DevOps Shack

50 Kubernetes Errors & Solutions

1. CrashLoopBackOff: Pod fails to start repeatedly.

- **Error Example**: Your pod status shows:

```
kubectl get pods

NAME                READY   STATUS             RESTARTS   AGE
myapp-12345-abcde   0/1     CrashLoopBackOff   10         5m
```

- **Cause**: This error typically indicates that the container crashes soon after starting.
- **Solution**: Check the logs to see why it's failing to start:

```
kubectl logs myapp-12345-abcde
```

- If there's an application error (e.g., a missing file or environment variable), correct the configuration or the container image itself. For instance:

```
# Deployment YAML Example
env:
  - name: DATABASE_URL
    value: "jdbc:postgresql://mydb:5432/mydatabase"
```

---

2. ImagePullBackOff: Kubernetes can't pull the specified container image.

- **Error Example**:

```
kubectl describe pod myapp-12345-abcde

# Output

Failed to pull image "myregistry/myapp:latest": image not found
```

- **Cause**: This occurs if Kubernetes cannot find or access the specified container image.
- **Solution**:
    - Check the image name and tag for errors.
    - Ensure that your image is available in the container registry.
    - If using a private registry, make sure your Kubernetes cluster can access it with proper credentials, as shown below:

imagePullSecrets:

- name: myregistrykey

    - Create a secret if necessary:

kubectl create secret docker-registry myregistrykey --docker-server=myregistry --docker-username=myuser --docker-password=mypass

---

3. ErrImagePull: Failure in pulling the image.

- **Solution**: This error is similar to ImagePullBackOff. Ensure:
    - The image name is correct.
    - The registry is accessible.
    - Use a secure registry if necessary and validate permissions.
    - An example command for pulling:

docker pull myregistry/myapp:latest

    - You can also inspect node-level issues or check firewall settings if you're in a restricted network environment.

---

4. Pending Pods: Pods remain in pending state due to lack of resources or node unavailability.

- **Error Example**:

kubectl get pods

NAME READY STATUS RESTARTS AGE

myapp 0/1 Pending 0 1m

- **Solution**:
    - Check the pod events with:

kubectl describe pod myapp

    - If you see messages like "Insufficient CPU," then the cluster is out of resources. Either scale up your cluster, increase node resources, or decrease the pod's resource requests.

```
resources:
  requests:
    memory: "64Mi"
    cpu: "250m"
```

---

5. Node NotReady: A node is not in a ready state, which prevents pods from running on it.

- **Error Example**:

kubectl get nodes

NAME STATUS ROLES AGE VERSION

worker-node NotReady <none> 30m v1.20.0

- **Solution**:
    - Describe the node to understand why it's not ready:

kubectl describe node worker-node

    - Common causes include network issues, disk pressure, or kubelet failures. You may need to restart the kubelet or fix network configurations on the affected node.

---

6. OOMKilled: The pod gets killed because it uses more memory than allocated.

- **Error Example**:
    - You may find events similar to:

Last State: Terminated

Reason: OOMKilled

Exit Code: 137

- **Solution**:
    - Increase the memory limit of the container:

```
resources:
  limits:
    memory: "128Mi"
  requests:
    memory: "64Mi"
```

    - If your application is prone to memory spikes, you might want to optimize it or provide more resources.

---

7. Unauthorized: Authentication failure when trying to access resources.

- **Error Example**:

Error from server (Unauthorized): pods is forbidden: User "system:serviceaccount:default:myserviceaccount" cannot list resource "pods" in API group "" in the namespace "default"

- **Solution**:
    - Make sure the ServiceAccount is associated with the correct roles.

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: read-pods
  namespace: default
subjects:
  - kind: ServiceAccount
    name: myserviceaccount
    namespace: default
roleRef:
  kind: Role
  name: pod-reader
  apiGroup: rbac.authorization.k8s.io
```

---

8. Forbidden: Authorization issue where the user does not have permissions.

- **Error Example**:

```
Error from server (Forbidden): services is forbidden: User "user" cannot list resource "services" in API group ""
```

- **Solution**:
    - This error generally occurs due to lack of access in RBAC. Check if the user or ServiceAccount has the correct role binding.
    - Example:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: my-clusterrolebinding
subjects:
- kind: User
  name: user
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io
```

---

9. Evicted Pods: Pods are removed from a node due to resource constraints.

- **Error Example**:
    - When nodes are low on resources, pods may be evicted:

```
kubectl get pod myapp-12345-abcde
NAME              READY   STATUS    RESTARTS   AGE
myapp-12345-abcde     0/1     Evicted   0          1m
```

- **Solution**:
    - Either add resources to nodes or reduce resource requests in your workloads. Additionally, ensure resource limits are configured properly to avoid excessive resource consumption.

---

10. PVC Bound Issues: PersistentVolumeClaims (PVCs) are not bound to PersistentVolumes (PVs).

- **Error Example**:
    - If a PVC status remains “Pending”:

kubectl get pvc

NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE

my-pvc Pending <none> <none> <none> standard 5m

- **Solution**:
    - Check that a matching PersistentVolume is available and meets the PVC requirements.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0001
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: standard
  hostPath:
    path: "/mnt/data"
```

11. Service Not Accessible: Services are not reachable within or outside the cluster.

- **Error Example**:
    - Trying to access a service at http://service-name:port but getting a timeout.
- **Solution**:
    - Verify the service is correctly configured and check if pods are assigned to the service.
    - Run:

```
kubectl describe service my-service
```

- Ensure service type (e.g., ClusterIP, NodePort, LoadBalancer) is correctly configured and check firewall settings if necessary.

---

12. DNS Resolution Failures: Pods cannot resolve DNS names within the cluster.

- **Error Example**:
  - Pod logs may show errors like "Unable to resolve host: [hostname]."
- **Solution**:
  - Verify CoreDNS pods are running:

```
kubectl get pods -n kube-system -l k8s-app=kube-dns
```

  - Restart CoreDNS if needed, and verify your pod's DNS settings:

```
dnsPolicy: ClusterFirst
```

---

13. Certificate Errors: TLS certificates are invalid or expired.

- **Error Example**:
  - Accessing a Kubernetes service with TLS and receiving "Certificate Expired" error.
- **Solution**:
  - Renew certificates with kubeadm certs renew (if using kubeadm). Update your TLS secret:

```
kubectl delete secret my-cert
```

```
kubectl create secret tls my-cert --cert=path/to/tls.crt --key=path/to/tls.key
```

---

14. API Server Unreachable: Cannot connect to the Kubernetes API server.

- **Error Example**:
  - kubectl commands fail with "Unable to connect to the server: connection refused."
- **Solution**:
  - Check the API server status with:

```
systemctl status kube-apiserver
```

  - Ensure firewall rules allow access to port 6443 and review kube-apiserver logs for errors.

---

15. Scheduler Failures: Pods are not being scheduled.

- **Error Example**:
    - Pods stuck in Pending status without apparent resource issues.
- **Solution**:
    - Check if the scheduler is running:

```
kubectl get pods -n kube-system | grep kube-scheduler
```

    - Review the scheduler logs and verify that pod anti-affinity rules aren't too restrictive.

---

16. Controller Manager Issues: Controllers aren't managing resources properly.

- **Error Example**:
    - Resources like Deployments or ReplicaSets not behaving as expected.
- **Solution**:
    - Check the controller manager pod's logs for errors and ensure it's running:

```
kubectl logs -n kube-system kube-controller-manager-[pod-name]
```

---

17. Network Plugin Errors: Issues with the network plugin can cause connectivity problems.

- **Error Example**:
    - Pods cannot communicate across nodes.
- **Solution**:
    - Check if the network plugin (e.g., Calico, Flannel) pods are running and review their logs.
    - Inspect kubectl get pods -n kube-system to ensure the network plugin's pods are up.

---

18. Pod Stuck in Terminating State: Pod doesn't terminate after issuing delete command.

- **Solution**:
    - Force delete the pod:

```
kubectl delete pod [pod-name] --grace-period=0 --force
```

    - Investigate why the pod failed to terminate, such as open connections or finalizers.

---

19. ConfigMap Not Found: Pod references a missing ConfigMap.

- **Error Example**:

Warning FailedMount … ConfigMap "myconfig" not found

- **Solution**:
    - Create the missing ConfigMap:

kubectl create configmap myconfig --from-literal=key=value

---

20. Secret Not Found: Pod references a missing Secret.

- **Error Example**:

Warning FailedMount … Secret "mysecret" not found

- **Solution**:
    - Create or update the secret:

kubectl create secret generic mysecret --from-literal=username=admin --from-literal=password=pass

---

21. HPA Not Scaling: Horizontal Pod Autoscaler is not scaling as expected.

- **Error Example**:
    - HPA status shows "Desired Replicas: 1" despite high load.
- **Solution**:
    - Ensure the metrics server is running:

kubectl get deployment metrics-server -n kube-system

    - Confirm HPA is targeting the correct metrics and configure thresholds if needed.

---

22. Ingress Not Working: Ingress does not route traffic as expected.

- **Error Example**:
    - Ingress setup is complete, but external requests fail.
- **Solution**:
    - Ensure an ingress controller (e.g., NGINX Ingress) is deployed.
    - Check ingress resource configuration:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
```

```
  name: example-ingress
spec:
  rules:
    - host: example.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: example-service
                port:
                  number: 80
```

---

23. DaemonSet Pods Not Running: DaemonSet pods do not start on all nodes.

- **Error Example**:

```
kubectl get daemonsets -A
```

Shows fewer than expected pods.

- **Solution**:
    - Review node taints/tolerations and add tolerations to the DaemonSet spec if needed.

---

24. Job Not Completing: Kubernetes Job fails to finish successfully.

- **Error Example**:
    - Job remains in "Running" or "Failed" status.
- **Solution**:
    - Check pod logs for errors and review job spec:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: my-job
spec:
  backoffLimit: 3
  template:
    spec:
      containers:
        - name: my-container
          image: busybox
          command: ["echo", "Hello World"]
      restartPolicy: Never
```

---

25. PVC Pending: PersistentVolumeClaim remains in "Pending" status.

- **Error Example**:
    - PVC status shows:

```
kubectl get pvc
```

```
NAME     STATUS    VOLUME    CAPACITY   ACCESS MODES   STORAGECLASS   AGE
my-pvc   Pending   <none>    <none>     <none>         standard       1h
```

- **Solution**:
    - Ensure there's a matching PersistentVolume with the correct access modes and storage class.

---

26. Node Disk Pressure: Node has high disk usage, causing evictions.

- **Error Example**:
    - Pods get evicted with DiskPressure status on the node.
- **Solution**:
    - Free up space on the node or add additional storage, as shown:

```
df -h
```

---

27. Pod Affinity/Anti-Affinity Issues: Pods are unscheduled due to restrictive affinity rules.

- **Solution**:
    - Ensure affinity rules are not too restrictive in the deployment YAML:

```
affinity:
  podAntiAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
            - key: app
              operator: In
              values:
                - myapp
        topologyKey: "kubernetes.io/hostname"
```

---

28. ServiceAccount Not Found: Pods reference a missing ServiceAccount.

- **Solution**:
    - Create or specify a valid ServiceAccount:

```
kubectl create serviceaccount myserviceaccount
```

29. Node NotSchedulable: Node is marked as unschedulable, preventing pods from being placed.

- **Error Example**:

```
kubectl get nodes
NAME        STATUS        ROLES              AGE   VERSION
worker-node   Ready,SchedulingDisabled   <none>   1h    v1.20.0
```

- **Solution**:
    - Enable scheduling on the node:

```
kubectl uncordon worker-node
```

---

30. Readiness Probe Failures: Containers fail readiness checks, causing them to stay in a "Not Ready" state.

- **Error Example**:
    - Describe the pod and see repeated readiness probe failures.
- **Solution**:
    - Adjust probe parameters to suit the application's startup time:

```
readinessProbe:
  httpGet:
    path: /health
    port: 8080
  initialDelaySeconds: 10
  periodSeconds: 5
```

---

31. Liveness Probe Failures: Containers fail liveness checks, resulting in restarts.

- **Solution**:
    - Similar to readiness probes, increase the delay and interval to allow the application more time to become live:

```
livenessProbe:
  httpGet:
    path: /live
    port: 8080
  initialDelaySeconds: 15
  periodSeconds: 10
```

---

32. Namespace Not Found: Resource references a non-existent namespace.

- **Error Example**:

Error from server (NotFound): namespaces "test-namespace" not found

- **Solution**:
    - Create the namespace before deploying resources:

```
kubectl create namespace test-namespace
```

---

33. ClusterRoleBinding Misconfiguration: Access issues due to incorrect ClusterRoleBinding setup.

- **Solution**:
    - Check and configure the ClusterRoleBinding correctly:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: example-binding
subjects:
  - kind: ServiceAccount
    name: example-sa
    namespace: default
roleRef:
  kind: ClusterRole
  name: view
  apiGroup: rbac.authorization.k8s.io
```

---

34. PVC Not Bound: PVC fails to bind to PV due to storage class mismatch.

- **Solution**:
    - Ensure the storageClassName matches between PVC and PV.

---

35. Node Memory Pressure: Nodes experience high memory pressure, causing evictions.

- **Solution**:
    - Check node memory usage with kubectl top nodes and consider scaling up nodes or reducing pod resource requests.

---

36. Service Endpoint Not Updated: Service does not update endpoints, resulting in unreachable services.

- **Solution**:
    - Ensure that the selector matches pod labels:

```
selector:
  app: myapp
```

---

37. Endpoint Slices Issues: Endpoint slices are missing, causing network issues.

- **Solution**:
    - Reconfigure or manually create endpoint slices if needed. Ensure kube-proxy is running and check its logs.

---

38. DaemonSet Not Deploying on All Nodes: DaemonSet skips certain nodes.

- **Solution**:
    - Check taints on nodes and add tolerations to the DaemonSet:

```
tolerations:
  - key: "key"
    operator: "Exists"
```

---

39. Finalizer Preventing Resource Deletion: Resource remains due to finalizers.

- **Solution**:
    - Remove finalizers to allow deletion:

```
kubectl patch resource resource-name -p '{"metadata":{"finalizers":[]}}' --type=merge
```

---

40. Ingress 404 Errors: Requests to Ingress return 404.

- **Solution**:
    - Ensure correct path definitions and verify that the Ingress controller is properly configured.

---

41. LoadBalancer IP Not Assigned: LoadBalancer service fails to get an external IP.

- **Solution**:
    - Verify cloud provider configuration or use a different service type like NodePort for testing.

---

42. HPA Targets Not Matching Metrics: HPA doesn't scale as it's not receiving target metrics.

- **Solution**:
    - Check the HPA target settings and ensure metrics are available via metrics server or Prometheus.

---

43. PersistentVolume Deleted but PVC Bound: PVC remains bound even though PV was deleted.

- **Solution**:
    - Manually unbind and delete PVC or recreate PV with the same name to bind back.

---

44. Helm Release Fails: Helm fails due to missing charts or resources.

- **Solution**:
    - Ensure Helm chart dependencies are installed using:

helm dependency update mychart/

---

45. API Version Deprecated: Using outdated API versions causes compatibility issues.

- **Solution**:
    - Update your YAML files to use the latest API versions (e.g., apps/v1 instead of extensions/v1beta1 for Deployments).

---

46. Namespace Resource Quota Exceeded: Deployments fail due to resource quota limits in a namespace.

- **Solution**:
    - Check quota with:

kubectl describe quota -n [namespace]

    - Adjust resources or increase the quota if needed.

---

47. Cannot Attach Volume to Multiple Pods: Persistent volumes with RWO access can't be shared across pods.

- **Solution**:
    - Switch to a storage class supporting ReadWriteMany (RWX), like NFS, for shared access.

---

48. CPU Throttling: Containers experience high CPU throttling.

- **Solution**:
    - Increase the CPU limit or optimize application code to avoid exceeding CPU quotas:

```
resources:
  limits:
    cpu: "500m"
  requests:
    cpu: "250m"
```

---

49. Pods Evicted Due to Overcommit: Overcommitted resources lead to evictions.

- **Solution**:
    - Allocate resources more conservatively or scale cluster resources accordingly.

---

50. PodSecurityPolicy Issues: Pods fail to start due to restrictive PodSecurityPolicy.

- **Solution**:
    - Adjust the PodSecurityPolicy to allow necessary permissions:

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: example-psp
spec:
  privileged: true
```