Extend a Linux file system after resizing a volume

PDF (ec2-ug.pdf#recognize-expanded-volume-linux) | RSS (amazon-ec2-release-notes.rss)

After you increase the size of an EBS volume (./requesting-ebs-volume-modifications.html) , you must use file system–specific commands to extend the file system to the larger size. You can resize the file system as soon as the volume enters the `optimizing` state.

> ⚠ **Important**
>
> Before extending a file system that contains valuable data, it is best practice to create a snapshot of the volume, in case you need to roll back your changes. For more information, see Create Amazon EBS snapshots (./ebs-creating-snapshot.html) . If your Linux AMI uses the MBR partitioning scheme, you are limited to a boot volume size of up to 2 TiB. For more information, see Requirements for Linux volumes (./modify-volume-requirements.html#linux-volumes) and Constraints on the size and configuration of an EBS volume (./volume_constraints.html) .

The process for extending a file system on Linux is as follows:

1. Your EBS volume might have a partition that contains the file system and data. Increasing the size of a volume does not increase the size of the partition. Before you extend the file system on a resized volume, check whether the volume has a partition that must be extended to the new size of the volume.
2. Use a file system-specific command to resize each file system to the new volume capacity.

For information about extending a Windows file system, see Extend a Windows file system after resizing a volume (https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/recognize-expanded-volume-windows.html) in the *Amazon EC2 User Guide for Windows Instances*.

The following examples walk you through the process of extending a Linux file system. For file systems and partitioning schemes other than the ones shown here, refer to the documentation for those file systems and partitioning schemes for instructions.

> ⓘ **Note**
>
> If you are using logical volumes on the Amazon EBS volume, you must use Logical Volume Manager (LVM) to extend the logical volume. For instructions on how to do this, see the **Extend the logical volume** section in the How do I create an LVM logical

volume on an entire EBS volume? (http://aws.amazon.com/premiumsupport/knowledge-center/create-lv-on-ebs-volume/) AWS Knowledge Center article.

Examples

- Example: Extend the file system of NVMe EBS volumes (#extend-file-system-nvme)
- Example: Extend the file system of EBS volumes (#extend-file-system)

Example: Extend the file system of NVMe EBS volumes

For this example, suppose that you have an instance built on the Nitro System (./instance-types.html#ec2-nitro-instances) , such as an M5 instance. You resized the boot volume from 8 GB to 16 GB and an additional volume from 8 GB to 30 GB. Use the following procedure to extend the file system of the resized volumes.

To extend the file system of NVMe EBS volumes

1. Connect to your instance (./AccessingInstances.html) .
2. To verify the file system and type for each volume, use the **df -hT** command.

```
[ec2-user ~]$ df -hT
```

The following is example output for an instance that has a boot volume with an XFS file system and an additional volume with an XFS file system. The naming convention `/dev/nvme[0-26]n1` indicates that the volumes are exposed as NVMe block devices.

```
[ec2-user ~]$ df -hT
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/nvme0n1p1  xfs   8.0G  1.6G  6.5G  20% /
/dev/nvme1n1    xfs   8.0G   33M  8.0G   1% /data
...
```

3. To check whether the volume has a partition that must be extended, use the **lsblk** command to display information about the NVMe block devices attached to your instance.

```
[ec2-user ~]$ lsblk
NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
nvme1n1       259:0    0  30G  0 disk /data
nvme0n1       259:1    0  16G  0 disk
```

```
└─nvme0n1p1   259:2    0   8G  0 part /
└─nvme0n1p128 259:3    0   1M  0 part
```

This example output shows the following:

- The root volume, `/dev/nvme0n1`, has a partition, `/dev/nvme0n1p1`. While the size of the root volume reflects the new size, 16 GB, the size of the partition reflects the original size, 8 GB, and must be extended before you can extend the file system.
- The volume `/dev/nvme1n1` has no partitions. The size of the volume reflects the new size, 30 GB.

4. For volumes that have a partition, such as the root volume shown in the previous step, use the **growpart** command to extend the partition. Notice that there is a space between the device name and the partition number.

```
[ec2-user ~]$ sudo growpart /dev/nvme0n1 1
```

5. (Optional) To verify that the partition reflects the increased volume size, use the **lsblk** command again.

```
[ec2-user ~]$ lsblk
NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
nvme1n1       259:0    0  30G  0 disk /data
nvme0n1       259:1    0  16G  0 disk
└─nvme0n1p1   259:2    0  16G  0 part /
└─nvme0n1p128 259:3    0   1M  0 part
```

6. To verify the size of the file system for each volume, use the **df -h** command. In this example output, both file systems reflect the original volume size, 8 GB.

```
[ec2-user ~]$ df -h
Filesystem       Size  Used Avail Use% Mounted on
/dev/nvme0n1p1   8.0G  1.6G  6.5G  20% /
/dev/nvme1n1     8.0G   33M  8.0G   1% /data
...
```

7. To extend the file system on each volume, use the correct command for your file system, as follows:

   - [XFS file system] To extend the file system on each volume, use the **xfs_growfs** command. In this example, `/` and `/data` are the volume mount points shown in the output for **df -h**.

```
[ec2-user ~]$ sudo xfs_growfs -d /
[ec2-user ~]$ sudo xfs_growfs -d /data
```

If the XFS tools are not already installed, you can install them as follows.

```
[ec2-user ~]$ sudo yum install xfsprogs
```

- [ext4 file system] To extend the file system on each volume, use the **resize2fs** command.

```
[ec2-user ~]$ sudo resize2fs /dev/nvme0n1p1
[ec2-user ~]$ sudo resize2fs /dev/nvme1n1
```

- [Other file system] To extend the file system on each volume, refer to the documentation for your file system for instructions.

8. (Optional) To verify that each file system reflects the increased volume size, use the **df -h** command again.

```
[ec2-user ~]$ df -h
Filesystem        Size  Used Avail Use% Mounted on
/dev/nvme0n1p1     16G  1.6G   15G  10% /
/dev/nvme1n1       30G   33M   30G   1% /data
...
```

Example: Extend the file system of EBS volumes

For this example, suppose that you have resized the boot volume of an instance, such as a T2 instance, from 8 GB to 16 GB and an additional volume from 8 GB to 30 GB. Use the following procedure to extend the file system of the resized volumes.

To extend the file system of EBS volumes

1. Connect to your instance (./AccessingInstances.html) .
2. To verify the file system in use for each volume, use the **df -hT** command.

```
[ec2-user ~]$ df -hT
```

The following is example output for an instance that has a boot volume with an ext4 file system and an additional volume with an XFS file system.

```
[ec2-user ~]$ df -hT
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/xvda1      ext4  8.0G  1.9G  6.2G  24% /
/dev/xvdf1      xfs   8.0G   45M  8.0G   1% /data
...
```

3. To check whether the volume has a partition that must be extended, use the **lsblk** command to display information about the block devices attached to your instance.

```
[ec2-user ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  16G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdf    202:80   0  30G  0 disk
└─xvdf1 202:81   0   8G  0 part /data
```

This example output shows the following:

- The root volume, `/dev/xvda`, has a partition, `/dev/xvda1`. While the size of the volume is 16 GB, the size of the partition is still 8 GB and must be extended.
- The volume `/dev/xvdf` has a partition, `/dev/xvdf1`. While the size of the volume is 30G, the size of the partition is still 8 GB and must be extended.

4. For volumes that have a partition, such as the volumes shown in the previous step, use the **growpart** command to extend the partition. Notice that there is a space between the device name and the partition number.

```
[ec2-user ~]$ sudo growpart /dev/xvda 1
[ec2-user ~]$ sudo growpart /dev/xvdf 1
```

5. (Optional) To verify that the partitions reflect the increased volume size, use the **lsblk** command again.

```
[ec2-user ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  16G  0 disk
└─xvda1 202:1    0  16G  0 part /
```

```
xvdf    202:80   0  30G  0 disk
└─xvdf1 202:81   0  30G  0 part /data
```

6. To verify the size of the file system for each volume, use the **df -h** command. In this example output, both file systems reflect the original volume size, 8 GB.

```
[ec2-user ~]$ df -h
Filesystem       Size  Used Avail Use% Mounted on
/dev/xvda1       8.0G  1.9G  6.2G  24% /
/dev/xvdf1       8.0G   45M  8.0G   1% /data
...
```

7. To extend the file system on each volume, use the correct command for your file system, as follows:

   - [XFS volumes] To extend the file system on each volume, use the **xfs_growfs** command. In this example, `/` and `/data` are the volume mount points shown in the output for **df -h**.

     ```
     [ec2-user ~]$ sudo xfs_growfs -d /
     [ec2-user ~]$ sudo xfs_growfs -d /data
     ```

     If the XFS tools are not already installed, you can install them as follows.

     ```
     [ec2-user ~]$ sudo yum install xfsprogs
     ```

   - [ext4 volumes] To extend the file system on each volume, use the **resize2fs** command.

     ```
     [ec2-user ~]$ sudo resize2fs /dev/xvda1
     [ec2-user ~]$ sudo resize2fs /dev/xvdf1
     ```

   - [Other file system] To extend the file system on each volume, refer to the documentation for your file system for instructions.

8. (Optional) To verify that each file system reflects the increased volume size, use the **df -h** command again.

```
[ec2-user ~]$ df -h
Filesystem       Size  Used Avail Use% Mounted on
/dev/xvda1        16G  1.9G  14G  12% /
```

```
/dev/xvdf1        30G   45M  30G   1% /data
...
```