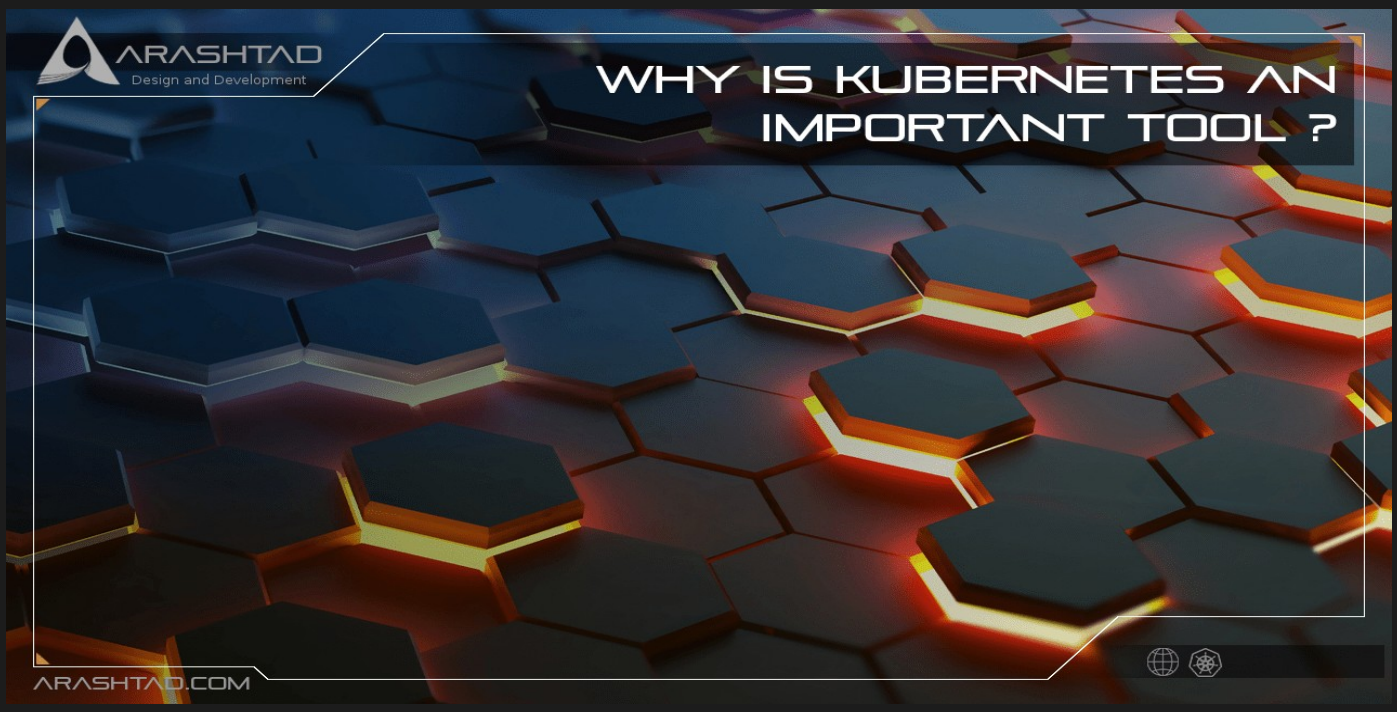
Title	WHY IS KUBERNETES AN IMPORTANT TOOL ?
Description	Intro
Date	2022 21 Augus
Author	Arashtad
Author URI	https://Arashtad.com



Kubernetes also known as “K8s”, automates the deployment and management of cloud-native applications by orchestrating containerized applications running on a cluster of hosts. With this tool, workloads are distributed across a cluster and dynamic container networking is automated. It allocates storage and persistent volumes to running containers, scales automatically, and continuously maintains the desired state of applications, providing resilience. The Kubernetes platform is an open-source platform for managing containerized workloads and services. It facilitates declarative configuration as well as automation. There is a large, rapidly expanding ecosystem of Kubernetes services, support, and tools available.

WHY DO WE NEED KUBERNETES?

When it comes to the necessity of Kubernetes, the summarized answer is that it saves developers and operators a lot of time and effort, allowing them to focus on building the features they need instead of trying to figure out and implement ways to keep their applications running well, at scale. As Kubernetes keeps applications running in spite of challenges (such as failed servers, crashed containers, traffic spikes, etc.), it reduces business impacts, reduces the need for fire drills to restore broken applications, and protects against other liabilities, such as the costs of not meeting Service Level Agreements).

Kubernetes automates the process of building and running complex applications. here are just a few of its many benefits:

1. The standard services that most applications need, such as local DNS and basic load balancing.

2. The majority of the work of keeping applications running, available, and performant depends on standard behaviors (such as restarting this container if it dies).

3. Pods, replica sets, and deployments are abstract “objects” that wrap around containers, enabling easy configurations around collections of containers.

4. A standard API that applications can call to easily enable more sophisticated behaviors, making it much easier to create applications that manage other applications.

KUBERNETES USE CASES:

You can bundle and run your applications using containers. In a production environment, you must make sure that the containers that run the applications don’t go down. For example, if a container goes down, another container needs to start. Wouldn’t it be easier if a system handled this behavior? As a result, Kubernetes can be your solution. Kubernetes provides a framework for running distributed systems in a resilient manner. Kubernetes handles scaling and failover for your application provides deployment patterns, etc. For instance, it can handle a canary deployment for your system easily. Its use cases include:

AUTOMATED ROLLOUTS AND ROLLBACKS

The Kubernetes platform allows you to specify the desired state of your deployed containers, and it can automatically change the state to the desired state at a controlled rate. For example, you can automate it to create new containers for your deployment, remove existing containers, and adopt all of their resources into the new container.

STORAGE ORCHESTRATION

You can use Kubernetes to automatically mount local storage, public cloud providers, and more.

SERVICE DISCOVERY AND LOAD BALANCING

The Kubernetes platform allows exposing containers either via their DNS names or by using their own IP addresses. In the event that high traffic to a container occurs, Kubernetes will load balance and distribute traffic to keep the deployment stable.

AUTOMATIC BIN PACKING

Kubernetes uses a cluster of nodes to run containerized tasks. You tell the system how much CPU and memory (RAM) each container needs. Kubernetes can fit containers onto your nodes to maximize resource utilization.

SECRET AND CONFIGURATION MANAGEMENT

It is possible to store and manage sensitive information, including passwords, OAuth tokens, and SSH keys, without having to rebuild your container images or expose secrets to your stack configuration with Kubernetes. You can deploy and update secrets and application configuration without having to rebuild your container images, and you do not have to expose secrets to your stack.

SELF-HEALING

A Kubernetes cluster restarts applications that fail, replaces containers when they fail, kills containers that fail the user-defined health check, and does not advertise them until they are ready to be used by clients.

HOW DOES KUBERNETES WORK?

It is important for developers to plan out how all the components fit together and work together. They should also plan out how many of each component should run, and what should happen if challenges occur (such as many users logging in at the same time.) Typically, they store their containerized application components in a container registry (local or remote) and define their configuration in a text file. To deploy the application, they “apply” these configurations to Kubernetes. Its job is to evaluate and implement this configuration and maintain it until told otherwise.

1. It Analyzes the configuration and aligns its requirements with those of all other applications on the system.

2. Provides resources appropriate for running new containers (for example, some containers may require GPUs that are not present on every host).

3. It Grabs container images from the registry and starts up the new containers and helps them connect to one another and to system resources (e.g., persistent storage), so the application works as a whole

Once Kubernetes has monitored everything, it tries to fix things and adapt when real events diverge from desired states. if a container crashes, Kubernetes restarts it. if an underlying server fails, Kubernetes finds resources elsewhere to run the containers that the node was hosting. As traffic spikes to an application, Kubernetes can scale out containers to handle the additional load, in accordance with configuration rules.

ADVANTAGES OF KUBERNETES?

There are several important advantages to Kubernetes that have made it so popular:
1. Scalability
As the number of containers increases, Kubernetes spins up additional container instances and scales them out automatically, which is similar to how cloud-native applications scale horizontally.
2. Integration and extensibility
It supports many open source solutions complementary to it, including logging, monitoring, alerting, and more. Its community is working on a variety of open source solutions complementary to Kubernetes.
3. Portability
It is possible to run containerized applications on K8s across an array of environments, including virtual environments and bare metal. Kubernetes is supported in all major public clouds.
4. Cost efficiency
Due to its inherent resource optimization, automated scaling, and flexibility to run workloads where they are most valuable, you can control your IT spending.
5. API-based
It is built upon its REST API, which allows all its components to be programmed.
6. Simplified CI/CD
CI/CD is a DevOps practice for automating building, testing and deploying applications. Enterprises are now integrating Kubernetes and CI/CD pipelines to create scalable CI/CD pipelines.

KUBERNETES AND DOCKER

In Kubernetes, Docker can serve as an orchestration platform for running containers. When Kubernetes schedules a pod on a node, its kubelet instructs Docker to launch the specified containers. Container status is continuously collected from Docker by the kubelet, which aggregates this information in the control plane. Docker pulls containers onto the node and starts and stops them as necessary. When Kubernetes and Docker are used together, the automated system asks Docker to do those things instead of the admin doing them manually on all nodes.

WRAPPING UP

In this tutorial, you learned about Kubernetes, its advantages, use cases, and how it works. In summary, it Automates container networking needs and distributes application workloads across a Kubernetes cluster. It also allocates storage and persistent volumes to running containers, provides automatic scaling, and continuously keeps applications in the desired state, ensuring resiliency.