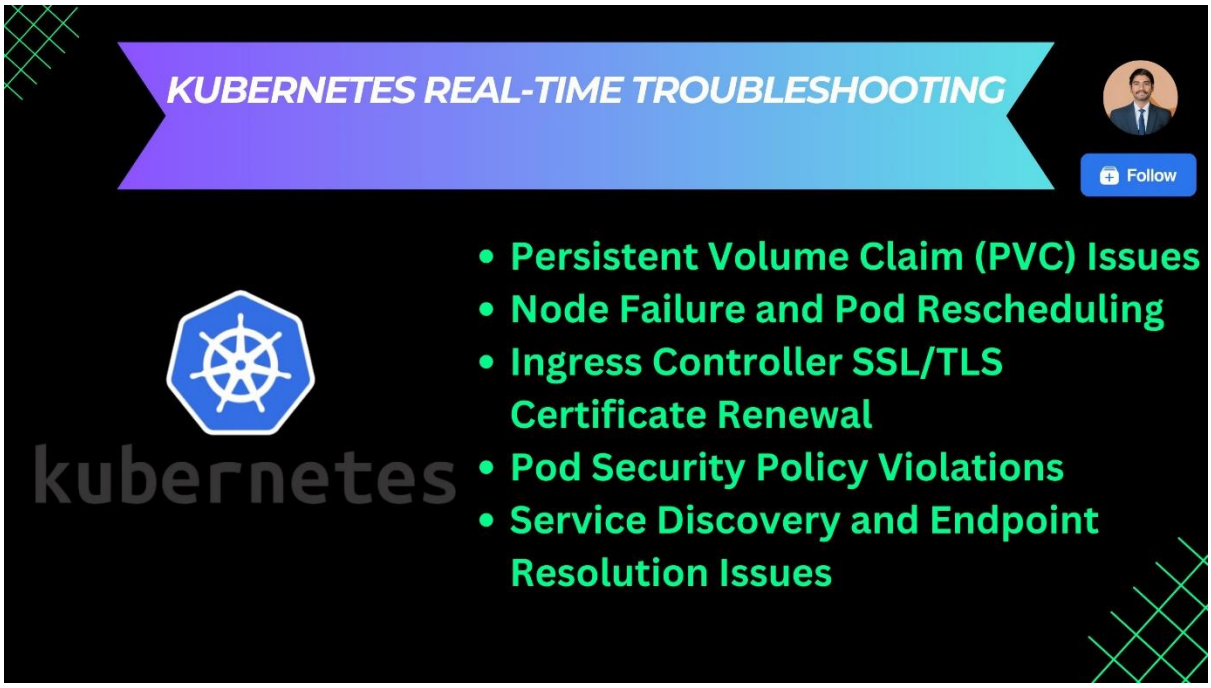
Part 3 - Kubernetes Real-Time Troubleshooting

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.



Scenario 11: Persistent Volume Claim (PVC) Issues

Symptoms: Pods fail to start or remain in a pending state due to issues with attaching persistent volumes claimed by PVCs.

Diagnosis: Check the PVC status (`kubectl get pvc`) and events (`kubectl describe pvc <pvc_name>`) for any errors related to volume provisioning or attachment.

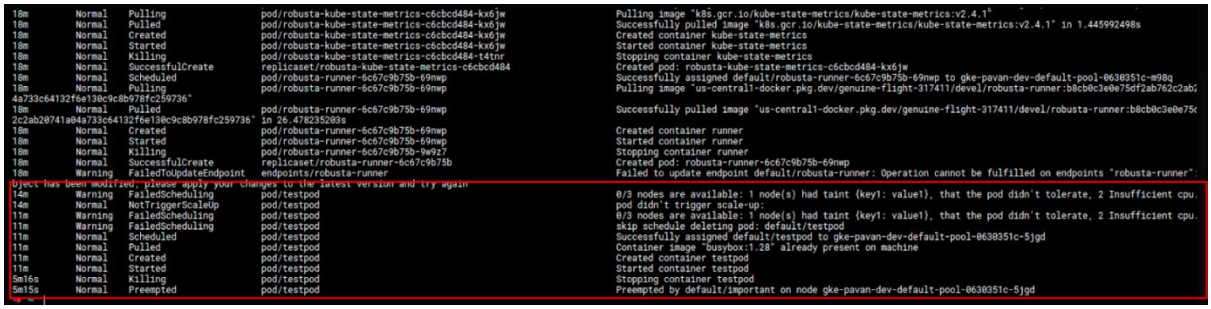
Solution:

1. Ensure that the storage class referenced by the PVC is available and properly configured.
2. Verify that the underlying storage provider (e.g., AWS EBS, EFS, NFS, Azure Disk) is functioning correctly and accessible from the Kubernetes cluster.

```
prasad@ubuntu:~/ansible/kind/kubernetes-troubleshooting-zero-to-hero/04-statefulset-pv$ kubectl get pods
NAME                                READY   STATUS    RESTARTS      AGE
nginx-deployment-7c79c4bf97-7t5d7   1/1     Running   1 (22h ago)   5d23h
nginx-deployment-7c79c4bf97-8qxhc   1/1     Running   1 (22h ago)   5d23h
nginx-deployment-7c79c4bf97-k8pgd   1/1     Running   1 (22h ago)   5d23h
web-0                               0/1     Pending   0             33m
prasad@ubuntu:~/ansible/kind/kubernetes-troubleshooting-zero-to-hero/04-statefulset-pv$ kubectl describe pod web-0
Name:             web-0
Namespace:        default
Priority:         0
Service Account:  default
Node:             <none>
Labels:           app=nginx
                  apps.kubernetes.io/pod-index=0
                  controller-revision-hash=web-79dc58f667
                  statefulset.kubernetes.io/pod-name=web-0
Annotations:      <none>
Status:           Pending
IP:
IPs:              <none>
Controlled By:    StatefulSet/web
Containers:
  nginx:
    Image:        registry.k8s.io/nginx-slim:0.8
Conditions:
  Type           Status
  PodScheduled   False
Volumes:
  www:
    Type:       PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
    ClaimName:  www-web-0
    ReadOnly:   false
  kube-api-access-njbnc:
    Type:                     Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:   3607
    ConfigMapName:            kube-root-ca.crt
    ConfigMapOptional:        <nil>
    DownwardAPI:              true
QoS Class:                    BestEffort
Node-Selectors:               <none>
Tolerations:                  node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason            Age                From               Message
  ----     ------            ----               ----               -------
  Warning  FailedScheduling  34m                default-scheduler  0/3 nodes are available: pod has unbound immediate Pe
rsistentVolumeClaims. preemption: 0/3 nodes are available: 3 Preemption is not helpful for scheduling.
  Warning  FailedScheduling  9m7s (x5 over 29m)  default-scheduler  0/3 nodes are available: pod has unbound immediate Pe
rsistentVolumeClaims. preemption: 0/3 nodes are available: 3 Preemption is not helpful for scheduling.
prasad@ubuntu:~/ansible/kind/kubernetes-troubleshooting-zero-to-hero/04-statefulset-pv$
```

3. Check for any resource constraints or quota limitations preventing volume provisioning.
4. Manually delete and recreate the PVC if stuck in a pending or failed state, ensuring correct specifications and annotations.

Scenario 12: Node Failure and Pod Rescheduling



Symptoms: Pods are terminated or evicted due to node failures, causing service disruptions or downtime.

Diagnosis: Review node status (`kubectl get nodes`) and events (`kubectl get events --field-selector involvedObject.kind=Node`) for indications of node failures or issues.

Solution:

1. Configure node maintenance and auto-scaling policies to automatically replace failed nodes and reschedule affected pods.
2. Implement node affinity and anti-affinity rules to distribute pods across multiple nodes and prevent single points of failure.
3. Monitor node health metrics (e.g., CPU, memory, disk utilization) and set up alerts for abnormal behaviour or resource exhaustion.
4. Consider using node draining and cordoning strategies to gracefully evacuate pods from failing nodes before maintenance or decommissioning.

Scenario 13: Ingress Controller SSL/TLS Certificate Renewal

```
C:\Users\b.vishnupriya>kubectl logs -f ingress-nginx-controller-84797d6955-zl72k -n ingress-nginx -c controller
-------------------------------------------------------------------------------
NGINX Ingress controller
  Release:       v1.5.1
  Build:         d003aae913cc25f375deb74f898c7f3c65c06f05
  Repository:    https://github.com/kubernetes/ingress-nginx
  nginx version: nginx/1.21.6

-------------------------------------------------------------------------------

W0324 09:17:12.004126       8 client_config.go:617] Neither --kubeconfig nor --master was specified.  Using the inClusterConfig.  This might not work.
I0324 09:17:12.004335       8 main.go:209] "Creating API client" host="https://10.0.0.1:443"
I0324 09:17:12.035033       8 main.go:253] "Running in Kubernetes cluster" major="1" minor="24" git="v1.24.6" state="clean" commit="08d3594304660f86cfbd17bbb862041b4b75fe6c
" platform="linux/amd64"
I0324 09:17:12.040755       8 main.go:86] "Valid default backend" service="ingress-nginx/ingress-nginx-defaultbackend"
F0324 09:17:12.438882       8 ssl.go:390] unexpected error storing fake SSL Cert: could not create PEM certificate file /etc/ingress-controller/ssl/default-fake-certificate
.pem: open /etc/ingress-controller/ssl/default-fake-certificate.pem: permission denied
```

Symptoms: SSL/TLS certificate used by the ingress controller for HTTPS termination is expired or nearing expiration, causing security warnings or errors in client browsers.

Diagnosis: Check the certificate expiration date and validity (`kubectl describe secret <tls_secret_name>`). Monitor SSL/TLS certificate renewal logs for any warnings or errors.

Solution:

1. Renew SSL/TLS certificates well before their expiration date using automated certificate management solutions or manual renewal processes.
2. Implement a certificate monitoring and alerting system to notify administrators of upcoming certificate expirations.
3. Configure the ingress controller to automatically reload SSL/TLS certificates upon renewal without requiring pod restarts or service disruptions.
4. Test certificate renewal processes in a staging environment to ensure seamless transition and minimal downtime in production.

Scenario 14: Pod Security Policy Violations

```
Name:           mymariadb-2016424017
Namespace:      default
Selector:       pod-template-hash=2016424017,run=mymariadb
Labels:         pod-template-hash=2016424017
                run=mymariadb
Annotations:    deployment.kubernetes.io/desired-replicas=1
                deployment.kubernetes.io/max-replicas=2
                deployment.kubernetes.io/revision=1
Replicas:       0 current / 1 desired
Pods Status:    0 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:       pod-template-hash=2016424017
                run=mymariadb
  Containers:
   mymariadb:
    Image:      bitnami/mariadb:latest
    Port:       3306/TCP
    Environment:
      ALLOW_EMPTY_PASSWORD:     yes
    Mounts:                     <none>
  Volumes:                      <none>
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath   Type            Reason          Message
  ---------     --------        -----   ----                    -------------   --------        ------          -------
  31s           10s             13      replicaset-controller                   Warning         FailedCreate    Error creating:
 pods "mymariadb-2016424017-" is forbidden: unable to validate against any pod security policy: [spec.containers[0].securityCo
ext.containers[0].hostPort: Invalid value: 3306: Host port 3306 is not allowed to be used.  Allowed ports: [{0 0}]]
```

Symptoms: Pods fail to start or are denied creation due to violations of pod security policies (PSPs), resulting in security compliance issues.

Diagnosis: Review pod security policy logs (`kubectl describe psp <psp_name>`) and audit logs for any policy violations or enforcement failures.

Solution:

1. Update pod specifications to comply with pod security policies, including container runtime constraints, privilege escalation prevention, and volume access controls.
2. Implement admission control webhooks or third-party tools to enforce pod security policies at runtime and prevent non-compliant pods from starting.
3. Monitor and review audit logs regularly to identify potential security breaches or policy violations and take corrective actions.
4. Provide developers with training and resources on secure pod configuration practices and best practices for compliance with organizational security policies.

Scenario 15: Service Discovery and Endpoint Resolution Issues

```
NAME                                   READY     STATUS             RESTARTS   AGE
dnsmasq-alv8k                          1/1       Running            3          4d
dnsmasq-c9y52                          1/1       Running            2          4d
dnsmasq-sjouh                          1/1       Running            2          4d
flannel-kubemaster                     1/1       Running            4          4d
flannel-kubeminion1                    1/1       Running            3          4d
flannel-kubeminion2                    1/1       Running            3          4d
kube-apiserver-kubemaster              1/1       Running            4          4d
kube-controller-manager-kubemaster     1/1       Running            4          4d
kube-proxy-kubemaster                  1/1       Running            24         4d
kube-proxy-kubeminion1                 1/1       Running            23         4d
kube-proxy-kubeminion2                 1/1       Running            23         4d
kube-scheduler-kubemaster              1/1       Running            4          4d
kubedns-aod0u                          2/3       CrashLoopBackOff   23         1h
[root@kubemaster ~]#
```

Symptoms: Services fail to discover or communicate with backend endpoints due to DNS resolution failures or misconfigured service definitions.

Diagnosis: Use `kubectl exec` to troubleshoot DNS resolution from within the affected pods. Check for any misconfigured service definitions or selector labels.

Solution:

1. Verify that the service definition (`kubectl describe service <service_name>`) correctly selects backend pods and exposes the desired ports.
2. Check for any network policies or firewall rules blocking traffic between services and endpoints.
3. Ensure that DNS service discovery is enabled and functioning correctly for all pods and services.
4. Monitor DNS-related logs and metrics for any indications of errors or failures and take corrective actions as needed.

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)