Lesson 03 Demo 13

Configuring Pods Using Liveness Probes

Objective: To create and configure a pod using liveness probes to ensure the stability and reliability of the applications running inside the pod

Tools required: kubeadm, kubectl, kubelet, and containerd

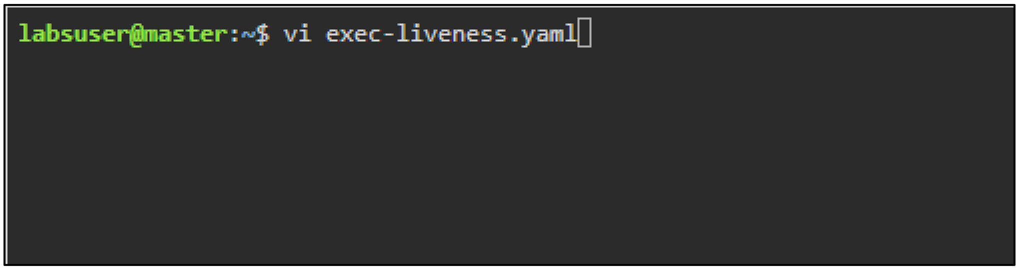
Prerequisites: A Kubernetes cluster (refer to Demo 01 from Lesson 01 for setting up a cluster)

Steps to be followed:

1. Create a pod using liveness probes
2. Describe the pod

Step 1: Create a pod using liveness probes

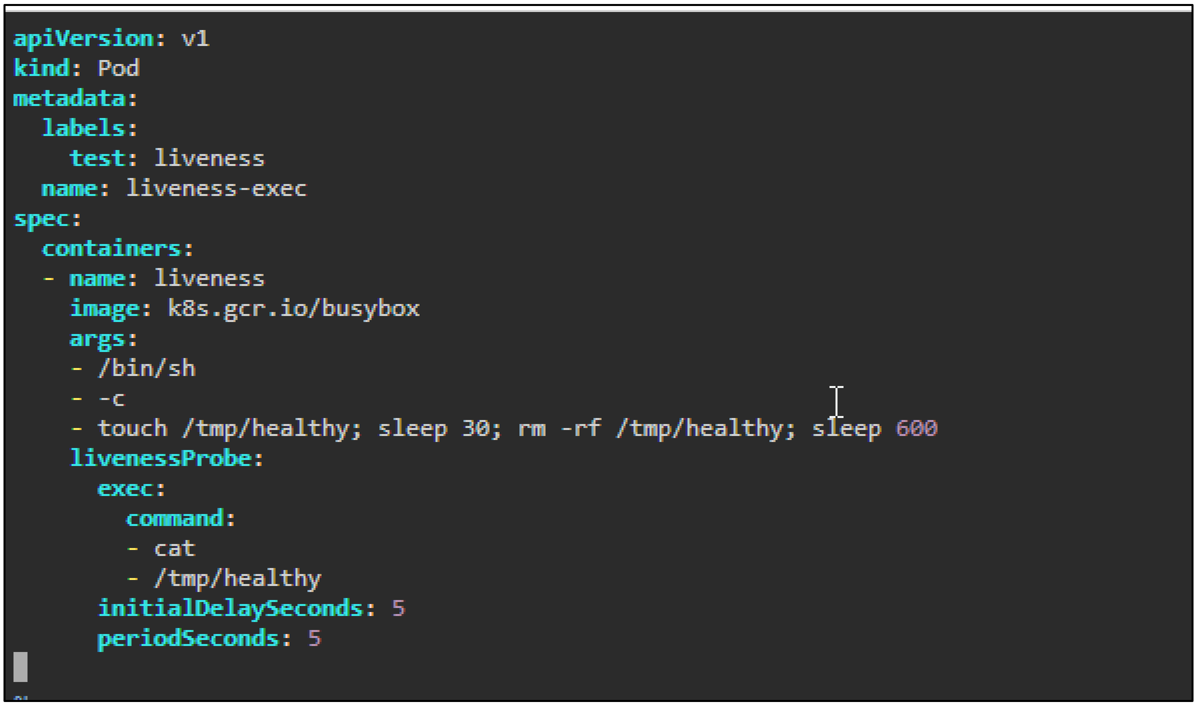
1.1 On the master node, enter the command **vi exec-liveness.yaml** to create a YAML file



```
labsuser@master:~$ vi exec-liveness.yaml
```

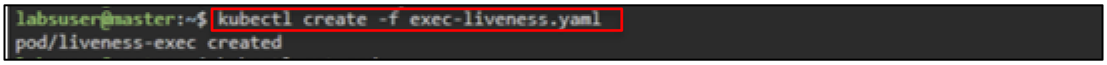
1.2 Add the following code to the YAML file:

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    test: liveness
  name: liveness-exec
spec:
  containers:
  - name: liveness
    image: k8s.gcr.io/busybox
    args:
    - /bin/sh
    - -c
    - touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600
    livenessProbe:
      exec:
        command:
        - cat
        - /tmp/healthy
      initialDelaySeconds: 5
      periodSeconds: 5
```



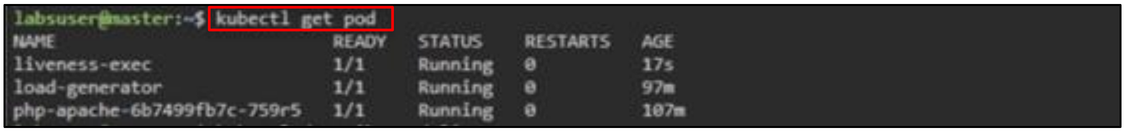
```
apiVersion: v1
kind: Pod
metadata:
  labels:
    test: liveness
  name: liveness-exec
spec:
  containers:
  - name: liveness
    image: k8s.gcr.io/busybox
    args:
    - /bin/sh
    - -c
    - touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600
    livenessProbe:
      exec:
        command:
        - cat
        - /tmp/healthy
      initialDelaySeconds: 5
      periodSeconds: 5
```

1.3 Create a pod using the command below:
kubectl create -f exec-liveness.yaml



```
labsuser@master:~$ kubectl create -f exec-liveness.yaml
pod/liveness-exec created
```

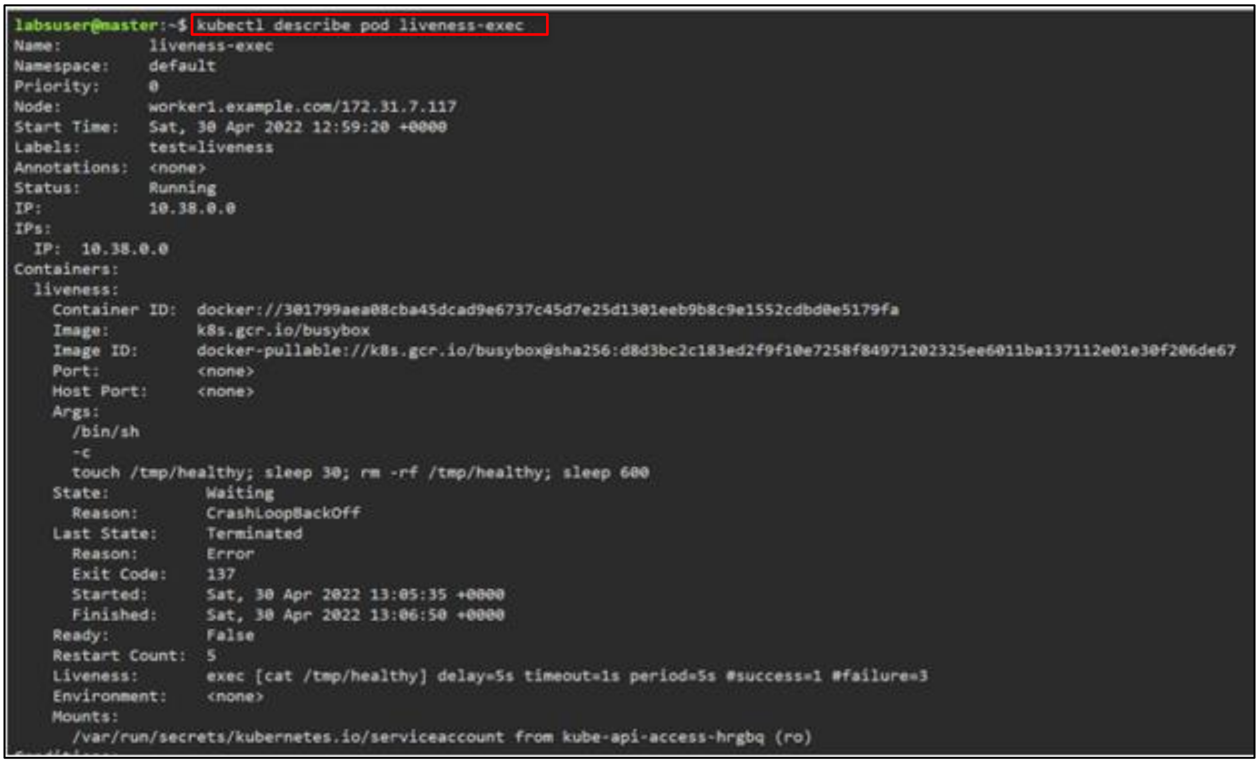
1.4 Enter the following command to get the pod status:
kubectl get pod



```
labsuser@master:~$ kubectl get pod
NAME                          READY   STATUS    RESTARTS   AGE
liveness-exec                 1/1     Running   0          17s
load-generator                1/1     Running   0          97m
php-apache-6b7499fb7c-759r5   1/1     Running   0          107m
```

Step 2: Describe the pod

2.1 Describe the pod using the following command:
kubectl describe pod liveness-exec



```
labsuser@master:~$ kubectl describe pod liveness-exec
Name:         liveness-exec
Namespace:    default
Priority:     0
Node:         worker1.example.com/172.31.7.117
Start Time:   Sat, 30 Apr 2022 12:59:20 +0000
Labels:       test=liveness
Annotations:  <none>
Status:       Running
IP:           10.38.0.0
IPs:
  IP:  10.38.0.0
Containers:
  liveness:
    Container ID:  docker://301799aea08cba45dcad9e6737c45d7e25d1301eeb9b8c9e1552cdbd0e5179fa
    Image:         k8s.gcr.io/busybox
    Image ID:      docker-pullable://k8s.gcr.io/busybox@sha256:d8d3bc2c183ed2f9f10e7258f84971202325ee6011ba137112e01e30f206de67
    Port:          <none>
    Host Port:     <none>
    Args:
      /bin/sh
      -c
      touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600
    State:          Waiting
      Reason:       CrashLoopBackOff
    Last State:     Terminated
      Reason:       Error
      Exit Code:    137
      Started:      Sat, 30 Apr 2022 13:05:35 +0000
      Finished:     Sat, 30 Apr 2022 13:06:50 +0000
    Ready:          False
    Restart Count:  5
    Liveness:       exec [cat /tmp/healthy] delay=5s timeout=1s period=5s #success=1 #failure=3
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-hrgbq (ro)
```

By following these steps, you have successfully created and configured a pod using liveness probes.