Security:
Q1) What is the importance of SSL?

Q2) What is a SQL injection?

Q3) What is cross-site scripting (XSS)?

Q4) Why shouldn’t you roll your own crypto?

Q5) How are passwords stored on databases?

Q6) What is a Man-in-the-middle attack?

Q7) How do you safely manage environment variables in a cloud environment?

Q8) How do you manage security updates?

Q9) How do you keep encryption keys and credentials secure but make them available to machines that need them?

Docker
Q1) What is docker for?

Q2) How to ask docker cli to show all containers&

Q3) How to delete image with container, who use this image?

Q4) What command help you to delete all old unused images?

Q5) What is docker-compose? What is docker-compose.yml?

Q6) How to expose ports in docker-compose file?

Q7) How to reduce docker images?

Q8) Where you can store docker images?

Q9) What is alpine and why we need it?

kubernetes
Q1) What is Kubernetes? -A: Kubernetes is an open-source container management tool that holds the responsibilities of container deployment, scaling & descaling of containers & load balancing. Being 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.

Q2) How is Kubernetes related to Docker? -A: 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.

-Q3) What is Container Orchestration? -A: 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.\

-Q4) What do you know about clusters in Kubernetes? -A: The fundamental behind Kubernetes is that we can enforce the desired state management, by which I mean that we can feed the cluster services of a specific configuration, and it will be up to the cluster services to go out and run that configuration in the infrastructure. So, as you can see in the above diagram, the deployment file will have all the configurations required to be fed into the cluster services. Now, the deployment file will be fed to the API and then it will be up to the cluster services to figure out how to schedule these pods in the environment and make sure that the right number of pods are running.

So, the API which sits in front of services, the worker nodes & the Kubelet process that the nodes run, all together make up the Kubernetes Cluster.

-Q5) How to do maintenance activity on the K8 node? -A:Whenever there are security patches available the Kubernetes administrator has to perform the maintenance task to apply the security patch to the running container in order to prevent it from vulnerability, which is often an unavoidable part of the administration. The following two commands are useful to safely drain the K8s node.

kubectl cordon kubectl drain –ignore-daemon set The first command moves the node to maintenance mode or makes the node unavailable, followed by kubectl drain which will finally discard the pod from the node. After the drain command is a success you can perform maintenance.

Note: If you wish to perform maintenance on a single pod following two commands can be issued in order:

kubectl get nodes: to list all the nodes kubectl drain : drain a particular node

-Q6) What is the role of Load Balance in Kubernetes? -A: Load balancing is a way to distribute the incoming traffic into multiple backend servers, which is useful to ensure the application available to the users. In Kubernetes, as shown in the above figure all the incoming traffic lands to a single IP address on the load balancer which is a way to expose your service to outside the internet which routes the incoming traffic to a particular pod (via service) using an algorithm known as round-robin. Even if any pod goes down load balances are notified so that the traffic is not routed to that particular unavailable node. Thus load balancers in Kubernetes are responsible for distributing a set of tasks (incoming traffic) to the pods

-Q7) How to monitor the Kubernetes cluster? -A: Prometheus is used for Kubernetes monitoring. The Prometheus ecosystem consists of multiple components.

Mainly Prometheus server which scrapes and stores time-series data. Client libraries for instrumenting application code. Push gateway for supporting short-lived jobs. Special-purpose exporters for services like StatsD, HAProxy, Graphite, etc. An alert manager to handle alerts on various support tools

-Q8) Can you explain the differences between Docker Swarm and Kubernetes? -A: Below are the main difference between Kubernetes and Docker:

The installation procedure of the K8s is very complicated but if it is once installed then the cluster is robust. On the other hand, the Docker swarm installation process is very simple but the cluster is not at all robust. Kubernetes can process the auto-scaling but the Docker swarm cannot process the auto-scaling of the pods based on incoming load. Kubernetes is a full-fledged Framework. Since it maintains the cluster states more consistently so autoscaling

is not as fast as Docker Swarm.

-Q9) How can containers within a pod communicate with each other? -A: Containers within a pod share networking space and can reach other on localhost. For instance, if you have two containers within a pod, a MySQL container running on port 3306, and a PHP container running on port 80, the PHP container could access the MySQL one through localhost:3306.

-Q10) Explain what is a Master Node and what component does it consist of? -A: The master node is the most vital component responsible for Kubernetes architecture It is the central controlling unit of Kubernetes and manages workload and communications across the clusters The master node has various components, each having its process. They are: -ETCD -Controller Manager -Scheduler -API Server

ETCD (Cluster store): -This component stores the configuration details and essential values -It communicates with all other components to receive the commands and work in order to perform an action -It also manages network rules and posts forwarding activity

Controller Manager -It is responsible for most of the controllers and performs a task -It is a daemon which runs in a continuous loop and is responsible for collecting and sending information to API server -The key controllers handle nodes, endpoints, etc.

Scheduler: -It is one of the key components of the master node associated with the distribution of workload -The scheduler is responsible for workload utilization and allocating pod to a new node -The scheduler should have an idea of the total resources available as well as resources allocated to existing workloads on each node