kubernetes

Nitish Kumar

Software Engineer, Mindfire Solutions

I am working as Java developer and passionate about learning new technology and open source technology.
I believe the best way to learn is learn by doing and in fun way.

Today Agenda

- Introduction to Kubernetes
- Kubernetes Architecture
  - Node
  - Pod
  - etcd
  - Service
  - Scheduler
  - Controller
  - Api Server
  - Kube Proxy
  - kubelet
- kubectl Commands
- The Web UI Dashboard

Kubernetes?

Kubernetes (commonly stylized as **K8s**) is an open-source container orchestration system for automating computer application deployment, scaling, and management. It was originally designed by Google and is now maintained by the Cloud Native Computing Foundation.

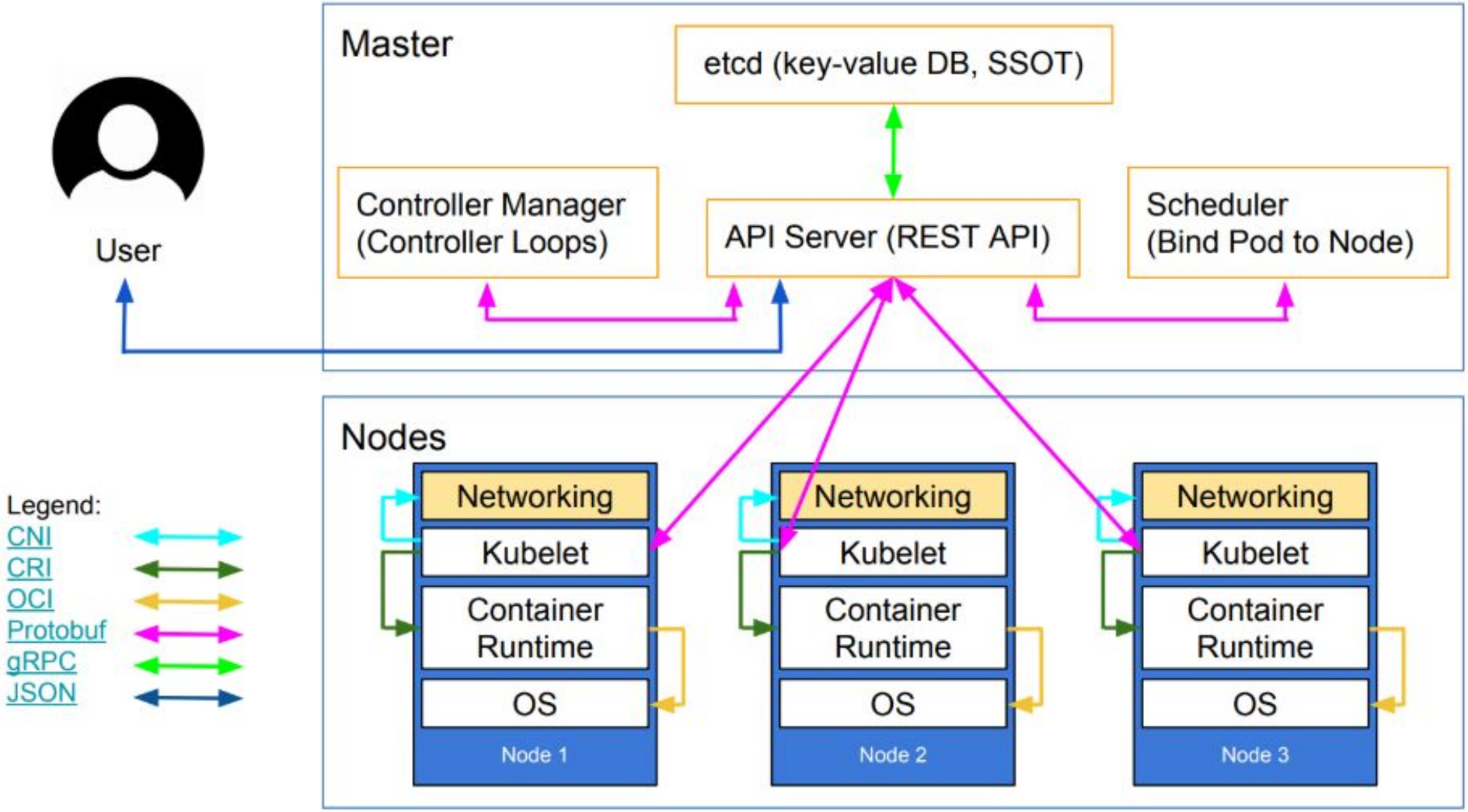
Source: https://en.wikipedia.org/wiki/Kubernetes

K8s History?

- 2003-2004: Birth of the Borg System
- 2013: From Borg to Omega

  Google introduced the Omega cluster management system, a flexible, scalable scheduler for large compute clusters

- 2014: Google Introduces Kubernetes
- 2015: The year of Kube v1.0 & CNCF


Master
etcd (key-value DB, SSOT)
Controller Manager
(Controller Loops)
API Server (REST API)
Scheduler
(Bind Pod to Node)
User
Nodes
Networking
Kubelet
Container
Runtime
OS
Node 1
Networking
Kubelet
Container
Runtime
OS
Node 2
Networking
Kubelet
Container
Runtime
OS
Node 3
Legend:
CNI
CRI
OCI
Protobuf
gRPC
JSON

#Master Node

Master is the control-plane or the brain of k8s cluster. A Master comprises of few components:

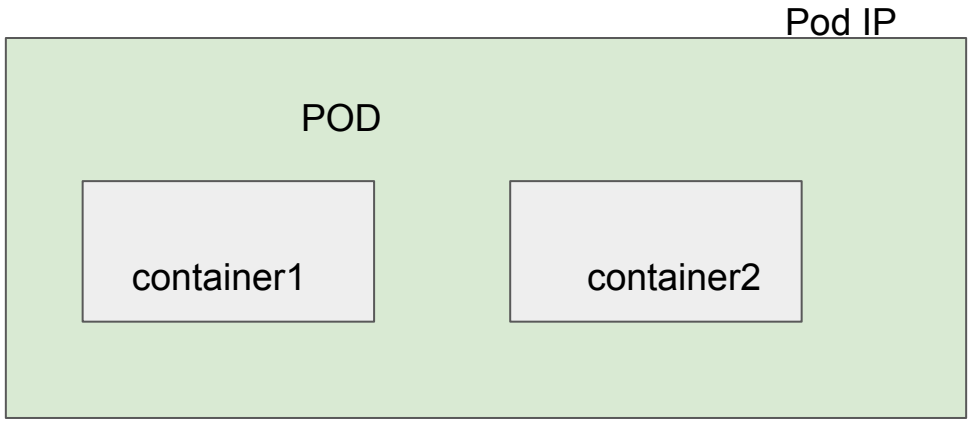
- **api-serve**r - Exposes REST API to talk to k8s cluster, consumes json, only api-server talks to Cluster Store.
- **etcd (KV store)** - Cluster state and config management.
- **Scheduler** - Watches api-server for new pods and assign node to work
- **Controller** - A daemon that watches the state of the cluster to maintain desired state. Example are replication-controller, namespace-controller etc. Other than this it performs garbage collection of pods, nodes, events etc.

#Node

- **Kubelet** - k8s agent which register nodes with cluster, watches api-server, instantiate pods, report back to the api-server. If pod fails, it reports to master and master decides what to do. Exposes port 10255 on node
- **Container Engine** - It does container management like pulling images, starting/stopping containers. Usually Docker is used for container runtime.
- **kube-proxy** - Responsible for networking, Provide unique IP to Pods, All container in a pod share same IP, Load balances across all pods in a **service**

#Pods

- An environment to run containers
- It have network stack, kernel namespaces and one or more container running
- Container always runs inside a pod
- Pod can have multiple containers
- It is unit of scaling in k8s



#Services

Pods comes and go with different IPs. To distribute load and act as a single source of interaction to all pods of an application, **service** play the role.

- Has single IP and DNS
- Created with a manifest JSON file
- All new pods gets added/registered to the service
- Which pod should be assigned to which services is decided by **labels**
- **service** and **pods** have **labels** on the basis of which service identifies its **pods**
- only sends traffic to healthy pods
- uses tcp by default (udp is also supported)

#Deployments

It is a k8s object whose task is to manage identical pods running and upgrading them in controlled way.

- Deployed using YAML/JSON manifest
- Deployed via api-server
- Provide update of pods
- Provide rollbacks

```
apiVersion: apis/v1
Kind: Deployment
metadata:
    name: xyz
spec:
   replicas: 4
```

#Overall Flow

- kubectl writes to the API Server
- API Server validates the request and persists it to Cluster store(etcd)
- Cluster store (etcd) notifies back the API Server
- API Server invokes the Scheduler
- Scheduler decides where to run the pod on and return that to the API Server
- API Server persists it to etcd
- etcd notifies back the API Server.
- API Server invokes the Kubelet in the corresponding node
- Kubelet talks to the Docker daemon using the API over the Docker socket to create the container
- Kubelet updates the pod status to the API Server
- API Server persists the new state in etcd

Kubectl?

Kubectl is a **command line interface for running commands against Kubernetes clusters.**

Kubectl <action> <resource> <resource name> <flags>

```
kubectl config get-contexts                          # display list of contexts
kubectl config current-context                       # display the current-context
kubectl config use-context my-cluster-name           # set the default context to my-cluster-name
kubectl create deployment nginx --image=nginx        # start a single instance of nginx
kubectl get services                                 # List all services in the namespace
kubectl get pods --all-namespaces                    # List all pods in all namespaces
kubectl get pods -o wide                             # List all pods in the current namespace, with more details
kubectl get deployment my-dep                        # List a particular deployment
kubectl get pods                                     # List all pods in the namespace
kubectl get pod my-pod -o yaml                       # Get a pod's YAML
Kubectl get nodes                                    # List all nodes
kubectl describe nodes my-node
kubectl describe pods my-pod
Kubectl cluster-info
kubectl port-forward my-pod 5000:6000
Kubectl api-resources
kubectl set image deployment/frontend www=image:v2
kubectl rollout undo deployment/frontend
kubectl edit svc/docker-registry                     # Edit the service named docker-registry
kubectl scale --replicas=3 -f foo.yaml
kubectl delete -f ./pod.json                         # Delete a pod using the type and name specified in pod.json
```

K8s Dashboard:

- To create k8s dashboard we need to create some resource, run the below command to create the resource.

  kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-beta8/aio/deploy/recommended.yaml
- To access the dashboard need service account with full admin role to make any changes from dashboard. Copy the below content in yaml file and create with kubelet.

Command to get token:

kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep eks-admin | awk '{print $1}')

URL to access dashboard:

http://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/#/login

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eks-admin
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRoleBinding
metadata:
  name: eks-admin
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: eks-admin
  namespace: kube-system
```


THANK YOU