**Extend AWS EBS Volume**

**Info:** This can be applied whenever you need to extend your EBS volume size. It can be also perform with no downtime to avoid/stop the instance and detach the volume. Cool if you need to do it on prod.

**Author:** Rajeev Ranjan

**Date: 16**-July-2022, revised \_\_\_\_\_\_\_\_\_

**Introduction**

This document explains how a disk size can be increased of aws instance. The instructions are aimed at any competent System Administrator.

**Server Details**

For example we are showing the steps which had taken at the time of our epm-prod-db server size extend. A server details has summerized below :

|  |  |
| --- | --- |
| CPU | Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz, 8-Core |
| RAM | 32 GB |
| DISK Size /DB | 1 TB |
| OS | 4.14.55-68.37.amzn2.x86\_64 |
| IP | 13.232.82.175 |

Here we are in need to add more 200 GB space in our /DB partition which is for database.

**In order to extend the volume size, follow these simple steps:**

**Step-1:** Login to your aws console and go to ec2 service**.**

**Step-2:** From the left pane, Click on the snapshot to take the backup of that volume first for security purpose.

**Step-3:** After complete of the snapshot, login to instance and stop that service which are using that volume. In our case it is mysqld.

# systemctl stop mysqld.

**Step-4:** After that unmount that volume.

# umount /DB (comment into in /etc/fstab also)

**Step-5:** After that Click on “Volumes” under ELASTIC BLOCK STORE menu (on the left)

**Step-6:** Select that volume and detach from action menu.

**Step-7:** Now it’s time to increase the size, select the volume and modify it from the action menu.

And Set the new size for your EBS volume (in this case i extended an 1TB volume to 1.2TB)

**Step-8:** Click on modify.

**Step-9: Attach the volume from the action menu now.**

**Step-10:** Now, we need to extend the partition itself. SSH to the EC2 instance where the EBS we’ve just extended is attached to. Type the following command to list our block devices:

[root@epm\_db\_prod ~]# lsblk

**You should be able to see a similar output:**

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

xvda 202:0 0 75G 0 disk

└─xvda1 202:1 0 75G 0 part /

xvdf 202:80 0 1.2T 0 disk /DB

**Step-11:** As you can see size of the /DB volume reflects the new size, 1.2TB, while the size of the partition reflects the original size, 1 TB, and must be extended before you can extend the file system. To do so, type the following command:

[ec2-user ~]$ sudo growpart /dev/xvdf 0

***Be careful, there is a space between device name and partition number!***

Now we can check that the partition reflects the increased volume size (we can check it with the lsblk command we already used):

[root@epm\_db\_prod ~]# lsblk

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

xvda 202:0 0 75G 0 disk

└─xvda1 202:1 0 75G 0 part /

xvdf 202:80 0 1.2T 0 disk /DB

**Step-12:** Last but not least, we need to extend the filesystem itself.

If your filesystem is an ext2, ext3, or ext4, type:

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

If your filesystem is an XFS, then type:

[ec2-user ~]$ sudo xfs\_growfs /dev/xvda1

**Step-12:** Now mount the volume

[root@epm\_db\_prod ~]# mount /*dev/*xvdf1 /DB

(Comment out from /etc/fstab also if done before)

**Step-13:** Start the service which we had stop before.

[root@epm\_db\_prod ~]# systemctl start mysqld

**Step-14:** Finally we can check our extended filesystem by typing:

[root@epm\_db\_prod ~]# df -Th

If everything went right, we should be able to see our effective filesystem extended size:

Filesystem Type Size Used Avail Use% Mounted on

tmpfs tmpfs 16G 0 16G 0% /sys/fs/cgroup

/dev/xvda1 xfs 75G 52G 24G 69% /

**/dev/xvdf ext4 1.2T 969G 204G 83% /DB**

*You have just extended your EBS volume size, enjoy!*