Lab Exercise14-Implementing Resource Quota in Kubernetes

Name : Yuvraj Pratap Singh
Batch : 2(Devops)
Sap id. : 500125214

Objective:

In Kubernetes, Resource Quotas are used to control the resource consumption of namespaces. They help in managing and enforcing limits on the usage of resources like CPU, memory, and the number of objects (e.g., Pods, Services) within a namespace. This exercise will guide you through creating and managing Resource Quotas to limit the resources used by applications in a specific namespace.

Step 1: Understand Resource Quotas

Resource Quotas allow you to:

- Limit the amount of CPU and memory a namespace can use.
- Control the number of certain types of resources (e.g., Pods, Services, PersistentVolumeClaims) in a namespace.
- Prevent a namespace from consuming more resources than allocated, ensuring fair usage across multiple teams or applications.

Step 2: Create a Namespace

First, create a namespace where you will apply the Resource Quota. This helps in isolating and controlling resource usage within that specific namespace.

Create a YAML file named **quota-namespace.yaml** with the following content:

```
apiVersion: v1
kind: Namespace
metadata:
  name: myns
```

Apply the YAML to create the namespace:

```
kubectl apply -f quota-namespace.yaml
```

Verify that the namespace is created:

```
kubectl get namespaces
```

You should see quota-example listed in the output.

Step 3: Define a Resource Quota

Next, create a Resource Quota YAML file named **resource-quota.yaml** with the following content:

```
apiVersion: v1
kind: ResourceQuota
metadata:
  name: myns-quota    # The name of the Resource Quota.
  namespace: myns # The namespace to which the Resource Quota will apply.
spec:
  hard:              # The hard limits imposed by this Resource Quota.
requests.cpu: "2"    # The total CPU resource requests allowed in the namespace (2 cores).
requests.memory: "4Gi" # The total memory resource requests allowed in the namespace (4 GiB).
limits.cpu: "4"      # The total CPU resource limits allowed in the namespace (4 cores).
limits.memory: "8Gi" # The total memory resource limits allowed in the namespace (8 GiB).
    pods: "10"         # The total number of Pods allowed in the namespace.
persistentvolumeclaims: "5" # The total number of PersistentVolumeClaims allowed in the namespace.
configmaps: "10"    # The total number of ConfigMaps allowed in the namespace.
    services: "5"       # The total number of Services allowed in the namespace.
```

Step 4: Apply the Resource Quota

Apply the Resource Quota YAML to the namespace:

```
kubectl apply -f resource-quota.yaml
```

Verify that the Resource Quota is applied:

```
kubectl get resourcequota -n myns
```

To see the details of the applied Resource Quota:

```
kubectl describe resourcequotamyns-quota -n myns
```

Step 5: Test the Resource Quota

Let's create some resources in the quota-example namespace to see how the Resource Quota affects them.

Deploy a ReplicaSet with Resource Requests and Limits
Create a YAML file named **nginx-replicaset-quota.yaml** with the following content:

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
 name: nginx-replicaset
 namespace: myns
spec:
 replicas: 5         # Desired number of Pod replicas.
 selector:
matchLabels:
    app: nginx
 template:
  metadata:
   labels:
    app: nginx
  spec:
   containers:
   - name: nginx
     image: nginx:latest
     ports:
```

```
      - containerPort: 80
    resources:      # Define resource requests and limits.
     requests:
      memory: "100Mi"
cpu: "100m"
      limits:
       memory: "200Mi"
cpu: "200m"
```

Explanation:

This ReplicaSet requests a total of 500m CPU and 500Mi memory across 5 replicas.
It also limits each replica to use a maximum of 200m CPU and 200Mi memory.

Apply this YAML to create the ReplicaSet:

```
kubectl apply -f nginx-replicaset-quota.yaml
```

Check the status of the Pods and ensure they are created within the constraints of the Resource Quota:

```
kubectl get pods -n myns
```

To describe the Pods and see their resource allocations:

```
kubectl describe pods -l app=nginx -n quota-example
```

Attempt to Exceed the Resource Quota

Try creating additional resources to see if they are rejected when exceeding the quota. For example, create more Pods or increase the CPU/memory requests to exceed the quota limits.

Create a YAML file named **nginx-extra-pod.yaml** with the following content:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-extra-pod
  namespace: myns
spec:
  containers:
  - name: nginx
    image: nginx:latest
    resources:
      requests:
        memory: "3Gi" # Requests a large amount of memory.
cpu: "2"     # Requests a large amount of CPU.
      limits:
        memory: "4Gi"
cpu: "2"
```

Apply this YAML to create the Pod:

```
kubectl apply -f nginx-extra-pod.yaml
```

This should fail due to exceeding the Resource Quota. Check the events to see the failure reason:

```
kubectl get events -n quota-example
```

Look for error messages indicating that the Pod creation was denied due to resource constraints.

Step 6: Clean Up Resources

To delete the resources you created:

```
kubectl delete -f nginx-replicaset-quota.yaml
kubectl delete -f nginx-extra-pod.yaml
kubectl delete -f resource-quota.yaml
kubectl delete namespace myns
```

```
PS C:\Users\LENOVO\Ex8> kubectl apply -f quota-namespace.yaml
namespace/myns created
PS C:\Users\LENOVO\Ex8> kubectl get namespaces
NAME                     STATUS     AGE
default                  Active     30m
kube-node-lease          Active     30m
kube-public              Active     30m
kube-system              Active     30m
kubernetes-dashboard     Active     27m
myns                     Active     7s
PS C:\Users\LENOVO\Ex8> kubectl apply -f resource-quota.yaml
error: error parsing resource-quota.yaml: error converting YAML to JSON: yaml: line 11: did not
PS C:\Users\LENOVO\Ex8> kubectl apply -f resource-quota.yaml
resourcequota/myns-quota created
PS C:\Users\LENOVO\Ex8> kubectl get resourcequota -n myns
NAME           REQUEST
                         AGE
myns-quota    configmaps: 1/10, persistentvolumeclaims: 0/5, pods: 0/10, requests.cpu: 0/2, reque
limits.memory: 0/8Gi   11s
PS C:\Users\LENOVO\Ex8> kubectl describe resourcequotamyns-quota -n myns
error: the server doesn't have a resource type "resourcequotamyns-quota"
PS C:\Users\LENOVO\Ex8> ^C
PS C:\Users\LENOVO\Ex8> kubectl apply -f nginx-replicaset-quota.yaml
replicaset.apps/nginx-replicaset created
PS C:\Users\LENOVO\Ex8> kubectl get pods -n myns
No resources found in myns namespace.
PS C:\Users\LENOVO\Ex8> kubectl describe pods -l app=nginx -n quota-example
No resources found in quota-example namespace.
PS C:\Users\LENOVO\Ex8> kubectl delete -f nginx-replicaset-quota.yaml
replicaset.apps "nginx-replicaset" deleted from myns namespace
PS C:\Users\LENOVO\Ex8> kubectl delete -f nginx-extra-pod.yaml
error: the path "nginx-extra-pod.yaml" does not exist
PS C:\Users\LENOVO\Ex8> kubectl delete -f resource-quota.yaml
resourcequota "myns-quota" deleted from myns namespace
```