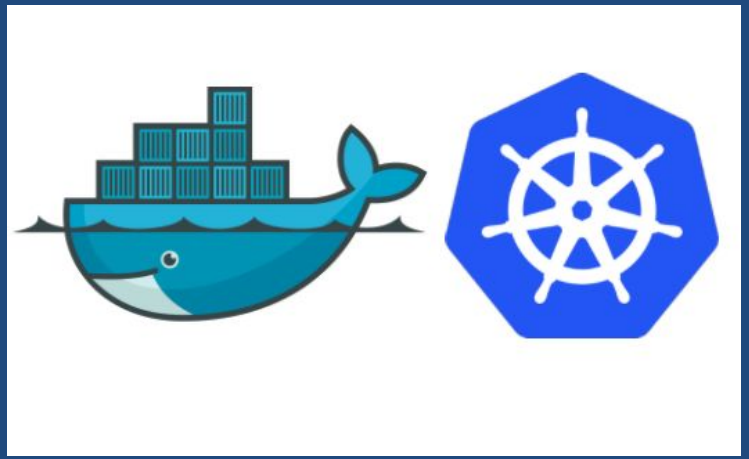
Docker and Kubernetes



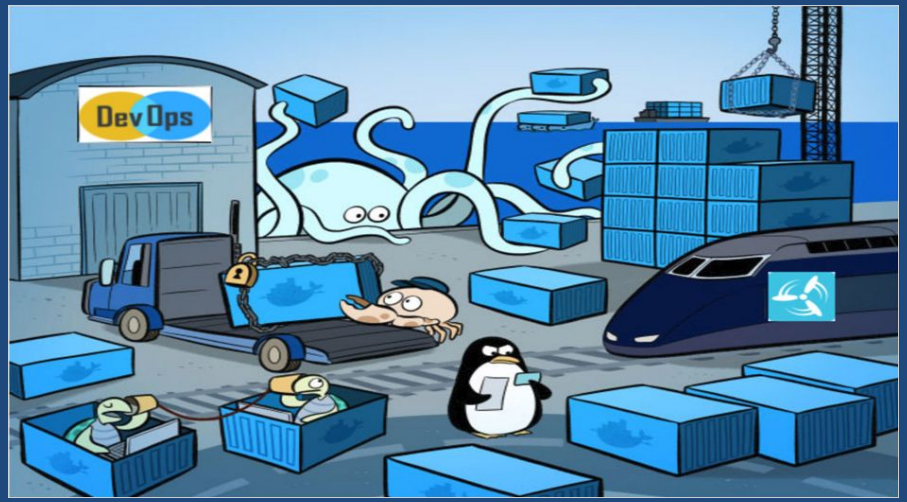
Session 2

May 2018

Rama Krishna Bhupathi

Docker

Docker is a platform for developers and sysadmins to develop, deploy, and run applications with containers. The use of Linux containers to deploy applications is called containerization. Containers are not new, but their use for easily deploying applications is.



Docker

Containerization is increasingly popular because containers are:

- Flexible: Even the most complex applications can be containerized.
- Lightweight: Containers leverage and share the host kernel.
- Interchangeable: You can deploy updates and upgrades on-the-fly.
- Portable: You can build locally, deploy to the cloud, and run anywhere.
- Scalable: You can increase and automatically distribute container replicas.
- Stackable: You can stack services vertically and on-the-fly.

Images and Containers

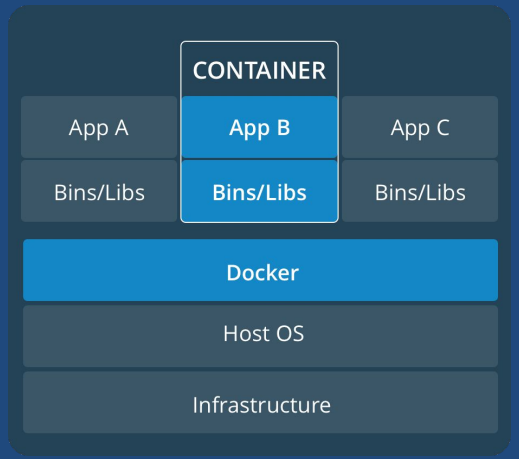
A container is launched by running an image. An image is an executable package that includes everything needed to run an application--the code, a runtime, libraries, environment variables, and configuration files.

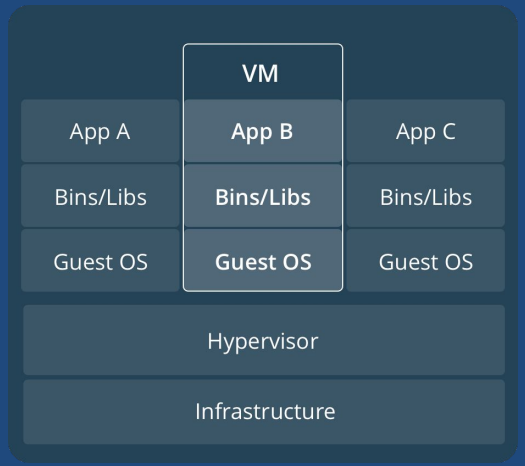
A container is a runtime instance of an image..what the image becomes in memory when executed (that is, an image with state, or a user process). You can see a list of your running containers with the command, docker ps, just as you would in Linux.

Containers and VMs

A **container** runs *natively* on Linux and shares the kernel of the host machine with other containers. It runs a discrete process, taking no more memory than any other executable, making it lightweight.

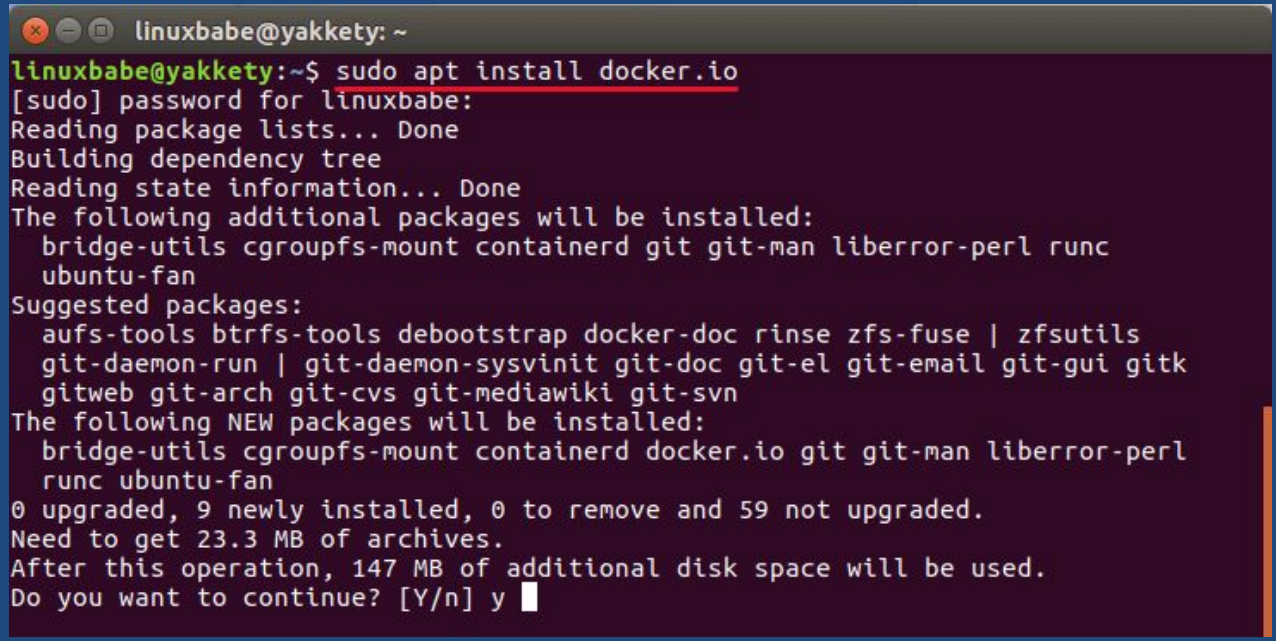
By contrast, a **virtual machine** (VM) runs a full-blown “guest” operating system with *virtual* access to host resources through a hypervisor. In general, VMs provide an environment with more resources than most applications need.





Docker Installation

https://docs.docker.com/install/



Docker works on MacOS , Windows and all Linux flavors

Docker Installation

Try the following…

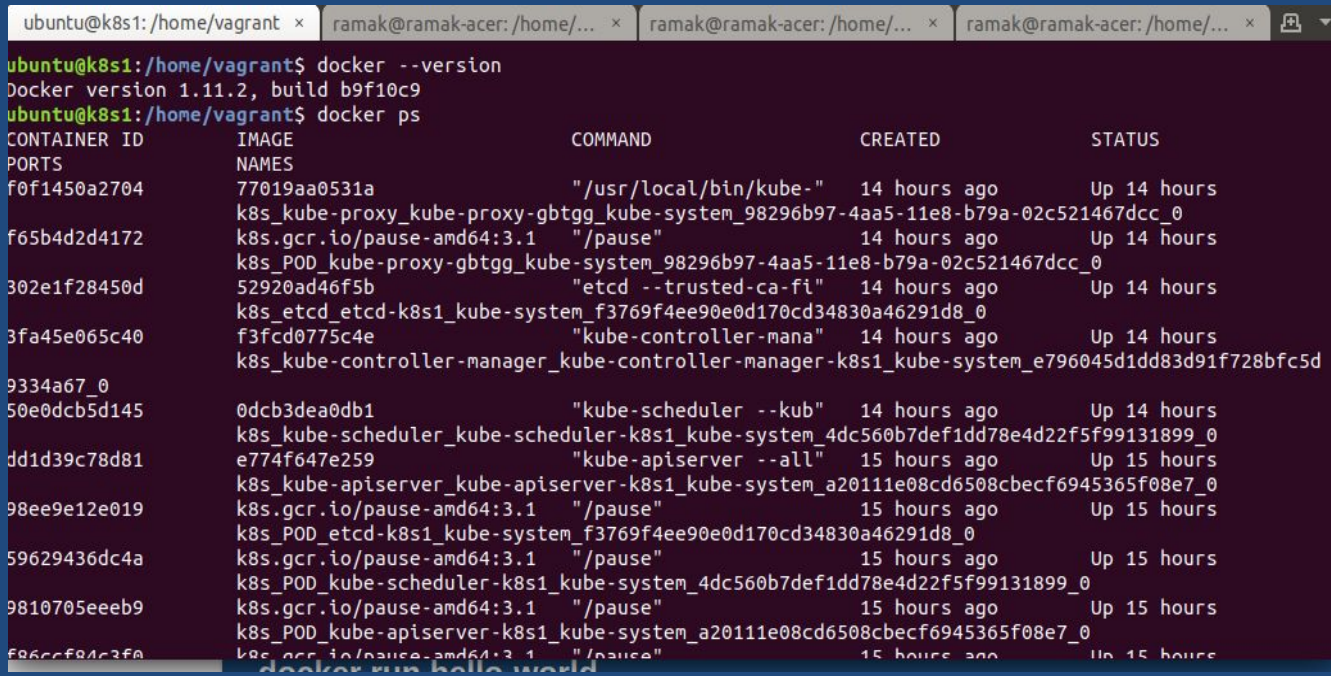
docker --version

docker info

docker ps

docker run hello-world

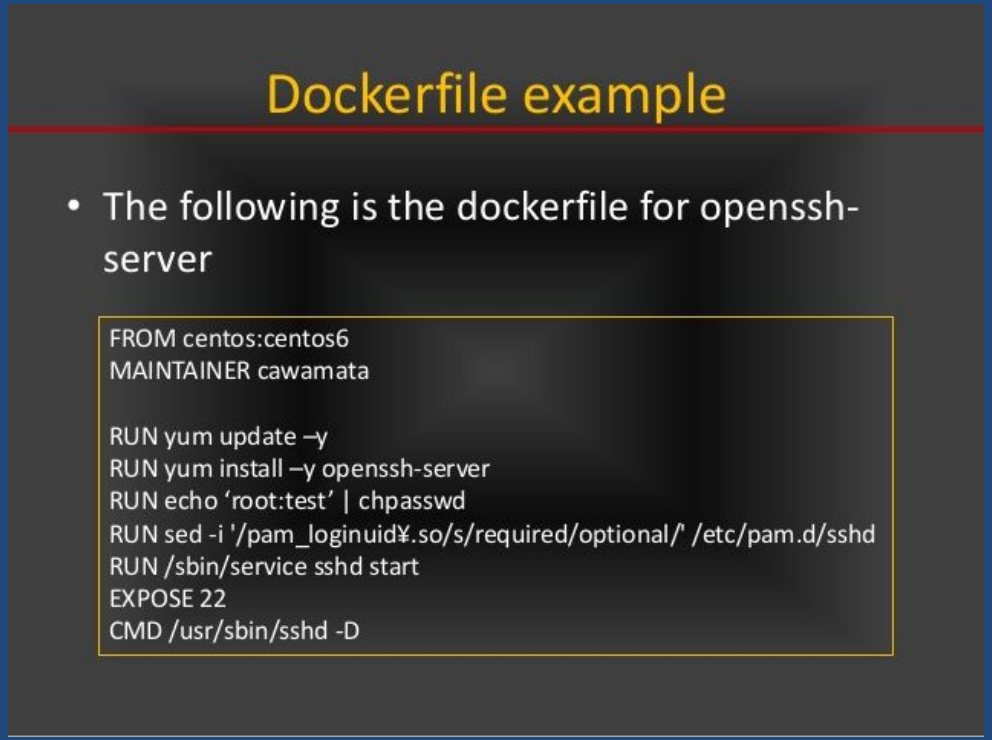
docker images



Dockerfile

https://docs.docker.com/get-started/#test-docker-installation

Dockerfile defines what goes on in the environment inside your container like Packages to install , download , configure and run.



Kubernetes

Kubernetes is a cluster manager for Linux containers. Kubernetes came out of Google . Kubernetes provides the following benefits:

- **Microservices by breaking an application into smaller, manageable, scalable components.**
- **Fault-tolerant cluster in which if a single Pod replica fails (due to node failure, for example), another is started automatically.**
- **Horizontal scaling in which additional or fewer replicas of a Pod could be run by just modifying the “replicas” setting in the Replication Controller**
- **Higher resource utilization and efficiency.**
- **Separation of concerns. The Service development team does not need to interface with the cluster infrastructure team .**

Kubernetes

Kubernetes concepts include Pod, Service, and Replication controller and nodes

What is a node?

A node is a machine (physical or virtual) running Kubernetes onto which Pods may be scheduled. The node could be the master node or one of the worker nodes .

What Is a Cluster?

A cluster is a collection of nodes including other resources such as storage to run Kubernetes applications. A cluster has a single Kubernetes master node and zero or more worker nodes. A highly available cluster consists of multiple masters or master nodes.

Kubernetes

What is a Pod ?

A Pod is a collection of containers that are collocated and form an atomic unit. Multiple applications may be run within a Pod typically different containers are for different applications. A Pod is a higher level abstraction for managing a group of containers with shared volumes and network namespace. All the applications (containers) in a Pod share the same filesystem and IP address with the port on which each application is exposed being different.

What Is a Label?

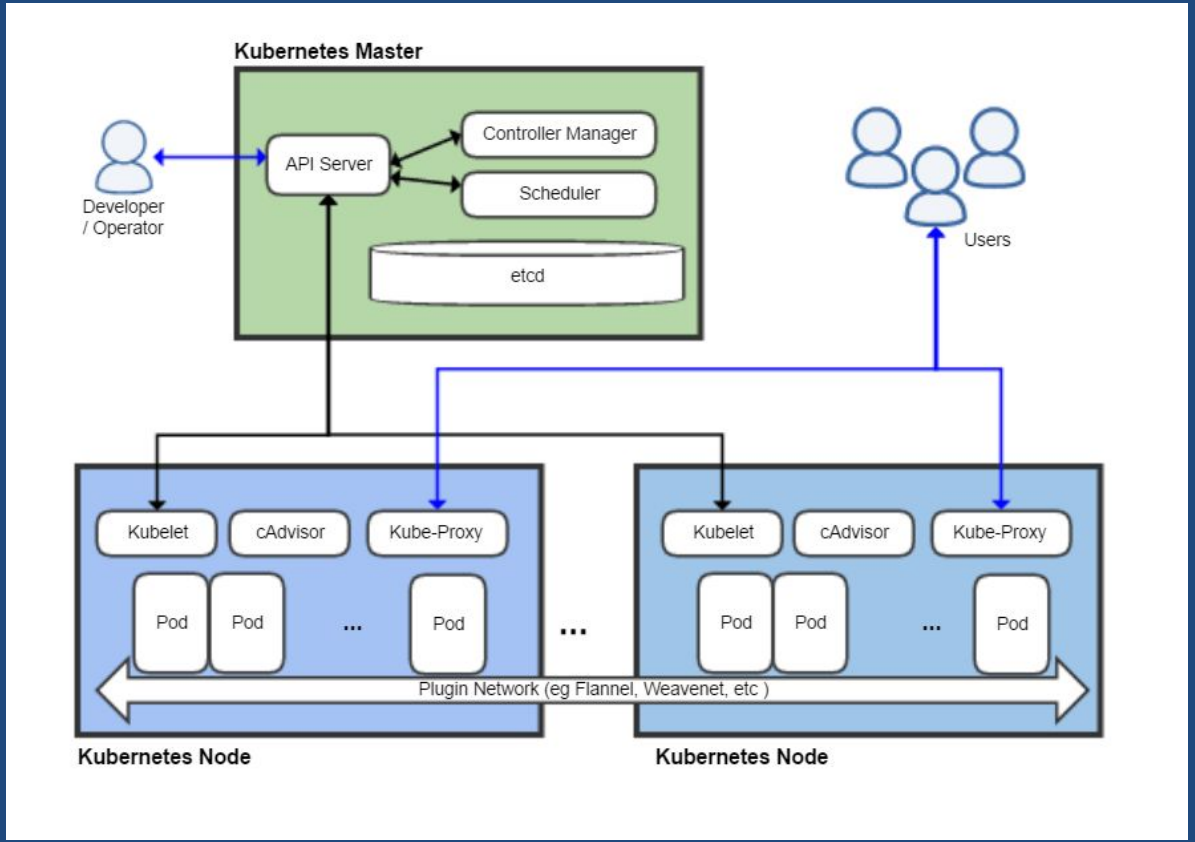
A Label is a key-value pair identifying a resource such as a Pod, Service, or Replication Controller: most commonly a Pod. Labels are used to identify a group or subset of resources for tasks such as assigning them to a Service. Ex: “app = helloApp”

Kubernetes Architecture

Key components

On Master
Kube-API Server
Kube-Controller-Manager
Kube Scheduler
Etcd (deamon)

On Nodes
Kubelet
CAdvisor
Kube-Proxy



Kubernetes Architecture Components

The Kubernetes Master is the main controlling unit of the cluster that manages its workload and directs communication across the system.

etcd : A persistent lightweight distributed key-value data store that reliably stores the configuration data of the cluster, representing the overall state of the cluster.

API Server: The API server is a key component and serves the Kubernetes API using JSON over HTTP, which provides both the internal and external interface to Kubernetes.

Kubernetes Architecture Components

Controller Manager :The controller manager is the process in which the core Kubernetes controllers like DaemonSet Controller and Replication Controller run. The controllers communicate with the API server to create, update and delete the resources they manage (pods, service endpoints, etc.)

Scheduler : The scheduler is the pluggable component that selects which node an unscheduled pod (the basic entity managed by the scheduler) should run on based on resource availability.

Kubernetes Node

The Node, also known as Worker or Minion, is a machine where containers (workloads) are deployed. Every node in the cluster must run a container runtime such as **Docker**, as well as the below-mentioned components, for communication with master for network configuration of these containers.

Kubelet is responsible for the running state of each node, ensuring that all containers on the node are healthy. It takes care of starting, stopping, and maintaining application containers organized into pods as directed by the control plane. Kubelet monitors the state of a pod and if not in the desired state, the pod will be redeployed to the same node. The node status is relayed every few seconds via heartbeat messages to the master. Once the master detects a node failure, the Replication Controller observes this state change and launches pods on other healthy nodes.

Kubernetes Node

A **Container** resides inside a Pod. The container is the lowest level of a micro-service which holds the running application, the libraries and their dependencies. Containers can be exposed to the world through an external IP address.

The **Kube-proxy** is an implementation of a network proxy and a load balancer, and it supports the service abstraction along with other networking operation.[24] It is responsible for routing traffic to the appropriate container based on IP and port number of the incoming request.

cAdvisor is an agent that monitors and gathers resource usage and performance metrics such as CPU, memory, file and network usage of containers on each node.