**Kubernetes Tutorial: Architecture, Basics and Features.**

Before we learn Kubernetes, let's learn:

**Why you need containers?**

Today's internet user never accept downtime. Therefore developers have to find a method to perform maintenance and update without interrupting their services.

Therefore container, which is isolated environments. It includes everything needed for application to run. It makes it easy for a developer to edit and deploying apps. Moreover, containerization has become a preferred method for packaging, deploying, and update web apps.

**What is Kubernetes?**

**KUBERNETES** is a container management system developed in the Google platform. The purpose of kubernetes is to manage a containerized application in various types of Physical, virtual, and cloud environments. Google Kubernetes is a highly flexible container tool to deliver even complex applications, consistently. Applications run on clusters of hundreds to thousands of individual servers.

Google Kubernetes is highly flexible container tool to deliver even complex applications, consistently. Applications 'run on clusters of hundreds to thousands of individual servers."

**In this beginners Kubernetes tutorial, you will learn:**

* [Why you need containers?](https://www.guru99.com/kubernetes-tutorial.html#1)
* [What is Kubernetes?](https://www.guru99.com/kubernetes-tutorial.html#2)
* [What task are performed by Kubernetes?](https://www.guru99.com/kubernetes-tutorial.html#3)
* [Why use Kubernetes?](https://www.guru99.com/kubernetes-tutorial.html#4)
* [Features of Kubernetes](https://www.guru99.com/kubernetes-tutorial.html#5)
* [Kubernetes Basics](https://www.guru99.com/kubernetes-tutorial.html#6)
* [Kubernetes Architecture](https://www.guru99.com/kubernetes-tutorial.html#7)
* [Other Key Terminologies](https://www.guru99.com/kubernetes-tutorial.html#8)
* [Kubernetes vs. Docker Swarm](https://www.guru99.com/kubernetes-tutorial.html#9)
* [Disadvantages of Kubernetes](https://www.guru99.com/kubernetes-tutorial.html#10)

**What task are performed by Kubernetes?**

Kubernetes is the Linux kernel which is used for distributed systems. It helps you to be abstract the underlying hardware of the nodes(servers) and offers a consistent interface for applications that consume the shared pool of resources.

**Why use Kubernetes?**

Here, are pros/benefits of using Kubernetes.

* Kubernetes can run on-premises bare metal, OpenStack, public clouds Google, Azure, AWS, etc.
* Helps you to avoid vendor lock issues as it can use any vendor-specific APIs or services except where Kubernetes provides an abstraction, e.g., load balancer and storage.
* Containerization using kubernetes allows package software to serve these goals. It will enable applications that need to be released and updated without any downtime.
* Kubernetes allows you to assure those containerized applications run where and when you want and helps you to find resources and tools which you want to work.

**Features of Kubernetes**

Here are the essential features of Kubernetes:

* Automated Scheduling
* Self-Healing Capabilities
* Automated rollouts & rollback
* Horizontal Scaling & Load Balancing
* Offers environment consistency for development, testing, and production
* Infrastructure is loosely coupled to each component can act as a separate unit
* Provides a higher density of resource utilization
* Offers enterprise-ready features
* Application-centric management
* Auto-scalable infrastructure
* You can create predictable infrastructure

**Kubernetes Basics**

Here, are some important Basic of Kubernetes:

* **Cluster:**

It is a collection of hosts(servers) that helps you to aggregate their available resources. That includes ram, CPU, ram, disk, and their devices into a usable pool.

* **Master:**

The master is a collection of components which make up the control panel of Kubernetes. These components are used for all cluster decisions. It includes both scheduling and responding to cluster events.

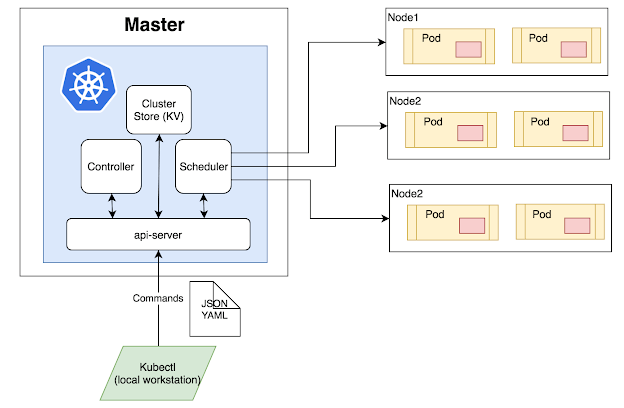
* **Node:**

It is a single host which is capable of running on a physical or virtual machine. A node should run both kube-proxy, minikube, and kubelet which are considered as a part of the cluster.

* **Namespace:**

It is a logical cluster or environment. It is a widely used method which is used for scoping access or dividing a cluster.

**Kubernetes Architecture**

[](https://www.guru99.com/images/1/061419_0430_KubernetesT1.png)Kubernetes Architecture Diagram

**Master Node**

The master node is the first and most vital component which is responsible for the management of Kubernetes cluster. It is the entry point for all kind of administrative tasks. There might be more than one master node in the cluster to check for fault tolerance.

The master node has various components like API Server, Controller Manager, Scheduler, and ETCD. Let see all of them.

API Server**:**The API server acts as an entry point for all the REST commands used for controlling the cluster.

**Scheduler**

The scheduler schedules the tasks to the slave node. It stores the resource usage information for every slave node. It is responsible for distributing the workload.

It also helps you to track how the working load is used on cluster nodes. It helps you to place the workload on resources which are available and accept the workload.

**Etcd**

etcd components store configuration detail and wright values. It communicates with the most component to receive commands and work. It also manages network rules and port forwarding activity.

**Pods**

A pod is the smallest and simplest unit in the Kubernetes object model. It represents a single instance of an [application](https://www.redhat.com/en/topics/cloud-native-apps). Each pod is made up of a container or a series of tightly coupled containers, along with options that govern how the containers are run. Pods can be connected to persistent storage in order to run stateful applications.

**Container runtime engine**

To run the containers, each compute node has a container runtime engine. [Docker](https://www.redhat.com/en/topics/containers/what-is-docker) is one example, but Kubernetes supports other Open Container Initiative-compliant runtimes as well, such as rkt and CRI-O.

**kubelet**

Each compute node contains a kubelet, a tiny application that communicates with the control plane. The kublet makes sure containers are running in a pod. When the control plane needs something to happen in a node, the kubelet executes the action.

**kube-proxy**

Each compute node also contains kube-proxy, a network proxy for facilitating Kubernetes networking services. The kube-proxy handles network communications inside or outside of your cluster—relying either on your operating system’s packet filtering layer, or forwarding the traffic itself.

**Worker/Slave nodes**

Worker nodes are another essential component which contains all the required services to manage the networking between the containers, communicate with the master node, which allows you to assign resources to the scheduled containers.

* Kubelet: gets the configuration of a Pod from the API server and ensures that the described containers are up and running.
* Docker Container: Docker container runs on each of the worker nodes, which runs the configured pods
* Kube-proxy:Kube-proxy acts as a load balancer and network proxy to perform service on a single worker node
* Pods: A pod is a combination of single or multiple containers that logically run together on nodes

**Other Key Terminologies**

**Replication Controllers**

A replication controller is an object which defines a pod template. It also controls parameters to scale identical replicas of Pod horizontally by increasing or decreasing the number of running copies.

**Replication sets**

Replication sets are an interaction on the replication controller design with flexibility in how the controller recognizes the pods it is meant to manage. It replaces replication controllers because of their higher replicate selection capability.

**Deployments**

Deployment is a common workload which can be directly created and manage. Deployment use replication set as a building block which adds the feature of life cycle management.

**Stateful Sets**

It is a specialized pod control which offers ordering and uniqueness. It is mainly used to have fine-grained control, which you have a particular need regarding deployment order, stable networking, and persistent data.

**Daemon Sets**

Daemon setsare another specialized form of pod controller that runs a copy of a pod on every node in the cluster. This type of pod controller is an effective method for deploying pods that allows you to perform maintenance and offers services for the nodes themselves.

**Kubernetes vs. Docker Swarm**

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Docker Swarm** | **Kubernetes** |
| Scaling | No Autoscaling | Auto-scaling |
| Load balancing | Does auto load balancing | Manually configure your load balancing settings |
| Storage volume sharing | Shares storage volumes with any other container | Shares storage volumes between multiple containers inside the same Pod |
| Use of logining and monitoring tool | Use 3rd party tool like ELK | Provide an in-built tool for logging and monitoring. |
| Installation | Easy & fast | Complicated & time-consuming |
| GUI | GUI not available | GUI is available |
| Scalability | Scaling up is faster than K8S, but cluster strength not as robust | Scaling up is slow compared to Swarm, but guarantees stronger cluster state Load balancing requires manual service configuration |
| Load Balancing | Provides a built-in load balancing technique | Process scheduling to maintain services while updating |
| Updates & Rollbacks Data Volumes Logging & Monitoring | Progressive updates and service health monitoring. | Only shared with containers in same Pod Inbuilt logging & monitoring tools. |

**Disadvantages of Kubernetes**

* Kubenetes dashboard not as useful as it should be
* Kubernetes is a little bit complicated and unnecessary in environments where all development is done locally.
* Security is not very effective.

**Summary:**

* Container helps an organization to perform maintenance and update without interrupting services
* Kubernetes is an example of a container management system developed in the Google platform.
* The biggest advantage of using Kubernetes is that it can run on-premises OpenStack, public clouds Google, Azure, AWS, etc.
* Kubernetes offers automated Scheduling and Self-Healing Capabilities.
* Cluster, master, node, and namespace are important basic of kubernetes
* Master node and work node are important components of Kubernetes architecture.
* Replication Controllers, Replication sets, Deployments, Stateful Sets, Daemon Sets are other important terms used in Kubernetes.
* Docker swarm does not allow auto-scaling while Kubernetes allows auto-scaling.
* The biggest drawback of Kubenetes is that it's dashboard not very useful and effective

**Top Kubernetes Interview Questions and Answers**

Here are Kubernetes Interview Questions for fresher as well as experienced candidates to get the dream job.

**1) What is Kubernetes?**

Kubernetes is a container management system developed in the Google platform. The purpose of kubernetes is to manage a containerized application in various types of physical, virtual, and cloud environments. Google Kubernetes is a highly flexible container tool to deliver even complex applications, consistently. Applications run on clusters of hundreds to thousands of individual servers.

**2) Define node in Kubernetes**

A node the smallest unit of hardware. It defines a single machine in a cluster that can be a virtual machine from a cloud provider or physical machine in the data center. Every machine available in the Kubernetes cluster can substitute other machines.

**3) What is the work of a kube-scheduler?**

Kube-scheduler is the default scheduler for Kubernetes. It assigns nodes to newly created pods.

**4) Define daemon sets**

Daemon sets are a set of pods that runs on a host. They are used for host layers attributes like monitoring network or simple network.

Skip Ad

**5) Define Heapster in Kubernetes**

A Heapster is a metrics collection and performance monitoring system for data that are collected by the Kublet.

**6) What tasks are performed by Kubernetes?**

Kubernetes is the Linux kernel which is used for distributed systems. It helps you to be abstract the underlying hardware of the nodes(servers) and offers a consistent interface for applications that consume the shared pool of resources.

**7) Define Kubernetes controller manager**

The controller manager is a daemon used for garbage collection, core control loops, and namespace creation. It enables the running of more than one process on the master node.

**8) Why use namespace in Kubernetes?**

Namespaces in Kubernetes are used for dividing cluster resources between users. It helps the environment where more than one user spread projects or teams and provides a scope of resources.

**9) Why use Kubernetes?**

Kubernetes is used because:

* Kubernetes can run on-premises bare metal, OpenStack, public clouds Google, Azure, AWS, etc.
* It helps you to avoid vendor lock issues as it can use any vendor-specific APIs or services except where Kubernetes provides an abstraction, e.g., load balancer and storage.
* It will enable applications that need to be released and updated without any downtime.
* Kubernetes allows you to assure those containerized apps run where and when you want and help you to find resources and tools which you want to work.

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

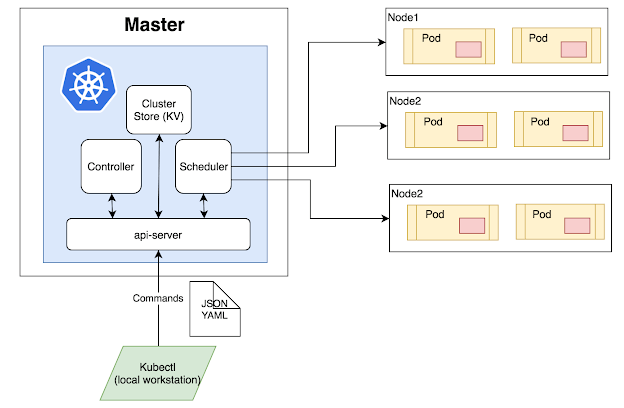
The features of Kubernetes are:

* Automated Scheduling
* Self-Healing Capabilities
* Automated rollouts & rollback
* Horizontal Scaling & Load Balancing
* Offers environment consistency for development, testing, and production
* Infrastructure is loosely coupled to each component can act as a separate unit
* Provides a higher density of resource utilization
* Offers enterprise-ready features
* Application-centric management
* Auto-scalable infrastructure
* You can create predictable infrastructure

**11) Mention the types of controller managers**

Types of controller managers are: 1) endpoints controller, 2) service accounts controller, 3) node controller, 4) namespace controller, 5) replication controller, 6) token controller.

**12) Explain Kubernetes Architecture**

[](https://www.guru99.com/images/1/061419_0430_KubernetesT1.png)Kubernetes Architecture Diagram

* **Master Node:**The master node is the first and most vital component which is responsible for the management of Kubernetes cluster. It is the entry point for all kinds of administrative tasks. There may be more than one master node in the cluster to check for fault tolerance.
* **API Server:** The API server acts as an entry point for all the REST commands used for controlling the cluster.
* **Scheduler:** The scheduler schedules the tasks to the slave node. It stores the resource usage information for every slave node. It is responsible for distributing the workload.
* **Etcd:**etcd components, store configuration detail, and wright values. It communicates with the most component to receive commands and work. It also manages network rules and port forwarding activity.
* **Worker/Slave nodes:**Worker nodes are another essential component that contains all the required services to manage the networking between the containers, communicate with the master node, which allows you to assign resources to the scheduled containers.
* **Kubelet:** It gets the configuration of a Pod from the API server and ensures that the described containers are up and running.
* **Docker Container:** Docker container runs on each of the worker nodes, which runs the configured pods.
* **Pods:** A pod is a combination of single or multiple containers that logically run together on nodes.

**13) List various services available in Kubernetes**

Various services available in Kubernetes are 1) Cluster IP service, 2) Load Balancer service, 3) Node Port service, 4) External Name Creation service.

**14) Define Cluster IP**

The Cluster IP is a Kubernetes service that offers a service inside the cluster that other apps inside cluster can access.

**15) 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.

**16) Define kubelet**

The kubelet is a service agent which controls and maintains group pf pods by checking pod specification using Kubernetes. The kubelet runs on each node and allows to communicate between a master node and a slave node.

**17) What are the disadvantages of Kubernetes?**

* Kubernetes dashboard is not as helpful as it should be
* Security is not very effective.
* It is very complex and can reduce productivity
* Kubernetes is more costly than its alternatives.

**18) 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.

**19)What is the difference between Kubernetes and Docker Swarm?**

The difference between Kubernetes and Docker Swarm is:

|  |  |
| --- | --- |
| **Kubernetes** | **Docker Swarm** |
| Kubernetes Provides an auto-scaling feature. | Docker Swarm does not provide an auto-scaling feature. |
| Manually configure your load balancing settings. | Does auto load balancing |
| Installation is complicated & time-consuming. | Installation is easy & fast. |
| GUI is available. | GUI not available. |
| It provides a built-in load balancing technique. | Process scheduling is done to maintain services while updating. |

**20) Define Ingress Network**

Ingress network is defined as a collection of rules which allow permission for connections into the Kubernetes cluster.

**21) 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.

**22) What is GKE?**

GKE or Google Container Engine is a management platform that supports clusters and Docker containers that run within public cloud services of Google.

**23) Why load balancer is needed?**

A load balancer is needed because it gives a standard way to distribute network traffic among different services, which runs in the backend.

**24) How to run Kubernetes locally?**

Kubernetes can be run locally using the Minikube tool. It runs a single-node cluster in a VM (virtual machine) on the computer. Therefore, it offers the ideal way for users who have just started learning Kubernetes.

**25) What are the tools that are used for container monitoring?**

Tools that are used for container monitoring are:

* Heapster
* cAdvisor
* Prometheus
* InfluxDB
* Grafana

**26) List components of Kubernetes**

There are three components of Kubernetes, they are:

* Addons
* Node components
* Master Components

**27) Define headless service**

Headless service is defined as a service that uses IP address, but instead of load balancing, it returns of associated pods.

**28) What are the important components of node status?**

The important component of node status are:

* Condition
* Capacity
* Info
* Address

**29) 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.

**30) Mention the uses of GKE**

The uses of the GKE (Google Kubernetes Engine) are:

* It can be used to create docker container clusters
* Resize application controllers
* Update and then upgrade the clusters of container
* Debug cluster of the container.
* GKE can be used to creates a replication controller, jobs, services, container pods, or load balancer.

**31) Define orchestration in Kubernetes**

Orchestration in Kubernetes defines as an automatic method of scheduling the work of every container. It is used for applications that are based on microservices within clusters.

**32) 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.

**33) List tools for container orchestration**

The tools for container orchestration are 1) Docker swarm, 2) Apache Mesos, and 3) Kubernetes.

**34) Mention the list of objects of Kubernetes?**

Objects that are used in Kubernetes are: 1) Pods, 2) Replication sets and controllers, 3) Jobs and cron jobs, 4) Daemon sets, 5) Distinctive identities, 6) Deployments, 7) and Stateful sets.

**35) 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.

**36) Why use Daemon sets?**

Daemon sets are used because:

* It enables to runs storage platforms like ceph and glusterd on each node.
* Daemon sets run the logs collection on every node such as filebeat or fluentd.
* It performs node monitoring on each and every node.

**37) 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.

**38) List out some important Kubectl commands:**

The important Kubectl commands are:

* kubectl annotate
* kubectl cluster-info
* kubectl attach
* kubectl apply
* kubectl config
* kubectl autoscale
* kubectl config current-context
* kubectl config set.

**39) 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.

**40) 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.

**41) 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.

**42) What are the objectives of the replication controller?**

The objectives of the replication controller are:

* It is responsible for controlling and administering the pod lifecycle.
* It monitors and verifies whether the allowed number of replicas are running or not.
* The replication controller helps the user to check the pod status.
* It enables to alter a pod. The user can drag its position the way interested in it.

**43) What do you mean by persistent volume?**

A persistent volume is a storage unit that is controlled by the administrator. It is used to manage an individual pod in a cluster.

**44) What are Secrets in Kubernetes?**

Secrets are sensitive information like login credentials of the user. They are objects in Kubernetes that stores sensitive information like username and password after performing encryption.

**45) 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.

**46) 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.

**47) Define K8s**

K8s (K-eight characters-S) is a term for Kubernetes. It is an open-source orchestration framework for the containerized applications.

**48) What are federated clusters?**

Federated clusters multiple clusters that are managed as a single cluster.

**49) Mention the difference between Docker volumes and Kubernetes Volumes**

|  |  |
| --- | --- |
| **Kubernetes Volumes** | **Docker Volumes** |
| Volumes are not limited to any container. | Volumes are limited to a pod in the container. |
| Kubernetes volumes support all containers deployed in a pod of Kubernetes. | Docker volumes do not support all containers deployed in Docker. |

**50) What are the ways to provide API-Security on Kubernetes?**

The ways to provide API-Security on Kubernetes are:

* Using correct auth mode with API server authentication mode= Node.
* Make kubeless that protects its API via authorization-mode=Webhook.
* Ensure the kube-dashboard uses a restrictive RBAC (Role-Based Access Control) policy

**51) What is ContainerCreating pod?**

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

**52) What are the types of Kubernetes Volume?**

The types of Kubernetes Volume are:

* EmptyDir
* GCE persistent disk
* Flocker
* HostPath
* NFS
* ISCSI
* rbd
* PersistentVolumeClaim
* downwardAPI

**53) 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.

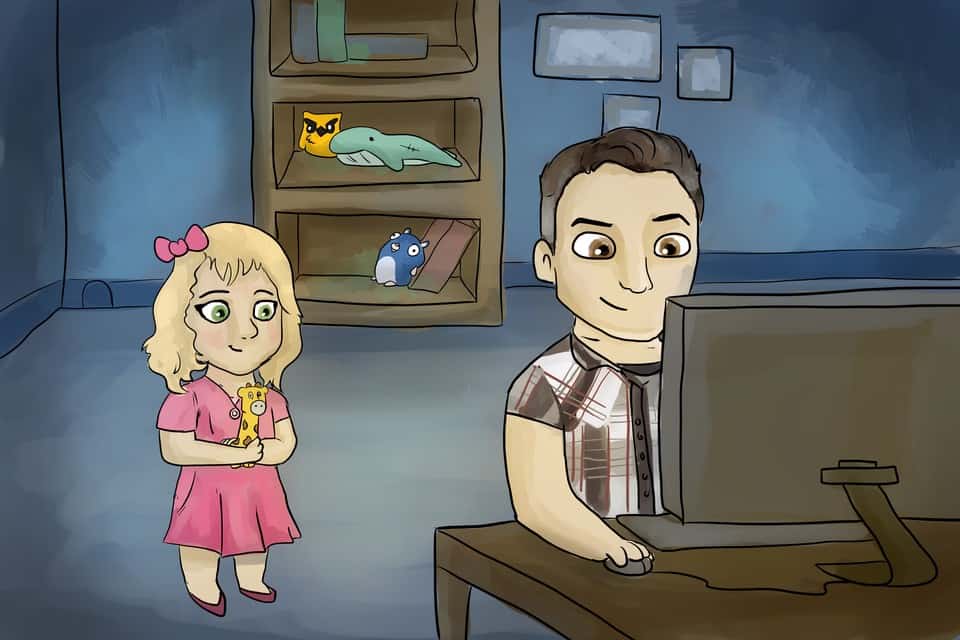
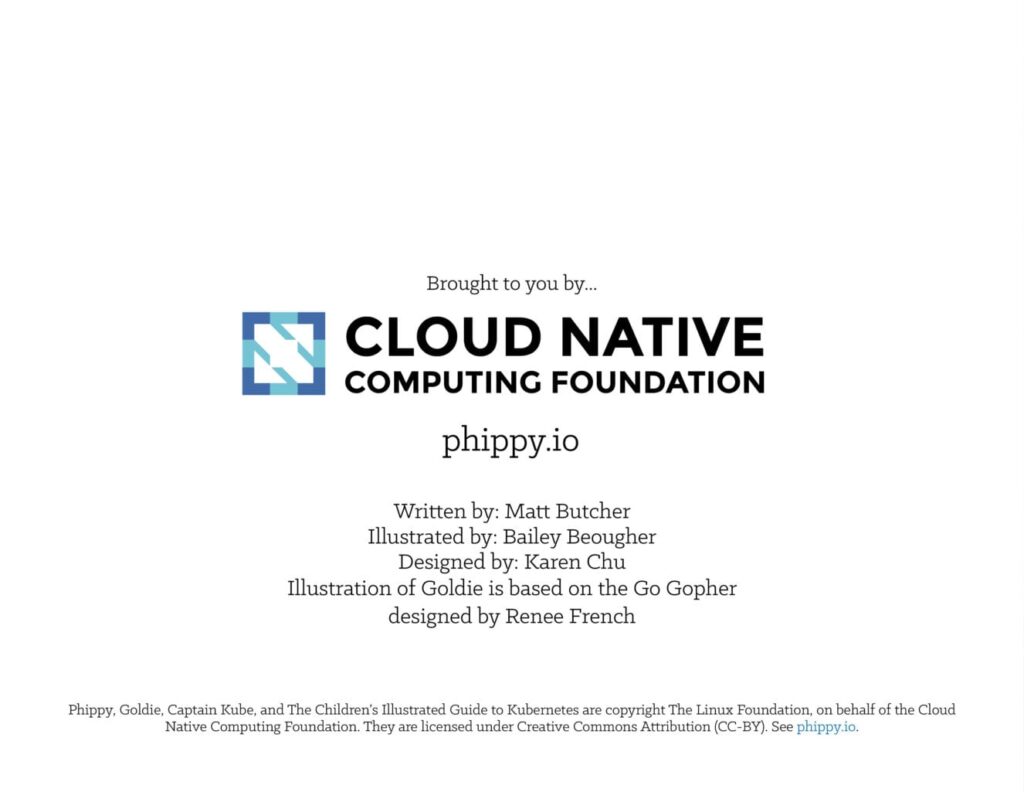
**54) What is the Kubernetes Network Policy?**

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

**55) What is Kubernetes proxy service?**

Kubernetes proxy service is a service which runs on the node and helps in making it available to an external host.

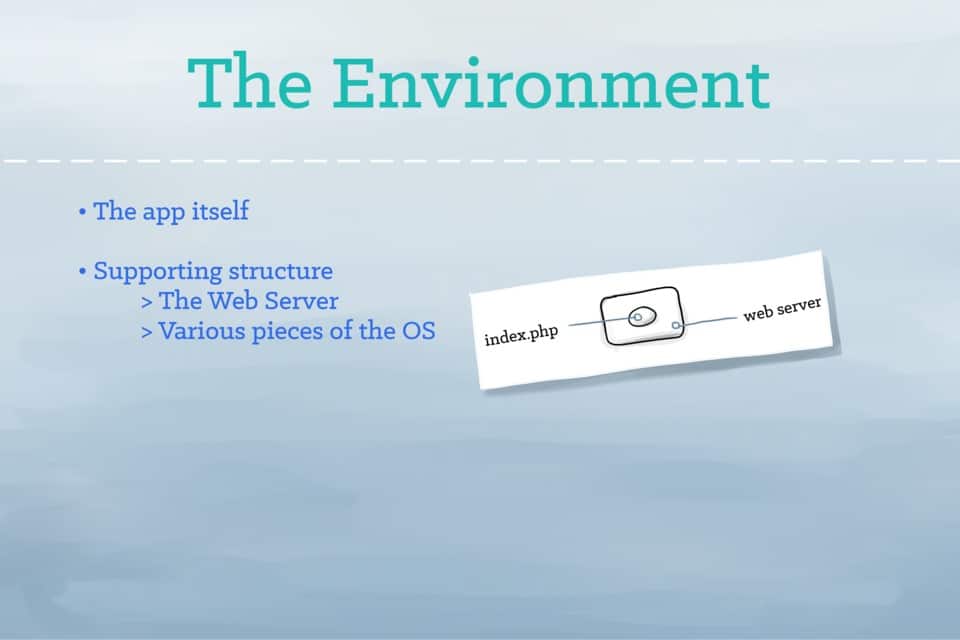
# Guide to Kubernetes



Dedicated to all the parents who try to explain software engineering to their children.



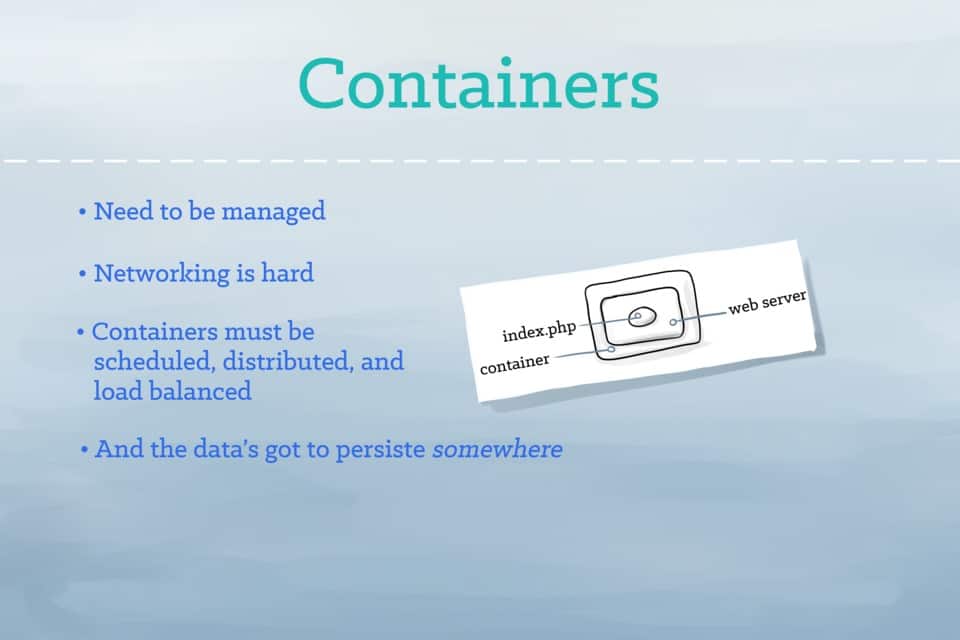
Once upon a time there was an app named Phippy. And she was a simple app. She was written in PHP and had just one page. She lived on a hosting provider and she shared her environment with scary other apps that she didn’t know and didn’t care to associate with. She wished she had her own environment: just her and a webserver she could call home.



An app has an environment that it relies upon to run. For a PHP app, that environment might include a webserver, a readable file system, and the PHP engine itself.



One day, a kindly whale came along. He suggested that little Phippy might be happier living in a container. And so the app moved. And the container was nice, but… It was a little bit like having a fancy living room floating in the middle of the ocean.



A container provides an isolated environment in which an app, together with its environment, can run. But those isolated containers often need to be managed and connected to the external world. Shared file systems, networking, scheduling, load balancing, and distribution are all challenges.



The whale shrugged his shoulders. “Sorry, kid,” he said, and disappeared beneath the ocean’s surface. But before Phippy could even begin to despair, a captain appeared on the horizon, piloting a gigantic ship. The ship was made of dozens of rafts all lashed together, but from the outside, it looked like one giant ship.

“Hello there, friend PHP app. My name is Captain Kube” said the wise old captain.



“Kubernetes” is the Greek word for a ship’s captain. We get the words Cybernetic and Gubernatorial from it. The Kubernetes project focuses on building a robust platform for running thousands of containers in production.



“I’m Phippy,” said the little app.

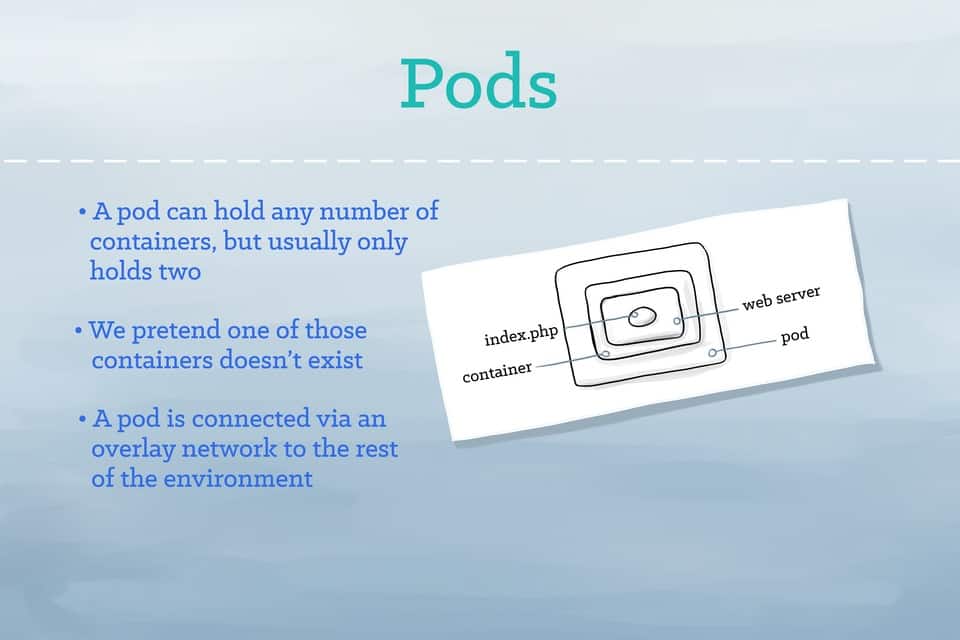
“Nice to make your acquaintance,” said the Captain as he slapped a name tag on her.



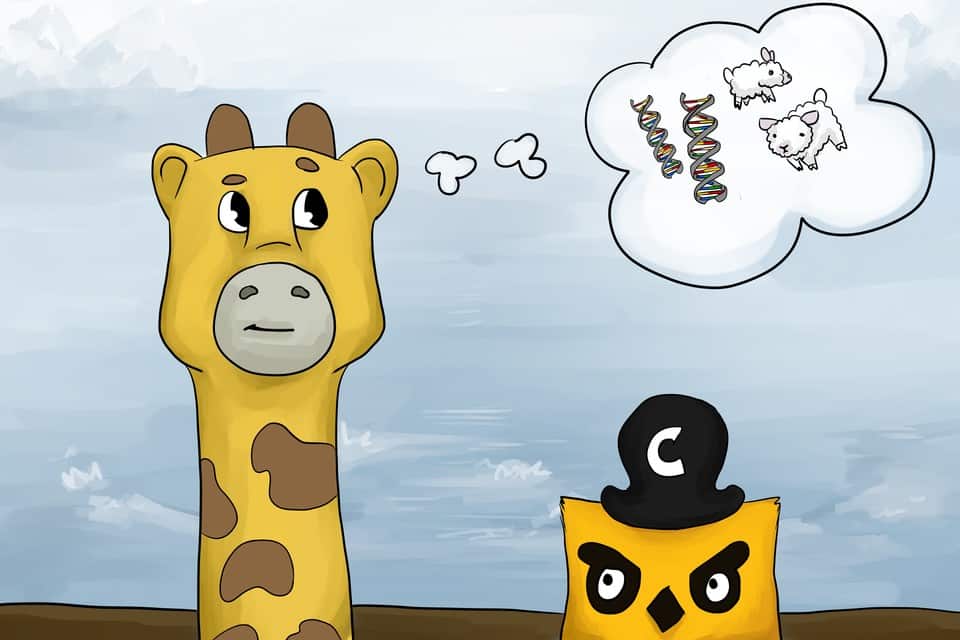
Kubernetes uses labels as “nametags” to identify things. And it can query based on these labels. Labels are open-ended: You can use them to indicate roles, stability, or other important attributes.



Captain Kube suggested that the app might like to move her container to a pod on board the ship. Phippy happily moved her container inside of the pod aboard Kube’s giant ship. It felt like home.

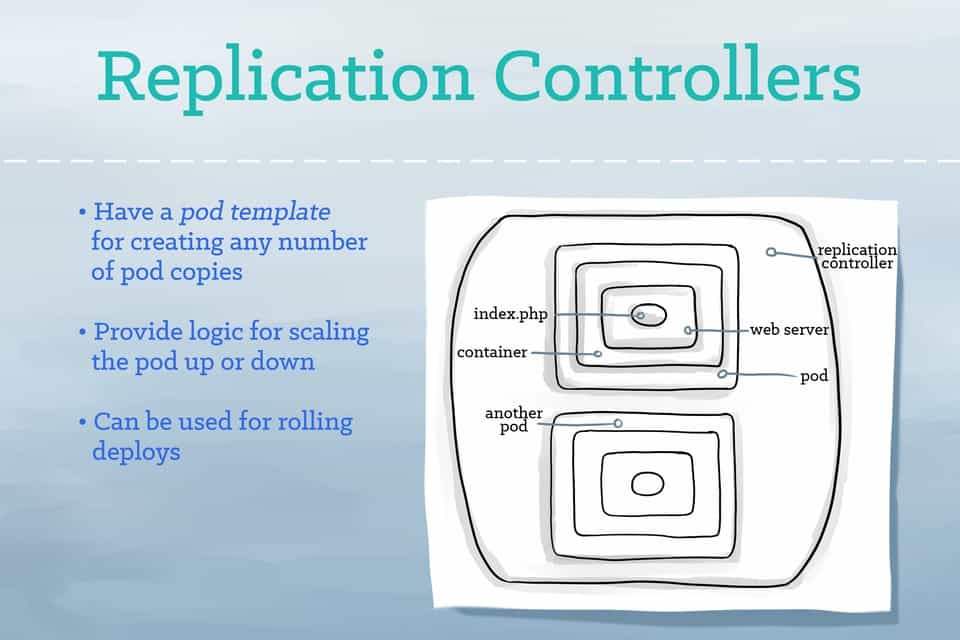


In Kubernetes, a Pod represents a runnable unit of work. Usually, you will run a single container inside of a Pod. But for cases where a few containers are tightly coupled, you may opt to run more than one container inside of the same Pod. Kubernetes takes on the work of connecting your pod to the network and the rest of the Kubernetes environment.



Phippy had some unusual interests. She was really into genetics and sheep. And so she asked the captain, “What if I want to clone myself… On demand… Any number of times?”

“That’s easy,” said the captain. And he introduced her to the replication controllers.



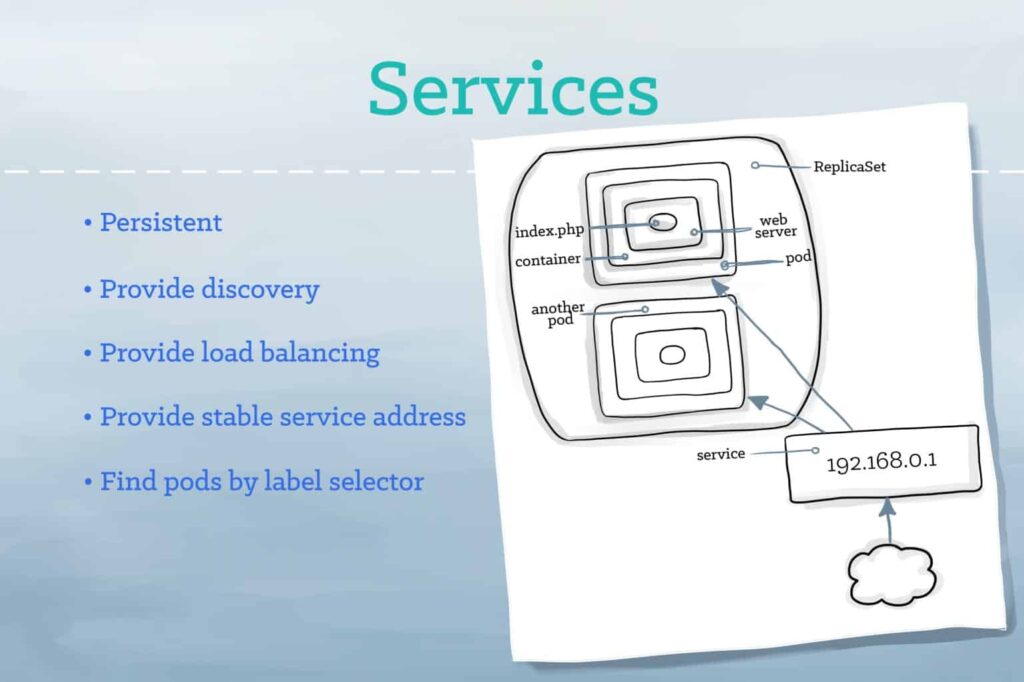
Replication controllers provide a method for managing an arbitrary number of pods. A replication controller contains a pod template, which can be replicated any number of times. Through the replication controller, Kubernetes will manage your pods’ lifecycle, including scaling up and down, rolling deployments, and monitoring.



For many days and nights the little app was happy with her pod and happy with her replicas. But only having yourself for company is not all it’s cracked up to be…. even if it is N copies of yourself.

Captain Kube smiled benevolently, “I have just the thing.”

No sooner had he spoken than a tunnel opened between Phippy’s replication controller and the rest of the ship. With a hearty laugh, Captain Kube said, “Even when your clones come and go, this tunnel will stay here so you can discover other pods, and they can discover you!”



A service tells the rest of the Kubernetes environment (including other pods and replication controllers) what services your application provides. While pods come and go, the service IP addresses and ports remain the same. And other applications can find your service through Kurbenetes service discovery.

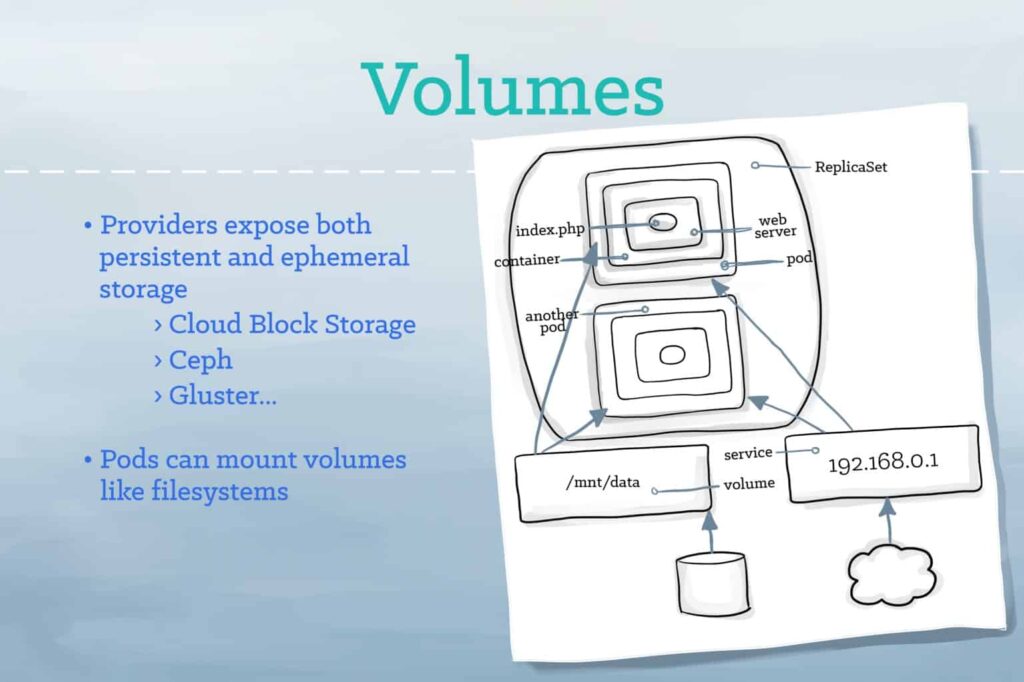


Thanks to the services, Phippy began to explore the rest of the ship. It wasn’t long before Phippy met Goldie. And they became the best of friends. One day, Goldie did something extraordinary. She gave Phippy a present. Phippy took one look and the saddest of sad tears escaped her eye.

“Why are you so sad?” asked Goldie.

“I love the present, but I have nowhere to put it!” sniffled Phippy.

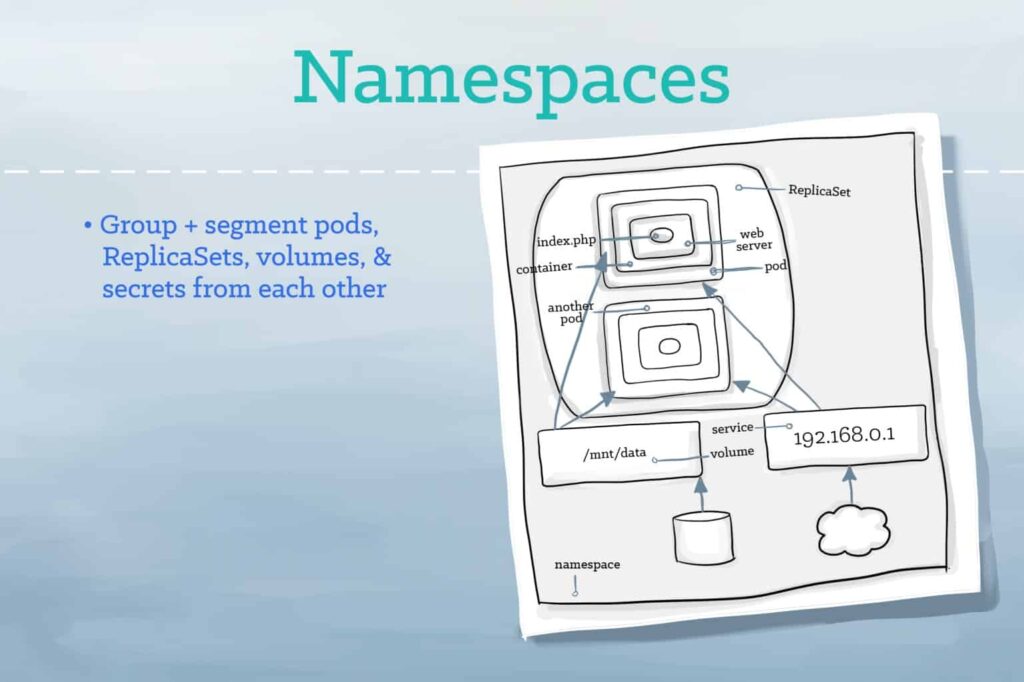
But Goldie knew what to do. “Why not put it in a volume?”



A volume represents a location where containers can access and store information. For the application, the volume appears as part of the local filesystem. But volumes may be backed by local storage, Ceph, Gluster, Elastic Block Storage, and a number of other storage backends.

Phippy loved life aboard Captain Kube’s ship and she enjoyed the company of her new friends (every replicated pod of Goldie was equally delightful). But as she thought back to her days on the scary hosted provider, she began to wonder if perhaps she could also have a little privacy.

“It sounds like what you need,” said Captain Kube, “is a namespace.”



A namespace functions as a grouping mechanism inside of Kubernetes. Services, pods, replication controllers, and volumes can easily cooperate within a namespace, but the namespace provides a degree of isolation from the other parts of the cluster.



Life was good aboard Captain Kube’s boat. Together with her new friends, Phippy sailed the seas. She had many grand adventures, but most importantly, Phippy had found her home.

And so Phippy lived happily ever after.