**1. What is Kubernetes?**

Kubernetes is an open-source container management tool which holds the responsibilities of container deployment, scaling & descaling of containers & load balancing. Being the Google’s brainchild, it offers excellent community and works brilliantly with all the cloud providers. So, we can say that Kubernetes is not a containerization platform, but it is a multi-container management solution.

**2. How is Kubernetes related to Docker?**

It’s a known fact that Docker provides the lifecycle management of containers and a Docker image builds the runtime containers. But, since these individual containers have to communicate, Kubernetes is used. So, Docker builds the containers and these containers communicate with each other via Kubernetes. So, containers running on multiple hosts can be manually linked and orchestrated using Kubernetes.

**3. What is Container Orchestration?**

Consider a scenario where you have 5-6 microservices for an application. Now, these microservices are put in individual containers, but won’t be able to communicate without container orchestration. So, as orchestration means the amalgamation of all instruments playing together in harmony in music, similarly container orchestration means all the services in individual containers working together to fulfill the needs of a single server.

**4. What is Kubectl?**

This is an agent service which runs on each node and enables the slave to communicate with the master. So, Kubelet works on the description of containers provided to it in the PodSpec and makes sure that the containers described in the PodSpec are healthy and running.

**5. Explain node port**

The node port service is a fundamental way to get external traffic to your service. It opens a particular port on all nodes and forwards network traffic sent to this port.

**6. What is Kubectl used for?**

Kubectl is a software for controlling Kubernetes clusters. Ctl stands for control, which is a command-line interface to pass the command to the cluster and manage the Kubernetes component.

**7. What is minikube?**

Minikube is a software that helps the user to run Kubernetes. It runs on the single nodes that are inside VM on your computer. This tool is also used by programmers who are developing an application using Kubernetes.

**8. What is orchestration when it comes to software and DevOps?**

Orchestration refers to the integration of multiple services that allows them to automate processes or synchronize information in a timely fashion. Say, for example, you have six or seven microservices for an application to run. If you place them in separate containers, this would inevitably create obstacles for communication. Orchestration would help in such a situation by enabling all services in individual containers to work seamlessly to accomplish a single goal.

**9. What are the main differences between the Docker Swarm and Kubernetes?**

Docker Swarm is Docker’s native, open-source container orchestration platform that is used to cluster and schedule Docker containers. Swarm differs from Kubernetes in the following ways:

Docker Swarm is more convenient to set up but doesn’t have a robust cluster, while Kubernetes is more complicated to set up but the benefit of having the assurance of a robust cluster

Docker Swarm can’t do auto-scaling (as can Kubernetes); however, Docker scaling is five times faster than Kubernetes

Docker Swarm doesn’t have a GUI; Kubernetes has a GUI in the form of a dashboard

Docker Swarm does automatic load balancing of traffic between containers in a cluster, while Kubernetes requires manual intervention for load balancing such traffic

Docker requires third-party tools like ELK stack for logging and monitoring, while Kubernetes has integrated tools for the same

Docker Swarm can share storage volumes with any container easily, while Kubernetes can only share storage volumes with containers in the same pod

Docker can deploy rolling updates but can’t deploy automatic rollbacks; Kubernetes can deploy rolling updates as well as automatic rollbacks

**10. What are the features of Kubernetes?**

The interesting features of Kubernetes are:

- Automated scheduling

- Self-healing capabilities

- Automated rollout and rollbacks

- Horizontal scaling and role balancing

**11. What do you understand by Kube-proxy?**

Kube-proxy is a network proxy that can run on each and every node and is capable of simple TCP/UDP packet forwarding across backend network service.

**12. What is ETCD?**

Etcd is written in Go programming language and is a distributed key-value store used for coordinating between distributed work. So, Etcd stores the configuration data of the Kubernetes cluster, representing the state of the cluster at any given point in time.

**13. What do you understand by load balancer in Kubernetes?**

A load balancer is one of the most common and standard ways of exposing service. You can see two kinds of load balancers based on the working environment i.e. Internal Load Balancer and External Load Balancer. The Internal Load Balancer balances load and allocates the pods with the demanded configuration. External Load Balancer directs the traffic from the external load to the backend pods.

**14. What is the difference between a replica set and replication controller?**

Replica Set and Replication Controller are not very different from each other, as both of them make sure that a specified number of pod replicas are running at any given time. The only difference between them is Replica Set use Set-Based selectors, and on the other hand replication controllers use Equity-Based selectors.

**15. What do you know about clusters in Kubernetes?**

A Kubernetes cluster is a set of node machines for running containerized applications. At a minimum, a cluster contains a worker node and a master node. The master node is responsible for maintaining the desired state of the cluster, such as which applications are running and which container images they use. Worker nodes actually run the applications and workloads.

**16. What are the different components of Kubernetes Architecture?**

There are 2 main components of Kubernetes Architecture

- The master node

- The worker node.

The master node has components like kube-controller-manager, kube-apiserver, kube-scheduler.

**17. What is the role of kube-apiserver and kube-scheduler?**

The kube — apiserver is based on a scale-out architecture. It is the front-end of the master node control panel. It exposes all the APIs of the Kubernetes Master node components and establishes communication between the Kubernetes Node and the Kubernetes master components.

The kube-scheduler distributes and manages workload on the worker nodes. It selects the most suitable node to run the unscheduled pod based on resource requirement and keeps a track of resource utilization. It ensures that the workload is not scheduled on nodes that are already full.

**18. What is a node in Kubernetes?**

A node is the smallest fundamental unit of computing hardware. It 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. Each machine can substitute any other machine in a Kubernetes cluster. The master in Kubernetes controls the nodes that have containers

**19. What does the node status contain?**

The main components of a node status are Address, Condition, Capacity, and Info.

**20. What process runs on Kubernetes Master Node?**

The Kube-api server process runs on the master node and serves to scale the deployment of more instances.

**21. What is a pod in Kubernetes?**

Pods are high-level structures that wrap one or more containers. 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 as if they were on the same machine while at the same time maintaining a degree of isolation.

**22. What is the Google Container Engine?**

The Google Container Engine is an open-source management platform tailor-made for Docker containers and clusters to provide support for the clusters that run in Google public cloud services.

**23. What are Daemon sets?**

A Daemon set is a set of pods that runs only once on a host. They are used for host layer attributes like a network or for monitoring a network, which you may not need to run on a host more than once.

**24. What is a Namespace in Kubernetes?**

Namespaces are used for dividing cluster resources between multiple users. They are meant for environments where there are many users spread across projects or teams and provide a scope of resources.

**25. What are the types of controller managers?**

The primary controller managers that can run on the master node are the endpoints controller, service accounts controller, namespace controller, node controller, token controller, and replication controller.

**26. How can you get a static IP for a Kubernetes load balancer?**

A static IP for the Kubernetes load balancer can be achieved by changing DNS records since the Kubernetes Master can assign a new static IP address.

**27. Can we use many claims out of a psistent volume?**

The mapping between persistentVolume and persistentVolumeClaim is always one to one. Even When you delete the claim, PersistentVolume still remains as we set persistentVolumeReclaimPolicy is set to Retain and It will not be reused by any other claims. Below is the spec to create the Persistent Volume.

apiVersion: v1

kind: PersistentVolume

metadata:

name: mypv

spec:

capacity:

storage: 5Gi

volumeMode: Filesystem

accessModes:

- ReadWriteOnce

persistentVolumeReclaimPolicy: Retain

**28. What is the difference between a Pod and a Job?**

A Pod always ensure that a container is running whereas the Job ensures that the pods run to its completion. Job is to do a finite task.

Examples:

kubectl run mypod1 --image=nginx --restart=Never

kubectl run mypod2 --image=nginx --restart=onFailure

○ → kubectl get pods

NAME READY STATUS RESTARTS AGE

mypod1 1/1 Running 0 59s

○ → kubectl get job

NAME DESIRED SUCCESSFUL AGE

mypod1 1 0 19s

**29. Does the rolling update with state full set replicas =1 makes sense?**

No, because there is only 1 replica, any changes to state full set would result in an outage. So rolling update of a StatefulSet would need to tear down one (or more) old pods before replacing them. In case 2 replicas, a rolling update will create the second pod, which it will not be succeeded, the PD is locked by first (old) running pod, the rolling update is not deleting the first pod in time to release the lock on the PDisk in time for the second pod to use it. If there's only one that rolling update goes 1 -> 0 -> 1.f the app can run with multiple identical instances concurrently, use a Deployment and roll 1 -> 2 -> 1 instead.