Lesson 03 Demo 02

Understanding the Working of Nodes

Objective: To understand the operations and management of nodes in a Kubernetes cluster

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).

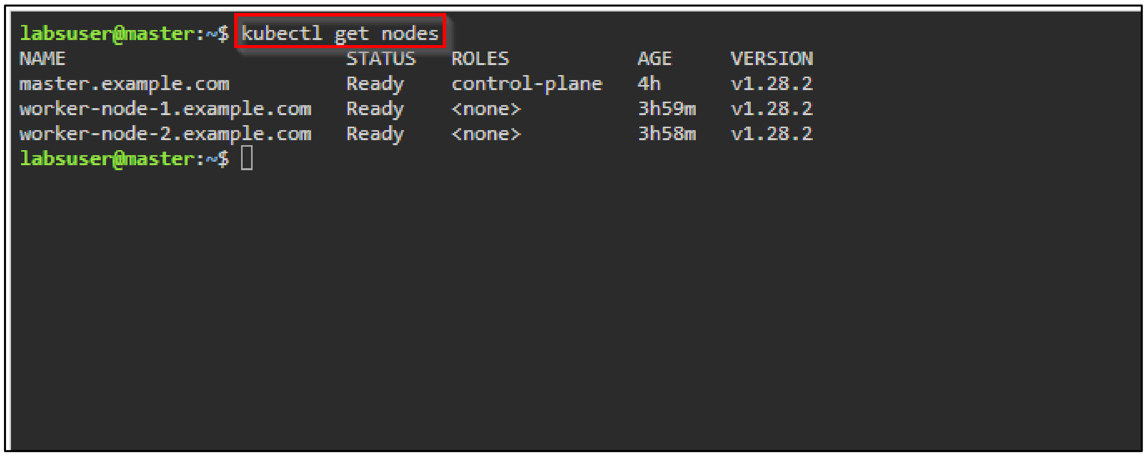
Steps to be followed:

1. Verify the status of a node
2. Delete a worker node
3. Register a worker node using a config file
4. Identify the node conditions, capacity and allocatable resources

Step 1: Verify the status of a node

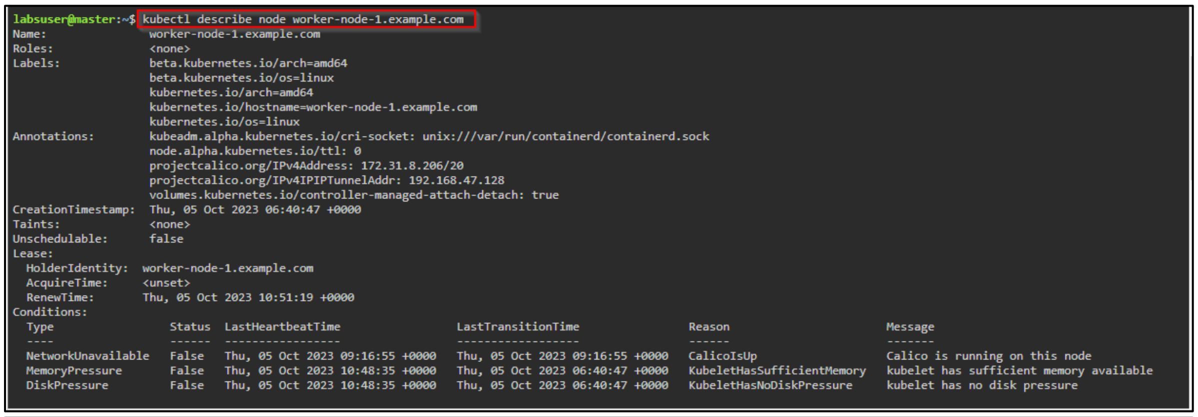
1.1 List all the running nodes in a cluster using the following command:

kubectl get nodes

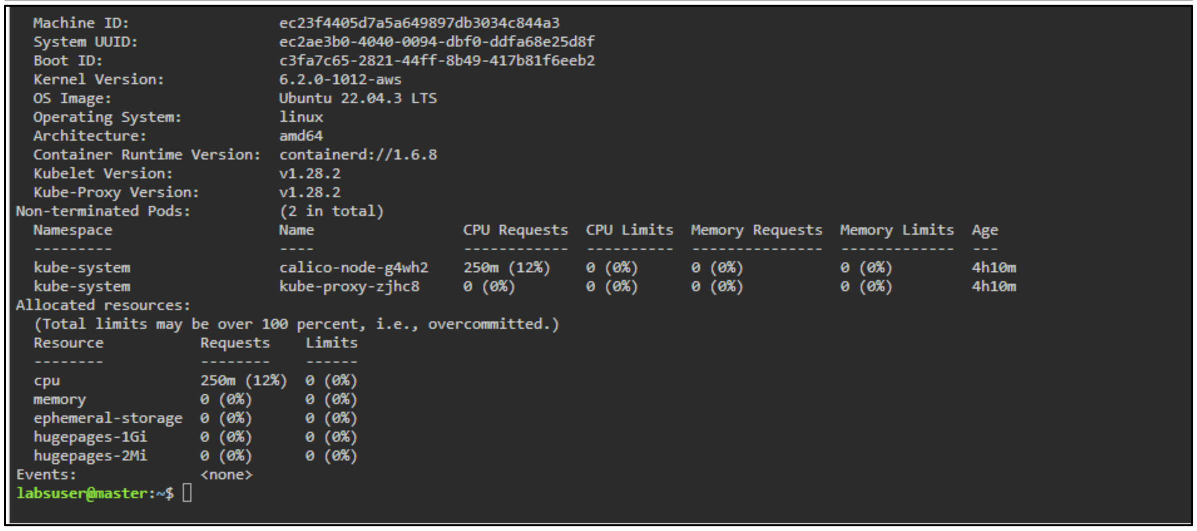


```
labsuser@master:~$ kubectl get nodes
NAME                         STATUS   ROLES           AGE     VERSION
master.example.com           Ready    control-plane   4h      v1.28.2
worker-node-1.example.com    Ready    <none>          3h59m   v1.28.2
worker-node-2.example.com    Ready    <none>          3h58m   v1.28.2
labsuser@master:~$ 
```

1.2 Verify the status of the worker node you wish to inspect by running the following command:
kubectl describe node worker-node-1.example.com



```
labsuser@master:~$ kubectl describe node worker-node-1.example.com
Name:               worker-node-1.example.com
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=worker-node-1.example.com
                    kubernetes.io/os=linux
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: unix:///var/run/containerd/containerd.sock
                    node.alpha.kubernetes.io/ttl: 0
                    projectcalico.org/IPv4Address: 172.31.8.206/20
                    projectcalico.org/IPv4IPIPTunnelAddr: 192.168.47.128
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Thu, 05 Oct 2023 06:40:47 +0000
Taints:             <none>
Unschedulable:      false
Lease:
  HolderIdentity:  worker-node-1.example.com
  AcquireTime:     <unset>
  RenewTime:       Thu, 05 Oct 2023 10:51:19 +0000
Conditions:
  Type                 Status  LastHeartbeatTime                 LastTransitionTime                Reason                       Message
  ----                 ------  -----------------                 ------------------                ------                       -------
  NetworkUnavailable   False   Thu, 05 Oct 2023 09:16:55 +0000   Thu, 05 Oct 2023 09:16:55 +0000   CalicoIsUp                   Calico is running on this node
  MemoryPressure       False   Thu, 05 Oct 2023 10:48:35 +0000   Thu, 05 Oct 2023 06:40:47 +0000   KubeletHasSufficientMemory   kubelet has sufficient memory available
  DiskPressure         False   Thu, 05 Oct 2023 10:48:35 +0000   Thu, 05 Oct 2023 06:40:47 +0000   KubeletHasNoDiskPressure     kubelet has no disk pressure
```

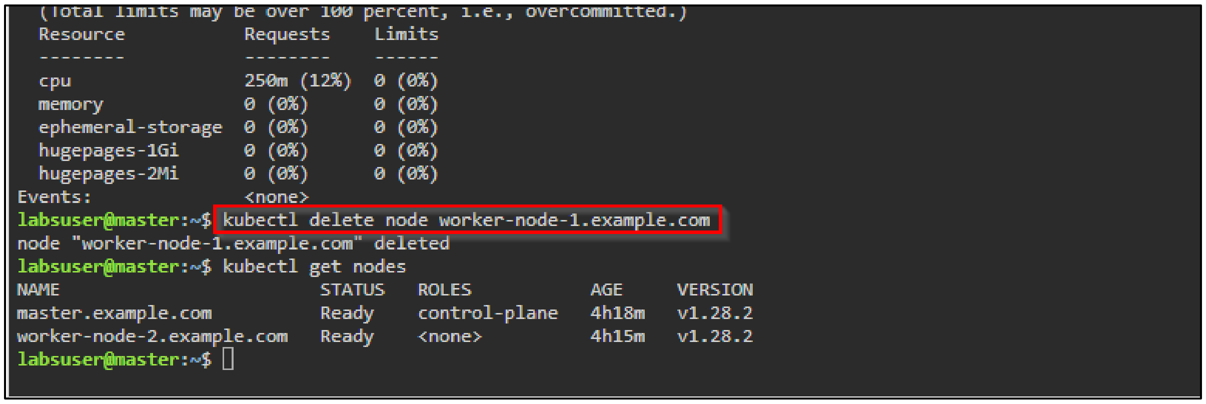


```
  Machine ID:                 ec23f4405d7a5a649897db3034c844a3
  System UUID:                ec2ae3b0-4040-0094-dbf0-ddfa68e25d8f
  Boot ID:                    c3fa7c65-2821-44ff-8b49-417b81f6eeb2
  Kernel Version:             6.2.0-1012-aws
  OS Image:                   Ubuntu 22.04.3 LTS
  Operating System:           linux
  Architecture:               amd64
  Container Runtime Version:  containerd://1.6.8
  Kubelet Version:            v1.28.2
  Kube-Proxy Version:         v1.28.2
Non-terminated Pods:          (2 in total)
  Namespace                   Name                 CPU Requests  CPU Limits  Memory Requests  Memory Limits  Age
  ---------                   ----                 ------------  ----------  ---------------  -------------  ---
  kube-system                 calico-node-g4wh2    250m (12%)    0 (0%)      0 (0%)           0 (0%)         4h10m
  kube-system                 kube-proxy-zjhc8     0 (0%)        0 (0%)      0 (0%)           0 (0%)         4h10m
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests    Limits
  --------           --------    ------
  cpu                250m (12%)  0 (0%)
  memory             0 (0%)      0 (0%)
  ephemeral-storage  0 (0%)      0 (0%)
  hugepages-1Gi      0 (0%)      0 (0%)
  hugepages-2Mi      0 (0%)      0 (0%)
Events:              <none>
labsuser@master:~$ 
```

.

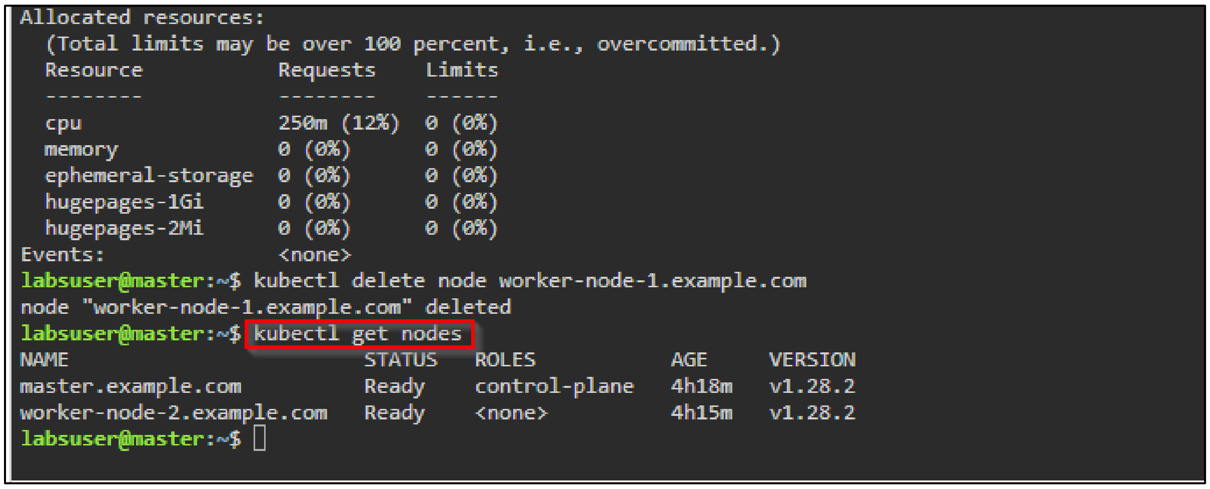
Step 2: Delete a worker node

2.1 Use the following command to delete a worker node:
kubectl delete node worker-node-1.example.com



```
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests    Limits
  --------           --------    ------
  cpu                250m (12%)  0 (0%)
  memory             0 (0%)      0 (0%)
  ephemeral-storage  0 (0%)      0 (0%)
  hugepages-1Gi      0 (0%)      0 (0%)
  hugepages-2Mi      0 (0%)      0 (0%)
Events:              <none>
labsuser@master:~$ kubectl delete node worker-node-1.example.com
node "worker-node-1.example.com" deleted
labsuser@master:~$ kubectl get nodes
NAME                        STATUS   ROLES           AGE     VERSION
master.example.com          Ready    control-plane   4h18m   v1.28.2
worker-node-2.example.com   Ready    <none>          4h15m   v1.28.2
labsuser@master:~$ 
```

2.2 Fetch the list of nodes in the cluster:
kubectl get nodes

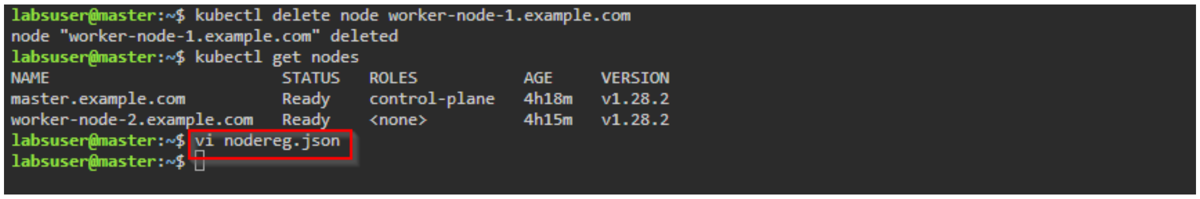


```
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests    Limits
  --------           --------    ------
  cpu                250m (12%)  0 (0%)
  memory             0 (0%)      0 (0%)
  ephemeral-storage  0 (0%)      0 (0%)
  hugepages-1Gi      0 (0%)      0 (0%)
  hugepages-2Mi      0 (0%)      0 (0%)
Events:              <none>
labsuser@master:~$ kubectl delete node worker-node-1.example.com
node "worker-node-1.example.com" deleted
labsuser@master:~$ kubectl get nodes
NAME                        STATUS   ROLES           AGE     VERSION
master.example.com          Ready    control-plane   4h18m   v1.28.2
worker-node-2.example.com   Ready    <none>          4h15m   v1.28.2
labsuser@master:~$ 
```

Step 3: Register a worker node using a config file

3.1 Create a file named **nodereg.json**

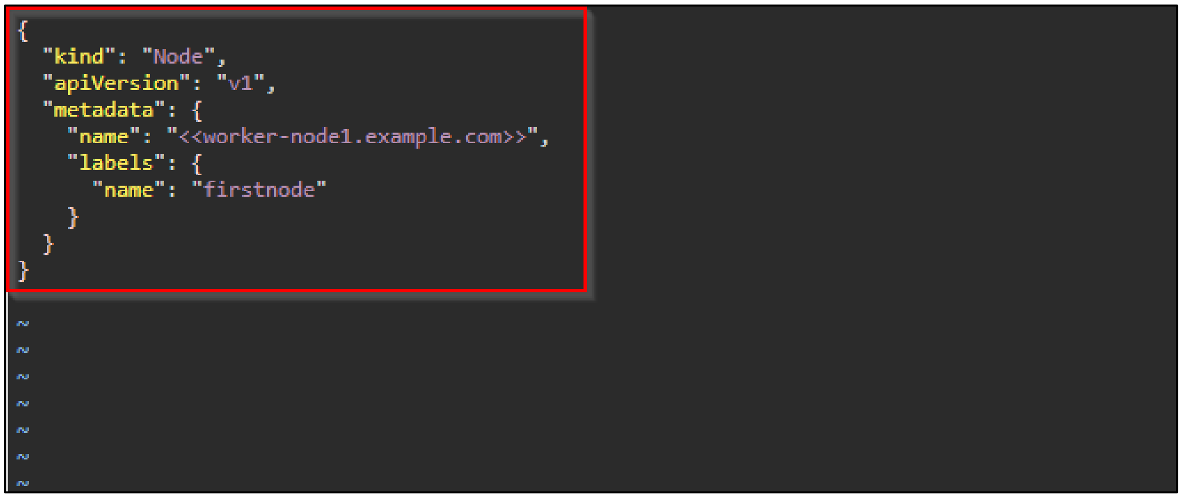
vi nodereg.json



```
labsuser@master:~$ kubectl delete node worker-node-1.example.com
node "worker-node-1.example.com" deleted
labsuser@master:~$ kubectl get nodes
NAME                        STATUS   ROLES           AGE     VERSION
master.example.com          Ready    control-plane   4h18m   v1.28.2
worker-node-2.example.com   Ready    <none>          4h15m   v1.28.2
labsuser@master:~$ vi nodereg.json
labsuser@master:~$ 
```

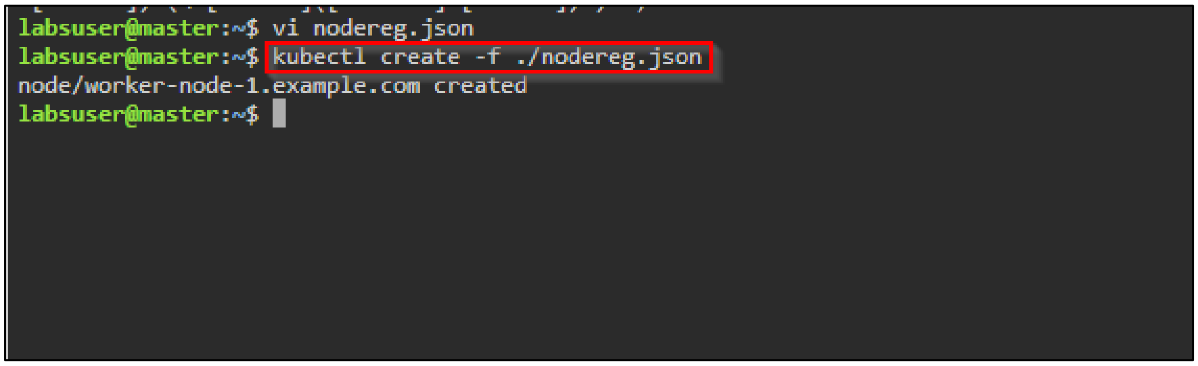
3.2 Now, inside the **nodereg.json** file, input the following JSON code:

```
{
 "kind": "Node",
 "apiVersion": "v1",
 "metadata": {
  "name": "worker-node-1.example.com",
  "labels": {
   "name": "firstnode"
  }
 }
}
```



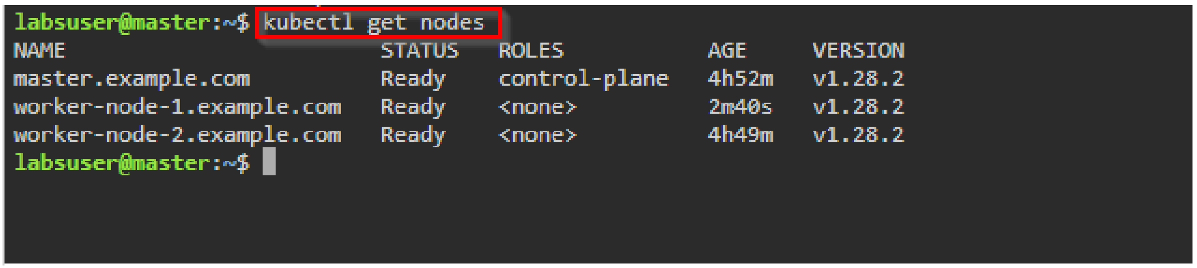
```
{
  "kind": "Node",
  "apiVersion": "v1",
  "metadata": {
    "name": "<<worker-node1.example.com>>",
    "labels": {
      "name": "firstnode"
    }
  }
}
~
~
~
~
~
~
~
```

3.3 Run the following command to register the node using the **nodereg.json** file:
kubectl create -f ./nodereg.json



```
labsuser@master:~$ vi nodereg.json
labsuser@master:~$ kubectl create -f ./nodereg.json
node/worker-node-1.example.com created
labsuser@master:~$ 
```

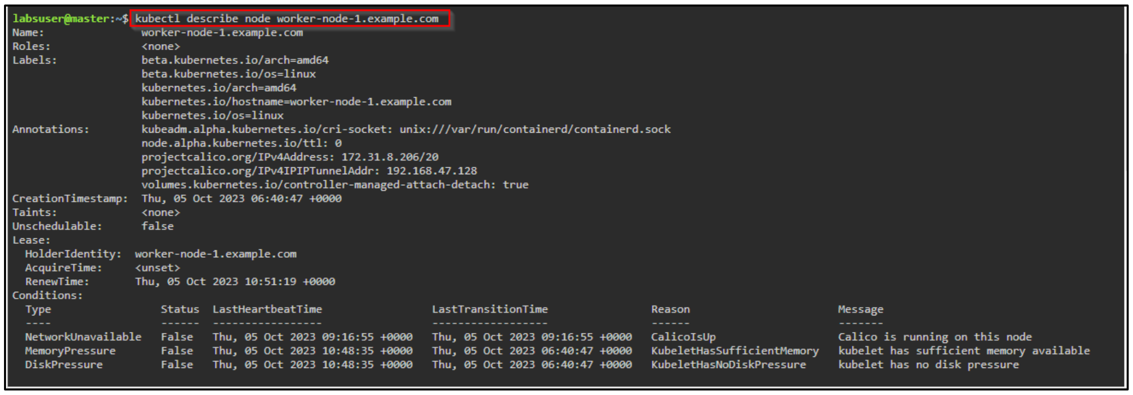
3.4 Execute the following command to verify the created node:
kubectl get nodes



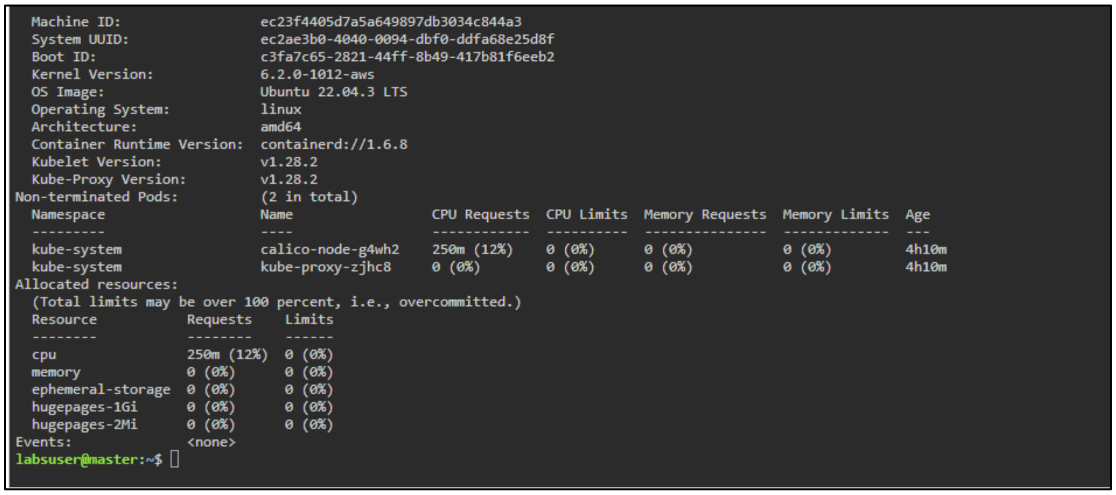
```
labsuser@master:~$ kubectl get nodes
NAME                         STATUS   ROLES           AGE     VERSION
master.example.com           Ready    control-plane   4h52m   v1.28.2
worker-node-1.example.com    Ready    <none>          2m40s   v1.28.2
worker-node-2.example.com    Ready    <none>          4h49m   v1.28.2
labsuser@master:~$ 
```

Step 4: Identify the node conditions, capacity and allocatable resources

4.1 To View the 4 node conditions and status, capacity and allocatable size of each resources use the following command:
kubectl describe node worker-node-1.example.com



```
labsuser@master:~$ kubectl describe node worker-node-1.example.com
Name:               worker-node-1.example.com
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=worker-node-1.example.com
                    kubernetes.io/os=linux
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: unix:///var/run/containerd/containerd.sock
                    node.alpha.kubernetes.io/ttl: 0
                    projectcalico.org/IPv4Address: 172.31.8.206/20
                    projectcalico.org/IPv4IPIPTunnelAddr: 192.168.47.128
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Thu, 05 Oct 2023 06:40:47 +0000
Taints:             <none>
Unschedulable:      false
Lease:
  HolderIdentity:  worker-node-1.example.com
  AcquireTime:     <unset>
  RenewTime:       Thu, 05 Oct 2023 10:51:19 +0000
Conditions:
  Type                 Status  LastHeartbeatTime                 LastTransitionTime                Reason                       Message
  ----                 ------  -----------------                 ------------------                ------                       -------
  NetworkUnavailable   False   Thu, 05 Oct 2023 09:16:55 +0000   Thu, 05 Oct 2023 09:16:55 +0000   CalicoIsUp                   Calico is running on this node
  MemoryPressure       False   Thu, 05 Oct 2023 10:48:35 +0000   Thu, 05 Oct 2023 06:40:47 +0000   KubeletHasSufficientMemory   kubelet has sufficient memory available
  DiskPressure         False   Thu, 05 Oct 2023 10:48:35 +0000   Thu, 05 Oct 2023 06:40:47 +0000   KubeletHasNoDiskPressure     kubelet has no disk pressure
```


```
  Machine ID:                 ec23f4405d7a5a649897db3034c844a3
  System UUID:                ec2ae3b0-4040-0094-dbf0-ddfa68e25d8f
  Boot ID:                    c3fa7c65-2821-44ff-8b49-417b81f6eeb2
  Kernel Version:             6.2.0-1012-aws
  OS Image:                   Ubuntu 22.04.3 LTS
  Operating System:           linux
  Architecture:               amd64
  Container Runtime Version:  containerd://1.6.8
  Kubelet Version:            v1.28.2
  Kube-Proxy Version:         v1.28.2
Non-terminated Pods:          (2 in total)
  Namespace                   Name                 CPU Requests  CPU Limits  Memory Requests  Memory Limits  Age
  ---------                   ----                 ------------  ----------  ---------------  -------------  ---
  kube-system                 calico-node-g4wh2    250m (12%)    0 (0%)      0 (0%)           0 (0%)         4h10m
  kube-system                 kube-proxy-zjhc8     0 (0%)        0 (0%)      0 (0%)           0 (0%)         4h10m
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests    Limits
  --------           --------    ------
  cpu                250m (12%)  0 (0%)
  memory             0 (0%)      0 (0%)
  ephemeral-storage  0 (0%)      0 (0%)
  hugepages-1Gi      0 (0%)      0 (0%)
  hugepages-2Mi      0 (0%)      0 (0%)
Events:              <none>
labsuser@master:~$ 
```

By following these steps, you have successfully verified node status, deleted a worker node, and registered a new worker node in the Kubernetes cluster, demonstrating effective node management in the cluster.