1. EXAM

Graphical user interface, text, application

Description automatically generated

controlplane ~ ➜ k run nginx-pod --image=nginx:alpine

pod/nginx-pod created

controlplane ~ ➜ k get pod

NAME READY STATUS RESTARTS AGE

nginx-pod 1/1 Running 0 69s

controlplane ~ ➜ k describe pod nginx-pod

Graphical user interface, text, application

Description automatically generated

controlplane ~ ➜ k run --help

controlplane ~ ➜ k run messaging --image=redis:alpine --labels="tier=msg"

controlplane ~ ➜ k describe pod messaging

Graphical user interface

Description automatically generated with medium confidence

controlplane ~ ➜ k create namespace apx-x9984574

namespace/apx-x9984574 created

controlplane ~ ➜ k get ns

NAME STATUS AGE

apx-x9984574 Active 5s

default Active 95m

kube-flannel Active 95m

kube-node-lease Active 95m

kube-public Active 95m

kube-system Active 95m

Text

Description automatically generated with medium confidence

k get nodes

NAME STATUS ROLES AGE VERSION

controlplane Ready control-plane 108m v1.26.0

controlplane ~ ➜ k get nodes -o json > /opt/outputs/nodes-z3444kd9.jsonGraphical user interface, text, application, email

Description automatically generated

controlplane ~ ➜ k get pods

NAME READY STATUS RESTARTS AGE

messaging 1/1 Running 0 28m

nginx-pod 1/1 Running 0 38m

controlplane ~ ➜ k get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 121m

controlplane ~ ➜ k expose –help

controlplane ~ ➜ k expose pod messaging --port 6379 --name messaging-service

service/messaging-service exposed

controlplane ~ ➜ k get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 122m

messaging-service ClusterIP 10.108.46.43 <none> 6379/TCP 7s

controlplane ~ ➜ k describe svc messaging-service

Name: messaging-service

Namespace: default

Labels: tier=msg

Annotations: <none>

Selector: tier=msg

Type: ClusterIP

IP Family Policy: SingleStack

IP Families: IPv4

IP: 10.108.46.43

IPs: 10.108.46.43

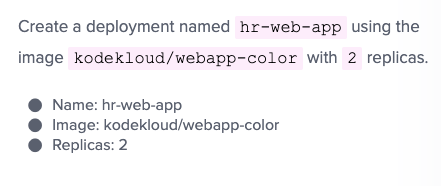
Port: <unset> 6379/TCP

TargetPort: 6379/TCP

Endpoints: 10.244.0.5:6379

Session Affinity: None

Events: <none>



controlplane ~ ➜ k create deployment hr-web-app --image=kodekloud/webapp-color --replicas=2

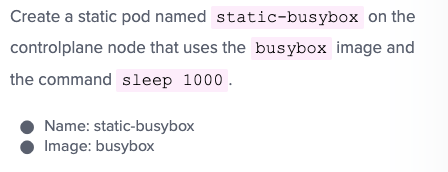
deployment.apps/hr-web-app created

controlplane ~ ➜ k get deploy

NAME READY UP-TO-DATE AVAILABLE AGE

hr-web-app 2/2 2 2 69s

controlplane ~ ➜ k describe deploy hr-web-app



controlplane ~ ➜ k run static-busybox --image=busybox --dry-run=client -o yaml --command -- sleep 1000 > static-busybox.yaml

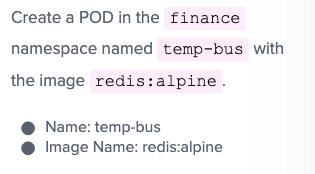
controlplane ~ ➜ mv static-busybox.yaml /etc/kubernetes/manifests/

controlplane ~ ➜ k get pods

NAME READY STATUS RESTARTS AGE

static-busybox-controlplane 1/1 Running 0 7s

controlplane ~ ➜ k describe pod static-busybox-controlplane



controlplane ~ ➜ k run temp-bus --image=redis:alpine -n finance

pod/temp-bus created

controlplane ~ ➜ k get pod -n finance

NAME READY STATUS RESTARTS AGE

temp-bus 1/1 Running 0 13s

controlplane ~ ➜ k describe pod temp-bus -n finance

Graphical user interface, text, application

Description automatically generated

controlplane ~ ➜ k get pods

NAME READY STATUS RESTARTS AGE

orange 0/1 Init:CrashLoopBackOff 2 (16s ago) 34s

static-busybox-controlplane 1/1 Running 0 4m12s

controlplane ~ ➜ k describe pod orange

controlplane ~ ➜ k logs orange init-myservice

sh: sleeeep: not found

controlplane ~ ➜ k edit pod orange

# Please edit the object below. Lines beginning with a '#' will be ignored,

# and an empty file will abort the edit. If an error occurs while saving this file will be

# reopened with the relevant failures.

#

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: "2023-02-28T09:16:26Z"

name: orange

namespace: default

resourceVersion: "6747"

uid: 06320c73-a8e9-4a5e-9d7a-a21896c8ca49

spec:

containers:

- command:

- sh

- -c

- echo The app is running! && sleep 3600

image: busybox:1.28

imagePullPolicy: IfNotPresent

name: orange-container

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /var/run/secrets/kubernetes.io/serviceaccount

name: kube-api-access-827k6

readOnly: true

dnsPolicy: ClusterFirst

enableServiceLinks: true

initContainers:

- command:

- sh

- -c

- sleep 2;



image: busybox

imagePullPolicy: Always

name: init-myservice

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /var/run/secrets/kubernetes.io/serviceaccount

name: kube-api-access-827k6

"/tmp/kubectl-edit-2984148973.yaml" 145L, 4180C 1,1 Top

A copy of your changes has been stored to "/tmp/kubectl-edit-1817101667.yaml"

error: Edit cancelled, no valid changes were saved.

controlplane ~ ✖

controlplane ~ ✖ k replace --force -f /tmp/kubectl-edit-1817101667.yaml

pod "orange" deleted

pod/orange replaced

controlplane ~ ➜ k get pods --watch

NAME READY STATUS RESTARTS AGE

orange 1/1 Running 0 41s

static-busybox-controlplane 1/1 Running 0 13m

Text

Description automatically generated

controlplane ~ ➜ k get deploy

NAME READY UP-TO-DATE AVAILABLE AGE

hr-web-app 2/2 2 2 3m38s

controlplane ~ ➜ k expose deploy hr-web-app --name=hr-web-app-service --type NodePort --port 8080

service/hr-web-app-service exposed

controlplane ~ ➜ k get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

hr-web-app-service NodePort 10.107.9.42 <none> 8080:30242/TCP 5s

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 64m

messaging-service ClusterIP 10.97.234.20 <none> 6379/TCP 6m26s

controlplane ~ ➜ k describe svc hr-web-app-service

Name: hr-web-app-service

Namespace: default

Labels: app=hr-web-app

Annotations: <none>

Selector: app=hr-web-app

Type: NodePort

IP Family Policy: SingleStack

IP Families: IPv4

IP: 10.107.9.42

IPs: 10.107.9.42

Port: <unset> 8080/TCP

TargetPort: 8080/TCP

NodePort: <unset> 30242/TCP

Endpoints: 10.244.0.6:8080,10.244.0.7:8080

Session Affinity: None

External Traffic Policy: Cluster

Events: <none>

controlplane ~ ➜ k edit svc hr-web-app-service

# Please edit the object below. Lines beginning with a '#' will be ignored,

# and an empty file will abort the edit. If an error occurs while saving this file will be

# reopened with the relevant failures.

#

apiVersion: v1

kind: Service

metadata:

creationTimestamp: "2023-02-28T09:50:56Z"

labels:

app: hr-web-app

name: hr-web-app-service

namespace: default

resourceVersion: "5500"

uid: e755835a-afa0-4375-a42b-bbd43d129a68

spec:

clusterIP: 10.107.9.42

clusterIPs:

- 10.107.9.42

externalTrafficPolicy: Cluster

internalTrafficPolicy: Cluster

ipFamilies:

- IPv4

ipFamilyPolicy: SingleStack

ports:



- nodePort: 30082

port: 8080

protocol: TCP

targetPort: 8080



selector:

app: hr-web-app

sessionAffinity: None

type: NodePort

status:

loadBalancer: {}

controlplane ~ ➜ k get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

hr-web-app-service NodePort 10.107.9.42 <none> 8080:30082/TCP 3m38s

kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 67m

messaging-service ClusterIP 10.97.234.20 <none> 6379/TCP 9m59s

Graphical user interface, text, application

Description automatically generated

controlplane ~ ➜ k get nodes

NAME STATUS ROLES AGE VERSION

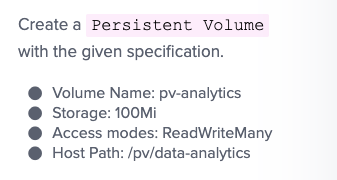
controlplane Ready control-plane 69m v1.26.0

controlplane ~ ✖ k get nodes -o jsonpath='{.items[\*].status.nodeInfo.osImage}'

Ubuntu 20.04.5 LTS

controlplane ~ ➜ k get nodes -o jsonpath='{.items[\*].status.nodeInfo.osImage}' > /opt/outputs/nodes\_os\_x43kj56.txt

controlplane ~ ➜ cat /opt/outputs/nodes\_os\_x43kj56.txt



controlplane ~ ➜ vi pv.yaml

apiVersion: v1

kind: PersistentVolume

metadata:

name: pv-analytics

spec:

capacity:

storage: 100Mi

accessModes:

- ReadWriteMany

hostPath:

path: /pv/data-analytics

controlplane ~ ➜ vi pv.yaml

controlplane ~ ➜ k create -f pv.yaml

persistentvolume/pv-analytics created

controlplane ~ ➜ k get pv

NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS REASON AGE

pv-analytics 100Mi RWX Retain Available 8s

controlplane ~ ➜ k describe pv pv-analytics

1. Exam

Graphical user interface

Description automatically generated with low confidence

controlplane ~ ✖ ETCDCTL\_API=3 etcdctl snapshot save -h

controlplane ~ ➜ cat /etc/kubernetes/manifests/etcd.yaml | grep file

- --cert-file=/etc/kubernetes/pki/etcd/server.crt

- --key-file=/etc/kubernetes/pki/etcd/server.key

- --peer-cert-file=/etc/kubernetes/pki/etcd/peer.crt

- --peer-key-file=/etc/kubernetes/pki/etcd/peer.key

- --peer-trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt

- --trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt

seccompProfile:

controlplane ~ ➜ vi /etc/kubernetes/manifests/etcd.yaml

- --listen-client-urls=https://127.0.0.1:2379,https://192.17.230.3:2379

controlplane ~ ➜ ETCDCTL\_API=3 etcdctl --endpoints 127.0.0.1:2379 snapshot save /opt/etcd-backup.db \

> --cacert=/etc/kubernetes/pki/etcd/ca.crt \

> --cert=/etc/kubernetes/pki/etcd/server.crt \

> --key=/etc/kubernetes/pki/etcd/server.key

Snapshot saved at /opt/etcd-backup.db

ls /opt/etcd-backup.db

Text

Description automatically generated

controlplane ~ k run redis-storage --image=redis:alpine --dry-run=client -o yaml > redis-storage.yaml

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

run: redis-storage

name: redis-storage

spec:

containers:

- image: redis:alpine

name: redis-storage

volumeMounts:

- mountPath: /data/redis

name: cache-volume

resources: {}

dnsPolicy: ClusterFirst

restartPolicy: Always

volumes:

- name: cache-volume

emptyDir:

status: {}

controlplane ~ ➜ k create -f redis-storage.yaml

pod/redis-storage created

controlplane ~ ➜ k get pods –watch

controlplane ~ ➜ k describe pod redis-storage

Text

Description automatically generated with medium confidence

controlplane ~ ➜ k run super-user-pod --image=busybox:1.28 --dry-run=client -o yaml > super-user-pod.yaml

controlplane ~ ➜ vi super-user-pod.yaml

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

run: super-user-pod

name: super-user-pod

spec:

containers:

- image: busybox:1.28

name: super-user-pod

command: ["sleep","4800"]

resources: {}

securityContext:

capabilities:

add: ["SYS\_TIME"]

dnsPolicy: ClusterFirst

restartPolicy: Always

status: {}

controlplane ~ ➜ k create -f super-user-pod.yaml

pod/super-user-pod created

controlplane ~ ➜ k get pods

controlplane ~ ➜ k describe pod super-user-pod

Text

Description automatically generated

controlplane ~ ➜ cat /root/CKA/use-pv.yaml

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

run: use-pv

name: use-pv

spec:

containers:

- image: nginx

name: use-pv

resources: {}

dnsPolicy: ClusterFirst

restartPolicy: Always

status: {}

controlplane ~ ➜ k get pv

NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS REASON AGE

pv-1 10Mi RWO Retain Available 82s

controlplane ~ ➜ vi pvc.yaml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: my-pvc

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 10Mi

controlplane ~ ➜ k create -f pvc.yaml

persistentvolumeclaim/mypvc created

controlplane ~ ➜ k get pvc

NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE

mypvc Bound pv-1 10Mi RWO 6s

controlplane ~ ✖ vi /root/CKA/use-pv.yaml

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

run: use-pv

name: use-pv

spec:

containers:

- image: nginx

name: use-pv

resources: {}

volumeMounts:

- mountPath: "/data"

name: mypd

dnsPolicy: ClusterFirst

restartPolicy: Always

volumes:

- name: mypd

persistentVolumeClaim:

claimName: my-pvc

status: {}

controlplane ~ ➜ k create -f /root/CKA/use-pv.yaml

pod/use-pv created

controlplane ~ ➜ k get pod –watch



controlplane ~ ➜ k create deployment nginx-deploy --image=nginx:1.16 --replicas=1

deployment.apps/nginx-deploy created

controlplane ~ ➜ kubectl set image deployment/nginx-deploy nginx=nginx:1.17

deployment.apps/nginx-deploy image updated

Text

Description automatically generated with medium confidence

controlplane ~ ➜ ls /root/CKA/

john.csr john.key use-pv.yaml

controlplane ~ ✖ cd /root/CKA/

controlplane ~/CKA ➜ vi jhon-csr.yaml

apiVersion: certificates.k8s.io/v1

kind: CertificateSigningRequest

metadata:

name: john-developer

spec:

request: 

signerName: kubernetes.io/kube-apiserver-client

usages:

- client auth

EOF

controlplane ~/CKA ➜ cat john.csr | base64 | tr -d "\n"

controlplane ~/CKA ➜ vi jhon-csr.yaml

controlplane ~/CKA ➜ k create -f jhon-csr.yaml

certificatesigningrequest.certificates.k8s.io/john-developer created

controlplane ~/CKA ➜ k get csr

controlplane ~/CKA ➜ k certificate approve john-developer

certificatesigningrequest.certificates.k8s.io/john-developer approved

controlplane ~/CKA ➜ kubectl create role developer --verb=create,get,list,update,delete --resource=pods -n development

controlplane ~/CKA ➜ k get role -n development

controlplane ~/CKA ➜ k describe role -n development

Name: developer

Labels: <none>

Annotations: <none>

PolicyRule:

Resources Non-Resource URLs Resource Names Verbs

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

pods [] [] [create get list update delete]

controlplane ~/CKA ➜ k auth can-i get pods --namespace=development --as john

no

controlplane ~/CKA ✖ k auth can-i create pods --namespace=development --as john

no

controlplane ~/CKA ➜ k create rolebinding –help

controlplane ~/CKA ➜ kubectl create rolebinding john-developer --role=developer --user=john -n development

rolebinding.rbac.authorization.k8s.io/john-developer created

controlplane ~/CKA ✖ k get rolebindings.rbac.authorization.k8s.io -n development NAME ROLE AGE

john-developer Role/developer 51s

controlplane ~/CKA ➜ k describe rolebindings.rbac.authorization.k8s.io -n development

Name: john-developer

Labels: <none>

Annotations: <none>

Role:

Kind: Role

Name: developer

Subjects:

Kind Name Namespace

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

User john

controlplane ~/CKA ✖ k auth can-i create pods --namespace=development --as john

yes

controlplane ~/CKA ➜ k auth can-i create pods --namespace=development --as john

yes

Text

Description automatically generated

controlplane ~/CKA ➜ k run nginx-resolver --image=nginx

pod/nginx-resolver created

controlplane ~/CKA ➜ k expose pod nginx-resolver --name=nginx-resolver-service --port=80

service/nginx-resolver-service exposed

controlplane ~/CKA ➜ k describe svc nginx-resolver-service

Name: nginx-resolver-service

Namespace: default

Labels: run=nginx-resolver

Annotations: <none>

Selector: run=nginx-resolver

Type: ClusterIP

IP Family Policy: SingleStack

IP Families: IPv4

IP: 10.99.191.223

IPs: 10.99.191.223

Port: <unset> 80/TCP

TargetPort: 80/TCP

Endpoints: 10.244.192.4:80

Session Affinity: None

Events: <none>

controlplane ~/CKA ➜ k run busybox --image=busybox:1.28 -- sleep 4000

pod/busybox created

controlplane ~/CKA ➜ k get pods

NAME READY STATUS RESTARTS AGE

busybox 1/1 Running 0 5s

nginx-deploy-c848b6868-mz2hz 1/1 Running 0 30m

nginx-resolver 1/1 Running 0 3m4s

redis-storage 1/1 Running 0 41m

super-user-pod 1/1 Running 0 40m

use-pv 1/1 Running 0 37m

controlplane ~/CKA ✖ k exec busybox -- nslookup nginx-resolver-service

Server: 10.96.0.10

Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local

Name: nginx-resolver-service

Address 1: 10.99.191.223 nginx-resolver-service.default.svc.cluster.local

controlplane ~/CKA ➜ k exec busybox -- nslookup nginx-resolver-service > /root/CKA/nginx.svc

controlplane ~/CKA ➜ k get pods -o wide

controlplane ~/CKA ➜ k exec busybox -- nslookup 10-244-192-4.default.pod.cluster.local > /root/CKA/nginx.pod

Graphical user interface, text, application, email

Description automatically generated

controlplane ~ ➜ k get nodes -o wide

controlplane ~ ➜ ssh 192.24.174.3

root@node01 ~ ➜ ls /etc/kubernetes/manifests/

controlplane ~ ➜ k run nginx-critical --image=nginx --restart=Always --dry-run=client -o yaml

root@node01 ~ ➜ cat > /etc/kubernetes/manifests/nginx-critical.yaml