

Name: Kriti
College: Lovely Professional University
Week5 Assignment
Email: kritijulia@gmail.com

Here I made two virtual machine on azure and install docker, kubelet, kubeadm and kubectl on both vm worknode and masternode

Create Virtual Machines (1 Master + 1 Workers)

Use the Azure Portal

1. Go to <https://portal.azure.com>
2. Click on **Virtual Machines > Create**
3. Select the following:
 - o **Image:** Ubuntu Server 22.04 LTS
 - o **Size:** Standard B1s (1 vCPU, 1 GB RAM) – cheapest for testing
 - o **Authentication:** Use **SSH public key**
 - o **Open ports:** Allow SSH (port 22)

Install Docker on All Nodes

```
sudo apt update  
sudo apt install -y docker.io  
sudo systemctl start docker  
sudo systemctl enable docker
```

Update and install prerequisites

```
sudo apt-get update  
sudo apt-get install -y apt-transport-https ca-certificates curl
```

Add the Kubernetes signing key

```
sudo mkdir -p /etc/apt/keyrings  
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor  
-o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

Add the Kubernetes APT repository

```
echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]  
https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /" | sudo tee  
/etc/apt/sources.list.d/kubernetes.list
```

Update package list and install

```
sudo apt-get update  
sudo apt-get install -y kubelet kubeadm kubectl  
sudo apt-mark hold kubelet kubeadm kubectl
```

```

ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
JUL 4 4:03 PM
worker_node@workernode1:~$ sudo apt-get update
Hit:2 http://azure.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://azure.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://azure.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 http://azure.archive.ubuntu.com/ubuntu jammy-security InRelease
Ign:6 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Err:6 https://packages.cloud.google.com/apt kubernetes-xenial Release
        404  Not Found [IP: 142.250.182.78 443]
Reading package lists... Done
E: The repository 'https://apt.kubernetes.io kubernetes-xenial' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
worker_node@workernode1:~$ sudo swapoff -a
worker_node@workernode1:~$ docker --version
Docker version 27.5.1, build 27.5.1-0ubuntu3-22.04.2
worker_node@workernode1:~$ sudo apt-get install kubelet
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package kubelet
worker_node@workernode1:~$ sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package kubelet
E: Unable to locate package kubeadm
E: Unable to locate package kubectl
worker_node@workernode1:~$ curl --version
curl 7.81.0 (x86_64-pc-linux-gnu) libcurl/7.81.0 OpenSSL/3.0.2 zlib/1.2.11 brotli/1.0.9 zstd/1.4.8 libidn2/2.3.2 libpsl/0.21.0 (+libidn2/2.3.2) libssh
/0.9.6/openssl/2/lib nghttp2/1.43.0 librtmp/2.3 OpenLDAP/2.5.19
Release-Date: 2022-01-05
Protocols: dict file ftp ftps gopher gophers http https imap imaps ldap ldaps mqtt pop3 pop3s rtmp rtsp scp sftp smb smbs smtp smtps telnet tftp
Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM NTLM_WB PSL SPNEGO SSL TLS-SRP UnixSockets zst
d
worker_node@workernode1:~$ sudo mkdir -p /etc/apt/keyrings
worker_node@workernode1:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
worker_node@workernode1:~$ echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /" | sudo tee
/etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /
worker_node@workernode1:~$ sudo apt-get update

```

Ubuntu [Running] - Oracle VirtualBox : 1

File Machine View Input Devices Help

JUL 4 4:06 PM

worker_node@workernode1:~\$ kubectl --version
error: unknown flag: --version
See 'kubectl --help' for usage.

worker_node@workernode1:~\$ kubeadm

KUBEADM
Easily bootstrap a secure Kubernetes cluster

Please give us feedback at:
<https://github.com/kubernetes/kubeadm/issues>

Example usage:

Create a two-machine cluster with one control-plane node
(which controls the cluster), and one worker node
(where your workloads, like Pods and Deployments run).

On the first machine:
control-plane# kubeadm init

On the second machine:
worker# kubeadm join <arguments-returned-from-init>

You can then repeat the second step on as many other machines as you like.

Usage:
kubeadm [command]

Available Commands:
certs Commands related to handling kubernetes certificates
completion Output shell completion code for the specified shell (bash or zsh)
config Manage configuration for a kubeadm cluster persisted in a ConfigMap in the cluster

```
ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 4:06 PM
worker_node@workernode1:~ ~
--skip-log-headers      If true, avoid headers when opening log files (no effect when -logtostderr=true)
-v, --v Level           number for the log level verbosity

Additional help topics:
kubeadm alpha       Kubeadm experimental sub-commands

Use "kubeadm [command] --help" for more information about a command.
worker_node@workernode1: $ kubectl
kubectl controls the Kubernetes cluster manager.

Find more information at: https://kubernetes.io/docs/reference/kubectl/

Basic Commands (Beginner):
create      Create a resource from a file or from stdin
expose     Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
run        Run a particular image on the cluster
set         Set specific features on objects

Basic Commands (Intermediate):
explain    Get documentation for a resource
get        Display one or many resources
edit       Edit a resource on the server
delete     Delete resources by file names, stdin, resources and names, or by resources and label selector

Deploy Commands:
rollout   Manage the rollout of a resource
scale     Set a new size for a deployment, replica set, or replication controller
autoscale Auto-scale a deployment, replica set, stateful set, or replication controller

Cluster Management Commands:
certificate Modify certificate resources
cluster-info Display cluster information
top          Display resource (CPU/memory) usage
cordon      Mark node as unschedulable
uncordon    Mark node as schedulable
drain       Drain node in preparation for maintenance
taint       Update the taints on one or more nodes

Troubleshooting and Debugging Commands:
describe    Show details of a specific resource or group of resources
logs       Print the logs for a container in a pod
```



```
ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 4:06 PM
worker_node@workernode1:~ ~
--skip-log-headers      If true, avoid headers when opening log files (no effect when -logtostderr=true)
-v, --v Level           number for the log level verbosity

Additional help topics:
kubeadm alpha       Kubeadm experimental sub-commands

Use "kubeadm [command] --help" for more information about a command.
worker_node@workernode1: $ kubectl
kubectl controls the Kubernetes cluster manager.

Find more information at: https://kubernetes.io/docs/reference/kubectl/

Basic Commands (Beginner):
create      Create a resource from a file or from stdin
expose     Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
run        Run a particular image on the cluster
set         Set specific features on objects

Basic Commands (Intermediate):
explain    Get documentation for a resource
get        Display one or many resources
edit       Edit a resource on the server
delete     Delete resources by file names, stdin, resources and names, or by resources and label selector

Deploy Commands:
rollout   Manage the rollout of a resource
scale     Set a new size for a deployment, replica set, or replication controller
autoscale Auto-scale a deployment, replica set, stateful set, or replication controller

Cluster Management Commands:
certificate Modify certificate resources
cluster-info Display cluster information
top          Display resource (CPU/memory) usage
cordon      Mark node as unschedulable
uncordon    Mark node as schedulable
drain       Drain node in preparation for maintenance
taint       Update the taints on one or more nodes

Troubleshooting and Debugging Commands:
describe    Show details of a specific resource or group of resources
logs       Print the logs for a container in a pod
```

Q1. Create a Kubernetes cluster using minikube

Step1: Start Minikube Cluster

```
minikube start --driver=docker
```

Step 2: Check Cluster Status

```
kubectl get nodes
```

```
kubectl cluster-info
```

Step 3: **Test the Cluster** by deploying a test Nginx pod to verify everything is working.

```
kubectl create deployment nginx --image=nginx
```

```
kubectl expose deployment nginx --port=80 --type=NodePort
```

```
kubectl get svc
```

minikube service nginx –url

```
ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 4 7:52 PM
masternode@masternode:~
```

Building dependency tree... Done
Reading state information... Done
E: Unable to locate package minikube
masternode@masternode:~\$ sudo apt-get install -y minikube
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package minikube
masternode@masternode:~\$ sudo apt-get update
sudo apt-get install -y minikube
sudo apt-mark hold minikube
Hit:1 http://azure.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://azure.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://azure.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://azure.archive.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package minikube
E: Unable to locate package minikube
E: No packages found
masternode@masternode:~\$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
% Total % Received % Xferd Average Speed Time Time Current
 Dload Upload Total Spent Left Speed
 0 0 0 0 0 0 0:00:00 0:00:00 0:00:00 126M 43.55.
 1 1630k 0 0 1204k 0 0:01:47 0:0 21 126M 21 26.8M 0 0 11.3M 0 0:00:11 0:0 43 126M 43.55.
 0M 0 0 16.3M 0 0:00:07 0:0 65 126M 65 82.6M 0 0 18.9M 0 0:00:06 0:0 75 126M 75 95.4M 0
 0 16.8M 0 0:00:07 0:0 88 126M 88 111M 0 0 17.5M 0 0:00:07 0:0 100 126M 100 126M 0 0 18.
 3M 0 0:00:06 0:00:06 0:00:06 21.9M
masternode@masternode:~\$ chmod +x minikube-linux-amd64
masternode@masternode:~\$ sudo mv minikube-linux-amd64 /usr/local/bin/minikube
masternode@masternode:~\$ minikube version
minikube version: v1.36.0
commit: f8f52f5de1fc6ad8244afac475e1d0f96841df1-dirty
masternode@masternode:~\$

```
ubuntu [Running] - Oracle VirtualBox : 1
File Machine View Input Devices Help
Jul 5 4:20 PM
masternode@masternode:~
```

Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
masternode@masternode:~\$ kubectl get node
E0705 10:43:35.733370 7534 run.go:74] "command failed" err="unknown command get"
masternode@masternode:~\$ kubectl get nodes
E0705 10:43:45.482453 7684 run.go:74] "command failed" err="unknown command get"
masternode@masternode:~\$ kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 20h v1.33.1
masternode@masternode:~\$ minikube dashboard
📦 Enabling dashboard ...
 ■ Using image docker.io/kubernetessui/dashboard:v2.7.0
 ■ Using image docker.io/kubernetessui/metrics-scraper:v1.0.8
💡 Some dashboard features require the metrics-server addon. To enable all features please run:
 minikube addons enable metrics-server
💡 Verifying dashboard health ...
🚀 Launching proxy ...
💡 Verifying proxy health ...
🌐 Opening http://127.0.0.1:32781/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
👉 http://127.0.0.1:32781/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/
kubectl create deployment nginx --image=nginx
^C
masternode@masternode:~\$ kubectl create deployment nginx --image=nginx
error: unknown flag: --image
See 'kubectl create --help' for usage.
masternode@masternode:~\$ kubectl create deployment nginx --image=nginx

deployment.apps/nginx created
masternode@masternode:~\$ kubectl expose deployment nginx --port=80 --type=NodePort
service/nginx exposed
masternode@masternode:~\$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 20h
nginx NodePort 10.98.38.214 <none> 80:30675/TCP 25s
masternode@masternode:~\$ minikube service nginx --url
http://192.168.49.2:30675
masternode@masternode:~\$

Q2. Create a Kubernetes cluster using kubeadm

Step 1: Initialize the Cluster a link is generate for the worker node using that link connect the workernode to master node

```
sudo kubeadm init --pod-network-cidr=192.168.0.0/16
```

Step 2: Configure kubectl for User

```
mkdir -p $HOME/.kube
```

```
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
```

```
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

step 3: test

```
kubectl get nodes
```

step 4: Install CNI Plugin (Calico)

```
kubectl apply -f
```

```
https://raw.githubusercontent.com/projectcalico/calico/v3.27.0/manifests/calico.yaml
```

step5: Check status:

```
kubectl get pods -n kube-system
```

ubuntu [Running] - Oracle VirtualBox : 1

File Machine View Input Devices Help

Jul 4 7:34 PM

worker1@workernode:~

```
net.ipv4.conf.default.promote_secondaries = 1
sysctl: setting key "net.ipv4.conf.all.promote_secondaries": Invalid argument
net.ipv4.ping_group_range = 0 2147483647
net.core.default_qdisc = fq_codel
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
fs.protected_regular = 1
fs.protected_fifos = 1
* Applying /usr/lib/sysctl.d/50-pid-max.conf ...
kernel.pid_max = 4194304
* Applying /etc/sysctl.d/99-cloudimg-ipv6.conf ...
net.ipv6.conf.all.use_tempaddr = 0
net.ipv6.conf.default.use_tempaddr = 0
* Applying /etc/sysctl.d/99-cloudimg-udp.conf ...
net.core.rmem_max = 1048576
net.core.rmem_default = 1048576
* Applying /usr/lib/sysctl.d/99-protect-links.conf ...
fs.protected_fifos = 1
fs.protected_hardlinks = 1
fs.protected_regular = 2
fs.protected_symlinks = 1
* Applying /etc/sysctl.d/99-sysctl.conf ...
* Applying /etc/sysctl.d/k8s.conf ...
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
* Applying /etc/sysctl.conf ...
worker1@workernode:~$ kubeadm join 10.0.0.4:6443 --token bx1nkwew4ihxly0pvm7gr \
--discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
accepts at most 1 arg(s), received 3
To see the stack trace of this error execute with --v=5 or higher
r
worker1@workernode:~$
```

masternode@masternode:~

```
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 10.0.0.4:6443 --token bx1nkwew4ihxly0pvm7gr \
--discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
masternode@masternode:~$
```

ubuntu [Running] - Oracle VirtualBox : 1

File Machine View Input Devices Help

Jul 4 7:45 PM

vboxuser@masterubuntu:~/Desktop

```
fs.protected_symlinks = 1
fs.protected_regular = 1
fs.protected_fifos = 1
* Applying /usr/lib/sysctl.d/50-pid-max.conf ...
kernel.pid_max = 4194304
* Applying /etc/sysctl.d/99-cloudimg-ipv6.conf ...
net.ipv6.conf.all.use_tempaddr = 0
net.ipv6.conf.default.use_tempaddr = 0
* Applying /etc/sysctl.d/99-cloudimg-udp.conf ...
net.core.rmem_max = 1048576
net.core.rmem_default = 1048576
* Applying /usr/lib/sysctl.d/99-protect-links.conf ...
fs.protected_fifos = 1
fs.protected_hardlinks = 1
fs.protected_regular = 2
fs.protected_symlinks = 1
* Applying /etc/sysctl.d/99-sysctl.conf ...
* Applying /etc/sysctl.d/k8s.conf ...
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
* Applying /etc/sysctl.conf ...
worker1@workernode:~$ kubeadm join 10.0.0.4:6443 --token bx1nkwew4ihxly0pvm7gr \
--discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
accepts at most 1 arg(s), received 3
To see the stack trace of this error execute with --v=5 or higher
r
worker1@workernode:~$ kubelet get nodes
E0704 14:12:10.978462    3688 run.go:74] "command failed" err="unknown command get"
worker1@workernode:~$ exit
logout
Connection to 20.2.81.219 closed.
vboxuser@masterubuntu:~/Desktop$
```

masternode@masternode:~

```
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 10.0.0.4:6443 --token bx1nkwew4ihxly0pvm7gr \
--discovery-token-ca-cert-hash sha256:ce5b921b262cd1a54fd89e6f1d9f8f63dabd465086755a67b60971c5bc68a691
masternode@masternode:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
masternode@masternode:~$ kubectl get nodes
NAME      STATUS   ROLES   AGE   VERSION
masternode  NotReady   control-plane   11m   v1.28.15
masternode@masternode:~$ ls
masternode@masternode:~$ kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.1/manifests/calico.yaml
The connection to the server 10.0.0.4:6443 was refused - did you specify the right host or port?
masternode@masternode:~$ exit
logout
Connection to 20.2.89.124 closed.
vboxuser@masterubuntu:~/Desktop$
```

Q3. Deploy an AKS cluster using the portal.

Access the dashboard and create roles for multiple users

Step1: Deploy an AKS cluster using the portal

The screenshot shows the Microsoft Azure portal interface for an AKS cluster named 'kubecluster'. The left sidebar has a 'Kubernetes service' icon. The main content area displays the 'Overview' tab for the cluster. Key details shown include:

- Resource group: kubecluster_group
- Power state: Running
- Cluster operation status: Succeeded
- Subscription: Azure for Students
- Location: East Asia
- Subscription ID: e09ba5dc-dac9-4055-b444-6048786ca660
- Fleet Manager: Click here to assign
- Tags: Add tags

The 'Properties' tab is selected. Under 'Kubernetes services', it shows:

- Encryption type: Encryption at-rest with a platform-managed key
- Virtual node pools: Not enabled

The right panel contains a 'Connect to kubecluster' section with links for 'Open Cloud Shell' and 'Sample commands' related to Kubernetes and Azure services.

Step 2: Access AKS Cluster via Cloud Shell

After deployment:

1. Click "**Go to Resource**".
2. On the AKS cluster page, click "**Connect**".
3. Open **Azure Cloud Shell (Bash)** in top menu or install az CLI locally.

```
az login
```

```
az account set --subscription "<your-subscription-name>"
```

```
az aks get-credentials --resource-group aks-rg --name myAKSCluster
```

```
kubectl get nodes
```

Step 3: Access Kubernetes Dashboard

```
az aks browse --resource-group kubeclustergroup --name myxyz
```

Connect to kubecluster

Set cluster context

1. Open Cloud Shell

Automatically runs the following commands

Set the cluster subscription
az account set --subscription e09ba5dc-dac9-4055-b444-6048786ca660

Download cluster credentials
az aks get-credentials --resource-group kubecluster_group --name kubecluster --admin

Sample commands

Once you have run the command above to connect to the cluster, you can run any kubectl commands. Here are a few examples of useful commands you can try.

- List all deployments in all namespaces
kubectl get deployments --all-namespaces=true
- List all deployments in a specific namespace
kubectl get deployments --namespace <namespace-name>
- List details about a specific deployment
kubectl describe deployment <deployment-name> --namespace <namespace-name>
- Get logs for all pods with a specific label
kubectl logs -l <label-key>=<label-value>

Networking

API server address
Network configuration
Pod CIDR
Service CIDR
DNS service IP
Cilium dataplane
Network Policy
Load balancer
Private cluster

Useful links

[Get started](#) [Properties](#) [Monitoring](#) [Recommendations](#)

Upgrades

Kubernetes version: 1.32.5

Workloads

Deployments Pods Replica sets Stateful sets Daemon sets Jobs Cron

Name	Namespace	Ready	Age
coredns	kube-system	2/2	9 minutes
coredns-autoscaler	kube-system	1/1	9 minutes
connectivity-agent	kube-system	2/2	9 minutes
connectivity-agent-aut	kube-system	1/1	9 minutes
metrics-server	kube-system	2/2	9 minutes
azure-wi-webhook-con	kube-system	2/2	8 minutes
eraser-controller-manag	kube-system	1/1	8 minutes

Step 4: Create Roles for Multiple Users (RBAC)

```
# role.yaml

apiVersion: rbac.authorization.k8s.io/v1

kind: Role

metadata:

namespace: default

name: pod-reader
```

```
rules:  
- apiGroups: [""]  
  resources: ["pods"]  
  verbs: ["get", "watch", "list"]
```

And bind it to a user:

```
# rolebinding.yaml  
  
apiVersion: rbac.authorization.k8s.io/v1  
kind: RoleBinding  
  
metadata:  
  name: read-pods  
  namespace: default  
  
subjects:  
- kind: User  
  name: <azure-ad-user-email>  
  apiGroup: rbac.authorization.k8s.io  
  
roleRef:  
  kind: Role  
  name: pod-reader  
  apiGroup: rbac.authorization.k8s.io
```

Apply with:

```
kubectl apply -f role.yaml  
kubectl apply -f rolebinding.yaml
```

```
rolebinding.yaml
2 apiVersion: rbac.authorization.k8s.io/v1
3 kind: RoleBinding
4 metadata:
5   name: read-pods
6   namespace: default
7 subjects:
8 - kind: User
9   name: <azure-ad-user-email>
10  apiGroup: rbac.authorization.k8s.io
11 roleRef:
12   kind: Role
13   name: pod-reader
14  apiGroup: rbac.authorization.k8s.io
15
16
role.yaml
2 apiVersion: rbac.authorization.k8s.io/v1
3 kind: Role
4 metadata:
5   namespace: default
6   name: pod-reader
7 rules:
8 - apiGroups: []
9   resources: ["pods"]
10  verbs: ["get", "watch", "list"]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

details please go to <https://aka.ms/ANRResourceNotFoundFix>

Code: ResourceNotFound

Message: The Resource 'Microsoft.ContainerService/managedClusters/myAKSCluster' under resource group 'ASKrg' was not found. For more details please go to <https://aka.ms/ANRResourceNotFoundFix>

Kubernetes resources view on <https://portal.azure.com/#resource/subscriptions/e09ba5dc-dac9-4055-b444-6048786ca660/resourceGroups/ASKrg/providers/Microsoft.ContainerService/managedClusters/kubecluster/workloads>

"Kubernetes resources view on [https://portal.azure.com/#resource/subscriptions/e09ba5dc-dac9-4055-b444-6048786ca660/resourceGroups/ASKrg/providers/Microsoft.ContainerService/managedClusters/kubecluster/workloads"](https://portal.azure.com/#resource/subscriptions/e09ba5dc-dac9-4055-b444-6048786ca660/resourceGroups/ASKrg/providers/Microsoft.ContainerService/managedClusters/kubecluster/workloads)

kubectl apply -f kubeyaml/rolebinding.yaml

PS D:\internship-celebal\week1\celebal-intern> kubectl apply -f kubeyaml/role.yaml

rolebinding.rbac.authorization.k8s.io/read-pods created

PS D:\internship-celebal\week1\celebal-intern>

In 16, Col 1 Spaces:2 UTF-8 CRLF YAML ✓ Prettier

Q4. Deploy a microservice application on AKS cluster and access it using public internet

I'll deploy a simple **multi-container microservice** (e.g., frontend + backend) and expose it using a **LoadBalancer** service.

Step 1: Create a Sample Microservice App (YAML files)

Step 2: Deploy to AKS

```
az aks get-credentials --resource-group aks-rg --name myAKSCluster
kubectl apply -f hello-app-deployment.yaml
kubectl apply -f hello-app-service.yaml
```

Step 3: Check the LoadBalancer External IP

The screenshot shows a terminal window with the following command history:

```

PS D:\internship-celebal\week1\celebal-intern> kubectl apply -f solution_w5/microservices/hello-app-service.yaml
service/hello-app-service created
PS D:\internship-celebal\week1\celebal-intern> kubectl get svc hello-app-service
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
hello-app-service   LoadBalancer   10.0.206.14   <none>        80:30349/TCP   2s
PS D:\internship-celebal\week1\celebal-intern> kubectl get svc hello-app-service
Unreachable host: dial tcp: lookup kubecluster-dns-f65gkqp.hcp.malaysiawest.azmk8s.io: no such host
PS D:\internship-celebal\week1\celebal-intern> kubectl get svc hello-app-service
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
hello-app-service   LoadBalancer   10.0.206.14   85.211.156.10   80:30349/TCP   63s
PS D:\internship-celebal\week1\celebal-intern>
PS D:\internship-celebal\week1\celebal-intern>

```

Below the terminal, a browser window displays the Kubernetes logo and the text "Hello world!".

Q5. Expose services in the cluster with node port, cluster IP, load balancer

Step 1: Create Deployment

```
kubectl create deployment nginx --image=nginx
```

Step 2: Expose as ClusterIP (default)

This is the **default service type**, only accessible **within the cluster**.

```
kubectl expose deployment nginx --port=80 --target-port=80 --name=nginx-clusterip
```

Check: `kubectl get svc nginx-clusterip`

Step 3: Expose as NodePort

Makes the app available on each node's IP and a high port (e.g., 30000-32767).

```
kubectl expose deployment nginx --port=80 --target-port=80 --type=NodePort --name=nginx-nodeport
```

Check: `kubectl get svc nginx-nodeport`

Example output:

```
nginx-nodeport  NodePort  10.0.x.x <none>  80:31123/TCP  1m
```

```
kubectl get nodes -o wide
```

Use the node's external IP + node port, e.g.:

👉 `http://<NodePublicIP>:31123`

⚠️ On AKS, **NodePort access requires NSG/Firewall to allow that port**, which can be tricky. That's why LoadBalancer is preferred.

Step 4: Expose as LoadBalancer

```
kubectl expose deployment nginx --port=80 --target-port=80 --  
type=LoadBalancer --name=nginx-lb
```

```
kubectl get svc nginx-lb
```

Wait until the EXTERNAL-IP appears and Access:

👉 `http://<EXTERNAL-IP>`

Final: Clean Up

```
kubectl delete svc nginx-clusterip nginx-nodeport nginx-lb
```

```
kubectl delete deployment nginx
```

Two screenshots of a terminal session in a code editor showing the deployment of a microservice application.

Screenshot 1:

```

solution_w5> microservices > ! hello-app-service.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: hello-app-service
5  spec:
6    type: LoadBalancer
7    selector:
8      app: hello-app
9    ports:
10      - protocol: TCP
11        port: 80
12        targetPort: 8080
13
14
solution_w5> microservices > ! hello-app-deployment.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: hello-app
5  spec:
6    replicas: 2
7    selector:
8      matchLabels:
9        app: hello-app
10     template:
11       metadata:
12         labels:
13           app: hello-app
14         spec:
15           containers:
16             - name: hello-app
17               image: paulbouwer/hello-kubernetes:1.10
18             ports:
19               - containerPort: 8080
20
21
PS D:\internship\celebal\week1\celebal-intern> kubectl get svc nginx-clusterip
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
nginx-clusterip  ClusterIP  10.0.76.1   <none>        80/TCP   8s
PS D:\internship\celebal\week1\celebal-intern> kubectl expose deployment nginx --port=80 --target-port=80 --name=nginx-nodeport
service/nginx-nodeport exposed
PS D:\internship\celebal\week1\celebal-intern> kubectl get svc nginx-nodeport
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
nginx-nodeport  NodePort  10.0.236.54  <none>        80:32313/TCP   8s

```

Screenshot 2:

```

solution_w5> microservices > ! hello-app-service.yaml
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: hello-app-service
5  spec:
6    type: LoadBalancer
7    selector:
8      app: hello-app
9    ports:
10      - protocol: TCP
11        port: 80
12        targetPort: 8080
13
14
solution_w5> microservices > ! hello-app-deployment.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: hello-app
5  spec:
6    replicas: 2
7    selector:
8      matchLabels:
9        app: hello-app
10     template:
11       metadata:
12         labels:
13           app: hello-app
14         spec:
15           containers:
16             - name: hello-app
17               image: paulbouwer/hello-kubernetes:1.10
18             ports:
19               - containerPort: 8080
20
21
PS D:\internship\celebal\week1\celebal-intern> kubectl get nodes -o wide
● NAME          STATUS   ROLES   AGE   VERSION   INTERNAL-IP   EXTERNAL-IP
  OS-IMAGE      KERNEL-VERSION   CONTAINER-RUNTIME
aks-agentpool-23356145-vms-00000000  Ready   <none>   34m   v1.31.8   10.224.0.4   <none>
Ubuntu 22.04.5 LTS  5.15.0-1090-azure  containerd://1.7.27-1
aks-agentpool-23356145-vms-00000001  Ready   <none>   34m   v1.31.8   10.224.0.5   <none>
Ubuntu 22.04.5 LTS  5.15.0-1090-azure  containerd://1.7.27-1
○ PS D:\internship\celebal\week1\celebal-intern>
● PS D:\internship\celebal\week1\celebal-intern> kubectl get nodes -o wide
NAME          STATUS   ROLES   AGE   VERSION   INTERNAL-IP   EXTERNAL-IP
  OS-IMAGE      KERNEL-VERSION   CONTAINER-RUNTIME
0.224.0.4   <none>   Ubuntu 22.04.5 LTS  5.15.0-1090-azure  containerd://1.7.27-1
aks-agentpool-23356145-vms-00000002  Ready   <none>   34m   v1.31.8   10.224.0.5   <none>
Ubuntu 22.04.5 LTS  5.15.0-1090-azure  containerd://1.7.27-1
PS D:\internship\celebal\week1\celebal-intern> kubectl expose deployment nginx --port=80 --target-port=80 --name=nginx-lb
service/nginx-lb exposed
PS D:\internship\celebal\week1\celebal-intern> kubectl get svc nginx-lb

```

The terminal session shows the creation of a service and deployment for the "hello-app" application, followed by the creation of an external load balancer service ("nginx-lb") that routes traffic to the internal "nginx" service. The browser window shows the "Welcome to nginx!" page at <http://85.211.168.34>.