CKA-安全管理

讲师： 老段 RHCE/RHCA/COA/CKA

了解验证authentication

用户登录方式

vim /etc/kubernetes/manifests/kube-apiserver.yaml

basic-auth-file ----输入用户名和密码

token-auth-file ----输入token来进行验证

base authentication

wget https://storage.googleapis.com/kubernetes-release/release/v1.10.11/kubernetes-client-linux-amd64.tar.gz

vim /etc/kubernetes/manifests/kube-apiserver.yaml
- --basic-auth-file=/etc/kubernetes/pki/aa.csv #放在其他目录有可能访问不到
密码文件格式：密码,用户名,uid
cat /etc/kubernetes/pki/aa.csv
redhat,admin,1
redhat,tom,2
redhat,bob,3

kubectl create clusterrolebinding cluster-test2 --clusterrole=cluster-admin --user=admin

/tmp/kubectl -s="https://192.168.26.51:6443" --username="admin" --password="redhat" get pods -n kube-system

使用证书登录

在其他机器上
/tmp/kubectl -s="https://192.168.26.51:6443" --insecure-skip-tls-verify=true --username="admin" --password="redhat" get pods -n kube-system

生成私钥
openssl genrsa -out client.key 2048
生成证书请求文件
openssl req -new -key client.key -subj "/CN=vms60" -out client.csr
拷贝到CA
scp client.csr 192.168.26.51:/tmp/
签名证书
openssl x509 -req -in /tmp/client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out client.crt -days 3650
拷贝回client
scp ca.crt client.crt 192.168.26.60:/ca
/tmp/kubectl -s="https://192.168.26.51:6443" --certificate-authority=ca.crt --client-certificate=client.crt --client-key=client.key --username="admin" --password="redhat" get pods -n kube-system

token authentication

openssl rand -hex 10

cat bb.csv
50817fba8bc7491b257d,admin2,3

- --token-auth-file=/etc/kubernetes/pki/bb.csv

/tmp/kubectl -s="https://192.168.26.51:6443" --token="50817fba8bc7491b257d" get pods
-n kube-system

了解授权authorization

- --authorization-mode=Node,RBAC
- --authorization-mode=AlwaysAllow #允许所有请求
- --authorization-mode=AlwaysDeny #拒绝所有请求
- --authorization-mode=ABAC
  Attribute-Based Access Control 不够灵活放弃

- --authorization-mode=RBAC
  Role Based Access Control
- --authorization-mode=Node

Node授权器主要用于各个node上的kubelet访问apiserver时使用的，其他一般均由RBAC授权器来授权

role管理

```
[root@vms51 tmp]# /tmp/kubectl -s="https://192.168.26.51:6443" --username="tom" --
password="redhat" get pods
Error from server (Forbidden): pods is forbidden: User "tom" cannot list pods in t
he namespace "default"
```

kubectl api-versions

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: [""]
 resources: ["pods"]
 verbs: ["get", "watch", "list"]
```

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: [""]
 resources: ["pods","services"]
 verbs: ["get", "watch", "list"]
```

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: ["extensions"]
 resources: ["deployments"]
 verbs: ["get", "watch", "list"]
```

```
[root@vms51 ~]# /tmp/kubectl -s="https://192.168.26.51:6443" --username="tom" --pas
sword="redhat" run nginx --image=nginx
Error from server (Forbidden): deployments.extensions is forbidden: User "tom" cann
ot create deployments.extensions in the namespace "default"
```

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: ["extensions"]
 resources: ["deployments"]
 verbs: ["get", "watch", "list","create"]
```

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: ["extensions","apps"]
 resources: ["deployments","replicasets"]
 verbs: ["get", "watch", "list","create","update","delete"]
```

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: default
 name: pod-reader
rules:
- apiGroups: ["extensions","apps"]
 resources: ["deployments","replicasets"]
 verbs: ["get", "watch",
"list","create","update","delete"]
```

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: read-pods
 namespace: default
subjects:
- kind: User
 name: tom
 apiGroup: rbac.authorization.k8s.io
 namespace: ns01
roleRef:
 kind: ClusterRole
 name: pod-reader
 apiGroup: rbac.authorization.k8s.io
```

命令行

```
kubectl create clusterrolebinding myclusterbind1 --clusterrole=cluster-admin --user=tom
```

rolebinding

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: read-pods
 namespace: default
subjects:
- kind: User
 name: tom
 apiGroup: rbac.authorization.k8s.io
roleRef:
 kind: Role name: pod-reader
 apiGroup: rbac.authorization.k8s.io
```

```
[root@vms51 ~]# /tmp/kubectl -s="https://192.168.26.51:6443" --username="tom" --pa
ssword="redhat" get pods
No resources found.
[root@vms51 ~]#
```

apiversion

pod--------- v1
deployment------ extensions/v1beta1

daemonset---- ----extensions/v1beta1

job ----batch/v1

cronjob ---- batch/v2alpha1

service ---- v1

role ---rbac.authorization.k8s.io/v1
RoleBinding ---rbac.authorization.k8s.io/v1

创建admin用户

```
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: admin
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin
  namespace: kube-system
```

配置dashboard

kubectl describe secrets admin-token-6d5h5 -n kube-system

资源请求与限制

docker pull hub.c.163.com/library/centos

```
apiVersion: v1
kind: Pod
metadata:
 name: demo
 labels:
  purpose: demonstrate-envars
spec:
 volumes:
 - name: volume1
  hostPath:
   path: /zz
 containers:
 - name: demo1
  image: hub.c.163.com/library/centos
  command: ['sh','-c','sleep 50000']
  volumeMounts:
  - mountPath: /xx
   name: volume1
```

```
containers:
- name: demo1
 image: hub.c.163.com/library/centos
 command: ['sh','-c','sleep 50000']
 resources:
  requests:
   memory: "256Mi"
   cpu: "500m"
  limits:
   memory: "512Mi"
   cpu: "1000m"
```

requests
---容器所需最小资源

limits
---容器申请资源上限

LimitRange

设置默认

如果容器里没有声明请求和限制则使用默认的请求和限制

```
apiVersion: v1
kind: LimitRange
metadata:
 name: mem-limit-range
spec:
 limits:
 - default:
   memory: 512Mi
  defaultRequest:
   memory: 256Mi
  type: Container
```

```
apiVersion: v1
kind: LimitRange
metadata:
 name: cpu-limit-range
spec:
 limits:
 - default:
   cpu: 1
  defaultRequest:
   cpu: 0.5
  type: Container
```

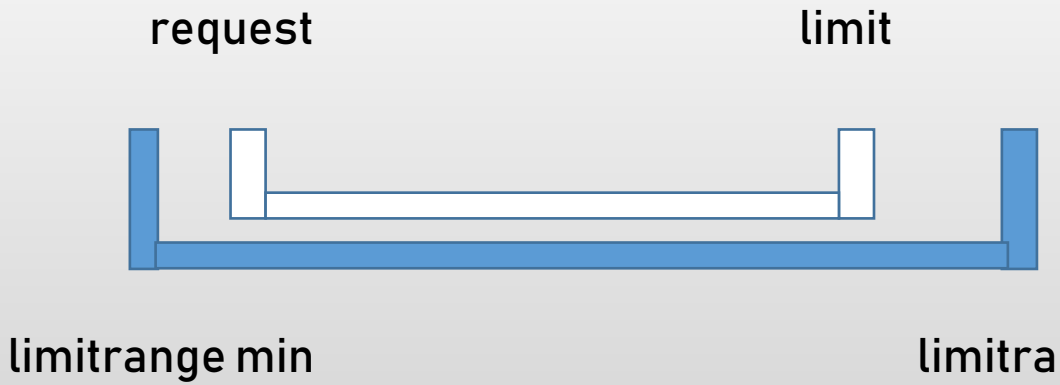
```
apiVersion: v1
kind: LimitRange
metadata:
 name: storagelimits
spec:
 limits:
 - type:
PersistentVolumeClaim
  max:
   storage: 2Gi
  min:
   storage: 1Gi
```

```
apiVersion: v1
kind: LimitRange
metadata:
 name: mem-min-max-demo-lr
 namespace: ns01
spec:
 limits:
 - max:
    memory: 1Gi
   min:
    memory: 512Mi
   type: Container
```

如果容器内没有声明最大值最小值
则使用这里设置的

如果容器里声明了请求和限制大于或者小于 limitrange里的max或者min，都会导致pod创建不成功。

容器申请的资源不能超过limit



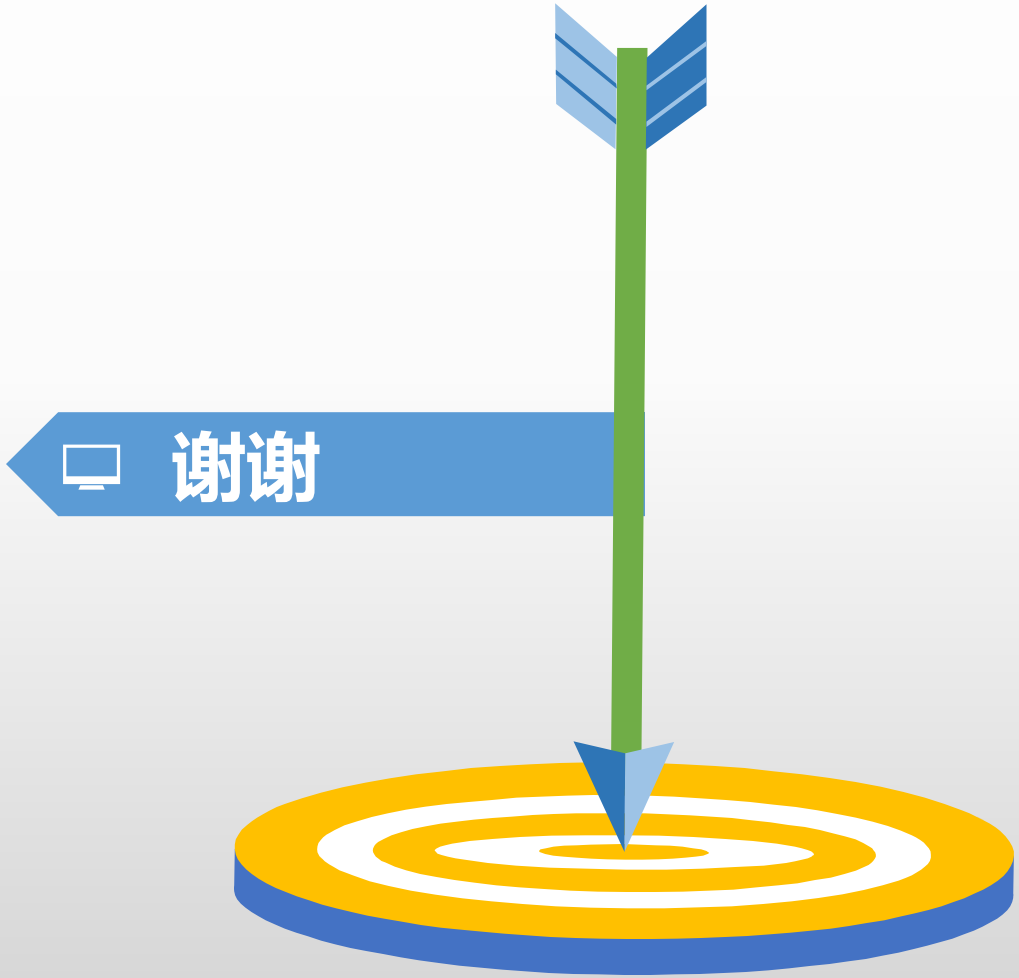
resourcequota

```
apiVersion: v1
kind: ResourceQuota
metadata:
 name: compute-resources
spec:
 hard:
  pods: "4"
  requests.cpu: "1"
  requests.memory: 1Gi
  limits.cpu: "2"
  limits.memory: 2Gi
  requests.nvidia.com/gpu: 4
  configmaps: "10"
  persistentvolumeclaims: "4"
  replicationcontrollers: "20"
  secrets: "10"
  services: "10"
  services.loadbalancers: "2"
```

limitrange 用来限制每个pod的资源

resourcequtoa 用来限制项目里可以使用多少资源

```
apiVersion: v1
kind: ResourceQuota
metadata:
 name: compute-resources
spec:
 hard:
  pods: "4"
```


谢谢