区（Extended）和一到三个主分区。主分区和扩展分区总共不能超过四个，因为操作系统的限制，扩展分区不能超过一个。其中，扩展分区可以再继续分区，叫做逻辑分区。



***主分区的作用：***就是计算机用来进行启动操作系统的，因此每个操作系统的启动，或者称作是引导程序，都应该存放在主分区上。***扩展分区的目的***是使用额外的扇区来记录分区信息，不能直接使用，只有分成逻辑分区才能被格式化后作为数据访问的分区。：我们的面不能吃，但是面做成了饺子或者饼就能吃。

4.在Linux下，我们根据系统中的设备文件名，如/dev/hda3和/dev/hda5来使用扩展分区和逻辑分区。（逻辑分区的设备名称号码由5开始）

## fdisk –l

#显示所有分区信息

/dev/fd0(第一个软盘) 、dev/cdrom（第一个光盘）

/dev/hda（第一个IDE硬盘）、dev/sda(第一个SCSI硬盘)

可以参看磁盘的信息，然后是分区信息。分别是分区名，是否是启动分区（\*），起始柱面，终止柱面，分区的总块数，分区id，分区类型。

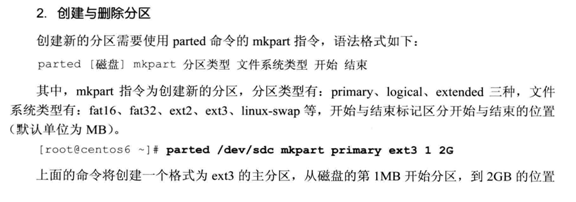
## fdisk /dev/hdb

|  |  |
| --- | --- |
| #创建分区  m(显示帮助信息) ，  n（添加分区p/e ,size可以为柱面数也可以是+…K/M/G），  l(显示已知的分区类型)，  t(改变分区类型83主分区，5扩展分区),  d(删除分区)，p(显示当前分区情况)，  w(保存退出)，  q（不保存退出）  注意：使用命令，使内核立即读取分区表而不用重启。   |  | | --- | | partprobe /dev/xvdb | |

## GPT分区

parted 选项 [磁盘 [命令]]





## 在分区上建立文件系统

|  |
| --- |
| mkfs –t ext3(文件系统类型) /dev/hda(设备文件)  mkfs.ext4 /dev/xvdb  mkswap /dev/xvdb #格式化文件系统为swap类型 |

# 挂载和卸载文件系统

## 挂载命令

mount -l(查询当前系统已经挂载的设备，-l显示卷标)

mount -a（根据/etc/fstab的内容，自动挂载）

mount命令格式： mount  [-t  vfstype]  [-o  options]  device  dir

-t 常用文件系统：

*  光盘或光盘镜像：iso9660
*  DOS fat16文件系统：msdos
*  Windows 9x fat32文件系统：vfat
*  Windows NT ntfs文件系统：ntfs
*  Mount Windows文件网络共享：smbfs
*  UNIX(LINUX) 文件网络共享：nfs
*  Ext2,3,4

-o option主要用来秒速设备的挂接方式。ro,rw,remount，noexec等

device 要挂接(mount)的设备文件名。

dir：设备在系统上的挂接点(mount point)。

### 挂接光盘镜像文件

|  |
| --- |
| #创建光盘挂载点（/media系统默认）  #挂载光盘  mount -t iso9660 /dev/cdrom /media  #卸载光盘  unmount /media或者/dev/cdrom |

### 挂接U盘

|  |
| --- |
| 创建U盘挂载点：mkdir /mnt/usb  查看usb设备文件名：fdisk –l  挂载U盘：mount -t vfat /dev/sdb1 /mnt/usb  卸载u盘：unmount /dev/sdb1或者/mnt/usb  注意：linux不会识别ntfs文件格式 |

## 卸载文件系统

unmount 设备文件名/挂载点

注意：umount的时候老显示 device busy？

#选项 –l 并不是马上umount，而是在该目录空闲后再umount。

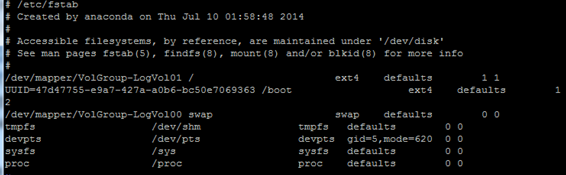
umount -l /mnt/hda5

## 文件系统的自动安装

可以通过配置/etc/fstab来自动安装，linux系统启动后，init进程会自动读取/etc/fstab配置文件的内容，并挂载相应的文件系统。

字段一 字段二 字段三 字段四 字段五 字段六

文件系统 挂载点 文件系统类型 命令选项 备份频率 检查次序



### /etc/fstab文件解析

第一个字段：分区设备文件名或者UUID(硬盘通用唯一识别码)

第二个字段：挂载点

第三个字段：默认的文件系统

第四个字段：挂载参数（defaults)

第五个字段：指定分区是否被dump备份，0代表不备份，系统1代表每天备份，2代表不定期备份（每个分区的lost+found)

第六个字段：指定分区是否被fsck检测，0代表不检测，其他数字代表检测的优先级，数字越小优先级越高

例如：UUID=6634633e-001d-43ba-8fab-202f1df93339 / ext4 defaults,barrier=0 1 1

注意：UUID是真正的唯一标志符,UUID为系统中的存储设备提供唯一的标识字符串，不管这个设备是什么类型的。如果你在系统中添加了新的存储设备如硬盘，很可能会造成一些麻烦，比如说启动的时候因为找不到设备而失败，

而使用UUID则不会有这样的问题；设备名并非总是不变的自动分配的设备名称并非总是一致的，它们依赖于启动时内核加载模块的顺序。如果你在插入了USB盘时启动了系统，而下次启动时又把它拔掉了，就有可能导致设备名分配不一致。通过命令来查看：dumpe2fs -h 分区设备文件名

### 挂载参数

* auto:系统启动时自动挂载文件系统
* defaults:默认挂在文件系统，启动时自行挂载并可读可写
* grpquata:该文件系统支持组配额管理
* noauto:启动时不自动挂载文件系统，用户需要手动挂载
* ro:以只读的方式挂载该文件系统
* rw:以读写的方式挂载文件系统
* usrquota:该文件系统支持用户配额管理
* user:允许普通用户安装该文件系统
* noexec：不允许该文件系统运行程序

### 技巧

#检测/etc/fstab ：mount -a /etc/fstab

#文件修复：mount -o remount，rw /

### SWAPFILE实现

swap分区：新建分区（分区id改成82）；

然后格式化：mkswap 分区设备文件名；

挂载：swapon 分区设备文件名;

取消swap分区：swapoff 分区设备文件名

通过/etc/fstab永久挂载： 分区设备文件名 swap swap defaults 0 0

# 磁盘配额的设置

## 检查内核是否支持quota

dmesg | grep quota

## 修改/etc/fstab文件

添加命令选项usrquota,grpquota

/dev/sdb /mnt/test ext3 defaults,usrquota,grpquota 1 2

当配置错误了应该使用：

mount -o remount,rw / #以可读写方式重新挂载文件系统。

## 重新启动系统

init 6

## 使用命令建立aquota.user,aquota.group文件

setenforce 0

quatacheck –avug//a检查所有已安装并配置配额的文件系统

//g检查组的配额，u检查用户的配额，v显示检查是产生的信息。

## 设置用户配额

edquota –u user #设置用户的配额

edquota –g group #设置用户组配额

edquota –p user/group(模板） user/group #复制配额给另一个用户/组

* 软限制（Soft limit）：定义用户可以占用的磁盘空间数。当用户超过该限制后会收到以超过 配额的警告。
* 硬限制（Hard limit）：当用户试图将文件存放在其已经超过该限制值目录时，报告文件系统 错误。
* 宽限期（Grace period）：定义用户在软限制下可以使用其文件系统的期限。

## 启动配额管理

quotaon –avug #启动磁盘配额

quotaoff –avug #关闭磁盘配额

管理员查看配额信息：repquota –a

# LVM逻辑卷

## LVM逻辑卷概述

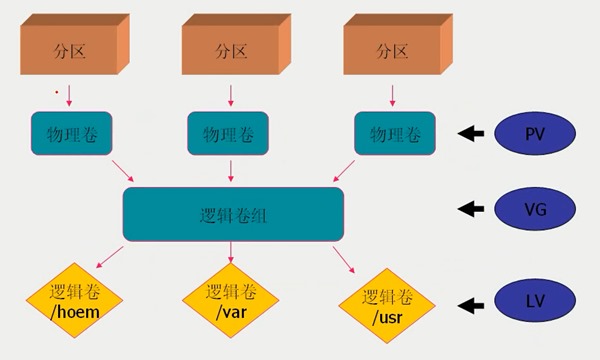
LVM是逻辑盘卷管理（Logical Volume Manager）的简称，它是Linux环境下对磁盘分区进行管理的一种机制，LVM是建立在硬盘和分区之上的一个逻辑层，来提高磁盘分区管理的灵活性。

LVM的工作原理其实很简单，它就是通过将底层的物理硬盘抽象的封装起来，然后以逻辑卷的方式呈现给上层应用。在传统的磁盘管理机制中，我们的上层应用是直接访问文件系统，从而对底层的物理硬盘进行读取，而在LVM中，其通过对底层的硬盘进行封装，当我们对底层的物理硬盘进行操作时，其不再是针对于分区进行操作，而是通过一个叫做逻辑卷的东西来对其进行底层的磁盘管理操作。比如说我增加一个物理硬盘，这个时候上层的服务是感觉不到的，因为呈现给上层服务的是以逻辑卷的方式。

LVM最大的特点就是可以对磁盘进行动态管理。因为逻辑卷的大小是可以动态调整的，而且不会丢失现有的数据。如果我们新增加了硬盘，其也不会改变现有上层的逻辑卷。作为一个动态磁盘管理机制，逻辑卷技术大大提高了磁盘管理的灵活性。

LVM逻辑卷常用的几个概念：

1. PV（Physical Volume）- 物理卷 ：物理卷在逻辑卷管理中处于最底层，它可以是实际物理硬盘上的分区，也可以是整个物理硬盘，也可以是raid设备。
2. VG（Volumne Group）- 卷组 ：卷组建立在物理卷之上，一个卷组中至少要包括一个物理卷，在卷组建立之后可动态添加物理卷到卷组中。一个逻辑卷管理系统工程中可以只有一个卷组，也可以拥有多个卷组。
3. LV（Logical Volume）- 逻辑卷 ：逻辑卷建立在卷组之上，卷组中的未分配空间可以用于建立新的逻辑卷，逻辑卷建立后可以动态地扩展和缩小空间。系统中的多个逻辑卷可以属于同一个卷组，也可以属于不同的多个卷组。
4. PE（Physical Extent）- 物理块：LVM 默认使用4MB的PE区块，而LVM的LV最多仅能含有65534个PE (lvm1 的格式)，因此默认的LVM的LV最大容量为4M\*65534/(1024M/G)=256G。PE是整个LVM 最小的储存区块，也就是说，其实我们的资料都是由写入PE 来处理的。简单的说，这个PE 就有点像文件系统里面的block 大小。所以调整PE会影响到LVM的最大容量！不过，在 CentOS 6.x 以后，由于直接使用 lvm2 的各项格式功能，因此这个限制已经不存在了



## LVM命令

将分区转换成物理卷的命令是pvcreate;将物理卷装换为卷组的命令是vgcreate;从卷组中提取容量创建逻辑卷是lvcreate。创建后分别使用pvdisplay,vgdisplay,lvdisplay查看效果。

### pvcreate命令

描述：LVM对磁盘或者分区进行初始化

用法：pvcreate [选项] 物理卷 [物理卷…]

例子：pvcreate /dev/xvdb2

### vgcreate命令

描述：创建卷组

用法：vgcreate [选项] 卷组名称 物理设备路径 [物理设备路径]

例子：vgcreate –s 16M test\_vg /dev/xvdb2 #设置pe为16M

### lvcreate命令

描述：在已经存在的卷组上创建逻辑卷

用法：lvcreate [选项] 卷组名称或路径 [物理设备路径]

选项：

* -l：指定在VG中使用多少个PE创建逻辑卷
* -L：直接指定逻辑卷容量（B,K,M,G,T,P小写也可以）
* -n：指定逻辑卷名称

#指定使用/dev/xvdb2这个物理存储空间

例子：lvcreate –L 2G –n test\_lv1 test\_vg /dev/xvdb2

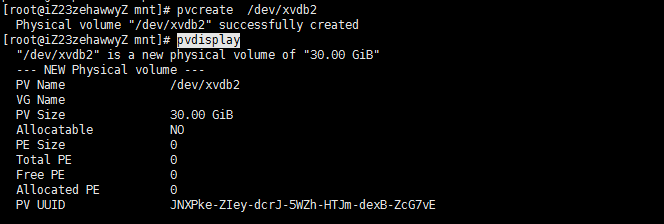
## LVM逻辑卷的使用

### 创建LVM分区实例

#创建物理卷

pvcreate /dev/xvdb2

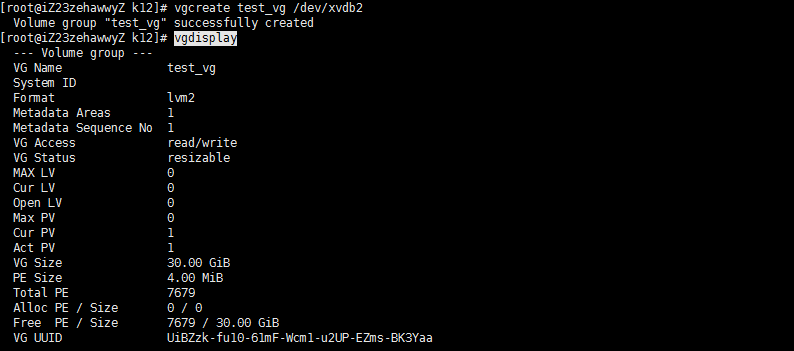
pvdisplay



#创建卷组

vgcreate test\_vg /dev/xvdb2

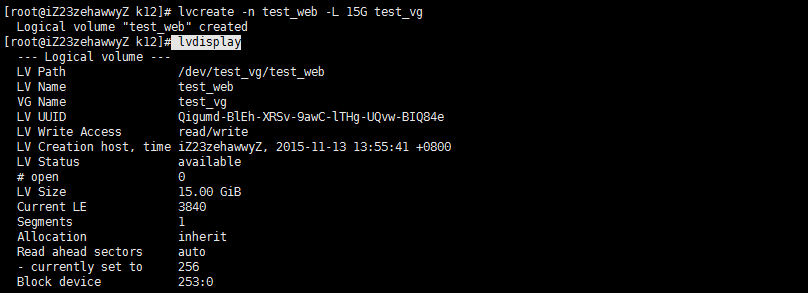
vgdisplay



#创建逻辑卷

lvcreate -n test\_web -L 15G test\_vg

lvdisplay



#格式化并挂载逻辑卷

mkfs.ext4 /dev/test\_vg/test\_web

mount /dev/test\_vg/test\_web /test

### 修改LMV分区容量

#利用剩余卷组空间直接扩展LVM分区

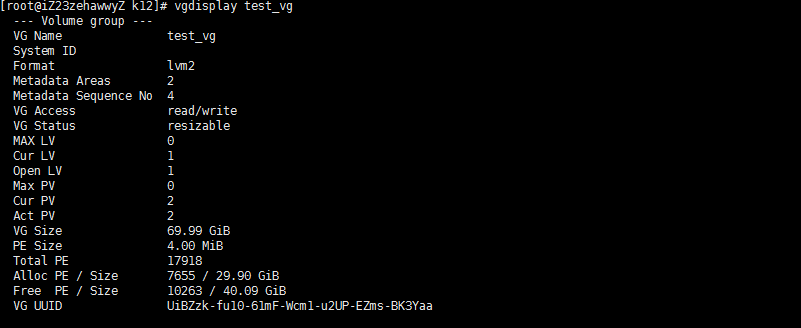
lvextend -L +15G /dev/test\_vg/test\_web

resize2fs /dev/test\_vg/test\_web

#加入分区到卷组扩展LVM分区

pvcreate /dev/xvdb3

vgextend test\_vg /dev/xvdb3



vgextend -L 69.99G /dev/test\_vg/test\_web

resize2fs /dev/test\_vg/test\_web

### 删除LMV逻辑分区

#顺序和使用逻辑卷完全相反

|  |
| --- |
| #卸载文件系统  umount /test  #删除逻辑卷  lvremove /dev/test\_vg/test\_web –y  #删除卷组  vgremove test\_vg –y  #删除物理卷  pvremove /dev/xvdb{2,3} |

# RAID磁盘矩阵

## RAID磁盘阵列概述

RAID是英文Redundant Array of Independent Disks的缩写，中文简称为独立冗余磁盘阵列。简单的说，RAID是一种把多块独立的硬盘（物理硬盘）按不同的方式组合起来形成一个硬盘组（逻辑硬盘），从而提供比单个硬盘更高的存储性能和提供数据备份技术。组成磁盘阵列的不同方式成为RAID级别（RAID Levels）。在用户看起来，组成的磁盘组就像是一个硬盘，用户可以对它进行分区，格式化等等。总之，对磁盘阵列的操作与单个硬盘一模一样。不同的是，磁盘阵列的存储速度要比单个硬盘高很多，而且可以提供自动数据备份。数据备份的功能是在用户数据一旦发生损坏后，利用备份信息可以使损坏数据得以恢复，从而保障了用户数据的安全性。

RAID分为软件RAID和硬件RAID。软RAID的功能和硬件RAID相同的功能。由于没有独立的硬件控制设备，所以性能不如硬件RAID。软RAID实现简单，不需要额外的硬件设备。硬件RAID通常需要RAID卡，RAID卡有自己的控制部件和内存所以不会占用过多的系统 资源，效率高性能强。

RAID的特点：

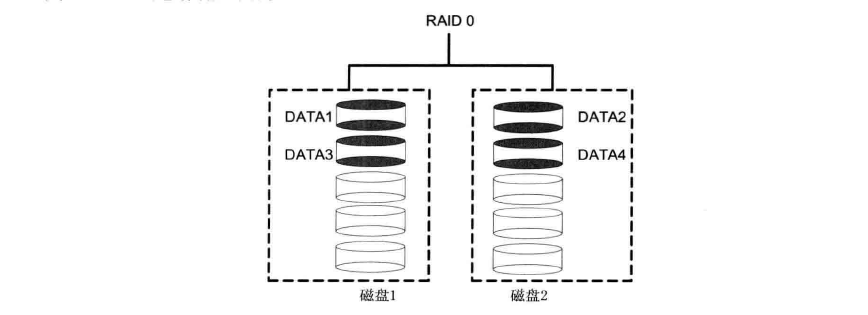
* 通过把多个磁盘组织在一起作为一个逻辑卷提供磁盘跨越功能；
* 通过把数据分成多个数据块（Block）并行写入/读出多个磁盘以提高访问磁盘的速度；
* 通过镜像或校验操作提供容错能力；

## RAID级别

stripe width是指可被并行写入的 stripe 的个数，即等于磁盘阵列中磁盘的个数。STRIPE SIZE也可称为block size(chunk size，stripe length，granularity)，指写入每个磁 盘的数据块大小。

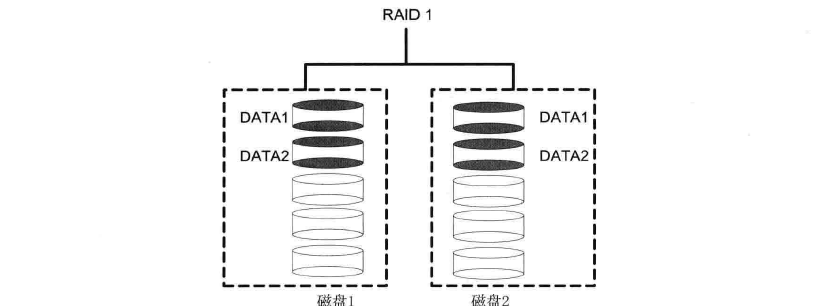
### RAID0(不含校验与冗余的条带存储)

RAID 0连续以位或字节为单位分割数据，并行读/写于多个磁盘上，因此具有很高的数据传输率，但它没有数据冗余，因此并不能算是真正的RAID结构。RAID 0只是单纯地提高性能，并没有为数据的可靠性提供保证，而且其中的一个磁盘失效将影响到所有数据。因此，RAID 0不能应用于数据安全性要求高的场合。



### RAID1(不含校验的镜像存储)

它是通过磁盘数据镜像实现数据冗余，在成对的独立磁盘上产生互 为备份的数据。当原始数据繁忙时，可直接从镜像拷贝中读取数据，因此RAID 1可以提高读取性能。RAID 1是磁盘阵列中单位成本最高的，但提供了很高的数据安全性和可用性。当一个磁盘失效时，系统可以自动切换到镜像磁盘上读写，而不需要重组失效的数据。

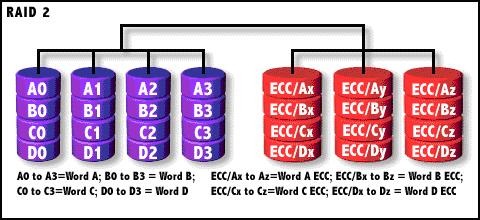


### RAID2(位级别的校验式条带存储)

RAID0的改良版，加入了汉明码(Hanmming Code)错误校验。汉明码能够检测最多两个同时发生的比特错误，并且能够更正单一比特的错误。汉明码的位 数与数据的位数有一个不等式关系，即: 2^P ≥ P + D +1。P代表汉明码的个数，D代表数据位的个数，比如4位数据需要3位汉明码，7位数据需要4位汉 明码，64位数据时就需要7位汉明码。RAID2是按1bit来分割数据写入的，而P:D就代表了数据 盘与校验盘的个数。所以如果数据位宽越大，用于校验的盘的比例就越小。由于汉明码能够 纠正单一比特的错误，所以当单个磁盘损坏时，汉明码便能够纠正数据。

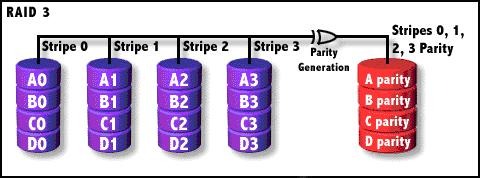
RAID 2 因为每次读写都需要全组磁盘联动，所以为了最大化其性能，最好保证每块磁盘主 轴同步，使同一时刻每块磁盘磁头所处的扇区逻辑编号都一致，并存并取，达到最佳性能。 如果不能同步，则会产生等待，影响速度。

如果一个文件要写入RAID2，则该文件会被分成数据位分别同步写入不同的磁盘中。DATA1与DATA2进行海明码运算后在写入校验盘中。拥有校验位的RAID2允许三块磁盘任意两块磁盘损坏，能够通过另外两块磁盘进行运算还原。RAID2对大数据量写入具有很高的性能。至少需要三块硬盘。



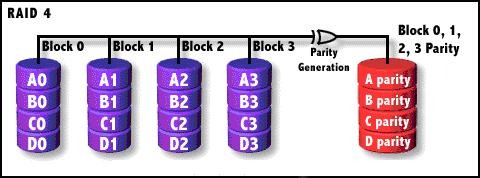
### RAID3(字节级别的校验式条带存储)

类似于RAID2，数据条带化(stripe)存储于不同的硬盘，数据以字节为单位，只是RAID3使用单块磁盘存储简单的奇偶校验信息，所以最终磁盘数量为 N+1 。当这N+1个硬盘中的其中一个硬盘出现故障时， 从其它N个硬盘中的数据也可以恢复原始数据，当更换一个新硬盘后，系统可以重新恢复完整 的校验容错信息。



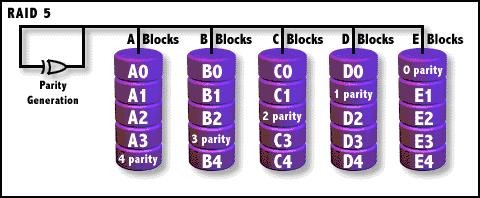
### RAID4(数据块级别的校验式条带存储)

与RAID3类似，但RAID4是按块(扇区)存取。无须像RAID3那样，哪怕每一次小I/O操作也要涉 及全组，只需涉及组中两块硬盘（一块数据盘，一块校验盘）即可，从而提高了小量数据 I/O速度。



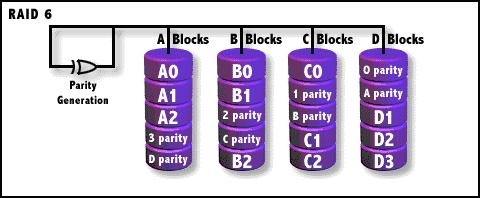
### RAID5(数据块级别的分布式校验式条带存储)

RAID 5不单独指定的奇偶盘，而是在所有磁盘上交叉地存取数据及奇偶校验信息。RAID5把数据和相对应的奇偶校验信息存储到组成RAID5的各个磁盘上，并且奇偶校验信息和 相对应的数据分别存储于不同的磁盘上，其中任意N-1块磁盘上都存储完整的数据，也就是 说有相当于一块磁盘容量的空间用于存储奇偶校验信息。因此当RAID5的一个磁盘发生损坏 后，不会影响数据的完整性，从而保证了数据安全。当损坏的磁盘被替换后，RAID还会自动 利用剩下奇偶校验信息去重建此磁盘上的数据，来保持RAID5的高可靠性。



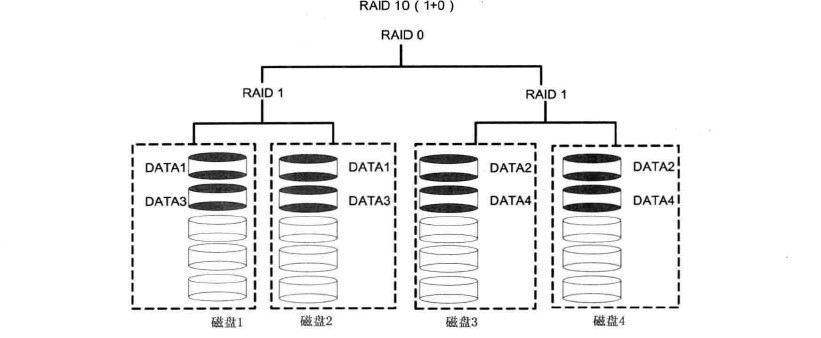
### RAID6

类似RAID5，但是增加了第二个独立的奇偶校验信息块，两个独立的奇偶系统使用不同的算法， 数据的可靠性非常高，即使两块磁盘同时失效也不会影响数据的使用。但RAID 6需要分配给 奇偶校验信息更大的磁盘空间，相对于RAID 5有更大的“写损失”，因此“写性能”非常差。



### RAID10

也被称为RAID 10标准，实际是将RAID 0和RAID 1标准结合的产物，在连续地以位或字节为单位分割数据并且并行读/写多个磁盘的同时，为每一块磁盘作磁盘镜像进行冗余。它的优点是同时拥有RAID 0的超凡速度和RAID 1的数据高可靠性，但是CPU占用率同样也更高，而且磁盘的利用率比较低。



## 创建和管理软RAID实例

### 创建RAID

Centos中mdadm命令可以用于创建RAID

名称：mdadm

描述：管理linux软件RAID设备

用法：mdadm [模式] 选项 <raid设备>

选项：

* -C：创建软件RAID
* -l：指定RAID级别
* -n：指定磁盘个数
* -x：指定备用设备个数

#创建RAID

mdadm -C /dev/md0 -l 0 -n 2 /dev/xvdb2 /dev/xvdb1

mdadm -C /dev/md1 -l 5 -n 2 /dev/xvdb3 /dev/xvdb4

#查看RAID

mdadm --detail /dev/md0

|  |
| --- |
| /dev/md0:  Version : 1.2  Creation Time : Fri Nov 13 17:16:27 2015 #创建时间  Raid Level : raid0 #RAID级别  Array Size : 62913536 (60.00 GiB 64.42 GB) #磁盘空间  Raid Devices : 2 #磁盘个数  Total Devices : 2  Persistence : Superblock is persistent  Update Time : Fri Nov 13 17:16:27 2015  State : clean  Active Devices : 2 #活动磁盘个数  Working Devices : 2 #工作磁盘个数  Failed Devices : 0 #错误磁盘个数  Spare Devices : 0 #空闲磁盘个数  Chunk Size : 512K  Name : iZ23zehawwyZ:0 (local to host iZ23zehawwyZ)  UUID : 4e88461a:10ccc830:28eba61e:df215e87  Events : 0  Number Major Minor RaidDevice State  0 202 18 0 active sync /dev/xvdb2  1 202 17 1 active sync /dev/xvdb1 |

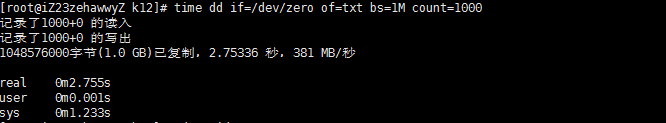
### 格式化和挂载

|  |
| --- |
| #临时挂载  mkfs.ext4 /dev/md0  mkfs.ext4 /dev/md1  mkdir /raid0  mkdir /raid5 mount /dev/md0 /raid0  mount /dev/md1 /raid5  #磁盘矩阵设置为开机自动挂载  echo "DEVICE /dev/xvdb1 /dev/xvdb2 /dev/xvdb3 /dev/xvdb4" > /etc/mdadm.conf  mdadm -Evs >> /dev/mdadm.conf  echo "/dev/md0 /raid0 ext4 defaults 0 0">> /etc/fstab  echo "/dev/md1 /raid5 ext4 defaults 0 0">> /etc/fstab |

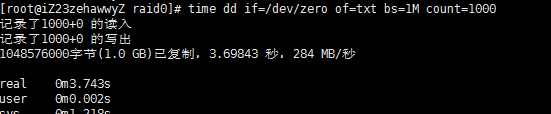
## RAID性能测试

time dd if=/dev/zero of=txt bs=1M count=1000

#普通磁盘



#raid0



#raid5



## RAID故障模拟

mdadm /dev/md1 -f /dev/xvdb4

mdadm --detail /dev/md1

