**Study Material: Creating and Managing Pods in Kubernetes**

**Introduction**

In this section, we’ll learn how to create and manage **Pods** in Kubernetes. Pods are the smallest deployable units in Kubernetes, and they can contain one or more containers. We’ll use the kubectl run command to create a Pod and explore its details.

**Step 1: Create a Pod Using kubectl run**

The kubectl run command is similar to the docker run command, but it creates a **Pod** instead of just a container. Let’s create a Pod running an Nginx web server.

**Create an Nginx Pod:**

1. Run the following command to create a Pod named nginx using the nginx Docker image:
2. kubectl run nginx --image=nginx
   * nginx: Name of the Pod.
   * --image=nginx: Specifies the Docker image to use (in this case, nginx).
3. You’ll see the output:
4. pod/nginx created

**Verify the Pod:**

1. List all Pods in the default namespace:
2. kubectl get pods

You’ll see the nginx Pod with a status of **ContainerCreating** initially, and then **Running** after a few seconds.

1. Describe the Pod to see detailed information:
2. kubectl describe pod nginx

You’ll see details like:

* + **Namespace**: default
  + **Node**: The node where the Pod is running.
  + **IP Address**: The internal IP address assigned to the Pod.
  + **Containers**: The container(s) running inside the Pod.

**Step 2: Explore the Pod Internals**

Let’s explore what happens inside the Kubernetes node when a Pod is created.

**SSH into the Minikube Node:**

1. Get the IP address of the Minikube node:
2. minikube ip
3. SSH into the node:
4. ssh docker@<minikube-ip>

Use the password: **tcuser**.

**List Docker Containers:**

1. Inside the Minikube node, list all running Docker containers:
2. docker ps

You’ll see two containers related to the nginx Pod:

* + **k8s\_nginx\_nginx\_default**: The Nginx container.
  + **k8s\_POD\_nginx\_default**: The **Pause container**, which manages the Pod’s namespaces.

**Connect to the Nginx Container:**

1. Connect to the Nginx container:
2. docker exec -it <container-id> sh

Replace <container-id> with the ID of the k8s\_nginx\_nginx\_default container.

1. Inside the container, check the hostname and IP address:
2. hostname
3. ifconfig

You’ll see the Pod’s internal IP address.

1. Test the Nginx web server:
2. curl http://localhost

You’ll see the default Nginx welcome page.

1. Exit the container:
2. exit

**Step 3: Access the Pod from Outside**

By default, Pods have internal IP addresses that are not accessible from outside the cluster. To expose a Pod to the outside world, you need to create a **Service**. We’ll cover Services in the next section.

**Step 4: Delete the Pod**

Let’s delete the nginx Pod to clean up.

**Delete the Pod:**

1. Delete the Pod:
2. kubectl delete pod nginx

You’ll see the output:

pod "nginx" deleted

1. Verify that the Pod is deleted:
2. kubectl get pods

You’ll see: **No resources found in default namespace.**

**Step 5: Create an Alias for kubectl**

Typing kubectl repeatedly can be tedious. Let’s create an alias to shorten it to k.

**Create an Alias:**

1. For Linux/macOS users:
2. alias k="kubectl"
3. For Windows users using Git Bash:
   * Open Git Bash.
   * Run the same command:
   * alias k="kubectl"
4. Test the alias:
5. k get pods

You’ll see the same output as kubectl get pods.

**Make the Alias Permanent (Optional):**

To make the alias permanent, add it to your shell configuration file:

* For **bash**: Add the alias to ~/.bashrc or ~/.bash\_profile.
* For **zsh**: Add the alias to ~/.zshrc.

Example:

echo 'alias k="kubectl"' >> ~/.bashrc

source ~/.bashrc

**Key Takeaways**

* **Pods** are the smallest deployable units in Kubernetes.
* Use kubectl run to create a Pod.
* Pods have internal IP addresses and are not accessible from outside the cluster by default.
* Use kubectl describe pod to get detailed information about a Pod.
* Create an alias (k) for kubectl to save time.

**Next Steps**

Now that you’ve created and explored a Pod, you can:

* Create **Deployments** to manage multiple Pods.
* Expose Pods to the outside world using **Services**.
* Scale applications by increasing the number of Pods.

**Commands Summary**

| **Command** | **Description** |
| --- | --- |
| kubectl run nginx --image=nginx | Create a Pod named nginx using the nginx image. |
| kubectl get pods | List all Pods in the default namespace. |
| kubectl describe pod nginx | Show detailed information about the nginx Pod. |
| kubectl delete pod nginx | Delete the nginx Pod. |
| alias k="kubectl" | Create an alias for kubectl. |

**Troubleshooting**

* **Pod stuck in ContainerCreating**: Check if the Docker image is being pulled. Use kubectl describe pod to see the logs.
* **Cannot connect to Pod**: Pods have internal IPs. Use a **Service** to expose them externally.

With this knowledge, you’re ready to start deploying and managing applications in Kubernetes! 🚀

### Study Material: Creating and Managing Pods in Kubernetes

---

#### \*\*Introduction\*\*

In this section, we’ll learn how to create and manage \*\*Pods\*\* in Kubernetes. Pods are the smallest deployable units in Kubernetes, and they can contain one or more containers. We’ll use the `kubectl run` command to create a Pod and explore its details.

---

### \*\*Step 1: Create a Pod Using `kubectl run`\*\*

The `kubectl run` command is similar to the `docker run` command, but it creates a \*\*Pod\*\* instead of just a container. Let’s create a Pod running an Nginx web server.

---

#### \*\*Create an Nginx Pod:\*\*

1. Run the following command to create a Pod named `nginx` using the `nginx` Docker image:

```bash

kubectl run nginx --image=nginx

```

- `nginx`: Name of the Pod.

- `--image=nginx`: Specifies the Docker image to use (in this case, `nginx`).

2. You’ll see the output:

```

pod/nginx created

```

---

#### \*\*Verify the Pod:\*\*

1. List all Pods in the default namespace:

```bash

kubectl get pods

```

You’ll see the `nginx` Pod with a status of \*\*ContainerCreating\*\* initially, and then \*\*Running\*\* after a few seconds.

2. Describe the Pod to see detailed information:

```bash

kubectl describe pod nginx

```

You’ll see details like:

- \*\*Namespace\*\*: `default`

- \*\*Node\*\*: The node where the Pod is running.

- \*\*IP Address\*\*: The internal IP address assigned to the Pod.

- \*\*Containers\*\*: The container(s) running inside the Pod.

---

### \*\*Step 2: Explore the Pod Internals\*\*

Let’s explore what happens inside the Kubernetes node when a Pod is created.

---

#### \*\*SSH into the Minikube Node:\*\*

1. Get the IP address of the Minikube node:

```bash

minikube ip

```

2. SSH into the node:

```bash

ssh docker@<minikube-ip>

```

Use the password: \*\*tcuser\*\*.

---

#### \*\*List Docker Containers:\*\*

1. Inside the Minikube node, list all running Docker containers:

```bash

docker ps

```

You’ll see two containers related to the `nginx` Pod:

- \*\*k8s\_nginx\_nginx\_default\*\*: The Nginx container.

- \*\*k8s\_POD\_nginx\_default\*\*: The \*\*Pause container\*\*, which manages the Pod’s namespaces.

---

#### \*\*Connect to the Nginx Container:\*\*

1. Connect to the Nginx container:

```bash

docker exec -it <container-id> sh

```

Replace `<container-id>` with the ID of the `k8s\_nginx\_nginx\_default` container.

2. Inside the container, check the hostname and IP address:

```bash

hostname

ifconfig

```

You’ll see the Pod’s internal IP address.

3. Test the Nginx web server:

```bash

curl http://localhost

```

You’ll see the default Nginx welcome page.

4. Exit the container:

```bash

exit

```

---

### \*\*Step 3: Access the Pod from Outside\*\*

By default, Pods have internal IP addresses that are not accessible from outside the cluster. To expose a Pod to the outside world, you need to create a \*\*Service\*\*. We’ll cover Services in the next section.

---

### \*\*Step 4: Delete the Pod\*\*

Let’s delete the `nginx` Pod to clean up.

---

#### \*\*Delete the Pod:\*\*

1. Delete the Pod:

```bash

kubectl delete pod nginx

```

You’ll see the output:

```

pod "nginx" deleted

```

2. Verify that the Pod is deleted:

```bash

kubectl get pods

```

You’ll see: \*\*No resources found in default namespace.\*\*

---

### \*\*Step 5: Create an Alias for `kubectl`\*\*

Typing `kubectl` repeatedly can be tedious. Let’s create an alias to shorten it to `k`.

---

#### \*\*Create an Alias:\*\*

1. For Linux/macOS users:

```bash

alias k="kubectl"

```

2. For Windows users using Git Bash:

- Open Git Bash.

- Run the same command:

```bash

alias k="kubectl"

```

3. Test the alias:

```bash

k get pods

```

You’ll see the same output as `kubectl get pods`.

---

#### \*\*Make the Alias Permanent (Optional):\*\*

To make the alias permanent, add it to your shell configuration file:

- For \*\*bash\*\*: Add the alias to `~/.bashrc` or `~/.bash\_profile`.

- For \*\*zsh\*\*: Add the alias to `~/.zshrc`.

Example:

```bash

echo 'alias k="kubectl"' >> ~/.bashrc

source ~/.bashrc

```

---

### \*\*Key Takeaways\*\*

- \*\*Pods\*\* are the smallest deployable units in Kubernetes.

- Use `kubectl run` to create a Pod.

- Pods have internal IP addresses and are not accessible from outside the cluster by default.

- Use `kubectl describe pod` to get detailed information about a Pod.

- Create an alias (`k`) for `kubectl` to save time.

---

### \*\*Next Steps\*\*

Now that you’ve created and explored a Pod, you can:

- Create \*\*Deployments\*\* to manage multiple Pods.

- Expose Pods to the outside world using \*\*Services\*\*.

- Scale applications by increasing the number of Pods.

---

### \*\*Commands Summary\*\*

| \*\*Command\*\* | \*\*Description\*\* |

|--------------------------------------|------------------------------------------------------|

| `kubectl run nginx --image=nginx` | Create a Pod named `nginx` using the `nginx` image. |

| `kubectl get pods` | List all Pods in the default namespace. |

| `kubectl describe pod nginx` | Show detailed information about the `nginx` Pod. |

| `kubectl delete pod nginx` | Delete the `nginx` Pod. |

| `alias k="kubectl"` | Create an alias for `kubectl`. |

---

### \*\*Troubleshooting\*\*

- \*\*Pod stuck in ContainerCreating\*\*: Check if the Docker image is being pulled. Use `kubectl describe pod` to see the logs.

- \*\*Cannot connect to Pod\*\*: Pods have internal IPs. Use a \*\*Service\*\* to expose them externally.

---

With this knowledge, you’re ready to start deploying and managing applications in Kubernetes! 🚀