**KUBERNETES - TECHNICAL GUFTGU**

--------------------------------------------------------------------------------------------------------------------------------------

**Installation on master and worker node on a single ec2 instance using kubectl, minikube and conntrack**

# Switch to root user

sudo su

# Install docker

sudo apt update && apt -y install docker.io

# Download and install Kubectl

curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl"

chmod +x ./kubectl && sudo mv ./kubectl /usr/local/bin/kubectl

# Download and install Minikube

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube

# Install

sudo apt-get install -y conntrack

# Start Minikube

minikube start --vm-driver=none

or

minikube start --driver=none

# Check status and version

minikube status

minikube version

kubectl version

# Create .yml file to create a POD

vi pod1.yml

#Pod1.yml file code - copy and paste

**kind: Pod**

**apiVersion: v1**

**metadata:**

**name: testpod**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priyanka; sleep 5 ; done"]**

**- name: c01**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priya; sleep 2 ; done"]**

**restartPolicy: Never # Defaults to Always**

# Save it

Esc

:wq!

# apply the yml file to create the POD

kubectl apply -f pod1.yml

# Get the list of active POD running on the machine

kubectl get pods

# Get the complete information of a particular POD, here it is “testpod”

kubectl describe pod testpod

or

kubectl describe pod/testpod

# To get echo statement “Hello-priyanka” of a container named “c00”

kubectl logs -f testpod -c c00

# Press Ctrl+c to exit

# Delete the POD named testpod

kubectl delete pod/testpod

# Delete the POD named testpod using its .yml file

kubectl delete -f pod1.yml

# Delete node in case created an ec2 instance from My AMI image and start with a new ip

kubectl delete node node-ip

minikube stop

minikube start

# To check the running status of minikube

minikube status

**MULTI CONTAINER POD ENVIRONMENT**

# Create the POD using .yml file “pod2.yaml”

vi pod2.yaml

# Pod2.yaml code – copy and paste

**kind: Pod**

**apiVersion: v1**

**metadata:**

**name: testpod1**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo hello priyanka; sleep 5 ; done"]**

**- name: c01**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-world; sleep 5 ; done"]**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod2.yaml

# Get inside the container “c01” running on the POD “testpod1”

kubectl exec testpod 1 -it -c c01 -- /bin/bash

# check process status with ps command

ps

ps –ef

# Exit the container

Exit

# Get the host machine of POD ip of the container “c01”

kubectl exec testpod -c c01 -- hostname -i

**POD ENVIRONMENT VARIABLES**

# Create the POD using .yml file “pod3.yaml”

vi pod3.yaml

# Pod3.yaml code – copy and paste

**kind: Pod**

**apiVersion: v1**

**metadata:**

**name: environmentstestpod**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priya; sleep 5 ; done"]**

**env: # List of environment variables to be used inside the pod**

**- name: MYNAME**

**value: PRIYANKA**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod3.yml

# Get inside the container “c01” running on the POD, “environmentstestpod”

kubectl exec environmentstestpod -it -c c01 -- /bin/bash

# List the environment variable with env command inside the container “”c01

Env

# Exit the container

exit

# Delete the POD “environmentstestpod”

kubectl delete pod/environmentstestpod

**POD WITH PORTS**

# Create the POD using .yml file “pod4.yaml”

vi pod4.yaml

# Pod4.yaml code – copy and paste

**kind: Pod**

**apiVersion: v1**

**metadata:**

**name: testpod4**

**spec:**

**containers:**

**- name: c00**

**image: httpd**

**ports:**

**- containerPort: 80**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod4.yml

# Get the IP of the POD

kubectl get pods -o wide

# Run httpd webserver at port 80

curl 172.17.0.5:80

**EXAMPLE OF LABELS**

# Create the POD using .yml file “pod5.yaml”

vi pod5.yaml

# Pod5.yaml code – copy and paste

**kind: Pod**

**apiVersion: v1**

**metadata:**

**name: delhipod**

**labels:**

**env: development**

**class: pods**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priya; sleep 5 ; done"]**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod5.yaml

# Show labels

kubectl get pods --show-labels

# Command to label “myname=priyanka” an existing POD “delhipod”

kubectl label pods delhipod myname=Priyanka

# Check the created label

kubectl get pods --show-labels

# List all the PODs with the label “env=development”

kubectl get pods -l env=development

# List all the PODs without the label “env=development”

kubectl get pods -l env!=development

# List all the PODs with any label mentioned in the set env

kubectl get pods -l 'env in(development, testing)'

# List all the PODs with any label not included in the set env

kubectl get pods -l 'env notin(development, testing)'

# Delete the Pod with label “env=development”

kubectl delete pods -l env = development

**NODE SELECTOR EXAMPLE**

# Create the POD using .yml file “pod6.yaml”

vi pod6.yaml

# Pod6.yaml code – copy and paste

**kind: Pod**

**apiVersion: v1**

**metadata:**

**name: nodelabels**

**labels:**

**env: development**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priya; sleep 5 ; done"]**

**nodeSelector:**

**hardware: t2-medium**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod6.yaml

# Get the list of running PODs

kubectl get pods

# Describe the POD “nodelabels”

kubectl describe pods/nodelabels

# Command to label “myname=priyanka” an existing node “ip-172-31-16-231”

kubectl label nodes ip-172-31-16-231 hardware=t2-medium

# Describe the POD “nodelabels”, to view the selector

kubectl describe pods/nodelabels

**EXAMPLE OF REPLICATION CONTROLLER (Works only on equality based selector)**

# Create the POD using .yml file “pod7.yaml”

vi pod7.yaml

# Pod7.yaml code – copy and paste

**kind: ReplicationController**

**apiVersion: v1**

**metadata:**

**name: myreplica**

**spec:**

**replicas: 2**

**selector:**

**myname: priya singh**

**template:**

**metadata:**

**name: testpod6**

**labels:**

**myname: priya**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priya; sleep 5 ; done"]**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod7.yaml

# Get the list of running PODs

kubectl get pods

# Get the list of running replication controller( rc )

kubectl get rc

# Describe the replication controller “myreplica”

kubectl describe rc myreplica

# Get the list of all the POD replicas

kubectl get pods

# Delete a POD by its replica name “myreplica-gqlml”

kubectl delete pod/myreplica-gqlml

# verify the deleted POD

kubectl get pods

# Verify the replica count and status

kubectl get rc myreplica

# Describe the “myreplica” to view all the deleted and the newly created PODs

kubectl describe rc myreplica

# Check the labels

kubectl get pods --show-labels

# Scale- up the replica count to “8” from “2” mentioned in the yml file and select using label “myname=priya”

kubectl scale --replicas=8 rc -l myname=priya

# Verify the list of pod replicas

kubectl get pods

# Verify the replica count and status

kubectl get rc

# Verify the labels

kubectl get pods --show-labels

**EXAMPLE OF REPLICA SET (Works on equality and Set based selector)**

# Create the POD using .yml file “pod8.yaml”

vi pod8.yaml

# Pod8.yaml code – copy and paste

**kind: ReplicaSet**

**apiVersion: apps/v1**

**metadata:**

**name: myrs**

**spec:**

**replicas: 2**

**selector:**

**matchExpressions: # these must match the labels**

**- {key: myname, operator: In, values: [riya, priya, priyanka]}**

**- {key: env, operator: NotIn, values: [production]}**

**template:**

**metadata:**

**name: testpod7**

**labels:**

**myname: priya**

**spec:**

**containers:**

**- name: c00**

**image: ubuntu**

**command: ["/bin/bash", "-c", "while true; do echo Hello-priya; sleep 5 ; done"]**

# Save the file

:wq!

# apply the yml file to create the POD

kubectl apply -f pod8.yaml

# Get the created of running replication set( rs )

kubectl get rs

# Get the list of running PODs

kubectl get pods

# Scale- down the replica count to “1” from “2” mentioned in the yml file

kubectl scale --replicas=1 rs/myrs

# Verify the PODs

kubectl get pods

# Verify the replica count

kubectl get rs

# Delete a POD, “myrs-bt8rz”

kubectl delete pod/myrs-bt8rz

# verify the replica count again

kubectl get rs

# verify the Running PODs

kubectl get pods

# Delete the POD by deleting the yml file

kubectl delete -f pod8.yml

# Verify the status of the deleted PODs

kubectl get rs

kubectl get pods