Tutorial: Deploy an Azure Kubernetes Service (AKS) cluster

Article • 08/10/2021 • 4 minutes to read • 19 contributors

In this article

Kubernetes provides a distributed platform for containerized applications. With AKS, you can quickly create a production ready Kubernetes cluster. In this tutorial, part three of seven, a Kubernetes cluster is deployed in AKS. You learn how to:

- ✓ Deploy a Kubernetes AKS cluster that can authenticate to an Azure container registry
- ✓ Install the Kubernetes CLI (kubectl)
- ✓ Configure kubectl to connect to your AKS cluster

In later tutorials, the Azure Vote application is deployed to the cluster, scaled, and updated.

Before you begin

In previous tutorials, a container image was created and uploaded to an Azure Container Registry instance. If you haven't done these steps, and would like to follow along, start at Tutorial 1 – Create container images.

Azure CLI | Azure PowerShell

This tutorial requires that you're running the Azure CLI version 2.0.53 or later. Run `az --version` to find the version. If you need to install or upgrade, see Install Azure CLI.

Create a Kubernetes cluster

AKS clusters can use Kubernetes role-based access control (Kubernetes RBAC). These controls let you define access to resources based on roles assigned to users. Permissions are combined if a user is assigned multiple roles, and permissions can be scoped to either a single namespace or across the whole cluster. By default, the Azure CLI automatically enables Kubernetes RBAC when you create an AKS cluster.

Azure CLI Azure PowerShell

Create an AKS cluster using az aks create. The following example creates a cluster named *myAKSCluster* in the resource group named *myResourceGroup*. This resource group was created in the previous tutorial in the *eastus* region. The following example does not specify a region so the AKS cluster is also created in the *eastus* region. For more information, see Quotas, virtual machine size restrictions, and region availability in Azure Kubernetes Service (AKS) for more information about resource limits and region availability for AKS.

To allow an AKS cluster to interact with other Azure resources, a cluster identity is automatically created, since you did not specify one. Here, this cluster identity is granted the right to pull images from the Azure Container Registry (ACR) instance you created in the previous tutorial. To execute the command successfully, you're required to have an **Owner** or **Azure account administrator** role on the Azure subscription.

Azure CLI Copy

```
az aks create \
    --resource-group myResourceGroup \
    --name myAKSCluster \
    --node-count 2 \
    --generate-ssh-keys \
    --attach-acr <acrName>
```

To avoid needing an **Owner** or **Azure account administrator** role, you can also manually configure a service principal to pull images from ACR. For more information, see ACR authentication with service principals or Authenticate from Kubernetes with a pull secret. Alternatively, you can use a managed identity instead of a service principal for easier management.

After a few minutes, the deployment completes, and returns JSON-formatted information about the AKS deployment.

Note

To ensure your cluster to operate reliably, you should run at least 2 (two) nodes.

Install the Kubernetes CLI

To connect to the Kubernetes cluster from your local computer, you use kubectl , the Kubernetes command-line client.

Azure CLI | Azure PowerShell

If you use the Azure Cloud Shell, `kubectl` is already installed. You can also install it locally using the az aks install-cli command:

Azure CLI — Copy

```
az aks install-cli
```

Connect to cluster using kubectl

Azure CLI | Azure PowerShell

To configure `kubectl` to connect to your Kubernetes cluster, use the az aks get-credentials command. The following example gets credentials for the AKS cluster named *myAKSCluster* in the *myResourceGroup*:

Azure CLI — Copy

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

To verify the connection to your cluster, run the kubectl get nodes command to return a list of the cluster nodes:

Azure CLI — Copy — Try It

```
kubectl get nodes
```

The following example output shows the list of cluster nodes.

Copy

```
$ kubectl get nodes

NAME                                STATUS   ROLES   AGE     VERSION
aks-nodepool1-37463671-vmss000000   Ready    agent   2m37s   v1.18.10
aks-nodepool1-37463671-vmss000001   Ready    agent   2m28s   v1.18.10
```

Next steps

In this tutorial, a Kubernetes cluster was deployed in AKS, and you configured `kubectl` to connect to it. You learned how to:

- ✓ Deploy a Kubernetes AKS cluster that can authenticate to an Azure container registry
- ✓ Install the Kubernetes CLI (kubectl)
- ✓ Configure kubectl to connect to your AKS cluster

Advance to the next tutorial to learn how to deploy an application to the cluster.

Deploy application in Kubernetes

Recommended content

Kubernetes on Azure tutorial - Deploy an application - Azure Kubernetes Service

In this Azure Kubernetes Service (AKS) tutorial, you deploy a multi-container application to your cluster using a custom image stored in Azure Container Registry.

Kubernetes on Azure tutorial - Create a container registry - Azure Kubernetes Service

In this Azure Kubernetes Service (AKS) tutorial, you create an Azure Container Registry instance and upload a sample application container image.

Kubernetes on Azure tutorial - Prepare an application - Azure Kubernetes Service

In this Azure Kubernetes Service (AKS) tutorial, you learn how to prepare and build a multi-container app with Docker Compose that you can then deploy to AKS.

Kubernetes on Azure tutorial - Update an application - Azure Kubernetes Service

In this Azure Kubernetes Service (AKS) tutorial, you learn how to update an existing application deployment to AKS with a new version of the application code.

Kubernetes on Azure tutorial - Upgrade a cluster - Azure Kubernetes Service

In this Azure Kubernetes Service (AKS) tutorial, you learn how to upgrade an existing AKS cluster to the latest available Kubernetes version.

Quickstart: Deploy an AKS cluster by using Azure CLI - Azure Kubernetes Service

Learn how to quickly create a Kubernetes cluster, deploy an application, and monitor performance in Azure Kubernetes Service (AKS) using the Azure CLI.

Build, test, and deploy containers to Azure Kubernetes Service using GitHub Actions - Azure Kubernetes Service

Learn how to use GitHub Actions to deploy your container to Kubernetes

Kubernetes on Azure tutorial - Scale Application - Azure Kubernetes Service

In this Azure Kubernetes Service (AKS) tutorial, you learn how to scale nodes and pods in Kubernetes, and implement horizontal pod autoscaling.

Show more