Troubleshooting a Broken Kubernetes Cluster

Your company, BeeBox, has a new Kubernetes cluster that was just built by an outside contractor. Last night, someone restarted the servers used to run this cluster. Ever since the restart, some of your team members are reporting issues with one of the worker nodes. Unfortunately, the contractor is no longer working with the company, so you will need to find and fix the problem.

Explore the cluster and determine what is causing the issues. Then take steps to fix the problem and ensure that it does not happen again.

Reading above we need to determine what is wrong with the cluster, and why the sendond node is not functioning as expected. The of course we fix the problem and get the worker node back up and running.

Determine What is Wrong with the Cluster

1. Lets start off by looking at the nodes

Figure 1-1

```
cloud_user@k8s-control:~$ kubectl get nodes
NAME           STATUS     ROLES           AGE   VERSION
k8s-control    Ready      control-plane   19m   v1.24.0
k8s-worker1    Ready      <none>          19m   v1.24.0
k8s-worker2    NotReady   <none>          19m   v1.24.0
```

Above we can see that the worker2 node isn't set to ready. Looks like the others are just working fine. So we can focus on worker2.

2. Lets do a describe command for worker2

Doing the kubectl describe node k8s-worker2 command. I can see several things. First thing that caught my eye was at the bottom for events.

"Node K8s-worker2 status is now: NodeHasSufficentMemory"

I also see that going up some under "Conditions" I see several types of conditions that says "Kubelete stopped posting node status". This means the control plane server is not getting its regular updates from kubelet. Kubelet is not reaching out successfully to the control plane server. So lets log into that node and see what is going on.

3. So connect and log onto the worker2 node
4. Lets check out the kubelet logs

Figure 1-2

```
Nov 27 15:13:46 k8s-worker2 systemd[1]: kubelet.service: Succeeded.
Nov 27 15:13:46 k8s-worker2 systemd[1]: Stopped kubelet: The Kubernetes Node Agent.
```

Turns out kubelet is stopped. You can press Shift+G to go to the bottom to see the last logs

5. So lets check the status even tho it says it is stopped

sudo systemctl status kubelet

Figure 1-3

```
cloud_user@k8s-worker2:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/lib/systemd/system/kubelet.service; disabled; vendor preset: enabled)
    Drop-In: /etc/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: inactive (dead)
       Docs: https://kubernetes.io/docs/home/
```

So turns out it is disabled and in the inactive (dead) state. So what may have happened was the server may have restarted the kubelet service, because it's disabled, just didn't start up again after the restart. So all we need to do is start the service to continue to work.

When the server restarts in the future we need to enable kubelet in addition to just starting it.

Fix the Problem

6. Enable kubelet and start kubelet

sudo systemctl enable kubelet

sudo systemctl start kubelet

7. After issuing those 2 commands it should be up and running, so lets verify again.

It took a few for the status to come up active and enabled

Figure 1-4

```
cloud_user@k8s-worker2:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
    Drop-In: /etc/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Sun 2022-11-27 15:55:07 UTC; 3s ago
       Docs: https://kubernetes.io/docs/home/
   Main PID: 3040 (kubelet)
      Tasks: 14 (limit: 4623)
     Memory: 27.2M
     CGroup: /system.slice/kubelet.service
             └─3040 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --
```

So it should be good now, lets sign back into our control node.

8. Repeat the same command we did on step 1

Figure 1-5

```
cloud_user@k8s-control:~$ kubectl get nodes
NAME          STATUS   ROLES           AGE   VERSION
k8s-control   Ready    control-plane   44m   v1.24.0
k8s-worker1   Ready    <none>          43m   v1.24.0
k8s-worker2   Ready    <none>          43m   v1.24.0
```

So we have the worker2 node now in the ready status. We have completed the lab