逻辑卷管理

对于普通的分区，扩展度不高，一旦分区格式化完成，很难灵活的再增加或者减少分区大小。为了解决这个问题，可以使用LVM（逻辑卷）。基本过程是把物理磁盘或者分区初始化称为物理卷（PV），然后把PV加入VG（卷组），最后在VG上划分逻辑的分区（LVM），LVM可以当做普通的分区进行格式化和挂载。

LVM:可以动态调整分区大小

PV:(physical volume)物理卷
VG:(volume Group)卷组
LV:(logical volume)逻辑卷

步骤：1、创建分区 分区的ID要变成LVM的ID:8e
2、将分区创建成PV pvcreate /dev/sdg2 /dev/sdh1
3、将PV加入卷组VG vgcreate huateng /dev/sdg2 /dev/sdh1
4、在VG上创建LV lvcreate -l 25 -n jishu huateng
5、格式化LV，并挂载使用

pvcreate 设备名1[设备名2] （dev/sda{1,2,3}）
vgcreate 卷组名(自定义) 物理卷名1 （dev/sda{1,2,3}）
vgcreate VG_NAME /PATH/TO/PV
-s #: PE大小，默认为4MB

扩展逻辑卷
lvcreate -n LV_NAME -L #G VG_NAME
名字 大小 卷名
mkfs -t ext4 lv名
lvextend -L +大小 /dev/卷组名/逻辑卷名

resize2fs /dev/myvg/mylv 扩展文件系统
如果是xfs文件系统，需要如下命令：
xfs_growfs /dev/myvg/mylv 扩展文件系统

LVM具体案例：

将准备的磁盘或分区创建PV

```
[root@server1 ~]# pvcreate /dev/sdb[123]
  Physical volume "/dev/sdb1" successfully created
  Physical volume "/dev/sdb2" successfully created
  Physical volume "/dev/sdb3" successfully created
[root@server1 ~]# pvscan
  PV /dev/sda2   VG rhel            lvm2 [39.51 GiB / 0    free]
  PV /dev/sdb1                      lvm2 [10.00 GiB]
  PV /dev/sdb2                      lvm2 [15.00 GiB]
  PV /dev/sdb3                      lvm2 [20.00 GiB]
  Total: 4 [84.51 GiB] / in use: 1 [39.51 GiB] / in no VG: 3 [45.00 GiB]
```

可以执行pvdisplay查看PV的详细信息，pvremove删除PV

创建完PV，之后，需要创建VG，然后添加PV到VG中

可以通过vgdisplay查看具体的信息，注意PE的Size是4M，这个是增减的最小计算单位

```
[root@server1 ~]# vgcreate vg00 /dev/sdb[12]
  Volume group "vg00" successfully created
[root@server1 ~]# vgscan
  Reading all physical volumes.  This may take a while...
  Found volume group "rhel" using metadata type lvm2
  Found volume group "vg00" using metadata type lvm2
[root@server1 ~]# vgdisplay vg00
  --- Volume group ---
  VG Name               vg00
  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               24.99 GiB
  PE Size               4.00 MiB
  Total PE              6398
  Alloc PE / Size       0 / 0
  Free  PE / Size       6398 / 24.99 GiB
  VG UUID               3XmR3g-fCj2-u9sI-n2uJ-hQ1S-qDK4-q1zS0s
```

注：创建VG时:使用–s选项的作用是在创建时指定PE块（物理扩展单元）的大小，默认是4M。

如：# vgcreate volGroup03 -s 8M /dev/sdb[12]

我们可以继续往vg里面添加新的分区

```
[root@server1 ~]# vgextend vg00 /dev/sdb3
  Volume group "vg00" successfully extended
[root@server1 ~]# vgdisplay vg00
  --- Volume group ---
  VG Name               vg00
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               44.99 GiB
  PE Size               4.00 MiB
  Total PE              11517
  Alloc PE / Size       0 / 0
  Free  PE / Size       11517 / 44.99 GiB
  VG UUID               3XmR3g-fCj2-u9sI-n2uJ-hQ1S-qDK4-q1zS0s
```

若事先没有把sdb3转化为pv，而是直接添加到vg里面，不过一旦添加了他自动就初始化成pv了。

可以添加当然也可以减少pv。 #vgreduce vg00 /dev/sdb3

VG准备就绪，可以创建了LVM了

```
[root@server1 ~]# lvcreate -L 110M -n lv00 vg00
  Rounding up size to full physical extent 112.00 MiB
  Logical volume "lv00" created
[root@server1 ~]# lvscan
  ACTIVE            '/dev/rhel/swap' [4.00 GiB] inherit
  ACTIVE            '/dev/rhel/root' [35.51 GiB] inherit
  ACTIVE            '/dev/vg00/lv00' [112.00 MiB] inherit
```

注意看他的大小其实是112M，因为PE的大小是4M，这个4M是最小单位，不能破开，因此28个PE就是112M

```
[root@server1 ~]# echo 112/4 | bc
28
```

```
[root@server1 ~]# lvdisplay /dev/vg00/lv00
  --- Logical volume ---
  LV Path                /dev/vg00/lv00
  LV Name                lv00
  VG Name                vg00
  LV UUID                OCqr4P-Wval-VmHO-1VAT-NKe1-w1xS-UhALMO
  LV Write Access        read/write
  LV Creation host, time server1.benet.com, 2015-05-11 13:42:10 +0800
  LV Status              available
  # open                 0
  LV Size                112.00 MiB
  Current LE             28
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:2
```

注：大L可以直接指定大小，小l是指定多少个PE的值

也可以设置剩余空间的百分比

```
[root@server1 ~]# lvcreate -l 10%free -n lv01 vg00
  Logical volume "lv01" created
```

删除逻辑卷　#lvremove /dev/vg00/lv01

对已经创建的逻辑卷，可以当做普通分区一样格式化和挂载

```
[root@server1 ~]# mkfs.xfs /dev/vg00/lv00
meta-data=/dev/vg00/lv00         isize=256    agcount=4, agsize=7168 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=0
data     =                       bsize=4096   blocks=28672, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=0
log      =internal log           bsize=4096   blocks=853, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
[root@server1 ~]# mount /dev/vg00/lv00 /data
```

修改/etc/fstab文件实现开机自动挂载。

扩展一个逻辑卷，增加300M，首先要确保卷组有大于300M的空闲空间。

```
[root@server1 ~]# vgdisplay vg00
  --- Volume group ---
  VG Name               vg00
  System ID
  Format                lvm2
  Metadata Areas        3
  Metadata Sequence No  4
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                2
  Open LV               1
  Max PV                0
  Cur PV                3
  Act PV                3
  VG Size               44.99 GiB
  PE Size               4.00 MiB
  Total PE              11517
  Alloc PE / Size       1176 / 4.59 GiB
  Free  PE / Size       10341 / 40.39 GiB
  VG UUID               3XmR3g-fCj2-u9sI-n2uJ-hQ1S-qDK4-q1zS0s
```

执行lvextend扩展逻辑卷大小

```
[root@server1 ~]# lvextend -L +200M /dev/vg00/lv00
  Extending logical volume lv00 to 312.00 MiB
  Logical volume lv00 successfully resized
[root@server1 ~]# lvscan
  ACTIVE            '/dev/rhel/swap' [4.00 GiB] inherit
  ACTIVE            '/dev/rhel/root' [35.51 GiB] inherit
  ACTIVE            '/dev/vg00/lv00' [312.00 MiB] inherit
  ACTIVE            '/dev/vg00/lv01' [4.48 GiB] inherit
[root@server1 ~]# df -hT
Filesystem              Type      Size  Used Avail Use% Mounted on
/dev/mapper/rhel-root   xfs        36G  5.2G   31G  15% /
devtmpfs                devtmpfs  485M     0  485M   0% /dev
tmpfs                   tmpfs     494M     0  494M   0% /dev/shm
tmpfs                   tmpfs     494M  7.0M  487M   2% /run
tmpfs                   tmpfs     494M     0  494M   0% /sys/fs/cgroup
/dev/sda1               xfs       497M  119M  379M  24% /boot
/dev/mapper/vg00-lv00   xfs       109M  5.8M  103M   6% /data
```

注意逻辑卷的文件系统仍然是109M没有改变，我们还需要填充文件系统的空白。

RHEL7可以用xfs_growfs来扩大XFS文件系统，也可以直接用resize2fs 来处理设备

注意的是 XFS系统只能增长，不能减少！因此如果需要减少LVM的话，分区只能使用ext4了

```
[root@server1 ~]# xfs_growfs /dev/vg00/lv00
meta-data=/dev/mapper/vg00-lv00  isize=256    agcount=4, agsize=7168 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=0
data     =                       bsize=4096   blocks=28672, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=0
log      =internal               bsize=4096   blocks=853, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
data blocks changed from 28672 to 79872
```

执行df查看扩展后的文件系统

```
[root@server1 ~]# df -hT
Filesystem            Type      Size  Used Avail Use% Mounted on
/dev/mapper/rhel-root xfs        36G  5.2G   31G  15% /
devtmpfs              devtmpfs  485M     0  485M   0% /dev
tmpfs                 tmpfs     494M     0  494M   0% /dev/shm
tmpfs                 tmpfs     494M  7.0M  487M   2% /run
tmpfs                 tmpfs     494M     0  494M   0% /sys/fs/cgroup
/dev/sda1             xfs       497M  119M  379M  24% /boot
/dev/mapper/vg00-lv00 xfs       309M  6.1M  303M   2% /data
```

课后习题：

1、添加一块大小为20G的硬盘，然后划分为6个分区

2、将 6 个分区创建为 PV，并加入到名为 vg0 的卷组中。

3、创建大小为 2G 的逻辑卷 lv0，并格式化为 ext4 文件系统，永久挂载到/lv0 目录中。

4、将 lv0 在线扩展到 4G。

5、创建大小为 3G 的 lv1，并格式化为 xfs 文件系统，永久挂载到/lv1 目录中。

6、将 lv1 在线扩展到 5G。