Quickstart: Deploy a container instance in Azure using Bicep

Article • 04/09/2023

Use Azure Container Instances to run serverless Docker containers in Azure with simplicity and speed. Deploy an application to a container instance on-demand when you don't need a full container orchestration platform like Azure Kubernetes Service. In this quickstart, you use a Bicep file to deploy an isolated Docker container and make its web application available with a public IP address.

Bicep is a domain-specific language (DSL) that uses declarative syntax to deploy Azure resources. It provides concise syntax, reliable type safety, and support for code reuse. Bicep offers the best authoring experience for your infrastructure-as-code solutions in Azure.

Prerequisites

If you don't have an Azure subscription, create a free account before you begin.

Review the Bicep file

The Bicep file used in this quickstart is from Azure Quickstart Templates .

Bicep

```
@description('Name for the container group')
param name string = 'acilinuxpublicipcontainergroup'

@description('Location for all resources.')
param location string = resourceGroup().location

@description('Container image to deploy. Should be of the form repoN-
ame/imagename:tag for images stored in public Docker Hub, or a fully
qualified URI for other registries. Images from private registries
require additional registry credentials.')
param image string = 'mcr.microsoft.com/azuredocs/aci-helloworld'

@description('Port to open on the container and the public IP
address.')
param port int = 80

@description('The number of CPU cores to allocate to the container.')
param cpuCores int = 1
```

```
@description('The amount of memory to allocate to the container in
gigabytes.')
param memoryInGb int = 2

@description('The behavior of Azure runtime if container has
stopped.')
@allowed([
  'Always'
  'Never'
  'OnFailure'
])
param restartPolicy string = 'Always'

resource containerGroup
'Microsoft.ContainerInstance/containerGroups@2021-09-01' = {
  name: name
  location: location
  properties: {
    containers: [
      {
        name: name
        properties: {
          image: image
          ports: [
            {
              port: port
              protocol: 'TCP'
            }
          ]
          resources: {
            requests: {
              cpu: cpuCores
              memoryInGB: memoryInGb
            }
          }
        }
      }
    ]
    osType: 'Linux'
    restartPolicy: restartPolicy
    ipAddress: {
      type: 'Public'
      ports: [
        {
          port: port
          protocol: 'TCP'
        }
      ]
    }
  }
}

output containerIPv4Address string = containerGroup.properties.ipAd-
dress.ip
```

The following resource is defined in the Bicep file:

- **Microsoft.ContainerInstance/containerGroups**: create an Azure container group. This Bicep file defines a group consisting of a single container instance.

More Azure Container Instances template samples can be found in the quickstart template gallery .

Deploy the Bicep file

1. Save the Bicep file as **main.bicep** to your local computer.
2. Deploy the Bicep file using either Azure CLI or Azure PowerShell.

CLI

Azure CLI

```
az group create --name exampleRG --location eastus
az deployment group create --resource-group exampleRG --tem-
plate-file main.bicep
```

When the deployment finishes, you should see a message indicating the deployment succeeded.

Review deployed resources

Use the Azure portal, Azure CLI, or Azure PowerShell to list the deployed resources in the resource group.

CLI

Azure CLI

```
az resource list --resource-group exampleRG
```

View container logs

Viewing the logs for a container instance is helpful when troubleshooting issues with your container or the application it runs. Use the Azure portal, Azure CLI, or Azure

PowerShell to view the container's logs.

CLI

Azure CLI

```
az container logs --resource-group exampleRG --name
acilinuxpublicipcontainergroup
```

> **Note**
>
> It may take a few minutes for the HTTP GET request to generate.

Clean up resources

When no longer needed, use the Azure portal, Azure CLI, or Azure PowerShell to delete the container and all of the resources in the resource group.

CLI

Azure CLI

```
az group delete --name exampleRG
```

Next steps

In this quickstart, you created an Azure container instance using Bicep. If you'd like to build a container image and deploy it from a private Azure container registry, continue to the Azure Container Instances tutorial.

Tutorial: Create a container image for deployment to Azure Container Instances