Certified Kubernetes Administrator

Kubernetes: Scheduling

Learning Objectives

By the end of this lesson, you will be able to:

- Schedule a pod using the binding concept
- Work with labels and selectors in ReplicaSetController
- Create a pod with all resource requirements and limits
- Create a DaemonSet controller
- Work with Docker commands
- Create a custom scheduler in kubeadm
- Configure a pod with a custom scheduler

Manual Scheduling

Manual Scheduling Process

It is possible to schedule a pod manually, without using a scheduler, by specifying it in **spec.nodeName**. It can also be done using DaemonSet.

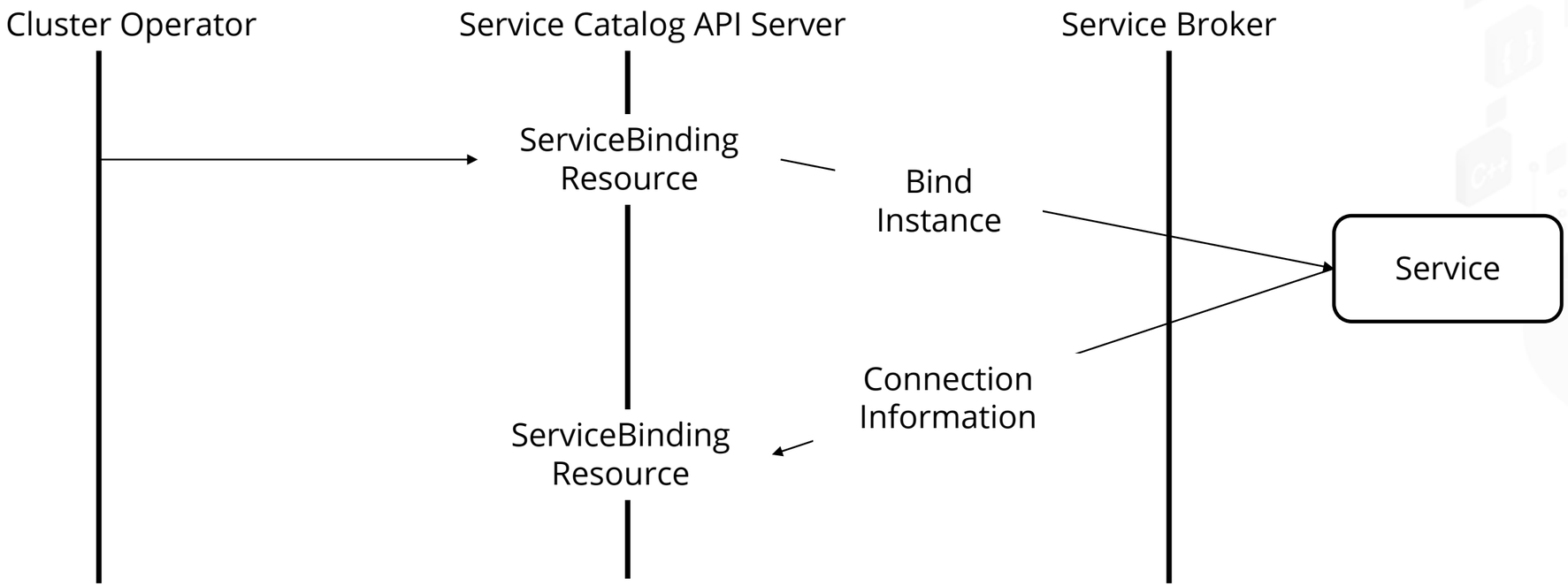
Below is the syntax or example to show how to schedule Pods manually:

```
[root@my-node1 ~] $ mkdir /etc/kubelet.d/
[root@my-node1 ~] $ cat <<EOF >/etc/kubelet.d/myapp.yaml
apiVersion: v1
kind: Pod
metadata:
  name: myapp
  labels:
    role: myrole
spec:
  name: <pod-name>
  containers:
    - name: <name>
      image: nginx
      ports:
        - name: <name>
          containerPort: 80
          protocol: TCP
```

Binding Concepts

Binding is the process in which the cluster operator is bound with the managed service to acquire the account details and connection credentials of the application.

It uses ServiceBinding resource which requests for all necessary information to bind the service instance.



Steps involved in binding to a managed service

Assisted Practice: Scheduling Pods Using Binding

Problem Statement: You are given a project to schedule pods using the binding concept.

ASSISTED PRACTICE

Labels and Selectors

Importance of Labels and Selectors

Labels are key-value pairs used to define attributes of an object.

They can be specified when modified values are created, if required.

Syntax:

```
"metadata": {
"labels": {
"key1" : "value1",
"key2" : "value2"
}
}
```

Importance of Labels and Selectors

Labels do not maintain uniqueness because of which it is difficult to identify objects.

Label selectors provide core grouping due through which users can identify objects.

Two types of selectors:

- Equity-based selector
- Set-based selector

Selectors in Service

A selector in a service is used to determine the set of pods targeted by a service.

Characteristics of services without selectors:

- Users can only have their own database in a test environment
- Users must point their service to a service in different namespaces
- Users can run only a proportion of backends while migrating a workload to Kubernetes

Assisted Practice: Working with Labels and Selectors in ReplicaSet

Problem Statement: You are given a project to work with labels and selectors in ReplicaSet.

ASSISTED PRACTICE

Assisted Practice: Working with Selectors in Service

Problem Statement: You are given a project to work with selectors in service.

ASSISTED PRACTICE

Resource Requirements and Limits

Resource Requirements and Limits

Here are the default requests and limits that are required to be set:

Parameter	CPU	Memory	Storage
Requests	0.1 Cores	None	None
Limits	None	None	None

Assisted Practice: Pod Creation

Problem Statement: You are given a project to create a pod with all resource requirements and limits.

ASSISTED PRACTICE

DaemonSets

DaemonSet

DaemonSet is used to check that each node contains a pod.

A pod is added to each node created in a cluster. If the node is deleted, pods are used as backup and are stored as garbage collectors.

Uses of DaemonSet:

- Running a storage cluster daemon on each node
- Running a log collection daemon on each node
- Running a node monitoring daemon on each node

DaemonSet vs. ReplicaSet

ReplicaSet	DaemonSet
It only checks if all pods are working irrespective of the worker node they are running on	It ensures that each node has one copy of pod that is running
Total number of pods running will be equal to total number of nodes	There can be more pods than the total number of nodes
If a node gets destroyed, then pods running on that node will also get destroyed	If a node gets destroyed, then pods that are running will be allocated to a different node

Use Cases of DaemonSets

Following are the use cases of DaemonSet:	
➢ Glusterd	➢ Collectd
➢ Ceph	➢ Dynatrace OneAgent
➢ Fluentd	➢ AppDynamics Agent
➢ Logstash	➢ Datadog Agent
➢ Prometheus Node Exporter	➢ New Relic Agent
➢ Sysdig Agent	➢ Ganglia
➢ Instana Agent	➢ Gmond

Assisted Practice: DaemonSet Creation

Problem Statement: You are given a project to create a DaemonSet.

ASSISTED PRACTICE

Static Pods

Static Pods

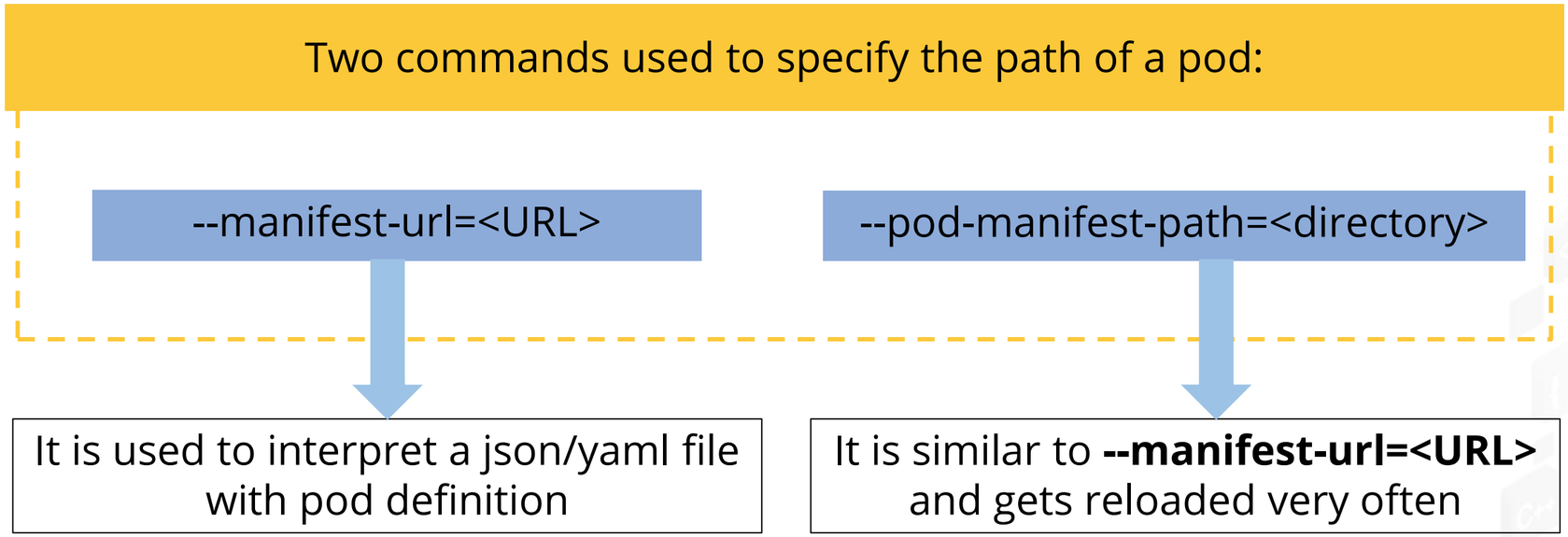
A static pod directly connects to the kubelet daemon and is not managed by kube-apiserver.

The kubelet daemon restarts the pod when it crashes, as it does not have any replication controller.

Characteristics of static pods:

- They can be created by either configuration files or HTTP
- Pods get started automatically when kubelet starts
- Pods cannot be controlled but are visible on the API server

Static Pod Path Location



Note:

--pod-manifest-path= <directory> or --pod-manifest-url=<URL>: This will start all pods defined in a directory. If the path specified as an argument is incorrect, then the pod will not start.

Assisted Practice: Working with Docker Commands

Problem Statement: You are given a project to work with Docker commands instead of kubectl commands in a static pod.

ASSISTED PRACTICE

Multiple Scheduler

Multiple Scheduler

Kubernetes may have the same cluster running different types of workloads, such as streaming, DAG batch, and traditional batch.

Each workload has to be scheduled in a different way; therefore, multi-scheduling is required.

Challenges in multi-scheduling:

- Separating the pods
- Dealing with conflicts

Multiple Scheduler

Few things to consider before designing a multiple scheduler:

- Add an annotation in the pod template
- Add a name to each scheduler
- Add a pod in a scheduler queue
- Deal with conflicts, if any

Assisted Practice: Multiple Scheduler

Problem Statement: You are given a project to demonstrate the workflow of multiple scheduler in Kubernetes.

ASSISTED PRACTICE

Assisted Practice: Custom Scheduler in Kubeadm

Problem Statement: You are given a project to create a custom scheduler in kubeadm.

ASSISTED PRACTICE

Configuring Kubernetes Scheduler

Why Build a Custom Scheduler?

- It is easy to implement
- It can be used for some special pods
- It gives you a better understanding of how a scheduler fits in the Kubernetes architecture

Assisted Practice: Configuring Pods

Problem Statement: You are given a project to configure a pod with custom scheduler.

ASSISTED PRACTICE

Key Takeaways

You are now able to:

- Schedule a pod using the binding concept
- Work with labels and selectors in ReplicaSetController
- Create a pod with all resource requirements and limits
- Create a DaemonSet controller
- Work with Docker commands
- Create a custom scheduler in kubeadm
- Configure a pod with a custom scheduler

Knowledge Check

Knowledge Check 1

Which of the following attributes can be updated to schedule a pod without using a scheduler?

a. metadata.nameNode

b. container.nameNode

c. spec.nameNode

d. port.nameNode

Knowledge Check

1

Which of the following attributes can be updated to schedule Pod without using a scheduler?

a. metadata.nameNode

b. container.nameNode

c. spec.nameNode

d. port.nameNode

The correct answer is **c**

spec.nameNode attribute can be updated to schedule a pod without using a scheduler.

Knowledge Check 2

Which of the following selectors allows filtering of keys according to a set of values?

a. Equity-based selector

b. Set-based selector

c. Value-based selector

d. Label-based selector

Knowledge Check 2

Which of the following selectors allows filtering of keys according to a set of values?

a. Equity-based selector

b. Set-based selector

c. Value-based selector

d. Label-based selector

The correct answer is **b**

Set-based selectors allow filtering of keys according to a set of values.

Knowledge Check

3

What is the default request for CPU?

a. 0.01 Cores

b. 0.1 Cores

c. 0.5 Cores

d. 1.0 Cores

Knowledge Check

3

What is the default request for CPU?

a. 0.01 Cores

b. 0.1 Cores

c. 0.5 Cores

d. 1.0 Cores

The correct answer is **b**

The default request for CPU is 0.1 cores.

Knowledge Check

4

Which of the following is NOT a use of DaemonSet?

a. It runs a storage cluster daemon on each node

b. It runs a node monitoring daemon on just one node

c. It runs a log collection daemon on every node

d. It runs a node monitoring daemon on each node

Knowledge Check 4

Which of the following is NOT a use of DaemonSet?

a. It runs a storage cluster daemon on each node

b. It runs a node monitoring daemon on just one node

c. It runs a log collection daemon on every node

d. It runs a node monitoring daemon on each node

The correct answer is **b**

DaemonSet runs a node monitoring daemon on each node and not just on one node.

Knowledge Check

5

Which of the following is NOT a use case of DaemonSet?

a. Logstash

b. Sysdig

c. Collectd

d. None of the above

Knowledge Check

5

Which of the following is NOT a use case of DaemonSet?

a. Logstash

b. Sysdig

c. Collectd

d. None of the above

The correct answer is **d**

Logstash, Sysdig, and Collectd are all use cases of DaemonSet.

Knowledge Check

6

Which of the following statements is NOT true about static pods?

a. They can be created either by configuration files or HTTP

b. Once kubelet is started, it manually starts the pods

c. Pods get started automatically when kubelet starts

d. Pods cannot be controlled but are visible on the API server

Knowledge Check

6

Which of the following statements is NOT true about static pods?

a. They can be created either by configuration files or HTTP

b. Once kubelet is started, it manually starts the pods

c. Pods get started automatically when kubelet starts

d. Pods cannot be controlled but are visible on the API server

The correct answer is **b**

kubelet does not start the pods manually. Pods start automatically once kubelet starts.

Knowledge Check 7

Which of the following changes cannot be made in the first step of designing a multiple scheduler?

a. Adding an annotation in the pod template

b. Dealing with conflicts

c. Adding a pod in the scheduler queue

d. Removing the scheduler name for each scheduler

Knowledge Check

7

Which of the following changes cannot be made in the first step of designing a multiple scheduler?

a. Adding an annotation in the pod template

b. Dealing with conflicts

c. Adding a pod in the scheduler queue

d. Removing the scheduler name for each scheduler

The correct answer is **d**

Removing the scheduler name for each scheduler cannot be done in the first step of designing a multiple scheduler.

Knowledge Check

8

Which of the following is NOT true about using a custom scheduler in Kubernetes?

a. It is easy to implement

b. It can be used for some special pods

c. It cannot be used on all pods

d. It gives you a better understanding of how a scheduler fits in Kubernetes architecture

Knowledge Check

8

Which of the following is NOT true about using a custom scheduler in Kubernetes?

a. It is easy to implement

b. It can be used for some special pods

c. It cannot be used on all pods

d. It gives you a better understanding of how a scheduler fits in Kubernetes architecture

The correct answer is **c**

Custom schedulers are easy to implement, can be used for some special pods, and give you a better understanding of how a scheduler fits in Kubernetes architecture.

Lesson-End Project

Problem Statement: BPO is a place where people work in shifts. Calls are assigned to them based on their shift. Let's consider a possibility where some people are absent during their shift. Being a Kubernetes expert, how can you design a system where calls are allocated to the people present during the shift dynamically?

Objective: Solve the task allocation problem based on shift allocation using multiple scheduler.

LESSON-END PROJECT