Week 3 - Basic concepts behind Kubernetes and Amazon Elastic Kubernetes Service (Amazon EKS)

Introduction to Kubernetes

Kubernetes is a popular, open-source system for managing containerized workloads and services.

With Kubernetes, you interact with the Kubernetes API to describe your desired state. Kubernetes works to make changes in your cluster to meet the desired state. Kubernetes facilitates a declarative configuration.

Kubernetes concepts

A Kubernetes cluster is made up of worker nodes, where your containers run. You interact with a Kubernetes API inside the control plane nodes to describe the desired state of containers and services in your cluster. The control plane coordinates the containers that run on worker nodes.

You use a tool (such as kubectl) to create objects that describe your desired state. For example, a Pod is the smallest deployable unit in Kubernetes. A Pod is the home for one or more containers running on your worker nodes. You probably aren't creating Pods directly. You can use a workload resource (such as a deployment) to describe your requirements to Kubernetes. From there, Kubernetes handles a lot of the operational work for you.

Kubernetes scaling and service discovery

Kubernetes is designed to scale with your workloads. You can scale in two different dimensions: the number of Pods hosting your application, and the numbers of nodes that form your cluster. Kubernetes offers many options for scaling. A HorizontalPodAutoscaler can observe metrics to increase (or decrease) the numbers of Pods for a service. A VerticalPodAutoscaler can automatically adjust the resource requirements for your Pods to better inform the Kubernetes scheduler.

Kubernetes Cluster Autoscaler and Karpenter can intelligently grow and shrink your worker nodes.

Imagine that you are hosting a service over multiple Pods—the number of Pods might grow or shrink. How will dependent applications discover the location to communicate with your services?

Services inside a Kubernetes cluster are accessed through their clusterIP address. The clusterIP is a load balanced IP address. Traffic to this IP address will be forwarded to the matching Pods for the service. The clusterIP is proxied by the kube-proxy that runs on each of your Kubernetes nodes. kube-proxy contains rules that are updated as Pods and services are created and removed.

The clusterIP can be discovered by using environment variables set by Kubernetes, or by Domain Name System (DNS)-based service discovery. For more information about discovering services, see Discovering services in the Kubernetes documentation.

Amazon EKS

For more detailed information about Amazon Elastic Kubernetes Service (Amazon EKS), see the AWS re:Invent 2021 Deep Dive on Amazon EKS.

Upgrading your Amazon EKS cluster

For more information about how the upgrade process works for Amazon EKS, see Updating a cluster.

EKS Best Practices Guide

For more information about best practices for working with Amazon EKS, see the EKS Best Practices Guide.

Kubernetes objects

In the demonstration, we showed how to apply a few different types of objects to a Kubernetes cluster. For more information about Kubernetes objects, see Understanding Kubernetes Objects in the Kubernetes documentation.

Pod networking

Amazon EKS supports native VPC networking with the Amazon VPC Container Network Interface (CNI) plugin for Kubernetes. This plugin assigns a private IPv4 or IPv6 address from

your virtual private cloud (VPC) to each Pod. We saw these Pods run on the EKS cluster after it was created. For more information about Pod networking with Amazon EKS, see Pod networking using the Amazon VPC Container network interface (CNI) plugin.

Logging

For more general information about logging with Kubernetes, see Logging Architecture.

For more information about Amazon EKS control plane logging, see Amazon EKS control plane logging.

EKS Workshop

For more information about examples and tutorials, see EKS Workshop.

Debugging Amazon EKS

When you need to diagnose issues in your Amazon Elastic Kubernetes Service (Amazon EKS) workloads, you can use some kubectl commands.

- kubectl logs: View the logs of a container in a Pod.
- kubectl port-forward: Connect ports on your local host to ports on a container.
- kubectl exec: Run commands inside a container.

Additional resources

For more information about kubectl commands, see the Kubectl Reference Docs in the Kubernetes documentation.

For more detailed information about the Kubernetes control plane and logging, see Amazon EKS control plane logging.

For more information about additional troubleshooting steps, see Amazon EKS troubleshooting.