Create Extended EBS Backed LVM Volume on EC2

Sharing one of my use-cases, Jenkins has been increasing over past few weeks, and we were about to hit the 50GB capping for Elastic Block Storage volumes on the Amazon EC2. And this is a problem that is affecting a lot of our team developers to test their build packages on the different environments. But, every time we have to increase it, a maintenance/downtime window has to be scheduled where the Jenkins server service is stopped, an EBS snapshot of the data volume is created and a newer volume with increase capacity is created and attached to the instance. And the network performance bottlenecks between EC2 instances and the EBS volume that generally impact the complete system performance.

To fix these issues, we decide to leverage existing Logical Volume Managers(LVM) feature which provides the option to easily expand the size of their volume by adding one or more EBS volumes. With multiple EBS volumes, network performance is increased between AWS resources like EC2 instances and EBS volumes.

In my post, I will walk through the procedure of setting up with the LVMs on Ubuntu in the AWS EC2 environment, and some time period for the maintenance to add and remove (where possible) storage to avoid the interruption or any downtime.

Getting Started

- In my case, we are attaching three EBS volumes with 1GB SSD . The EBS volumes are given the following device names: *'/dev/xvdb*, *'/dev/xvdc'* and *'/dev/xvdd'*. (To Make LVM utilities installed):

```
root@ip-172-31-53-165:/home/ubuntu# lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdb    202:16   0   1G  0 disk
xvdc    202:32   0   1G  0 disk
xvdd    202:48   0   1G  0 disk
```

- The first step, launch a base image Ubuntu EC2 instance launched with the IAM Role/Profile; the root file system is an EBS Volume.
- Initialize “pvcreate” on the additional volumes to use with LVM filesystem. In order to do this, we need to use the “pvcreate” command. And “pvcreate” command is mainly used to initialize disk or partitions that will be used by LVM. It can either initialize a whole disk or a partition on the physical disks.Syntax so “pvcreate <device-a> <device-b> <device-c> <device-n>”

```
root@ip-172-31-53-165:/home/ubuntu# pvcreate /dev/xvdb /dev/xvdc
  Physical volume "/dev/xvdb" successfully created
  Physical volume "/dev/xvdc" successfully created
```

- Use the “pvdisplay” command utility to display information about mount physical disks:

```
root@ip-172-31-53-165:/home/ubuntu# pvdisplay
  --- Physical volume ---
  PV Name               /dev/xvdb
  VG Name               vgebs
  PV Size               1.00 GiB / not usable 4.00 MiB
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              255
  Free PE               255
  Allocated PE          0
  PV UUID               usyAZS-imps-y912-kOCb-eFz2-vodp-81DoXi

  --- Physical volume ---
  PV Name               /dev/xvdc
  VG Name               vgebs
  PV Size               1.00 GiB / not usable 4.00 MiB
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              255
  Free PE               255
  Allocated PE          0
  PV UUID               PuwepN-O87v-G2jr-2FPR-QmsM-Q2YG-twLRA3
```

- Then volume group called “vgebs” is created. This volume group includes our two disks:

```
root@ip-172-31-53-165:/home/ubuntu# vgcreate vgebs /dev/xvdb /dev/xvdc
  Volume group "vgebs" successfully created
```

- Once the LVM volume group is created, use the “vgdisplay” command to show its attributes.
- Create Logical Volumes. Once the LVM volume group is created, now it’s time to create logical volumes Syntax: sudo lvcreate –name <logical-volume-name> –size <size-of-volume> <lvm-volume-name>

```
root@ip-172-31-53-165:/home/ubuntu# lvcreate -n lvebs -L 1.9G vgebs
  Rounding up size to full physical extent 1.90 GiB
  Logical volume "lvebs" created
```

- This will create a new device located at “/dev/mapper/vgebs-lvebs”. And this is our LVM volume, and we can now create a filesystem and mount it. Once the LVM volumes are created, then we can format them using any another type of filesystem like ext3, XFS, ext4 and more. If you are using the filesystem”ext3″ :
  mkfs.ext3 <logical-volume-path>

```
root@ip-172-31-53-165:/home/ubuntu# mkfs.ext3 /dev/vgebs/lvebs
mke2fs 1.42.9 (4-Feb-2014)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
124672 inodes, 498688 blocks
24934 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=511705088
16 block groups
32768 blocks per group, 32768 fragments per group
7792 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

- To mount the logical volumes using the mount command:

```
root@ip-172-31-53-165:/home/ubuntu# mount /dev/vgebs/lvebs /var/lib/jenkins
```

- If we want then mount points to be available after reboot the system, we can add mount point entries in file *‘/etc/fstab’*.And check the status using mount volume:

```
root@ip-172-31-53-165:/home/ubuntu# df -h
Filesystem                Size  Used Avail Use% Mounted on
udev                      240M   12K  240M   1% /dev
tmpfs                      49M  348K   49M   1% /run
/dev/xvda1                7.8G  801M  6.6G  11% /
none                      4.0K     0  4.0K   0% /sys/fs/cgroup
none                      5.0M     0  5.0M   0% /run/lock
none                      245M     0  245M   0% /run/shm
none                      100M     0  100M   0% /run/user
/dev/mapper/vgebs-lvebs   1.9G  2.9M  1.8G   1% /var/lib/jenkins
```

- Our first step is to create two more EBS volumes (1GB each) and attach to AWS ec2 instance. And now we are running out of disk space on EBS for our mount point? So we need to increase it out by now we initialize this as the Physical volume for the LVM. After the new volumes have been

attached to our EC2 instance we check “dmesg” to check the exact mapping (EC2 instances may sometimes change the name of the devices):

```
root@ip-172-31-53-165:/home/ubuntu# pvcreate /dev/xvdd
  Physical volume "/dev/xvdd" successfully created
root@ip-172-31-53-165:/home/ubuntu# pvdisplay
  --- Physical volume ---
  PV Name               /dev/xvdb
  VG Name               vgebs
  PV Size               1.00 GiB / not usable 4.00 MiB
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              255
  Free PE               0
  Allocated PE          255
  PV UUID               usyAZS-imps-y912-kOCb-eFz2-vodp-81DoXi

  --- Physical volume ---
  PV Name               /dev/xvdc
  VG Name               vgebs
  PV Size               1.00 GiB / not usable 4.00 MiB
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              255
  Free PE               23
  Allocated PE          232
  PV UUID               PuwepN-O87v-G2jr-2FPR-QmsM-Q2YG-twLRA3

  "/dev/xvdd" is a new physical volume of "1.00 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/xvdd
  VG Name
  PV Size               1.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               m5632a-QBOA-I7Wt-gqmD-cjhM-fGaE-CUxZ32
```

- And then add this disk to our existing “vgebs” Volume Group:

```
root@ip-172-31-53-165:/home/ubuntu# vgextend vgebs /dev/xvdd
  Volume group "vgebs" successfully extended
```

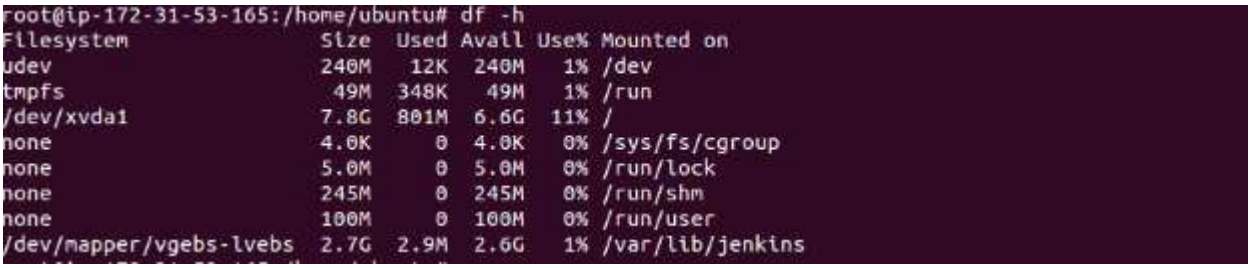
- Now we extend the LVM volume to use the whole size of the group. Please note that in theory, we should “lvextend” to use 800Mb or 0.8G, but since the LVM needs to reserve some space for internal data we must leave a few GiB available:

```
root@ip-172-31-53-165:/home/ubuntu# lvextend -L+0.8G /dev/mapper/vgebs-lvebs
  Rounding size to boundary between physical extents: 820.00 MiB
  Extending logical volume lvebs to 2.70 GiB
  Logical volume lvebs successfully resized
root@ip-172-31-53-165:/home/ubuntu#
```

- We can now use “resize2fs” to extend the filesystem until the end of the LVM volume:

```
root@ip-172-31-53-165:/home/ubuntu# resize2fs /dev/mapper/vgebs-lvebs
resize2fs 1.42.9 (4-Feb-2014)
Filesystem at /dev/mapper/vgebs-lvebs is mounted on /var/lib/jenkins; on-line resizing required
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/mapper/vgebs-lvebs is now 708608 blocks long.
```

- Check the status of the mount point “/var/lib/jenkins”. Then we will find that the disk size of the device “/dev/mapper/vgebs-lvebs” increased to 1.9GB to 2.7GB:



Once the LVMs are created and mounted, we can use these as normal volumes. However, LVM offers:

- **With Better performance** – If data is spread across all the multiple EBS volumes using LVM, we can leverage the dedicated network throughput between AWS EC2 and EBS. This provides us better network throughput over a single network channel between EC2 instances and EBS volumes.
- **Ability to grow** – We also expand volume at any time according to requirement. More EBS volumes can be added to existing LVM volume instead of creating a snapshot of an EBS volume and expanding it.
- **The EBS volume snapshots** – Also, need to ensure that on disk there are no operations happening on EBS volume during snapshots. Also to suspend the operation (Read or Write) on the LVM volume by using “dmsetup” command.

Syntax for the suspend : # dmsetup suspend <lvm-volume-name>
Syntax for the resume : # dmsetup resume <lvm-volume-name>
As mentioned in my post, LVMs volume backup can be created using the EBS snapshot procedure, but need to ensure that LVM volume operations are suspended for that time duration .