Troubleshooting

Troubleshooting

The following section highlights common issues seen when installing and using the HyperFlex CSI integration. The information provided includes symptoms to help diagnose the issue as well as a solution to resolve the issue.

ImagePullBackOff Status Errors when Deploying HXCSI Pods

- **Symptom 1:** Running the command “kubectl get pods [-n <namespace>]” shows that the HXCSI pods are showing a status of “ImagePullBackOff”.
- **Symptom 2:** Running the command “kubectl describe pod <csi-pod_name>” shows a message containing the following error: “Error: ErrImagePull” and “Back-off pulling image…”

Solution:

- **Solution 1:** Ensure the HXCSI container image name provided to the hxcsi-setup script is correct
- **Solution 2:** Ensure the HXCSI container image exists, either directly within docker on each Kubernetes worker node or on the local container image registry.
- **Solution 3:** Ensure that the “imagePullPolicy” lines in the following YAML files generated by the hxcsi-setup script are set to “IfNotPresent”: `csi-attacher-hxcsi.yaml`, `csi-nodeplugin-hxcsi.yaml`, `csi-provisioner-hxcsi.yaml`

Volume Delete Fails

Volume delete fails due to stale volume attachments present even after NodeUnpublish succeeds and the volume is unmounted. This happens when the delete volumeattachment kubernetes api is lost during etcd leader election. Even after nodeUnpublish completes and the volume is successfully unmounted from the node, delete volume fails.

The log in the external-provisioner appears as:

```
volume deletion failed: persistentvolume <pv-name> is still attached to node <node-name>.
```

The log in the external-attacher appears as:

```
<Volume-attachment> is already attached.
```

Solution:

Delete the stale volumeattachments using the following command:

```
kubectl delete volumeattachments <VA-name>
```

After few seconds when the provisioner retries, pv is deleted.

Alternatively, you can delete manually using the following command:

```
kubectl delete pv <pv-name>
```

Application Pod in ContainerCreating State During Node Deletion

Application pods that get stuck in a ContainerCreating state or multi attach error state and are not able to mount the volumes. This may happen during the remove or delete a k8s worker node from the cluster and pods migrate to a new worker node.

The recommended way to delete a K8s worker node is by using the following commands:

```
kubectl drain <node-name>
kubectl delete  node   <node-name>
```

For more information, see How to gracefully remove a node from Kubernetes?.

Application Pods Stuck in Terminating or ContainerCreating State

Application pods may be seen in Terminating or ContainerCreating state. Common reasons include: The VM restart did not complete and is in hung state. This can happen if there is a process on the VM that places an inhibit lock on systemd and the process never completes its pre-shutdown task(s), and systemd fails to terminate the process after the grace period as shown in the following example:

```
ubuntu@m5-k8-3:~$ systemd-inhibit --list
Who: Unattended Upgrades Shutdown (UID 0/root, PID 778/unattended-upgr)
What: shutdown
Why: Stop ongoing upgrades or perform upgrades before shutdown
```

```
Mode: delay

1 inhibitors listed.
ubuntu@m5-k8-3:~$ ps aux | grep 778
root 778 0.0 0.1 185948 20028 ? Ssl May13 0:00 /usr/bin/python3
 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wait-for-signal
ubuntu@m5-k8-3:~$
```

The VM may also be left in hung state when iscsid failed to shutdown correctly. The shutdown process may wait for a maximum period before killing iscsid but the VM may not reboot. Check for any iSCSI connection errors reported such as shown in the following example:

```
Jun 09 19:12:46 m5-k19-3 iscsid[967]: Kernel reported iSCSI connection 2:0 error
 (1010 - ISCSI_ERR_BAD_ITT: Received invalid initiator task tag from target) state (3)
Jun 09 19:12:48 m5-k19-3 iscsid[967]: connection2:0 is operational after recovery (1 attempts)
Jun 11 15:17:27 m5-k19-3 iscsid[967]: Kernel reported iSCSI connection 2:0 error (1010 -
ISCSI_ERR_BAD_ITT: Received invalid initiator task tag from target) state (3)
Jun 11 15:17:29 m5-k19-3 iscsid[967]: connection2:0 is operational after recovery (1 attempts)
```

Solution:

Log into the VM console and ensure that VM has rebooted correctly and kubelet is responding.

- The worker node must be in Ready state. If the VM has not fully shutdown, restart it again.
- If iscsid fails to shutdown gracefully during a VM restart, check the iSCSI data path connection to the HXDP target (connection errors, MTU, retries, and so on).

Application pods may be seen in Terminating or ContainerCreating state, after a VM restart or reboot is performed. The root cause may likely be a hung VM that did not complete the shutdown process and therefore left the Kubelet in an unresponsive state. By default, the API controller-manager waits for 5 minutes for kubernetes node to come up and then decides to evict or recreate the Application pod elsewhere.

To recover from the situation, reboot or reset the VM again and ensure the VM comes up.

Deleted Pod Scheduled Back on the Same Node

A pod in **Running** state may get recreated after deleting using the `kubectl delete pod` command and then get stuck in **Terminating** state on deleting its namespace.

Instead of using the `kubectl delete pod` command on any running pod, the following best practices way is recommended:

1. Note the node name on which pod to be deleted is running on.

   ```
   kubectl get pods -o wide --all-namespaces
   ```

2. Cordon off the node on which the pod is running.

   ```
   kubectl cordon <node-name>
   ```

3. Delete the pod.

   ```
   kubectl delete pod <pod-name>
   ```

4. Check that the deleted pod is scheduled on a different node using the following command:

   ```
   kubectl get pods -o wide --all-namespaces
   ```

5. Uncordon the cordoned node.

```
kubectl uncordon <node-name>
```

Pod in ContainerCreating State When Using Volume From Snapshot

When creating a volume from a snapshot, the storage class used for persistenvolumeclaim should be the same as the source volume. Using a different storage class with attributes that result in conflict with the data stored on the snapshot is not allowed. For example, the storage class should not specify a filesystem different than the filesystem on the snapshot or source volume. If such a change is made, then the user pod created to consume the volume fails or waits at the "ContainerCreating" state.

To recover from this state, proceed as follows:

- Delete the failed user pod.
- Fix the storage class used in PVC.
- Create a new PVC.
- Redeploy the user pod.

Snapshot Fails After HXCSI Re-install

HXCSI installs CRDs per the CSI specification to support snapshots. If HXCSI is uninstalled without deleting the resources that use the CRD, then Kubernetes triggers a force clean up during uninstall. If a re-install is attempted when the previous clean up is not complete, then some of the CRDs are not created correctly and snapshot operation fails.

To avoid the problem when you uninstall HXCSI, verify the following:

- Before uninstalling HXCSI, delete all volumeSnapshot. Note that you can retain the volumes created from the snapshots.
- Use `kubectl get volumesnapshot` to list all snapshots.
- After uninstalling, verify that you have deleted all CRDs.
- Run the `kubectl api-resources|grep snapshot` command, and verify that no resources are returned.

If you cannot verify any of the above, then you will need to force delete all of the resources indicated before re-installing.

Snapshot Resource Count on Kubernetes and HXDP Does Not Match

Kubernetes and HXDP resources will be in sync if all operations are triggered through the Kubernetes request objects. For snapshots, VolumeSnapshot is the request object that ensures calls to the HXCSI plugin, and then subsequently creates or deletes on HXDP. If operations are not triggered through the request object it can lead to HXDP resources not being deleted.

This can occur in the following cases:

- Snapshot class uses a Retain policy that deletes only VolumeSnapshot but retains volumesnapshotcontent. This retains the snapshot on the HXDP and requires manual clean up
- Volumesnapshotcontent or volume snapshot is force deleted on Kubernetes
- Connectivity issue with HXDP during snapshot deletion followed by a force delete of snapshot

For all the above cases, you will need to manually sync the resources between HXDP and Kubernetes.