**Kubernetes**

1. If a company with monolithic application wants to move to microservices architecture, then how containers and k8s plays role in it?
2. K8s architecture
3. Docker swarm vs K8s



1. K8s services, types and use cases.
2. Explain Load balancers in k8s.

Load balancers in Kubernetes help distribute traffic across multiple pods, ensuring high availability and reliability for applications. There are 2 types of load balancers – Internal and External.

**Traffic Distribution**: Distributes incoming traffic to multiple pod replicas to balance the load and prevent any single pod from becoming a bottleneck.

**Health Checks**: Continuously monitors the health of pods. If a pod fails, the load balancer stops sending traffic to it, ensuring high availability.

**Scaling**: Works with Kubernetes’ scaling features to adapt to changing traffic demands. As new pod replicas are added, the load balancer automatically distributes traffic accordingly.

Hence, Load balancers are essential components in Kubernetes that enhance application reliability and performance by distributing traffic across multiple pods. They come in various types, each serving different use cases, allowing for flexible and efficient traffic management in a Kubernetes environment.

1. What is namespace?
2. How to enhance security in k8s?

Use Network policies to control pod communications.

Use RBAC.

1. What is Ingress default backend?

**Ingress** in Kubernetes is an API object that manages external access to services within a cluster, typically HTTP/S traffic. It allows you to define rules for routing traffic to different services based on the request's host and path.

**Default Backend Overview**

* **Purpose**: The **default backend** is a service that handles requests that do not match any specific rules defined in the Ingress resource. It acts as a catch-all for incoming traffic.
* **Functionality**:
  + If a request comes in that doesn't match any of the defined paths or hosts in the Ingress rules, the traffic is routed to the default backend.
  + This is particularly useful for serving error pages or providing a generic response for unmatched routes.

**Configuration**

* The default backend is usually defined as part of the Ingress resource configuration. Here’s a simplified example:

A screenshot of a computer

Description automatically generated

In this example:

* If a request is made to example.com for any path other than /foo, it will be directed to default-backend-service.

**Benefits of Using Default Backend**

* **Error Handling**: It can serve custom error pages (e.g., 404 Not Found) for unmatched routes.
* **Simplicity**: Reduces the need for complex rules by providing a fallback option.
* **User Experience**: Helps maintain a smooth user experience by guiding users instead of returning generic errors.

1. How to use Prometheus and Grafana?

Install both using helm charts. Prometheus is nothing but a data source which has our cluster information. We can use queries to get specific data. But data returned by Prometheus is in json format which is not readable. Hence, we use Grafana to create dashboards using Prometheus as data source. Json data output of Prometheus is input for Grafana and Grafana creates dashboards out of it so that visualization is easy.

1. What are custom resources and operator in k8s?
2. Multi-container pod patterns:

<https://chatgpt.com/share/66fa9ab7-d130-800c-b2bf-ad41db88dd95>

Basic to advance level questions –

https://chatgpt.com/share/66fd39a7-ec9c-800c-9fdd-74a28d342f3a