在RedHat Linux上使用LVM管理磁盘并创建文件系统

其它功能 安恒 2007-09-17 发表

在RedHat Linux上使用LVM管理磁盘并创建文件系统

一、 组网需求：
RedHat Linux AS 4 Update2，H3C Neocean系列存储产品
RedHat Linux AS 4 Update2上通过软件iSCSI inititator访问Neocean存储设备上分配来的磁盘空间
二、 组网图：
无
三、 配置步骤：
1.首先在存储服设备上通过iSCSI协议为RedHat Linux主机分配存储空间，在RedHat Linux配置软件iSCSI initiator并识别到Neocean存储设备上分配来的磁盘空间，磁盘设备名为/dev/sdb，使用fdisk把/dev/sdb创建为1个主分区，分区类型设置为0x8e，它表示"Linux LVM"：

```
[root@as4 ~]# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.
The number of cylinders for this disk is set to 61439.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-61439, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-61439, default 61439):61439
Command (m for help): t
Selected partition 1
Hex code (type L to list codes):8e
Changed system type of partition 1 to 8e (Linux LVM)
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.
```

使用sfdisk -l /dev/sdb确认/dev/sdb上的分区情况和类型

```
[root@as4 ~]# sfdisk -l /dev/sdb
Disk /dev/sdb: 61439 cylinders, 64 heads, 32 sectors/track
Units = cylinders of 1048576 bytes, blocks of 1024 bytes, counting from 0
   Device Boot Start     End   #cyls    #blocks   Id  System
/dev/sdb1          0+  61438   61439-  62913520   8e  Linux LVM
/dev/sdb2          0      -       0          0    0  Empty
/dev/sdb3          0      -       0          0    0  Empty
/dev/sdb4          0      -       0          0    0  Empty
```

2.使用以下命令把/dev/sdb1设置为PV（物理卷）

```
[root@as4 ~]# pvcreate -f /dev/sdb1
Physical volume "/dev/sdb1" successfully created
```

pvcreate 在 /dev/sdb1 上设置一个特殊的"记帐"区域，称作 VGDA（"卷组描述符区域"），LVM 使用该区域来记录物理范围是如何分配的，以及其它一些操作。
3.创建卷组（VG）vg1并向该卷组添加 /dev/sdb1：

```
[root@as4 ~]# vgcreate vg1 /dev/sdb1
Volume group "vg1" successfully created
```

创建卷组之后，可以通过vgdisplay来查看一下vg1的属性:

```
[root@as4 ~]# vgdisplay -v vg1
    Using volume group(s) on command line
    Finding volume group "vg1"
  --- Volume group ---
  VG Name               vg1
  System ID
  Format                lvm2
  Metadata Areas        1
  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                1
  Act PV                1
  VG Size               60.00 GB
  PE Size               4.00 MB
  Total PE              15359
  Alloc PE / Size       0 / 0
  Free  PE / Size       15359 / 60.00 GB
  VG UUID               Z7W8Kd-8qtM-dk2Z-rzTo-Koyj-l4wH-lRRmkM
    --- Physical volumes ---
  PV Name               /dev/sdb1
  PV UUID               oScqA6-7BQc-WlLR-Ni3T-AbBm-ujxP-JJghXp
  PV Status             allocatable
  Total PE / Free PE    15359 / 15359
```

vgcreate 命令除了创建 "vg1" 卷组以外，它还设置 /dev/sdb1，设置它的物理分区 Physical Extents (PE)大小为4 MB，4 MB是缺省的物理分区 Physical Extents (PE)大小。这意味着在卷组上创建的所有逻辑卷都可以以 4 MB 为增量单位来进行扩充或缩减。

由于内核限制的原因，物理分区 Physical Extents (PE)大小决定了逻辑卷的最大大小。4 MB 的物理分区 Physical Extents (PE)大小决定了逻辑卷大小限制为 256 GB。如果要创建的卷大于 256 GB，那么你可以在运行 vgcreate 时指定更大一些的物理分区 Physical Extents (PE)大小。物理分区 Physical Extents (PE)大小可以是从 8 KB 到 512 MB 之间的任何值，并且必须总是 2 的倍数。通过将范围大小增加到 4 MB 以上，最大的物理卷大小将相应地增加到最大为 1 Petabyte（当前，许多x86 系统上的大小限制是 2 Terabytes）。

如果希望使用 32MB 的物理分区 Physical Extents (PE)创建卷组，可以使用如下操作：

```
[root@as4 ~]# vgcreate -s 32M vg1 /dev/sdb1
```

32 MB 是个合适的物理分区 Physical Extents (PE)大小，它使最大逻辑卷大小增加到 2 TB。

4.已创建了卷组vg1，那接下来就可以在卷组vg1上创建逻辑卷 Logical Volume (LV) 了。

在卷组vg1上创建一个名字为lvtest1，大小为36GB的逻辑卷

```
[root@as4 ~]# lvcreate -L 36G -n lvtest1 vg1
Logical volume "lvtest1" created
```

可以使用lvdisplay查看逻辑卷 Logical Volume (LV) lvtest1的属性:

```
[root@as4 ~]# lvdisplay /dev/vg1/lvtest1
  --- Logical volume ---
  LV Name                /dev/vg1/lvtest1
  VG Name                vg1
  LV UUID                KQf5zY-SGi2-KKdb-tmbn-KyWR-twGL-p2KdCB
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                36.00 GB
  Current LE             9216
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:0
```

6.接下来在逻辑卷lvtest1上创建ext3类型的文件系统：

```
[root@as4 ~]# mkfs -t ext3 /dev/vg1/lvtest1
mke2fs 1.35 (28-Feb-2004)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
4718592 inodes, 9437184 blocks
471859 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=12582912
288 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
This filesystem will be automatically checked every 39 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

使用tune2fs修改/dev/vg1/lvtest1的属性，去掉自动检查的属性。

```
[root@as4 ~]# tune2fs -c -1 -i 0 /dev/vg1/lvtest1
tune2fs 1.35 (28-Feb-2004)
Setting maximal mount count to -1
Setting interval between check 0 seconds
```

8.设置文件系统自动挂载

设置软件iSCSI服务随系统启动而自动启动

新建一个目录做为文件系统的挂载点：

```
[root@as4 ~]# mkdir /data
```

用vi编辑/etc/fstab文件，在文件中添加以下一行：

```
/dev/vg1/lvtest1    /data       ext3     _netdev        0  0
```

四、 配置关键点：

配置步骤1不是必须的，可以跳过（使用超过2TB的磁盘时，这一步必须跳过），然后在步骤2中直接把整块硬盘设置为PV，如：

```
[root@as4 ~]# pvcreate -f /dev/sdb
```