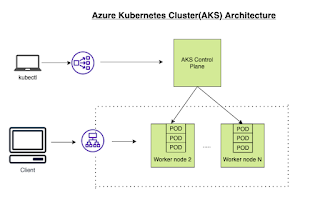
**How to create AKS cluster using Terraform | Create Kubernetes Cluster using Terraform | How to Create Azure Kubernetes Cluster using Terraform**

[](https://1.bp.blogspot.com/-Ywoolp8yG4w/YUCjvFQqkgI/AAAAAAAADrs/QwUxj_DfPNseCm-gZ_-IxlzdNPM1SYSIgCLcBGAsYHQ/s664/Screen%2BShot%2B2021-09-14%2Bat%2B8.29.01%2BAM.png)

**What is Azure Kubernetes Service (AKS)**

Azure Kubernetes Service (AKS) is a managed container orchestration service, based on the open source [Kubernetes system](https://searchitoperations.techtarget.com/definition/Google-Kubernetes), which is available on the Microsoft Azure public cloud. AKS allows you to quickly deploy a production ready Kubernetes cluster in Azure, deploy and manage containerized applications more easily with a fully managed Kubernetes service. We will see how to create AKS cluster in Azure cloud using Terraform.

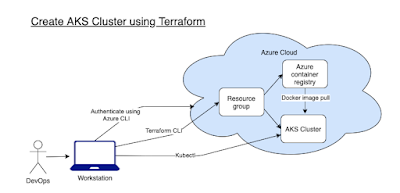
AKS cluster can be created by many ways as mentioned below:

1. [Create AKS cluster in Azure portal](https://www.coachdevops.com/2020/09/how-to-create-aks-cluster-in-azure.html) directly

2. [Create AKS cluster using Azure CLI](https://www.coachdevops.com/2020/10/how-to-create-aks-cluster-using-aure.html)

3. [Create AKS cluster using Terraform.](https://www.coachdevops.com/2021/09/how-to-create-aks-cluster-using.html)

Creating an AKS resource with Terraform is incredibly easy, it only requires a single resource azurerm\_kubernetes\_cluster and in this post, we are going to walk through the necessary steps to create this with Terraform. We will create ACR and create a role with ACRpull assignment as well

[](https://1.bp.blogspot.com/-Nmor1kN3MVM/YUCimP7xXyI/AAAAAAAADrk/04lU2vmiK4UxNJt-XmU53lTLJceG0kiIgCLcBGAsYHQ/s731/Screen%2BShot%2B2021-09-14%2Bat%2B8.24.04%2BAM.png)

**Pre-requisites:**

* [Terraform](https://www.cidevops.com/2020/04/how-to-install-terraform-on-mac-os.html) is installed on your machine.
* Azure subscription
* [Kubectl](https://www.coachdevops.com/2021/09/install-kubectl-on-mac-os-how-to.html) is installed on your machine
* [Azure cli](https://www.coachdevops.com/2021/09/install-azure-cli-in-mac-os-how-to.html) is installed

**Login to Azure using credentials**

Make sure you are login to Azure portal first.

az login

Choose your Microsoft credentials.

Let's create following tf files using Visual studio Code:

1. Variables.tf - where we will define the variables used in main.tf  
2. terraform.tfvars - Declare the values for the variables

3. providers.tf - declare the providers with version

4. main.tf - main configuration file with all the resources which will be created

5. output.tf - Export some data to output file

**create providers.tf**

provider "azurerm" {

  features {}

}

terraform {

  required\_providers {

    azurerm = {

      source  = "hashicorp/azurerm"

      version = "3.62.1"

    }

  }

}

**create variables.tf**

variable "resource\_group\_name" {  
  type        = string  
  description = "RG name in Azure"  
}  
variable "location" {  
  type        = string  
  description = "Resources location in Azure"  
}  
variable "cluster\_name" {  
  type        = string  
  description = "AKS name in Azure"  
}  
variable "kubernetes\_version" {  
  type        = string  
  description = "Kubernetes version"  
}  
variable "system\_node\_count" {  
  type        = number  
  description = "Number of AKS worker nodes"  
}  
variable "acr\_name" {  
  type        = string  
  description = "ACR name"  
}

**create terraform.tfvars**

resource\_group\_name = "aks\_tf\_rg"

location            = "CentralUS"

cluster\_name        = "my-aks-cluster"

kubernetes\_version  = "1.26.3"

system\_node\_count   = 2

acr\_name            = "myacr321012"

**create main.tf**

#In Azure, all infrastructure elements such as virtual machines, storage, and our Kubernetes cluster need to be attached to a resource group.

resource "azurerm\_resource\_group" "aks-rg" {

  name     = var.resource\_group\_name

  location = var.location

}

resource "azurerm\_role\_assignment" "role\_acrpull" {

  scope                            = azurerm\_container\_registry.acr.id

  role\_definition\_name             = "AcrPull"

  principal\_id                     = azurerm\_kubernetes\_cluster.aks.kubelet\_identity.0.object\_id

  skip\_service\_principal\_aad\_check = true

}

resource "azurerm\_container\_registry" "acr" {

  name                = var.acr\_name

  resource\_group\_name = azurerm\_resource\_group.aks-rg.name

  location            = var.location

  sku                 = "Standard"

  admin\_enabled       = false

}

resource "azurerm\_kubernetes\_cluster" "aks" {

  name                = var.cluster\_name

  kubernetes\_version  = var.kubernetes\_version

  location            = var.location

  resource\_group\_name = azurerm\_resource\_group.aks-rg.name

  dns\_prefix          = var.cluster\_name

  default\_node\_pool {

    name                = "system"

    node\_count          = var.system\_node\_count

    vm\_size             = "Standard\_DS2\_v2"

    type                = "VirtualMachineScaleSets"

    zones  = [1, 2, 3]

    enable\_auto\_scaling = false

  }

  identity {

    type = "SystemAssigned"

  }

  network\_profile {

    load\_balancer\_sku = "standard"

    network\_plugin    = "kubenet"

  }

}

**create output.tf**

output "aks\_id" {

  value = azurerm\_kubernetes\_cluster.aks.id

}

output "aks\_fqdn" {

  value = azurerm\_kubernetes\_cluster.aks.fqdn

}

output "aks\_node\_rg" {

  value = azurerm\_kubernetes\_cluster.aks.node\_resource\_group

}

output "acr\_id" {

  value = azurerm\_container\_registry.acr.id

}

output "acr\_login\_server" {

  value = azurerm\_container\_registry.acr.login\_server

}

resource "local\_file" "kubeconfig" {

  depends\_on   = [azurerm\_kubernetes\_cluster.aks]

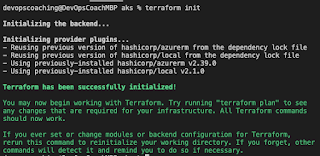
  filename     = "kubeconfig"

  content      = azurerm\_kubernetes\_cluster.aks.kube\_config\_raw

}

**Run terraform commands**

terraform init

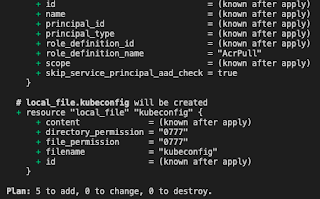
**[](https://1.bp.blogspot.com/-kWNqGLqz2Qs/YUAVcuS5L4I/AAAAAAAADq0/mmi4MxCotuAkWh-p_ATX_bb53urGtF3nwCLcBGAsYHQ/s567/Screen%2BShot%2B2021-09-13%2Bat%2B10.20.22%2BPM.png)**

terraform validate

[](https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEhhpluqavmFrNiEvukcgl-QrAfu4GWUt2OitijsVj4PPjq4GI48QPP3ucnK1MpqfQxLUdK6fzK-i8b6ilGyUn67CxDoZaL5Ymc458yRJZNUyLmQRY_pvDBkpQ6UM5vLVs37yOW0fz30Sfnn3DUFNvBuIHa_evqSElUokcyS3ZdL2lz1aEEnKLe33G63830/s1264/Screenshot%202023-06-23%20at%209.52.38%20PM.png)

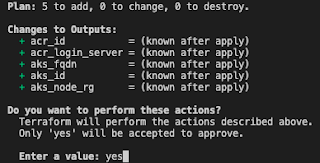
just to make sure syntax is right..

terraform plan

[](https://1.bp.blogspot.com/-k3xos0UOhKk/YUAVlVJStKI/AAAAAAAADq4/nsEs9jPkOXIIBXvr0RynWmWCn_PAx79cgCLcBGAsYHQ/s456/Screen%2BShot%2B2021-09-13%2Bat%2B10.22.47%2BPM.png)

terraform apply

and type **yes**

[](https://1.bp.blogspot.com/-8Ow1LQo8zXY/YUAV3Q0pvCI/AAAAAAAADrE/kxQ6vKB1VxwJmoh_bY74_svLHlwvI3gDwCLcBGAsYHQ/s388/Screen%2BShot%2B2021-09-13%2Bat%2B10.24.04%2BPM.png)

You will see following resources are created:

[](https://1.bp.blogspot.com/-3W9O1ytodGI/YUAXkIAAgSI/AAAAAAAADrM/e-hxyPDPf3IOytijiknclC74XFZdqrZoACLcBGAsYHQ/s1105/Screen%2BShot%2B2021-09-13%2Bat%2B10.30.51%2BPM.png)

**Move the generated Kubeconfig file to ~/.kube/config**

mv kubeconfig ~/.kube/config

To verify if worker nodes are created, use the kubectl get nodes command to return a list of the cluster nodes.

kubectl get nodes

[](https://1.bp.blogspot.com/-JZSqELw7KxI/YUAYOPIKSOI/AAAAAAAADrU/MEwaft0oJ6INlaVRt_1fWghtGCuH-73mQCLcBGAsYHQ/s702/Screen%2BShot%2B2021-09-13%2Bat%2B10.33.47%2BPM.png)

You will see worker nodes with health status ready.

Let's deploy some apps into AKS cluster.

**Deploy Nginx App**

kubectl create -f https://raw.githubusercontent.com/kubernetes/website/master/content/en/examples/controllers/nginx-deployment.yaml

Once the deployment is created, use kubectl to check on the deployments by running this command:

kubectl get deployments

[](https://1.bp.blogspot.com/-QdR0_7-4he8/X3YGflxIxNI/AAAAAAAADD0/EvUhf7QstSsEEaXjVm3yFLz4jbwsfvk7wCLcBGAsYHQ/s886/deploymens.png)

**To see the list of pods**

kubectl get pods

[](https://1.bp.blogspot.com/-Auc6y8gqMiM/YUC1K4GtjUI/AAAAAAAADr0/RaURD6A5QSUqhXh8MJH6fTd3x2GLiiViQCLcBGAsYHQ/s547/Screen%2BShot%2B2021-09-14%2Bat%2B9.43.40%2BAM.png)

**Perform cleanup by deleting the AKS cluster**

To avoid Azure charges, you should clean up unneeded resources. When the cluster is no longer needed, use terraform destroy command to remove the resource group, AKS cluster service, and all related resources.

terraform destroy --auto-approve