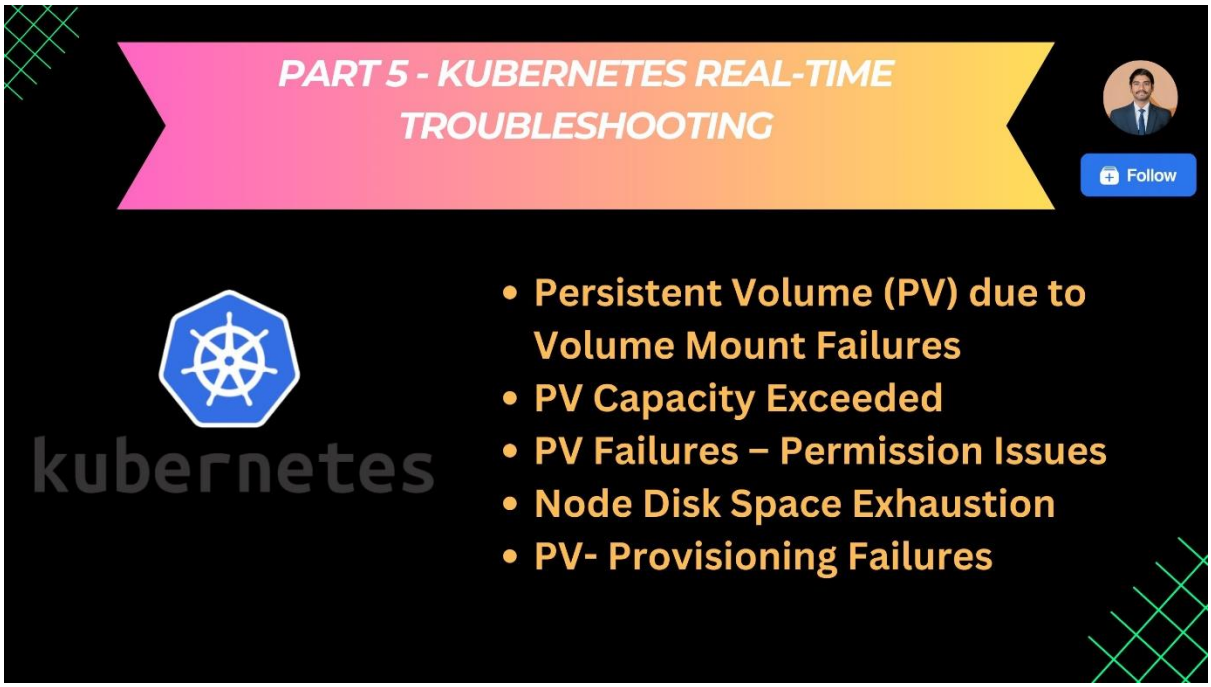
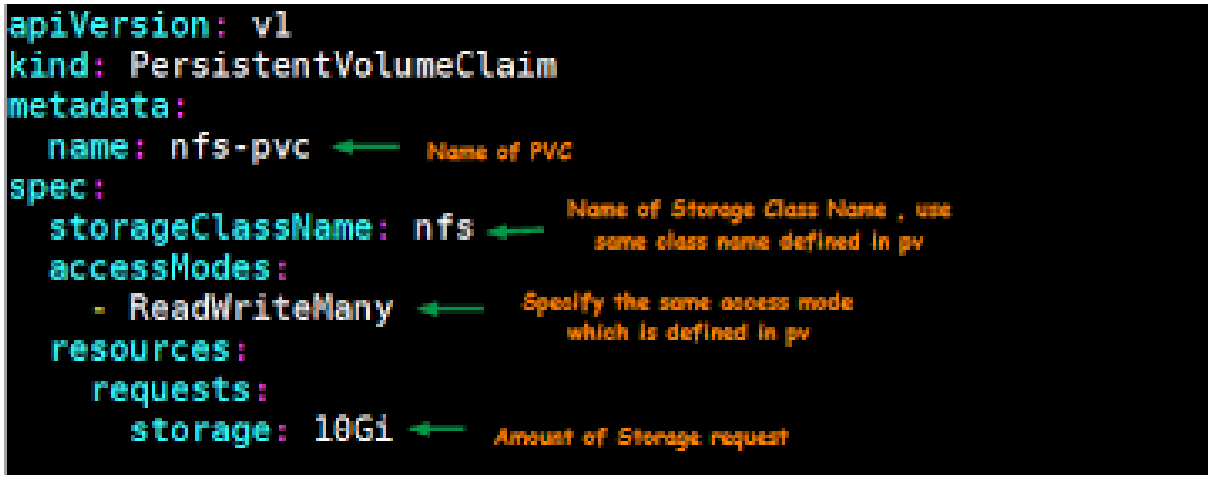
Part 5 - Kubernetes Real-Time Troubleshooting

Introduction 🌐

Welcome to the world of Kubernetes troubleshooting, where every challenge is an opportunity to sharpen your skills and emerge victorious. Join us as we embark on a journey through common real-time scenarios, unraveling mysteries, and uncovering solutions along the way. Below, we address cases related to Kubernetes storage failures:



Scenario 21: Persistent Volume (PV) due to Volume Mount Failures



Symptoms: Pods fail to start or restart due to persistent volume mount failures, resulting in volume-related errors.

Diagnosis: Check pod events (`kubectl describe pod <pod_name>`) and PV status (`kubectl get pv`) for any mount errors or volume provisioning issues.

Solution:

1. Verify that the PV is bound to the correct PersistentVolumeClaim (PVC) and that the PVC is in the Bound state.
2. Check storage provider logs or cloud storage services for any errors or connectivity issues impacting PV provisioning.
3. Ensure that the correct storage class is specified in the PVC and that it matches the storage backend configuration.
4. Troubleshoot node connectivity issues or network policies that may prevent pods from accessing PVs.

Scenario 22: Persistent Volume (PV) Capacity Exceeded

```
radamuz@radamuz:~$ kubectl describe pvc pvcsc-vsan
Name:          pvcsc-vsan
Namespace:     default
StorageClass:  fast
Status:        Pending
Volume:
Labels:        <none>
Annotations:   volume.beta.kubernetes.io/storage-class: fast
               volume.beta.kubernetes.io/storage-provisioner: kubernetes.io/vsphere-volume
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Used By:       <none>
Events:
  Type     Reason              Age              From                         Message
  ----     ------              ----             ----                         -------
  Warning  ProvisioningFailed  3s (x6 over 74s)  persistentvolume-controller  Failed to provision
 volume with StorageClass "fast": cloud provider not initialized properly
```

Symptoms: Persistent volumes reach full capacity, causing pod failures or data loss due to inability to write data to disk.

Diagnosis: Check PV and PVC status (`kubectl get pv,pvc`) and examine storage provider metrics for disk usage and capacity utilization.

Solution:

1. Increase the size of existing PVs or provision additional PVs with larger storage capacity to accommodate growing data volumes.
2. Implement data retention policies and automatic data pruning mechanisms to remove obsolete or unused data and free up storage space.
3. Monitor disk usage trends and forecast storage requirements to proactively scale storage resources and prevent capacity bottlenecks.
4. Configure alerts and notifications to notify administrators when PV capacity thresholds are nearing or exceeded to take timely action.

Scenario 23: Persistent Volume Mount Failures – Permission Issues

```
invisible-squirrel-mariadb         Bound     local-pv-87bc97ab   9647604Ki   RWO            l
al-storage   17h
nonplussed-swan-mariadb            Bound     local-pv-6923a82c   9647604Ki   RWO            l
al-storage   17h
ornery-lemur-joomla-apache         Bound     local-pv-9bfd6008   9647604Ki   RWO            l
al-storage   17h
ornery-lemur-joomla-joomla         Bound     local-pv-125a2537   9647604Ki   RWO            l
al-storage   17h
ornery-lemur-mariadb               Bound     local-pv-20cd15aa   9647604Ki   RWO            l
al-storage   17h
bitnami@bitnami-kubernetessandbox-dm-35c0:~$ kubectl describe pvc incindiary-opossum-wordpre
ss
Name:          incindiary-opossum-wordpress
Namespace:     default
StorageClass:  local-storage
Status:        Pending
Volume:
Labels:        app=incindiary-opossum-wordpress
               chart=wordpress-0.7.7
               heritage=Tiller
               release=incindiary-opossum
Annotations:   volume.beta.kubernetes.io/storage-provisioner=local-storage
Capacity:
Access Modes:
Events:
  Type    Reason                Age                  From                         Message
  ----    ------                ----                 ----                         -------
  Normal  ExternalProvisioning  16h (x322 over 17h)  persistentvolume-controller  waiting fo
r a volume to be created, either by external provisioner "local-storage" or manually created
 by system administrator
  Normal  ExternalProvisioning  3m (x162 over 43m)   persistentvolume-controller  waiting fo
r a volume to be created, either by external provisioner "local-storage" or manually created
 by system administrator
```

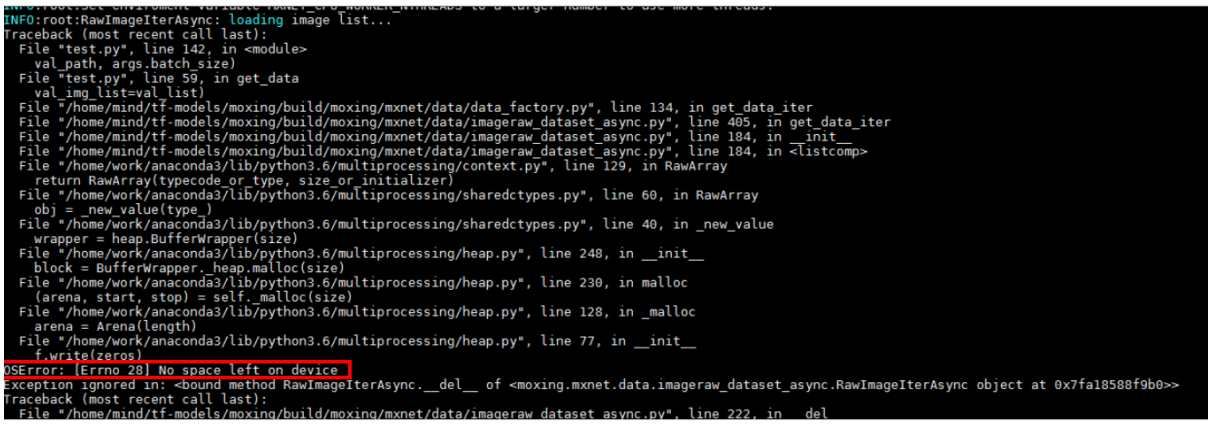
Symptoms: Pods fail to mount persistent volumes (PVs) due to storage provisioning errors, permissions issues, or volume attachment failures.

Diagnosis: Check pod events (`kubectl describe pod <pod_name>`) and PV status (`kubectl get pv`) for any errors or warnings related to volume provisioning, attachment, or access.

Solution:

1. Ensure that PVs are correctly provisioned and dynamically provisioned storage classes are available and accessible to the Kubernetes cluster.
2. Verify storage provider credentials and permissions to ensure that pods have sufficient privileges to mount and access persistent volumes.
3. Check node disk and filesystem integrity to identify any disk failures, corruption, or capacity constraints that may affect volume attachment and mounting operations.
4. Implement volume health checks and monitoring to detect and remediate storage issues proactively, such as disk failures, I/O errors, or storage latency spikes.

Scenario 24: Node Disk Space Exhaustion



Symptoms: Nodes experience disk space exhaustion, causing critical system components (e.g., kubelet, container runtime) to fail and impacting pod scheduling and execution.

Diagnosis: Monitor node disk usage (`df -h`) and inspect system logs (`/var/log/messages`, `journalctl`) for disk-related errors or warnings indicating low disk space conditions.

Solution:

1. Identify and remove unnecessary files or temporary data consuming disk space, such as old log files, unused containers/images, or orphaned volumes.
2. Implement disk space monitoring and alerting to detect and mitigate disk space issues proactively before they lead to service disruptions.
3. Scale out storage capacity by attaching additional disks or expanding existing volumes to accommodate growing storage requirements and prevent future disk space shortages.
4. Configure log rotation and retention policies to limit the size of log files and prevent them from consuming excessive disk space over time.

Scenario 25: Persistent Volume (PV) Provisioning Failures

Symptoms: Pods fail to mount persistent volumes (PVs) due to provisioning failures or storage backend issues, resulting in data loss or application downtime.

Diagnosis: Check PV and PersistentVolumeClaim (PVC) statuses (`kubectl get pv,pvc`) and inspect storage provider logs (e.g., CSI driver logs, storage backend logs) for any provisioning errors or resource constraints.

Solution:

1. Troubleshoot storage backend connectivity issues, such as network connectivity, authentication, or authorization problems between Kubernetes cluster nodes and storage systems.
2. Verify storage class configurations (`kubectl get storageclass`) and ensure that the appropriate storage classes are available and accessible for dynamic provisioning of PVs based on PVC requests.
3. Monitor storage resource utilization and capacity metrics to identify any resource shortages or bottlenecks that may be impacting PV provisioning and allocation.
4. Implement storage resiliency features, such as data replication, redundancy, or snapshotting, to protect against data loss and ensure high availability of persistent volumes in case of storage failures.

In the up-coming parts, we will discussion on more troubleshooting steps for the different Kubernetes based scenarios. So, stay tuned for the and follow @Prasad Suman Mohan for more such posts.

FOLLOW – Prasad Suman Mohan (for more updates)