Kubernetes interview questions

1. How is Kubernetes related to Docker? ↑

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.

2. 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.

3. What are the features of Kubernetes? ↑

Automated Scheduling - Self healing capabilities

Automated Rollouts and rollback - Horizaontal Scaling and Load Balancing

4. How does Kubernetes simplify containerized Deployment? ↑

As a typical application would have a cluster of containers running across multiple hosts, all these containers would need to talk to each other. So, to do this you need something big that would load balance, scale & monitor the containers. Since Kubernetes is cloud-agnostic and can run on any public/private providers it must be your choice simplify containerized deployment.

5. What is Google Container Engine? ↑

Google Container Engine (GKE) is an open source management platform for Docker containers and the clusters. This Kubernetes based engine supports only those clusters which run within the Google's public cloud services.

6. What is Heapster? ↑

Heapster is a cluster-wide aggregator of data provided by Kubelet running on each node. This container management tool is supported natively on Kubernetes cluster and runs as a pod, just like any other pod in the cluster. So, it basically discovers all nodes in the cluster and queries usage information from the Kubernetes nodes in the cluster, via on-machine Kubernetes agent.

7. What is Minikube? ↑

Minikube is a tool that makes it easy to run Kubernetes locally. This runs a single-node Kubernetes cluster inside a virtual machine.

8. What is Kubectl? ↑

Kubectl is the platform using which you can pass commands to the cluster. So, it basically provides the CLI to run commands against the Kubernetes cluster with various ways to create and manage the Kubernetes component.

9. What is the syntax for the Kube-proxy command? ↑

The syntax for Kubectl is kubectl [command] [TYPE] [NAME] [flags]

10. What is the syntax for the Kubectl command? ↑

The syntax to configure Proxy is: kube-proxy [flags]

11. What is Kubelet? ↑

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.

12. What are the different components of Kubernetes Architecture? ↑

The Kubernetes Architecture has mainly 2 components – the master node and the worker node. The master and the worker nodes have many inbuilt components within them. The master node has the kube-controller-manager, kube-apiserver, kube-scheduler, etcd. Whereas the worker node has kubelet and kube-proxy running on each node.

13. What do you understand by Kube-proxy? ↑

Kube-proxy can run on each and every node and can do simple TCP/UDP packet forwarding across backend network service. So basically, it is a network proxy which reflects the services as configured in Kubernetes API on each node. So, the Docker-linkable compatible environment variables provide the cluster IPs and ports which are opened by proxy.

14. What is the Kubernetes controller manager? ↑

Multiple controller processes run on the master node but are compiled together to run as a single process which is the Kubernetes Controller Manager. So, Controller Manager is a daemon that embeds controllers and does namespace creation and garbage collection. It owns the responsibility and communicates with the API server to manage the end-points.

15. What are the different types of controller manager running on the master node? ↑

Node Controller - Replication controller

Service account and token controller - Endpoints controller

16. 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.

17. What do you understand by load balancer in Kubernetes? ↑

A load balancer is one of the most common and standard ways of exposing service. There are two types of load balancer used based on the working environment i.e. either the Internal Load Balancer or the External Load Balancer. The Internal Load Balancer automatically balances load and allocates the pods with the required configuration whereas the External Load Balancer directs the traffic from the external load to the backend pods.

18. Write a command to create and fetch the deployment. ↑

To create: kubectl create –f Deployment.yaml –record

To fetch: kubectl get deployments

19. What is Ingress network? ↑

Ingress network is a collection of rules that acts as an entry point to the Kubernetes cluster. This allows inbound connections, which can be configured to give services externally through reachable URLs, load balance traffic, or by offering name-based virtual hosting. So, Ingress is an API object that manages external access to the services in a cluster, usually by HTTP and is the most powerful way of exposing service.

20. What do you understand by Cloud controller manager? ↑

The Cloud Controller Manager is responsible for persistent storage, network routing, abstracting the cloud-specific code from the core Kubernetes specific code, and managing the communication with the underlying cloud services. It might be split out into several different containers depending on which cloud platform you are running on and then it enables the cloud vendors and Kubernetes code to be developed without any inter-dependency. So, the cloud vendor develops their code and connects with the Kubernetes cloud-controller-manager while running the Kubernetes.

21. What are the different types of cloud controller manager? ↑

Node Controller - Route Controller

Volume Controller - Service Controller

22. What is a Headless Service? ↑

Headless Service is similar to that of a 'Normal' services but does not have a Cluster IP. This service enables you to directly reach the pods without the need of accessing it through a proxy.

23. What are federated clusters? ↑

Multiple Kubernetes clusters can be managed as a single cluster with the help of federated clusters. So, you can create multiple Kubernetes clusters within a data center/cloud and use federation to control/manage them all at one place.

24. 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.

25. What is the difference between a daemonset, a deployment, and a replication controller? ↑

A daemonset 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.

26. 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.

27. 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.

28. What are K8s? ↑

K8s is another term for Kubernetes.

29. 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.

30. What does the node status contain? ↑

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

31. 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.

32. What is the job of the kube-scheduler? ↑

The kube-scheduler assigns nodes to newly created pods.

33. What is a cluster of containers in Kubernetes? ↑

A cluster of containers is a set of machine elements that are nodes. Clusters initiate specific routes so that the containers running on the nodes can communicate with each other. In Kubernetes, the container engine (not the server of the Kubernetes API) provides hosting for the API server.

34. 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.

35. Name the initial namespaces from which Kubernetes starts? ↑

Default

Kube – system

Kube – public

36. What are the different services within Kubernetes? ↑

Cluster IP service

Node Port service

External Name Creation service and

Load Balancer service

37. What is ClusterIP? ↑

The ClusterIP is the default Kubernetes service that provides a service inside a cluster (with no external access) that other apps inside your cluster can access.

38. What is NodePort? ↑

The NodePort service is the most fundamental way to get external traffic directly to your service. It opens a specific port on all Nodes and forwards any traffic sent to this port to the service.

39. What is Kube-proxy? ↑

Kube-proxy is an implementation of a load balancer and network proxy used to support service abstraction with other networking operation. Kube-proxy is responsible for directing traffic to the right container based on IP and the port number of incoming requests.

40. 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.

41. What is the difference between config map and secret? ↑

Config maps ideally stores application configuration in a plain text format whereas Secrets store sensitive data like password in an encrypted format. Both config maps and secrets can be used as volume and mounted inside a pod through a pod definition file.

42. If a node is tainted, is there a way to still schedule the pods to that node? ↑

When a node is tainted, the pods don't get scheduled by default, however, if we have to still schedule a pod to a tainted node we can start applying tolerations to the pod spec.

43. Can we use many claims out of a persistent 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.

44. What is Kube-proxy? ↑

Kube-proxy is an implementation of both a network proxy and a load balancer. It is used to support service abstraction used with other networking operations. It is responsible for directing traffic to the container depend on IP and the port number.

45. What are the tools that are used for container monitoring? ↑

Heapster

cAdvisor

Prometheus

InfluxDB

Grafana

46. What are the important components of node status? ↑

Condition

Capacity

Info

Address

47. 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.

48. Explain Prometheus in Kubernetes. ↑

Prometheus is an application that is used for monitoring and alerting. It can be called out to your systems, grab real-time metrics, compress it, and stores properly in a database.

49. List tools for container orchestration. ↑

Docker swarm

Apache Mesos

Kubernetes.

50. Define Stateful sets in Kubernetes. ↑

The stateful set is a workload API object that is used to manage the stateful application. It can also be used to manage the deployments and scaling the sets of pods. The state information and other data of stateful pods are store in the disk storage, which connects with stateful set.

51. Explain Replica set. ↑

A Replica set is used to keep replica pods stable. It enables us to specify the available number of identical pods. This can be considered a replacement for the replication .controller.

52. Why uses Kube-apiserver? ↑

Kube-apiserver is an API server of Kubernetes that is used to configure and validate API objects, which include services, controllers, etc. It provides the frontend to the cluster's shared region using which components interact with each other.

53. Explain the types of Kubernetes pods. ↑

There are two types of pods in Kubernetes:

Single Container Pod: It can be created with the run command.

Multicontainer pods: It can be created using the "create" command in Kubernetes.

54. What are the labels in Kubernetes? ↑

Labels are a collection of keys that contain some values. The key values are connected to pods, replication controllers, and associated services. Generally, labels are added to some object during its creation time. They can be modified by the users at run time.

55. What is Sematext Docker Agent? ↑

Sematext Docker agent is a log collection agent with events and metrics. It runs as a small container in each Docker host. These agents gather metrics, events, and logs for all cluster nodes and containers.

56. Define OpenShift. ↑

OpenShift is a public cloud application development and hosting platform developed by Red Hat. It offers automation for management so that developers can focus on writing the code.

57. What is ContainerCreating pod? ↑

A ContainerCreating pod is one that can be scheduled on a node but can’t start up properly.

58. What do you mean by Persistent Volume Claim? ↑

Persistent Volume Claim is actually the storage provided to the pods in Kubernetes after the request from Kubernetes. User is not expected to have knowledge in the provisioning and the claims has to be created where the pod is created and in the same namespace.

59. What will happen while adding new API to Kubernetes? ↑

If you add a fresh API to Kubernetes, the same will provide extra features to Kubernetes. So, adding a new API will improve the functioning ability of Kubernetes. But, this will increase the cost and maintenance of the entire system. So, there is a need to maintain the cost and complexity of the system. This can be achieved by defining some sets for the new API.

60. How do you make changes in the API? ↑

Changes in the API server has to be done by the team members of Kubernetes. They are responsible to add a new API without affecting the functions in the existing system.

61. What is kubectl drain? ↑

kubectl drain command is used to drain a specific node during maintenance. Once this command is given, the node goes for maintenance and is made unavailable to any user. This is done to avoid assigning this node to a new container. The node will be made available once it completes maintenance.

62. Define Autoscaling in Kubernetes. ↑

One of the important feature of Kubernetes is Autoscaling. Autoscaling can be defined as scaling the nodes according to the demand for service response. Through this feature, cluster increases the number of nodes as per the service response demand and decreases the nodes in case of the decrease in service response requirement. This feature is supported currently in Google Container Engine and Google Cloud Engine and AWS is expected to provide this feature at the earliest.

63. What is the “Master”? ↑

Master refers to a central point of control, which gives a unified view of a cluster. There is a single master node, which controls different minions. Master servers then work together to accept user requests and determine the best means of scheduling the workload containers, authenticate clients and nodes as well as adjust on the cluster wide networking and managing the scaling and health checking of responsibilities.

64. What is a Swarm in Docker? ↑

The docker Swarm is a clustering and scheduling tool for the Docker containers. When it comes to Swarm, the IT administrators and developers would establish and manage a cluster of Docker nodes as part of the single virtual system.

65. What is Kubernetes Log? ↑

Kubernetes container logs are much similar to Docker container logs. But, Kubernetes allows users to view logs of deployed pods i.e running pods.

66. What are the types of Kubernetes Volume? ↑

The types of Kubernetes Volume are:

EmptyDir

GCE persistent disk

Flocker

HostPath

NFS

ISCSI

rbd

PersistentVolumeClaim

downwardAPI

67. Explain PVC. ↑

The full form of PVC stands for Persistent Volume Claim. It is storage requested by Kubernetes for pods. The user does not require to know the underlying provisioning. This claim should be created in the same namespace where the pod is created.

68. What is the Kubernetes Network Policy? ↑

Network Policy defines how the pods in the same namespace would communicate with each other and the network endpoint.