Perforce Helix Core Sample Storage Setup - AWS

Perforce Professional Services

Version v2025.1, 2025-06-04

Table of Contents

Preface

This document illustrates a basic set of sample commands to format data storage to make it usable.

The goal in these examples is to configure 3 EBS storage volumes on an EC2 instance destined to become a Helix Core server. At the start of this procedure, volumes with no data are formatted.

There are many ways to provision a server machine and storage for Helix Core. This method covers a specific and basic example of how it can be done in AWS, though much of this would also apply in Azure, GCP, other clouds, or even for on-prem installation. Essentially, once you have storage associated with the virtual machine ("EC2 instance" in AWS parlance), the steps on the machine are the same. This samples in this document do not illustrate more sophisticated options such as using logical volume managers or NFS storage.

Please Give Us Feedback

Perforce welcomes feedback from our users. Please send any suggestions for improving this document or the SDP to consulting-helix-core@perforce.com.

Chapter 1. Sample Storage Setup for SDP in AWS - Mounts and Storage

1.1. Starting State

This procedure assumes the following start state:

- EC2 instance is launched with an AMI that has RHEL/Rocky 8, RHEL/Rocky 9, Ubuntu 22.04, or 24.04.
- 3 EBS gp3 storage volumes are attached to the EC2 instance (in addition to the OS root volume), intended to become `/hxdepots`, `/hxlogs`, and `/hxmetadta`:
  - `/hxdepots` - Give this as much space as you think you'll need. This is highly variable. You may want 30G source code projects, or 1T or more for virtual production or game development. Size can be increased easily and non-disruptively later in AWS, so you don't need to overprovision (and overpay) for storage.
  - `/hxmetadata` - Use 25G to start.
  - `/hxlogs` - Use 20G to start.
- You can somehow login to the machine in at terminal session with a user that has full `sudo` priviledges. Commonly this user is the `rocky` user for RHEL/Rocky systems, and `ubuntu` (for Ubuntu OS). The user is determined by the AMI you choose.

In AWS, you may be able to use the AWS SSM service to get a terminal on the machine.

1.2. Storage Formatting and Mounting Procedure

First, become root with the `sudo su -` command, and then list the mounted storage before we mount the new volumes:

```
sh-4.4$ sudo su -
```

```
[root@ip-172-31-66-160 ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        1.8G     0  1.8G   0% /dev
tmpfs           1.8G     0  1.8G   0% /dev/shm
tmpfs           1.8G   17M  1.8G   1% /run
tmpfs           1.8G     0  1.8G   0% /sys/fs/cgroup
/dev/nvme0n1p1   10G  1.6G  8.5G  16% /
tmpfs           356M     0  356M   0% /run/user/0
```

You don't yet see the newly attached EBS volumes, as they're not yet mounted. But you can list them with `lsblk`:

```
[root@ip-172-31-66-160 ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
nvme0n1     259:0    0   10G  0 disk
└─nvme0n1p1 259:1    0   10G  0 part /
nvme1n1     259:2    0  100G  0 disk
nvme2n1     259:3    0  500G  0 disk
nvme3n1     259:4    0  120G  0 disk
```

Use the `mkfs` (Make Filesystem) command to format the devices with the XFS filesystem so they can be used for storage. See the left column showing things like `nvme3n1`, which is the device name. It is referenced on the command line by prefixing with `/dev/`, as in `/dev/nvme1n1`. So to create the 3 filesystems, do:

```
[root@ip-172-31-66-160 ~]# mkfs -t xfs /dev/nvme1n1
meta-data=/dev/nvme1n1           isize=512    agcount=16, agsize=1638400 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1
data     =                       bsize=4096   blocks=26214400, imaxpct=25
         =                       sunit=1      swidth=1 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=12800, version=2
         =                       sectsz=512   sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

```
[root@ip-172-31-66-160 ~]# mkfs -t xfs /dev/nvme2n1
meta-data=/dev/nvme2n1           isize=512    agcount=16, agsize=8192000 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1
data     =                       bsize=4096   blocks=131072000, imaxpct=25
         =                       sunit=1      swidth=1 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=64000, version=2
         =                       sectsz=512   sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

```
[root@ip-172-31-66-160 ~]# mkfs -t xfs /dev/nvme3n1
meta-data=/dev/nvme3n1           isize=512    agcount=16, agsize=1966080 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1
data     =                       bsize=4096   blocks=31457280, imaxpct=25
         =                       sunit=1      swidth=1 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=15360, version=2
```

```
         =                       sectsz=512   sunit=1 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Next, make a backup copy of the `/etc/fstab` file, and then modify that file to create new volumes.

```
[root@ip-172-31-66-160 ~]# cd /etc
```

```
[root@ip-172-31-66-160 etc]# ls -l fstab*
-rw-r--r--. 1 root root 394 Nov 15 04:43 fstab
```

```
[root@ip-172-31-66-160 etc]# cp -p fstab fstab.bak.2022-03-08
```

Next, figure out the "block IDs" of each storage device using the `blkid` command to get the block id from the device name. Figure out which one is to be `/hxdepots` (biggest), which is to be `/hxmetadata` (in between), and which is to be `/hxlogs` (smallest). In our example here, `/hxdepots` is the 500G one, `/hxmetadata` is 120G, and `/hxlogs` is `100G`.

That will look something like this:

```
[root@ip-172-31-66-160 etc]# blkid /dev/nvme3n1
/dev/nvme3n1: UUID="b80da85c-9a29-4d7f-b536-821916f9f1d4" BLOCK_SIZE="512" TYPE="xfs"
```

```
[root@ip-172-31-66-160 etc]# echo 'UUID=b80da85c-9a29-4d7f-b536-821916f9f1d4
/hxmetadata xfs defaults 0 0' >> /etc/fstab
```

```
[root@ip-172-31-66-160 etc]# blkid /dev/nvme1n1
/dev/nvme1n1: UUID="4e88d42c-b0b5-42cf-bb78-b297e6878a3f" BLOCK_SIZE="512" TYPE="xfs"
```

```
[root@ip-172-31-66-160 etc]# echo 'UUID=4e88d42c-b0b5-42cf-bb78-b297e6878a3f /hxlogs
xfs defaults 0 0' >> /etc/fstab
```

```
[root@ip-172-31-66-160 etc]# blkid /dev/nvme2n1
/dev/nvme2n1: UUID="f517a730-d501-4535-bf79-ab4f3cbf80e7" BLOCK_SIZE="512" TYPE="xfs"
```

```
[root@ip-172-31-66-160 etc]# echo 'UUID=f517a730-d501-4535-bf79-ab4f3cbf80e7 /hxdepots
xfs defaults 0 0' >> /etc/fstab
```

Be careful when typing the above, especially with quoting. For example, the `blkid` output displays `UUID="BIG_STRING"` (with double quotes), and when you create the `echo` command, you'll specify

`UUID='BIG_STRING'` with single quites. Also be careful to use `>>` to append to the end of the `/etc/fstab` file — do not use `>` which will overwrite it.

Proceed with creating empty directories that will be the "mount points" for the volume to be mounted.

```
[root@ip-172-31-66-160 etc]# mkdir /hxdepots /hxlogs /hxmetadata
```

Next, use the `mount -a` command. This will now assocate the mount points you just created with the storage device information that is now in that `/etc/fstab` file, and mount the volumes.

```
[root@ip-172-31-66-160 etc]# mount -a
```

Then see if they are mounted. This is what victory looks like, with the `/hx*` volumes all mounted with desired sizes:

```
[root@ip-172-31-66-160 etc]# df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        1.8G     0  1.8G   0% /dev
tmpfs           1.8G     0  1.8G   0% /dev/shm
tmpfs           1.8G   17M  1.8G   1% /run
tmpfs           1.8G     0  1.8G   0% /sys/fs/cgroup
/dev/nvme0n1p1   10G  1.6G  8.5G  16% /
tmpfs           356M     0  356M   0% /run/user/0
/dev/nvme3n1    120G  890M  120G   1% /hxmetadata
/dev/nvme1n1    100G  747M  100G   1% /hxlogs
/dev/nvme2n1    500G  3.6G  497G   1% /hxdepots
```

At this point, you are ready to install the Server Deployment Package (SDP) software.