

Elastic Volume Service

Best Practices

Issue 01
Date 2020-03-12



Copyright © Huawei Technologies Co., Ltd. 2020. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 EVS Disk Management Using LVM.....	1
1.1 LVM Overview.....	1
1.2 Installing LVM.....	2
1.3 Creating Logical Volumes Using LVM.....	3
1.4 Creating and Mounting a File System.....	7
1.5 Expanding the Capacity of a Logical Volume Using the Unallocated Capacity.....	9
1.6 Adding an EVS Disk to Expand the Capacity of a Volume Group.....	11
1.7 Glossary.....	13
2 Handling Insufficient Disk Space on a Windows ECS.....	14
2.1 Overview.....	14
2.2 Clearing Disk Space Using the System Built-in Cleanup Tool.....	14
2.3 Uninstalling Unnecessary Programs on Control Panel.....	16
3 RAID Array Creation with EVS Disks.....	19
3.1 Overview.....	19
3.2 Resource Planning.....	21
3.3 Creating Resources.....	22
3.3.1 Creating an ECS.....	22
3.3.2 Creating and Attaching EVS Disks.....	23
3.4 Creating a RAID Array Using mdadm.....	24
3.5 Configuring Automatic Start of the RAID Array at Server Startup.....	27
4 Extending Disk Partitions and File Systems (Linux with a Kernel Earlier Than 3.6.0).....	29
4.1 Partition and File System Extension Preparations (Linux).....	29
4.2 Extending Partitions and File Systems for System Disks (Linux).....	33
4.3 Extending Partitions and File Systems for Data Disks (Linux).....	43
4.4 Extending Partitions and File Systems for SCSI Disks (Linux).....	63
5 FAQs.....	76
5.1 How Can I Obtain ECS NIC Information?.....	76

1 EVS Disk Management Using LVM

1.1 LVM Overview

To use an Elastic Volume Service (EVS) disk in Linux, partitions need to be created on this disk. If the partition space becomes insufficient, you can handle the insufficiency in either of the following ways:

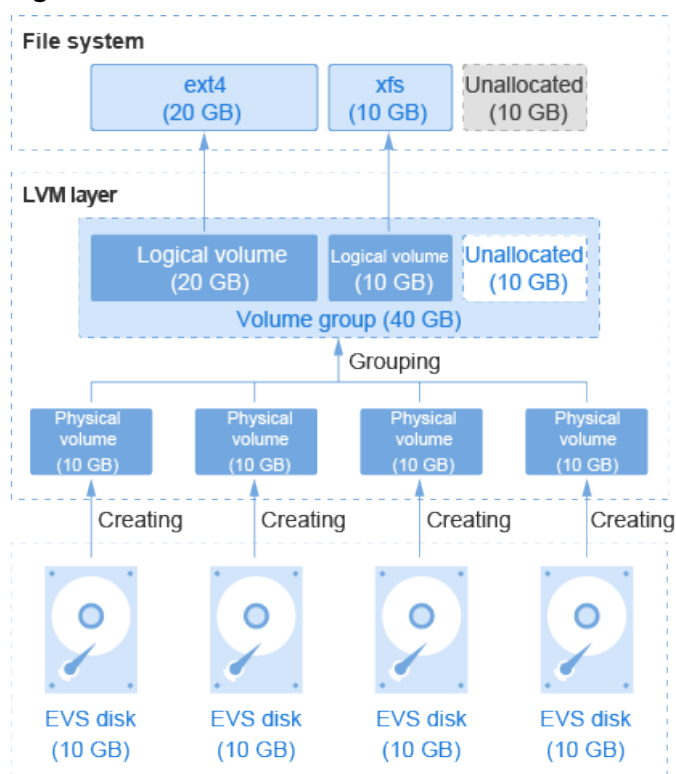
- Expand an existing partition (services will be interrupted).
- Create a new partition (services will not be interrupted).

Both ways are not ideal as either services need to be interrupted or new partitions need to be created. In this case, LVM can perfectly handle the space insufficiency.

Basic Principles of LVM

LVM is short for Logical Volume Manager, which is a mechanism used for managing disk partitions in Linux.

By adding a logical layer between EVS disks and file systems, LVM shields the EVS disk partition layout for file systems, thus improving flexibility in managing EVS disks. [Figure 1-1](#) shows the LVM architecture.

Figure 1-1 LVM architecture

The process of managing EVS disks through LVM is as follows:

1. **Create physical volumes for EVS disks.**
2. **Create a volume group for the physical volumes.**
3. **Create logical volumes in the volume group.**
4. **Create file systems on logical volumes.**

When EVS disks are managed using LVM, file systems are distributed on multiple EVS disks and can be easily resized as needed. Therefore, file system spaces will no longer be limited by EVS disk capacities.

For example, you can expand the size of a ext4 file system in either of the following methods:

- Expand the capacity of the logical volume directly if the unallocated space in the volume group is sufficient.
- Expand the capacity of the volume group and then expand the capacity of the logical volume if the unallocated space in a volume group is insufficient.

1.2 Installing LVM

Scenarios

By default, LVM is not installed in the Elastic Cloud Server (ECS) operating system (OS), and you need to manually install it. This topic shows how to check whether LVM is installed on your ECS and how to install LVM.

In this document, CentOS 7.4 64bit is used as the sample OS. The method for formatting an EVS disk varies depending on the OS running on the server. This document is used for reference only.

Prerequisites

You have an ECS and have bound an elastic IP address (EIP) for the ECS.

Procedure

Step 1 Log in to the ECS as user **root**.

Step 2 Run the following command to check whether LVM is installed:

```
rpm -qa |grep lvm2
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# rpm -qa |grep lvm2
lvm2-libs-2.02.177-4.el7.x86_64
lvm2-2.02.177-4.el7.x86_64
```

- If the preceding information is displayed, LVM has been installed. Go to [Creating Logical Volumes Using LVM](#).
- If the preceding information is not displayed, LVM has not been installed. Go to [Step 3](#).

Step 3 Run the following command and follow the prompts to install LVM:

```
yum install lvm2
```

Information similar to the following is displayed:

```
.....
Installed:
  lvm2.x86_64 7:2.02.177-4.el7

Dependency Installed:
  device-mapper-event.x86_64 7:1.02.146-4.el7                device-mapper-event-libs.x86_64
  7:1.02.146-4.el7
  device-mapper-persistent-data.x86_64 0:0.7.3-3.el7         lvm2-libs.x86_64 7:2.02.177-4.el7

Dependency Updated:
  device-mapper.x86_64 7:1.02.146-4.el7                    device-mapper-libs.x86_64 7:1.02.146-4.el7

Complete!
```

When **Complete!** is displayed, LVM is successfully installed.

----End

1.3 Creating Logical Volumes Using LVM

Scenarios

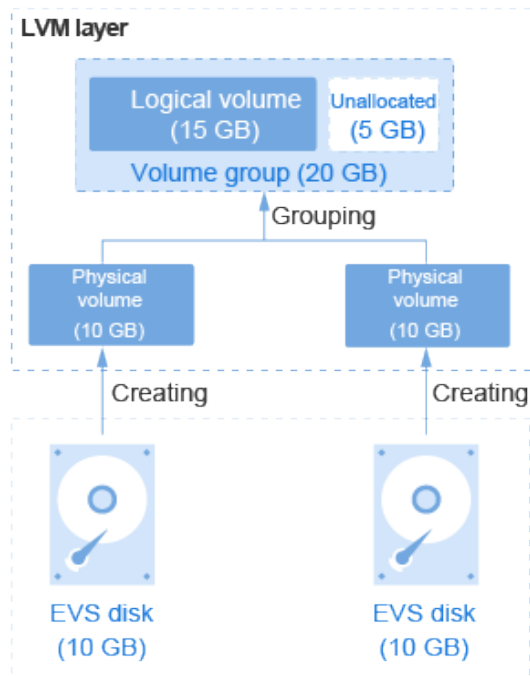
This topic shows how to create a 15 GB logical volume based on two 10 GB EVS disks.

NOTE

Multiple EVS disks with different specifications can be used to create logical volumes.

The process of creating a logical volume includes creating physical volumes, create a volume group, and create logical volumes.

Figure 1-2 Process of creating an LVM logical volume



Prerequisites

Two EVS disks have been attached to the ECS where LVM is installed.

Procedure

Step 1 Log in to the ECS as user **root**.

Step 2 Run the following command to query and take note of the disk device names:

```
fdisk -l | grep /dev/vd | grep -v vda
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# fdisk -l | grep /dev/vd | grep -v vda  
Disk /dev/vdb: 10.7 GB, 10737418240 bytes, 20971520 sectors  
Disk /dev/vdc: 10.7 GB, 10737418240 bytes, 20971520 sectors
```

In the command output, two EVS disks are attached to the ECS, and the device names are **/dev/vdb** and **/dev/vdc**.

Step 3 Run the following command to create physical volumes based on the EVS disks:

```
pvcreate Device name 1 Device name 2 Device name 3..
```

Parameter description:

Device name: indicates the disk device name. If multiple physical volumes need to be created in a batch, specify multiple device names and separate them with spaces.

For example, run the following command:

pvcreate /dev/vdb /dev/vdc

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# pvcreate /dev/vdb /dev/vdc
Physical volume "/dev/vdb" successfully created.
Physical volume "/dev/vdc" successfully created.
```

Step 4 Run the following command to query details of the physical volumes:

pvdisplay

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# pvdisplay
"/dev/vdc" is a new physical volume of "10.00 GiB"
--- NEW Physical volume ---
PV Name          /dev/vdc
VG Name
PV Size          10.00 GiB
Allocatable      NO
PE Size          0
Total PE         0
Free PE          0
Allocated PE     0
PV UUID          dypyLh-xjlj-PvG3-jD0j-yup5-O7SI-462R7C

"/dev/vdb" is a new physical volume of "10.00 GiB"
--- NEW Physical volume ---
PV Name          /dev/vdb
VG Name
PV Size          10.00 GiB
Allocatable      NO
PE Size          0
Total PE         0
Free PE          0
Allocated PE     0
PV UUID          srv5H1-tgLu-GRTL-Vns8-GfNK-jtHk-Ag4HHB
```

In the command output, the system has two new physical volumes named **/dev/vdc** and **/dev/vdb**.

Step 5 Run the following command to create a volume group:

```
vgcreate Volume group name Physical volume name 1 Physical volume name 2
Physical volume name 3...
```

Parameter descriptions:

- *Volume group name*: Specify a volume group name, for example, **vgdata**.
- *Physical volume name*: Specify the names of the physical volumes to be added to the volume group. Multiple names are separated with spaces.

For example, run the following command:

```
vgcreate vgdata /dev/vdb /dev/vdc
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# vgcreate vgdata /dev/vdb /dev/vdc
Volume group "vgdata" successfully created
```

Step 6 Run the following command to query details of the volume group:

vgdisplay

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# vgdisplay
--- Volume group ---
VG Name                vgdata
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               19.99 GiB
PE Size               4.00 MiB
Total PE              5118
Alloc PE / Size       0 / 0
Free PE / Size        5118 / 19.99 GiB
VG UUID                NLkZV7-hYYE-0w66-tnlt-Y6jL-lk7S-76w4P6
```

Step 7 Run the following command to create a logical volume:

lvcreate -L *Logical volume size* **-n** *Logical volume name* *Volume group name*

Parameter descriptions:

- *Logical volume size*: Specify a value smaller than the volume group's available space, either in MB or GB.
- *Logical volume name*: Specify the volume name, for example, **lvdata1**.
- *Volume group name*: Specify the name of the volume group where the logical volumes belong.

For example, run the following command:

lvcreate -L 15GB -n lvdata1 vgdata

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# lvcreate -L 15GB -n lvdata1 vgdata
Logical volume "lvdata1" created.
```

Step 8 Run the following command to query details of the logical volume:

lvdisplay

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# lvdisplay
--- Logical volume ---
LV Path                /dev/vgdata/lvdata1
LV Name                lvdata1
VG Name                vgdata
LV UUID                c7mNcF-CdPW-5PLD-1gVj-QZpB-nHfy-PHXchV
LV Write Access        read/write
LV Creation host, time ecs-lvmtest.novalocal, 2018-11-29 11:28:18 +0800
LV Status              available
# open                 0
LV Size                15.00 GiB
Current LE             3840
Segments               2
Allocation              inherit
Read ahead sectors     auto
```

```
- currently set to 8192  
Block device      252:0
```

----End

1.4 Creating and Mounting a File System

Scenarios

After the logical volume is created, you need to create a file system on the logical volume and mount the file system to the corresponding directory. This topic shows how to create a ext4 file system on a logical volume and mount the file system to the **/Data1** directory.

Procedure

Step 1 Log in to the ECS as user **root**.

Step 2 Run the following command to create a file system:

```
mkfs.File system format Logical volume path
```

For example, run the following command:

```
mkfs.ext4 /dev/vgdata/lvdata1
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# mkfs.ext4 /dev/vgdata/lvdata1  
mke2fs 1.42.9 (28-Dec-2013)  
Filesystem label=  
OS type: Linux  
Block size=4096 (log=2)  
Fragment size=4096 (log=2)  
Stride=0 blocks, Stripe width=0 blocks  
983040 inodes, 3932160 blocks  
196608 blocks (5.00%) reserved for the super user  
First data block=0  
Maximum filesystem blocks=2151677952  
120 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, 1605632, 2654208  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (32768 blocks): done  
Writing superblocks and filesystem accounting information: done
```

Step 3 Run the following command to create a mounting directory:

```
mkdir Mounting directory
```

For example, run the following command:

```
mkdir /Data1
```

Step 4 Run the following command to mount the file system on the directory:

```
mount Logical volume path Mounting directory
```

For example, run the following command:

```
mount /dev/vgdata/lvdata1 /Data1
```

Step 5 Run the following command to query the file system mounting information:

```
mount | grep Mounting directory
```

For example, run the following command:

```
mount | grep /Data1
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# mount | grep /Data1
/dev/mapper/vgdata-lvdata1 on /Data1 type ext4 (rw,relatime,data=ordered)
```

In the command output, **dev/mapper/vgdata-lvdata1** indicates the file system path. Take note of this path, which will be used in [Step 6](#).

Step 6 Perform the following operations to enable automatic mounting of the file system at the system start:

If this is not configured, you need to manually mount the file system when the ECS is restarted.

1. Run the following command to query the file system UUID:

```
blkid File system path
```

For example, run the following command to query the UUID of **dev/mapper/vgdata-lvdata1**:

```
blkid /dev/mapper/vgdata-lvdata1
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# blkid /dev/mapper/vgdata-lvdata1
/dev/mapper/vgdata-lvdata1: UUID="c6a243ce-5150-41ac-8816-39db54d1a4b8" TYPE="ext4"
```

In the command output, the UUID is **c6a243ce-5150-41ac-8816-39db54d1a4b8**.

2. Run the following command to open the **/etc/fstab** file:

```
vi /etc/fstab
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# vi /etc/fstab

#
# /etc/fstab
# Created by anaconda on Tue Nov 7 14:28:26 2017
#
# 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
#
UUID=27f9be47-838b-4155-b20b-e4c5e013cdf3 / ext4 defaults 1 1
UUID=2b2000b1-f926-4b6b-ade8-695ee244a901 /boot ext4 defaults 1 2
```

3. Press **i** to enter editing mode.

4. Move the cursor to the end of the file and press **Enter**. Then add the following information:

```
UUID=c6a243ce-5150-41ac-8816-39db54d1a4b8 /Data1 ext4 defaults 0 0
```

The file content is described as follows:

- Column 1: indicates the UUID. Enter the UUID queried in [1](#).
- Column 2: indicates the file system's mounting directory. Enter mounting directory **/Data1** created in [Step 3](#).

- Column 3: indicates the file system format. Enter file system format **ext4** configured in [Step 2](#).
 - Column 4: indicates the mounting option. In this example, **defaults** is used.
 - Column 5: indicates the backup option. Enter either **1** (the system automatically backs up the file system) or **0** (does not back up the file system). In this example, **0** is used.
 - Column 6: indicates the scanning option. Enter either **1** (the system automatically scans the file system at system start) or **0** (does not scan the file system). In this example, **0** is used.
5. Press **Esc**, enter **:wq!**, and press **Enter**.
The system saves the modifications and exits the vi editor.

Step 7 Perform the following operations to verify the automatic mounting function:

1. Run the following command to unmount a file system:
umount *Logical volume path*
For example, run the following command:
umount /dev/vgdata/lvdata1
2. Run the following command to reload all the content in the **/etc/fstab** file:
mount -a
3. Run the following command to query the file system mounting information:
mount | grep Mounting directory
For example, run the following command:
mount | grep /Data1
If information similar to the following is displayed, the automatic mounting function takes effect:

```
[root@ecs-lvmtest ~]# mount | grep /Data1  
/dev/mapper/vgdata-lvdata1 on /Data1 type ext4 (rw,relatime,data=ordered)
```

----End

1.5 Expanding the Capacity of a Logical Volume Using the Unallocated Capacity

Scenarios

When the capacity of a logical volume becomes insufficient, you can expand its capacity. This topic shows how to add 4 GB space for a 15 GB logical volume, which cannot meet requirements.

NOTE

During the capacity expansion, ensure that the volume group where the to-be-expanded logical volume belongs has sufficient available space. If the volume group's available space is also insufficient, expand the capacity of the volume group according to [Adding an EVS Disk to Expand the Capacity of a Volume Group](#).

Procedure

Step 1 Log in to the ECS as user **root**.

Step 2 Run the following command to expand the capacity of the logical volume:

```
lvextend -L +Additional capacity Logical volume path
```

Parameter descriptions:

- *Additional capacity*: Specify a value smaller than the volume group's available space, either in MB or GB.
- *Logical volume path*: Specify the path of the to-be-expanded logical volume.

For example, run the following command:

```
lvextend -L +4GB /dev/vgdata/lvdata1
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# lvextend -L +4GB /dev/vgdata/lvdata1
Size of logical volume vgdata/lvdata1 changed from 15.00 GiB (3840 extents) to 19.00 GiB (4864 extents).
Logical volume vgdata/lvdata1 successfully resized.
```

This step expands only the capacity of the logical volume. You also need to expand the size of the file system on this volume.

Step 3 Run the following command to expand the size of the file system:

```
resize2fs Logical volume path
```

For example, run the following command:

```
resize2fs /dev/vgdata/lvdata1
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# resize2fs /dev/vgdata/lvdata1
resize2fs 1.42.9 (28-Dec-2013)
Filesystem at /dev/vgdata/lvdata1 is mounted on /Data1; on-line resizing required
old_desc_blocks = 4, new_desc_blocks = 28
The filesystem on /dev/vgdata/lvdata1 is now 3657728 blocks long.
```

Step 4 Run the following command to check whether the file system size increases:

```
df -h
```

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda2        39G  1.5G  35G   5% /
devtmpfs        487M   0  487M   0% /dev
tmpfs           496M   0  496M   0% /dev/shm
tmpfs           496M  6.7M  490M   2% /run
tmpfs           496M   0  496M   0% /sys/fs/cgroup
/dev/vda1       976M  131M  779M  15% /boot
tmpfs           100M   0  100M   0% /run/user/0
/dev/mapper/vgdata-lvdata1 19G  44M  18G   1% /Data1
```

In the command output, the size of file system **/dev/mapper/vgdata-lvdata1** increases by 4 GB.

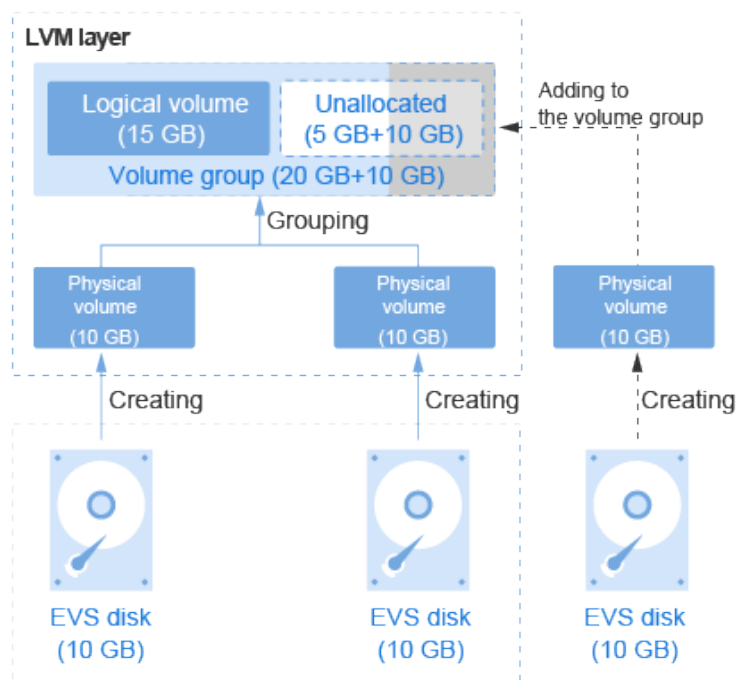
----End

1.6 Adding an EVS Disk to Expand the Capacity of a Volume Group

Scenarios

When the space of the LVM volume group cannot meet your requirements, you can expand the volume group capacity by creating EVS disks, creating physical volumes, and adding physical volumes to the volume group.

Figure 1-3 Sample of expanding a volume group



Procedure

Step 1 Create an EVS disk and attach it.

1. Log in to the management console.
2. Under **Storage**, click **Elastic Volume Service**. The disk list page is displayed.
3. Click **Buy Disk** and create a disk.
For details about how to create EVS disks, see [Elastic Volume Service User Guide](#).
4. In the disk list, locate the row that contains the new disk and click **Attach** in the **Operation** column.
5. Select the target ECS and select the device name from the drop-down list. Ensure that the EVS disk and ECS are in the same AZ.

Return to the disk list page. The status of the disk is **Attaching**, indicating that the disk is being attached to the ECS. When the disk status changes to **In-use**, the disk is successfully attached.

Step 2 Log in to the ECS as user **root**.

Step 3 Run the following command to query the capacity of the volume group:

vgdisplay

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# vgdisplay
--- Volume group ---
VG Name          vgdata
System ID
Format           lvm2
Metadata Areas   2
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          2
Act PV           2
VG Size          19.99 GiB
PE Size          4.00 MiB
Total PE         5118
Alloc PE / Size  4864 / 19.00 GiB
Free PE / Size   254 / 1016.00 MiB
VG UUID          NLkZV7-hYYE-0w66-tnlt-Y6jL-lk7S-76w4P6
```

In the command output, the **VG Size** value indicates the volume group capacity, which is **19.99 GiB** in this example.

Step 4 Run the following command to view and take note of the disk device names:

fdisk -l | grep /dev/vd | grep -v vda

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# fdisk -l | grep /dev/vd | grep -v vda
Disk /dev/vdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/vdc: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/vdd: 10.7 GB, 10737418240 bytes, 20971520 sectors
```

In the command output, the new EVS disk is attached to the ECS, and the device name is **/dev/vdd**.

Step 5 Run the following command to create a physical volume based on the new EVS disk:

pvcreate *Disk device name*

For example, run the following command:

pvcreate /dev/vdd

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# pvcreate /dev/vdd
Physical volume "/dev/vdd" successfully created.
```

Step 6 Run the following command to add the physical volume to the volume group to expand the volume group capacity:

vgextend *Volume group name physical volume name*

For example, run the following command:

vgextend vgdata /dev/vdd

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# vgextend vgdata /dev/vdd
Volume group "vgdata" successfully extended
```

Step 7 Run the following command to query details of the volume group:

vgdisplay

Information similar to the following is displayed:

```
[root@ecs-lvmtest ~]# vgdisplay
--- Volume group ---
VG Name                vgdata
System ID
Format                 lvm2
Metadata Areas         3
Metadata Sequence No   4
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                 <29.99 GiB
PE Size                 4.00 MiB
Total PE                7677
Alloc PE / Size        4864 / 19.00 GiB
Free PE / Size          2813 / <10.99 GiB
VG UUID                 NLkZV7-hYYE-0w66-tnlt-YGjL-lk7S-76w4P6
```

In the command output, 10 GB has been added to the **VG Size** volume group.

----End

1.7 Glossary

- **Physical Volume**
Physical volumes are basic storage devices of LVM and are created based on EVS disks and LVM management parameters.
- **Volume Group**
A volume group concatenates physical volumes into a large storage pool that can be consecutively addressed connects all physical volumes at the beginning and end of a logic.
- **Logical Volume**
Logical volumes are obtained by partitioning the volume group according to the logic.

2 Handling Insufficient Disk Space on a Windows ECS

2.1 Overview

When the disk space on a server is insufficient, the server running speed will be affected, which will further affect user experience. You can handle the disk insufficiency in either of the following ways:

- Clear the disk space.
 - [Clearing Disk Space Using the System Built-in Cleanup Tool](#)
 - [Uninstalling Unnecessary Programs on Control Panel](#)
- Expand the disk capacity.
 - [Expanding an In-use EVS Disk](#)
 - [Expanding an Available EVS Disk](#)

This document is written based on the best practices of Elastic Volume Service (EVS) disks. Windows 2016 is used as the sample server OS in the following topics, which describe the common operations for clearing disk space. In addition, it is recommended that you clear redundant files on a regular basis to save disk space.

- Periodically compress and save the files that are not frequently used.
- Periodically use the disk cleanup tool to clean up disk space, delete unnecessary files, and clean up the recycle bin.
- Uninstall unnecessary programs to release disk space.

2.2 Clearing Disk Space Using the System Built-in Cleanup Tool

Scenarios

This topic shows how to use the disk cleanup tool provided by the Windows OS to clean up disks with insufficient space.

In this document, Windows Server 2016 Standard 64bit is used as the sample server OS. The method for cleaning up disk space varies depending on the server OS. This document is used for reference only. For detailed operations and differences, see the corresponding OS documents.

Procedure

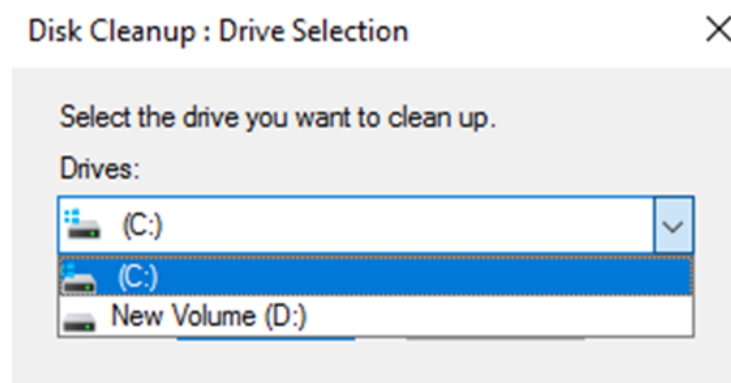
Step 1 On the server desktop, click the start icon in the lower left corner.

The start menu is displayed.

Step 2 In the navigation pane on the left, choose **Windows Administrative Tools > Disk Cleanup**.

The **Disk Cleanup: Drive Selection** window is displayed.

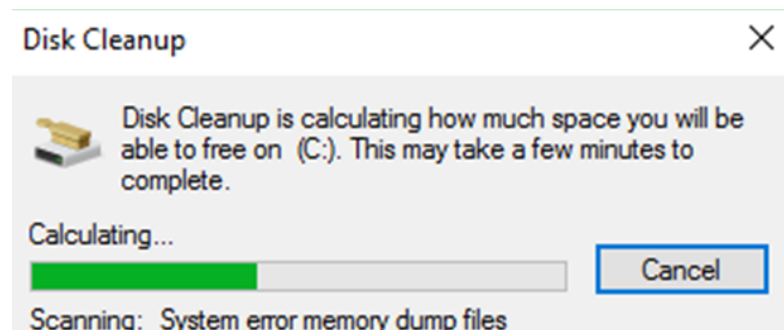
Figure 2-1 Disk Cleanup: Drive Selection



Step 3 Select the target disk from the drop-down list. In this example, disk (C:) is selected.

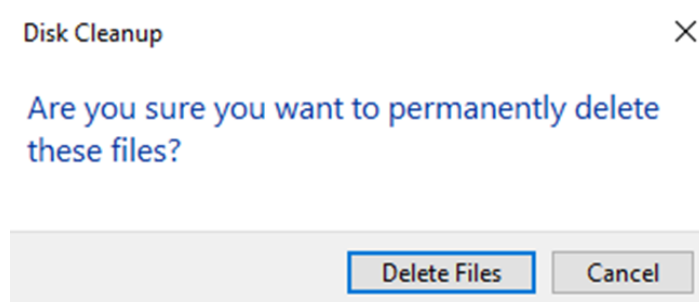
The **Disk Cleanup** window is displayed, and the system automatically calculates the space that can be freed on disk (C:).

Figure 2-2 Disk Cleanup



Step 4 After the automatic calculation is complete, select the files to be deleted on the displayed window and click **OK**.

A confirmation dialog box is displayed.

Figure 2-3 Deletion confirmation

Step 5 Click **Delete Files** to clean up the disk space.

----End

2.3 Uninstalling Unnecessary Programs on Control Panel

Scenarios

This topic shows how to uninstall unnecessary programs on the server control panel.

In this document, Windows Server 2016 Standard 64bit is used as the sample server OS. The method for cleaning up disk space varies depending on the server OS. This document is used for reference only. For detailed operations and differences, see the corresponding OS documents.

Procedure

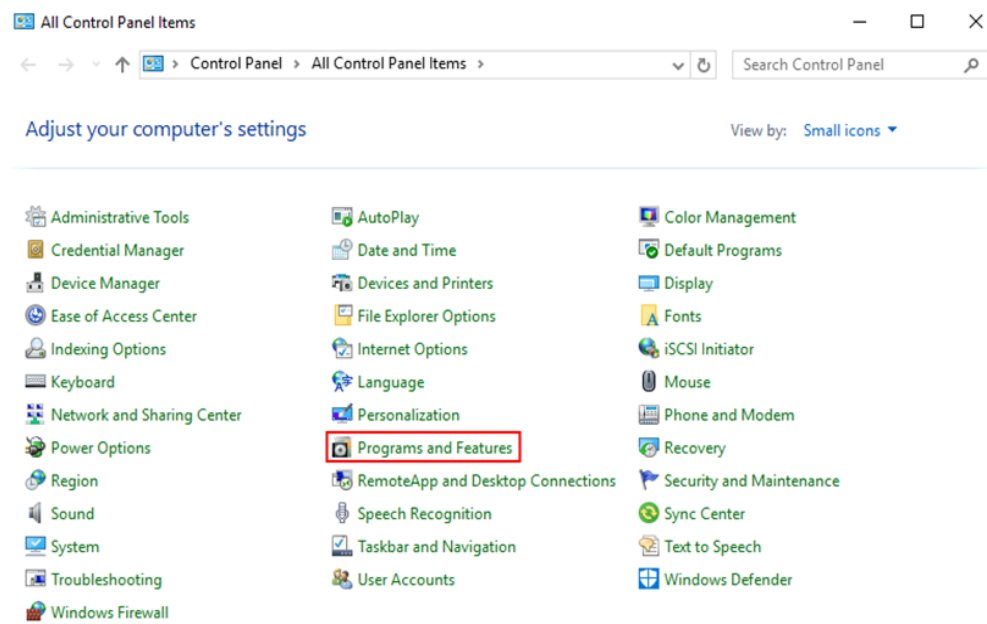
Step 1 On the server desktop, click the start icon in the lower left corner.

The start menu is displayed.

Step 2 In the navigation pane on the left, choose **Windows System > Control Panel**.

The **All Control Panel Items** window is displayed.

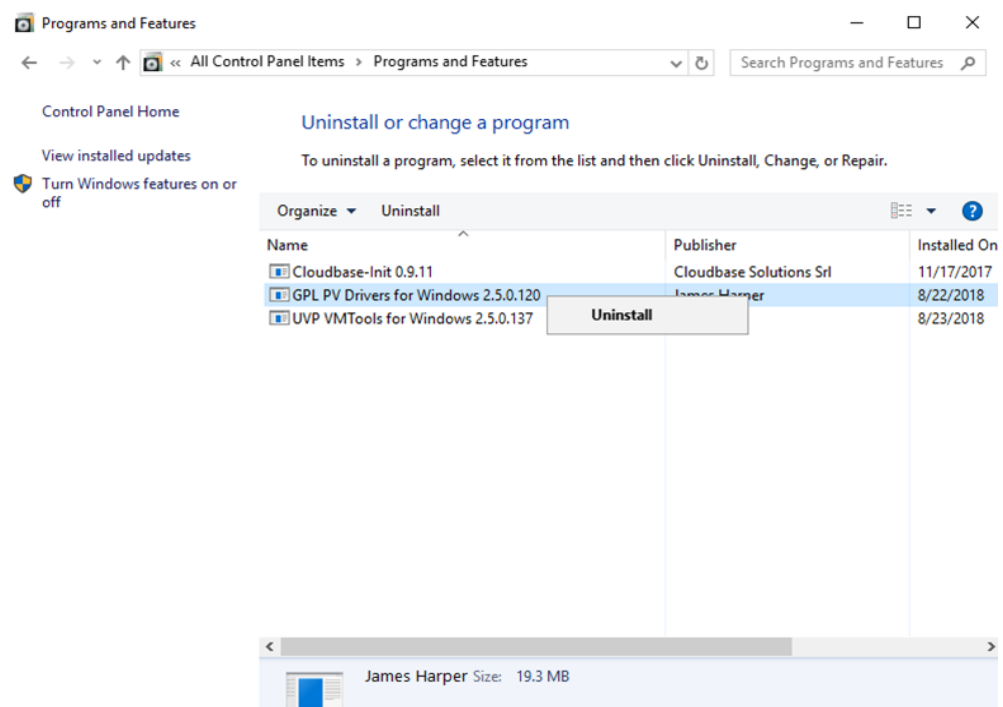
Figure 2-4 All Control Panel Items



Step 3 In the list, select **Programs and Features**.

The **Programs and Features** window is displayed.

Figure 2-5 Programs and Features

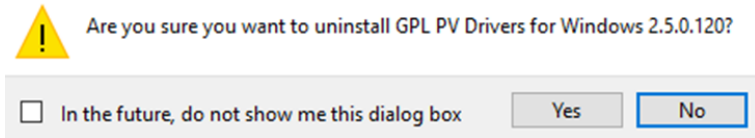


Step 4 In the program list, right-click the program to be uninstalled and choose **Uninstall** from the shortcut menu.

A confirmation dialog box is displayed.

Figure 2-6 Uninstallation confirmation

Programs and Features



Step 5 Click **Yes** to uninstall the program.

----End

3 RAID Array Creation with EVS Disks

3.1 Overview

Redundant Array of Independent Disks (RAID) is a technology that combines multiple physical disks into one or more logical units for the purposes of data redundancy and performance improvement.

NOTE

In this document, Elastic Volume Service (EVS) disks instead of physical disks are used to create RAID arrays. The working principles are the same.

This document uses CentOS 7.4 as the sample OS to describe how to create a RAID 10 array with four EVS disks. A RAID 10 array consists of RAID 0 and RAID 1 arrays. In this example, EVS disks are used to create a mirroring array (RAID 1) and then create a RAID 0 array to store data in stripes. At least four EVS disks are required. The resource information is as follows:

- Resource planning: [Resource Planning](#)
- Resource creation: [Creating an ECS](#) and [Creating and Attaching EVS Disks](#)

Introduction to Common RAID Arrays

Table 3-1 Introduction to common RAID arrays

RAID Level	Description	Read/Write Performance	Security	Disk Usage	Min. Number of Disks Required
RAID 0	RAID 0 stores data on multiple disks, implementing parallel read/write and providing the fastest read/write speed.	Parallel read/write from multiple disks achieves high performance.	Worst No redundancy capability. If one disk is damaged, the data of the entire RAID array is unavailable.	100%	2
RAID 1	RAID 1 implements data redundancy based on data mirroring. Half of the disk capacity in the RAID array is used, and the other half is used for mirroring to provide data backup.	Read performance: Same as a single disk Write performance: Data needs to be written into two disks. The write performance is lower than that of a single disk.	Highest Provides full backup of disk data. If a disk in the RAID array fails, the system automatically uses the data on the mirror disk.	50%	2
RAID 01	RAID 01 combines RAID 0 and RAID 1, in which half disks are first grouped into RAID 0 stripes and then used together with the other half to set up a RAID 1 array.	Read performance: Same as RAID 0 Write performance: Same as RAID 1	The security of RAID 01 is lower than that of RAID 10.	50%	4

RAID Level	Description	Read/Write Performance	Security	Disk Usage	Min. Number of Disks Required
RAID 10	RAID 10 combines RAID 1 and RAID 0, in which half disks are first set up as a RAID 1 array and then used together with the other half to create RAID 0 stripes.	Read performance: Same as RAID 0 Write performance: Same as RAID 1	The security performance of RAID 10 is the same as that of RAID 1.	50%	4
RAID 5	RAID 5 does not specify a dedicated parity disk and consists of block-level striping with parity information distributed among the disks.	Read performance: Same as RAID 0 Write performance: Because parity data needs to be written into disks, the write performance is lower than that of a single disk.	The security of RAID 5 is lower than that of RAID 10.	75%	3

3.2 Resource Planning

This topic describes the servers and disks planned for creating a RAID 10 array.

Servers

In this example, one Elastic Cloud Server (ECS) is created, and [Table 3-2](#) shows the parameter specifications.

Table 3-2 ECS parameter configurations

Parameter	Configuration Information
Name	ecs-raid10
Image	CentOS 7.4 64bit

Parameter	Configuration Information
Specifications	General computing, s2.medium.2 (1 vCPU, 2 GB memory)
Elastic IP Address (EIP)	139.XX.XX.XX
Private IP Address	192.168.1.189

EVS Disks

Setting up RAID 10 requires at least 4 disks. Therefore, 4 EVS disks are created and attached to the ECS in this example.

3.3 Creating Resources

3.3.1 Creating an ECS

Scenarios

This topic shows how to create an ECS. In this example, one ECS needs to be created. For details about the ECS parameter configurations, see [Resource Planning](#).

Procedure

- Step 1** Log in to the management console.
- Step 2** Under **Computing**, click **Elastic Cloud Server**.
The **Elastic Cloud Server** page is displayed.
- Step 3** Click **Buy ECS**.

For details about how to create an ECS, see the *Elastic Cloud Server User Guide*.

Set the following parameters as planned:

- **Image:** Select **CentOS 7.4 64bit**.
- **EIP:** An EIP is mandatory if the ECS needs to access the public network. In this example, the multiple devices admin (mdadm) tool needs to be installed. Therefore, an EIP must be configured. Buy an EIP or configure an existing one based on the environment condition.

[Figure 3-1](#) shows how to buy a new EIP.

Figure 3-1 Configuring EIP

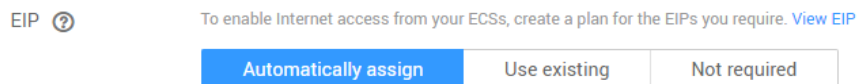


Table 3-3 shows the ECS parameter configurations.

Table 3-3 ECS parameter configurations

ECS Specifications		Billing Mode	Quantity
Specifications	General computing s2.medium. 2 1 vCPU 2 GB	Pay-per-use	1
Image	CentOS 7.4 64bit		
System disk	High I/O, 40 GB		
VPC	vpc-1a55		
Security group	Sys-default		
NIC	subnet-1a55(192.168.1.0/24)		
EIP	Specification: Static BGP Billing mode: By bandwidth (Bandwidth: 5 Mbit/s)		
ECS Name	ecs-raid10		

----End

3.3.2 Creating and Attaching EVS Disks

Scenarios

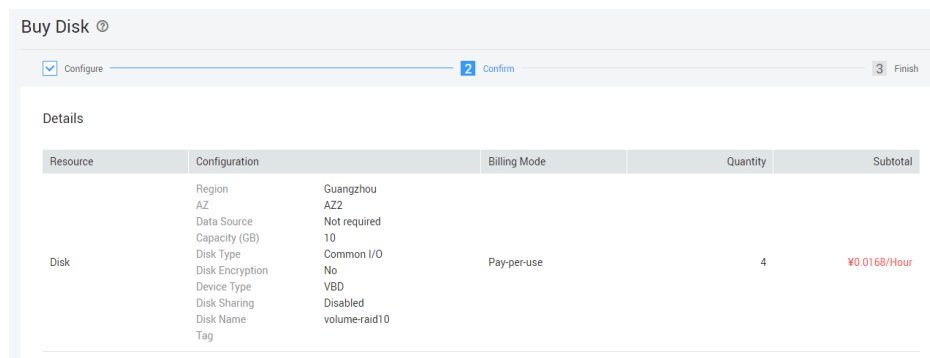
This topic shows how to create four EVS disks in a batch and attach the disks to the ECS.

Procedure

- Step 1** Log in to the management console.
- Step 2** Under **Storage**, click **Elastic Volume Service**.
The disk list page is displayed.
- Step 3** Click **Buy Disk** and create a disk.

For details about how to create EVS disks, see the *Elastic Volume Service User Guide*.

In this example, four EVS disks are created in a batch, and **Figure 3-2** shows the detailed parameter configurations.

Figure 3-2 EVS disk specifications

Resource	Configuration	Billing Mode	Quantity	Subtotal	
Disk	Region	Guangzhou			
	AZ	AZ2			
	Data Source	Not required			
	Capacity (GB)	10			
	Disk Type	Common I/O			
	Disk Encryption	No	Pay-per-use	4	¥0.0168/Hour
	Device Type	VBD			
	Disk Sharing	Disabled			
	Disk Name	volume-raid10			
	Tag				

Step 4 Attach the disks to the ECS.

----End

3.4 Creating a RAID Array Using mdadm

Scenarios

This topic shows how to create a RAID 10 array using mdadm.

In this example, the ECS runs CentOS 7.4 64bit. Configurations vary depending on the OS running on the ECS. This document is used for reference only. For the detailed operations and differences, see the corresponding OS documents.

Procedure

Step 1 Log in to the ECS as user **root**.

Step 2 Run the following command to view the disk and take note of the device name:

```
fdisk -l | grep /dev/vd | grep -v vda
```

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# fdisk -l | grep /dev/vd | grep -v vda
Disk /dev/vdb: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/vdc: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/vdd: 10.7 GB, 10737418240 bytes, 20971520 sectors
Disk /dev/vde: 10.7 GB, 10737418240 bytes, 20971520 sectors
```

In the command output, four disks are attached to the ECS, and the device names are **/dev/vdb**, **/dev/vdc**, **/dev/vdd**, and **/dev/vde**, respectively.

Step 3 Run the following command to install mdadm:

```
yum install mdadm -y
```

NOTE

mdadm is a Linux utility used to manage and monitor software RAID devices. Ensure that the EIP has been enabled for the ESC where mdadm is to be installed.

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# yum install mdadm -y
.....
Installed:
```

```
mdadm.x86_64 0:4.0-13.el7
Dependency Installed:
  libreport-filessystem.x86_64 0:2.1.11-40.el7.centos
Complete!
```

Step 4 Run the following command to create a RAID array using the four disks queried in [Step 2](#):

```
mdadm -Cv RAID array device name -a yes -n Disk quantity -l RAID level Device name of disk1 Device name of disk2 Device name of disk3 Device name of disk4
```

Parameter descriptions:

- **RAID array device name:** The value can be user-definable. In this example, **/dev/md0** is used.
- **Disk quantity:** Set this parameter based on the actual condition. In this example, RAID 10 is created, and at least four disks are required. The minimum number of disks required varies depending on the RAID level. For details, see [Overview](#).
- **RAID level:** Set this parameter based on the actual condition. In this example, set it to RAID 10.
- **Disk device name:** Enter the device names of all the disks that are used to create the RAID array. Multiple names are separated with spaces.

For example, run the following command:

```
mdadm -Cv /dev/md0 -a yes -n 4 -l 10 /dev/vdb /dev/vdc /dev/vdd /dev/vde
```

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# mdadm -Cv /dev/md0 -a yes -n 4 -l 10 /dev/vdb /dev/vdc /dev/vdd /dev/vde
mdadm: layout defaults to n2
mdadm: layout defaults to n2
mdadm: chunk size defaults to 512K
mdadm: size set to 10476544K
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
```

Step 5 Run the following command to format the created RAID array:

```
mkfs.File system format Device name of the RAID array
```

For example, run the following command:

```
mkfs.ext4 /dev/md0
```

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# mkfs.ext4 /dev/md0
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=128 blocks, Stripe width=256 blocks
1310720 inodes, 5238272 blocks
261913 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2153775104
160 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, 1605632, 2654208,
```

```
4096000
```

```
Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

Step 6 Run the following command to create a mounting directory:

```
mkdir Mounting directory
```

For example, run the following command:

```
mkdir /RAID10
```

Step 7 Run the following command to mount the RAID array:

```
mount RAID array device name Mounting directory
```

For example, run the following command:

```
mount /dev/md0 /RAID10
```

Step 8 Run the following command to view the mount result:

```
df -h
```

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/vda2        39G  1.5G  35G   5% /
devtmpfs         911M   0 911M   0% /dev
tmpfs            920M   0 920M   0% /dev/shm
tmpfs            920M  8.6M  911M   1% /run
tmpfs            920M   0 920M   0% /sys/fs/cgroup
/dev/vda1        976M  146M  764M  17% /boot
tmpfs            184M   0 184M   0% /run/user/0
/dev/md0         20G   45M  19G   1% /RAID10
```

Step 9 Perform the following operations to enable automatic mounting of the RAID array at the system start:

1. Run the following command to open the **/etc/fstab** file:

```
vi /etc/fstab
```

2. Press **i** to enter editing mode.

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# vi /etc/fstab

#
# /etc/fstab
# Created by anaconda on Tue Nov  7 14:28:26 2017
#
# 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
#
UUID=27f9be47-838b-4155-b20b-e4c5e013cdf3 / ext4 defaults 1 1
UUID=2b2000b1-f926-4b6b-ade8-695ee244a901 /boot ext4 defaults 1 2
```

3. Add the following information to the end of the file:

```
/dev/md0 /RAID10 ext4 defaults 0 0
```

4. Press **Esc**, enter **:wq!**, and press **Enter**.

The system saves the modifications and exits the vi editor.

Step 10 Run the following command to view the RAID array information:

mdadm -D *RAID array device name*

For example, run the following command:

mdadm -D /dev/md0

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# mdadm -D /dev/md0
/dev/md0:
  Version : 1.2
  Creation Time : Thu Nov  8 15:49:02 2018
  Raid Level : raid10
  Array Size : 20953088 (19.98 GiB 21.46 GB)
  Used Dev Size : 10476544 (9.99 GiB 10.73 GB)
  Raid Devices : 4
  Total Devices : 4
  Persistence : Superblock is persistent

  Update Time : Thu Nov  8 16:15:11 2018
  State : clean
  Active Devices : 4
  Working Devices : 4
  Failed Devices : 0
  Spare Devices : 0

  Layout : near=2
  Chunk Size : 512K

Consistency Policy : resync

  Name : ecs-raid10.novalocal:0 (local to host ecs-raid10.novalocal)
  UUID : f400dbf9:60d211d9:e006e07b:98f8758c
  Events : 19

Number Major Minor RaidDevice State
 0   253   16    0   active sync set-A /dev/vdb
 1   253   32    1   active sync set-B /dev/vdc
 2   253   48    2   active sync set-A /dev/vdd
 3   253   64    3   active sync set-B /dev/vde
```

----End

3.5 Configuring Automatic Start of the RAID Array at Server Startup

Scenarios

This topic shows how to add RAID array information, such as the device name and UUID to the mdadm configuration file. In this case, the RAID array can be started by querying information in the configuration file when the system starts.

In this example, the ECS runs CentOS 7.4 64bit. Configurations vary depending on the OS running on the ECS. This document is used for reference only. For the detailed operations and differences, see the corresponding OS documents.

Procedure

Step 1 Log in to the ECS as user **root**.

Step 2 Run the following command to view the RAID array information:

mdadm --detail --scan

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# mdadm --detail --scan  
ARRAY /dev/md0 metadata=1.2 name=ecs-raid10.novalocal:0 UUID=f400dbf9:60d211d9:e006e07b:98f8758c
```

Step 3 Perform the following operations to add information of the new RAID array to the mdadm file:

1. Run the following command to open the **mdadm.conf** file:

```
vi /etc/mdadm.conf
```

2. Press **i** to enter editing mode.
3. Add the following information to the end of the file:

```
DEVICE /dev/vdb /dev/vdc /dev/vdd /dev/vde  
ARRAY /dev/md0 metadata=1.2 name=ecs-raid10.novalocal:0 UUID=f400dbf9:60d211d9:e006e07b:  
98f8758c
```

Description:

- **DEVICE** line: Indicates the device names of the disks that set up the RAID array. Multiple disk device names are separated with spaces.
- **ARRAY** line: Indicates RAID array information. Input the RAID array information obtained in [Step 2](#).

 **NOTE**

The preceding information is used for reference only. Add RAID array information based on the site information.

4. Press **Esc**, enter **:wq!**, and press **Enter**.

The system saves the modifications and exits the vi editor.

Step 4 Run the following command to check whether the **mdadm.conf** file is modified:

```
more /etc/mdadm.conf
```

Information similar to the following is displayed:

```
[root@ecs-raid10 ~]# more /etc/mdadm.conf  
DEVICE /dev/vdb /dev/vdc /dev/vdd /dev/vde  
ARRAY /dev/md0 metadata=1.2 name=ecs-raid10.novalocal:0 UUID=f400dbf9:60d211d9:e006e07b:98f8758c
```

If the information added in [Step 3](#) is displayed, the file is successfully modified.

----End

4 Extending Disk Partitions and File Systems (Linux with a Kernel Earlier Than 3.6.0)

4.1 Partition and File System Extension Preparations (Linux)

Before extending the disk partition and file system, you must check the disk partition style and file system format, and then select the appropriate operation accordingly.

1. To view the disk partition style, see the following methods:
 - [Method 1: Check Partition Style and File System Format Using fdisk](#)
 - [Method 2: Check Partition Style and File System Format Using parted](#)
2. To select the appropriate operation, see [Table 4-1](#).

Table 4-1 Disk partition and file system extension scenarios

Disk	Scenario	Method
System disk	Create a new MBR partition with the additional space.	Creating a New MBR Partition
	Allocate the additional space to an existing MBR partition.	<ul style="list-style-type: none">• Extending an Existing MBR Partition (Kernel Version Later Than 3.6.0)• Extending an Existing MBR Partition (Kernel Version Earlier Than 3.6.0)
Data disk	Create a new MBR partition with the additional space.	Creating a New MBR Partition

Disk	Scenario	Method
	Allocate the additional space to an existing MBR partition.	Extending an Existing MBR Partition
	Create a new GPT partition with the additional space.	Creating a New GPT Partition
	Allocate the additional space to an existing GPT partition.	Extending an Existing GPT Partition
SCSI data disk	Create a new MBR partition with the additional space.	Creating a New MBR Partition
	Allocate the additional space to an existing MBR partition.	Extending an Existing MBR Partition

 **NOTE**

The maximum disk capacity that MBR supports is 2 TB, and the disk space exceeding 2 TB cannot be used.

If your disk uses MBR and you need to expand the disk capacity to over 2 TB, change the partition style from MBR to GPT. Ensure that the disk data has been backed up before changing the partition style because services will be interrupted and data on the disk will be cleared during this change.

Method 1: Check Partition Style and File System Format Using fdisk

Step 1 Run the following command to view all the disks attached to the server:

lsblk

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vda 253:0 0 40G 0 disk
└─vda1 253:1 0 40G 0 part /
vdb 253:16 0 150G 0 disk
└─vdb1 253:17 0 100G 0 part /mnt/sdc
```

In this example, data disk **/dev/vdb** already has partition **/dev/vdb1** before capacity expansion, and the additional 50 GB added has not been allocated yet. Therefore, **/dev/vdb** has 150 GB, and **/dev/vdb1** has 100 GB.

 **NOTE**

If you run the **lsblk** command and find out that disk **/dev/vdb** has no partition, the additional space may not be properly used after expansion. For the problem solution, see [How Do I Extend the File System of an Unpartitioned Data Disk in Linux?](#)

Step 2 Run the following command to view the current disk partition style:

fdisk -l

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# fdisk -l
Disk /dev/vda: 42.9 GB, 42949672960 bytes, 83886080 sectors
```

```
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000bcb4e

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1 *        2048     83886079   41942016   83  Linux

Disk /dev/vdb: 161.1 GB, 161061273600 bytes, 314572800 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x38717fc1

   Device Boot      Start         End      Blocks   Id  System
/dev/vdb1          2048     209715199  104856576   83  Linux
```

The value in the **System** column indicates the disk partition style. Value **Linux** indicates the MBR partition style. Value **GPT** indicates the GPT partition style.

- If the disk partitions displayed are inconsistent with those obtained in [Step 1](#), the partition that is not displayed uses the GPT partition style and has unallocated space. In this case, you cannot query all the partition information using the **fdisk -l** command. Go to [Method 2: Check Partition Style and File System Format Using parted](#).
- If the disk partitions displayed are consistent with those obtained in [Step 1](#), continue with the following operations.

Step 3 Run the following command to view the partition's file system format:

```
blkid Disk partition
```

In this example, run the following command:

```
blkid /dev/vdb1
```

In the command output, the **TYPE** value is **ext4**, indicating that **/dev/vdb1**'s file system format is **ext4**.

Step 4 Run the following command to view the file system status:

```
ext*: e2fsck -n Disk partition
```

```
xfs: xfs_repair -n Disk partition
```

In this example, the ext4 file system is used. Therefore, run the following command:

```
e2fsck -n /dev/vdb1
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# e2fsck -n /dev/vdb1
e2fsck 1.42.9 (28-Dec-2013)
Warning! /dev/vdb1 is mounted.
Warning: skipping journal recovery because doing a read-only filesystem check.
/dev/vdb1: clean, 11/6553600 files, 459544/26214144 blocks
```

If the file system status is **clean**, the file system status is normal. Otherwise, rectify the faulty and then perform the capacity expansion.

----End

Method 2: Check Partition Style and File System Format Using parted

Step 1 Run the following command to view all the disks attached to the server:

lsblk

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vda 253:0 0 40G 0 disk
└─vda1 253:1 0 40G 0 part /
vdb 253:16 0 150G 0 disk
└─vdb1 253:17 0 100G 0 part /mnt/sdc
```

In this example, data disk **/dev/vdb** already has partition **/dev/vdb1** before capacity expansion, and the additional 50 GB added has not been allocated yet. Therefore, **/dev/vdb** has 150 GB, and **/dev/vdb1** has 100 GB.

NOTE

If you run the **lsblk** command and find out that disk **/dev/vdb** has no partition, the additional space may not be properly used after expansion. For the problem solution, see [How Do I Extend the File System of an Unpartitioned Data Disk in Linux?](#)

Step 2 Run the following command and enter **p** to view the disk partition style:

parted Disk

For example, run the following command to view **/dev/vdb**'s partition style:

parted /dev/vdb

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# parted /dev/vdb
GNU Parted 3.1
Using /dev/vdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) p
Error: The backup GPT table is not at the end of the disk, as it should be. This might mean that another
operating system believes the
disk is smaller. Fix, by moving the backup to the end (and removing the old backup)?
Fix/Ignore/Cancel? Fix
Warning: Not all of the space available to /dev/vdb appears to be used, you can fix the GPT to use all of
the space (an extra 104857600
blocks) or continue with the current setting?
Fix/Ignore? Fix
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 161GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number Start End Size File system Name Flags
1 1049kB 107GB 107GB ext4 test

(parted)
```

In the command output, parameter **Partition Table** indicates the disk partition style. Value **msdos** indicates the MBR partition style, and value **gpt** indicates the GPT partition style.

- If the following error information is displayed, enter **Fix**.
Error: The backup GPT table is not at the end of the disk, as it should be. This might mean that another operating system believes the disk is smaller. Fix, by moving the backup to the end (and removing the old backup)?

The GPT partition table information is stored at the start of the disk. To reduce the risk of damages, a backup of the information is saved at the end of the disk. When you expand the disk capacity, the end of the disk changes accordingly. In this case, enter **Fix** to move the backup file of the information to new disk end.

- If the following warning information is displayed, enter **Fix**.

```
Warning: Not all of the space available to /dev/vdb appears to be used, you can fix the GPT to use all
of the space (an extra 104857600
blocks) or continue with the current setting?
Fix/Ignore? Fix
```

Enter **Fix** as prompted. The system automatically sets the GPT partition style for the additional space.

Step 3 Enter **q** and press **Enter** to exit parted.

```
----End
```

4.2 Extending Partitions and File Systems for System Disks (Linux)

Scenarios

After a disk has been expanded on the management console, only the disk storage capacity is enlarged, but its additional space cannot be used directly.

In Linux, you must allocate the additional space to an existing partition or a new partition.

If the disk capacity is expanded when its server is stopped, the additional space of a Linux system disk will be automatically added to the partition at the disk end upon the server startup. In this case, the additional space can be used directly.

This topic uses CentOS 7.4 64bit and CentOS 6.5 64bit as the sample OSs to describe how to extend the disk partition using growpart and fdisk. The method for allocating the additional space varies depending on the server OS. This document is used for reference only. For the detailed operations and differences, see the corresponding OS documents.

- [Extending an Existing MBR Partition \(Kernel Version Later Than 3.6.0\)](#)
- [Extending an Existing MBR Partition \(Kernel Version Earlier Than 3.6.0\)](#)
- [Creating a New MBR Partition](#)

For how to query the Linux kernel version, see [Querying the Linux Kernel Version](#).

NOTICE

Exercise caution when expanding the disk capacity. Misoperations may lead to data loss or exceptions. You are advised to back up the disk data using backups or snapshots before expansion. For details about backups, see [Managing EVS Backup](#). For details about snapshots, see [Creating a Snapshot \(OBT\)](#).

Prerequisites

- You have expanded the system disk capacity and attached the system disk to a server on the management console. For details, see [Expanding Capacity for an In-use EVS Disk](#) or [Expanding Capacity for an Available EVS Disk](#).
- You have logged in to the server.
 - For how to log in to an ECS, see [Logging In to an ECS](#).
 - For how to log in to a BMS, see [Logging In to a BMS](#).

Querying the Linux Kernel Version

Run the following command to query the Linux kernel version:

```
uname -a
```

Then, perform corresponding operations depending on whether the Linux kernel version is later than 3.6.0.

- For CentOS 7.4 64bit, information similar to the following is displayed:
Linux ecs-test-0001 3.10.0-957.5.1.el7.x86_64 #1 SMP Fri Feb 1 14:54:57 UTC 2019 x86_64 x86_64
The kernel version is 3.10.0, which is later than 3.6.0. For subsequent operations, see [Extending an Existing MBR Partition \(Kernel Version Later Than 3.6.0\)](#).
- For CentOS 6.5 64bit, information similar to the following is displayed:
Linux ecs-test-0002 2.6.32-754.10.1.el6.x86_64 #1 SMP Tue Jan 15 17:07:28 UTC 2019 x86_64
The kernel version is 2.6.32, which is earlier than 3.6.0. In this case, the disk partition and file system extension takes effect only after a server reboot. For subsequent operations, see [Extending an Existing MBR Partition \(Kernel Version Earlier Than 3.6.0\)](#).

Extending an Existing MBR Partition (Kernel Version Later Than 3.6.0)

CentOS 7.4 64bit is used as the sample OS. Originally, system disk `/dev/vda` has 40 GB and one partition (`/dev/vda1`), and then 60 GB is added to the disk. The following procedure shows you how to allocate the additional 60 GB to the existing MBR partition `/dev/vda1`.

Step 1 (Optional) Run the following command to install the growpart tool:

```
yum install cloud-utils-growpart
```

NOTE

You can run the `growpart` command to check whether the growpart tool has been installed. If the command output displays the tool usage instructions, the tool has been installed and you do not need to install it separately.

Step 2 Run the following command to view the total capacity of the `/dev/vda` system disk:

```
fdisk -l
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# fdisk -l  
  
Disk /dev/vda: 107.4 GB, 107374182400 bytes, 209715200 sectors  
Units = sectors of 1 * 512 = 512 bytes
```

```
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000bcb4e
```

Device	Boot	Start	End	Blocks	Id	System
/dev/vda1	*	2048	83886079	41942016	83	Linux

Step 3 Run the following command to view the capacity of the `/dev/vda1` partition:

df -TH

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1      ext4      43G   2.0G  39G   5% /
devtmpfs       devtmpfs  2.0G   0  2.0G   0% /dev
tmpfs          tmpfs     2.0G   0  2.0G   0% /dev/shm
tmpfs          tmpfs     2.0G   9.0M  2.0G   1% /run
tmpfs          tmpfs     2.0G   0  2.0G   0% /sys/fs/cgroup
tmpfs          tmpfs     398M   0  398M   0% /run/user/0
```

Step 4 Run the following command to extend the partition using `growpart`:

growpart *System disk Partition number*

In this example, run the following command:

growpart /dev/vda 1

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# growpart /dev/vda 1
CHANGED: partition=1 start=2048 old: size=83884032 end=83886080 new: size=209713119,end=209715167
```

Step 5 Run the following command to extend the file system of the partition:

resize2fs *Disk partition*

In this example, run the following command:

resize2fs /dev/vda1

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# resize2fs /dev/vda1
resize2fs 1.42.9 (28-Dec-2013)
Filesystem at /dev/vda1 is mounted on /; on-line resizing required
old_desc_blocks = 5, new_desc_blocks = 13
The filesystem on /dev/vda1 is now 26214139 blocks long.
```

Step 6 Run the following command to view the new capacity of the `/dev/vda1` partition:

df -TH

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1      ext4     106G   2.0G  99G   2% /
devtmpfs       devtmpfs  2.0G   0  2.0G   0% /dev
tmpfs          tmpfs     2.0G   0  2.0G   0% /dev/shm
tmpfs          tmpfs     2.0G   9.0M  2.0G   1% /run
tmpfs          tmpfs     2.0G   0  2.0G   0% /sys/fs/cgroup
tmpfs          tmpfs     398M   0  398M   0% /run/user/0
```

----End

Extending an Existing MBR Partition (Kernel Version Earlier Than 3.6.0)

NOTICE

If the OS kernel version is earlier than 3.6.0, the extension of an existing MBR disk partition and file system takes effect only after a server reboot, and services will be interrupted.

CentOS 6.5 64bit is used as the sample OS. Originally, system disk `/dev/vda` has 40 GB and one partition (`/dev/vda1`), and then 60 GB is added to the disk. The following procedure shows you how to allocate the additional 60 GB to the existing MBR partition `/dev/vda1`.

Step 1 (Optional) Run the following command to install the growpart tool:

```
yum install cloud-utils-growpart
```

NOTE

You can run the `growpart` command to check whether the growpart tool has been installed. If the command output displays the tool usage instructions, the tool has been installed and you do not need to install it separately.

Step 2 Run the following command to install the dracut-modules-growroot tool:

```
yum install cloud-utils-growpart
```

Information similar to the following is displayed:

```
[root@ecs-test-0002 ~]# yum install cloud-utils-growpart
Loaded plugins: fastestmirror, security
Setting up Install Process
Determining fastest mirrors
...
Package cloud-utils-growpart-0.27-10.el6.x86_64 already installed and latest version
Nothing to do
```

Step 3 Run the following command to regenerate the initramfs file:

```
dracut -f
```

Step 4 Run the following command to view the total capacity of the `/dev/vda` system disk:

```
fdisk -l
```

Information similar to the following is displayed:

```
[root@ecs-test-0002 ~]# fdisk -l

Disk /dev/vda: 107.4 GB, 107374182400 bytes
255 heads, 63 sectors/track, 13054 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: 0x0004e0be

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1 *           1         5222     41942016   83  Linux
```

Step 5 Run the following command to view the capacity of the `/dev/vda1` partition:

```
df -TH
```

Information similar to the following is displayed:

```
[root@ecs-test-0002 ~]# df -TH
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/vda1       ext4  43G  1.7G  39G   5% /
tmpfs           tmpfs 2.1G   0  2.1G   0% /dev/shm
```

Step 6 Run the following command to extend the partition using growpart:

```
growpart System disk Partition number
```

In this example, run the following command:

```
growpart /dev/vda 1
```

Information similar to the following is displayed:

```
[root@ecs-test-0002 ~]# growpart /dev/vda 1
CHANGED: partition=1 start=2048 old: size=83884032 end=83886080 new: size=209710462,end=209712510
```

Step 7 Run the following command to restart the server:

```
reboot
```

After the server is restarted, reconnect to the server and perform the following steps.

Step 8 Run the following command to extend the file system of the partition:

```
resize2fs Disk partition
```

In this example, run the following command:

```
resize2fs /dev/vda1
```

Information similar to the following is displayed:

```
[root@ecs-test-0002 ~]# resize2fs /dev/vda1
resize2fs 1.41.12 (17-May-2010)
The filesystem is already 26213807 blocks long. Nothing to do!
```

Step 9 Run the following command to view the new capacity of the **/dev/vda1** partition:

```
df -TH
```

Information similar to the following is displayed:

```
[root@ecs-test-0002 ~]# df -TH
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/vda1       ext4  106G  1.7G  99G   2% /
tmpfs           tmpfs 2.1G   0  2.1G   0% /dev/shm
```

```
----End
```

Creating a New MBR Partition

Originally, system disk **/dev/vda** has 40 GB and one partition (**/dev/vda1**), and then 40 GB is added to the disk. The following procedure shows you how to create a new MBR partition **/dev/vda2** with this 40 GB.

Step 1 Run the following command to view the disk partition information:

```
fdisk -l
```

Information similar to the following is displayed:

```
[root@ecs-2220 ~]# fdisk -l
Disk /dev/vda: 85.9 GB, 85899345920 bytes, 167772160 sectors
```

```
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x0008d18f

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1 *        2048     83886079     41942016   83  Linux
```

In the command output, the capacity of the **dev/vda** system disk is 80 GB, in which the in-use **dev/vda1** partition takes 40 GB and the additional 40 GB has not been allocated.

Step 2 Run the following command to enter fdisk:

```
fdisk /dev/vda
```

Information similar to the following is displayed:

```
[root@ecs-2220 ~]# fdisk /dev/vda
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help):
```

Step 3 Enter **n** and press **Enter** to create a new partition.

Information similar to the following is displayed:

```
Command (m for help): n
Partition type:
  p  primary (1 primary, 0 extended, 3 free)
  e  extended
```

There are two types of disk partitions:

- Choosing **p** creates a primary partition.
- Choosing **e** creates an extended partition.

NOTE

If the MBR partition style is used, a maximum of 4 primary partitions, or 3 primary partitions and 1 extended partition can be created. The extended partition cannot be used directly and must be divided into logical partitions before use.

Disk partitions created using GPT are not categorized.

Step 4 In this example, a primary partition is created. Therefore, enter **p** and press **Enter** to create a primary partition.

Information similar to the following is displayed:

```
Select (default p): p
Partition number (2-4, default 2):
```

Step 5 Partition number **2** is used in this example. Therefore, enter **2** and press **Enter**.

Information similar to the following is displayed:

```
Partition number (2-4, default 2): 2
First sector (83886080-167772159, default 83886080):
```

Step 6 Enter the new partition's start sector and press **Enter**. In this example, the default start sector is used.

The system displays the start and end sectors of the partition's available space. You can customize the value within this range or use the default value. The start sector must be smaller than the partition's end sector.

Information similar to the following is displayed:

```
First sector (83886080-167772159, default 83886080):
Using default value 83886080
Last sector, +sectors or +size{K,M,G} (83886080-167772159,default 167772159):
```

Step 7 Enter the new partition's end sector and press **Enter**. In this example, the default end sector is used.

The system displays the start and end sectors of the partition's available space. You can customize the value within this range or use the default value. The start sector must be smaller than the partition's end sector.

Information similar to the following is displayed:

```
Last sector, +sectors or +size{K,M,G} (83886080-167772159,
default 167772159):
Using default value 167772159
Partition 2 of type Linux and of size 40 GiB is set

Command (m for help):
```

Step 8 Enter **p** and press **Enter** to view the new partition.

Information similar to the following is displayed:

```
Command (m for help): p

Disk /dev/vda: 85.9 GB, 85899345920 bytes, 167772160 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x0008d18f

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1 *         2048     83886079     41942016   83  Linux
/dev/vda2          83886080    167772159     41943040   83  Linux
Command (m for help):
```

Step 9 Enter **w** and press **Enter** to write the changes to the partition table.

Information similar to the following is displayed:

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.

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

The partition is created.

NOTE

In case that you want to discard the changes made before, you can exit fdisk by entering **q**.

Step 10 Run the following command to synchronize the new partition table to the OS:

partprobe

Step 11 Run the following command to set the file system format for the new partition:

mkfs -t *File system Disk partition*

- Sample command of the ext* file system:
(The ext4 file system is used in this example.)

mkfs -t ext4 /dev/vda2

Information similar to the following is displayed:

```
[root@ecs-2220 ~]# mkfs -t ext4 /dev/vda2
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
2621440 inodes, 10485760 blocks
524288 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2157969408
320 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, 1605632, 2654208,
    4096000, 7962624

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

- Sample command of the xfs file system:

mkfs -t xfs /dev/vda2

Information similar to the following is displayed:

```
[root@ecs-2220 ~]# mkfs -t xfs /dev/vda2
meta-data=/dev/vda2          isize=512    agcount=4, agsize=2621440 blks
        =                   sectsz=512   attr=2,   projid32bit=1
        =                   crc=1      finobt=0, sparse=0
data     =                   bsize=4096   blocks=10485760, imaxpct=25
        =                   sunit=0    swidth=0 blks
naming   =version2          bsize=4096   ascii-ci=0  ftype=1
log      =internal log     bsize=4096   blocks=5120, version=2
        =                   sectsz=512   sunit=0 blks, lazy-count=1
realtime =none             extsz=4096   blocks=0,  rtextents=0
```

The formatting takes a while, and you need to observe the system running status. Once **done** is displayed in the command output, the formatting is complete.

Step 12 (Optional) Run the following command to create a mount point:

Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/opt** mount point:

mkdir /opt

Step 13 Run the following command to mount the new partition:

mount *Disk partition* *Mount point*

In this example, run the following command to mount the new partition **/dev/vda2** on **/opt**:

mount /dev/vda2 /opt

 NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

Step 14 Run the following command to view the mount result:

df -TH

Information similar to the following is displayed:

```
[root@ecs-2220 ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1       ext4      43G   2.0G  39G   5% /
devtmpfs        devtmpfs  509M    0 509M   0% /dev
tmpfs           tmpfs     520M    0 520M   0% /dev/shm
tmpfs           tmpfs     520M   7.2M  513M   2% /run
tmpfs           tmpfs     520M    0 520M   0% /sys/fs/cgroup
tmpfs           tmpfs     104M    0 104M   0% /run/user/0
/dev/vda2       ext4      43G   51M  40G   1% /opt
```

 NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the `/etc/fstab` file. For details, see [Setting Automatic Mounting at System Start](#).

----End

Setting Automatic Mounting at System Start

To automatically mount disk partitions at system start, do not specify partitions, for example `/dev/vdb1`, in `/etc/fstab` because the sequence of cloud devices, and therefore their names may change during the server stop and start. You are advised to use the universally unique identifier (UUID) in `/etc/fstab` to set automatic mounting at system start.

 NOTE

UUID is the unique character string for disk partitions in a Linux system.

Step 1 Run the following command to query the partition UUID:

blkid Disk partition

In this example, run the following command to query the UUID of the `/dev/vdb1` partition:

blkid /dev/vdb1

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# blkid /dev/vdb1
/dev/vdb1: UUID="0b3040e2-1367-4abb-841d-ddb0b92693df" TYPE="ext4"
```

The UUID of the `/dev/vdb1` partition is displayed.

Step 2 Run the following command to open the `fstab` file using the vi editor:

vi /etc/fstab

Step 3 Press **i** to enter the editing mode.

Step 4 Move the cursor to the end of the file and press **Enter**. Then, add the following information:

```
UUID=0b3040e2-1367-4abb-841d-ddb0b92693df /mnt/sdc      ext4  defaults  0 2
```

The preceding content is used for reference only. Add the information that is used in the environment. The parameters are described as follows:

- The first column indicates the partition UUID obtained in [Step 1](#).
- The second column indicates the directory on which the partition is mounted. You can query the mount point using the **df -TH** command.
- The third column indicates the file system format of the partition. You can query the file system format using the **df -TH** command.
- The fourth column indicates the partition mount option. Normally, this parameter is set to **defaults**.
- The fifth column indicates the Linux dump backup option.
 - **0**: not use Linux dump backup. Normally, dump backup is not used, and you can set this parameter to **0**.
 - **1**: use Linux dump backup.
- The sixth column indicates the fsck option, that is, whether to use fsck to check the attached disk during startup.
 - **0**: not use fsck.
 - If the mount point is the root partition (**/**), this parameter must be set to **1**.
When this parameter is set to **1** for the root partition, this parameter for other partitions must start with **2** so that the system checks the partitions in the ascending order of the values.

Step 5 Press **Esc**, enter **:wq**, and press **Enter**.

The system saves the configurations and exits the vi editor.

Step 6 Perform the following operations to verify the automatic mounting function:

1. Run the following command to unmount the partition:

```
umount Disk partition
```

In this example, run the following command:

```
umount /dev/vdb1
```

2. Run the following command to reload all the content in the **/etc/fstab** file:

```
mount -a
```

3. Run the following command to query the file system mounting information:

```
mount | grep Mount point
```

In this example, run the following command:

```
mount | grep /mnt/sdc
```

If information similar to the following is displayed, the automatic mounting function takes effect:

```
root@ecs-test-0001 ~]# mount | grep /mnt/sdc  
/dev/vdb1 on /mnt/sdc type ext4 (rw,relatime,data=ordered)
```

----End

4.3 Extending Partitions and File Systems for Data Disks (Linux)

Scenarios

After a disk has been expanded on the management console, only the disk storage capacity is enlarged, but its additional space cannot be used directly.

In Linux, you must allocate the additional space to an existing partition or a new partition.

This topic uses CentOS 7.4 64bit as the sample OS to describe how to extend an MBR or GPT partition. The method for allocating the additional space varies depending on the server OS. This document is used for reference only. For the detailed operations and differences, see the corresponding OS documents.

- [Creating a New MBR Partition](#)
- [Extending an Existing MBR Partition](#)
- [Creating a New GPT Partition](#)
- [Extending an Existing GPT Partition](#)

NOTICE

Exercise caution when expanding the disk capacity. Misoperations may lead to data loss or exceptions. You are advised to back up the disk data using backups or snapshots before expansion. For details about backups, see [Managing EVS Backup](#). For details about snapshots, see [Creating a Snapshot \(OBT\)](#).

Prerequisites

- You have expanded the system disk capacity and attached the system disk to a server on the management console. For details, see [Expanding Capacity for an In-use EVS Disk](#) or [Expanding Capacity for an Available EVS Disk](#).
- You have logged in to the server.
 - For how to log in to an ECS, see [Logging In to an ECS](#).
 - For how to log in to a BMS, see [Logging In to a BMS](#).

Creating a New MBR Partition

Originally, data disk **/dev/vdb** has 100 GB and one partition (**/dev/vdb1**), and then 50 GB is added to the disk. The following procedure shows you how to create a new MBR partition **/dev/vdb2** with this 50 GB.

Step 1 Run the following command to view the disk partition information:

```
fdisk -l
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# fdisk -l

Disk /dev/vda: 42.9 GB, 42949672960 bytes, 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000bcb4e

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1  *          2048     83886079     41942016   83  Linux

Disk /dev/vdb: 161.1 GB, 161061273600 bytes, 314572800 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x38717fc1

   Device Boot      Start         End      Blocks   Id  System
/dev/vdb1                2048    209715199    104856576   83  Linux
```

Step 2 Run the following command to enter fdisk:

```
fdisk Disk
```

In this example, run the following command:

```
fdisk /dev/vdb
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# fdisk /dev/vdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help):
```

Step 3 Enter **n** and press **Enter** to create a new partition.

Information similar to the following is displayed:

```
Command (m for help): n
Partition type:
  p   primary (1 primary, 0 extended, 3 free)
  e   extended
Select (default p):
```

There are two types of disk partitions:

- Choosing **p** creates a primary partition.
- Choosing **e** creates an extended partition.

NOTE

If the MBR partition style is used, a maximum of 4 primary partitions, or 3 primary partitions and 1 extended partition can be created. The extended partition cannot be used directly and must be divided into logical partitions before use.

Disk partitions created using GPT are not categorized.

Step 4 In this example, a primary partition is created. Therefore, enter **p** and press **Enter** to create a primary partition.

Information similar to the following is displayed:

```
Select (default p): p
Partition number (2-4, default 2):
```

Partition number indicates the serial number of the primary partition. Because partition number 1 has been used, the value ranges from **2** to **4**.

- Step 5** Enter the serial number of the primary partition and press **Enter**. Partition number **2** is used in this example. Therefore, enter **2** and press **Enter**.

Information similar to the following is displayed:

```
Partition number (2-4, default 2): 2
First sector (209715200-314572799, default 209715200):
```

First sector indicates the start sector. The value ranges from **209715200** to **314572799**, and the default value is **209715200**.

- Step 6** Enter the new partition's start sector and press **Enter**. In this example, the default start sector is used.

The system displays the start and end sectors of the partition's available space. You can customize the value within this range or use the default value. The start sector must be smaller than the partition's end sector.

Information similar to the following is displayed:

```
First sector (209715200-314572799, default 209715200):
Using default value 209715200
Last sector, +sectors or +size{K,M,G} (209715200-314572799, default 314572799):
```

Last sector indicates the end sector. The value ranges from **209715200** to **314572799**, and the default value is **314572799**.

- Step 7** Enter the new partition's end sector and press **Enter**. In this example, the default end sector is used.

The system displays the start and end sectors of the partition's available space. You can customize the value within this range or use the default value. The start sector must be smaller than the partition's end sector.

Information similar to the following is displayed:

```
Last sector, +sectors or +size{K,M,G} (209715200-314572799, default 314572799):
Using default value 314572799
Partition 2 of type Linux and of size 50 GiB is set
```

```
Command (m for help):
```

- Step 8** Enter **p** and press **Enter** to view the new partition.

Information similar to the following is displayed:

```
Command (m for help): p

Disk /dev/vdb: 161.1 GB, 161061273600 bytes, 314572800 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x38717fc1
```

Device	Boot	Start	End	Blocks	Id	System
/dev/vdb1		2048	209715199	104856576	83	Linux
/dev/vdb2		209715200	314572799	52428800	83	Linux

```
Command (m for help):
```

- Step 9** Enter **w** and press **Enter** to write the changes to the partition table.

Information similar to the following is displayed:

```
Command (m for help): w
The partition table has been altered!
```

```
Calling ioctl() to re-read partition table.
```

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

NOTE

In case that you want to discard the changes made before, you can exit fdisk by entering **q**.

Step 10 Run the following command to synchronize the new partition table to the OS:

```
partprobe
```

Step 11 Run the following command to set the file system format for the new partition:

```
mkfs -t File system Disk partition
```

- Sample command of the ext* file system:

```
mkfs -t ext4 /dev/vdb2
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# mkfs -t ext4 /dev/vdb2
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
3276800 inodes, 13107200 blocks
655360 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2162163712
400 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, 1605632, 2654208,
    4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

- Sample command of the xfs file system:

```
mkfs -t xfs /dev/vdb2
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# mkfs -t xfs /dev/vdb2
meta-data=/dev/vdb2          isize=512    agcount=4, agsize=3276800 blks
=                           sectsz=512   attr=2,    projid32bit=1
=                           crc=1      finobt=0, sparse=0
data      =                   bsize=4096 blocks=13107200, imaxpct=25
=                           sunit=0     swidth=0 blks
naming   =version2           bsize=4096  ascii-ci=0  ftype=1
log      =internal log      bsize=4096  blocks=6400, version=2
=                           sectsz=512   sunit=0    blks, lazy-count=1
realtime =none              extsz=4096  blocks=0,  rtextents=0
```

The formatting takes a while, and you need to observe the system running status. Once **done** is displayed in the command output, the formatting is complete.

Step 12 (Optional) Run the following command to create a mount point:

Perform this step if you want to mount the partition on a new mount point.

```
mkdir Mount point
```

In this example, run the following command to create the **/mnt/test** mount point:

```
mkdir /mnt/test
```

Step 13 Run the following command to mount the new partition:

```
mount Disk partition Mount point
```

In this example, run the following command to mount the new partition **/dev/vdb2** on **/mnt/test**:

```
mount /dev/vdb2 /mnt/test
```

NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

Step 14 Run the following command to view the mount result:

```
df -TH
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1       ext4      43G   1.9G   39G   5% /
devtmpfs        devtmpfs  2.0G   0   2.0G   0% /dev
tmpfs           tmpfs     2.0G   0   2.0G   0% /dev/shm
tmpfs           tmpfs     2.0G   9.1M   2.0G   1% /run
tmpfs           tmpfs     2.0G   0   2.0G   0% /sys/fs/cgroup
tmpfs           tmpfs     398M   0   398M   0% /run/user/0
/dev/vdb1       ext4     106G   63M  101G   1% /mnt/sdc
/dev/vdb2       ext4     53G   55M   50G   1% /mnt/test
```

NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the **/etc/fstab** file. For details, see [Setting Automatic Mounting at System Start](#).

----End

Extending an Existing MBR Partition

NOTICE

If the additional space is allocated to an existing partition, data on the disk will not be cleared but you must use **umount** to unmount the existing partition. In this case, services will be affected.

Originally, data disk **/dev/vdb** has 150 GB and two partitions (**/dev/vdb1** and **/dev/vdb2**, and then 80 GB is added to the disk. The following procedure shows you how to add this 80 GB to the existing MBR partition **/dev/vdb2**.

NOTICE

During an expansion, the additional space is added to the end of the disk. Therefore, if the disk has multiple partitions, the additional space can only be allocated to the partition at the disk end.

Step 1 Run the following command to view the disk partition information:

fdisk -l

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# fdisk -l
```

```
Disk /dev/vda: 42.9 GB, 42949672960 bytes, 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000bcb4e
```

```
Device Boot      Start         End      Blocks   Id  System
/dev/vda1  *          2048     83886079     41942016   83  Linux
```

```
Disk /dev/vdb: 247.0 GB, 246960619520 bytes, 482344960 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x38717fc1
```

```
Device Boot      Start         End      Blocks   Id  System
/dev/vdb1          2048    209715199    104856576   83  Linux
/dev/vdb2    209715200    314572799     52428800   83  Linux
```

In the command output, take note of the partition's start and end sectors. In this example, **/dev/vdb2**'s start sector is **209715200**, and its end sector is **314572799**.

View the **/dev/vdb** capacity and check whether the additional space is included.

- If the additional space is not included, refresh the capacity according to [Extending Partitions and File Systems for SCSI Disks \(Linux\)](#).
- If the additional space is included, take note of the start and end sectors of the target partition and then go to **Step 2**. These values will be used in the subsequent operations.

Step 2 Run the following command to unmount the partition:

umount *Disk partition*

In this example, run the following command:

umount /dev/vdb2

Step 3 Run the following command to enter fdisk:

fdisk *Disk*

In this example, run the following command:

fdisk /dev/vdb

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# fdisk /dev/vdb  
Welcome to fdisk (util-linux 2.23.2).
```

```
Changes will remain in memory only, until you decide to write them.  
Be careful before using the write command.
```

```
Command (m for help):
```

Step 4 Run the following command to delete the partition to be extended:

1. Enter **d** and press **Enter** to delete the partition.

Information similar to the following is displayed:

```
Command (m for help): d  
Partition number (1,2, default 2):
```

2. Enter the partition number and press **Enter** to delete the partition. In this example, enter **2**.

Information similar to the following is displayed:

```
Partition number (1,2, default 2): 2  
Partition 2 is deleted
```

```
Command (m for help):
```

NOTE

After deleting the partition, recreate the partition according to the following steps, and data on this disk will not be lost.

Step 5 Enter **n** and press **Enter** to create a new partition.

Information similar to the following is displayed:

```
Command (m for help): n  
Partition type:  
  p primary (1 primary, 0 extended, 3 free)  
  e extended  
Select (default p):
```

There are two types of disk partitions:

- Choosing **p** creates a primary partition.
- Choosing **e** creates an extended partition.

NOTE

If the MBR partition style is used, a maximum of 4 primary partitions, or 3 primary partitions and 1 extended partition can be created. The extended partition cannot be used directly and must be divided into logical partitions before use.

Disk partitions created using GPT are not categorized.

Step 6 Ensure that the entered partition type is the same as the partition had before. In this example, a primary partition is used. Therefore, enter **p** and press **Enter** to create a primary partition.

Information similar to the following is displayed:

```
Select (default p): p  
Partition number (2-4, default 2):
```

Partition number indicates the serial number of the primary partition.

Step 7 Ensure that entered partition number is the same as the partition had before. In this example, partition number **2** is used. Therefore, enter **2** and press **Enter**.

Information similar to the following is displayed:

```
Partition number (2-4, default 2): 2
First sector (209715200-482344959, default 209715200):
```

In the command output, **First sector** specifies the start sector.

 **NOTE**

Data will be lost if the following operations are performed:

- Select a start sector other than the partition had before.
- Select an end sector smaller than the partition had before.

Step 8 Ensure that the entered start sector is the same as the partition had before. In this example, start sector **209715200** is recorded in [Step 1](#). Therefore, enter **209715200** and press **Enter**.

Information similar to the following is displayed:

```
First sector (209715200-482344959, default 209715200):
Using default value 209715200
Last sector, +sectors or +size{K,M,G} (209715200-482344959, default 482344959):
```

In the command output, **Last sector** specifies the end sector.

Step 9 Ensure that the entered end sector is larger than or equal to the end sector recorded in [Step 1](#). In this example, the recorded end sector is **314572799**, and the default end sector is used. Therefore, enter **482344959** and press **Enter**.

Information similar to the following is displayed:

```
Using default value 209715200
Last sector, +sectors or +size{K,M,G} (209715200-482344959, default 482344959):
Using default value 482344959
Partition 2 of type Linux and of size 130 GiB is set

Command (m for help):
```

The partition is created.

Step 10 Enter **p** and press **Enter** to view the partition details.

Information similar to the following is displayed:

```
Command (m for help): p

Disk /dev/vdb: 247.0 GB, 246960619520 bytes, 482344960 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x38717fc1
```

```
   Device Boot      Start         End      Blocks   Id  System
/dev/vdb1            2048    209715199    104856576    83  Linux
/dev/vdb2    209715200    482344959    136314880    83  Linux
```

```
Command (m for help):
```

Step 11 Enter **w** and press **Enter** to write the changes to the partition table.

Information similar to the following is displayed:

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
```

```
WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table. The new table will be used at
```

the next reboot or after you run `partprobe(8)` or `kpartx(8)`
Syncing disks.

 NOTE

In case that you want to discard the changes made before, you can exit `fdisk` by entering `q`.

Step 12 Run the following command to synchronize the new partition table to the OS:

partprobe

Step 13 Perform the following operations based on the file system of the disk:

- For the **ext*** file system
 - a. Run the following command to check the correctness of the file system on the partition:

e2fsck -f *Disk partition*

In this example, run the following command:

e2fsck -f /dev/vdb2

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# e2fsck -f /dev/vdb2
e2fsck 1.42.9 (28-Dec-2013)
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/vdb2: 11/3276800 files (0.0% non-contiguous), 251790/13107200 blocks
```

- b. Run the following command to extend the file system of the partition:

resize2fs *Disk partition*

In this example, run the following command:

resize2fs /dev/vdb2

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# resize2fs /dev/vdb2
resize2fs 1.42.9 (28-Dec-2013)
Resizing the filesystem on /dev/vdb2 to 34078720 (4k) blocks.
The filesystem on /dev/vdb2 is now 34078720 blocks long.
```

- c. (Optional) Run the following command to create a mount point:
Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

- d. Run the following command to mount the partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the partition **/dev/vdb2** on **/mnt/test**:

mount /dev/vdb2 /mnt/test

 NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

- For the **xfs** file system
 - a. (Optional) Run the following command to create a mount point:
Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

- b. Run the following command to mount the partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the partition **/dev/vdb2** on **/mnt/test**:

mount /dev/vdb2 /mnt/test

 NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

- c. Run the following command to extend the file system of the partition:

sudo xfs_growfs *Disk partition*

In this example, run the following command:

sudo xfs_growfs /dev/vdb2

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# sudo xfs_growfs /dev/vdb2
meta-data=/dev/vdb2          isize=512    agcount=4, agsize=3276800 blks
=                               sectsz=512   attr=2,    projid32bit=1
=                               crc=1      finobt=0, spinodes=0
data     =                       bsize=4096 blocks=13107200, imaxpct=25
=                               sunit=0    swidth=0 blks
naming   =version2               bsize=4096 ascii-ci=0 ftype=1
log      =internal               bsize=4096 blocks=6400, version=2
=                               sectsz=512   sunit=0   blks, lazy-count=1
realtime =none                  extsz=4096  blocks=0,  rtextents=0
data blocks changed from 13107200 to 34078720.
```

Step 14 Run the following command to view the mount result:

df -TH

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# df -TH
Filesystem Type Size Used Avail Use% Mounted on
/dev/vda1 ext4 43G 1.9G 39G 5% /
```

```
devtmpfs    devtmpfs 2.0G  0 2.0G  0% /dev
tmpfs       tmpfs    2.0G  0 2.0G  0% /dev/shm
tmpfs       tmpfs    2.0G  9.1M 2.0G  1% /run
tmpfs       tmpfs    2.0G  0 2.0G  0% /sys/fs/cgroup
tmpfs       tmpfs    398M  0 398M  0% /run/user/0
/dev/vdb1   ext4    106G  63M 101G  1% /mnt/sdc
/dev/vdb2   ext4    138G  63M 131G  1% /mnt/test
```

NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the `/etc/fstab` file. For details, see [Setting Automatic Mounting at System Start](#).

----End

Creating a New GPT Partition

Originally, data disk `/dev/vdb` has 100 GB and one partition (`/dev/vdb1`), and then 50 GB is added to the disk. The following procedure shows you how to create a new GPT partition `/dev/vdb2` with this 50 GB.

Step 1 Run the following command to view the disk partition information:

lsblk

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# lsblk
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda   253:0   0  40G  0 disk
├─vda1 253:1   0  40G  0 part /
vdb   253:16  0 150G  0 disk
├─vdb1 253:17  0 100G  0 part /mnt/sdc
```

Step 2 Run the following command to enter parted:

parted Disk

In this example, run the following command:

parted /dev/vdb

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# parted /dev/vdb
GNU Parted 3.1
Using /dev/vdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted)
```

Step 3 Enter **unit s** and press **Enter** to set the measurement unit of the disk to sector.

Step 4 Enter **p** and press **Enter** to view the disk partition information.

Information similar to the following is displayed:

```
(parted) unit s
(parted) p
Error: The backup GPT table is not at the end of the disk, as it should be. This might mean that another
operating system believes the
disk is smaller. Fix, by moving the backup to the end (and removing the old backup)?
Fix/Ignore/Cancel? Fix
Warning: Not all of the space available to /dev/vdb appears to be used, you can fix the GPT to use all of
the space (an extra 104857600
blocks) or continue with the current setting?
Fix/Ignore? Fix
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 314572800s
```

```
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number Start End      Size      File system Name Flags
 1    2048s 209713151s 209711104s ext4      test

(parted)
```

In the command output, take note of the partition's end sector. In this example, the end sector of the **/dev/vdb1** partition is **209713151s**.

- If the following error information is displayed, enter **Fix**.
Error: The backup GPT table is not at the end of the disk, as it should be. This might mean that another operating system believes the disk is smaller. Fix, by moving the backup to the end (and removing the old backup)?
The GPT partition table information is stored at the start of the disk. To reduce the risk of damages, a backup of the information is saved at the end of the disk. When you expand the disk capacity, the end of the disk changes accordingly. In this case, enter **Fix** to move the backup file of the information to new disk end.
- If the following warning information is displayed, enter **Fix**.
Warning: Not all of the space available to /dev/vdb appears to be used, you can fix the GPT to use all of the space (an extra 104857600 blocks) or continue with the current setting?
Fix/Ignore? Fix
Enter **Fix** as prompted. The system automatically sets the GPT partition style for the additional space.

Step 5 Run the following command and press **Enter**:

```
mkpart Partition name Start sector End sector
```

In this example, run the following command:

```
mkpart data 209713152s 100%
```

In this example, the additional space is used to create a new partition. In [Step 4](#), the end sector of partition **dev/vdb1** is **209713151s**. Therefore, the start sector of the new partition **dev/vdb2** is set to **209713152s** and the end sector **100%**. This start and end sectors are for reference only. You can plan the number of partitions and partition size based on service requirements.

Information similar to the following is displayed:
(parted) mkpart data 209713152s 100%
(parted)

NOTE

The maximum sector can be obtained in either of the following ways:

- Query the disk's maximum end sector. For details, see [Step 2](#) to [Step 4](#).
- Enter **-1s** or **100%**, and the value displayed is the maximum end sector.

Step 6 Enter **p** and press **Enter** to view the new partition.

Information similar to the following is displayed:
(parted) p
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 314572800s
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

```
Number Start      End          Size         File system Name  Flags
 1    2048s    209713151s  209711104s  ext4      test
 2    209713152s  314570751s  104857600s                data
(parted)
```

Step 7 Enter **q** and press **Enter** to exit parted.

Information similar to the following is displayed:

```
(parted) q
Information: You may need to update /etc/fstab.
```

You can set automatic disk mounting by updating the **/etc/fstab** file. Before updating the file, set the file system format for the partition and mount the partition on the mount point.

Step 8 Run the following command to set the file system format for the new partition:

mkfs -t *File system Disk partition*

- Sample command of the ext* file system:

mkfs -t ext4 /dev/vdb2

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# mkfs -t ext4 /dev/vdb2
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
3276800 inodes, 13107200 blocks
655360 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2162163712
400 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, 1605632, 2654208,
    4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

- Sample command of the xfs file system:

mkfs -t xfs /dev/vdb2

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# mkfs -t xfs /dev/vdb2
meta-data=/dev/vdb2          isize=512    agcount=4, agsize=3276800 blks
       =                   sectsz=512   attr=2,    projid32bit=1
       =                   crc=1      finobt=0, sparse=0
data     =                   bsize=4096  blocks=13107200, imaxpct=25
       =                   sunit=0    swidth=0 blks
naming   =version2          bsize=4096  ascii-ci=0  ftype=1
log      =internal log     bsize=4096  blocks=6400, version=2
       =                   sectsz=512   sunit=0    blks, lazy-count=1
realtime =none             extsz=4096  blocks=0,  rtextents=0
```

The formatting takes a while, and you need to observe the system running status. Once **done** is displayed in the command output, the formatting is complete.

Step 9 (Optional) Run the following command to create a mount point:

Perform this step if you want to mount the partition on a new mount point.

```
mkdir Mount point
```

In this example, run the following command to create the **/mnt/test** mount point:

```
mkdir /mnt/test
```

Step 10 Run the following command to mount the partition:

```
mount Disk partition Mount point
```

In this example, run the following command to mount the partition **/dev/vdb2** on **/mnt/test**:

```
mount /dev/vdb2 /mnt/test
```

NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

Step 11 Run the following command to view the mount result:

```
df -TH
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1       ext4      43G   1.9G  39G   5% /
devtmpfs        devtmpfs  2.0G   0   2.0G   0% /dev
tmpfs           tmpfs     2.0G   0   2.0G   0% /dev/shm
tmpfs           tmpfs     2.0G   9.1M  2.0G   1% /run
tmpfs           tmpfs     2.0G   0   2.0G   0% /sys/fs/cgroup
tmpfs           tmpfs     398M   0   398M   0% /run/user/0
/dev/vdb1       ext4     106G   63M  101G   1% /mnt/sdc
/dev/vdb2       ext4     53G   55M   50G   1% /mnt/test
```

NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the **/etc/fstab** file. For details, see [Setting Automatic Mounting at System Start](#).

```
----End
```

Extending an Existing GPT Partition

NOTICE

If the additional space is allocated to an existing partition, data on the disk will not be cleared but you must use **umount** to unmount the existing partition. In this case, services will be affected.

Originally, data disk **/dev/vdb** has 150 GB and two partitions (**/dev/vdb1** and **/dev/vdb2**, and then 80 GB is added to the disk. The following procedure shows you how to add this 80 GB to the existing GPT partition **/dev/vdb2**.

During an expansion, the additional space is added to the end of the disk. Therefore, if the disk has multiple partitions, the additional space can only be allocated to the partition at the disk end.

Step 1 Run the following command to view the disk partition information:

lsblk

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# lsblk
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda  253:0  0  40G  0 disk
└─vda1 253:1  0  40G  0 part /
vdb  253:16  0 230G  0 disk
├─vdb1 253:17  0 100G  0 part /mnt/sdc
└─vdb2 253:18  0  50G  0 part /mnt/test
```

View the **/dev/vdb** capacity and check whether the additional space is included.

- If the additional space is not included, refresh the capacity according to [Extending Partitions and File Systems for SCSI Disks \(Linux\)](#).
- If the additional space is included, go to [Step 2](#).

Step 2 Run the following command to unmount the partition:

umount *Disk partition*

In this example, run the following command:

umount /dev/vdb2

Step 3 Run the following command to view the unmount result:

lsblk

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# lsblk
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda  253:0  0  40G  0 disk
└─vda1 253:1  0  40G  0 part /
vdb  253:16  0 230G  0 disk
├─vdb1 253:17  0 100G  0 part /mnt/sdc
└─vdb2 253:18  0  50G  0 part
```

Step 4 Run the following command to enter parted:

parted *Disk*

In this example, run the following command:

parted /dev/vdb

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# parted /dev/vdb
GNU Parted 3.1
Using /dev/vdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted)
```

Step 5 Enter **unit s** and press **Enter** to set the measurement unit of the disk to sector.

Step 6 Enter **p** and press **Enter** to view the current disk partition style.

Information similar to the following is displayed:

```
(parted) unit s
(parted) p
Error: The backup GPT table is not at the end of the disk, as it should be. This might mean that another
operating system believes the
disk is smaller. Fix, by moving the backup to the end (and removing the old backup)?
Fix/Ignore/Cancel? Fix
Warning: Not all of the space available to /dev/vdb appears to be used, you can fix the GPT to use all of
the space (an extra 167772160
blocks) or continue with the current setting?
Fix/Ignore? Fix
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 482344960s
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start      End          Size         File system  Name  Flags
 1     2048s    209713151s  209711104s  ext4        test
 2     209713152s 314570751s  104857600s  ext4        data

(parted)
```

Take note of the start and end sectors of the **/dev/vdb2** partition. These values will be used during the partition recreation. In this example, the partition's start sector is **209713152s**, and its end sector is **314570751s**.

- If the following error information is displayed, enter **Fix**.
Error: The backup GPT table is not at the end of the disk, as it should be. This might mean that another operating system believes the disk is smaller. Fix, by moving the backup to the end (and removing the old backup)?

The GPT partition table information is stored at the start of the disk. To reduce the risk of damages, a backup of the information is saved at the end of the disk. When you expand the disk capacity, the end of the disk changes accordingly. In this case, enter **Fix** to move the backup file of the information to new disk end.

- If the following warning information is displayed, enter **Fix**.
Warning: Not all of the space available to /dev/vdb appears to be used, you can fix the GPT to use all of the space (an extra 104857600 blocks) or continue with the current setting?
Fix/Ignore? Fix

Enter **Fix** as prompted. The system automatically sets the GPT partition style for the additional space.

Step 7 Enter **rm** and the partition number, and then press **Enter**. In this example, partition number **2** is used.

Information similar to the following is displayed:

```
(parted) rm
Partition number? 2
(parted)
```

Step 8 Run the following command to recreate the partition and press **Enter**:

mkpart *Partition name Start sector End sector*

In this example, run the following command:

mkpart data 209713152s 100%

- Ensure that the entered start sector is the same as the partition had before. In this example, start sector **209713152s** is recorded in **Step 6**. Therefore, enter **209713152s**.
- Ensure that the entered end sector is smaller than the partition had before. In this example, the end sector recorded in **Step 6** is **314570751s**, and all the additional space needs to be allocated to **dev/vdb2**. Therefore, enter **100%**.

Information similar to the following is displayed:

```
(parted) mkpart data 209713152s 100%
(parted)
```

NOTE

Data will be lost if the following operations are performed:

- Select a start sector other than the partition had before.
- Select an end sector smaller than the partition had before.

Step 9 Enter **p** and press **Enter** to view the partition information.

Information similar to the following is displayed:

```
(parted) p
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 482344960s
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start      End          Size         File system  Name  Flags
  1      2048s     209713151s  209711104s  ext4         test
  2      209713152s 482342911s  272629760s  ext4         data

(parted)
```

Step 10 Enter **q** and press **Enter** to exit parted.

Information similar to the following is displayed:

```
(parted) q
Information: You may need to update /etc/fstab.
```

You can set automatic disk mounting by updating the **/etc/fstab** file. Before updating the file, set the file system format for the partition and mount the partition on the mount point.

Step 11 Perform the following operations based on the file system of the disk:

- For the **ext*** file system
 - a. Run the following command to check the correctness of the file system on the partition:

```
e2fsck -f Disk partition
```

In this example, run the following command:

```
e2fsck -f /dev/vdb2
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# e2fsck -f /dev/vdb2
e2fsck 1.42.9 (28-Dec-2013)
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/vdb2: 11/3276800 files (0.0% non-contiguous), 251790/13107200 blocks
```

- b. Run the following command to extend the file system of the partition:

resize2fs *Disk partition*

In this example, run the following command:

resize2fs /dev/vdb2

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# resize2fs /dev/vdb2
resize2fs 1.42.9 (28-Dec-2013)
Resizing the filesystem on /dev/vdb2 to 34078720 (4k) blocks.
The filesystem on /dev/vdb2 is now 34078720 blocks long.
```

- c. (Optional) Run the following command to create a mount point:

Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

- d. Run the following command to mount the partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the partition **/dev/vdb2** on **/mnt/test**:

mount /dev/vdb2 /mnt/test

 NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

- For the **xfs** file system

- a. (Optional) Run the following command to create a mount point:

Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

- b. Run the following command to mount the partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the partition **/dev/vdb2** on **/mnt/test**:

mount /dev/vdb2 /mnt/test

 NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

- c. Run the following command to extend the file system of the partition:

```
sudo xfs_growfs Disk partition
```

In this example, run the following command:

```
sudo xfs_growfs /dev/vdb2
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# sudo xfs_growfs /dev/vdb2
meta-data=/dev/vdb2          isize=512    agcount=4, agsize=3276800 blks
=                               sectsz=512   attr=2,    projid32bit=1
=                               crc=1       finobt=0,  spinodes=0
data     =                       bsize=4096  blocks=13107200, imaxpct=25
=                               sunit=0     swidth=0   blks
naming   =version2              bsize=4096  ascii-ci=0  ftype=1
log      =internal              bsize=4096  blocks=6400, version=2
=                               sectsz=512   sunit=0    blks, lazy-count=1
realtime =none                  extsz=4096  blocks=0,   rtextents=0
data blocks changed from 13107200 to 34078720.
```

- Step 12** Run the following command to view the mount result:

```
df -TH
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# df -TH
Filesystem Type      Size  Used Avail Use% Mounted on
/dev/vda1  ext4     43G  1.9G  39G   5% /
devtmpfs  devtmpfs 2.0G   0  2.0G   0% /dev
tmpfs     tmpfs    2.0G   0  2.0G   0% /dev/shm
tmpfs     tmpfs    2.0G  9.1M  2.0G   1% /run
tmpfs     tmpfs    2.0G   0  2.0G   0% /sys/fs/cgroup
tmpfs     tmpfs    398M   0  398M   0% /run/user/0
/dev/vdb1  ext4    106G  63M  101G   1% /mnt/sdc
/dev/vdb2  ext4    138G  63M  131G   1% /mnt/test
```

 NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the `/etc/fstab` file. For details, see [Setting Automatic Mounting at System Start](#).

----End

Setting Automatic Mounting at System Start

To automatically mount disk partitions at system start, do not specify partitions, for example `/dev/vdb1`, in `/etc/fstab` because the sequence of cloud devices, and therefore their names may change during the server stop and start. You are advised to use the universally unique identifier (UUID) in `/etc/fstab` to set automatic mounting at system start.

 NOTE

UUID is the unique character string for disk partitions in a Linux system.

Step 1 Run the following command to query the partition UUID:

blkid *Disk partition*

In this example, run the following command to query the UUID of the **/dev/vdb1** partition:

blkid /dev/vdb1

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# blkid /dev/vdb1
/dev/vdb1: UUID="0b3040e2-1367-4abb-841d-ddb0b92693df" TYPE="ext4"
```

The UUID of the **/dev/vdb1** partition is displayed.

Step 2 Run the following command to open the **fstab** file using the vi editor:

vi /etc/fstab

Step 3 Press **i** to enter the editing mode.

Step 4 Move the cursor to the end of the file and press **Enter**. Then, add the following information:

```
UUID=0b3040e2-1367-4abb-841d-ddb0b92693df /mnt/sdc          ext4  defaults  0 2
```

The preceding content is used for reference only. Add the information that is used in the environment. The parameters are described as follows:

- The first column indicates the partition UUID obtained in [Step 1](#).
- The second column indicates the directory on which the partition is mounted. You can query the mount point using the **df -TH** command.
- The third column indicates the file system format of the partition. You can query the file system format using the **df -TH** command.
- The fourth column indicates the partition mount option. Normally, this parameter is set to **defaults**.
- The fifth column indicates the Linux dump backup option.
 - **0**: not use Linux dump backup. Normally, dump backup is not used, and you can set this parameter to **0**.
 - **1**: use Linux dump backup.
- The sixth column indicates the fsck option, that is, whether to use fsck to check the attached disk during startup.
 - **0**: not use fsck.
 - If the mount point is the root partition (**/**), this parameter must be set to **1**.

When this parameter is set to **1** for the root partition, this parameter for other partitions must start with **2** so that the system checks the partitions in the ascending order of the values.

Step 5 Press **Esc**, enter **:wq**, and press **Enter**.

The system saves the configurations and exits the vi editor.

Step 6 Perform the following operations to verify the automatic mounting function:

1. Run the following command to unmount the partition:

umount *Disk partition*

In this example, run the following command:

```
umount /dev/vdb1
```

2. Run the following command to reload all the content in the `/etc/fstab` file:

```
mount -a
```

3. Run the following command to query the file system mounting information:

```
mount | grep Mount point
```

In this example, run the following command:

```
mount | grep /mnt/sdc
```

If information similar to the following is displayed, the automatic mounting function takes effect:

```
root@ecs-test-0001 ~]# mount | grep /mnt/sdc  
/dev/vdb1 on /mnt/sdc type ext4 (rw,relatime,data=ordered)
```

----End

4.4 Extending Partitions and File Systems for SCSI Disks (Linux)

Scenarios

After a disk has been expanded on the management console, only the disk storage capacity is enlarged, but its additional space cannot be used directly.

In Linux, you must allocate the additional space to an existing partition or a new partition.

This topic uses CentOS 7.4 64bit as the sample OS to describe how to extend an MBR partition of a SCSI data disk. The method for allocating the additional space varies depending on the server OS. This document is used for reference only. For the detailed operations and differences, see the corresponding OS documents.

- [Creating a New MBR Partition](#)
- [Extending an Existing MBR Partition](#)

NOTICE

Exercise caution when expanding the disk capacity. Misoperations may lead to data loss or exceptions. You are advised to back up the disk data using backups or snapshots before expansion. For details about backups, see [Managing EVS Backup](#). For details about snapshots, see [Creating a Snapshot \(OBT\)](#).

Prerequisites

- You have expanded the system disk capacity and attached the system disk to a server on the management console. For details, see [Expanding Capacity for an In-use EVS Disk](#) or [Expanding Capacity for an Available EVS Disk](#).
- You have logged in to the server.
 - For how to log in to an ECS, see [Logging In to an ECS](#).

- For how to log in to a BMS, see [Logging In to a BMS](#).

Creating a New MBR Partition

Originally, data disk `/dev/sda` has 50 GB and one partition (`/dev/sda1`), and then 50 GB is added to the disk. The following procedure shows you how to create a new MBR partition `/dev/sda2` with this 50 GB.

Step 1 Run the following command to view the disk partition information:

fdisk -l

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# fdisk -l

Disk /dev/vda: 42.9 GB, 42949672960 bytes, 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000bcb4e

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1  *           2048     83886079     41942016   83  Linux

Disk /dev/sda: 107.4 GB, 107374182400 bytes, 209715200 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x915ffe6a

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1           2048    104857599     52427776   83  Linux
```

View the `/dev/sda` capacity and check whether the additional space is included.

- If the additional space is not included, refresh the capacity according to [Step 2](#).
- If the additional space is included, go to [Step 3](#).

Step 2 (Optional) Run the following command to update the capacity of the SCSI data disk:

1. Run the following command to update the disk capacity on the server:

```
echo 1 > /sys/class/scsi_device/%d:%d:%d:%d/device/rescan &
```

In the command, `%d:%d:%d:%d` indicates a folder in the `/sys/class/scsi_device/` directory and can be obtained using `ll /sys/class/scsi_device/`.

Information similar to the following is displayed: (`2:0:0:0` indicates the folder to be obtained.)

```
cs-xen-02:/sys/class/scsi_device # ll /sys/class/scsi_device/
total 0
lrwxrwxrwx 1 root root 0 Sep 26 11:37 2:0:0:0 -> ../../devices/xen/vscsi-2064/host2/target2:0:0/2:0:0:0/
scsi_device/2:0:0:0
```

In this example, run the following command:

```
echo 1 > /sys/class/scsi_device/2:0:0:0/device/rescan &
```

2. After the disk capacity is updated, run the following command to view the disk partition information again:

fdisk -l

If the additional space is included, go to [Step 3](#).

Step 3 Run the following command to enter fdisk:

```
fdisk Disk
```

In this example, run the following command:

```
fdisk /dev/sda
```

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# fdisk /dev/sda
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help):
```

Step 4 Enter **n** and press **Enter** to create a new partition.

Information similar to the following is displayed:

```
Command (m for help): n
Partition type:
  p primary (1 primary, 0 extended, 3 free)
  e extended
Select (default p):
```

There are two types of disk partitions:

- Choosing **p** creates a primary partition.
- Choosing **e** creates an extended partition.

NOTE

If the MBR partition style is used, a maximum of 4 primary partitions, or 3 primary partitions and 1 extended partition can be created. The extended partition cannot be used directly and must be divided into logical partitions before use.

Disk partitions created using GPT are not categorized.

Step 5 In this example, a primary partition is created. Therefore, enter **p** and press **Enter** to create a primary partition.

Information similar to the following is displayed:

```
Select (default p): p
Partition number (2-4, default 2):
```

Partition number indicates the serial number of the primary partition. Because partition number 1 has been used, the value ranges from **2** to **4**.

Step 6 Enter the serial number of the primary partition and press **Enter**. Partition number **2** is used in this example. Therefore, enter **2** and press **Enter**.

Information similar to the following is displayed:

```
Partition number (2-4, default 2): 2
First sector (104857600-209715199, default 104857600):
```

First sector indicates the start sector. The value ranges from **104857600** to **209715199**, and the default value is **104857600**.

Step 7 Enter the new partition's start sector and press **Enter**. In this example, the default start sector is used.

The system displays the start and end sectors of the partition's available space. You can customize the value within this range or use the default value. The start sector must be smaller than the partition's end sector.

Information similar to the following is displayed:

```
First sector (104857600-209715199, default 104857600):
Using default value 104857600
Last sector, +sectors or +size{K,M,G} (104857600-209715199, default 209715199):
```

Last sector indicates the end sector. The value ranges from **104857600** to **209715199**, and the default value is **209715199**.

Step 8 Enter the new partition's end sector and press **Enter**. In this example, the default end sector is used.

The system displays the start and end sectors of the partition's available space. You can customize the value within this range or use the default value. The start sector must be smaller than the partition's end sector.

Information similar to the following is displayed:

```
Last sector, +sectors or +size{K,M,G} (104857600-209715199, default 209715199):
Using default value 209715199
Partition 2 of type Linux and of size 50 GiB is set
```

Command (m for help):

Step 9 Enter **p** and press **Enter** to view the new partition.

Information similar to the following is displayed:

```
Command (m for help): p

Disk /dev/sda: 107.4 GB, 107374182400 bytes, 209715200 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x915ffe6a
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1		2048	104857599	52427776	83	Linux
/dev/sda2		104857600	209715199	52428800	83	Linux

Command (m for help):

Step 10 Enter **w** and press **Enter** to write the changes to the partition table.

Information similar to the following is displayed:

```
Command (m for help): w
The partition table has been altered!
```

```
Calling ioctl() to re-read partition table.
```

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

NOTE

In case that you want to discard the changes made before, you can exit `fdisk` by entering **q**.

Step 11 Run the following command to synchronize the new partition table to the OS:

partprobe

Step 12 Run the following command to set the file system format for the new partition:

mkfs -t *File system Disk partition*

- Sample command of the ext* file system:

mkfs -t ext4 /dev/sda2

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# mkfs -t ext4 /dev/sda2
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
3276800 inodes, 13107200 blocks
655360 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2162163712
400 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, 1605632, 2654208,
    4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

- Sample command of the xfs file system:

mkfs -t xfs /dev/sda2

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# mkfs -t xfs /dev/sda2
meta-data=/dev/sda2          isize=512    agcount=4, agsize=3276800 blks
       =                   sectsz=512   attr=2,    projid32bit=1
       =                   crc=1      finobt=0, sparse=0
data     =                   bsize=4096   blocks=13107200, imaxpct=25
       =                   sunit=0    swidth=0 blks
naming   =version2          bsize=4096   ascii-ci=0 ftype=1
log      =internal log     bsize=4096   blocks=6400, version=2
       =                   sectsz=512   sunit=0 blks, lazy-count=1
realtime =none             extsz=4096   blocks=0, rtextents=0
```

The formatting takes a while, and you need to observe the system running status. Once **done** is displayed in the command output, the formatting is complete.

Step 13 (Optional) Run the following command to create a mount point:

Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

Step 14 Run the following command to mount the new partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the new partition **/dev/sda2** on **/mnt/test**:

mount /dev/sda2 /mnt/test

 NOTE

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

Step 15 Run the following command to view the mount result:

df -TH

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1       ext4      43G  2.0G  39G   5% /
devtmpfs        devtmpfs  509M    0  509M   0% /dev
tmpfs           tmpfs     520M    0  520M   0% /dev/shm
tmpfs           tmpfs     520M  7.2M  513M   2% /run
tmpfs           tmpfs     520M    0  520M   0% /sys/fs/cgroup
tmpfs           tmpfs     104M    0  104M   0% /run/user/0
/dev/sda1       ext4      53G   55M   50G   1% /mnt/sdc
/dev/sda2       ext4      53G   55M   50G   1% /mnt/test
```

 NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the `/etc/fstab` file. For details, see [Setting Automatic Mounting at System Start](#).

----End

Extending an Existing MBR Partition

NOTICE

If the additional space is allocated to an existing partition, data on the disk will not be cleared but you must use **umount** to unmount the existing partition. In this case, services will be affected.

Originally, SCSI data disk `/dev/sda` has 100 GB and two partitions (`/dev/sda1` and `/dev/sda2`, and then 50 GB is added to the disk. The following procedure shows you how to add this 50 GB to the existing MBR partition `/dev/sda2`.

During an expansion, the additional space is added to the end of the disk. Therefore, if the disk has multiple partitions, the additional space can only be allocated to the partition at the disk end.

Step 1 Run the following command to view the disk partition information:

fdisk -l

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# fdisk -l
Disk /dev/vda: 42.9 GB, 42949672960 bytes, 83886080 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disk label type: dos
Disk identifier: 0x000bcb4e

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1  *            2048     83886079   41942016   83  Linux

Disk /dev/sda: 161.1 GB, 161061273600 bytes, 314572800 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x915ffe6a

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1            2048    104857599   52427776   83  Linux
/dev/sda2    104857600    209715199   52428800   83  Linux
```

In the command output, take note of the partition's start and end sectors. In this example, **/dev/sda2**'s start sector is **104857600**, and its end sector is **209715199**.

View the **/dev/sda** capacity and check whether the additional space is included.

- If the additional space is not included, refresh the capacity according to [Step 2](#).
- If the additional space is included, take note of the start and end sectors of the target partition and then go to [Step 3](#). These values will be used in the subsequent operations.

Step 2 (Optional) Run the following command to update the capacity of the SCSI data disk:

1. Run the following command to update the disk capacity on the server:

```
echo 1 > /sys/class/scsi_device/%d:%d:%d:%d/device/rescan &
```

In the command, **%d:%d:%d:%d** indicates a folder in the **/sys/class/scsi_device/** directory and can be obtained using **ll /sys/class/scsi_device/**.

Information similar to the following is displayed: (**2:0:0:0** indicates the folder to be obtained.)

```
cs-xen-02:/sys/class/scsi_device # ll /sys/class/scsi_device/
total 0
lrwxrwxrwx 1 root root 0 Sep 26 11:37 2:0:0:0 -> ../../devices/xen/vscsi-2064/host2/target2:0:0/2:0:0:0/scsi_device/2:0:0:0
```

In this example, run the following command:

```
echo 1 > /sys/class/scsi_device/2:0:0:0/device/rescan &
```

2. After the disk capacity is updated, run the following command to view the disk partition information again:

```
fdisk -l
```

If the additional space is included, take note of the start and end sectors of the target partition and then go to [Step 3](#). These values will be used in the subsequent operations.

Step 3 Run the following command to unmount the partition:

```
umount Disk partition
```

In this example, run the following command:

```
umount /dev/sda2
```

Step 4 Run the following command to enter fdisk:

fdisk Disk

In this example, run the following command:

fdisk /dev/sda

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# fdisk /dev/sda
Welcome to fdisk (util-linux 2.23.2).
```

```
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

```
Command (m for help):
```

Step 5 Run the following command to delete the partition to be extended:

1. Enter **d** and press **Enter** to delete the partition.

Information similar to the following is displayed:

```
Command (m for help): d
Partition number (1,2, default 2):
```

2. Enter the partition number and press **Enter** to delete the partition. In this example, enter **2**.

Information similar to the following is displayed:

```
Partition number (1,2, default 2): 2
Partition 2 is deleted
```

```
Command (m for help):
```

 **NOTE**

After deleting the partition, recreate the partition according to the following steps, and data on this disk will not be lost.

Step 6 Enter **n** and press **Enter** to create a new partition.

Information similar to the following is displayed:

```
Command (m for help): n
Partition type:
  p primary (1 primary, 0 extended, 3 free)
  e extended
Select (default p):
```

There are two types of disk partitions:

- Choosing **p** creates a primary partition.
- Choosing **e** creates an extended partition.

 **NOTE**

If the MBR partition style is used, a maximum of 4 primary partitions, or 3 primary partitions and 1 extended partition can be created. The extended partition cannot be used directly and must be divided into logical partitions before use.

Disk partitions created using GPT are not categorized.

Step 7 Ensure that the entered partition type is the same as the partition had before. In this example, a primary partition is used. Therefore, enter **p** and press **Enter** to create a primary partition.

Information similar to the following is displayed:

```
Select (default p): p
Partition number (2-4, default 2):
```

Partition number indicates the serial number of the primary partition.

Step 8 Ensure that entered partition number is the same as the partition had before. In this example, partition number **2** is used. Therefore, enter **2** and press **Enter**.

Information similar to the following is displayed:

```
Partition number (2-4, default 2): 2
First sector (104857600-314572799, default 104857600):
```

In the command output, **First sector** specifies the start sector.

 **NOTE**

Data will be lost if the following operations are performed:

- Select a start sector other than the partition had before.
- Select an end sector smaller than the partition had before.

Step 9 Ensure that the entered start sector is the same as the partition had before. In this example, start sector **104857600** is recorded in [Step 1](#) or [Step 2](#). Therefore, enter **104857600** and press **Enter**.

Information similar to the following is displayed:

```
First sector (104857600-314572799, default 104857600):
Using default value 104857600
Last sector, +sectors or +size{K,M,G} (104857600-314572799, default 314572799):
```

In the command output, **Last sector** specifies the end sector.

Step 10 Ensure that the entered end sector is larger than or equal to the end sector recorded in [Step 1](#) or [Step 2](#). In this example, the recorded end sector is **209715199**, and the default end sector is used. Therefore, enter **314572799** and press **Enter**.

Information similar to the following is displayed:

```
Last sector, +sectors or +size{K,M,G} (104857600-314572799, default 314572799):
Using default value 314572799
Partition 2 of type Linux and of size 100 GiB is set
```

```
Command (m for help):
```

The partition is created.

Step 11 Enter **p** and press **Enter** to view the partition details.

Information similar to the following is displayed:

```
Command (m for help): p

Disk /dev/sda: 161.1 GB, 161061273600 bytes, 314572800 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x915ffe6a

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1            2048    104857599    52427776   83  Linux
/dev/sda2    104857600    314572799    104857600   83  Linux

Command (m for help):
```

Step 12 Enter **w** and press **Enter** to write the changes to the partition table.

Information similar to the following is displayed: (The partition is successfully created.)

```
Command (m for help): w
The partition table has been altered.

Calling ioctl() to re-read partition table.

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

NOTE

In case that you want to discard the changes made before, you can exit fdisk by entering **q**.

Step 13 Run the following command to synchronize the new partition table to the OS:

partprobe

Step 14 Perform the following operations based on the file system of the disk:

- For the **ext*** file system
 - a. Run the following command to check the correctness of the file system on the partition:

e2fsck -f *Disk partition*

In this example, run the following command:

e2fsck -f /dev/sda2

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# e2fsck -f /dev/sda2
e2fsck 1.42.9 (28-Dec-2013)
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/sda2: 11/3276800 files (0.0% non-contiguous), 251790/13107200 blocks
```

- b. Run the following command to extend the file system of the partition:

resize2fs *Disk partition*

In this example, run the following command:

resize2fs /dev/sda2

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# resize2fs /dev/sda2
resize2fs 1.42.9 (28-Dec-2013)
Resizing the filesystem on /dev/sda2 to 26214400 (4k) blocks.
The filesystem on /dev/sda2 is now 26214400 blocks long.
```

- c. (Optional) Run the following command to create a mount point:
Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

- d. Run the following command to mount the partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the partition **/dev/sda2** on **/mnt/test**:

mount /dev/sda2 /mnt/test **NOTE**

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

- For the **xfs** file system
 - a. (Optional) Run the following command to create a mount point:
Perform this step if you want to mount the partition on a new mount point.

mkdir *Mount point*

In this example, run the following command to create the **/mnt/test** mount point:

mkdir /mnt/test

- b. Run the following command to mount the partition:

mount *Disk partition Mount point*

In this example, run the following command to mount the partition **/dev/sda2** on **/mnt/test**:

mount /dev/sda2 /mnt/test

 **NOTE**

If the new partition is mounted on a directory that is not empty, the subdirectories and files in the directory will be hidden. Therefore, you are advised to mount the new partition on an empty directory or a new directory. If the new partition must be mounted on a directory that is not empty, move the subdirectories and files in this directory to another directory temporarily. After the partition is successfully mounted, move the subdirectories and files back.

- c. Run the following command to extend the file system of the partition:

sudo xfs_growfs *Disk partition*

In this example, run the following command:

sudo xfs_growfs /dev/sda2

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# sudo xfs_growfs /dev/sda2
meta-data=/dev/sda2      isize=512    agcount=4, agsize=3276800 blks
             =           sectsz=512   attr=2,    projid32bit=1
             =           crc=1        finobt=0,  spinodes=0
data      =             bsize=4096   blocks=13107200, imaxpct=25
             =           sunit=0     swidth=0 blks
naming    =version2     bsize=4096  ascii-ci=0  ftype=1
log       =internal    bsize=4096  blocks=6400, version=2
             =           sectsz=512   sunit=0 blks, lazy-count=1
realtime  =none         extsz=4096  blocks=0,  rtextents=0
data blocks changed from 13107200 to 26214400df .
```

Step 15 Run the following command to view the mount result:

df -TH

Information similar to the following is displayed:

```
[root@ecs-scsi ~]# df -TH
Filesystem      Type      Size  Used Avail Use% Mounted on
/dev/vda1       ext4      43G   2.0G   39G   5% /
devtmpfs        devtmpfs  509M   0 509M   0% /dev
tmpfs           tmpfs     520M   0 520M   0% /dev/shm
tmpfs           tmpfs     520M   7.2M  513M   2% /run
tmpfs           tmpfs     520M   0 520M   0% /sys/fs/cgroup
tmpfs           tmpfs     104M   0 104M   0% /run/user/0
/dev/sda1       ext4      53G   55M   50G   1% /mnt/sdc
/dev/sda2       ext4     106G   63M  101G   1% /mnt/test
```

NOTE

If the server is restarted, the mounting will become invalid. You can set automatic mounting for partitions at system start by modifying the `/etc/fstab` file. For details, see [Setting Automatic Mounting at System Start](#).

----End

Setting Automatic Mounting at System Start

To automatically mount disk partitions at system start, do not specify partitions, for example `/dev/vdb1`, in `/etc/fstab` because the sequence of cloud devices, and therefore their names may change during the server stop and start. You are advised to use the universally unique identifier (UUID) in `/etc/fstab` to set automatic mounting at system start.

NOTE

UUID is the unique character string for disk partitions in a Linux system.

Step 1 Run the following command to query the partition UUID:

```
blkid Disk partition
```

In this example, run the following command to query the UUID of the `/dev/vdb1` partition:

```
blkid /dev/vdb1
```

Information similar to the following is displayed:

```
[root@ecs-test-0001 ~]# blkid /dev/vdb1
/dev/vdb1: UUID="0b3040e2-1367-4abb-841d-ddb0b92693df" TYPE="ext4"
```

The UUID of the `/dev/vdb1` partition is displayed.

Step 2 Run the following command to open the `fstab` file using the vi editor:

```
vi /etc/fstab
```

Step 3 Press `i` to enter the editing mode.

Step 4 Move the cursor to the end of the file and press `Enter`. Then, add the following information:

```
UUID=0b3040e2-1367-4abb-841d-ddb0b92693df /mnt/sdc          ext4  defaults  0 2
```

The preceding content is used for reference only. Add the information that is used in the environment. The parameters are described as follows:

- The first column indicates the partition UUID obtained in [Step 1](#).
- The second column indicates the directory on which the partition is mounted. You can query the mount point using the `df -TH` command.

- The third column indicates the file system format of the partition. You can query the file system format using the **df -TH** command.
- The fourth column indicates the partition mount option. Normally, this parameter is set to **defaults**.
- The fifth column indicates the Linux dump backup option.
 - **0**: not use Linux dump backup. Normally, dump backup is not used, and you can set this parameter to **0**.
 - **1**: use Linux dump backup.
- The sixth column indicates the fsck option, that is, whether to use fsck to check the attached disk during startup.
 - **0**: not use fsck.
 - If the mount point is the root partition (**/**), this parameter must be set to **1**.
When this parameter is set to **1** for the root partition, this parameter for other partitions must start with **2** so that the system checks the partitions in the ascending order of the values.

Step 5 Press **Esc**, enter **:wq**, and press **Enter**.

The system saves the configurations and exits the vi editor.

Step 6 Perform the following operations to verify the automatic mounting function:

1. Run the following command to unmount the partition:

```
umount Disk partition
```

In this example, run the following command:

```
umount /dev/vdb1
```

2. Run the following command to reload all the content in the **/etc/fstab** file:

```
mount -a
```

3. Run the following command to query the file system mounting information:

```
mount | grep Mount point
```

In this example, run the following command:

```
mount | grep /mnt/sdc
```

If information similar to the following is displayed, the automatic mounting function takes effect:

```
root@ecs-test-0001 ~]# mount | grep /mnt/sdc  
/dev/vdb1 on /mnt/sdc type ext4 (rw,relatime,data=ordered)
```

----End

5 FAQs

5.1 How Can I Obtain ECS NIC Information?

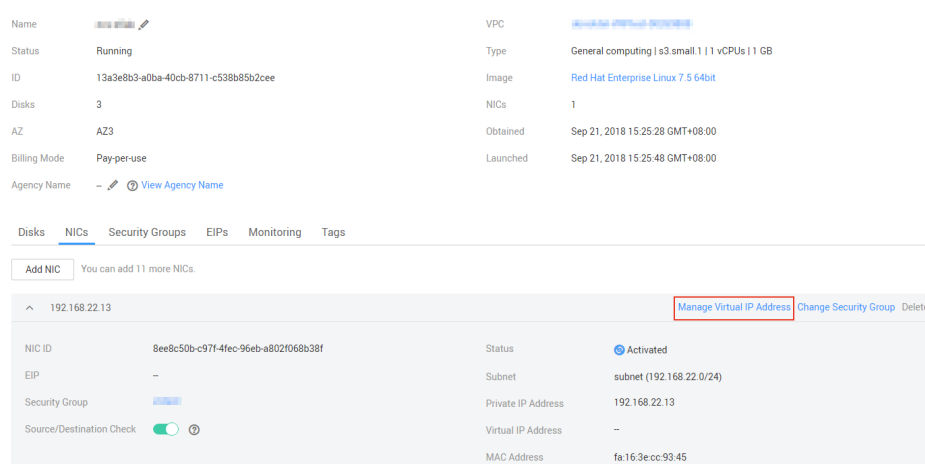
Scenarios

This topic shows how to obtain the ECS NIC information, such as the ECS IP address.

Procedure

- Step 1** Visit the HUAWEI CLOUD website and log in to the management console.
- Step 2** Under **Computing**, click **Elastic Cloud Server**.
The **Elastic Cloud Server** page is displayed.
- Step 3** In the ECS list, click the target ECS name.
The ECS details page is displayed.
- Step 4** Click the **NICs** tab.
The NIC details page is displayed.

Figure 5-1 NIC details page

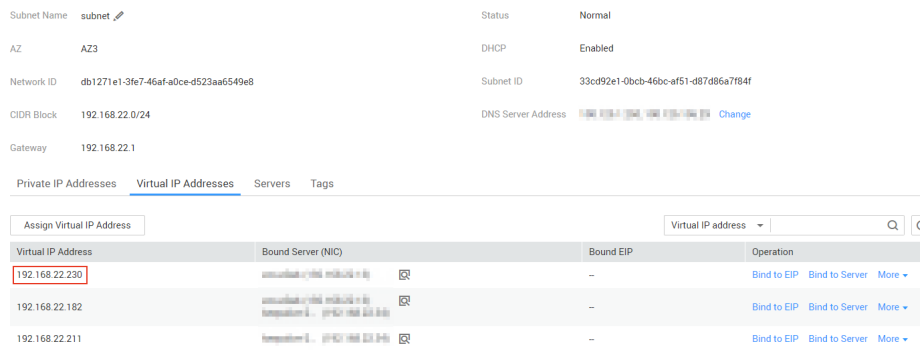



Step 5 On the **NICs** tab, click  to view ECS NIC information.

Step 6 To view the virtual IP address, click **Manage Virtual IP Address**.

The virtual IP address details page is displayed. You can find the bound virtual IP address based on the ECS NIC information.












Figure 5-2 Virtual IP address details page



Subnet Name: subnet  Status: Normal

AZ: AZ3 DHCP: Enabled













Network ID: db1271e1-3fe7-46af-a0ce-d523aa6549e8 Subnet ID: 33cd92e1-0bc6-46bc-ef51-d87d86a7f84f

CIDR Block: 192.168.22.0/24 DNS Server Address:            [Change](#)

Gateway: 192.168.22.1

Private IP Addresses **Virtual IP Addresses** Servers Tags

Assign Virtual IP Address Virtual IP address

Virtual IP Address	Bound Server (NIC)	Bound EIP	Operation
192.168.22.230	  	--	Bind to EIP Bind to Server More 
192.168.22.182	  	--	Bind to EIP Bind to Server More 
192.168.22.211	  	--	Bind to EIP Bind to Server More 

----End