                                    一.RAID篇

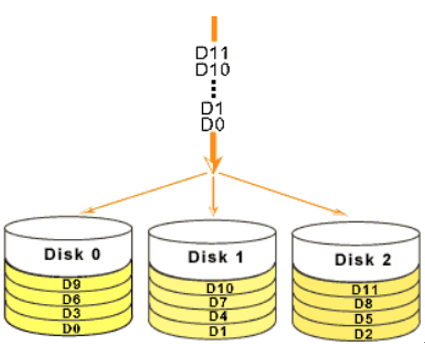
一．什么是RAID

磁盘阵列全名是『Redundant Arrays of Inexpensive Disks, RAID 』，英翻中的意思是：容错式廉价磁盘阵列。 RAID 可以透过一个技术(软件或硬件)，将多个较小的磁盘整合成为一个较大的磁盘装置；而这个较大的磁盘功能可不止是储存而已，他还具有数据保护的功能呢。RAID可以充分发 挥出多块硬盘的优势，可以提升硬盘速度，增大容量，提供容错功能够确保数据安全性，易于管理的优点，在任何一块硬盘出现问题的情况下都可以继续工作，不会 受到损坏硬盘的影响

二．Raid的级别

1.RAID 0

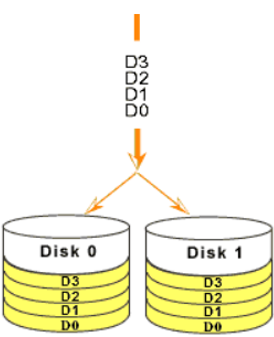
 RAID 0 又称为 Stripe 或 Striping，它代表了所有 RAID 级别中最高的存储性能。RAID 0 提高存储性能的原理是把连续的数据分散到多个磁盘上存取，这样，系统有数据请求就可以被多个磁盘并行的执行，每个磁盘执行属于它自己的那部分数据请求。这种数据上的并行操作可以充分利用总线的带宽，显著提高磁盘整体存取性能。  
如图 1 所示:系统向三个磁盘组成的逻辑硬盘(RADI 0 磁盘组)发出的 I/O 数据请求被转化为 3 项操作，其中的每一项操作都对应于一块物理硬盘。我们从图中可以清楚的看到通过建立 RAID 0，原先顺序的数据请求被分散到所有的三块硬盘中同时执行。



从理论上讲，三块硬盘的并行操作使同一时间内磁盘读写速度提升了 3 倍。 但由于总线带宽等多种因素的影响，实际的提升速率肯定会低于理论值，但是，大量数据并行传输与串行传输比较，提速效果显著显然毋庸置疑。  
RAID 0 的缺点是不提供数据冗余，因此一旦用户数据损坏，损坏的数据将无法得到恢复。  
RAID 0 具有的特点，使其特别适用于对性能要求较高。

2.RAID1

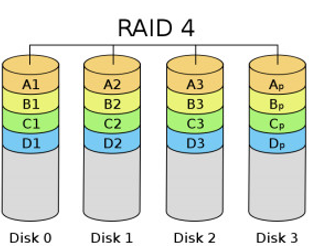
RAID 1 又称为 Mirror 或 Mirroring，它的宗旨是最大限度的保证用户数据的可用性和可修复性。 RAID1 的操作方式是把用户写入硬盘的数据百分之百地自动复制到另外一个硬盘上。当读取数据时，系统先从 RAID 0 的源盘读取数据，如果读取数据成功，则系统不去管备份盘上的数据;如果读取源盘数据失败，则系统自动转而读取备份盘上的数据，不会造成用户工作任务的中断。当然，我们应当及时地更换损坏的硬盘并利用备份数据重新建立 Mirror，避免备份盘在发生损坏时，造成不可挽。



3.Raid 4

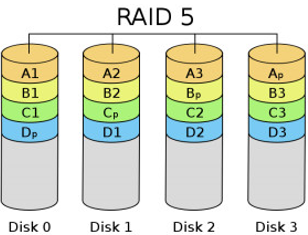
**Raid4带奇偶校验码的独立磁盘结构**

RAID 4是以扇区作数据分段,各磁盘相同位置的分段形成一个校验磁盘分段(parity block),放在校验磁盘。这种方式可在不同的磁盘平行执行不同的读取命今,大幅提高磁盘阵列的读取性能;但写入数据时,因受限于校验磁盘,同一时间只能作一次,启动所有磁盘读取数据形成同一校验分段的所有数据分段,与要写入的数据做好校验计算再写入。但校验磁盘形成RAID 4的瓶颈,降低了性能,因有RAID 5而使得RAID 4较少使用



4.raid 5

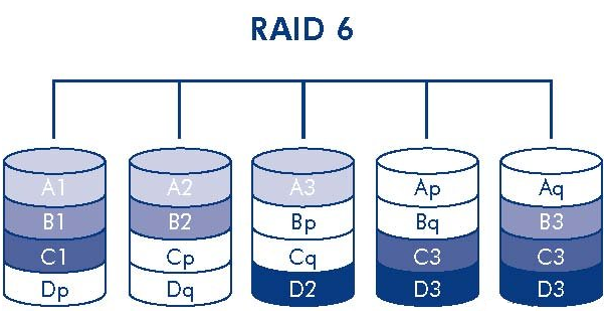
RAID 5 是一种存储性能、数据安全和存储成本兼顾的存储解决方案。 以四个硬盘组成的 RAID 5 为例，其数据存储方式如下图所示:图中，P0 为 D0，D1 和 D2 的奇偶校验信息，其它以此类推。



RAID 5 不对存储的数据进行备份，而是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。当 RAID5 的一个  
磁盘数据发生损坏后，利用剩下的数据和相应的奇偶校验信息去恢复被损坏的数据。  
RAID 5 可以理解为是 RAID 0 和 RAID 1 的折衷方案。RAID 5 可以为系统提供数据安全保障，但保障  
程度要比 Mirror 低而磁盘空间利用率要比 Mirror 高。RAID 5 具有和 RAID 0 相近似的数据读取速度，只  
是多了一个奇偶校验信息，写入数据的速度比对单个磁盘进行写入操作稍慢。同时由于多个数据对应一个

5.RAID6

RADI6技术是在RAID 5基础上，为了进一步加强数据保护而设计的一种RAID方式，实际上是一种扩展RAID 5等级。与RAID 5的不同之处于除了每个硬盘上都有同级数据XOR校验区外，还有一个针对每个数据块的XOR校验区。当然，当前盘数据块的校验数据不可能存在当前盘而是交错存储的，具体形式见图

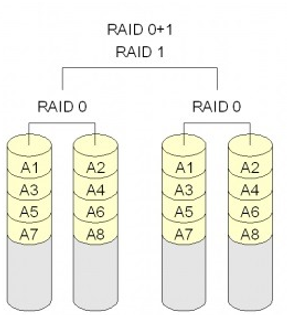


RAID-6和RAID-5一样对逻辑盘进行条带化然后存储数据和校验位，只是对每一位数据又增加了一位校验位。这样在使用RAID-6时会有两块硬盘用来存储校验位，增强了容错功能，同时必然会减少硬盘的实际使用容量。以前的raid级别一般只允许一块硬盘坏掉，而RAID-6可以允许坏掉两块硬盘，因此，RAID-6 要求至少4块硬盘。

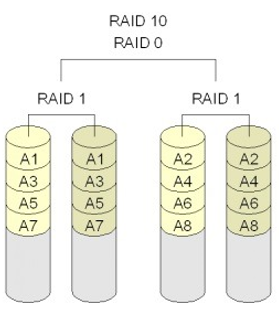
6.RAID 0+1与RAID10

RAID 0 和 RAID 1 的组合称为 RAID 0+1，或称为 RAID 10。如下所述，它具有一些有趣的优越性。通  
过将 RAID 0 分块的速度和 RAID 1 镜像的冗余进行组合，结果产生了一个快速、没有写开销、具有极好冗  
余性质的子系统。

Raid 0+1



Raid 10：先建立raid1，在建raid0



RAID１+０与RAID０+１的区别：

RAID 1+0是先镜射再分区数据，再将所有硬盘分为两组，视为是RAID 0的最低组合，然后将这两组各自视为RAID 1运作。RAID 0+1则是跟RAID 1+0的程序相反，是先分区再将数据镜射到两组硬盘。它将所有的硬盘分为两组，变成RAID 1的最低组合，而将两组硬盘各自视为RAID 0运作。性能上，RAID 0+1比RAID 1+0有着更快的读写速度。可靠性上，当RAID 1+0有一个硬盘受损，其余三个硬盘会继续运作。RAID 0+1 只要有一个硬盘受损，同组RAID 0的另一只硬盘亦会停止运作，只剩下两个硬盘运作，可靠性较低。因此，RAID 10远较RAID 01常用。

RAID总结：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| RAID Level | 性能提升 | 冗余能力 | 空间利用率 | 磁盘数量（块） | 允许磁盘故障数 | 优点 | 缺点 |
| RAID 0 | 读、写提升 | 无 | 100% | 至少2 | 不允许 | 读写性能，利用率高 | 缺乏冗余能力 |
| RAID 1 | 读性能提升，写性能下降 | 有 | 50% | 至少2 | 50% | 读性能提升，具有冗余 | 写性能下降，磁盘利用率低 |
| RAID 4 | 读、写提升 | 有 | n\*（n-1）/n | 至少3 | 1 | 读写性能提升 | 瓶颈在校验盘上 |
| RAID 5 | 读、写提升 | 有 | n\*（n-1）/n | 至少3 | 1 | 读写性能提升 | 仅允许坏一块盘 |
| RAID 6 | 读、写提升 | 有 | n\*（n-2）/n | 至少4块 | 2 | 读写性能提升 | 磁盘利用率降低，写入速度没有raid5块 |
| RAID 1+0 | 读、写提升 | 有 | 50% | 至少4 | 50% | 读写性能提升 | 磁盘利用率低 |
| RAID 0+1 | 读、写提升 | 有 | 50% | 至少4 | 50% | 读写性能提升 | 磁盘利用率低，容错性没有raid10好 |

六、mdadm 工具介绍

mdadm（multiple devices admin）是 linux下标准的的软raid管理工具，是一个模式化工具（在不同的模式下）；程序工作在内存用户程序区，为用户提供RAID接口来操作内核的模块，实现各种功能

**目前支持的模式**  
LINEAR(线性模式)、RAID0(striping条带模式)、RAID1(mirroring)、 RAID-4、RAID-5、 RAID-6、 RAID-10、 MULTIPATH和FAULTY

**LINEAR:**线性模式，该模式不是raid的标准模式，其主要作用是可以实现将几块小的硬盘组合为一块大硬盘来使用，数组存储时一次存满一个硬盘在使用下一个硬盘，对上层来说操作的是一个大硬盘，这里就不对该模式进行详解了。

命令的语法格式：mdadm [mode] <raiddevice> [options] <component-devices>

模式：

        创建：-C

        装配: -A

        监控: -F

        管理：-f, -r, -a

-C: 创建模式

                        -n #: 使用#个块设备来创建此RAID；

                        -l #：指明要创建的RAID的级别；

                        -a {yes|no}：自动创建目标RAID设备的设备文件；

                        -c CHUNK\_SIZE: 指明块大小；

                        -x #: 指明空闲盘的个数；

一．RAID创建

1.  创建分区/dev/sdbc5 /dev/sdbc6  /dev/sdb7 容量各5G，类型为fd

[root@localhost ~]# fdisk -l /dev/sdc

Disk /dev/sdc: 16.1 GB, 16106127360 bytes

255 heads, 63 sectors/track, 1958 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0xe57d5d2f

   Device Boot      Start         End      Blocks   Id  System

/dev/sdc1               1        1958    15727603+   5  Extended

/dev/sdc5               1         654     5253192   fd  Linux raid autodetect

/dev/sdc6             655        1308     5253223+  fd  Linux raid autodetect

/dev/sdc7            1309        1958     5221093+  fd  Linux raid autodetect

2.  创建Raid1阵列

[root@localhost ~]# mdadm -C /dev/md0 -a yes -c 128 -l 1 -n 2  /dev/sdc{5,6}

mdadm: Note: this array has metadata at the start and

    may not be suitable as a boot device.  If you plan to

    store '/boot' on this device please ensure that

    your boot-loader understands md/v1.x metadata, or use

    --metadata=0.90

Continue creating array?

Continue creating array? (y/n) y

mdadm: Defaulting to version 1.2 metadata

mdadm: array /dev/md0 started

3.查看raid阵列

[root@localhost ~]# mdadm -D /dev/md0

/dev/md0:

        Version : 1.2

  Creation Time : Sat Sep  3 22:06:01 2016

     Raid Level : raid1

     Array Size : 5216960 (4.98 GiB 5.34 GB)

  Used Dev Size : 5216960 (4.98 GiB 5.34 GB)

   Raid Devices : 2

  Total Devices : 2

    Persistence : Superblock is persistent

    Update Time : Sat Sep  3 23:43:28 2016

          State : clean

 Active Devices : 2

Working Devices : 2

 Failed Devices : 0

  Spare Devices : 0

           Name : localhost.localdomain:0  (local to host localhost.localdomain)

           UUID : c3e420e0:6c7fa523:050ae5f1:ea7febec

         Events : 18

    Number   Major   Minor   RaidDevice State

       0       8       37        0      active sync   /dev/sdc5

       1       8       38        1      active sync   /dev/sdc6

4.挂载并查看

[root@localhost /]# mkdir /tmp/raid

[root@localhost /]# mount /dev/md0 /tmp/raid/

[root@localhost /]# df -h

Filesystem                    Size  Used Avail Use% Mounted on

/dev/mapper/VolGroup-lv\_root   27G  3.6G   22G  15% /

tmpfs                         932M   72K  932M   1% /dev/shm

/dev/sda1                     485M   40M  421M   9% /boot

/dev/md0                      4.9G  138M  4.6G   3% /tmp/raid

5.  开机自动挂载

[root@localhost /]# vim /etc/fstab

# /etc/fstab

# Created by anaconda on Mon Aug 22 18:25:35 2016

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info

#

/dev/mapper/VolGroup-lv\_root /                       ext4    defaults        1 1

UUID=e36e7a7c-97a5-4712-ac29-7973c98c847c /boot                   ext4    defaults        1 2

/dev/mapper/VolGroup-lv\_swap swap                    swap    defaults        0 0

tmpfs                   /dev/shm                tmpfs   defaults        0 0

devpts                  /dev/pts                devpts  gid=5,mode=620  0 0

sysfs                   /sys                    sysfs   defaults        0 0

proc                    /proc                   proc    defaults        0 0

/tmp/raid               /de/md0                 ext4    defaults        0 0

第二．Raid信息的查看

-D：显示raid的详细信息；

         mdadm -D /dev/md#

[root@localhost ~]# mdadm -D /dev/md0

/dev/md0:

        Version : 1.2

  Creation Time : Sat Sep  3 22:06:01 2016

     Raid Level : raid1

     Array Size : 5216960 (4.98 GiB 5.34 GB)

  Used Dev Size : 5216960 (4.98 GiB 5.34 GB)

   Raid Devices : 2

  Total Devices : 2

    Persistence : Superblock is persistent

    Update Time : Sat Sep  3 23:43:28 2016

          State : clean

 Active Devices : 2

Working Devices : 2

 Failed Devices : 0

  Spare Devices : 0

           Name : localhost.localdomain:0  (local to host localhost.localdomain)

           UUID : c3e420e0:6c7fa523:050ae5f1:ea7febec

         Events : 18

    Number   Major   Minor   RaidDevice State

       0       8       37        0      active sync   /dev/sdc5

       1       8       38        1      active sync   /dev/sdc6

使用 watch -n 10 'cat /proc/mdstat'查看阵列同步情况

第三．Raid阵列的管理

管理模式：

                        -f: 标记指定磁盘为损坏；

                        -a: 添加磁盘

                        -r: 移除磁盘

                        -S：停止RAID

RAID管理

1.  给raid-5新增一个spare(空)盘，添加磁盘到阵列中做备用盘(spare)

[root@localhost /]# mdadm -a /dev/md0 /dev/sdc7

mdadm: added /dev/sdc7

2.  模拟故障

[root@localhost /]# mdadm  -f /dev/md0 /dev/sdc5

[root@localhost /]# mdadm /dev/md0

/dev/md0: 4.98GiB raid1 2 devices, 1 spare. Use mdadm --detail for more detail.

[root@localhost /]# mdadm -D /dev/md0

/dev/md0:

        Version : 1.2

  Creation Time : Sat Sep  3 22:06:01 2016

     Raid Level : raid1

     Array Size : 5216960 (4.98 GiB 5.34 GB)

  Used Dev Size : 5216960 (4.98 GiB 5.34 GB)

   Raid Devices : 2

  Total Devices : 3

    Persistence : Superblock is persistent

    Update Time : Sat Sep  3 23:54:39 2016

          State : clean, degraded, recovering

 Active Devices : 1

Working Devices : 2

 Failed Devices : 1

  Spare Devices : 1

 Rebuild Status : 68% complete

           Name : localhost.localdomain:0  (local to host localhost.localdomain)

           UUID : c3e420e0:6c7fa523:050ae5f1:ea7febec

         Events : 35

    Number   Major   Minor   RaidDevice State

       2       8       39        0      spare rebuilding   /dev/sdc7

       1       8       38        1      active sync   /dev/sdc6

       0       8       37        -      faulty   /dev/sdc5

3.  移除故障的硬盘

[root@localhost /]# mdadm -r /dev/md0 /dev/sdc5

mdadm: hot removed /dev/sdc5 from /dev/md0

[root@localhost /]# mdadm -D /dev/md0

/dev/md0:

        Version : 1.2

  Creation Time : Sat Sep  3 22:06:01 2016

     Raid Level : raid1

     Array Size : 5216960 (4.98 GiB 5.34 GB)

  Used Dev Size : 5216960 (4.98 GiB 5.34 GB)

   Raid Devices : 2

  Total Devices : 2

    Persistence : Superblock is persistent

    Update Time : Sat Sep  3 23:55:29 2016

          State : clean

 Active Devices : 2

Working Devices : 2

 Failed Devices : 0

  Spare Devices : 0

           Name : localhost.localdomain:0  (local to host localhost.localdomain)

           UUID : c3e420e0:6c7fa523:050ae5f1:ea7febec

         Events : 44

    Number   Major   Minor   RaidDevice State

       2       8       39        0      active sync   /dev/sdc7

       1       8       38        1      active sync   /dev/sdc6

4. 停止RAID

[root@localhost /]# umount /tmp/raid/

[root@localhost /]# mdadm -S /dev/md0

mdadm: stopped /dev/md0

[root@localhost /]# mdadm -D /dev/md0

mdadm: cannot open /dev/md0: No such file or directory

[root@localhost /]#

                                           二.LVM篇

**一、简介**

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。

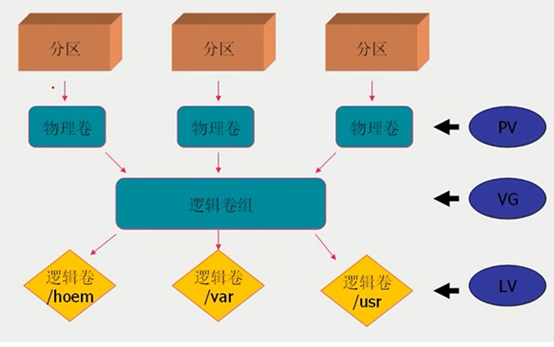
**基本的逻辑卷管理概念：**

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**。

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。

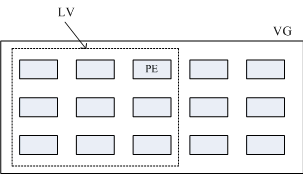
LV（Logical Volume）– 逻辑卷   
逻辑卷建立在卷组之上，卷组中的未分配空间可以用于建立新的逻辑卷，逻辑卷建立后可以动态地扩展和缩小空间。系统中的多个逻辑卷可以属于同一个卷组，也可以属于不同的多个卷组。

关系图如下：



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。



二.Linux lvm的工具

pv管理工具：

        pvs：简要pv信息显示

        pvdisplay：显示pv的详细信息

        pvcreate /dev/DEVICE: 创建pv

vg管理工具：

        vgs  简要vg信息显示

        vgdisplay  显示vg的详细信息

        vgcreate  [-s #[kKmMgGtTpPeE]] VolumeGroupName  PhysicalDevicePath [PhysicalDevicePath...]创建vg

         -s：指定pe大小

        vgextend  VolumeGroupName  PhysicalDevicePath [PhysicalDevicePath...]   扩展vg容量

        vgreduce  VolumeGroupName  PhysicalDevicePath [PhysicalDevicePath...]   减小vg容量

        vgremove  vg的移除

lv管理工具：

lvs  简要lv显示

        lvdisplay  详细lv显示

        lvcreate -L #[mMgGtT] -n NAME VolumeGroup    lv的创建

        lvremove /dev/VG\_NAME/LV\_NAME       lv的移除

扩展逻辑卷：

        # lvextend -L [+]#[mMgGtT] /dev/VG\_NAME/LV\_NAME

        # resize2fs /dev/VG\_NAME/LV\_NAME

缩减逻辑卷：

        # umount /dev/VG\_NAME/LV\_NAME

        # e2fsck -f /dev/VG\_NAME/LV\_NAME

        # resize2fs /dev/VG\_NAME/LV\_NAME #[mMgGtT]

        # lvreduce -L [-]#[mMgGtT] /dev/VG\_NAME/LV\_NAME

        # mount

快照：snapshot

lvcreate -L #[mMgGtT] -p r -s -n snapshot\_lv\_name original\_lv\_name

     -L：指定快照大小

     -p: 指定快照权限 r|rw

     -s:指定创建快照

     -n：快照名称

实验：

  一.lvm的创建

1、 创建分区

使用分区工具（如：fdisk等）创建LVM分区，方法和创建其他一般分区的方式是一样的，区别仅仅是LVM的分区类型为8e。

[root@localhost ~]# fdisk -l /dev/sdd

Disk /dev/sdd: 21.5 GB, 21474836480 bytes

255 heads, 63 sectors/track, 2610 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: 0xbbe9bf26

   Device Boot      Start         End      Blocks   Id  System

/dev/sdd1               1        2610    20964793+   5  Extended

/dev/sdd5               1         654     5253192   8e  Linux LVM

/dev/sdd6             655        1308     5253223+  8e  Linux LVM

/dev/sdd7            1309        1962     5253223+  8e  Linux LVM

2、 创建PV

[root@localhost ~]# pvcreate  /dev/sdd{5,6}

  Physical volume "/dev/sdd5" successfully created

  Physical volume "/dev/sdd6" successfully created

3.查看pv

[root@localhost ~]# pvdisplay

  "/dev/sdd5" is a new physical volume of "5.01 GiB"

  --- NEW Physical volume ---

  PV Name               /dev/sdd5

  VG Name

  PV Size               5.01 GiB

  Allocatable           NO

  PE Size               0

  Total PE              0

  Free PE               0

  Allocated PE          0

  PV UUID               32UtIW-NeDz-oTVa-hmHD-BJ4U-hD2o-5tN3yM

  "/dev/sdd6" is a new physical volume of "5.01 GiB"

  --- NEW Physical volume ---

  PV Name               /dev/sdd6

  VG Name

  PV Size               5.01 GiB

  Allocatable           NO

  PE Size               0

  Total PE              0

  Free PE               0

  Allocated PE          0

  PV UUID               GPNOTF-rv8H-DS1Z-kQT7-WAQC-bMX4-L5dowi

3.创建VG

[root@localhost ~]# vgcreate vg0 /dev/sdd{5,6}

  Volume group "vg0" successfully created

     4.查看VG

[root@localhost ~]# vgdisplay

  --- Volume group ---

  VG Name               vg0

  System ID

  Format                lvm2

  Metadata Areas        2

  Metadata Sequence No  1

  VG Access             read/write

  VG Status             resizable

  MAX LV                0

  Cur LV                0

  Open LV               0

  Max PV                0

  Cur PV                2

  Act PV                2

  VG Size               10.02 GiB

  PE Size               4.00 MiB

  Total PE              2564

  Alloc PE / Size       0 / 0

  Free  PE / Size       2564 / 10.02 GiB

  VG UUID               hf8cX5-Y2jb-FLh9-0ijm-cZmX-47wQ-KVU0pk

5.创建lv

创建了一个名字为lvm01，容量大小是6G的分区，其中：-L：指定LV的大小 -n：指定LV的名。vg0：表示从这个VG中划分LV；

[root@localhost ~]# lvcreate -L 6G -n lvm01 vg0

  Logical volume "lvm01" created

6.lv格式化及挂载

下一步需要对LV进行格式化（使用mksf进行格式化操作），然后LV才能存储资料

[root@localhost ~]# mke2fs -t ext4 /dev/vg0/lvm01

mke2fs 1.41.12 (17-May-2010)

Filesystem label=

OS type: Linux

Block size=4096 (log=2)

Fragment size=4096 (log=2)

Stride=0 blocks, Stripe width=0 blocks

393216 inodes, 1572864 blocks

78643 blocks (5.00%) reserved for the super user

First data block=0

Maximum filesystem blocks=1610612736

48 block groups

32768 blocks per group, 32768 fragments per group

8192 inodes per group

Superblock backups stored on blocks:

       32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done

Creating journal (32768 blocks): done

Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 25 mounts or

180 days, whichever comes first.  Use tune2fs -c or -i to override

7.挂载之后，可以看到此LV的容量

[root@localhost ~]# mkdir /tmp/lvm

[root@localhost ~]# mount /dev/vg0/lvm01 /tmp/lvm/

[root@localhost ~]# lvdisplay

  --- Logical volume ---

  LV Path                /dev/vg0/lvm01

  LV Name                lvm01

  VG Name                vg0

  LV UUID                y0q9jX-MiGc-eS6u-YVoL-n834-Hf2d-Bqcfz0

  LV Write Access        read/write

  LV Creation host, time localhost.localdomain, 2016-09-04 00:24:35 +0800

  LV Status              available

  # open                 1

  LV Size                6.00 GiB

  Current LE             1536

  Segments               2

  Allocation             inherit

  Read ahead sectors     auto

  - currently set to     256

  Block device           253:2

8.开机自动挂载

 [root@localhost ~]# vim /etc/fstab

#

# /etc/fstab

# Created by anaconda on Mon Aug 22 18:25:35 2016

#

# Accessible filesystems, by reference, are maintained under '/dev/disk'

# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info

#

/dev/mapper/VolGroup-lv\_root /                       ext4    defaults        1 1

UUID=e36e7a7c-97a5-4712-ac29-7973c98c847c /boot      ext4    defaults        1 2

/dev/mapper/VolGroup-lv\_swap swap                    swap    defaults        0 0

tmpfs                   /dev/shm                     tmpfs   defaults        0 0

devpts                  /dev/pts                     devpts  gid=5,mode=620  0 0

sysfs                   /sys                         sysfs   defaults        0 0

proc                    /proc                        proc    defaults        0 0

/tmp/lvm                /dev/vg0/lvm01               ext4    defaults        0 0

二、扩容VG，LV

vgextend vg0 /dev/sdd7    #扩展vg0

[root@localhost ~]# vgextend vg0 /dev/sdd7

  No physical volume label read from /dev/sdd7

  Physical volume /dev/sdd7 not found

  Physical volume "/dev/sdd7" successfully created

  Volume group "vg0" successfully extended

[root@localhost ~]# vgdisplay

  --- Volume group ---

  VG Name               vg0

  System ID

  Format                lvm2

  Metadata Areas        3

  Metadata Sequence No  3

  VG Access             read/write

  VG Status             resizable

  MAX LV                0

  Cur LV                1

  Open LV               1

  Max PV                0

  Cur PV                3

  Act PV                3

  VG Size               15.02 GiB

  PE Size               4.00 MiB

  Total PE              3846

  Alloc PE / Size       1536 / 6.00 GiB

  Free  PE / Size       2310 / 9.02 GiB

  VG UUID               hf8cX5-Y2jb-FLh9-0ijm-cZmX-47wQ-KVU0pk

2.lvextend –L 1G /dev/vg0/lvm01 #扩展lvm增加1G空间

[root@localhost ~]# lvextend  -L +1G /dev/vg0/lvm01

  Extending logical volume lvm01 to 7.00 GiB

  Logical volume lvm01 successfully resized

[root@localhost ~]# lvdisplay

  --- Logical volume ---

  LV Path                /dev/vg0/lvm01

  LV Name                lvm01

  VG Name                vg0

  LV UUID                y0q9jX-MiGc-eS6u-YVoL-n834-Hf2d-Bqcfz0

  LV Write Access        read/write

  LV Creation host, time localhost.localdomain, 2016-09-04 00:24:35 +0800

  LV Status              available

  # open                 1

  LV Size                7.00 GiB

  Current LE             1792

  Segments               2

  Allocation             inherit

  Read ahead sectors     auto

  - currently set to     256

  Block device           253:2

3.resize2fs /dev/vg0 /lvm01#执行该重设大小，对于当前正在使用的Lvm01有效

df –h #查看挂载情况，已经扩容

[root@localhost ~]# resize2fs /dev/vg0/lvm01

resize2fs 1.41.12 (17-May-2010)

Filesystem at /dev/vg0/lvm01 is mounted on /tmp/lvm; on-line resizing required

old desc\_blocks = 1, new\_desc\_blocks = 1

Performing an on-line resize of /dev/vg0/lvm01 to 1835008 (4k) blocks.

The filesystem on /dev/vg0/lvm01 is now 1835008 blocks long.

[root@localhost ~]# df -h

Filesystem                    Size  Used Avail Use% Mounted on

/dev/mapper/VolGroup-lv\_root   27G  3.6G   22G  15% /

tmpfs                         932M   72K  932M   1% /dev/shm

/dev/sda1                     485M   40M  421M   9% /boot

/dev/mapper/vg0-lvm01         6.9G  142M  6.5G   3% /tmp/lvm

三．减少VG,LV

缩减逻辑卷：

     1.# umount /dev/VG\_NAME/LV\_NAME  卸载设备

[root@localhost ~]# umount /dev/vg0/lvm01

    2. # e2fsck -f /dev/VG\_NAME/LV\_NAME 检查逻辑卷

[root@localhost ~]# e2fsck -f /dev/vg0/lvm01

e2fsck 1.41.12 (17-May-2010)

Pass 1: Checking inodes, blocks, and sizes

Pass 2: Checking directory structure

Pass 3: Checking directory connectivity

Pass 4: Checking reference counts

Pass 5: Checking group summary information

/dev/vg0/lvm01: 11/458752 files (0.0% non-contiguous), 65023/1835008 blocks

    3. # resize2fs /dev/VG\_NAME/LV\_NAME #[mMgGtT] 重设大小

[root@localhost ~]# resize2fs  /dev/vg0/lvm01 5G

resize2fs 1.41.12 (17-May-2010)

Resizing the filesystem on /dev/vg0/lvm01 to 1310720 (4k) blocks.

The filesystem on /dev/vg0/lvm01 is now 1310720 blocks long.

     4.# lvreduce -L [-]#[mMgGtT] /dev/VG\_NAME/LV\_NAME  减少逻辑卷容量

[root@localhost ~]# lvreduce -L 5G /dev/vg0/lvm01

  WARNING: Reducing active logical volume to 5.00 GiB

  THIS MAY DESTROY YOUR DATA (filesystem etc.)

Do you really want to reduce lvm01? [y/n]: y

  Reducing logical volume lvm01 to 5.00 GiB

  Logical volume lvm01 successfully resized

    5. # mount 重新挂载

[root@localhost ~]# mount /dev/vg0/lvm01  /tmp/lvm/

[root@localhost ~]# df -h

Filesystem                    Size  Used Avail Use% Mounted on

/dev/mapper/VolGroup-lv\_root   27G  3.6G   22G  15% /

tmpfs                         932M   72K  932M   1% /dev/shm

/dev/sda1                     485M   40M  421M   9% /boot

/dev/mapper/vg0-lvm01         5.0G  140M  4.6G   3% /tmp/lvm

四.快照

  快照技术是一种摄影技术,随着存储应用需求的提高,用户需要在线方式进行数据保护,快照就是在线存储设备防范数据丢失的有效方法之一

snapshot

lvcreate -L #[mMgGtT] -p r -s -n snapshot\_lv\_name original\_lv\_name

                  -L：指定快照大小

                  -p：指定逻辑卷的权限r:w

                  -s:指定为快照

                  -n:快照名称

1. 快照创建

[root@localhost ~]# lvcreate -L 1G -p r -s -n sna\_lvm01 /dev/vg0/lvm01

  Logical volume "sna\_lvm01" created

 2.查看快照

[root@localhost ~]# lvdisplay

  --- Logical volume ---

  LV Path                /dev/vg0/lvm01

  LV Name                lvm01

  VG Name                vg0

  LV UUID                y0q9jX-MiGc-eS6u-YVoL-n834-Hf2d-Bqcfz0

  LV Write Access        read/write

  LV Creation host, time localhost.localdomain, 2016-09-04 00:24:35 +0800

  LV snapshot status     source of

                         sna\_lvm01 [active]

  LV Status              available

  # open                 1

  LV Size                5.00 GiB

  Current LE             1280

  Segments               1

  Allocation             inherit

  Read ahead sectors     auto

  - currently set to     256

  Block device           253:2

转载请注明：[linux运维部落](http://www.178linux.com/) » [RAID+LVM详解](http://www.178linux.com/43388)