PAGE NO. 1 Mar 24
DATE : / /

INDUSTRY USE CASES - KUBERNETES/ OPENSHIFT

- Containers
  - less libs / binaries
    - ∴ less vulnerabilities
      - easier to secure
  - no guest OS
  - launches in very less time

- Single host env. If it goes down we don't have anything to fall back.
- We need a management tool for containers which simple run-time don't provides.
- Need an agent for life cycle, health check and monitoring.

Need more than Containers
- Authentication
- Scaling
- Networking
- Image Registry
- Metrics & Logging

All the requirements were not satisfied only by the containers.
∴ We shifted to Openshift or Kubernetes
- PAAS
  - Platform As A Service

PAGE NO. :
DATE : / /

Features of Openshift
- Pod autoscaling
- High Availability
- Choice of cloud infrastructure
- Responsive web console
- Rich cli tool set
- IDE integration
- Open source
- Operator hub
- CI/CD
- Service Mesh
- Serverless (OCP 4.3)
- Application Topology
- Quay 3.2
- Over The Air update

podman → runtimes (cri-o)
- By podman we can also use docker cmd.
- Create containers without root privileges.
- allows health checks.
-

PAGE NO. :
DATE : 9/11/21

SESSION : 111 Openshift Os

- openshift is an unified tool (management tool) for managing the entire life cycle of one K8s.
  - ↳ mostly automated.

SESSION: 112 DO180 OS

PAGE NO.:
DATE: 10/May/24

docker commit myweb nimal1s/reqs-flask:v1

upstream → Fedora (linux dist) → downstream → RHEL

- RHOCP (enterprise product) (paid)
  ↖ (downstream project)
- OKD → community
  (like fedora)
  (upstream project)

Install OS — Single Node Cluster

- minishift
  - installed on vbox.
- CRC (v4.5, ...)
  - Code Ready Container
  - doesn't support Vbox.

win
minishift start -h
——— --network-nameserver 8.8.8.8 (google dns server)
- starts with DNS server.

- build configuration: download the image and deploy the pod automatically.

PAGE NO. :
DATE : / /

minishift console
↳ opens the webUI of openshift

Github
Create a new repo
↳ okd-training-repo
↳ add readme file
> Create repo
↳ Create a file
index.php
<? php
print " welcome to ocp first ";
?>

- In OS instead of NodePort we have route.

DRIVE openshift install

MUST IMP
minishift config set vm-driver virtualbox

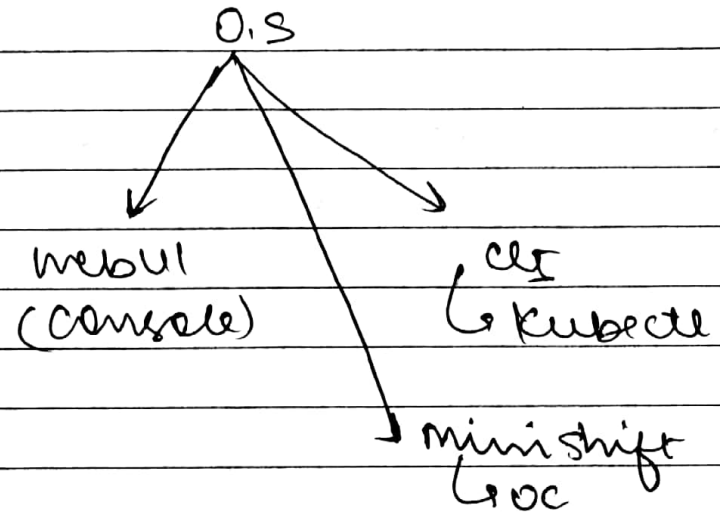
minishift start --vm-driver virtualbox --network-nameserver 8.8.8.8

minishift

PAGE NO.:
DATE: 17/May/21

SESSION: 115 OS

- Openshift works on the top of K8S. With the use of OS we also doesn't need any CI/CD program like Jenkins.



win # minishift oc-env

SET PATH=C:\Users\ ——— ;%PATH%
↳ Export/set the env variable in cli
↳ To make it permanent we have to put in env var

minishift console --url
↳ URL

oc login <URL> -u system:admin

oc new-project p3
↳ creates a new project : p3

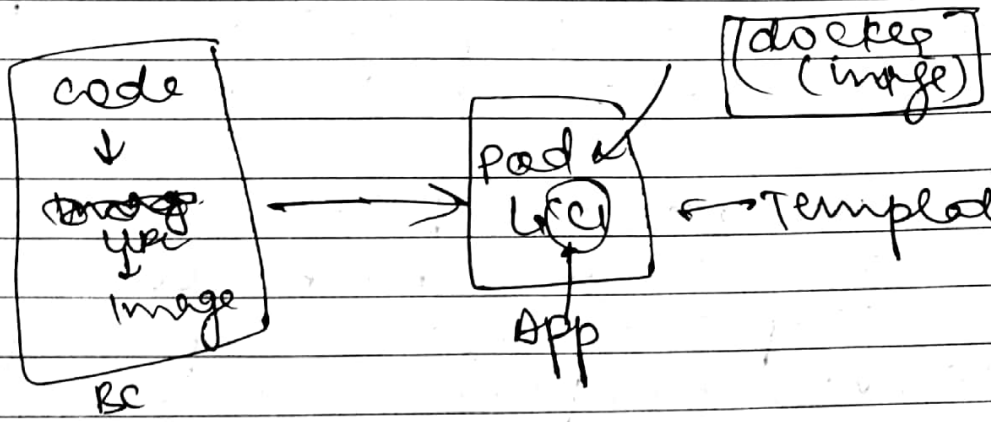
oc ~~get~~ projects
↳ p3 ← namespace

PAGE NO. :
DATE : / /

oc get pods

oc get bc → build configuration

↳ new feature in openshift



oc new-app <<github url>>

↑ They automatically figure the code & launches the image accordingly. In this case, php.

oc get bc

↳ git-workshop
(automatically created)

- dc == deployment configuration

oc logs -f bc/git-workshop

oc get svc

oc expose svc/git-workshop

↳ exposes the svc

↳ outside connectivity is done.

oc get route

↳ automatically created & gives the url

PAGE NO.
DATE / /

- It automatically create its own private domain names.

oc start-build gitworkshop
↳ new code is build. now we can see the new code.

- To automate start-build we can set a trigger in github. As soon as dev uploads a new code, openshift automatically updates the code.

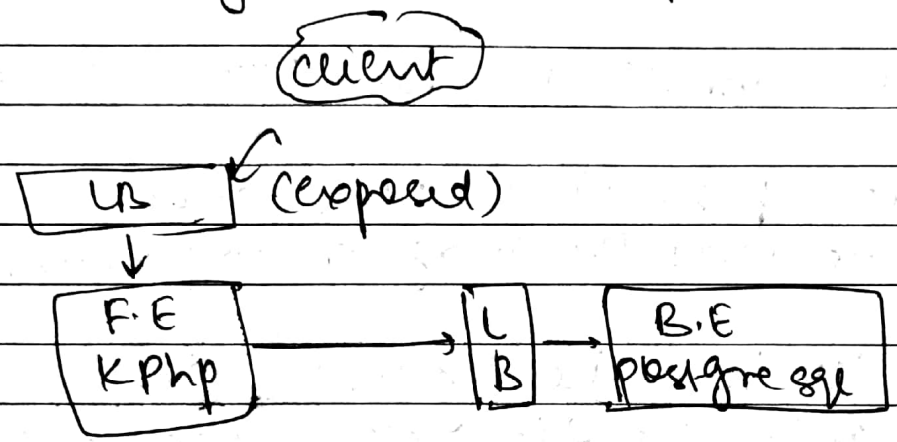
PAGE NO.:
DATE 18 Mar 21

SESSION: 116 OS

minishift start --vm-driver virtualbox
--network-nameserver 8.8.8.8

YAML → Code → Template

- Templates are the pre-created images used by OS to run a pod.



~~start~~ redhat os lab

oc login <API endpoint> -u <user>
-p <pass>

oc new-project lwp3

minishift status
minishift console
oc login <URL of console>
-u system:admin
oc get projects
↳ openshift
oc get template -n openshift
↳ lots of templates present by default

PAGE NO. :
DATE : / /

oc new-app cakephp-mysql-persistent

↳ 2 svc/pods started
↳ sql & php

oc get route
↳ URL is available for the app
↳ dc is not done.
TO run we first need to build it.

SESSION: 123 — 23 DSA

PAGE NO.:
DATE: 24/Mar/21

- DSA ∝ How you organise data
  - ↓ Time (CPU)
  - ↓ space (memory)

Openshift

- K8S is good in orchestration.

- MySQL
  - image
    - program → POD
      - var
        - env-var

login to ocp (web UI)
- New project (mydbproj)
  - Topology
    - ~~Database : Mysql~~
      - container image
        - mysql
      - mysql-app (app name)
      - mysqldb (name)
      - Deployment Config
      - Add route

→ Create

↳ It failed !!!

PAGE NO. :
DATE : / /

- In K8S

  deployment → RC
  deployment → RS

- In OS (openshift)

  deployment ⟶ RS

  deployment config (DC) ⟶ RC

- Pause creates the svc with NodePort, exposing the pod. It also provides much more services. ~~Etc~~,

~~#oc create~~

Project : mydbproj

↳ YAML

~~spec:~~

~~containers:~~

↳ Environment

↳ Name : MYSQL_DATABASE : mydb

↳ MYSQL_USER : vimal

↳ MYSQL_PASSWORD : redhat

↳ MYSQL_ROOT_PASSWORD : redhat

SESSION:131 OS

- Openshift
  - └ K8S
    - └ CRI-O
      - └ Runc (← Podman)
        - └ Container
          - └ Pod

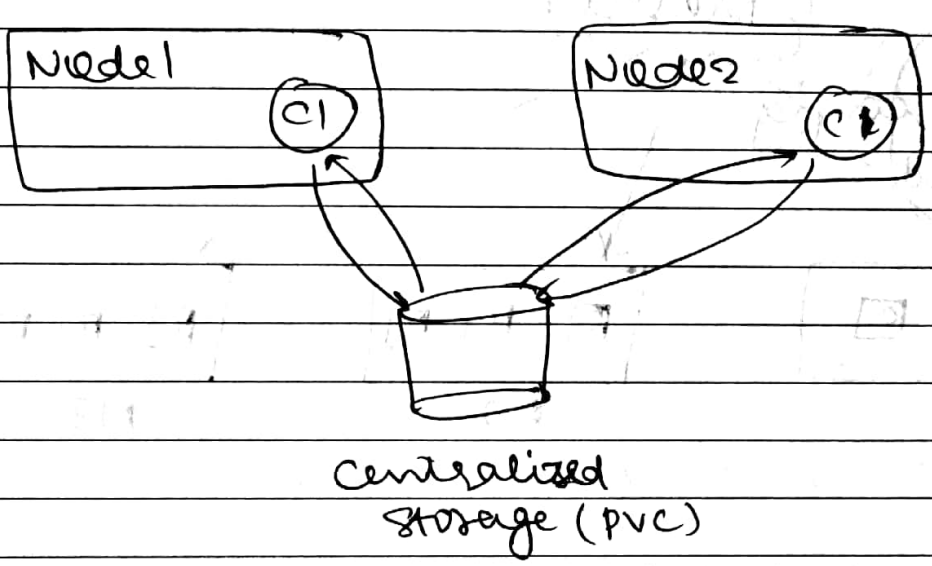
- command lines of docker & podman are nearly similar.

```
# yum install podman -y

# podman version
# podman pull vimal13/apache-webserver-php
# podman images

# podman run -itd -p 1234:80 --name os1 vimal13/apache-webserver-php

# podman exec os1 ifconfig
# podman exec -it os1 bash

# podman pull mysql

# podman run -it --name db1
  -e MYSQL_ROOT_PASSWORD=redhat
  -e MYSQL_USER=vimal -e MYSQL_DATABASE=mydb
  -e MYSQL_______ . mysql
  -v /mydata:/var/lib/mysql
```

PAGE NO. :
DATE : / /

mysql -h 127.0.0.1 -u vimal -predhat

K8S | OS

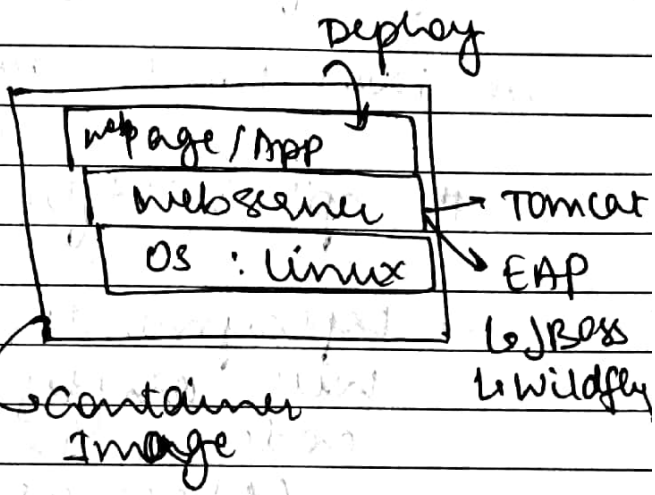


SESSION: 136 OS

PAGE NO.:
DATE: 7/8/2021

- Podman == DOCKER
- Custom Images

- ~~EAP~~



- Image: webapp
  - I : Linux
  - II : jdk
  - III : jboss server. → package url available on ~~github~~ drive
  - IV : deploy war package

wget <url jboss server>
tar -xzfv <file name> (xvzf)
/~~jboss/build~~

~~$~~ export JBOSS_HOME= /jboss/wildfly-23.0.0.Fin
/wildfly .
cd jboss/bin

./standalone.sh
↳ service starts
↳ service only available on local host

./standalone.sh -b 0.0.0.0
↳ service is also avail for everyone

PAGE NO. :
DATE : / /

- By default, jboss uses port 8080.
- Download sample code from gh.
  - war file
  - prog.

cp ^.war wildfly — /standalone/deployments

- Deployments is the plugin that will ~~deploy~~ publish the new code in real time.
  (/jboss-as-helloworld)
  ↳ route for hello world program.

~~mkdir~~ vim Dockerfile
FROM jboss/base-jdk:11
ADD ~~COPY~~ → wildfly-23.0.0.~~Final~~.tar.gz / ← in / dir
ENV JBOSS_HOME /wildfly-23.0.0.Final
COPY → USER jboss
CMD ["$JBOSS_HOME/bin/standalone.sh","-b","0.0.0.0"]

(↳ next page)
# vim /etc/containers/registries.conf
registries = ['docker.io']
↳ To change the $ pull order in podman.

- Diff bet ADD & COPY is copy only copies but add also extracts the file.
- ENV → creates environment variable

podman build

podman run -dit 1234:8080 name:v .

COPY jboss_as-helloworld.war /wildfly_/ standalone/ deployments

- This image will not work in openshift we need to add
  EXPOSE 8080
  to expose the container ~~within~~ on 8080 port.

SESSION : 137 08

PAGE NO. :
DATE 8 Apr 21

Dockerfile

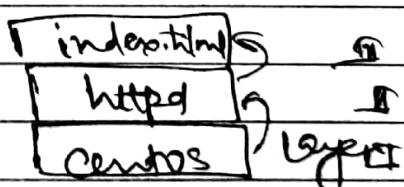
```
FROM centos:latest
MAINTAINER nimal daga <ndaga@lwindia.com>
LABEL env=dev
LABEL team=team1
LABEL description="this is a desc"
```

podman build -t myimage:v1 .
podman run -it myimage:v1
pwd
/
C by default our working dir is /.

Dockerfile

RUN → executes at build-time

```
RUN yum install httpd -y
RUN echo hello > /var/www/html/index.html
```




```
WORKDIR /var/www/html
```

L> changes the working directory or default directory

```
RUN echo lw > lw.html
```

L> This file will be created in the working dir. (/var/www/html)

PAGE NO. :
DATE : / /

- Whenever we use any cmd in dockerfile, they store the o/p of cmd in some memory, and the followed cmd. Thus, creating a linked list data structure.
- Each layer is similar to a node or a link.
- If you have more and more layer it will somehow impact the performance.

- Instead of running
  RUN date
  RUN cal
  we can use
  RUN date && cal

Dockerfile
RUN yum install httpd -y && echo hello > index.html
&& echo lw > lw.html

↳ This will add only one layer/node.

OR

RUN yum install httpd -y && \
echo hello > index.html && \
echo lw > lw.html

PAGE NO. :
DATE : / /

- CMD ~~&~~ ~~ENTRYPOINT~~
  - both are run-time command.
    - cmd are run when we run a container.
  - we create containers per cmd basis. And container is a process. As soon as the cmd is completed the container stops

~~#Dockerfile~~

```
# Dockerfile
FROM centos:latest
RUN cal
CMD ["date"]
```

- ENTRYPOINT will also do the same thing as CMD but the difference is entrypt also gives the ~~facility~~ facility of arguments.
  These entrypt acts as main cmd & CMD as argument.

```
ENTRYPOINT ["date"]
CMD        ["-u"]
```

Ex:

```
ENTRYPOINT ["httpd"]
CMD        ["-D", "FOREGROUND"]
```