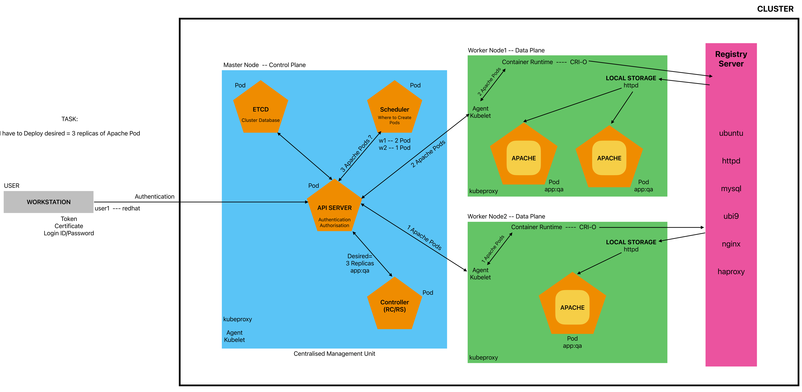
Kubernetes Architecture



OpenShift

It is 75% all about Kubernetes, it adds some functionality on top of Kubernetes

Developer --- has a index.php file, and wants to deploy it on Kubernetes

Using Kubernetes

1. Need to create a Containerfile.
2. Creating an image from that Containerfile.
3. Then, upload that image to any Registry Server.
4. Needs to create a Deployment ---- RC/RS ---- Pod Creation.
5. Setup Routing.
6. Enjoy the Application.

CaaS
(Container as a Service)

Using OpenShift

1. Creating a Containerfile.
2. We can handover this Containerfile to OpenShift.
3. OpenShift will automatically create an Container Image.
4. OpenShift will automatically push it to a Registry Server.
5. OpenShift will automatically Create the Deployment ---- RC/RS ---- Pods.
6. OpenShift will automatically setup Routing.
7. Enjoy the Application.

PaaS
(Platform as a Service)

1. Post that complete website data onto GitHub Repository.
2. Handover the GitHub Repository link to OpenShift.
3. OpenShift will read and understand and analyze (index.php)
4. OpenShift will implement a builder image of PHP.
5. Using the Builder Image OpenShift will create a Containerfile.
6. OpenShift will automatically create an Container Image.
7. OpenShift will automatically push it to a Registry Server.
8. OpenShift will automatically Create the Deployment ---- RC/RS ---- Pods.
9. OpenShift will automatically setup Routing.
10. Enjoy the Application.

s2i
(Source to Image)

KUBERNETES	OPENSHIFT	
CaaS (Container as a Service)	PaaS (Platform as a Service)	Also has s2i (Source to Image)
OpenSource	Enterprise	
Installed and Implemented Ex. Cent05, RHEL, Ubuntu	Master Node --- rhelcoreOS Worker Nodes --- rhelcoreOS or RHEL	
No Console Preconfigured No Monitoring Tool Preconfigured No Logging Tool Preconfigured No Ingress Controller Preconfigured No Registry Server Preconfigured	Console Preconfigured Monitoring Tool Preconfigured Logging Tool Preconfigured Ingress Controller Preconfigured Registry Server Preconfigured	WebGUI Prometheus, Grafana ELK Stack Router Pods are implemented Internal Private Registry Server
SELinux stays in Permissive Mode	SELinux stays in Enforcing Mode	

RHOCP (Red Hat OpenShift Container Platform)

Install oc (OpenShift Cluster) CLI

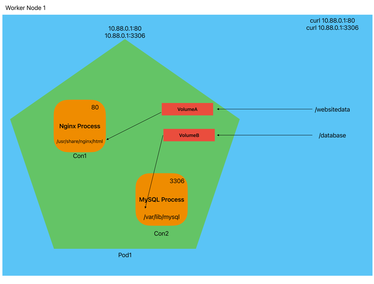
```
dnf install wget -y
wget https://mirror.openshift.com/pub/openshift-v4/x86_64/clients/ocp/stable/openshift-client-linux.tar.gz
ls
tar -xvzf openshift-client-linux.tar.gz
ls
mv oc /usr/bin
mv kubectl /usr/bin
oc version
```

oc Commands

-> oc project <name_of_project>
-> oc new-project <name_of_project>
-> oc get all
-> oc get pods
-> oc get deployment -n openshift-etcd
-> oc describe pod/<name_of_pod>
-> oc describe pod <name_of_pod>
-> oc whoami --show-console

POD

Group of one or more containers that share the same network and storage namespace



Methods to Create Custom Pods

Method 1
-> oc run <name_of_pod> --image <image_to_use>

Method 2
-> oc run <name_of_pod> --image <image_to_use> --dry-run=client -o yaml > new-pod.yml
-> oc create -f new-pod.yml

Method 3
-> Using Web Console

DEPLOYMENT

Number of Pods + It will create the RS (Replica Set)

Methods to Create Deployments

Method 1
-> oc create deployment <name_of_deploy> --image <image_to_use> --replicas <no_of_replicas>

Method 2
-> oc create deployment <name_of_deploy> --image <image_to_use> --replicas <no_of_replicas> --dry-run=client -o yaml > new-deploy.yml
-> oc create -f new-deploy.yml

Method 3
-> Using Web Console

Expose your Deployment ---> Service
Expose your Service ---> Route