Kubernetes introduction

Container orchestration

Container Orchestration

Why we need container orchestration?

Restart containers if they are not healthy.

Provide private container network.

Service discovery.

Manage cluster

Container orchestrations

Swarm

Hashicorp nomad

Kubernetes

Mesosphere

“A single container host by itself is like a tree without a forest.”

Kubernetes

What is Kubernetes?

Kubernetes is a platform for hosting docker containers in a clustered environment with multiple docker hosts

Project was started by google

Contributers: Google, CoreOS, Redhat, Mesosphere, Microsoft, HP, IBM, Vmware, ...

Kubernetes features

Schedule containers to physical machines

Service discovery

Load balancing

Auto healing

Scaling features

Kubernetes features

Automated rollouts and rollbacks

Storage orchestration

Secret management

Zero downtime deploy/update

Kubernetes components

ETCD

API Server

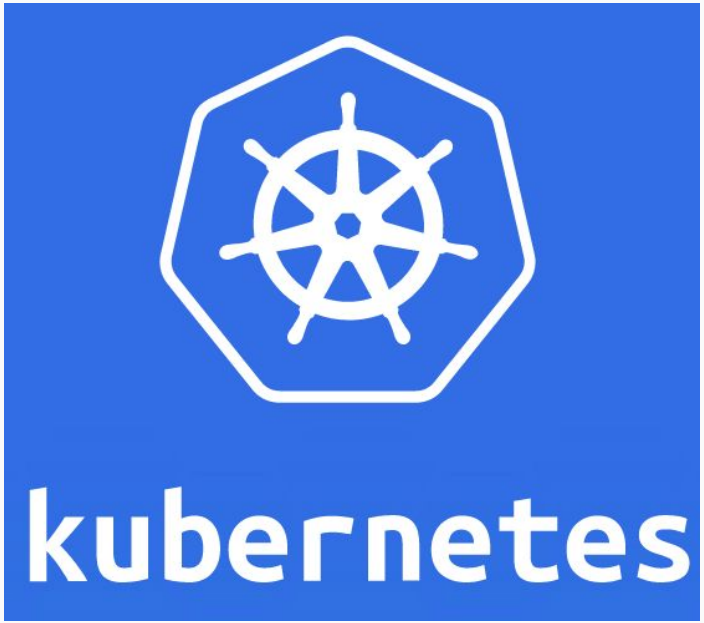
Scheduler

Controller Manager

Proxy

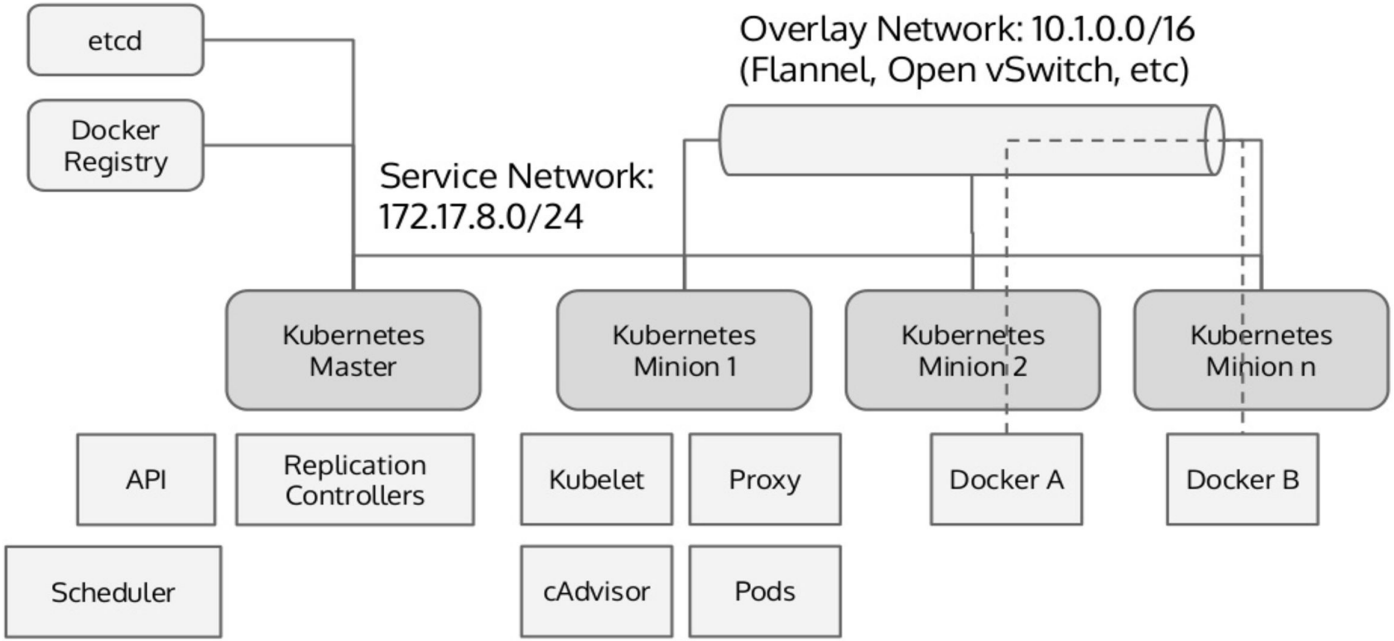
Kubelet

cAdvisor



“Kube may be a physical node or a vm”

Kubernetes Architecture



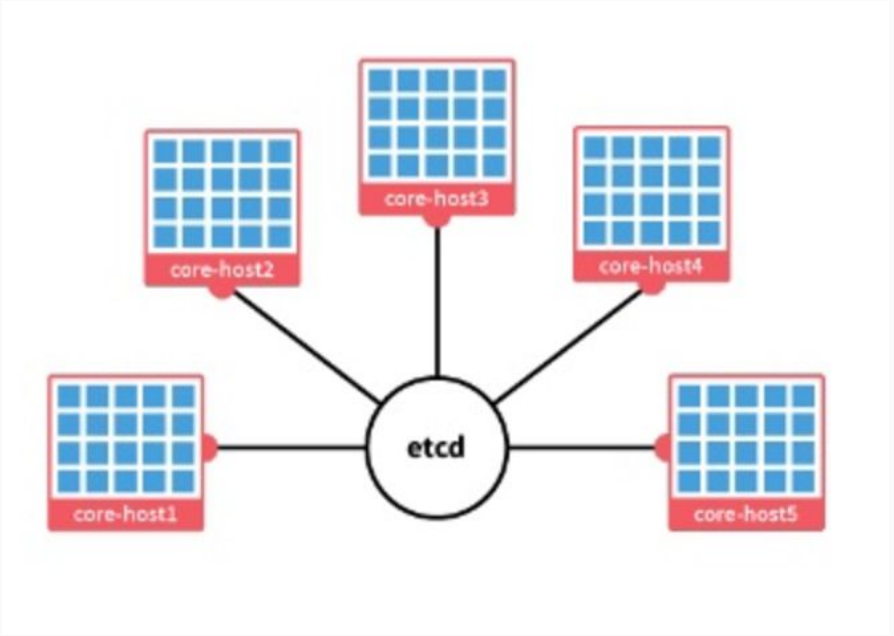
ETCD

Distributed key value store

Like a directory tree

JSON/REST API

Use a discovery url



Kubernetes API Server

The Kubernetes API server validates and configures data for the api objects which include pods, services, replicationcontrollers, and others. The API Server services REST operations and provides the frontend to the cluster’s shared state through which all other components interact.

Kubernetes Scheduler

The Kubernetes scheduler is a policy-rich, topology-aware, workload-specific function that significantly impacts availability, performance, and capacity. The scheduler needs to take into account individual and collective resource requirements, quality of service requirements, hardware/software/policy constraints, affinity and anti-affinity specifications, data locality, inter-workload interference, deadlines, and so on. Workload-specific requirements will be exposed through the API as necessary.

Kubernetes Controller Manager

The Kubernetes controller manager is a daemon that embeds the core control loops shipped with Kubernetes. In applications of robotics and automation, a control loop is a non-terminating loop that regulates the state of the system. In Kubernetes, a controller is a control loop that watches the shared state of the cluster through the apiserver and makes changes attempting to move the current state towards the desired state. Examples of controllers that ship with Kubernetes today are the replication controller, endpoints controller, namespace controller, and serviceaccounts controller.

Kubernetes Proxy

The Kubernetes network proxy runs on each node. This reflects services as defined in the Kubernetes API on each node and can do simple TCP,UDP stream forwarding or round robin TCP,UDP forwarding across a set of backends. Service cluster ips and ports are currently found through Docker-links-compatible environment variables specifying ports opened by the service proxy. There is an optional addon that provides cluster DNS for these cluster IPs. The user must create a service with the apiserver API to configure the proxy.

Kubernetes Kubelet

The kubelet is the primary “node agent” that runs on each node. The kubelet works in terms of a PodSpec. A PodSpec is a YAML or JSON object that describes a pod. The kubelet takes a set of PodSpecs that are provided through various mechanisms (primarily through the apiserver) and ensures that the containers described in those PodSpecs are running and healthy. The kubelet doesn’t manage containers which were not created by Kubernetes.

Pods

Smallest deployable unit of computing

Group of one or more containers

Pod has one ip address (localhost)

Containers access to shared volume

Replication sets

Ensure that a specified number of pod replicas are running

If there are too many, it will kill them

If there are too few, it will start more

Deployment

Provides declarives updates for pods/replica sets

Manages one or more replica sets

Primary mechanism for interacting with pods

Automatic rollouts and rollbacks

Daemon sets

Deploy a pod in all nodes

Ensure number of replications

If one pod on node exited, It will restart it

Service

An abstraction which defines a logical set of pods

Provides a mechanism to accessing them

Types: loadbalancer, clusterip, nodeport

Let’s play with kubernetes!

?

Thanks!

Mail: contact@salarmgh.me
Twitter: @salarmgh