What do you understand by Kubernetes?

Kubernetes is basically a type of an open – source container. Kubernetes has the potential to hold the container deployment, scaling and descaling of the container and load balancing. Kubernetes was being developed in the year of 2014. It is also used to manage the Linux containers across the privates, hybrid and cloud environments.

What is Kubernetes?

First let us compare Kubernetes with Docker Swarm

Comparison Kubernetes Docker Swarm

Controller Master Manager

Slave Nodes Worker

Deployment Unit Pod Task

Load balancing Service Ingress

Kubernetes is a container orchestration tool that is used for automating the managing, monitoring, scaling and deploying of containerized applications. It creates groups of containers that can be logical discovered and managed for easy operations on containers.

What are the difference between Kubernetes and Docker Swarm?

The followings are the main difference between Kubernetes and docker and they are:

* The installation structure of the Kubernetes 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 do the process of the auto scaling but the Docker swarm cannot do the process of the auto scaling.
* Kubernetes is highly scalable and also scales fast. But the Docker swarm scales are 5x faster than Kubernetes and is also highly scalable.
* Docker Docker Swarm is the default container orchestration tool that comes with Docker. Swarm can only orchestrate simple Docker containers. Kubernetes on the other hand helps to manage much more complex software application containers. Kubernetes offers support for larger demand production environment.

What are nodes in kubernetes?

A node is a type of work machine in Kubernetes that was previously known as a minion. A node can be a type of virtual machine or the physical machine. It always depends upon the clusters. Each of the nodes provides the services that are necessary to run pods, and it is also managed by the master components. The node services could include kubelet, kube-proxy and so on.

What does the nodes status contains?

The followings are the main components that the node status:

* Address
* Condition
* Capacity
* Info

What are pods in Kubernetes?

A Kubernetes pod is a group of containers that are being deployed in the same host. Pods have the capacity to operate one level higher than the individual containers. This is because pods have the group of containers that work together to produce an artefact or to process a set of work.

What are namespaces in Kubernetes?

Kubernetes is especially intended for the use of the environments with many other users that are being spread across multiple teams or projects. Namespaces are the way to divide the cluster resources between the multiple users. In future versions of Kubernetes, objects in the same namespace will have the same access control policies by default.

What are the initial namespaces from which the Kubernetes starts?

The followings are the three initial namespaces from which the Kubernetes starts:

* Default
* Kube – system
* Kube – public

What is heapster in Kubernetes?

Heapster is a type of cluster-wide aggregator that helps in the process of monitoring and event data. Heapster helps to enable the container cluster monitoring and performance analysis for Kubernetes.

What is minikube?

Minikube is a type of tool that makes the Kubernetes easy to run locally. Minikube basically runs on the single nodes Kubernetes cluster that is inside the virtual machine on your laptop. This is also used by the developers who are trying to develop by using Kubernetes day to day.

What are the features of Minikube?

The followings are the main features of the Minikube:

* DNS
* Nodeports
* Configure maps and secrets
* Dashboards
* Enabling CNI
* Ingress
* Container runtime: Docker, rkt, CRI – O and containerd

What is GKE in Kubernetes?

Firstly GKE stands for Google Kubernetes Engine. GKE is a management and an orchestration system that is used for Docker container and all the container clusters that basically run within the Google’s public cloud services. Google Kubernetes engine is based on Kubernetes.

What are the uses of Google Kubernetes Engine?

The followings are the uses of the Google Kubernetes Engine:

* Create or resize Docker container clusters
* Creates container pods, replication controller, jobs, services or load balancer
* Resize application controllers
* Update and upgrade container clusters
* Debug container clusters.

What do you mean by Kubelet?

* Kubelet is a type of primary node agents that especially runs on each node. Kubelet only works on the descriptions that the containers provide to the Podspec. Kubelet also makes sure that the container described in Podspec is healthy and running.
* You can think of Kubelet as the lowest level component in a Kubernetes. The Kubelet is responsible for making the individual machines to run. The sole purpose of a Kubelet is given a set of containers, it has to ensure that they are all running.

What are the different types of services being provided by Kubernetes?

The followings are the different types of services being provided by the Kubernetes:

* Cluster IP
* Node Port
* Load Balancer
* External name

What is Kubernetes Load Balancing?

Load Balancing is one of the most common and the standard ways of exposing the services.

There are two types of load balancing in Kubernetes and they are:

* Internal load balancer – This type of balancer automatically balances loads and allocates the pods with the required configuration.
* External Load Balancer – This type of balancer directs the traffic from the external loads to backend pods.

What are the main components of the Kubernetes?

The following are the main components of the Kubernetes:

* API server
* Scheduler
* Controller manager
* .etcd
* .Addons

The Kubernetes Master components are responsible for running and managing the Kubernetes cluster.

Pros of the Kubernetes

The following are the pros of the Kuberenetes:

* Kubernetes is open source and free
* It is highly scalable
* Kubernetes makes it a lot easier to establish effective CI/CD Pipelines
* It is less tightly bound to particular sets of tools
* It provides scheduler, rolling upgrades, health checks and autoscaling
* Kubernetes runs best in any of the operating systems
* Kubernetes provides more concepts and is more powerful than Docker swarm
* It has a flat networking space
* It also has a customizable functionality
* It is comprehensive and detailed documented

Cons of the Kubernetes

Some of the cons of the Kubernetes are as follows:

* Kubernetes is not simple to manage the services
* It is difficult to install and configure
* It takes time to get up and run
* No placements are still available yet

Main features of the Kubernetes

The following are the main features of the Kubernetes:

* Kubernetes automates various manual processes. They will automatically control the server for you and also help you host the container. It also controls how to Kubernetes will be launched.
* Help to interact with certain groups of the container. Kubernetes helps to manage the numerous clusters at the same time.
* Kubernetes also helps to check constantly the health of nodes and containers.
* Kubernetes will allow the scaling resources horizontally and vertically, which is easy and quick.
* Kubernetes also knows where to place the containers. The Kubernetes does this by calculating the best location for the containers.
* Kubernetes provides additional services to the containers.
* Kubernetes also offers security, networking and storage services to the container.

What are the benefits of Kubernetes?

With container orchestration tool Kubernetes, it becomes extremely easy to handle the containers. You can respond to customer demands by deploying the applications faster and in a more predictable manner.

Here we will list some of the benefits of Kubernetes:

Automatic scheduling

Automated rollback

Horizontal scaling

Auto healing capabilities.

What is orchestration in software?

Application orchestration in software process means you can integrate two or more applications. You will be able to automate arrangement, coordination and management of computer software. The goal of any orchestration process is to streamline and optimize frequent repeatable processes.

What is a container cluster?

A container cluster lets you place and manage the containers in a dynamic setup. It can be considered as a set of nodes or Compute Engine instances. The API server of the Kubernetes does not run on cluster nodes but instead the Container Engine hosts the API server.

What is Kubectl?

The Kubectl is a Kubernetes command line tool that is used for deploying and managing applications on Kubernetes. The Kubectl is especially useful for inspecting the cluster resources, for creating, updating and deleting the components.

What is Kube proxy?

The Kube proxy runs on each of the node. It can do simple tasks like TCP, UDP Forwarding and so on. It shows the services in the Kubernetes API on each node.

What is the use of kube-controller-manager?

It is the Kubernetes controller manager. The kube-controller-manager is a daemon that embeds the core control loops which regulates system state and it is a non-terminating loop.

what does a Kube-scheduler do?

The Kube-scheduler has to job of assigning the nodes to the newly created pods.

Where is the Kubernetes cluster data stored?

The primary data store of Kubernetes is etcd which is responsible for all Kubernetes cluster data store.

How to set a static IP for Kubernetes load balancer?

Kubernetes Master assigns a new IP address.

You can set a static IP for Kubernetes load balancer by changing the DNS records every time.

What is the meaning of Kubernetes?

Kubernetes (commonly referred to as “K8s”) is an open-source system for automating deployment, scaling and management of containerized applications that was originally designed by Google and donated to the Cloud Native Computing Foundation.

What is a docker?

Docker container is an open source software development platform. Its main benefit is to package applications in “containers,” allowing them to be portable among any system running the Linux operating system (OS).

What is a swarm in Docker?

Docker Swarm is a clustering and scheduling tool for Docker containers. With Swarm, IT administrators and developers can establish and manage a cluster of Docker nodes as a single virtual system.

What is Openshift?

OpenShift Online is Red Hat’s public cloud application development and hosting platform that automates the provisioning, management and scaling of applications so that you can focus on writing the code for your business, startup, or big idea.

What is Docker and what does it do?

Docker is a tool designed to make it easier to create, 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.

Why do we use Docker?

Docker provides this same capability without the overhead of a virtual machine. It lets you put your environment and configuration into code and deploy it. The same Docker configuration can also be used in a variety of environments. This decouples infrastructure requirements from the application environment.

What is a docker in cloud?

A node is an individual Linux host used to deploy and run your applications. Docker Cloud does not provide hosting services, so all of your applications, services, and containers run on your own hosts. Your hosts can come from several different sources, including physical servers, virtual machines or cloud providers.

What is k8s?

Kubernetes, also sometimes called K8S (K – eight characters – S), is an open source orchestration framework for containerized applications that was born from the Google data centers.

Which process runs on Kubernetes master node?

Kube-apiserver process runs on Kubernetes master node.

Which process runs on Kubernetes non-master node?

Kube-proxy process runs on Kubernetes non-master node.

Which process validates and configures data for the api objects like pods, services?

kube-apiserver process validates and configures data for the api objects.

Kubernetes objects made up of what?

Kubernetes objects are made up of Pod, Service and Volume.

What are Kubernetes controllers?

Kubernetes controllers are Replicaset, Deployment controller.

What is the role of kube-scheduler?

kube-scheduler is responsible for assigning a node to newly created pods.

Which container runtimes supported by Kubernetes?

Kubernetes supports docker and rkt container runtimes.

What are the components interact with Kubernetes node interface?

Kubectl, Kubelet, and Node Controller components interacts with Kubernetes node interface.

So what exactly is Docker? Something about “container applications” right?

* Docker is an open platform that both IT operations teams and Developer team use to build, ship and run their applications, giving them the agility, portability and control that each team requires across the software supply chain.
* We have created a standard Docker container that packages up an application, with everything that the applications requires to run. This standardization allows teams to containerize applications and run them in any environment, on any infrastructure and to be written in any language

What is a Docker container and how is it different than a VM? Does containerization replace my virtualization infrastructure?

* Containerization is very different from virtualization. It starts with the Docker engine, the tool that creates and runs containers (1 or more), and is the Docker installed software on any physical, virtual or cloud host with a compatible OS. Containerization leverages the kernel within the host operating system to run multiple root file systems.
* We call these root file systems “containers.” Each container shares the kernel within the host OS, allowing you to run multiple Docker containers on the same host. Unlike VMs, containers do not have an OS within it. They simply share the underlying kernel with the other containers.
* Each container running on a host is completely isolated so applications running on the same host are unaware of each other (you can use Docker Networking to create a multi-host overlay network that enables containers running on hosts to speak to one another).
* The image below shows containerization on the left and virtualization on the right. Notice how containerization (left), unlike virtualization (right) does not require a hypervisor or multiple OSs.
* Docker containers and traditional VMs are not mutually exclusive, so no, containers do not have to replace VMs. Docker containers can actually run within VMs. This allows teams to containerize each service and run multiple Docker containers per vm.

What’s the benefit of “Dockerizing?”

* By Dockerizing their environment enterprise teams can leverage the Docker Containers as a Service Platform (CaaS). CaaS gives development teams and IT operations teams agility, portability and control within their environment.
* Developers love Docker because it gives them the ability to quickly build and ship applications. Since Docker containers are portable and can run in any environment (with Docker Engine installed on physical, virtual or cloud hosts), developers can go from dev, test, staging and production seamlessly, without having to recode.
* This accelerates the application lifecycle and allows them to release applications 13x more often. Docker containers also makes it super easy for developers to debug applications, create an updated image and quickly ship an updated version of the application.
* IT Ops teams can manage and secure their environment while allowing developers to build and ship apps in a self-service manner. The Docker CaaS platform is supported by Docker, deploys on-premises and is chock full of enterprise security features like role-based access control, integration with LDAP/AD, image signing and many more.
* In addition, IT ops teams have the ability to manage deploy and scale their Dockerized applications across any environment. For example, the portability of Docker containers allows teams to migrate workloads running in AWS over to Azure, without having to recode and with no downtime.
* Team cans also migrate workloads from their cloud environment, down to their physical datacenter, and back. This enables teams to utilize the best infrastructure for their business needs, rather than being locked into a particular infrastructure type.
* The lightweight nature of Docker containers compared to traditional tools like virtualization, combined with the ability for Docker containers to run within VMs, allowing teams to optimize their infrastructure by 20X, and save money in the process.

From an infrastructure standpoint, what do I need from Docker? Is Docker a piece of hardware running in my datacenter, and how taxing is it on my environment?

The Docker engine is the software that is installed on the host (bare metal server, VM or public cloud instance) and is the only “Docker infrastructure” you’ll need. The tool creates, runs and manages Docker containers. So actually, there is no hardware installation necessary at all. The Docker Engine itself is very lightweight, weighing in around 80 MB total.

What exactly do you mean by “Dockerized node”? Can this node be on-premises or in the cloud?

* A Dockerized node is anything i.e a bare metal server, VM or public cloud instance that has the Docker Engine installed and running on it.
* Docker can manage nodes that exist on-premises as well as in the cloud. Docker Datacenter is an on-premises solution that enterprises use to create, manage, deploy and scale their applications and comes with support from the Docker team.
* It can manage hosts that exist in your datacenter as well as in your virtual private cloud or public cloud provider (AWS, Azure, Digital Ocean, SoftLayer etc.).

Do Docker containers package up the entire OS and make it easier to deploy?

Docker containers do not package up the OS. They package up the applications with everything that the application needs to run. The engine is installed on top of the OS running on a host. Containers share the OS kernel allowing a single host to run multiple containers.

What OS can the Docker Engine run on?

The Docker Engine runs on all modern Linux distributions. We also provide a commercially supported Docker Engine for Ubuntu, CentOS, OpenSUSE, RHEL. There is also a technical preview of Docker running on Windows Server 2016.

How does Docker help manage my infrastructure? Do I containerize all my infrastructure or something?

Docker isn’t focused on managing your infrastructure. The platform, which is infrastructure agnostic, manages your applications and helps ensure that they can run smoothly, regardless of infrastructure type via solutions like Docker Datacenter. This gives your company the agility, portability and control you require. Your team is responsible for managing the actual infrastructure.

How many containers can run per host?

As far as the number of containers that can be run, this really depends on your environment. The size of your applications as well as the amount of available resources (i.e like CPU) will all affect the number of containers that can be run in your environment. Containers unfortunately are not magical. They can’t create new CPU from scratch. They do, however, provide a more efficient way of utilizing your resources. The containers themselves are super lightweight (remember, shared OS vs individual OS per container) and only last as long as the process they are running. Immutable infrastructure if you will.

What do I have to do to begin the “Dockerization process”

* The best way for your team to get started is for your developers to download Docker for Mac or Docker Windows. These are native installations of Docker on a Mac or Windows device.
* From their, developers will take their applications and create a Dockerfile. The Dockerfile is where all of the application configuration is specified. It is essentially the blueprint for the Docker Image.
* The image is a snapshot of your application and is what the Docker Engine looks at so it knows what the container it is spinning up should look like.