

NSD ENGINEER DAY01

1. [案例1：硬盘分区及格式化](#)
2. [案例2：新建一个逻辑卷](#)
3. [案例3：调整现有磁盘的分区](#)
4. [案例4：扩展逻辑卷的大小](#)

1 案例1：硬盘分区及格式化

1.1 问题

本例要求熟悉硬盘分区结构，使用fdisk分区工具在磁盘 /dev/vdb 上按以下要求建立分区：

1. 采用默认的 msdos 分区模式
2. 第1个分区 /dev/vdb1 的大小为 200MiB
3. 第2个分区 /dev/vdb2 的大小为 2000MiB
4. 第3个分区 /dev/vdb3 的大小为 1000MiB

完成分区后，能够配置开机自动挂载 /dev/vdb2 分区：

1. 文件系统类型为 EXT4
2. 将其挂载到 /mnt/part2 目录

1.2 方案

fdisk分区工具用来建立msdos分区方案，其交互模式中的主要指令如下：

- m：列出指令帮助
- p：查看当前的分区表信息
- n：新建分区
- d：删除分区
- t：更改分区标识
- q：放弃分区更改并退出
- w：保存对分区表所做的更改

1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：新建分区表

1) 打开fdisk工具，操作磁盘/dev/vdb

```
01. [root@server0 ~]# fdisk /dev/vdb
02. Welcome to fdisk (util-linux 2.23.2).
03.
04. Changes will remain in memory only, until you decide to write them.
05. Be careful before using the write command.
06.
```

[Top](#)

07. Device does not contain a recognized partition table
08. Building a new DOS disklabel with disk identifier 0x9ac1bc10.
- 09.
10. Command (m for help): //交互操作提示信息

2) 新建第1个分区/dev/vdb1

01. Command (m for help): n //新建分区
02. Partition type:
03. p primary (0 primary, 0 extended, 4 free)
04. e extended
05. Select (default p): p //类型为p (主分区)
06. Partition number (1-4, default 1): 1 //分区编号1
07. First sector (2048-20971519, default 2048): //起始位置默认
08. Using default value 2048
09. Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519): +200M
10. Partition 1 of type Linux and of size 200 MB is set //结束位置+200MB大小
- 11.
12. Command (m for help): p //确认当前分区表
13. ...
14. Device Boot Start End Blocks Id System
15. /dev/vdb1 2048 411647 204800 83 Linux

3) 新建第2个分区/dev/vdb2

01. Command (m for help): n
02. Partition type:
03. p primary (1 primary, 0 extended, 3 free)
04. e extended
05. Select (default p): p //类型为p (主分区)
06. Partition number (2-4, default 2): 2 //分区编号2
07. First sector (411648-20971519, default 411648): //起始位置默认
08. Using default value 411648
09. Last sector, +sectors or +size{K,M,G} (411648-20971519, default 20971519): +2000M
10. Partition 2 of type Linux and of size 2 GiB is set //结束位置+2000MB大小
- 11.
12. Command (m for help): p //确认当前分区表 [Top](#)
13. ...
14. Device Boot Start End Blocks Id System

```

15.  /dev/vdb1      2048    411647    204800  83 Linux
16.  /dev/vdb2      411648   4507647   2048000  83 Linux

```

4) 新建第3个分区/dev/vdb3

```

01.  Command (m for help): n
02.  Partition type:
03.    p primary ( 2 primary, 0 extended, 2 free)
04.    e extended
05.  Select (default p): p
06.  Partition number ( 3, 4, default 3): 3
07.  First sector ( 4507648- 20971519, default 4507648):
08.  Using default value 4507648
09.  Last sector, +sectors or +size{ K,M,G} ( 4507648- 20971519, default 20971519): +1000M
10.  Partition 3 of type Linux and of size 1000 MB is set
11.
12.  Command (m for help): p                                //确认当前分区表
13.  .. ..
14.      Device Boot      Start         End      Blocks   Id  System
15.  /dev/vdb1           2048        411647        204800   83  Linux
16.  /dev/vdb2          411648       4507647       2048000   83  Linux
17.  /dev/vdb3          4507648       6555647       1024000   83  Linux

```

5) 调整分区类型标识 (可选)

将/dev/vdb1的类型 (默认为83,表示EXT2/3/4分区) 修改为8e (LVM设备) :

```

01.  Command (m for help): t                                //修改分区类型标识
02.  Partition number ( 1-3, default 3): 1                  //指定第1个分区
03.  Hex code (type L to list all codes): 8e                //类型改为8e
04.  Changed type of partition 'Linux' to 'Linux LVM'
05.
06.  Command (m for help): p                                //确认当前分区表
07.  .. ..
08.      Device Boot      Start         End      Blocks   Id  System
09.  /dev/vdb1           2048        411647        204800   8e  Linux LVM
10.  /dev/vdb2          411648       4507647       2048000   83  Linux
11.  /dev/vdb3          4507648       6555647       1024000   83  Linux

```

[Top](#)

6) 保存分区更改, 退出fdisk分区工具

```

01. Command (m for help): w //保存并退出
02. The partition table has been altered!
03.
04. Calling ioctl() to re-read partition table.
05. Syncing disks.

```

6) 刷新分区表

```

01. [ root@server0 ~] # partprobe /dev/vdb //重新检测磁盘分区
02. //或者
03. [ root@server0 ~] # reboot //对已使用中磁盘的分区调整, 应该重启一次
04. ...

```

步骤二: 格式化及挂载分区

1) 将分区/dev/vdb2格式化为EXT4文件系统

```

01. [ root@server0 ~] # mkfs.ext4 /dev/vdb2
02. ...
03. Allocating group tables: done
04. Writing inode tables: done
05. Creating journal ( 8192 blocks): done
06. Writing superblocks and filesystem accounting information: done

```

2) 配置开机自动挂载

```

01. [ root@server0 ~] # vim /etc/fstab
02. ...
03. /dev/vdb2 /mnt/part2 ext4 defaults 0 0

```

3) 创建挂载点, 并验证挂载配置

```

01. [ root@server0 ~] # mkdir /mnt/part2 //创建挂载点
02. [ root@server0 ~] # mount -a //挂载fstab中的可用设备

```

[Top](#)

```

03. [root@server0 ~]# df -hT /mnt/part2/ //检查文档所在的文件系统及设备
04. Filesystem Type Size Used Avail Use% Mounted on
05. /dev/vdb2 ext4 1.9G 5.9M 1.8G 1% /mnt/part2

```

2 案例2：新建一个逻辑卷

2.1 问题

本例要求沿用前一天案例，使用分区 /dev/vdb1 构建 LVM 存储，相关要求如下：

1. 新建一个名为 systemvg 的卷组
2. 在此卷组中创建一个名为 vo 的逻辑卷，大小为180MiB
3. 将逻辑卷 vo 格式化为 EXT4 文件系统
4. 将逻辑卷 vo 挂载到 /vo 目录，并在此目录下建立一个测试文件 votest.txt，内容为 "I AM KING."

2.2 方案

LVM创建工具的基本用法：

```

01. vgcreate 卷组名 物理设备...
02. lvcreate -L 大小 -n 逻辑卷名 卷组名

```

2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：创建卷组

1) 新建名为systemvg的卷组

```

01. [root@server0 ~]# vgcreate systemvg /dev/vdb1
02. Physical volume "/dev/vdb1" successfully created
03. Volume group "systemvg" successfully created

```

2) 确认结果

```

01. [root@server0 ~]# vgscan
02. Reading all physical volumes. This may take a while...
03. Found volume group "systemvg" using metadata type lvm2

```

[Top](#)

步骤二：创建逻辑卷

1) 新建名为vo的逻辑卷

```
01. [root@server0 ~]# lvcreate -L 180MB -n vo systemvg
02. Logical volume "vo" created
```

2) 确认结果

```
01. [root@server0 ~]# lvscan
02. ACTIVE          '/dev/systemvg/vo' [180.00 MB] inherit
```

步骤三：格式化及挂载使用

1) 格式化逻辑卷/dev/systemvg/vo

```
01. [root@server0 ~]# mkfs.ext4 /dev/systemvg/vo
02. ...
03. Allocating group tables: done
04. Writing inode tables: done
05. Creating journal (4096 blocks): done
06. Writing superblocks and filesystem accounting information: done
```

2) 挂载逻辑卷/dev/systemvg/vo

```
01. [root@server0 ~]# mkdir /vo //创建挂载点
02. [root@server0 ~]# mount /dev/systemvg/vo /vo //挂载
03. [root@server0 ~]# df -hT /vo/ //检查结果
04. Filesystem      Type  Size  Used Avail Use% Mounted on
05. /dev/mapper/systemvg-vo ext4 171M 1.6M 157M 1% /vo
```

3) 访问逻辑卷/dev/systemvg/vo

```
01. [root@server0 ~]# cat /vo/votest.txt
02. I AM KING.
```

[Top](#)

3 案例3：调整现有磁盘的分区

3.1 问题

本例要求沿用前一天案例，对磁盘/dev/vdb的分区表进行调整，要求如下：不更改原有分区，利用剩余空间新增三个分区，大小依次为：500MiB、2000MiB、512MiB

然后再基于刚建立的 2000MiB 分区构建新的 LVM 存储：

1. 新的逻辑卷命名为 database，大小为50个物理扩展单元（Physical Extent），属于 datastore 卷组
2. 在 datastore 卷组中的所有逻辑卷，其物理扩展单元（Physical Extent）的大小为16MiB
3. 使用 EXT3 文件系统对逻辑卷 database 格式化，此逻辑卷应该在开机时自动挂载到 /mnt/database 目录

3.2 方案

创建卷组时，可以通过-s选项指定PE的大小。

在给新建的逻辑卷分配空间时，空间大小只能是PE大小的倍数。

3.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：调整现有磁盘分区

1) 新建扩展分区（使用剩余可用空间）

```

01. [root@server0 ~]# fdisk /dev/vdb
02.
03. Command (m for help): p                                //确认原有分区表
04. .. ..
05.   Device Boot      Start         End      Blocks    Id System
06.   /dev/vdb1        2048        411647      204800    8e  Linux LVM
07.   /dev/vdb2        411648      4507647      2048000    83  Linux
08.   /dev/vdb3        4507648      6555647      1024000    83  Linux
09.
10. Command (m for help): n                                //新建分区
11. Partition type:
12.   p   primary (3 primary, 0 extended, 1 free)
13.   e   extended
14. Select (default e): e                                    //类型指定为e（扩展分区）
15. Selected partition 4                                    //只有一个可用编号，自动选取
16. First sector (6555648-20971519, default 6555648):      //起始位置默认
17. Using default value 6555648
18. Last sector, +sectors or +size{K,M,G} (6555648-20971519, default 20971519):
19. Using default value 20971519                            //结束位置默认
20. Partition 4 of type Extended and of size 6.9 GiB is set
21.
22. Command (m for help): p
  
```

[Top](#)

```

23.  ...
24.      Device Boot      Start         End      Blocks   Id  System
25.      /dev/vdb1          2048        411647        204800    8e  Linux LVM
26.      /dev/vdb2        411648        4507647       2048000    83  Linux
27.      /dev/vdb3        4507648        6555647       1024000    83  Linux
28.      /dev/vdb4        6555648       20971519       7207936     5  Extended

```

2) 在扩展分区中新建3个逻辑分区

创建第1个逻辑卷（由于主分区编号已用完，分区类型自动选逻辑分区）：

```

01.  Command (m for help): n
02.  All primary partitions are in use
03.  Adding logical partition 5                //分区编号5
04.  First sector (6557696-20971519, default 6557696):      //起始位置默认
05.  Using default value 6557696
06.  Last sector, +sectors or +size{K,M,G} (6557696-20971519, default 20971519): +500M
07.                                          //结束位置默认
08.  Partition 5 of type Linux and of size 500 MB is set

```

创建第2个逻辑卷：

```

01.  Command (m for help): n
02.  All primary partitions are in use
03.  Adding logical partition 6                //分区编号6
04.  First sector (7583744-20971519, default 7583744):      //起始位置默认
05.  Using default value 7583744
06.  Last sector, +sectors or +size{K,M,G} (7583744-20971519, default 20971519): +2000M
07.                                          //结束位置默认
08.  Partition 6 of type Linux and of size 2 GiB is set

```

创建第3个逻辑卷：

```

01.  Command (m for help): n
02.  All primary partitions are in use
03.  Adding logical partition 7                //分区编号7
04.  First sector (11681792-20971519, default 11681792):    //起始位置默认
05.  Using default value 11681792

```

[Top](#)

06. Last sector, +sectors or +size{ K,M,G} (11681792 20971519, default 20971519) : +512M
07. //结束位置默认
08. Partition 7 of type Linux and of size 512 MB is set

根据预计的用途调整分区类型 (可选) :

01. Command (m for help) : t //修改
02. Partition number (1- 7, default 7) : 5 //第5个分区
03. Hex code (type L to list all codes) : 8e //类型为8e (LVM)
04. Changed type of partition 'Linux' to 'Linux LVM'
- 05.
06. Command (m for help) : t //修改
07. Partition number (1- 7, default 7) : 6 //第6个分区
08. Hex code (type L to list all codes) : 8e //类型为8e (LVM)
09. Changed type of partition 'Linux' to 'Linux LVM'
- 10.
11. Command (m for help) : t //修改
12. Partition number (1- 7, default 7) : 7 //第7个分区
13. Hex code (type L to list all codes) : 82 //类型为82 (交换分区)
14. Changed type of partition 'Linux' to 'Linux swap / Solaris'

确认分区结果并保存 :

01. Command (m for help) : p
02.
03. Device Boot Start End Blocks Id System
04. /dev/vdb1 2048 411647 204800 8e Linux LVM
05. /dev/vdb2 411648 4507647 2048000 83 Linux
06. /dev/vdb3 4507648 6555647 1024000 83 Linux
07. /dev/vdb4 6555648 20971519 7207936 5 Extended
08. /dev/vdb5 6557696 7581695 512000 8e Linux LVM
09. /dev/vdb6 7583744 11679743 2048000 8e Linux LVM
10. /dev/vdb7 11681792 12730367 524288 82 Linux swap / Solaris
- 11.
12. Command (m for help) : w //保存退出
13. The partition table has been altered!
14. [Top](#)
15. Calling ioctl() to re-read partition table.
- 16.

17. WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
18. The kernel still uses the old table. The new table will be used at
19. the next reboot or after you run `partprobe(8)` or `kpartx(8)`
20. Syncing disks. //提示重启

3) 刷新分区表

01. [root@server0 ~] # `partprobe /dev/vdb`
02. [root@server0 ~] # `reboot`

步骤二：新建卷组、逻辑卷

1) 新建卷组datastore，指定PE大小为16MiB

01. [root@server0 ~] # `vgcreate -s 16MiB datastore /dev/vdb6`
02. Volume group "datastore" successfully created
03. [root@server0 ~] # `vgscan` //确认新建的卷组
04. Reading all physical volumes. This may take a while...
05. Found volume group "systemvg" using metadata type lvm2
06. Found volume group "datastore" using metadata type lvm2

2) 新建逻辑卷database，大小设置为50个PE

01. [root@server0 ~] # `lvcreate -l 50 -n database datastore`
02. Logical volume "database" created
03. [root@server0 ~] # `lvscan` //确认新建的逻辑卷
04. ACTIVE '/dev/systemvg/v0' [180.00 MiB] inherit
05. ACTIVE '/dev/datastore/database' [800.00 MiB] inherit

步骤三：格式化及使用逻辑卷

1) 格式化逻辑卷/dev/datastore/database

01. [root@server0 ~] # `mkfs.ext3 /dev/datastore/database`
02. ...
03. Allocating group tables: done
04. Writing inode tables: done
05. Creating journal (4096 blocks): done

[Top](#)

06. Writing superblocks and filesystem accounting information: done

2) 配置开机挂载

```
01. [root@server0 ~] # mkdir /mnt/database //创建挂载点
02. [root@server0 ~] # vim /etc/fstab
03. ...
04. /dev/datastore/database /mnt/database ext3 defaults 0 0
```

3) 验证挂载配置

```
01. [root@server0 ~] # mount -a
02. [root@server0 ~] # df -hT /mnt/database/ //确认挂载点设备
03. Filesystem Type Size Used Avail Use% Mounted on
04. /dev/mapper/datastore-database ext3 772M 828K 715M 1% /mnt/database
```

4 案例4：扩展逻辑卷的大小

4.1 问题

本例要求沿用练习一，将逻辑卷 vo 的大小调整为 300MiB，要求如下：

1. 原文件系统中的内容必须保持完整
2. 必要时可使用之前准备的分区 /dev/vdb5 来补充空间
3. 注意：分区大小很少能完全符合要求的大小，所以大小在270MiB和300MiB之间都是可以接受的

4.2 方案

对于已经格式化好的逻辑卷，在扩展大小以后，必须通知内核新大小。

如果此逻辑卷上的文件系统是EXT3/EXT4类型，需要使用resize2fs工具；

如果此逻辑卷上的文件系统是XFS类型，需要使用xfs_growfs。

4.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：确认逻辑卷vo的信息

1) 找出逻辑卷所在卷组

```
01. [root@server0 ~] # lvscan
02. ACTIVE          '/dev/systemvg/vo' [ 180.00 MB] inherit
```

[Top](#)

```
03.    ACTIVE    '/dev/datastore/database' [ 800.00 MB] inherit
```

2) 查看该卷组的剩余空间是否可满足扩展需要

```
01.    [root@server0 ~]# vgdisplay systemvg
02.    --- Volume group ---
03.    VG Name            systemvg
04.    System ID
05.    Format              lvm2
06.    Metadata Areas      1
07.    Metadata Sequence No 2
08.    VG Access           read/write
09.    VG Status           resizable
10.    MAX LV              0
11.    Cur LV              1
12.    Open LV            0
13.    Max PV              0
14.    Cur PV              1
15.    Act PV              1
16.    VG Size            196.00 MB           //卷组总大小
17.    PE Size            4.00 MB
18.    Total PE           49
19.    Alloc PE / Size    45 / 180.00 MB
20.    Free PE / Size     4 / 16.00 MB       //剩余空间大小
21.    VG UUID            czp8Lj-jihS-Ddoh-ny38-j521-5X8J-gqQfUN
```

此例中卷组systemvg的总大小都不够300MiB、剩余空间才16MiB，因此必须先扩展卷组。只有剩余空间足够，才可以直接扩展逻辑卷大小。

步骤二：扩展卷组

1) 将提前准备的分区/dev/vdb5添加到卷组systemvg

```
01.    [root@server0 ~]# vgextend systemvg /dev/vdb5
02.    Physical volume "/dev/vdb5" successfully created
03.    Volume group "systemvg" successfully extended
```

2) 确认卷组新的大小

[Top](#)

```

01. [root@server0 ~]# vgdisplay systemvg
02. --- Volume group ---
03. VG Name          systemvg
04. ...
05. VG Size          692.00 MB           //总大小已变大
06. PE Size          4.00 MB
07. Total PE         173
08. Alloc PE / Size  45 / 180.00 MB
09. Free PE / Size   128 / 512.00 MB     //剩余空间已达512MB
10. VG UUID          czp8lJ-jihS-Ddoh-ny38-j521-5X8J-gqQfUN

```

步骤三：扩展逻辑卷大小

1) 将逻辑卷/dev/systemvg/vo的大小调整为300MiB

```

01. [root@server0 ~]# lvextend -L 300MB /dev/systemvg/vo
02. Extending logical volume vo to 300.00 MB
03. Logical volume vo successfully resized

```

2) 确认调整结果

```

01. [root@server0 ~]# lvscan
02. ACTIVE          '/dev/systemvg/vo' [ 300.00 MB] inherit
03. ACTIVE          '/dev/datastore/database' [ 800.00 MB] inherit

```

3) 刷新文件系统大小

确认逻辑卷vo上的文件系统类型：

```

01. [root@server0 ~]# blkid /dev/systemvg/vo
02. /dev/systemvg/vo: UUID="d4038749-74c3-4963-a267-94675082a48a" TYPE="ext4"

```

选择合适的工具刷新大小：

```

01. [root@server0 ~]# resize2fs /dev/systemvg/vo
02. resize2fs 1.42.9 (28-Dec-2013)
03. Resizing the filesystem on /dev/systemvg/vo to 307200 (1k) blocks.

```

[Top](#)

04. The filesystem on `/dev/systemvg/vo` is now 307200 blocks long.

确认新大小 (约等于300MiB) :

```
01. [root@server0 ~]# mount /dev/systemvg/vo /vo/
02. [root@server0 ~]# df -hT /vo
03. Filesystem      Type  Size  Used Avail Use% Mounted on
04. /dev/mapper/systemvg-vo ext4 287M 2.1M 266M 1% /vo
```

[Top](#)