Kubernetes

Module Overview

* What is Kubernetes ?
* Kubernetes & Swarm
* Kubernetes Cluster Architecture
* Kubernetes Features
* Installing Kubernetes
* Running Applications in Kubernetes

What is Kubernetes ?

- Open source system for automating deployment, scaling and management of containerized applications
- It groups containers that makes up an application, into logical units, for easy management and discovery.
- Planet Scale
  - Designed on the same principles that allows Google to run billions of containers a week, Kubernetes can scale without increasing your ops team.
- Never Outgrow
  - Whether testing locally or running a global enterprise, Kubernetes flexibility grows with you to deliver your applications consistently and easily no matter how complex your need is.
- Run Anywhere
  - Kubernetes is open source giving you the freedom to take advantage of on-premises, hybrid, or public cloud infrastructure, letting you effortlessly move workloads to where it matters to you.

Kubernetes & Swarm

Kubernetes

- Complex setup but a strong resultant cluster
- Support any container runtime
- K8s offers more possibilities for customization
- Optimized for one single large cluster
- Complex installation and configuration, especially when using it on-premises
- High modularity and extensibility

Swarm

- Simple installation but the resultant cluster is not comparatively strong
- Supports only docker container runtime
- While the Swarm API offers ease in leveraging Docker with similar functionality, it isn’t easy to perform operations that aren’t covered under the API
- Optimized for smaller multiple clusters
- It doesn’t have a steep learning curve – most Docker commands can also work in Swarm

Kubernetes Features

- Service discovery and load balancing
  - Expose a container using the DNS name or using their own IP address. If traffic to a container is high, Kubernetes is able to load balance and distribute the network traffic so that the deployment is stable
- Storage orchestration
  - Allows you to automatically mount a storage system of your choice, such as local storages, public cloud providers, and more.IPv4/IPv6 dual stack
- Automated rollouts and rollbacks
  - Describe the desired state for your deployed containers using Kubernetes, and it can change the actual state to the desired state at a controlled rate
- Automatic bin packing
  - You tell Kubernetes how much CPU and memory (RAM) each container needs
- Self Healing
  - Kubernetes restarts containers that fail, replaces containers, kills containers that don't respond to your user-defined health check
- Secret and Configuration Management
  - Kubernetes lets you store and manage sensitive information, such as passwords, OAuth tokens, and ssh keys

Kubernetes comprises a set of independent, composable control processes that continuously drive the current state towards the provided desired state. It shouldn't matter how you get from A to C. Centralized control is also not required. This results in a system that is easier to use and more powerful, robust, resilient, and extensible.

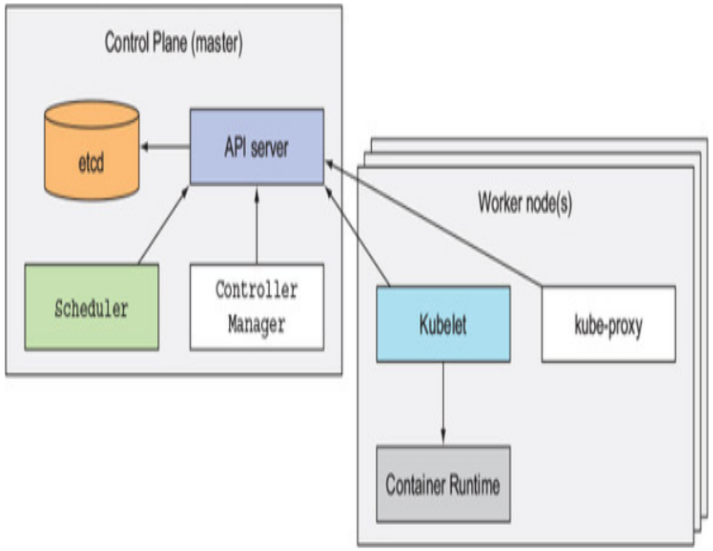
Kubernetes Does Not

* Limit the application it supports
* Build the application nor deploy source code
* Provide application level services
* Dictate alerting nor monitoring
* Provide nor adopt any comprehensive machine configuration

Kubernetes Architecture

Master components provide the cluster's control plane

* Master Components
  * Master components make global decisions about the cluster
  * Master components can be run on any machine in the cluster
* Node Components
  * Host the pods that are the components of the application
  * Node components run on every node, maintaining running pods and providing the Kubernetes runtime environment.



The Control Plane

Master components

The components of the Control Plane hold and control the state of the cluster, but they don't run your applications

- The Kubernetes API Server
  - The API server is the front end for the Kubernetes control plane
- The Scheduler
  - Schedules apps (assigns a worker node to each deployable component of your application)
- The Controller Manager
  - Performs cluster-level functions, such as replicating components, keeping track of worker nodes, handling node failures
- etcd
  - a reliable distributed data store that persistently stores the cluster configuration

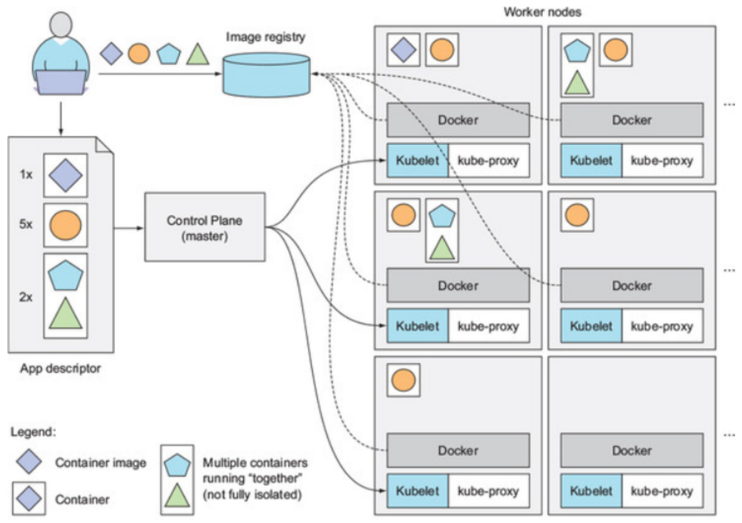
Node components

The worker nodes are the machines that run your containerized applications

- Kubelet
  - An agent that runs on each node in the cluster. It makes sure that containers are running in a pod.
- Kube-proxy
  - Load-balances network traffic between application components
- Container Runtime
  - The container runtime is the software that is responsible for running containers.

Application in Kubernetes

- Package application into one or more container images, push those images to an image registry, and then post a description of your app to the Kubernetes API server.
- When the API server processes your app's description, the Scheduler schedules the specified groups of containers onto the available worker nodes based on computational resources required by each group and the unallocated resources on each node at that moment
- The Kubelet on those nodes then instructs the Container Runtime (Docker, for example) to pull the required container images and run the containers
- Once the application is running, Kubernetes continuously makes sure that the deployed state of the application always matches the description you provided



Keeping the containers running

Running Application In Cluster

Create deployment.yml to deploy to kubernetes

kubectl create -f deployment.yml

Verify running pods

kubectl get pods

Describe Deployment

kubectl describe deployment <<$DEploymentId>>

Create Service

kubectl expose deployment <<$DEploymentId>> --port=80 --name=devops-1-service --type=LoadBalancer —target-port=8080

Display Services

kubectl get svc

Describe Services

kubectl describe svc <<$ServiceId>>

Note: Enter the external-ip of service in browser URL to verify service output

Lab

* Create Kubernetes Cluster in AWS EC2
* Deploy your service in Kubernetes Cluster