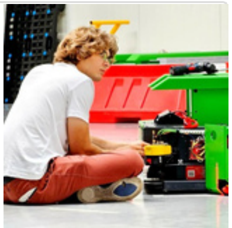
Kubernetes Azure: Deploy AKS microservice app with RabbitMQ using terraform

Check GitHub for helpful DevOps tools:

Michael Robotics

Hi, I'm Michal. I'm a Robotics Engineer and DevOps enthusiast. My mission is to create skill-learning platform that combats information overload by adhering to the set of principles: simplify, prioritize, and execute.

https://github.com/MichaelRobotics



Ask Personal AI Document assistant to learn interactively (FASTER)!

1 Download PDF

2 Go to website

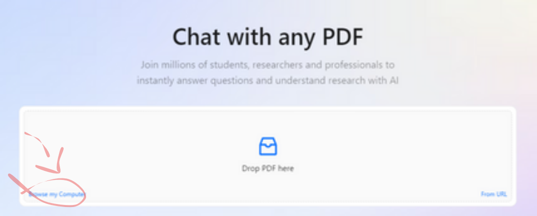
3 Browse file

4 Chat with Document

1 https://github.com/MichaelRobotics/DevOpsTools/blob/main/KubernetesAzure.pdf

Click there to go to ChatPdf website 2

3



Chat with any PDF

Join millions of students, researchers and professionals to instantly answer questions and understand research with AI

Drop PDF here

Browse my Computer

From URL

Ask questions about document! 4

Complety new to Linux and Networking?

Essential for this PDF is a thorough knowledge of networking. I highly recommend the HTB platform's networking module, which offers extensive information to help build a comprehensive understanding.

HTB - Your Cyber Performance Center

We provide a human-first platform creating and maintaining high performing cybersecurity individuals and organizations.

https://www.hackthebox.com/

What is Kubernetes?

Kubernetes is an open-source platform that automates the deployment, scaling, and management of containerized applications. It helps manage clusters of nodes running containers, ensuring efficient and reliable operation.

How Kubernetes clusters are made?

Kubernetes clusters consist of a control plane and multiple worker nodes. The control plane manages cluster operations, while worker nodes run the actual container workloads.

Why and When use Kubernetes

Kubernetes is ideal for deploying scalable, resilient, and automated containerized applications. It is used when managing multiple containers across different environments is necessary.

Example: Running a microservices-based e-commerce platform that scales up during peak hours.

System Requirements

- RAM: 2 GB per node (1 GB can work for testing but may lead to limited performance)
- 10 GB free storage
- Ubuntu

Kubernetes: Main components & packages

- **kube-apiserver:** Central management component that exposes the Kubernetes API; acts as the front-end for the cluster.
- **etcd:** Distributed key-value store for storing all cluster data, ensuring data consistency across nodes.
- **kube-scheduler:** Assigns pods to available nodes based on resource requirements and policies.
- **kube-controller-manager:** Manages core controllers that handle various functions like node status, replication, and endpoints.
- **kubelet:** Agent that runs on each node, responsible for managing pods and their containers.
- **kube-proxy:** Manages networking on each node, ensuring communication between pods and services within the cluster.

Kubernetes Azure: Project Introduction

1) What is AKS

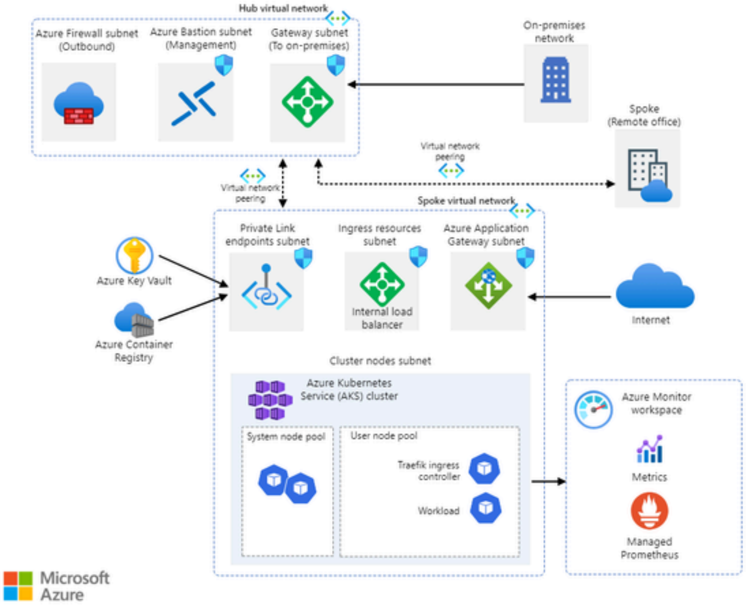
Azure Kubernetes Service (AKS) is a managed Kubernetes service that lets you quickly deploy and manage clusters.

2) Overview

This Project consits of:

- Deployment of AKS cluster using Terraform.
- Running a sample multi-container application with a group of microservices and web front ends simulating a retail scenario.

Before deploying a production-ready cluster, its recommend to you familiarize yourself with Azure **baseline reference architecture** to consider how it aligns with your business requirements.



Kubernetes Azure: Environment setup

1) Create account / login to Azure

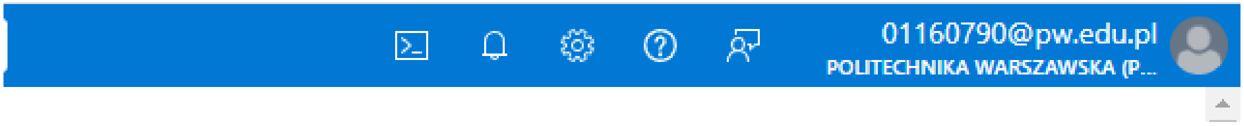
If you dont have an account already, create new and start free trial.

Azure

AZURE. LIMITLESS INNOVATION.

https://azure.microsoft.com/en-us/

Use Azure CLI to manage resources. It comes with preinstalled software used in this tutorial.



Download repository

```
git clone https://github.com/MichaelRobotics/Kubernetes.git
cd AKS
```

Kubernetes GCP: Provision AKS with Terraform code

1) ssh.tf

Automates creating and managing an SSH key pair using the azapi provider:

- Random Name Generation: Generates a unique SSH key name with random_pet, prefixed with "ssh".
- Azure SSH Public Key Resource: Creates an azapi_resource for Microsoft.Compute/sshPublicKeys, linked to the resource group.
- Key Pair Generation: Executes a generateKeyPair action via azapi_resource_action, triggering Azure to create the keys.
- Extracting Keys: Captures the generated public and private keys, exporting the public key for use.
- Output: Makes the public key available as an output variable for other resources or applications.

2) variables.tf

- Resource Group Location: Specifies the Azure region for the resource group. Defaults to "eastus".
- Resource Group Name Prefix: Sets a prefix ("rg") for the resource group name, ensuring uniqueness by appending a random ID.
- Node Count: Defines the initial number of nodes in a node pool, with a default value of 3.
- Managed Service Identity (MSI) ID: Allows specifying an MSI ID for authentication when using Managed Identity. Defaults to null if not needed.
- Username: Sets the admin username for the cluster or virtual machine. Defaults to "azureadmin".

3) main.tf

provisions an Azure Kubernetes Service (AKS) cluster with supporting resources:

- Azure Resource Group: Creates an Azure resource group using the generated name and the location defined in resource_group_location.
- Azure Kubernetes Cluster: Provisions an AKS cluster with the following configurations:
    - **Location** and Resource Group: Aligns with the created resource group.
    - **Identity**: Assigns a system-managed identity to the cluster.
    - **Node Pool**: Configures a default node pool named "agentpool", using the Standard_D2_v2 VM size and the node count defined in node_count.
    - **Linux Profile**: Sets the admin username from username and assigns an SSH public key generated by the generateKeyPair action.
    - **Network Profile**: Uses the kubenet network plugin and a standard load balancer.

Initialize Terraform

```
terraform plan -out main.tfplan
```

Create a terraform execution plan

```
terraform apply main.tfplan
```

Apply a terraform execution plan

```
kubectl apply -f aks-store-quickstart.yaml
```

Kubernetes GCP: Deploy application

1) Setup kubectl

Get the Kubernetes configuration from the Terraform state and store it in a file that kubectl can read using the following command.

```
echo "$(terraform output kube_config)" > ./azurek8s
```

Verify the previous command didn't add an ASCII EOT character using the following command.

```
cat ./azurek8s
```

If you see << EOT at the beginning and EOT at the end, remove these characters

Set an environment variable so kubectl can pick up the correct config using the following command.

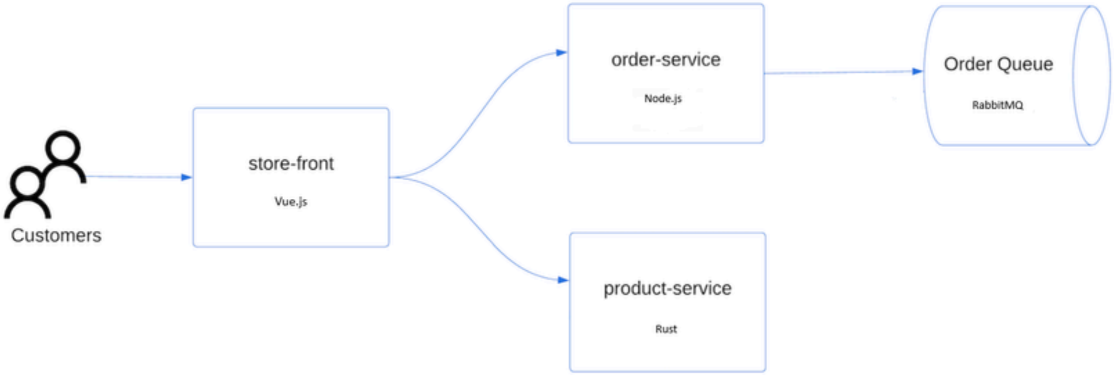
```
export KUBECONFIG=./azurek8s
```

Verify the health of the cluster using the kubectl get nodes command.

```
kubectl get nodes
```

2) App

k8s folder with manifests includs following Kuebrnetes deployments and services:



- **Store front**: Web application for customers to view products and place orders.
- **Product service**: Shows product information.
- **Order service**: Places orders.
- **Rabbit MQ**: Message queue for an order queue.

3) Deploy

Apply manifests inside k8s repo:

```
kubectl apply -f /k8s
```

Output should be as shown:

```
deployment.apps/rabbitmq created
service/rabbitmq created
deployment.apps/order-service created
service/order-service created
deployment.apps/product-service created
service/product-service created
deployment.apps/store-front created
service/store-front created
```

4) Test application

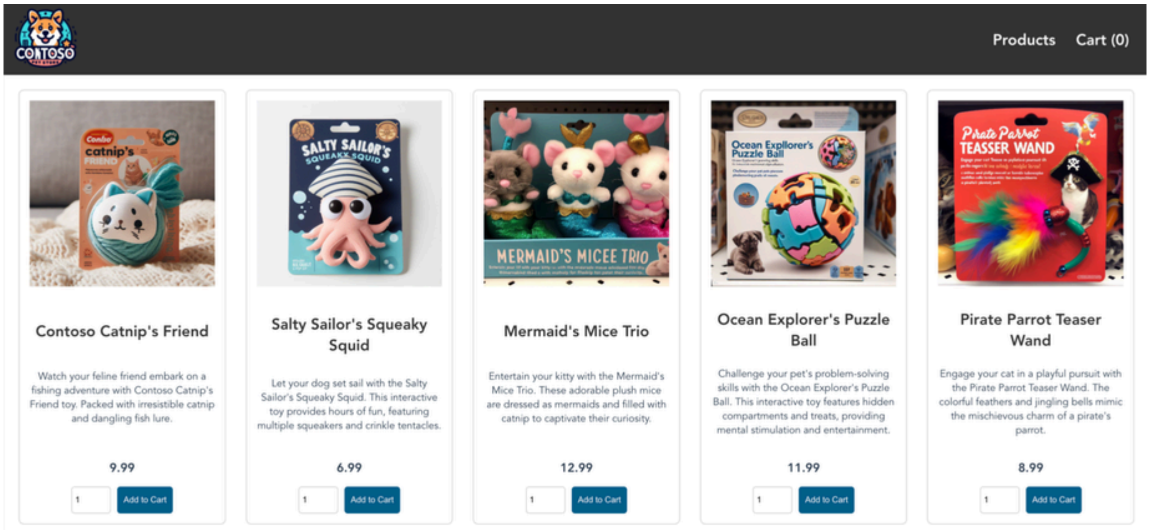
Check the status of the deployed pods. Make sure all pods are in running state

```
kubectl get pods
```

Check for a public IP address

```
kubectl get service store-front --watch
```

Once the EXTERNAL-IP address changes from pending to an actual public IP address, use CTRL-C to stop the kubectl watch process. Open a web browser to the external IP address of your service to see the Azure Store app in action.



Contoso Catnip's Friend

Watch your feline friend embark on a fishing adventure with Contoso Catnip's Friend toy. Packed with irresistible catnip and dangling fish lure.

9.99

Salty Sailor's Squeaky Squid

Let your dog set sail with the Salty Sailor's Squeaky Squid. This interactive toy provides hours of fun, featuring multiple squeakers and crinkle tentacles.

6.99

Mermaid's Mice Trio

Entertain your kitty with the Mermaid's Mice Trio. These adorable plush mice are dressed as mermaids and filled with catnip to captivate their curiosity.

12.99

Ocean Explorer's Puzzle Ball

Challenge your pet's problem-solving skills with the Ocean Explorer's Puzzle Ball. This interactive toy features hidden compartments and treats, providing mental stimulation and entertainment.

11.99

Pirate Parrot Teaser Wand

Engage your cat in a playful pursuit with the Pirate Parrot Teaser Wand. The colorful feathers and jingling bells mimic the mischievous charm of a pirate's parrot.

8.99

Kubernetes GCP: Delete AKS resources

Run terraform plan and specify the destroy flag.

```
terraform plan -destroy -out main.destroy.tfplan
```

Run terraform apply to apply the execution plan.

```
terraform apply main.destroy.tfplan
```

common troubleshooting

1) Terraform Apply Fails with Authentication Errors

Cause: Azure credentials are missing or misconfigured.
Solution: Ensure the az login command has been run and the Terraform Azure provider is properly configured with the correct subscription. Verify the ARM_CLIENT_ID, ARM_CLIENT_SECRET, and ARM_SUBSCRIPTION_ID environment variables if using a service

2) RabbitMQ Pods Stuck in Pending State

Cause: Insufficient node resources or no suitable node available for scheduling
Solution: Use kubectl describe pod to check events. Ensure the node pool has enough resources or scale it up. Check if taints or affinity rules are preventing scheduling

3) Terraform Apply Fails with Resource Group or Namespace Issues

Cause: The resource group or namespace required by AKS or RabbitMQ does not exist or is incorrectly referenced.
Solution: Verify the resource group exists using az group show. Confirm namespace creation and references in the manifests.

4) Check my Kubernetes Troubleshooting series

Michael Robotics

Hi, I'm Michal. I'm a Robotics Engineer and DevOps enthusiast. My mission is to create skill-learning platform that combats skill information overload by adhering to the set of principles: simplify, prioritize, and execute.

https://github.com/MichaelRobotics

Learn more about Kubernetes

Check Kubernetes and piyushsachdeva - great docs!

Setup a Multi Node Kubernetes Cluster

kubeadm is a tool to bootstrap the Kubernetes cluster

https://github.com/piyushsachdeva/CKA-2024/tree/main/Resources/Day27

Kubernetes Documentation

This section lists the different ways to set up and run Kubernetes

https://kubernetes.io/docs/setup/

Share, comment, DM and check GitHub for scripts & playbooks created to automate process.

Check my GitHub

Michael Robotics

Hi, I'm Michal. I'm a Robotics Engineer and DevOps enthusiast. My mission is to create skill-learning platform that combats skill information overload by adhering to the set of principles: simplify, prioritize, and execute.

https://github.com/MichaelRobotics

PS.

If you need a playbook or bash script to manage KVM on a specific Linux distribution, feel free to ask me in the comments or send a direct message!