**K8S IN AZURE INSTALLION**

1. Login in azure portal
2. URL for portal is portal.azure.com
3. First create in Resoure Group
4. az group create --name myResourceGroup --location eastus

Output for successfully created resource group:

JSON Copy

{

"id": "/subscriptions/<guid>/resourceGroups/myResourceGroup",

"location": "eastus",

"managedBy": null,

"name": "myResourceGroup",

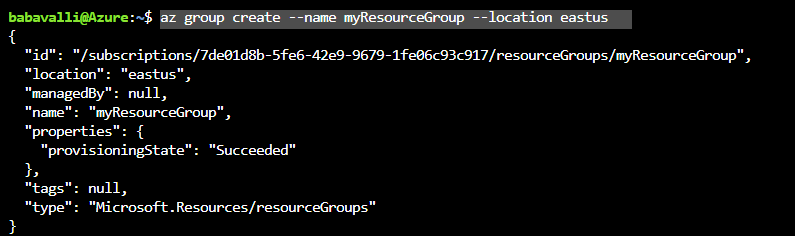
"properties": {

"provisioningState": "Succeeded"

},

"tags": null

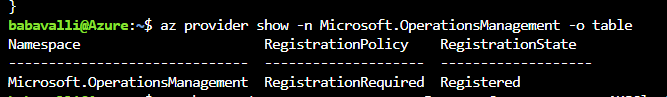
}



## 4. Enable cluster monitoring

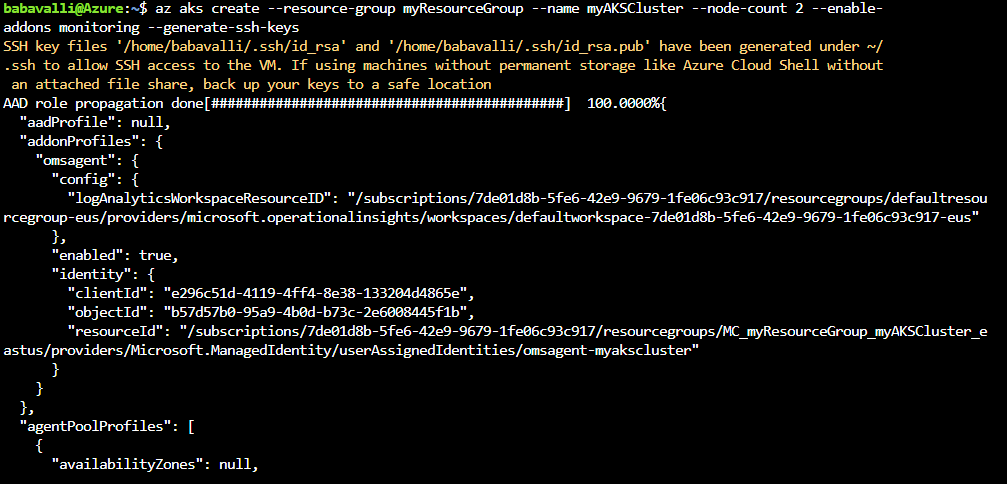
Verify *Microsoft.OperationsManagement* and *Microsoft.OperationalInsights* are registered on your subscription. To check the registration status:

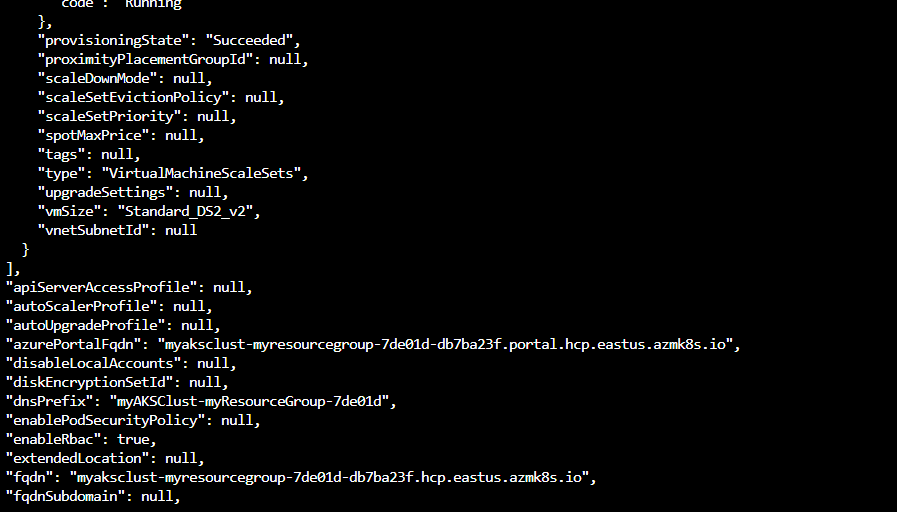
az provider show -n Microsoft.OperationsManagement -o table

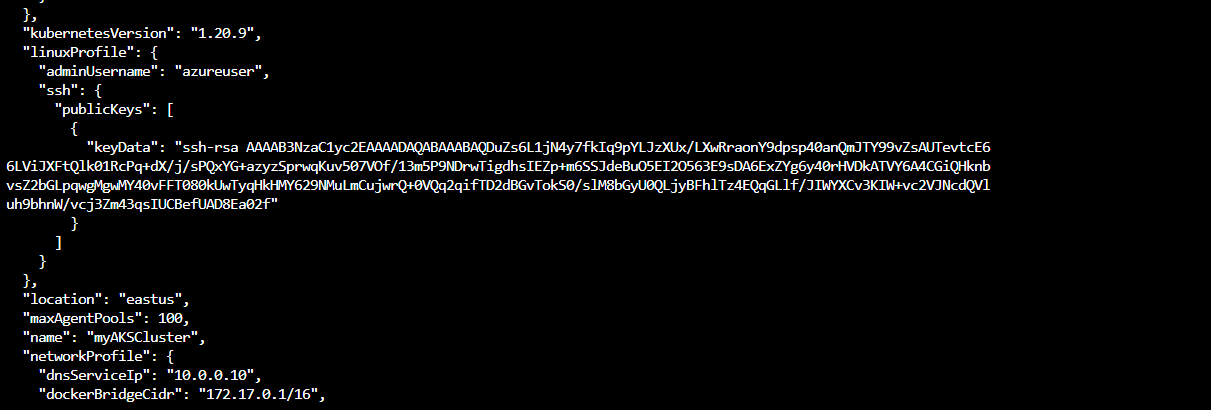


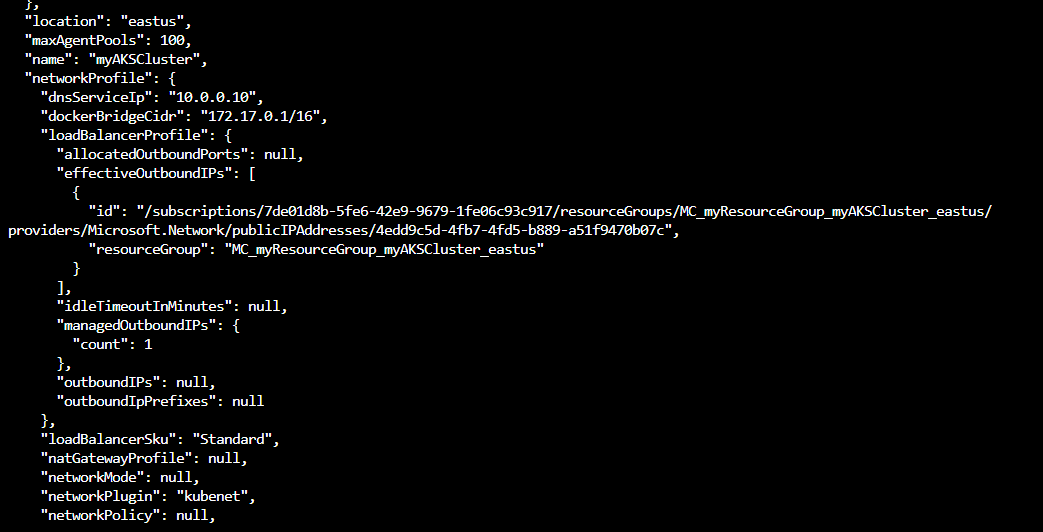
5.

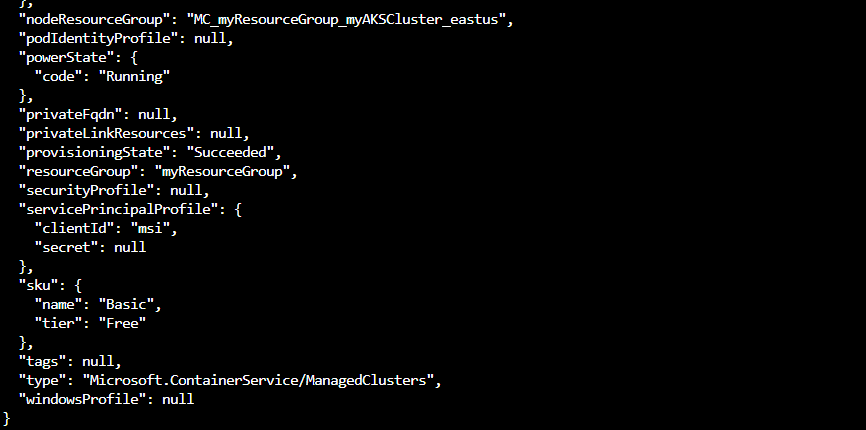
az aks create --resource-group myResourceGroup --name myAKSCluster --node-count 2 --enable-addons monitoring --generate-ssh-keys











Verify the connection to your cluster using the [kubectl get](https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#get) command. This command returns a list of the cluster nodes.

Azure CLICopy

Try It

kubectl get nodes

Reference follow the official documents

# Quickstart: Deploy an Azure Kubernetes Service cluster using the Azure CLI

* + +26

Azure Kubernetes Service (AKS) is a managed Kubernetes service that lets you quickly deploy and manage clusters. In this quickstart, you will:

* Deploy an AKS cluster using the Azure CLI.
* Run a multi-container application with a web front-end and a Redis instance in the cluster.
* Monitor the health of the cluster and pods that run your application.

This quickstart assumes a basic understanding of Kubernetes concepts. For more information, see [Kubernetes core concepts for Azure Kubernetes Service (AKS)](https://docs.microsoft.com/en-us/azure/aks/concepts-clusters-workloads).

If you don't have an [Azure subscription](https://docs.microsoft.com/en-us/azure/guides/developer/azure-developer-guide#understanding-accounts-subscriptions-and-billing), create a [free account](https://azure.microsoft.com/free/?ref=microsoft.com&utm_source=microsoft.com&utm_medium=docs&utm_campaign=visualstudio) before you begin.

To learn more about creating a Windows Server node pool, see [Create an AKS cluster that supports Windows Server containers](https://docs.microsoft.com/en-us/azure/aks/windows-container-cli).

## Prerequisites

* Use the Bash environment in [Azure Cloud Shell](https://docs.microsoft.com/en-us/azure/cloud-shell/quickstart).

[Launch Cloud Shell in a new window](https://shell.azure.com/)

* If you prefer, [install](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli) the Azure CLI to run CLI reference commands.
  + If you're using a local installation, sign in to the Azure CLI by using the [az login](https://docs.microsoft.com/en-us/cli/azure/reference-index" \l "az_login) command. To finish the authentication process, follow the steps displayed in your terminal. For additional sign-in options, see [Sign in with the Azure CLI](https://docs.microsoft.com/en-us/cli/azure/authenticate-azure-cli).
  + When you're prompted, install Azure CLI extensions on first use. For more information about extensions, see [Use extensions with the Azure CLI](https://docs.microsoft.com/en-us/cli/azure/azure-cli-extensions-overview).
  + Run [az version](https://docs.microsoft.com/en-us/cli/azure/reference-index?" \l "az_version) to find the version and dependent libraries that are installed. To upgrade to the latest version, run [az upgrade](https://docs.microsoft.com/en-us/cli/azure/reference-index?" \l "az_upgrade).
* This article requires version 2.0.64 or greater of the Azure CLI. If using Azure Cloud Shell, the latest version is already installed.
* The identity you are using to create your cluster has the appropriate minimum permissions. For more details on access and identity for AKS, see [Access and identity options for Azure Kubernetes Service (AKS)](https://docs.microsoft.com/en-us/azure/aks/concepts-identity).

**Note**

Run the commands as administrator if you plan to run the commands in this quickstart locally instead of in Azure Cloud Shell.

## Create a resource group

An [Azure resource group](https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/overview) is a logical group in which Azure resources are deployed and managed. When you create a resource group, you will be prompted to specify a location. This location is:

* The storage location of your resource group metadata.
* Where your resources will run in Azure if you don't specify another region during resource creation.

The following example creates a resource group named myResourceGroup in the eastus location.

Create a resource group using the [az group create](https://docs.microsoft.com/en-us/cli/azure/group" \l "az_group_create) command.

Azure CLICopy

Try It

az group create --name myResourceGroup --location eastus

Output for successfully created resource group:

JSONCopy

{

"id": "/subscriptions/<guid>/resourceGroups/myResourceGroup",

"location": "eastus",

"managedBy": null,

"name": "myResourceGroup",

"properties": {

"provisioningState": "Succeeded"

},

"tags": null

}

## Enable cluster monitoring

Verify Microsoft.OperationsManagement and Microsoft.OperationalInsights are registered on your subscription. To check the registration status:

Azure CLICopy

az provider show -n Microsoft.OperationsManagement -o table

az provider show -n Microsoft.OperationalInsights -o table

If they are not registered, register Microsoft.OperationsManagement and Microsoft.OperationalInsights using:

Azure CLICopy

az provider register --namespace Microsoft.OperationsManagement

az provider register --namespace Microsoft.OperationalInsights

## Create AKS cluster

Create an AKS cluster using the [az aks create](https://docs.microsoft.com/en-us/cli/azure/aks" \l "az_aks_create) command with the --enable-addons monitoring parameter to enable [Azure Monitor for containers](https://docs.microsoft.com/en-us/azure/azure-monitor/containers/container-insights-overview). The following example creates a cluster named myAKSCluster with one node:

Azure CLICopy

Try It

az aks create --resource-group myResourceGroup --name myAKSCluster --node-count 1 --enable-addons monitoring --generate-ssh-keys

After a few minutes, the command completes and returns JSON-formatted information about the cluster.

**Note**

When you create an AKS cluster, a second resource group is automatically created to store the AKS resources. For more information, see [**Why are two resource groups created with AKS?**](https://docs.microsoft.com/en-us/azure/aks/faq#why-are-two-resource-groups-created-with-aks)

## Connect to the cluster

To manage a Kubernetes cluster, use the Kubernetes command-line client, [kubectl](https://kubernetes.io/docs/user-guide/kubectl/). kubectl is already installed if you use Azure Cloud Shell.

1. Install kubectl locally using the [az aks install-cli](https://docs.microsoft.com/en-us/cli/azure/aks" \l "az_aks_install_cli) command:

Azure CLICopy

az aks install-cli

1. Configure kubectl to connect to your Kubernetes cluster using the [az aks get-credentials](https://docs.microsoft.com/en-us/cli/azure/aks" \l "az_aks_get_credentials) command. The following command:
   * Downloads credentials and configures the Kubernetes CLI to use them.
   * Uses ~/.kube/config, the default location for the [Kubernetes configuration file](https://kubernetes.io/docs/concepts/configuration/organize-cluster-access-kubeconfig/). Specify a different location for your Kubernetes configuration file using --file.

Azure CLICopy

Try It

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

1. Verify the connection to your cluster using the [kubectl get](https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#get) command. This command returns a list of the cluster nodes.

Azure CLICopy

Try It

kubectl get nodes

Output shows the single node created in the previous steps. Make sure the node status is Ready:

OutputCopy

NAME STATUS ROLES AGE VERSION

aks-nodepool1-31718369-0 Ready agent 6m44s v1.12.8

## Run the application

A [Kubernetes manifest file](https://docs.microsoft.com/en-us/azure/aks/concepts-clusters-workloads#deployments-and-yaml-manifests) defines a cluster's desired state, such as which container images to run.

In this quickstart, you will use a manifest to create all objects needed to run the [Azure Vote application](https://github.com/Azure-Samples/azure-voting-app-redis.git). This manifest includes two [Kubernetes deployments](https://docs.microsoft.com/en-us/azure/aks/concepts-clusters-workloads#deployments-and-yaml-manifests):

* The sample Azure Vote Python applications.
* A Redis instance.

Two [Kubernetes Services](https://docs.microsoft.com/en-us/azure/aks/concepts-network#services) are also created:

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

1. Create a file named azure-vote.yaml.
   * If you use the Azure Cloud Shell, this file can be created using code, vi, or nano as if working on a virtual or physical system
2. Copy in the following YAML definition:

YAMLCopy

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

1. Deploy the application using the [kubectl apply](https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#apply) command and specify the name of your YAML manifest:

ConsoleCopy

kubectl apply -f azure-vote.yaml

Output shows the successfully created deployments and services:

OutputCopy

deployment "azure-vote-back" created

service "azure-vote-back" created

deployment "azure-vote-front" created

service "azure-vote-front" created

## Test the application

When the application runs, a Kubernetes service exposes the application front end to the internet. This process can take a few minutes to complete.

Monitor progress using the [kubectl get service](https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#get) command with the --watch argument.

Azure CLICopy

Try It

kubectl get service azure-vote-front --watch

The **EXTERNAL-IP** output for the azure-vote-front service will initially show as pending.

OutputCopy

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

azure-vote-front LoadBalancer 10.0.37.27 <pending> 80:30572/TCP 6s

Once the **EXTERNAL-IP** address changes from pending to an actual public IP address, use CTRL-C to stop the kubectl watch process. The following example output shows a valid public IP address assigned to the service:

OutputCopy

azure-vote-front LoadBalancer 10.0.37.27 52.179.23.131 80:30572/TCP 2m

To see the Azure Vote app in action, open a web browser to the external IP address of your service.

View the cluster nodes' and pods' health metrics captured by [Azure Monitor for containers](https://docs.microsoft.com/en-us/azure/azure-monitor/containers/container-insights-overview) in the Azure portal.

## Delete the cluster

To avoid Azure charges, clean up your unnecessary resources. Use the [az group delete](https://docs.microsoft.com/en-us/cli/azure/group" \l "az_group_delete) command to remove the resource group, container service, and all related resources.

Azure CLICopy

Try It

az group delete --name myResourceGroup --yes --no-wait

**Note**

When you delete the cluster, the Azure Active Directory service principal used by the AKS cluster is not removed. For steps on how to remove the service principal, see [**AKS service principal considerations and deletion**](https://docs.microsoft.com/en-us/azure/aks/kubernetes-service-principal#additional-considerations).

If you used a managed identity, the identity is managed by the platform and does not require removal.

## Get the code

Pre-existing container images were used in this quickstart to create a Kubernetes deployment. The related application code, Dockerfile, and Kubernetes manifest file are [available on GitHub.](https://github.com/Azure-Samples/azure-voting-app-redis.git)