**Kubernetes**

|  |  |
| --- | --- |
| **DOCUMENT DETAILS** | |
| About the document | It is a theoretical article that briefly explains Kubernetes' core concepts, configuration, security, networking, etc. |
| Document Owner | Reshu Vishwakarma |

1. **CORE CONCEPTS**
   1. **Pods –** Smallest deployable unit of computing that you can create and manage in Kubernetes cluster
   2. **Replica Set –** Maintains a stable set of replica pods running at any given time.
   3. **Deployments –** Provides declarative updates for Pods and Replica sets
   4. **Namespaces –** Isolates group of resources within a single cluster
   5. **Imperative Commands –** Manages live objects. Used for creating, updating and deleting K8s objects
2. **CONFIGURATION**
   1. **Commands and arguments –** Defined in configuration file overwrites the commands and arguments provided in container image. Command fields may correspond to entry point for container runtimes.
   2. **Config Maps –** Stores non-confidential data in key-value pairs. Pods can consume Config Maps as environment variables, command line arguments, configuration file in a volume.
   3. **Secret –** Stores confidential data such as password, token or key.
   4. **Security Contexts –** Defines privilege and access control settings for a pod or container.
   5. **Resource Requests and Limits -**  are defined in POD specification and illustrates how much resources such as CPU and memory a container needs. Kubernetes scheduler uses them while allocating POD to a particular node. Kubelet uses these values to ensure that a POD does not use the resources more than the limit specified.
   6. **Service Account –** Used for authentication. It provides an identity for processes that run in a POD and maps it to Service Account object.
   7. **Taints and Tolerations –** Taints allow nodes to repel a set of Pods. Tolerations allow scheduler to schedule Pods with a matching taint.
   8. **Node Affinity –** Is a property of a Pod that attracts it to a set of nodes.
3. **MULTI CONTAINER POD-** 
   1. **Init Container-** Runs before app container within a Pod. Can contain utilities or setup scripts that are not present in application container.
   2. **Readiness Probes –** Kubelet uses readiness probes to know when Pod is ready to start accepting the traffic.
4. **POD DESIGN –** 
   1. **Labels and selectors –** Label is a key-value identifying attribute of an object. Selector is the core grouping primitive in Kubernetes, which can be either equity-based or set-based.
   2. **Rolling updated and rollback-** Rolling update results to zero downtime deployment, as it updates the pod instances with new ones using incremental approach. Rollback updates the revision of the deployment.
   3. **Jobs and Cron Jobs –** These are automated jobs that are useful for creating periodic and recurring tasks, like running backup and sending mails.
5. **SERVICES AND NETWORKING –** 
   1. **Kubernetes Services –** Abstract access to Kubernetes Pods
   2. **Network Policies –** specify how a pod can allow to communicate with various network entities (endpoint/ services)
   3. **Ingress Networking –** An API object that manages external access to the services in a cluster
6. **STATE PERSISTENCE –** 
   1. **Persistent Volume –** is a storage cluster resource. Unlike volume, it has its lifecycle independent of Pod that uses it
   2. **Persistent Volume Claim –** is a request for storage by a user

**References**

[**https://kubernetes.io/docs/home/**](https://kubernetes.io/docs/home/)