# RBAC授权

kubectl config use-context k8s

kubectl create clusterrole deployment-clusterrole --verb=create --resource=deployments,daemonsets,statefulsets

kubectl create serviceaccount cicd-token -n app-team1

kubectl create rolebinding cicd-token --serviceaccount=app-team1:cicd-token --clusterrole=deployment-clusterrole -n app-team1

测试服务账号（SA）权限：

kubectl -n app-team1 describe rolebinding cicd-token-rolebinding ​或者

kubectl --as=system:serviceaccount:app-team1:cicd-token get pods -n app-team1

# 节点设置不可用

kubectl config use-context ek8s

kubectl cordon ek8s-node-1

kubectl drain ek8s-node-1 **--ignore-daemonsets** # 设置不可调度并驱逐节点上 Pod

报错：--delete-local-data 或者 --delete-emptydir-data​ --force

kubectl uncordon ek8s-node-1

# 升级k8s版本

kubectl config use-context mk8s

kubectl cordon mk8s-master-0

kubectl drain mk8s-master-0 --ignore-daemonsets

ssh mk8s-master-0

sudo -i

apt install kubeadm=1.20.1-00 –y

kubeadm upgrade plan

kubeadm upgrade apply v1.20.1 **--etcd-upgrade=false** # 题目要求不升级etcd

# 升级kubelet和kubectl

apt install kubelet=1.20.1-00 kubectl=1.20.1-00 -y

systemctl restart kubelet

# 设置为可调度

kubectl uncordon mk8s-master-0

# etcd备份与恢复

# 备份

ETCDCTL\_API=3 **etcdctl** **snapshot** **save** /data/backup/etcd-snapshot.db --**endpoints**=https://127.0.0.1:2379 --**cacert**=/opt/KUIN00601/ca.crt --**cert**=/opt/KUIN00601/etcd-client.crt --**key**=/opt/KUIN00601/etcd-client.key

# 恢复

systemctl stop etcd

systemctl cat etcd # 确认下数据目录（--data-dir值）

mv /var/lib/etcd /var/lib/etcd.bak

ETCDCTL\_API=3 **etcdctl** **snapshot** **restore** /data/backup/etcd-snapshot-previous.db --**data-dir**=/var/lib/etcd

chown -R etcd:etcd /var/lib/etcd

systemctl start etcd

# 网络策略

kubectl config use-context hk8s

给命名空间打标签：kubectl label namespace big-corp name=big-corp

|  |
| --- |
| apiVersion: networking.k8s.io/v1  kind: NetworkPolicy  metadata:  name: allow-port-from-namespace  namespace: my-app  spec:  **podSelector: {}**  policyTypes:  - Ingress  ingress:  - from:  - namespaceSelector:  matchLabels:  name: big-corp  ports:  - protocol: TCP  port: 8080 |

# SVC暴露应用

kubectl config use-context k8s

|  |
| --- |
| kubectl edit deployment front-end  …  containers:  - image: nginx  imagePullPolicy: Always  name: nginx  ports:  - name: http  protocol: TCP  containerPort: 80  … |

kubectl expose deployment front-end --port=80 --target-port=80 --type=NodePort --name=front-end-svc

# Ingress

kubectl config use-context k8s

|  |
| --- |
| apiVersion: networking.k8s.io/v1  kind: Ingress  metadata:  namespace: ing-internal  annotations:  nginx.ingress.kubernetes.io/rewrite-target: /  spec:  rules:  - http:  paths:  - path: /hello  pathType: Prefix  backend:  service:  name: hello  port:  number: 5678 |

kubectl get ingress -n ing-internal

curl -kL <获取ingress的IP地址>/hello

# 扩容Pod数量

kubectl config use-context k8s

kubectl scale deployment loadbalancer --replicas=5

# nodeSelector

kubectl config use-context k8s

kubectl label nodes node02 disk=ssd​

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: nginx-kusc00401  spec:  containers:  - name: nginx  image: nginx  nodeSelector:  disk: ssd |

kubectl get po nginx-kusc00401 -o wide

# 统计准备就绪节点数量

|  |
| --- |
| kubectl config use-context k8s  kubectl describe nodes | grep -i Taints | grep -vc NoSchedule  echo "查出来的数字" > /opt/KUSC00402/kusc00402.txt ​ |

# Pod配置多容器

kubectl config use-context k8s

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: kucc4  spec:  containers:  - name: nginx  image: nginx  - name: redis  image: redis  - name: memcached  image: memcached |

# 创建PV

kubectl config use-context hk8s

|  |
| --- |
| apiVersion: v1  kind: PersistentVolume  metadata:  name: app-data  spec:  capacity:  storage: 2Gi  accessModes:  - ReadWriteOnce  hostPath:  path: "/srv/app-data" |

# Pod使用PVC

kubectl config use-context ok8s

|  |
| --- |
| apiVersion: v1  kind: PersistentVolumeClaim  metadata:  name: pv-volume  spec:  storageClassName: csi-hostpath-sc  accessModes:  - ReadWriteOnce  resources:  requests:  storage: 10Mi  ---  apiVersion: v1  kind: Pod  metadata:  name: web-server  spec:  containers:  - name: web-server  image: nginx  ports:  - containerPort: 80  name: http-server  volumeMounts:  - mountPath: /usr/share/nginx/html  name: data  volumes:  - name: data  persistentVolumeClaim:  claimName: pv-volume |

# 扩容PVC容量

kubectl edit pvc pv-volume --save-config

# 获取Pod错误日志

kubectl config use-context k8s

kubectl logs bar | grep file-not-found > /opt/KUTR00101/bar

# 给Pod增加一个容器（边车）

kubectl config use-context k8s

|  |
| --- |
| apiVersion: v1  kind: Pod  metadata:  name: legacy-app  spec:  containers:  - name: count  image: busybox  args:  - /bin/sh  - -c  - >  i=0;  while true;  do  echo "$i: $(date)" >> /var/log/legacy-app.log;  sleep 1;  done  **volumeMounts:**  **- name: varlog**  **mountPath: /var/log**  **- name: sidecar**  **image: busybox**  **args: [/bin/sh, -c, 'tail -n+1 -f /var/log/legacy-app.log']**  **volumeMounts:**  **- name: varlog**  **mountPath: /var/log**  volumes: # volumes块在导出的yaml下面已经有了，在已有的添加下面两行即可  **- name: varlog**  **emptyDir: {}** |

kubectl get pod legacy-app -o yaml > legacy-app.yaml # 导出后修改文件

kubectl delete pod legacy-app

kubectl apply -f legacy-app.yaml

# 统计使用CPU最高的Pod

kubectl config use-context k8s

kubectl top pod -l name=cpu-utilizer --sort-by="cpu" –A

# 然后将第一个Pod名称写到文件

echo "<podname>" > /opt/KUR00401.txt

# 节点NotReady处理

kubectl config use-context wk8s

kubectl get node

ssh wk8s-node-0

sudo -i

systemctl status kubelet

systemctl start kubelet

systemctl enable kubelet