—Kubernetes—

Kubernetes-> Greek word for helmsman or captain of the ship- the logo of Kubernetes is the wheel of the helmsman

K8s—> from k to s there are 8 characters between them

Kubernetes is a container management or container orchestration tool. Developed by google lab later - - donated to CNCF
- Open source

What is Container management tool?
Container management/ orchestration tool
Container orchestration tool automates deploying, scaling and managing containerized application on a group of servers.

What are containerized applications—> Docker (packages the application all the dependencies)
Docker is a tool designed to make it easier to deploy and run applications by using containers.
Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

Organizations have to use multiple containers to
- Ensure availability
- load balancing
- scale up and down based on user load.

(Deploying- scheduling-scaling-load balancing- batch execution-roll back-monitoring)

Note: there are other container orchestration tools like, Docker swarm and Apache Mesos Marathon.
Interview question: Tell me the difference between Docker swarm and K8s?

Kubernetes —> container management tool **—->** Manages containerized applications **—>** Applications available on a container platform like Docker.

Docker —> creates containers
Kubernetes —> manages containers

Kubernetes automatically packages your application and schedules the container based o the requirements and resources available. Automatically places containers based on their resource requirements like CPU & Memory (RAM), while not sacrificing availability. Save resources.

In Kubernetes we do not incheck directly with the container, we do it with the functional unit called pod.
Pods can have single or multiple container and are housed in Nodes.

Features of Kubernetes:

1. Automatic bin packing.

We have the option to specify how much resources like CPU and memory(RAM) each container i inside a pod needs. This will help Automatic bin packing.

2. Service discover and load balancing

A Kubernetes service is a set of pods that work together. K8s has control over network and communication between pods and can load balance across them.

3. Storage Orchestration
Containers running inside a pod may need to store data. Usually a single volume is shared within all the containers in a pod. Kubernetes allows to mount the storage system of your choice -Local-Cloud(AWS)-Network(NFS). (K8s provides us an option and the choice to select the volume storage resource.)

4. Self Healing
If a container fails K8s will restart a new container. I a node dies, K8s replaces and reschedule containers on other nodes. If container does not respond to the (monitoring) health check K8s kills container and replaces. All this is called **ReplicationController**.

5. Automated rollouts & rollbacks
Rollout: Whenever we make any changing to the application or deploy new features to the application this is called rollout.
Rollback: Revert the changes & restore to previous state.

K8s ensures there is no downtime during this process.

6. Secret and Configuration Management
Secret: In K8s sensitive data like passwords, keys, tokens are handled using Secrets.
- Secret is K8s object that separates sensitive data from pods & containers.
- created outside pods & containers
- makes sensitive data portable and easy to manage.

ConfigMap: are handled using ConfigMaps.
- ConfigMap is K8s object that separates configurations from pods & containers.

With this future in K8s we can manage secretes and configurations separately from the image and it also helps to deploy to update the secrets and configurations without rebuilding the image.
Notes: Secret & Configurations are stored in ETCD. (ETCD is a key-value database.

interview question: What is the max size limit for a secret? 1MB.

7. Batch execution
- Batch jobs require an executable/process to be run to completion. In K8s we have to run to completion jobs which are used for batch processing. Each job creates one or more pods (based on requirements), during job execution if any container or pods fails, **Job Controller** will reschedule the container, pods on another node.

Can run multiple pods in parallel and can scale up if required. As the job is completed, the pods will move from running state to shut down state.

K8s supports batch execution, long running jobs, and replaces failed containers.

8. Horizontal scaling
- In K8s we can scale up or down the containers.

Note: Scale-up: create more replicas of the containers if required.
Scale- down: kill the containers in case these containers are not required.
-scaling up & down of containers can be done by **using commands, from the dashboard (k8s ui) automatically based up on CPU usage.**

Architecture of K8s:
When we deploy K8s, we get a cluster. A cluster is a set of machines, called nodes. A cluster has at least one worker and one master node.

Interview question: What are the limits of the components that we can have in a cluster?

- A cluster can have max 5000 nodes
- max 150000 total pods
- 300000 containers
- 100 pods per node

Master node—>

- is responsible for managing the cluster
- monitors nodes & pods in a cluster
- when a node fails, moves the workload of the failed node to another worker node

Master has 4 components:

1- API Server—>is responsible for all the communications.
To interact with the API Server we use some APIs and to call the APIs we can use the command-line(kubectl) tool or from a UI (K8s dashboard). It is written in GO programming language. (interview question)

2- Scheduler—> schedules pods across multiple nodes
interview question: Where does the Scheduler gets all the information? from Configuration file and indirectly from etc datastore/database.
Scheduler knows about these configurations of nodes and whenever it has to schedule a pod, it will check what node will fit best for the configuration or hardware requirements of a pod. It will schedule the pods on nodes accordingly.

3- Controller Manager—> runs controllers
Controller manager is a component in master that has different controllers which runs different monitors.
- cube control manager—>runs monitors that will inform the master in case there is any node which has failed or is not available so that the necessary actions can be taken.
These controllers are the cube control manager will make sure that nodes are running all the time, it will also check the health of the cluster and also check the correct number of pods are running as per as the specifications file.

Interview question: These controllers are all separate processes however to reduce the complexity they are compiled into single binary and run as a single process, so we will not find a separate binary or bat file for every controller. We will find a single file or a single process however these are actually internally separate processes.

- Node Controller (responsible for monitoring the nodes)
- Replication Controller (checks the correct number of pods)
- Endpoints Controller (maintaining all the services and communications)
- service account and token controllers (responsible for maintaining accounts and API access and all the name spaces.)

- cloud control manager—>when we are using the infrastructure of a cloud provider all the monitors that needs to be run for interacting cloud service provider of the infrastructure of the cloud provider is done through cloud control manager.

Note: All these controllers runs some monitors in loop which will take care of the health of the cluster and it will take care of the configuration and the desired infrastructure whenever there is some mismatch or something is not as per the configurations it will send the signal or take necessary action and send the communication to the required component which will then take some action as required.

4- ETCD —> open source, distributed key-value database. (Stores all the data in k8s cluster)
- Stores all the data in k8s cluster
- **only API server is the component that can directly interact with the etcd data store, there is no other component that will interact with ETCD directly.
- ETCD can be a part of the master or it can be configured externally.

Worker Node and his components:

- Worker node can be a physical or virtual machine in the cluster and every node should run a container runtime environment like docker. We have three components in the worker node:

1- kubelet—> is an agent running on each node, communicates with the master node via API server. Kubelet gets the information from some specs called PodSpecs and based on those specifications it makes sure the containers are running accordingly on the pods. If there is any issue with the pod it will try to restart the pods on the same node or different node.
Note: kubelet doesn't manage the containers which were not created by K8s

2- kube proxy—> is the core networking component in kubernetes(node) , and it can also interact with the external world and this is a component or an agent which is responsible for maintaining the network configuration and the rules, so cube-proxy is the networking component.
Interview question: All worker nodes run a daemon called kube-proxy, which watches the API server on the master node. It can interact with the master node using API server and it gets all the information for addition of removal of services or endpoints.

3-container runtime—> container runtime is a software that running containers and the most common container platform used with kubernetes is docker. So in a node we have pods and pods have containers. With kubernetes we can use several container runtime, but Docker is the most common and widely used.
Note: K8s doesn't have the capability to directly handle containers, so it needs a container runtime like docker.

What is Minikube? minikube is a tool that lets you run Kubernetes locally. minikube runs a single-node Kubernetes cluster on your personal computer (including Windows, macOS and Linux PCs) so that you can try out Kubernetes, or for daily development work.