Day 30 Task: Kubernetes Architecture

Kubernetes Overview

Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It was originally developed by Google and is now maintained by the Cloud Native Computing Foundation (CNCF).

Kubernetes enables users to deploy containerized applications across a cluster of nodes, providing automatic scaling and management of the application's resources. It abstracts away the underlying infrastructure, allowing users to focus on building and running their applications without having to worry about the underlying infrastructure.

Kubernetes architecture is based on a master-worker node model. The master node runs the Control Plane components such as the API Server, etcd, and the scheduler, which manage the overall state of the cluster. Worker nodes run the application workloads and are managed by the Control Plane.

Tasks

1. What is Kubernetes? Write in your own words and why do we call it k8s?

   Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It was originally developed by Google and is now maintained by the Cloud Native Computing Foundation (CNCF).

   In Kubernetes, applications are deployed in containers, which are lightweight and portable units of software that can run consistently across different computing environments. Kubernetes provides a powerful set of features for managing containers, including automatic scaling, rolling updates, load balancing, and self-healing.

   The name "Kubernetes" comes from the Greek word for "helmsman" or "pilot", reflecting its role in steering and managing containerized applications. The abbreviation "k8s" is used as a shorthand for "Kubernetes" because there are eight letters between the "K" and the "s". This abbreviation is commonly used in documentation, online discussions, and social media posts related to Kubernetes.

   Overall, Kubernetes has become a popular platform for deploying and managing containerized applications, thanks to its scalability, flexibility, and rich set of features. It has become a key technology in the cloud-native ecosystem and is widely used in modern application development and deployment.

2. What are the benefits of using k8s?

   There are many benefits to using Kubernetes for deploying and managing containerized applications. Some of the key benefits include:

   1. Scalability: Kubernetes enables applications to scale seamlessly based on demand, by adding or removing container instances as needed. This helps ensure that applications can handle increased traffic and workloads without downtime or performance issues.

   2. Fault tolerance: Kubernetes provides built-in mechanisms for detecting and recovering from application and infrastructure failures, such as automatic restarts, self-healing, and node failure handling. This helps ensure that applications remain available and responsive, even in the face of hardware or software failures.

   3. Automation: Kubernetes automates many aspects of application deployment and management, including container orchestration, configuration management, service discovery, and networking. This helps reduce the risk of errors and improves the speed and consistency of deployments.

   4. Portability: Kubernetes provides a consistent and portable environment for running containerized applications, regardless of the underlying infrastructure. This helps simplify application deployment and management across different environments, such as on-premises data centers, public clouds, and hybrid clouds.

   5. Extensibility: Kubernetes has a modular architecture that allows users to extend and customize its functionality using a wide range of plugins, APIs, and tools. This helps enable integration with other systems and technologies, and allows users to tailor Kubernetes to their specific needs and use cases.

3. Explain the architecture of Kubernetes ?

   Kubernetes has a master-worker architecture that consists of the following components:

   1. Master components: These are responsible for managing the overall state of the Kubernetes cluster. They include:
   - API Server: Exposes the Kubernetes API for managing the clust
   - eretcd: Stores the configuration and state of the cluster
   - Controller Manager: Responsible for managing controllers that regulate the state of the cluster
   - Scheduler: Assigns workloads to worker nodes based on resource availability and workload requirements

   2. Worker components: These run on each node in the Kubernetes cluster and are responsible for managing the workload containers. They include:
   - ubelet: Ensures that containers are running on the node as expected
   - kube-proxy: Handles networking for the containers on the node
   - Container Runtime: Executes and manages containers on the node

   3. Add-on components: These are optional components that provide additional functionality to the Kubernetes cluster, such as logging, monitoring, and load balancing.

4. What is Control Plane?

   In Kubernetes, the Control Plane is the set of components that manage the overall state of the cluster. It consists of the Kubernetes master components, including the API Server, etcd, Controller Manager, and Scheduler.

   The Control Plane is responsible for maintaining the desired state of the cluster, handling node failures and cluster scaling, and scheduling workloads onto worker nodes. It provides a centralized point of control for managing the cluster, and all management operations are performed through the API Server.

5. Write the difference between kubectl and kubelets.

   kubectl and kubelet are two different components in Kubernetes that serve different purposes.

   1. kubectl:
   - kubectl is a command-line tool that allows users to interact with the Kubernetes API Server and perform management operations on the Kubernetes cluster.
   - It can be used to create, modify, and delete Kubernetes resources such as pods, services, deployments, and more.
   - kubectl is typically used by cluster administrators, developers, and operators to manage the Kubernetes cluster and its workloads.

   2. kubelet:
   - kubelet is a component that runs on each worker node in the Kubernetes cluster and is responsible for managing the state of the containers running on that node.
   - It communicates with the Kubernetes API Server to receive instructions on which containers to run and how to configure them.
   - kubelet ensures that the containers are running correctly and restarts them if they fail or become unresponsive.
   - kubelet also monitors the resources on the node and reports back to the API Server on the node's capacity and utilization.

6. Explain the role of the API server.

   The API Server is a critical component of the Kubernetes Control Plane, responsible for managing and exposing the Kubernetes API to users and other components within the cluster.

   The API Server serves as the gateway for all management operations performed on the Kubernetes cluster. All communication with the Kubernetes cluster occurs through the API Server, whether it's via kubectl commands or other Kubernetes components.

   The API Server stores the entire configuration state of the Kubernetes cluster and provides access to this information through the Kubernetes API. It validates and processes API requests, updating the state of the cluster as necessary. It also performs tasks such as authentication, authorization, and admission control, ensuring that only authorized users and components can access or modify the cluster's state.

Kubernetes architecture is important, so make sure you spend a day understanding it. This video will surely help you.

Happy Learning :)