Kubernetes, often abbreviated as K8s, is an open-source platform designed to automate the deployment, scaling, and management of containerized applications. Its architecture is designed to distribute the workload across a cluster of machines, allowing for high availability, fault tolerance, and scalability. Here's an overview of the key components of Kubernetes architecture:

**1. Cluster**

A Kubernetes cluster consists of at least one master node and multiple worker nodes that host the pods in which the containers run.

**2. Master Node**

The master node is responsible for managing the state of the cluster. It makes global decisions about the cluster (like scheduling), and detects and responds to cluster events (like starting up a new pod when a deployment's replicas field is unsatisfied).

* **API Server:** Serves as the front end for Kubernetes. The users, management tools, and components of the cluster all communicate through the API server.
* **etcd:** A consistent and highly-available key value store used as Kubernetes' backing store for all cluster data.
* **Scheduler:** Watches for newly created pods with no assigned node, and selects a node for them to run on based on resource availability.
* **Controller Manager:** Runs controller processes, which handle routine tasks in the cluster. These tasks include replicating pods, tracking worker nodes, handling node failures, etc.

**3. Worker Nodes**

These nodes do the actual work of running applications. Each worker node contains the services necessary to run pods, managed by the master components.

* **Kubelet:** An agent that runs on each node in the cluster. It makes sure that containers are running in a pod.
* **Kube-Proxy:** Maintains network rules on nodes. These rules allow network communication to your pods from network sessions inside or outside of your cluster.
* **Container Runtime:** The software that is responsible for running containers. Kubernetes supports several container runtimes: Docker, containerd, CRI-O, and any implementation of the Kubernetes CRI (Container Runtime Interface).

**4. Pods**

The basic scheduling unit in Kubernetes, which includes one or more containers that are guaranteed to be co-located on the host machine and can share resources.

**5. Control Plane**

The collection of master components that regulate the state of the cluster. This plane makes global decisions about the cluster (like scheduling), and detects and responds to cluster events (like starting up a new pod when a deployment's replicas field is unsatisfied).

**6. Services and Ingress**

* Service: Defines a logical set of pods and a policy by which to access them. This abstraction enables pod scaling and replacement.
* Ingress: Manages external access to the services in a cluster, typically HTTP.

**7. ConfigMaps and Secrets**

Provide a way to store and manage sensitive information like passwords, OAuth tokens, and ssh keys. Using these Kubernetes objects, you can provide containerized applications with configuration and secret data securely.

**8. Volumes**

Provide a way for a container to persist data. Kubernetes supports multiple types of volumes that a pod can use.

**9. Namespaces**

Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces.