Reserving Compute Capacity for Applications

Objectives

- Configure an application with resource requests so Kubernetes can make scheduling decisions.

Kubernetes Pod Scheduling

The Red Hat OpenShift Container Platform pod scheduler determines the placement of new pods onto nodes in the cluster. Appropriate scheduling prevents resource contention and places workloads on nodes that meet their specific requirements. The pod scheduler algorithm follows a three-step process:

Filtering nodes
A pod can define a node selector that matches the labels in the cluster nodes. The scheduler considers only nodes with matching labels as eligible.

Additionally, the scheduler filters the list of running nodes by evaluating each node against a set of predicates. A pod can define resource requests for compute resources such as CPU, memory, and storage. Only nodes with enough available computer resources are eligible.

The filtering step reduces the list of eligible nodes. In some cases, the pod could run on any of the nodes. In other cases, all the nodes are filtered out. The pod cannot be scheduled until a node with the appropriate prerequisites becomes available.

If all nodes are filtered out, then a `FailedScheduling` event is generated for the pod.

Prioritizing nodes
By using multiple priority criteria, the scheduler determines a weighted score for each node. Nodes with higher scores are better candidates to run the pod.

Selecting a node
The scheduler sorts the candidate list according to these scores, and selects the node with the highest score. If multiple nodes have the same high score, then one node is selected in a round-robin fashion. After a host is selected, the cluster generates a `Scheduled` event for the pod.

NOTE

The scheduler is flexible and can be customized for advanced scheduling situations. Customizing the scheduler is outside the scope of this course.

Configuring Compute Resource Requests

For applications that require a specific amount of compute resources, you can define a resource request in the pod definition. The resource requests assign hardware resources for the application deployment.

Resource requests specify the minimum required compute resources to schedule a pod. The scheduler tries to find a node with enough available compute resources to satisfy the pod requests.

In Kubernetes, memory resources are measured in bytes, and CPU resources are measured in CPU units. CPU units are allocated by using millicore units. A millicore is a CPU core, either virtual or physical, that is split into 1000 units. A request value of `"1000 m"` allocates an entire CPU core to a pod. You can also use fractional values to allocate CPU resources. For example, you can set the CPU resource request to the `0.1` value, which represents 100 millicores (`100 m`). Likewise, a CPU resource request with the `1.0` value represents an entire CPU or 1000 millicores (`1000 m`).

You can define resource requests for each container in a deployment resource. If you do not define resources, then the container specification shows a `resources: {}` line.

To specify requests, modify the `resources: {}` section in the deployment manifest. The following example defines a resource request of 100 millicores (`100m`) of CPU and one gibibyte (`1Gi`) of memory.

```
...output omitted...
    spec:
      containers:
      - image: quay.io/redhattraining/hello-world-nginx:v1.0
        name: hello-world-nginx
        resources:
          requests:
            cpu: "100m"
            memory: "1Gi"
```

If you use the `oc edit` command to modify a deployment, then ensure that you use the correct indentation. Indentation mistakes can result in the editor not saving changes. Alternatively, use the `oc set resources` command to specify resource requests. The following command sets the same requests as the preceding example:

```
[user@host ~]$ oc set resources deployment hello-world-nginx \
--requests cpu=100m,memory=1gi
```

The `oc set resources` command works with any resource that includes a pod template, such as `Deployment` and `Job` resources.

Inspecting Cluster Compute Resources

Cluster administrators can view the available and used compute resources of a node. As a cluster administrator, you can use the `oc describe node` command to determine the compute resource capacity of a node. The command shows the capacity of the node and how much of that capacity is allocatable. It also displays the amount of allocated and requested resources on the node.

```
[user@host ~]$ oc describe node master01
Name:               master01
Roles:              control-plane,master,worker
...output omitted...
Capacity:
  cpu:                  8
  ephemeral-storage:    125293548Ki
  hugepages-1Gi:        0
  hugepages-2Mi:        0
  memory:               20531668Ki
  pods:                 250
Allocatable:
  cpu:                  7500m
  ephemeral-storage:    114396791822
  hugepages-1Gi:        0
  hugepages-2Mi:        0
  memory:               19389692Ki
  pods:                 250
...output omitted...
Non-terminated Pods:                                        (88 in total)
  ... Name            CPU Requests  CPU Limits  Memory Requests  Memory Limits ...
  ... ----            ------------  ----------  ---------------  ------------- 
  ... controller-... 10m (0%)      0 (0%)      20Mi (0%)        0 (0%)        ...
  ... metallb-...    50m (0%)      0 (0%)      20Mi (0%)        0 (0%)        ...
  ... metallb-...    0 (0%)        0 (0%)      0 (0%)           0 (0%)        ...
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource            Requests       Limits
  --------            --------       ------
  cpu                 3183m (42%)    1202m (16%)
  memory              12717Mi (67%)  1350Mi (7%)
...output omitted...
```

RHOCP cluster administrators can also use the `oc adm top pods` command. This command shows the compute resource usage for each pod in a project. You must include the `--namespace` or `-n` options to specify a project. If you do not specify a project, then the command returns the resource usage for pods in the current project.

The following command displays the resource usage for pods in the `openshift-dns` project:

```
[user@host ~]$ oc adm top pods -n openshift-dns
NAME                  CPU(cores)   MEMORY(bytes)
dns-default-5kpn5     1m           33Mi
node-resolver-6kdxp   0m           2Mi
```

Additionally, cluster administrators can use the `oc adm top nodes` command to view the resource usage of cluster nodes. You can include the node name to view the resource usage of a particular node.

```
[user@host ~]$ oc adm top node master01
NAME       CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
master01   1250m        16%    10268Mi         54%
```

REFERENCES

For more information about pod scheduling, refer to the *Controlling Pod Placement onto Nodes (Scheduling)* chapter in the Red Hat OpenShift Container Platform 4.18 *Nodes* documentation at https://docs.redhat.com/en/documentation/openshift_container_platform/4.18/html-single/nodes/index#nodes-scheduler-about