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

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
练习：创建一个由两个物理卷组成的大小为20G的卷组myvg，要求其PE大小为16M；而后在此卷组中创建一个大小为5G的逻辑卷lv1，此逻辑卷要能在开机后自动挂载至/users目录，且支持ACL功能；缩减前面创建的逻辑卷lv1的大小至2G；
pvmove /dev/sda[n]
vgreduce myvg /dev/sda[n]>>>>vgs pvs pvremove /dev/sda[n]
一、扩展逻辑卷；

vgextend 名字 新加的物理卷

lvextend

-L [+]# /PATH/TO/LV

resize2fs

resize2fs -p /PATH/TO/LV

二、缩减逻辑卷：

步骤：

1、卸载已经挂载的逻辑卷分区

2、利用resize2fs指令修改文件系统大小以实现空间缩减

e2fsck -f /dev/huateng/jishu

resize2fs /dev/huateng/jishu 308M

3、使用lvreduce减少逻辑卷空间

lvreduce -L -80M /dev/huateng/jishu

4、挂载缩减后的逻辑卷分区

注意：1、不能在线缩减，得先卸载；

2、确保缩减后的空间大小依然能存储原有的所有数据；

3、在缩减之前应该先强行检查文件，以确保文件系统处于一至性状态；

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
```

逻辑卷快照

LVM提供一个极妙的设备，它是snapshot。允许管理员创建一个新的块装置，在某个时间点提供了一个精确的逻辑卷副本，快照提供原始卷的静态视图LVM 快照通过把文件系统的改变记录到一个快照分区，因此当你创建一个快照分区时，你不需要使用和你正创建快照的分区一样大小的分区,所需的空间大小取决于快照的使用，所以没有可循的方法来设置此大小。如果快照的大小等于原始卷的大小那么快照永远可用。

快照是特殊的逻辑卷，只可以对逻辑卷做快照。逻辑卷快照和需要做快照的逻辑卷必须在同一个卷组里面

现在在我们的系统中有个逻辑卷/dev/vg00/lv00，我们用lvdisplay来查询一下这个逻辑卷

```
[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                 1
  LV Size                312.00 MiB
  Current LE             78
  Segments               2
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:2
```

```
[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
```

可以看到，这个逻辑卷/dev/vg00/lv00的大小是309M。我们将这个逻辑卷/dev/vg00/lv00挂载到/data下面。复制一些数据到/data里面去。方便等下做试验

```
[root@server1 ~]# cp /etc/passwd /etc/shadow /etc/group /data/
[root@server1 ~]# ls -l /data
total 12
-rw-r--r-- 1 root root  826 May 12 07:34 group
-rw-r--r-- 1 root root 1908 May 12 07:34 passwd
---------- 1 root root 1127 May 12 07:34 shadow
```

现在我们就为逻辑卷/dev/vg00/lv00来做快照

```
[root@server1 ~]# lvcreate --size 300M --snapshot --name lvsp00 /dev/vg00/lv00
  Logical volume "lvsp00" created
```

执行lvscan查看创建好的逻辑卷

```
[root@server1 ~]# lvscan
  ACTIVE            '/dev/rhel/swap' [4.00 GiB] inherit
  ACTIVE            '/dev/rhel/root' [35.51 GiB] inherit
  ACTIVE   Original '/dev/vg00/lv00' [300.00 MiB] inherit
  ACTIVE   Snapshot '/dev/vg00/lvsp00' [300.00 MiB] inherit
```

可以看到/dev/vg00/lv00是原始逻辑卷，而/dev/vg00/lvsp00是快照

执行lvdisplay或lvs命令查看逻辑信息

```
[root@server1 ~]# lvdisplay /dev/vg00/lvsp00
  --- Logical volume ---
  LV Path                /dev/vg00/lvsp00
  LV Name                lvsp00
  VG Name                vg00
  LV UUID                CbiN6G-yQ3V-TrDC-AYOG-PGjb-9IOg-5x0r5k
  LV Write Access        read/write
  LV Creation host, time server1.benet.com, 2015-05-12 07:39:06 +0800
  LV snapshot status     active destination for lv00
  LV Status              available
  # open                 0
  LV Size                300.00 MiB
  Current LE             75
  COW-table size         300.00 MiB
  COW-table LE           75
  Allocated to snapshot  0.01%
  Snapshot chunk size    4.00 KiB
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:2

[root@server1 ~]# lvs
  LV     VG   Attr       LSize   Pool Origin Data%  Move Log Cpy%Sync Convert
  root   rhel -wi-ao----  35.51g
  swap   rhel -wi-ao----   4.00g
  lv00   vg00 owi-aos--- 300.00m
  lvsp00 vg00 swi-a-s--- 300.00m      lv00     0.01
```

可以看到逻辑卷快照创建成功了，

注意：这个快照卷建好之后，是不需要格式化也不需要进行挂载的。格式化或挂载都会出现的错误提示的。

模拟将原逻辑卷中的数据删除

```
[root@server1 ~]# rm -fr /data/*
[root@server1 ~]# ls /data/
```

如何恢复原逻辑卷的数据？有两方式可以恢复删除的数据

方式一是先将原逻辑卷卸除挂载#umount /dev/vg00/lv00

然后挂载逻辑卷快照即可 #mount /dev/vg00/lvsp00 /data，就可以正常访问数据了

```
[root@server1 ~]# umount /dev/vg00/lv00
[root@server1 ~]# mount /dev/vg00/lvsp00 /data
[root@server1 ~]# ls /data
group  passwd  shadow
[root@server1 ~]#
```

方式二可以通过 lvconvert把快照的内容重新写回原有的lvm

先将原逻辑卷卸除挂载#umount /dev/vg00/lv00

执行lvconvert将快照的数据合并到原逻辑卷 #lvconvert --merge /dev/vg00/lvsp00

最后挂载原逻辑卷，查看数据是否恢复成功

```
[root@server1 ~]# umount /dev/vg00/lv00
[root@server1 ~]# lvconvert --merge /dev/vg00/lvsp00
  Merging of volume lvsp00 started.
  lv00: Merged: 100.0%
  Merge of snapshot into logical volume lv00 has finished.
  Logical volume "lvsp00" successfully removed
[root@server1 ~]# mount /dev/vg00/lv00 /data
[root@server1 ~]# ls /data
group  passwd  shadow
```

注：当我们把原逻辑卷里面的数据给删除了，逻辑卷快照里面的数据还在，所以可以用快照恢复数据。而当我们在逻辑卷里面添加数据，快照是不会发生改变的，是没有这个文件的。因为快照只会备份当时逻辑卷的一瞬间。

使用ssm（系统存储管理器）进行逻辑管理

逻辑卷管理器（LVM）是一种极其灵活的磁盘管理工具，它让用户可以从多个物理硬驱创建逻辑磁盘卷，并调整大小，根本没有停机时间。最新版本的CentOS/RHEL 7现在随带系统存储管理器（又叫ssm），这是一种统一的命令行界面，由红帽公司开发，用于管理各种各样的存储设备。目前，有三种可供ssm使用的卷管理后端：LVM、Btrfs和Crypt

准备ssm，在CentOS/RHEL 7上，你需要首先安装系统存储管理器。可以通过rpm或yum工具安装

```
[root@server1 Packages]# rpm -vih system-storage-manager-0.4-5.el7.noarch.rpm
warning: system-storage-manager-0.4-5.el7.noarch.rpm: Header V3 RSA/SHA256 Signature, key ID fd431d5
1: MOKEY
Preparing...                          ################################# [100%]
Updating / installing...
   1:system-storage-manager-0.4-5.el7 ################################# [100%]
```

首先我们来检查关于可用硬盘和LVM卷的信息。下面这个命令将显示关于现有磁盘存储设备、存储池、LVM卷和存储快照的信息。

#ssm list

```
[root@server1 ~]# ssm list
[ 2416.572346] end_request: I/O error, dev fd0, sector 0
[ 2416.932282] end_request: I/O error, dev fd0, sector 0
----------------------------------------------------------
Device         Free       Used      Total  Pool  Mount point
----------------------------------------------------------
/dev/fd0                             4.00 KB
/dev/sda                            40.00 GB        PARTITIONED
/dev/sda1                          500.00 MB        /boot
/dev/sda2   0.00 KB   39.51 GB   39.51 GB  rhel
/dev/sdb                           100.00 GB
/dev/sdb1   9.70 GB  300.00 MB   10.00 GB  vg00
/dev/sdb2  15.00 GB    0.00 KB   15.00 GB  vg00
/dev/sdb3  20.00 GB    0.00 KB   20.00 GB  vg00
----------------------------------------------------------
[ 2417.292468] end_request: I/O error, dev fd0, sector 0
-------------------------------------------------
Pool  Type  Devices      Free       Used     Total
-------------------------------------------------
rhel  lvm   1         0.00 KB   39.51 GB  39.51 GB
vg00  lvm   3        44.70 GB  300.00 MB  44.99 GB
-------------------------------------------------
[ 2417.490048] end_request: I/O error, dev fd0, sector 0
[ 2417.943019] end_request: I/O error, dev fd0, sector 0
-------------------------------------------------------------------------------
Volume          Pool  Volume size  FS     FS size       Free  Type    Mount point
-------------------------------------------------------------------------------
/dev/rhel/root  rhel     35.51 GB  xfs   35.49 GB   30.35 GB  linear  /
/dev/rhel/swap  rhel      4.00 GB                             linear
/dev/vg00/lv00  vg00    300.00 MB  xfs  296.67 MB  296.53 MB  linear  /data
/dev/sda1               500.00 MB  xfs  496.67 MB  403.25 MB  part    /boot
-------------------------------------------------------------------------------
[ 2418.558159] end_request: I/O error, dev fd0, sector 0
```

在这个例子中，有两个物理设备（“/dev/sda”和“/dev/sdb”）、二个存储池（“rhel和vg00”）,以及存储池rhel中创建的两个LVM卷（“dev/rhel/root”和“/dev/rhel/swap”），存储池vg00中创建的一个LVM卷（/dev/vg00/lv00）。

下面来讲解如何通过ssm创建、管理逻辑卷和逻辑卷快照

至少新添加一块磁盘，执行ssm命令显示现有磁盘存储设备、存储池、LVM卷的信息

```
[root@server1 ~]# ssm list
[  220.277954] end_request: I/O error, dev fd0, sector 0
[  220.648105] end_request: I/O error, dev fd0, sector 0
----------------------------------------------------------
Device         Free       Used      Total  Pool  Mount point
----------------------------------------------------------
/dev/fd0                             4.00 KB
/dev/sda                            40.00 GB        PARTITIONED
/dev/sda1                          500.00 MB        /boot
/dev/sda2   0.00 KB   39.51 GB   39.51 GB  rhel
/dev/sdb                           100.00 GB
/dev/sdb1   9.70 GB  300.00 MB   10.00 GB  vg00
/dev/sdb2  15.00 GB    0.00 KB   15.00 GB  vg00
/dev/sdb3  20.00 GB    0.00 KB   20.00 GB  vg00
/dev/sdc                            20.00 GB
/dev/sdd                            20.00 GB
```

可以看到有两块空闲磁盘（sdc、sdd）

创建新的LVM池/卷

在这个示例中，不妨看一下如何在物理磁盘驱动器上创建新的存储池和新的LVM卷。如果使用传统的LVM工具，整个过程相当复杂，需要准备分区，需要创建物理卷、卷组、逻辑卷，最后还要建立文件系统。不过，若使用ssm，整个过程一蹴而就！

下面这个命令的作用是，创建一个名为mypool的存储池，创建存储池中名为lv01的500MB大小的LVM卷，使用XFS文件系统格式化卷，并将它挂载到/mnt/test下。

```
[root@server1 ~]# mkdir /mnt/test
[root@server1 ~]# ssm create -s 500M -n lv01 --fstype xfs -p mypool /dev/sdc /mnt/test
[  725.329942] end_request: I/O error, dev fd0, sector 0
[  725.590303] end_request: I/O error, dev fd0, sector 0
[  725.720327] end_request: I/O error, dev fd0, sector 0
  Physical volume "/dev/sdc" successfully created
  Volume group "mypool" successfully created
  Logical volume "lv01" created
meta-data=/dev/mypool/lv01       isize=256    agcount=4, agsize=32000 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=0
data     =                       bsize=4096   blocks=128000, 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
[  726.848317] end_request: I/O error, dev fd0, sector 0
```

验证ssm创建的结果

```
[root@server1 ~]# lvscan
  ACTIVE            '/dev/rhel/swap' [4.00 GiB] inherit
  ACTIVE            '/dev/rhel/root' [35.51 GiB] inherit
  ACTIVE            '/dev/mypool/lv01' [500.00 MiB] inherit
  ACTIVE            '/dev/vg00/lv00' [300.00 MiB] 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  7.1M  487M   2% /run
tmpfs                   tmpfs     494M     0  494M   0% /sys/fs/cgroup
/dev/sda1               xfs       497M  119M  379M  24% /boot
/dev/mapper/mypool-lv01 xfs       497M   26M  472M   6% /mnt/test
```

或执行ssm list

```
[  858.182269] end_request: I/O error, dev fd0, sector 0
[  858.374596] end_request: I/O error, dev fd0, sector 0
-------------------------------------------------------------
Device         Free       Used      Total  Pool    Mount point
-------------------------------------------------------------
/dev/fd0                             4.00 KB
/dev/sda                            40.00 GB          PARTITIONED
/dev/sda1                          500.00 MB          /boot
/dev/sda2   0.00 KB   39.51 GB   39.51 GB  rhel
/dev/sdb                           100.00 GB
/dev/sdb1   9.70 GB  300.00 MB   10.00 GB  vg00
/dev/sdb2  15.00 GB    0.00 KB   15.00 GB  vg00
/dev/sdb3  20.00 GB    0.00 KB   20.00 GB  vg00
/dev/sdc   19.51 GB  500.00 MB   20.00 GB  mypool
/dev/sdd                            20.00 GB
-------------------------------------------------------------
[  858.842410] end_request: I/O error, dev fd0, sector 0
-------------------------------------------------
Pool    Type  Devices      Free       Used     Total
-------------------------------------------------
mypool  lvm   1        19.51 GB  500.00 MB  20.00 GB
rhel    lvm   1         0.00 KB   39.51 GB  39.51 GB
vg00    lvm   3        44.70 GB  300.00 MB  44.99 GB
-------------------------------------------------
[  859.075024] end_request: I/O error, dev fd0, sector 0
[  859.810462] end_request: I/O error, dev fd0, sector 0
-------------------------------------------------------------------------------------------
Volume           Pool    Volume size  FS     FS size       Free  Type    Mount point
-------------------------------------------------------------------------------------------
/dev/rhel/root   rhel       35.51 GB  xfs   35.49 GB   30.35 GB  linear  /
/dev/rhel/swap   rhel        4.00 GB                             linear
/dev/vg00/lv00   vg00      300.00 MB  xfs  296.67 MB  296.53 MB  linear
/dev/mypool/lv01 mypool    500.00 MB  xfs  496.67 MB  496.54 MB  linear  /mnt/test
/dev/sda1                  500.00 MB  xfs  496.67 MB  403.25 MB  part    /boot
-------------------------------------------------------------------------------------------
[  860.323293] end_request: I/O error, dev fd0, sector 0
```

将物理磁盘(sdd)添加到LVM池

```
[root@server1 ~]# ssm add -p mypool /dev/sdd
[ 1031.319418] end_request: I/O error, dev fd0, sector 0
[ 1031.534547] end_request: I/O error, dev fd0, sector 0
[ 1031.649633] end_request: I/O error, dev fd0, sector 0
  Physical volume "/dev/sdd" successfully created
  Volume group "mypool" successfully extended
[root@server1 ~]#
```

新设备添加到存储池后，存储池会自动扩大，扩大多少取决于设备的大小。检查名为mypool的存储池的大小执行ssm list查看

```
---------------------------------------------------------
Device         Free       Used      Total  Pool    Mount point
---------------------------------------------------------
/dev/fd0                             4.00 KB
/dev/sda                            40.00 GB          PARTITIONED
/dev/sda1                          500.00 MB          /boot
/dev/sda2    0.00 KB   39.51 GB   39.51 GB  rhel
/dev/sdb                           100.00 GB
/dev/sdb1    9.70 GB  300.00 MB   10.00 GB  vg00
/dev/sdb2   15.00 GB    0.00 KB   15.00 GB  vg00
/dev/sdb3   20.00 GB    0.00 KB   20.00 GB  vg00
/dev/sdc    19.51 GB  500.00 MB   20.00 GB  mypool
/dev/sdd    20.00 GB    0.00 KB   20.00 GB  mypool
---------------------------------------------------------
[ 1061.921864] end_request: I/O error, dev fd0, sector 0
-----------------------------------------------
Pool    Type  Devices      Free       Used     Total
-----------------------------------------------
mypool  lvm   2        39.50 GB  500.00 MB  39.99 GB
rhel    lvm   1         0.00 KB   39.51 GB  39.51 GB
vg00    lvm   3        44.70 GB  300.00 MB  44.99 GB
```

接下来，我们来扩大现有的LVM卷

扩大LVM卷，将/dev/mypool/lv01卷的大小增加300MB。

如果你在存储池中有额外空间，可以扩大存储池中现有的磁盘卷。为此，使用ssm命令的resize选项

```
[root@server1 ~]# ssm resize -s+300M /dev/mypool/lv01
[ 1483.256570] end_request: I/O error, dev fd0, sector 0
[ 1483.568748] end_request: I/O error, dev fd0, sector 0
  Extending logical volume lv01 to 800.00 MiB
  Logical volume lv01 successfully resized
meta-data=/dev/mapper/mypool-lv01 isize=256    agcount=4, agsize=32000 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=0
data     =                       bsize=4096   blocks=128000, 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 128000 to 204800
[root@server1 ~]#
```

执行ssm list查看扩大后逻辑卷

```
-------------------------------------------------------------------------------
Volume           Pool    Volume size  FS     FS size       Free  Type    Mount point
-------------------------------------------------------------------------------
/dev/rhel/root   rhel       35.51 GB  xfs   35.49 GB   30.35 GB  linear  /
/dev/rhel/swap   rhel        4.00 GB                             linear
/dev/vg00/lv00   vg00      300.00 MB  xfs  296.67 MB  296.53 MB  linear
/dev/mypool/lv01 mypool    800.00 MB  xfs  496.67 MB  496.54 MB  linear  /mnt/test
/dev/sda1                  500.00 MB  xfs  496.67 MB  403.25 MB  part    /boot
-------------------------------------------------------------------------------
[ 1603.241232] end_request: I/O error, dev fd0, sector 0
```

可以看到逻辑卷扩大到800M，即在原来的基础上增加了300M，但文件系统大小（Fs size）还没有改变，仍然是原来的大小。

为了让文件系统识别增加后的卷大小，你需要“扩大”现有的文件系统本身。有不同的工具可用来扩大现有的文件系统，这取决于你使用哪种文件系统。比如说，有面向EXT2/EXT3/EXT4的resize2fs、面向XFS的xfs_growfs以及面向Btrfs的btrfs。

在CentOS 7中XFS文件系统在默认情况下创建。因而，我们使用xfs_growfs来扩大现有的XFS文件系统。

```
[root@server1 ~]# xfs_growfs /dev/mypool/lv01
meta-data=/dev/mapper/mypool-lv01 isize=256    agcount=7, agsize=32000 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=0
data     =                       bsize=4096   blocks=204800, 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
```

扩大XFS文件系统后，查看结果

```
Volume            Pool    Volume size  FS     FS size       Free  Type    Mount point
-----------------------------------------------------------------------------------
/dev/rhel/root    rhel       35.51 GB  xfs   35.49 GB   30.35 GB  linear  /
/dev/rhel/swap    rhel        4.00 GB                             linear
/dev/vg00/lv00    vg00      300.00 MB  xfs  296.67 MB  296.53 MB  linear
/dev/mypool/lv01  mypool    800.00 MB  xfs  796.67 MB  496.53 MB  linear  /mnt/test
/dev/sda1                   500.00 MB  xfs  496.67 MB  403.25 MB  part    /boot
-----------------------------------------------------------------------------------
```

或执行#df -hT

```
[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  7.1M  487M   2% /run
tmpfs                   tmpfs     494M     0  494M   0% /sys/fs/cgroup
/dev/sda1               xfs       497M  119M  379M  24% /boot
/dev/mapper/mypool-lv01 xfs       797M   26M  772M   4% /mnt/test
```

可以看到LVM扩展成功

逻辑卷快照

对现有的LVM卷（比如/dev/mypool/lv01）生成快照

一旦快照生成完毕，它将作为一个特殊的快照卷存储起来，存储了原始卷中生成快照时的所有数据

```
[root@server1 ~]# cp /etc/{passwd,shadow,group} /mnt/test/
[root@server1 ~]# ls /mnt/test/
group  passwd  shadow
```

```
[root@server1 ~]# ssm snapshot /dev/mypool/lv01
[ 2296.127906] end_request: I/O error, dev fd0, sector 0
[ 2296.437408] end_request: I/O error, dev fd0, sector 0
  Logical volume "snap20150512T101210" created
```

```
[root@server1 ~]# ssm list snapshots
[ 2437.090983] end_request: I/O error, dev fd0, sector 0
--------------------------------------------------------------------------------
Snapshot                          Origin  Pool    Volume size      Size  Type
--------------------------------------------------------------------------------
/dev/mypool/snap20150512T101210  lv01    mypool    160.00 MB  49.15 KB  linear
--------------------------------------------------------------------------------
```

每次原LVM中的数据更改，都可以手动执行ssm snapshot生成快照

当原LVM数据损坏就可以用快照恢复了

方式一是先将原逻辑卷卸除挂载 #umount /dev/vg00/lv00

然后挂载逻辑卷快照即可 #mount /dev/vg00/lvsp00 /data，就可以正常访问数据了

```
[root@server1 ~]# umount /dev/mypool/lv01
[root@server1 ~]# mount /dev/mypool/snap20150512T101821 /mnt/test/
[root@server1 ~]# ls /mnt/test/
group  passwd  shadow
```

方式二可以通过 lvconvert把快照的内容重新写回原有的lvm

先将原逻辑卷卸除挂载#umount /dev/vg00/lv00

执行lvconvert将快照的数据合并到原逻辑卷 #lvconvert --merge /dev/vg00/lvsp00

最后挂载原逻辑卷，查看数据是否恢复成功

```
[root@server1 ~]# lvconvert --merge /dev/mypool/snap20150512T101821
  Merging of volume snap20150512T101821 started.
  lv01: Merged: 100.0%
  lv01: Merged: 100.0%
  Merge of snapshot into logical volume lv01 has finished.
  Logical volume "snap20150512T101821" successfully removed
[root@server1 ~]# mount /dev/mypool/lv01 /mnt/test/
[root@server1 ~]# ls /mnt/test/
group  passwd  shadow
```

有关ssm的具体用法可以参考ssm的帮助手册页

如：删除LVM卷#ssm remove <volume>

删除存储池#ssm remove <pool-name>