Kubernetes Architecture

Kubernetes follows a **Master-Worker Node** architecture, where the **Control Plane (Master Node)** manages the **Worker Nodes** to run containerized applications.

Control Plane / Master Node

The control plane's components make global decisions about the cluster (e.g., scheduling) and detect/respond to cluster events (e.g., starting a new pod when a deployment's replicas field is unsatisfied). Control plane components can be run on any machine in the cluster. User containers should not be run on this machine.

Master Node Components:

1. **API Server**: The central control point for all interactions with the cluster. It exposes the Kubernetes API and handles requests from users and other components.
2. **Scheduler**: Assigns workloads (pods) to worker nodes based on resource requirements, constraints, and other policies.
3. **Controller Manager**: Runs various controllers that monitor the cluster state and drive it toward the desired state. Examples include:
    a. Replication Controller
    b. Node Controller
    c. Service Controller
4. **etcd**: A distributed key-value store used to store cluster state and configuration data.

Node Components / Worker Nodes

Node components run on every node, maintaining running pods and providing the Kubernetes runtime environment.

Kubernetes Commands

1. Create a Pod

```
kubectl run <pod-name> --image=<image-name> --port=<container-port>
kubectl run my-pod --image=nginx --port=80
```

2. View All Pods

In Default Namespace:

```
kubectl get pods
```

In All Namespaces:

```
kubectl get pods -A
```

In a Specific Namespace:

```
kubectl get pods -n kube-system
```

For a Specific Pod:

```
kubectl get pods <pod-name>
kubectl get pods <pod-name> -o wide
kubectl get pods <pod-name> -o yaml
kubectl get pods <pod-name> -o json
```

3. Describe a Pod

```
kubectl describe pod <pod-name>
kubectl describe pod my-pod
```

4. View Logs of a Pod

```
kubectl logs <pod-name>
kubectl logs my-pod
kubectl exec <pod-name> -- <command>
```

Pod Definition YAML

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
  labels:
    app: my-web-app
spec:
  containers:
  - name: nginx-container
    image: ashilin20/app:latest
    ports:
    - containerPort: 80
```

Services

A **Service** is an abstraction that defines a logical set of pods and a policy to access them. It enables network connectivity and load balancing for pods.

Service Types

1. **NodePort**: Exposes the service on a static port on each selected node's IP, making it accessible externally using `<NodeIP>:<NodePort>`.
2. **ClusterIP**: Exposes the service on a cluster-internal IP, making it reachable only within the cluster.

3. **LoadBalancer**: Creates an external load balancer in cloud environments to route traffic to the service.

Deployment Commands

1. Create Deployment

```
kubectl create -f web-deploy.yml
```

Modify Deployment:

```
kubectl replace -f web-deploy.yml
```

2. View Deployments

```
kubectl get deployments
kubectl get deploy
kubectl get deploy -o wide
kubectl get deploy <deployment-name> -o json
kubectl get deploy <deployment-name> -o yaml
```

3. View Deployment Description

```
kubectl describe deploy <deployment-name>
```

4. Modify Deployment YAML

```
kubectl edit deploy <deployment-name>
```

5. Apply Changes

```
kubectl apply -f web-deploy.yml
```

6. Scale Deployment

```
kubectl scale deploy <deployment-name> --replicas=<desired-replica-count>
```

7. Delete Deployment

```
kubectl delete deploy <deployment-name>
kubectl delete -f web-deploy.yml
```

Deployment YAML

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 4
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image: ashilin20/app:latest
        ports:
```

```
        - containerPort: 7070
```

ReplicaSet Commands

1. Create ReplicaSet

```
kubectl create -f rs-test.yml
kubectl apply -f rs-test.yml
kubectl replace -f rs-test.yml
```

2. View ReplicaSets

```
kubectl get replicasets
kubectl get rs
kubectl get rs -o wide
kubectl get rs <replica-set-name> -o json
kubectl get rs <replica-set-name> -o yaml
```

3. View ReplicaSet Description

```
kubectl describe rs <replica-set-name>
```

4. Modify ReplicaSet YAML

```
kubectl edit rs <replica-set-name>
```

5. Scale ReplicaSet

```
kubectl scale replicaset <replicaset-name> --replicas=<desired-
replica-count>
```

ReplicaSet YAML

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-rs
  labels:
    name: my-rs
spec:
  replicas: 4
  selector:
    matchLabels:
      apptype: web-backend
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image: ashilin20/app:latest
        ports:
        - containerPort: 8081
```

Namespaces

A **Namespace** is a virtual cluster or logical partition within a cluster that provides a way to organize and isolate resources. It allows multiple teams or projects to share the same physical cluster while maintaining resource separation and access control.

Namespace Commands

Create Namespace:

```
kubectl create namespace <namespace-name>
kubectl create ns my-bank
```

Switch to a Specific Namespace:

```
kubectl config set-context --current --namespace=<namespace-name>
```

List All Namespaces:

```
kubectl get namespaces
```

Get Resources in a Namespace:

```
kubectl get <resource-type> -n <namespace-name>
```

Delete Namespace:

```
kubectl delete namespace <namespace-name>
kubectl delete ns my-bank
```

This structured and formatted version improves clarity and usability.

```
Name:             my-pod
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Thu, 20 Mar 2025 04:39:31 +0000
Labels:           run=my-pod
Annotations:      <none>
Status:           Running
IP:               10.244.0.4
IPs:
  IP:  10.244.0.4
Containers:
  my-pod:
    Container ID:   docker://e576710bd5e50daae7808cda39897f65f59820e3e478636c684a945b9c7a1136
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:124b44bfc9ccd1f3cedf4b592d4d1e8bddb78b51ec2ed5056c52d3692baebc19
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Thu, 20 Mar 2025 04:40:52 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-m7tf8 (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-m7tf8:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
```

```
<none>                          <none>          8918adc2891d   17 hours ago   326MB
<none>                          <none>          76435f253459   17 hours ago   326MB
<none>                          <none>          b317a1109b7f   17 hours ago   326MB
<none>                          <none>          942535685e8e   17 hours ago   326MB
<none>                          <none>          8a4a596c9091   17 hours ago   326MB
<none>                          <none>          e460c86021a1   18 hours ago   326MB
<none>                          <none>          079f5d5095b2   18 hours ago   326MB
<none>                          <none>          a4c6c5329cc8   18 hours ago   326MB
<none>                          <none>          85f479806b0f   18 hours ago   326MB
<none>                          <none>          7b2dbb1b307e   18 hours ago   326MB
jenkins/jenkins                 lts             6a44d1dd2d60   2 weeks ago    468MB
nginx                           latest          53a18edff809   6 weeks ago    192MB
mysql                           latest          fa262c3a6564   8 weeks ago    797MB
hello-world                     latest          74cc54e27dc4   8 weeks ago    10.1kB
kicbase/stable                  v0.0.46         e72c4cbe9b29   2 months ago   1.31GB
openjdk                         17-jdk-alpine   264c9bdce361   3 years ago    326MB
```


```
<none>                          <none>          8918adc2891d   17 hours ago   326MB
<none>                          <none>          76435f253459   17 hours ago   326MB
<none>                          <none>          b317a1109b7f   17 hours ago   326MB
<none>                          <none>          942535685e8e   17 hours ago   326MB
<none>                          <none>          8a4a596c9091   17 hours ago   326MB
<none>                          <none>          e460c86021a1   18 hours ago   326MB
<none>                          <none>          079f5d5095b2   18 hours ago   326MB
<none>                          <none>          a4c6c5329cc8   18 hours ago   326MB
<none>                          <none>          85f479806b0f   18 hours ago   326MB
<none>                          <none>          7b2dbb1b307e   18 hours ago   326MB
jenkins/jenkins                 lts             6a44d1dd2d60   2 weeks ago    468MB
nginx                           latest          53a18edff809   6 weeks ago    192MB
mysql                           latest          fa262c3a6564   8 weeks ago    797MB
hello-world                     latest          74cc54e27dc4   8 weeks ago    10.1kB
kicbase/stable                  v0.0.46         e72c4cbe9b29   2 months ago   1.31GB
openjdk                         17-jdk-alpine   264c9bdce361   3 years ago    326MB
```