# **Pods:**

## Create pod with YAML & command line :

$ kubectl run nginx --image=nginx

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod  labels:  app: nginx  spec:  containers:  - name : nginx  image: nginx |

## Create Pod with environment variables :

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-with-env  labels:  app: nginx  spec:  containers:  - name : nginx  image: nginx  env:  - name: MY\_ENV\_VAR  value: "hello There"  - name : MY\_SECOND\_VAR  value: "send Env Var" |

## Create pod with command:

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-with-command  labels:  app: nginx  spec:  containers:  - name : nginx  image: nginx  command: ["echo" , "'Hi from container'"]  restartPolicy: OnFailure |

## Create Pod with initContainer :

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-initcontainer  labels:  app: nginx  spec:  initContainers:  - name: busybox  image: busybox  command: ['sh', '-c', 'echo The app is running! && sleep 600']  containers:  - name : nginx  image: nginx |

## Create static Pod:

On the node add this file to /etc/kubernetes/manifests:

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: static-pod-kube01  spec:  containers:  - name : nginx  image: nginx |

## Configure Memory Resources to Containers and Pods:

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-memory-limits  labels:  app: memory-tester  spec:  containers:  - name : mem-tester  image: busybox:1.29  command: ["nslookup"]  args: ["google.com"]  resources:  limits:  memory: "200Mi"  requests:  memory: "100Mi" |

## CPU Resources to Containers and Pods :

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-cpu-limits  labels:  app: cpu-tester  spec:  containers:  - name : busybox  image: busybox  resources:  limits:  cpu: "1"  requests:  cpu: "0.5" |

## Configure a Security Context for a Pod or Container

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-security-context  spec:  securityContext:  runAsUser: 1000  fsGroup : 2000  volumes:  - name: vol1  emptyDir: {}  containers:  - name: cont  image: gcr.io/google-samples/node-hello:1.0  volumeMounts:  - name: vol1  mountPath: " /data/demo"  securityContext:  allowPrivilegeEscalation: false |

## Configure Service Accounts for Pods:

|  |
| --- |
| apiVersion: v1  kind: ServiceAccount  metadata:  name: pod-srv-accnt  ---  apiVersion : v1  kind: Pod  metadata:  name: pod-with-serviceaccount  spec:  serviceAccountName: pod-srv-accnt  containers:  - name: nginx  image: nginx |

## Configure Liveness Probes

|  |
| --- |
| # Liveness commad  apiVersion: v1  kind: Pod  metadata:  name: pod-liveness-command  spec:  containers:  - name: liveness  image: k8s.gcr.io/busybox  args:  - /bin/sh  - -c  - touch /tmp/healthy; sleep 30 ; rm -rf /tmp/healthy;sleep 600  livenessProbe:  exec:  command:  - cat  - /tmp/healthy  initialDelaySeconds: 5  periodSeconds: 5 |

Configure and Readiness Probes

## Assign Pods to Nodes:

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-assigned-to-node01  spec:  nodeSelector:  name: kube-02  containers:  - name: nginx  image: nginx |

## Define postStart and preStop handlers

## Create pod and use secret as environment variables :

|  |
| --- |
| apiVersion: v1  kind: Secret  metadata:  name: my-secret  data:  my-username: YWRtaW4=  my-password: cGFzd3dvcmQ=  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-with-secert-env-var  spec:  containers:  - name: nginx  image: nginx  env:  - name: MY\_USERNAME  valueFrom:  secretKeyRef:  name: my-secret  key: my-username  - name: MY\_PASSWORD  valueFrom:  secretKeyRef:  name: my-secret  key: my-password |

## Create pod and use configMaps as environment variables :

|  |
| --- |
| apiVersion: v1  kind: ConfigMap  metadata:  name: pod-confmap  namespace: default  data:  connstring: "db-server"  maxsession: "10"  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-configmap-env-var  spec:  containers:  - name: nginx  image: nginx  env:  - name: DB\_CONN\_STRING  valueFrom:  configMapKeyRef:  name: pod-confmap  key: connstring  - name: MAX\_SESSIONS  valueFrom:  configMapKeyRef:  name: pod-confmap  key: maxsession |

## Create pod with storage :

### emptyDir:

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-emptydir-vol  spec:  volumes:  - name: vol1  emptyDir: {}  containers:  - name: nginx  image: nginx  volumeMounts:  - name: vol1  mountPath: "/data" |

### hostfile :

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-hostfile-vol  spec:  volumes:  - name: vol1  hostPath:  path: "/var/log"  containers:  - name: nginx  image: nginx  volumeMounts:  - name: vol1  mountPath: "/logs" |

### Mount configMap inside it as volume:

|  |
| --- |
| apiVersion: v1  kind: ConfigMap  metadata:  name: vol-confmap  data:  firstparam: "value01"  secondparam: "value02"  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-configmap-vol  spec:  volumes:  - name: vol  configMap:  name: vol-confmap  containers:  - name: nginx  image: nginx  volumeMounts:  - name: vol  mountPath: "/configdata" |

### Mount secret inside it as volume:

|  |
| --- |
| apiVersion: v1  kind: Secret  metadata:  name: secret-vol  data:  username: YWRtaW4=  password: cGFzc3dvcmQ=  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-secret-volume  spec:  volumes:  - name : myvol  secret:  secretName: secret-vol  containers:  - name: nginx  image: nginx  volumeMounts:  - name: myvol  mountPath: "/secrets" |

### Mount PV inside Pod :

|  |
| --- |
| apiVersion: v1  kind : PersistentVolume  metadata:  name: pod-pv  spec:  storageClassName: normal  capacity:  storage: 1Gi  accessModes:  - ReadWriteOnce  hostPath:  path: "/mnt"  ---  apiVersion: v1  kind: PersistentVolumeClaim  metadata:  name: pod-pvc  spec:  storageClassName : normal  accessModes:  - ReadWriteOnce  resources:  requests:  storage: 1Gi  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-pv-vol  spec:  volumes:  - name: vol  persistentVolumeClaim:  claimName: pod-pvc  containers:  - name: nginx  image: nginx  volumeMounts:  - name : vol  mountPath: "/userdata" |

## Configuring Pod DNS Policy:

## Checking Pod DNS name / IP

|  |
| --- |
| kubectl exec dns-client -- nslookup 192-168-115-129.default.pod |

## Adding entries to Pod /etc/hosts with HostAliases

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-host-aliases  spec:  hostAliases:  - ip: "127.0.0.1"  hostnames:  - "test.cluster.local"  - "test2.cluster.local"  - ip: "10.1.2.3"  hostnames:  - "db.server.com"  containers:  - name: nginx  image: nginx |

## Horizontal Pod Autoscaler

# **Deployment :**

## Creating a Deployment Command:

|  |
| --- |
| kubectl run nginx-deployment --image=nginx --replicas=3 --labels="app=nginx" |

## Updating a Deployment image YAML :

|  |
| --- |
| apiVersion: extensions/v1beta1  kind: Deployment  metadata:  annotations:  deployment.kubernetes.io/revision: "1"  labels:  app: nginx  name: nginx-deployment  namespace: default  spec:  progressDeadlineSeconds: 600  replicas: 3  revisionHistoryLimit: 2  selector:  matchLabels:  app: nginx  strategy:  rollingUpdate:  maxSurge: 25%  maxUnavailable: 25%  type: RollingUpdate  template:  metadata:  creationTimestamp: null  labels:  app: nginx  spec:  containers:  - image: nginx  imagePullPolicy: Always  name: nginx-deployment  resources: {}  terminationMessagePath: /dev/termination-log  terminationMessagePolicy: File  dnsPolicy: ClusterFirst  restartPolicy: Always  schedulerName: default-scheduler  securityContext: {}  terminationGracePeriodSeconds: 30 |

## Scale Deployment:

|  |
| --- |
| kubectl scale deployment nginx-deployment --replicas=5 |

## Check Checking Rollout History of a Deployment:

|  |
| --- |
| $ kubectl rollout history deployment nginx-deployment  $ kubectl rollout history deployment nginx-deployment --revision=1 |

## Rolling Back a Deployment:

|  |
| --- |
| $ kubectl set image deployment/nginx-deployment nginx=nginx:1.9  $ kubectl rollout status deployment nginx-deployment  $ kubectl rollout status deployment nginx-deployment  $ kubectl rollout undo deployment/nginx-deployment --to-revision=2 |

## Pausing and Resuming a Deployment

|  |
| --- |
| $ kubectl rollout pause deployment/nginx-deployment  $ kubectl rollout resume deploy/nginx-deployment |

## Deployment revision history limit:

|  |
| --- |
| revisionHistoryLimit: 2 |

## Deployment Update strategies:

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: deployment-rollingupdate-strategy  spec:  replicas: 2  selector:  matchLabels:  app: nginx  strategy:  type: RollingUpdate  rollingUpdate:  maxSurge: 20%  maxUnavailable: 20%  template:  metadata:  labels:  app: nginx  spec:  containers:  - name: nginx  image: nginx |

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: deployment-recreate-strategy  spec:  replicas: 2  selector:  matchLabels:  app: nginx  strategy:  type: Recreate  template:  metadata:  labels:  app: nginx  spec:  containers:  - name: nginx  image: nginx |

# Service :

### Create Service :

#### Multiport Service :

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: service-mulitports  spec:  selector:  app: nginx  ports:  - name: port80  protocol : TCP  port: 80  targetPort: 80  - name: port8080  protocol: TCP  port: 8080  targetPort: 80 |

#### CluserIP

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: service-clusterip  spec:  selector:  app: nginx  type: ClusterIP  ports:  - protocol: TCP  port: 80  targetPort: 80 |

#### ExternalName:

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: service-externalname  spec:  type: ExternalName  externalName: "www.cnn.net" |

#### NodePort:

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: service-nodeport  spec:  type: NodePort  selector:  app: nginx  ports:  - protocol: TCP  port: 8090  targetPort: 80 |

#### LoadBalancer:

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: service-lb  spec:  type: LoadBalancer  clusterIP: 10.101.154.178  ports:  - protocol: TCP  port: 80 |

#### Headless Service:

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: service-headless  spec:  type: ClusterIP  clusterIP: None  selector:  app: nginx |

## Expose pod ,Deployment using commandline:

|  |
| --- |
| $ kubectl expose pod pod --port 8080 --name=pod-srv --type=NodePort --target-port=80  $ kubectl expose deployment dep-name --port 8080 --name=deployment-srv --type=NodePort --target-port=80 |

## Securing the Service

# StatefulSets

## Create MySQL StatefulSet:

## Create Redis StatefulSet:

# Jobs:

## Create Job:

|  |
| --- |
| apiVersion: batch/v1  kind: Job  metadata:  name: job01  spec:  template:  spec:  containers:  - name: pi  image: perl  command: ["perl", "-Mbignum=bpi" ,"-wle", "print bpi(2000)"]  restartPolicy: Never |

## Job Termination and Cleanup

|  |
| --- |
| apiVersion: batch/v1  kind: Job  metadata:  name: job-termination-cleanup  spec:  backoffLimit: 5 #  activeDeadlineSeconds: 10 #Once a Job reaches activeDeadlineSeconds, the Job and all of its Pods are terminated  template:  spec:  containers:  - name: busybox  image: busybox  command: ["touch","x"]  restartPolicy: Never |

## Handling Parallelism:

|  |
| --- |
| apiVersion: batch/v1  kind: Job  metadata:  name: job-parallelism  spec:  parallelism: 2  template:  spec:  containers:  - name: busybox  image: busybox  restartPolicy: OnFailure |

# CronJob:

## Create Cronjob:

|  |
| --- |
| apiVersion: batch/v1beta1  kind: CronJob  metadata:  name: cronjob-everyminute  spec:  schedule: "\*/1 \* \* \* \*"  jobTemplate:  spec:  template:  spec:  containers:  - name: busybox  image: busybox  command: ["date"]  restartPolicy: OnFailure |

# Network Policies:

|  |
| --- |
| apiVersion: networking.k8s.io/v1  kind: NetworkPolicy  metadata:  name: test-network-policy  namespace: default  spec:  podSelector:  matchLabels:  role: db  policyTypes:  - Ingress  - Egress  ingress:  - from:  - ipBlock:  cidr: 172.17.0.0/16  except:  - 172.17.1.0/24  - namespaceSelector:  matchLabels:  project: myproject  - podSelector:  matchLabels:  role: frontend  ports:  - protocol: TCP  port: 6379  egress:  - to:  - ipBlock:  cidr: 10.0.0.0/24  ports:  - protocol: TCP  port: 5978 |

## Allow All Traffic :

|  |
| --- |
| By default, if no policies exist in a namespace, then all ingress and egress traffic is allowed to and from pods in that namespace. The following examples let you change the default behavior in that namespace |

## Deny All Traffic

## Default allow all ingress traffic

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-frontend  labels:  app: nginx  tier: frontend  spec:  containers:  - name: nginx  image: nginx  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-mdw  labels:  app: nginx  tier: mdw  spec:  containers:  - name: nginx  image: nginx  ---  apiVersion: networking.k8s.io/v1  kind: NetworkPolicy  metadata:  name: allow-all-ingress  spec:  podSelector: {}  ingress:  - {} |

## Default deny all egress traffic

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: pod-frontend  labels:  app: nginx  tier: frontend  spec:  containers:  - name: nginx  image: nginx  ---  apiVersion: v1  kind: Pod  metadata:  name: pod-mdw  labels:  app: nginx  tier: mdw  spec:  containers:  - name: nginx  image: nginx  ---  apiVersion: networking.k8s.io/v1  kind: NetworkPolicy  metadata:  name: deny-all-ingress  spec:  podSelector: {}  policyTypes:  - Ingress |

## Default allow all egress traffic

|  |
| --- |
| apiVersion: networking.k8s.io/v1  kind: NetworkPolicy  metadata:  name: allow-all  spec:  podSelector: {}  egress:  - {}  policyTypes:  - Egress |

## Default deny all ingress and all egress traffic

|  |
| --- |
| **apiVersion: networking.k8s.io/v1**  **kind: NetworkPolicy**  **metadata:**  **name: default-deny**  **spec:**  **podSelector: {}**  **policyTypes:**  **- Egress** |

# Storage :

## Create Persistent Volumes

## Create PersistentVolumeClaims

## Mount PersistentVolumeClaims in Pod

## Create Storage Classes

## Dynamic Volume Provisioning

# Administrating the Cluster:

## Create Master Node using kubeadm:

## Adding Node using kubeadm :

## Backup etcd

## Labeling Nodes:

## Resource Quatas :

### Enabling Resource Quata:

### Compute Resource Quota:

### Storage Resource Quota

### Object Count Quota

## Taints and Tolerations