Azure Cloud

By Sriharshini J

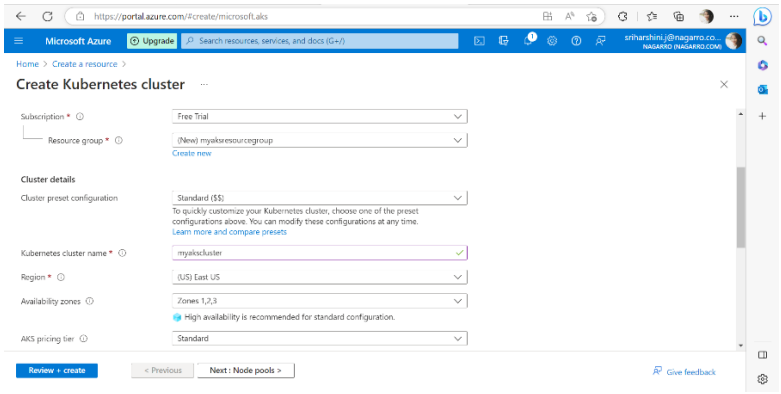
Create the AKS cluster (2 nodes, smallest size VM) and deploy any two services on it. Services should be accessible from the Internet.\

1. Sign in to the Azure portal.
2. On the Azure portal menu or from the Home page, select Create a resource.
3. Select Containers > Kubernetes Service.
4. On the Basics page, configure the following options:
    - Project details:
        - Select an Azure Subscription.
        - Select or create an Azure Resource group, such as myResourceGroup.
    - Cluster details:
        - Ensure the Preset configuration is Standard ($$). For more details on preset configurations, Enter a Kubernetes cluster name, such as myAKSCluster.
    - Primary node pool:
        - Leave the default values selected.

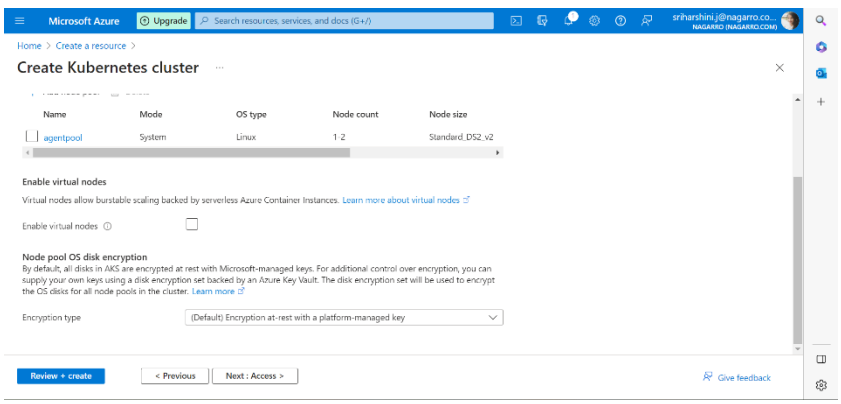
1. Select Next: Node pools when complete.
2. Keep the default Node pools options. At the bottom of the screen, click Next: Access.
3. On the Access page, configure the following options:
    - The default value for Resource identity is System-assigned managed identity details about managed identities,
    - The Kubernetes role-based access control (RBAC) option is the default value to provide more fine-grained control over access to the Kubernetes resources deployed in your AKS cluster.
4. Select Next: Networking when complete.

5. Keep the default Networking options. At the bottom of the screen, click Next: Integrations.
6. On the Integrations page, select Enable recommended alert rules. You can see the list of alerts that are automatically enabled if you select this option.
7. Click Review + create. When you navigate to the Review + create tab, Azure runs validation on the settings that you have chosen. If validation passes, can proceed to create the AKS cluster by selecting Create. If validation fails, then it indicates which settings need to be modified.
8. It takes a few minutes to create the AKS cluster. When your deployment is complete, navigate to your resource by either:
    - Selecting Go to resource, or
    - Browsing to the AKS cluster resource group and selecting the AKS resource. for myResourceGroup and select the resource myAKSCluster.

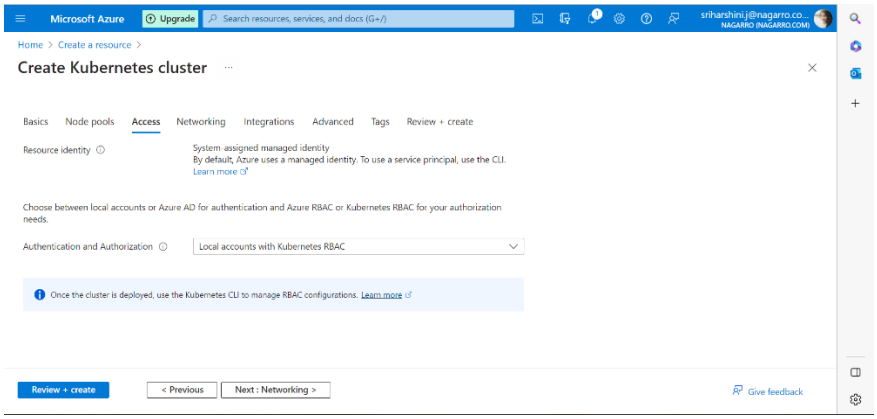
Snippet: 1



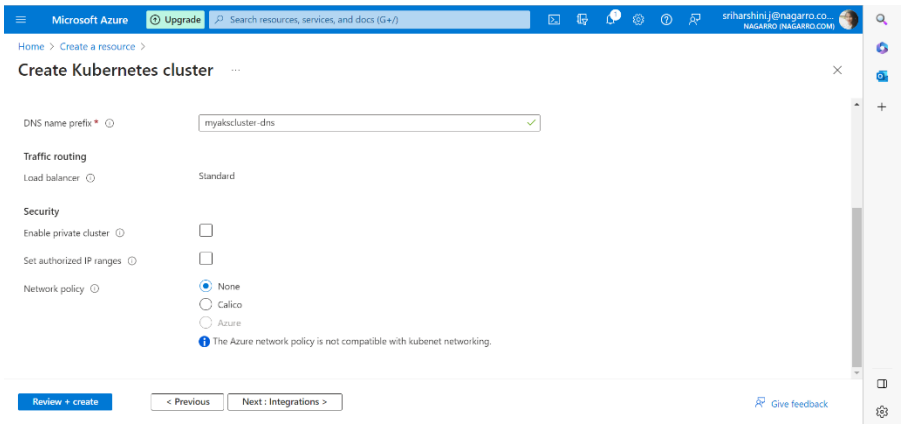
Snippet: 2



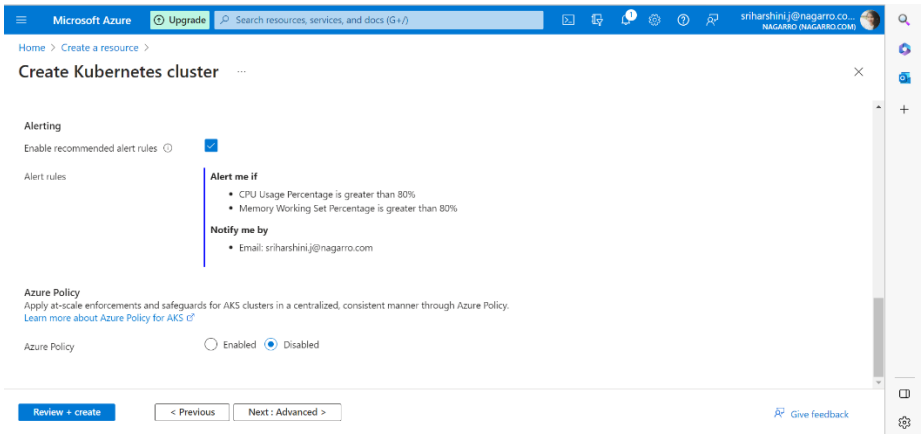
Snippet: 3



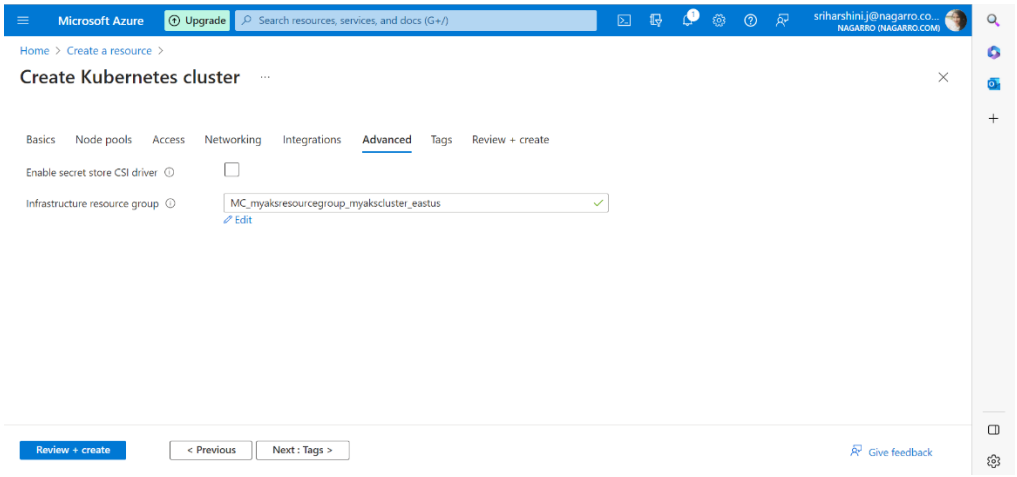
Snippet: 4

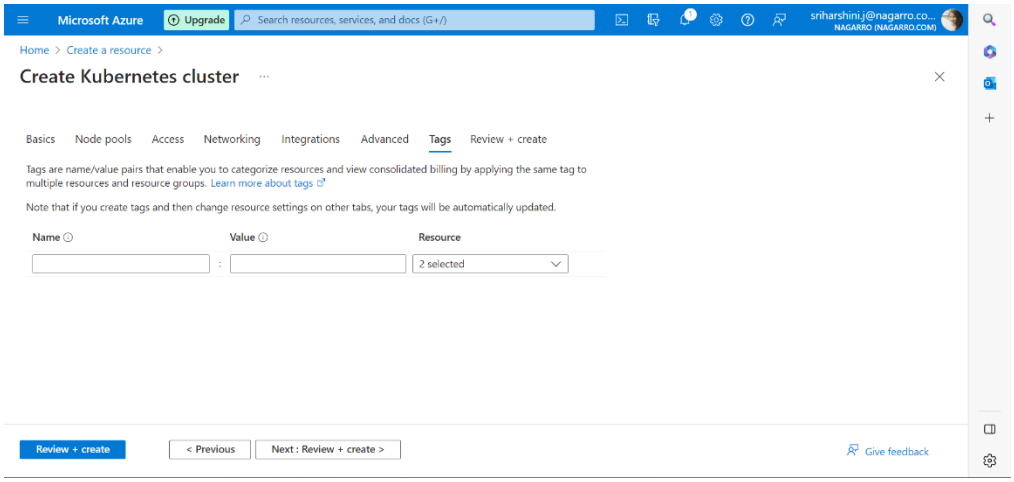


Snippet: 5

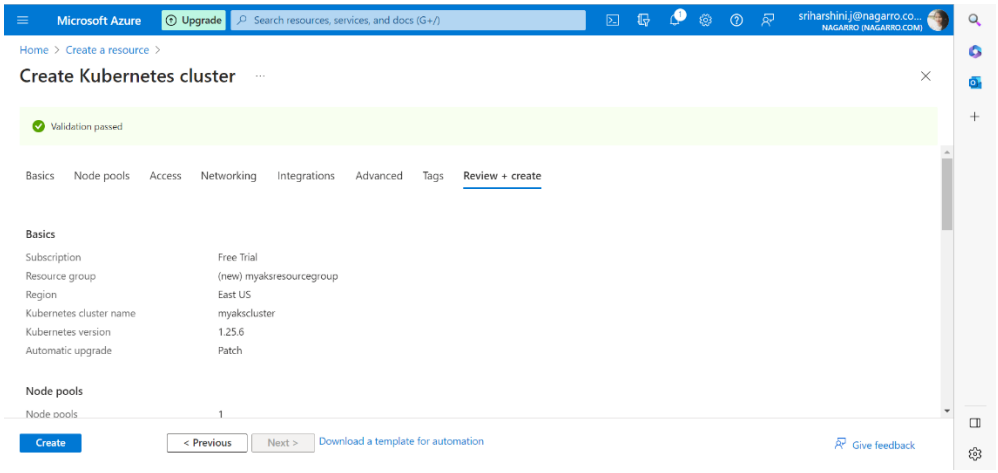


Snippet: 6 & 7

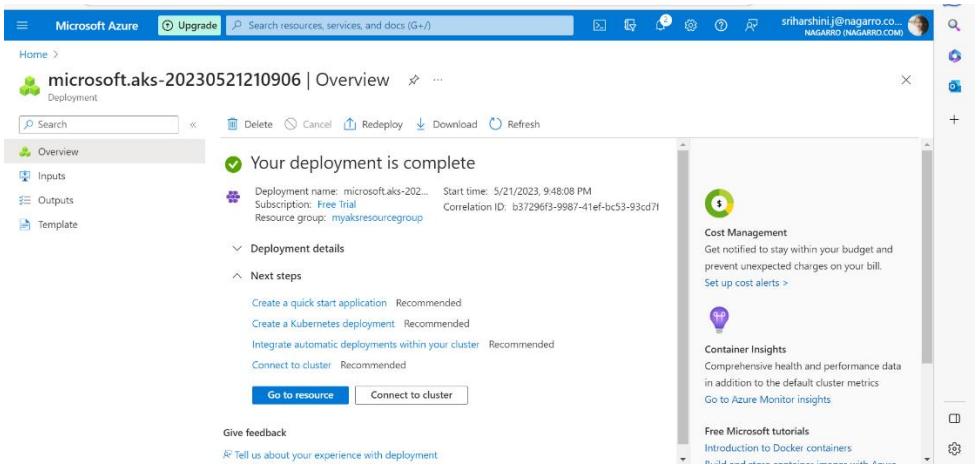


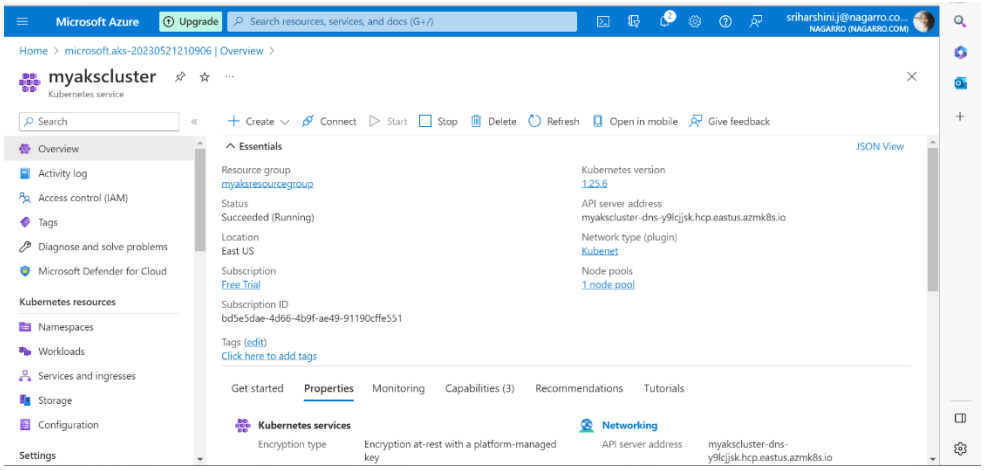


Snippet: 8



Snippet: 9 &10





Connect to the cluster

To manage a Kubernetes cluster, use the Kubernetes command-line client, kubectl. kubectl is already installed if you use Azure Cloud Shell. If you're unfamiliar with the Cloud Shell,

1. Open Cloud Shell using the >_ button on the top of the Azure portal

az aks get-credentials --resource-group myResourceGroup --name myAKSCluster

kubectl get nodes

NAME STATUS ROLES AGE VERSION

aks-agentpool-12345678-vmss000000 Ready agent 23m v1.19.11

aks-agentpool-12345678-vmss000001 Ready agent 24m v1.19.11

Two Kubernetes Services are also created:

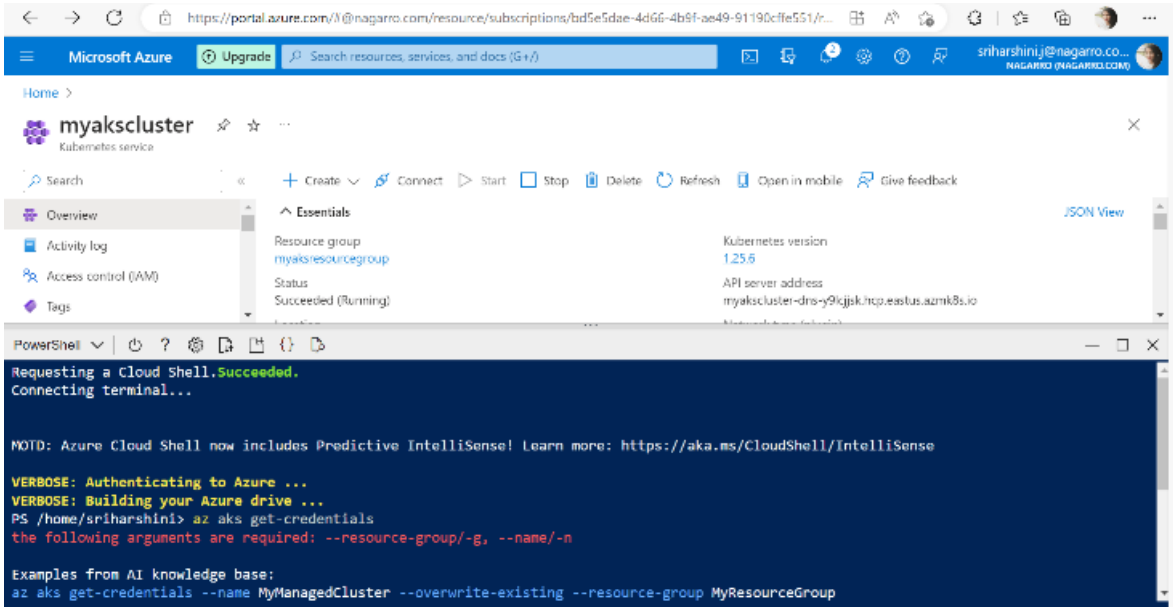
- An internal service for the Redis instance.
- An external service to access the Azure Vote application from the internet.

1. In the Cloud Shell, open an editor and create a file named azure-vote.yaml.
2. Paste in the following YAML definition:

Deploy the application using the `kubectl apply` command and specify the name of your YAML manifest:

- `kubectl apply -f azure-vote.yaml`
- deployment "azure-vote-back" created
- service "azure-vote-back" created
- deployment "azure-vote-front" created
- service "azure-vote-front" created
- azure-vote-front LoadBalancer 10.0.37.27 52.179.23.131 80:30572/TCP

CODE



```
apiVersion: apps/v1
kind: Deployment
metadata:
name: azure-vote-back
spec:
replicas: 1
selector:
matchLabels:
app: azure-vote-back
template:
metadata:
labels:
app: azure-vote-back
spec:
nodeSelector:
"kubernetes.io/os": linux
containers:
- name: azure-vote-back
image: mcr.microsoft.com/oss/bitnami/redis:6.0.8
env:
- name: ALLOW_EMPTY_PASSWORD
value: "yes"
resources:
requests:
cpu: 100m
memory: 128Mi
limits:
cpu: 250m
```

```
memory: 256Mi
ports:
- containerPort: 6379
name: redis
---
apiVersion: v1
kind: Service
metadata:
name: azure-vote-back
spec:
ports:
- port: 6379
selector:
app: azure-vote-back
---
apiVersion: apps/v1
kind: Deployment
metadata:
name: azure-vote-front
spec:
replicas: 1
selector:
matchLabels:
app: azure-vote-front
template:
metadata:
labels:
app: azure-vote-front
```

```
spec:
nodeSelector:
"kubernetes.io/os": linux
containers:
- name: azure-vote-front
image: mcr.microsoft.com/azuredocs/azure-vote-front:v1
resources:
requests:
cpu: 100m
memory: 128Mi
limits:
cpu: 250m
memory: 256Mi
ports:
- containerPort: 80
env:
- name: REDIS
value: "azure-vote-back"
---
apiVersion: v1
kind: Service
metadata:
name: azure-vote-front
spec:
type: LoadBalancer
ports:
- port: 80
selector:
```

app: azure-vote-front

RESULT

```
        TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE
ack     ClusterIP      10.0.225.88    <none>          6379/TCP       31s
ront    LoadBalancer   10.0.75.186    20.81.67.186    80:31394/TCP   30s
        ClusterIP      10.0.0.1       <none>          443/TCP        9m2s
eel> kubectl get rs
                  DESIRED   CURRENT   READY   AGE
ack-59cb7dc555    1         1         1       58s
ront-5f4d7db9c8   1         1         1       57s
eel> kubectl get pods
```