What is Kubernetes?

Kubernetes is a portable, extensible, open-source container orchestration tool or system from Google, that is used to automate containerized applications tasks such as the management, monitoring, scaling, and deployment. It is easy to manage several containers using Kubernetes as it is a multi-container management solution.

What is K8s?

K8s is yet another term for Kubernetes. As Kubernetes has 8 characters in between K and S so it is termed as k8s.

What does orchestration mean?

The planning or coordination of all the nodes of a cluster to produce a desired output. For example, you have x no.of microservices for an application to run. If you place them in separate containers, this would be an obstacle for communication. Orchestration helps in such situations by enabling all services in individual containers to work seamlessly to accomplish goal outputs.

What is the Docker role and How is it related to Kubernetes?

Docker is used to bundle the application and its dependencies for an easy development. That bundle is known as docker image and can be run in docker containers and those containers are orchestrated by Kubernetes.

What are the Difference between DockerSwarm and Kubernetes?

Kubernetes Docker Swarm

Setup is complicated, but its cluster is robust. Installation is easy, but its cluster is not that robust.

It has GUI (Kubernetes Dashboard). It does not have any GUI.

It is Highly scalable and scales fast. Highly scalable & scales 5x faster than Kubernetes.

Kubernetes can do auto-scaling. Docker swarm cannot do auto-scaling.

No Auto load balancing for the traffic

between different containers and pods. It does auto load balancing.

Can deploy updates and does automatic

rollbacks. Deploys rolling updates, but not automatic rollback.

Only the containers in the same pod can

share the storage volumes. Storage volumes can be shared with all containers.

In-built tools for logging and monitoring 3rd party tools like ELK stack should be used for logging and monitoring.

What are the main components of Kubernetes architecture?

Mainly there are two components: the Kubernetes master and the worker node. Each of these components has individual components in them.

What is a node in Kubernetes?

A node represents a single machine in a cluster, which could be a physical machine in a data center or a virtual machine from a cloud provider. The master in Kubernetes controls the nodes that have containers.

What does the node status contain?

A Node’s status mainly contains the following:

Addresses

Conditions

Capacity and Allocatable

Info

To view a Node’s status and other details, here is the command:

$ kubectl describe node <node-name>

What process runs on Kubernetes Master Node?

The Kube-apiserver process runs on the Kubernetes master node and serves pods, services, replicationcontrollers, and others.

What is a pod in Kubernetes?

Pods is a unit which contains one or more containers with a shared network & storage. This is because containers are not run directly in Kubernetes. Containers in the same pod share a local network and the same resources, allowing them to easily communicate with other containers in the same pod.

What is the job of the kube-scheduler?

The kube-scheduler assigns feasible nodes to newly created pods. If none of the nodes has suitable specifications for the pod, then the pod remains unscheduled until it is assigned.

What is Kubelet?

The kubelet is the primary “node agent” that runs on each worker node. It controls and maintains pods by watching pod specs through the Kubernetes API server. The kubelet runs on each worker node and enables the communication between the master and worker nodes and ensures all the containers up & running all the time.

What is Kubectl?

It is pronounce like “kube control”, “kube c-t-l”, or “kubie cuttle”, “kube cuddle”.

Kubectl is a CLI (command-line interface) to control the Kubernetes clusters. It allows you to command every possible Kubernetes operation.

What is etcd?

Kubernetes uses etcd as a distributed key-value database, stores all cluster data, including metadata and configuration data, and allows nodes in Kubernetes clusters to read and write data.

What is a cluster of containers in Kubernetes?

A cluster of containers is a set of machines(worker nodes & Kubernetes Master) in the same network. At most 5000 nodes in a cluster can be handled by kubernetes v1.18, no more than that. A cluster is a set of worker nodes(physical or virtual machines) run as kubernetes agents and are managed by a Kubernetes Master.

What is Minikube?

Minikube makes Kubernetes to run locally. It is a simple single-node Kubernetes cluster inside a virtual machine

What is the Google Container Engine?

The Google Container Engine is an open-source management platform to manage containers and clusters that runs on Google public cloud services.

What are DaemonSets?

A DaemonSet is a set of pods that runs only once on a host. It ensures that nodes run a copy of the pod, when some nodes are removed from the cluster then those pods list is maintained in the DaemonSets, if you want ro remove those garbage containers (pods) you have to delete DaemonSets.

What is ‘Heapster’ in Kubernetes?

A Heapster is a performance monitoring system for kubernetes through the Kublet. This not only monitors the performance but also keeps information like pods, containers, nodes, cluster, events and signals, etc. Heapster runs like any other pod within a Kubernetes cluster. So it is easy to keep track of all this information.

What is a Namespace in Kubernetes?

In simple words Namespaces are the virtual clusters in kubernetes which are backed by the same physical clusters. As it supports both physical and virtual clusters.

**Question 1. What is Kubernetes?**

Kubernetes is an open-source container orchestration platform. It was developed by Google and was donated to the Cloud Native Computing Foundation (CNCF) in 2015.

**Question 2. How does Kubernetes relate to Docker?**

Docker is a container runtime, which is a software that runs containerized applications. When Kubernetes schedules a pod to a node, the kubelet running on that node instructs Docker to launch the containers.

**Question 3. What is container orchestration?**

Container orchestration is the automation of components and processes related to running containers. It includes things like configuring and scheduling containers, the availability of containers, allocation of resources between containers, and securing the interaction between containers, among other things.

**Question 4. What do you know about Kubernetes clusters?**

A Kubernetes cluster is a set of nodes that containerized applications run on. These nodes can be physical machines or virtual machines.

**Question 5. What is kubectl?**

Kubectl is the command-line configuration tool for Kubernetes that communicates with a Kubernetes API server. Using kubectl allows you to create, inspect, update, and delete Kubernetes objects.

To learn all the stuff you can do with kubectl, check out our [kubectl cheatsheet](https://www.bluematador.com/learn/kubectl-cheatsheet).

**Question 6. What is a pod?**

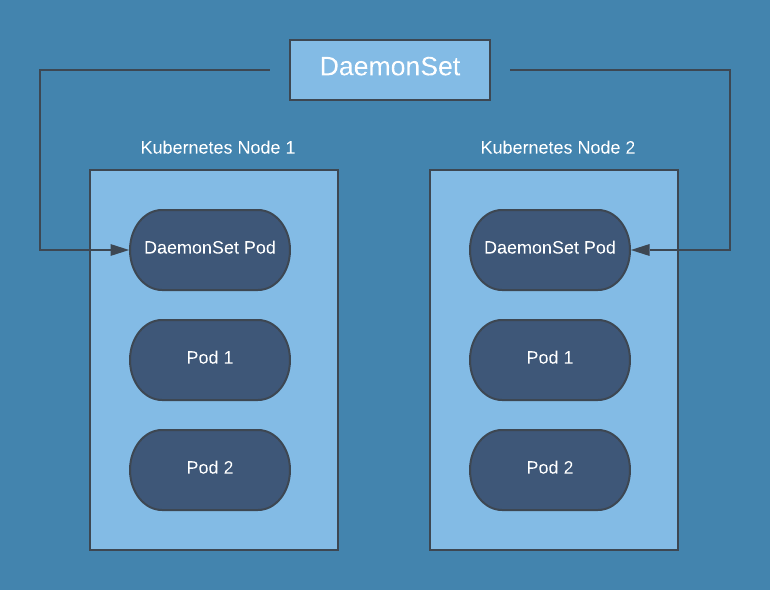
A pod is the most basic Kubernetes object. A pod consists of a group of containers running in your cluster. Most commonly, a pod runs a single primary container.

**Question 7. Can you explain the different components of Kubernetes architecture?**

Kubernetes is composed of two layers: a **control plane** and a **data plane**. The **control plane** is the container orchestration layer that includes 1. Kubernetes objects that control the cluster, and 2. the data about the cluster’s state and configuration. The **data plane**is the layer that processes the data requests and is managed by the control plane.

**Question 8. What is the difference between a daemonset, a deployment, and a replication controller?**

A [daemonset](https://www.bluematador.com/blog/an-introduction-to-kubernetes-daemonsets)ensures that all nodes you select are running exactly one copy of a pod.



A deployment is a resource object in Kubernetes that provides declarative updates to applications. It manages the scheduling and lifecycle of pods. It provides several key features for managing pods, including pod health checks, rolling updates of pods, the ability to roll back, and the ability to easily scale pods horizontally.

The replication controller specifies how many exact copies of a pod should be running in a cluster. It differs from a deployment in that it does not offer pod health checks, and the rolling update process is not as robust.  

**Question 9. Do all of the nodes have to be the same size in your cluster?**

No, they don’t. The Kubernetes components, like kubelet, will take up resources on your nodes, and you’ll still need more capacity for the node to do any work. In a larger cluster, it often makes sense to create a mix of different instance sizes. That way, pods that require a lot of memory with intensive compute workloads can be scheduled by Kubernetes on large nodes, and smaller nodes can handle smaller pods.

**Question 10. What is a sidecar container, and what would you use it for?**

A sidecar container is a utility container that is used to extend support for a main container in a Pod. Sidecar containers can be paired with one or more main containers, and they enhance the functionality of those main containers. An example would be using a sidecar container specifically to process system logs or for monitoring.

**Question 11. How do logs work for pods?**

With a traditional server setup, application logs are written to a file and then viewed either on each server or collected by a logging agent and sent to a centralized location. In Kubernetes, however, writing logs to disk from a pod is discouraged since you would then have to manage log files for pods. The better way is to have your application output logs to **stdout**and **stderr.**The kubelet on each node collects stdout and stderr on the running pods and then combines them into a log file managed by Kubernetes. Then you can use different [kubectl commands](https://www.bluematador.com/learn/kubectl-cheatsheet" \t "_blank) to view the logs.

**Question 12. How can you separate resources?**

You can separate resources by using namespaces. These can be created either using kubectl or applying a YAML file. After you have created the namespace you can then place resources, or create new resources, within that namespace. Some people think of namespaces in Kubernetes like a virtual cluster in your actual Kubernetes cluster.

# What is Kubernetes?

# Kubernetes is Google’s open-source container orchestration tool that can help in container deployment, container scaling and descaling, and load balancing.

# What is the relation between Kubernetes and Docker?

# Docker is capable of supporting container lifecycle management, and the Docker image provides the basis for runtime containers. Basically, Docker helps you build containers, and Kubernetes helps in ensuring communication between them.

# Define container orchestration?

# Assume a case in which you have 3-4 microservices for a specific application. Now, each microservice would be in a different container, thereby losing the ability to communicate with each other. Container orchestration helps in coordinating the work of all containers to address the requirements of a particular server.

# How can Kubernetes simplify containerized deployment?

# General applications comprise a cluster of containers that run across multiple hosts. Kubernetes is cloud-independent and could run on any public or private cloud service platform. It can help in load balancing, scaling, and monitoring of containers, thereby simplifying containerized deployment.

# What are the important components in Kubernetes architecture?

# The two most important components in the Kubernetes architecture are the worker node and the master node. The master node and the worker node have their own individual components.

# Define a node in Kubernetes.

# The node is a representation of a single machine in the cluster, thereby becoming the smallest fundamental unit of computing hardware. A node could either be a virtual machine or a physical machine in a data center.

# What are the processes that the Kubernetes Master Node runs?

# The master node in Kubernetes runs the Kube-API server process and helps in scalability and deployment of additional instances.

# Define a pod in Kubernetes.

# A pod in Kubernetes is a high-level structure wrapping up one or more containers because Kubernetes cannot run containers directly. Containers in a pod share similar local networks and resources, thereby allowing easy communication between them while remaining isolated.

# Define a cluster of containers in Kubernetes.

# The cluster of containers is a set of nodes. Clusters are responsible for starting certain routes for enabling containers on the nodes to communicate with each other.

# What are the benefits of Kubernetes?

# Kubernetes simplifies container management and improves responsiveness to customer demands through faster and predictable deployment. Users can avail the benefits of automatic rollback, horizontal scaling, and automated scheduling and auto-healing functionalities with Kubernetes.

# What are the components of a Kubernetes Master?

# The Kubernetes Master contains the API server, the etcd components, the Scheduler, and the controller manager. These components support the operations and management of Kubernetes cluster.

# What is load balancing on Kubernetes?

# Load balancing in Kubernetes is highly essential for exposing services. Users can find two types of load balancing in the case of Kubernetes, such as internal load balancing and external load balancing. Internal load balancing helps in automatic balancing of loads and allocation of pods with the desired configuration. External load balancing helps in directing traffic from external loads to backend pods.

# What is node affinity and pod affinity?

# Node affinity helps in ensuring the hosting of pods on specific nodes. On the other hand, pod affinity helps in ensuring that two pods could be co-located on a single node.

# How can you start a rollback for an application?

# The Rollback and rolling updates feature in Kubernetes is in-built with the Deployment object. If the existing state of a Deployment is unstable due to configuration or application code, then you can Rollback to earlier Deployment version. With every rollback, you can update the version of the Deployment.

# What are init containers?

# You can find many containers in a Kubernetes pod, and init container is the first container that is executed before running other containers in the pod.

# What is the role of Namespace in Kubernetes?

# Namespaces are ideal for distribution of cluster resources among multiple users in Kubernetes. Namespaces are specifically tailored for environments that involve multiple users spread across different projects or teams. In addition, they can also provide reliable insights into the scope of resources.

# What are the different controller managers in Kubernetes?

# The common types of controller managers on Kubernetes running on the master node include the service accounts controller, token controller, endpoints controller, replication controller, node controller, and namespace controller.

# What is the role of etcd in Kubernetes architecture?

# Etcd serves as a distributed key-value store for Kubernetes data, including configuration data and metadata. It helps the nodes in Kubernetes clusters for reading and writing data.

# What is Kube-proxy?

# Kube-proxy is a load balancer implementation and serves a network proxy that supports service abstraction and other networking operations.

# What is Kubectl?

# Kubectl is the command-line interface (CLI) that helps in running commands for Kubernetes clusters. It is a vital tool for controlling the Kubernetes cluster manager through various create and manage commands.