Lesson 09 Demo 01

Troubleshooting Kubernetes Cluster

Objective: To troubleshoot Kubernetes clusters by utilizing diagnostic commands

Tools required: kubeadm, kubectl, kubelet, and containerd

Prerequisites: A Kubernetes cluster should already be set up (refer to the steps provided in Lesson 02, Demo 01 for guidance).

Steps to be followed:

1. Troubleshoot using dumps

Step 1: Troubleshoot using dumps

1.1 Use the following command to view the nodes in your cluster:

kubectl get nodes

```
labsuser@master:~$ kubectl get nodes
NAME                       STATUS   ROLES           AGE    VERSION
master.example.com         Ready    control-plane   4d1h   v1.28.2
worker-node-1.example.com  Ready    <none>          4d1h   v1.28.2
worker-node-2.example.com  Ready    <none>          4d1h   v1.28.2
labsuser@master:~$
```

1.2 To fetch the overall cluster information, use the following command:

kubectl cluster-info

```
labsuser@master:~$ kubectl cluster-info
Kubernetes control plane is running at https://172.31.9.176:6443
CoreDNS is running at https://172.31.9.176:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
labsuser@master:~$
```

1.3 Obtain the cluster dump information with the following command:

kubectl cluster-info dump

```
}
==== START logs for container ngnix of pod default/admin-56d684dff9-zjfhc ====
Request log error: the server rejected our request for an unknown reason (get pods admin-56d684dff9-zjfhc)
==== END logs for container ngnix of pod default/admin-56d684dff9-zjfhc ====
==== START logs for container nginx of pod default/pod-demo ====
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/10/10 15:31:21 [notice] 1#1: using the "epoll" event method
2023/10/10 15:31:21 [notice] 1#1: nginx/1.25.2
2023/10/10 15:31:21 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/10/10 15:31:21 [notice] 1#1: OS: Linux 6.2.0-1013-aws
2023/10/10 15:31:21 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1024:524288
2023/10/10 15:31:21 [notice] 1#1: start worker processes
2023/10/10 15:31:21 [notice] 1#1: start worker process 29
2023/10/10 15:31:21 [notice] 1#1: start worker process 30
==== END logs for container nginx of pod default/pod-demo ====
labsuser@master:~$ kubectl cluster-info dump
```

1.4 To understand the various options available for the dump command and access its help documentation, use the following command:
kubectl cluster-info dump –help

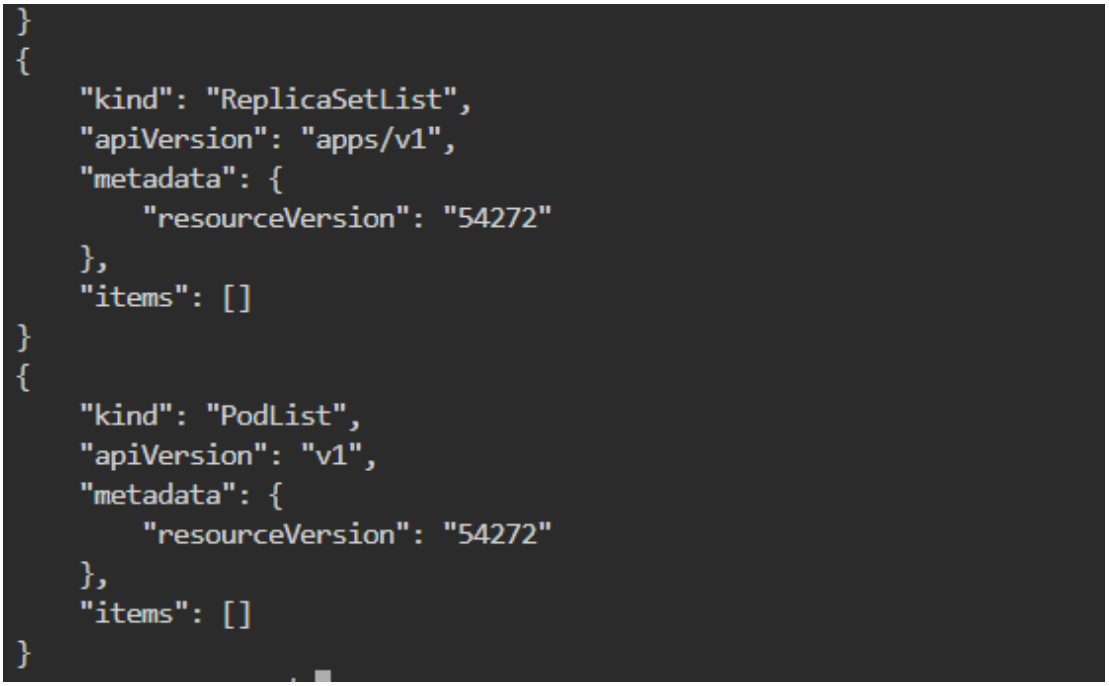
```
labsuser@master:~$ kubectl cluster-info dump -help
```

```
==== START logs for container ngnix of pod default/admin-56d684dff9-zjfhc ====
Request log error: the server rejected our request for an unknown reason (get pods admin-56d684dff9-zjfhc)
==== END logs for container ngnix of pod default/admin-56d684dff9-zjfhc ====
==== START logs for container nginx of pod default/pod-demo ====
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/10/10 15:31:21 [notice] 1#1: using the "epoll" event method
2023/10/10 15:31:21 [notice] 1#1: nginx/1.25.2
2023/10/10 15:31:21 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/10/10 15:31:21 [notice] 1#1: OS: Linux 6.2.0-1013-aws
2023/10/10 15:31:21 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1024:524288
2023/10/10 15:31:21 [notice] 1#1: start worker processes
2023/10/10 15:31:21 [notice] 1#1: start worker process 29
2023/10/10 15:31:21 [notice] 1#1: start worker process 30
==== END logs for container nginx of pod default/pod-demo ====
labsuser@master:~$
```

1.5 Enter the command to provide a dump on a specific namespace:
kubectl cluster-info -n <namespace> dump

For instance, if checking for the **test** namespace, use:
kubectl cluster-info -n test dump

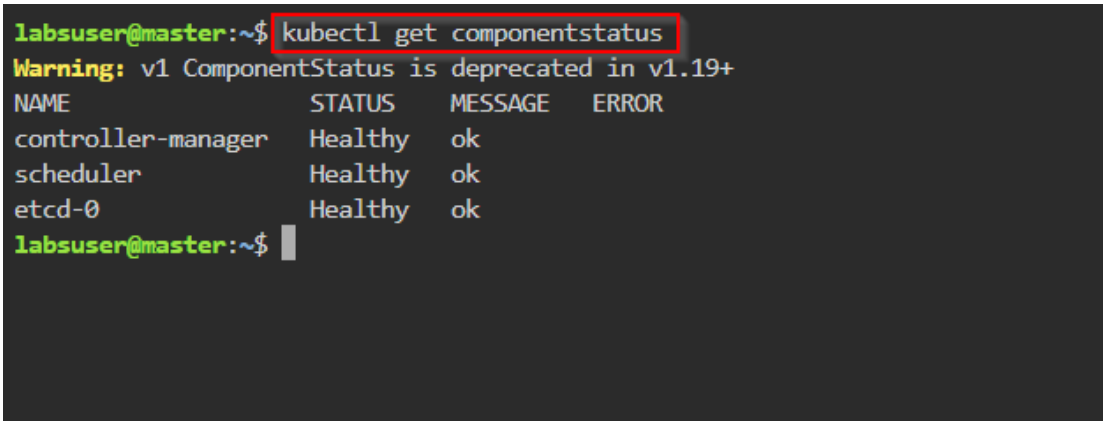
```
labsuser@master:~$ kubectl cluster-info -n test dump
```



Note: Replace the **<namespace>** in the command with the namespace for which you need to get the cluster-info dump. For example: **kubectl cluster-info -n test dump**

1.6 Check the health of your cluster components using the following command:

kubectl get componentstatus



By following these steps, you have successfully explored key commands for Kubernetes troubleshooting, enabling users to effectively diagnose cluster health and configurations.