🛠️ Lab 7 – Observability & Troubleshooting (CKA Focus)

🎯 Objectives

- Practice log inspection, probes, and metrics
- Troubleshoot common pod and cluster component issues
- Align with CKA expectations for observability (~30% exam weight)

⚙️ Prerequisites

- A running **Kind cluster**
- `kubectl` and `metrics-server` installed

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
kubectl patch deployment metrics-server -n kube-system \
  --type=json -p='[{"op":"add","path":"/spec/template/spec/containers/0/args/-","value":"--kubelet-insecure-tls"}]'
```

🔍 Step 1 – Investigate Logs & Pod Failures

1. Create a faulty deployment:

```
kubectl create deployment badapp --image=nginx:1.25
kubectl set image deployment/badapp nginx=nonexistent:latest
```

Edixos Labs
hello@edixos.com
+33 6 47 43 99 30

2. Inspect the pod:

```
kubectl get pods
kubectl describe pod <badapp-pod-name>
kubectl logs <badapp-pod-name>
```

Understand why the container fails to start.

Here are common pod error types you might encounter, this table summarizes them:

Edixos Labs
hello@edixos.com
+33 6 47 43 99 30

Pod Error Type	Error Description
`ErrImagePull`	If kubernetes is not able to pull the image mentioned in the manifest.
`ErrImagePullBackOff`	Container image pull failed, kubelet is backing off image pull
`ErrInvalidImageName`	Indicates a wrong image name.
`ErrImageInspect`	Unable to inspect the image.
`ErrImageNeverPull`	Specified Image is absent on the node and PullPolicy is set to NeverPullImage
`ErrRegistryUnavailable`	HTTP error when trying to connect to the registry
`ErrContainerNotFound`	The specified container is either not present or not managed by the kubelet, within the declared pod.
`ErrRunInitContainer`	Container initialization failed.
`ErrRunContainer`	Pod's containers don't start successfully due to misconfiguration.
`ErrKillContainer`	None of the pod's containers were killed successfully.
`ErrCrashLoopBackOff`	A container has terminated. The kubelet will not attempt to restart it.
`ErrVerifyNonRoot`	A container or image attempted to run with root privileges.
`ErrCreatePodSandbox`	Pod sandbox creation did not succeed.
`ErrConfigPodSandbox`	Pod sandbox configuration was not obtained.
`ErrKillPodSandbox`	A pod sandbox did not stop successfully.
`ErrSetupNetwork`	Network initialization failed.
`ErrTeardownNetwork`	Network teardown failed.

🩺 Step 2 – Liveness & Readiness Probes

Edixos Labs
hello@edixos.com
+33 6 47 43 99 30

Apply the following manifest:

```
apiVersion: v1
kind: Pod
metadata:
  name: probe-pod
spec:
  containers:
  - name: nginx
    image: nginx
    livenessProbe:
      httpGet:
        path: /doesnotexist
        port: 80
      initialDelaySeconds: 5
      periodSeconds: 10
    readinessProbe:
      httpGet:
        path: /ready
        port: 80
      initialDelaySeconds: 3
      periodSeconds: 5
```

Then:

```
kubectl apply -f module-7/manifests/probe-pod.yaml
kubectl describe pod probe-pod
kubectl get events
```

Observe how misconfigured probes affect pod health.

📊 Step 3 – Monitor Metrics

1. Create a pod that generates CPU load:

Edixos Labs
hello@edixos.com
+33 6 47 43 99 30

```
kubectl create deployment cpu-burner --image=busybox -- slee
p 3600
kubectl exec -it $(kubectl get pod -l app=cpu-burner -o nam
e) -- sh -c "while true; do yes > /dev/null; done"
```

2. Monitor usage:

```
kubectl top nodes
kubectl top pods
```

🧩 Final Challenge – Custom Logging & Resource Issue

🔸 **Goal:** Create a pod called `logger-challenge` that logs to a file instead of stdout.

🔸 **Your Tasks:**

- Configure a sidecar container to tail the log file and print to stdout
- Ensure logs from the sidecar appear with `kubectl logs logger-challenge -c sidecar`

Use the Kubernetes documentation to complete this challenge.

✅ End of Lab 7 – You've practiced key CKA troubleshooting and observability scenarios, including logs, probes, metrics, and core component failure!

Edixos Labs
hello@edixos.com
+33 6 47 43 99 30