**Study Guide: Creating and Using Services in Kubernetes**

**Introduction**

In Kubernetes, a **Service** is an abstraction that enables network access to applications running in Pods. Services provide a stable endpoint for communication, ensuring that even if individual Pods are replaced, the application remains accessible. Kubernetes offers different types of Services, such as **ClusterIP**, **NodePort**, and **LoadBalancer**. This guide focuses on creating a **ClusterIP** Service, which provides internal communication within the cluster.

**Step 1: Creating a ClusterIP Service**

A **ClusterIP** Service assigns a virtual IP address, allowing communication between Pods within the cluster.

**Expose a Deployment Using ClusterIP**

1. Ensure you have an existing Deployment. For example, create an Nginx Deployment:
2. kubectl create deployment nginx-deployment --image=nginx
3. Create a ClusterIP Service to expose the Deployment:
4. kubectl expose deployment nginx-deployment --port=88 --target-port=80
   * --port=88: The Service port clients will connect to.
   * --target-port=80: The port on which the Nginx container is listening.
5. Verify that the Service was created:
6. kubectl get services

The output will display a Service named nginx-deployment with a **ClusterIP** and an IP address like 10.96.x.x.

**Step 2: Exploring Service Details**

Understanding the Service properties helps in debugging and managing Kubernetes networking.

**View Service Details**

1. Describe the Service to inspect its properties:
2. kubectl describe service nginx-deployment

This command provides information such as:

* + **ClusterIP**: The virtual IP assigned to the Service.
  + **Ports**: The mapped ports for communication.
  + **Endpoints**: The Pods assigned to the Service.

**Step 3: Accessing the Service**

A **ClusterIP** Service is only accessible within the Kubernetes cluster.

**Access the Service from Within the Cluster**

1. SSH into the Minikube node (if using Minikube):
2. minikube ssh
3. Retrieve the ClusterIP of the Service:
4. kubectl get services
5. Use curl to test connectivity:
6. curl <cluster-ip>:88

This should return the Nginx welcome page.

1. Exit the SSH session:
2. exit

**Access the Service from Outside the Cluster**

Attempting to access the ClusterIP Service from your local machine:

curl <cluster-ip>:88

This will fail because ClusterIP Services are not externally accessible. To expose a Service externally, consider using **NodePort** or **LoadBalancer** types.

**Key Takeaways**

* **ClusterIP Services** facilitate internal communication within a Kubernetes cluster.
* They provide a stable IP address for connecting multiple Pods.
* Use kubectl expose to create a Service for a Deployment.
* **ClusterIP Services are not accessible from outside the cluster.** Use **NodePort** or **LoadBalancer** for external access.

**Command Summary**

| **Command** | **Description** |
| --- | --- |
| kubectl expose deployment <name> --port=<p1> --target-port=<p2> | Creates a ClusterIP Service for a Deployment. |
| kubectl get services | Lists all Services. |
| kubectl describe service <name> | Displays detailed information about a Service. |
| minikube ssh | SSH into the Minikube node. |
| curl <cluster-ip>:<port> | Tests connectivity to a Service from within the cluster. |

**Next Steps**

With a basic understanding of ClusterIP Services, you can now:

* Deploy a **NodePort** Service to expose applications externally.
* Use a **LoadBalancer** Service in cloud environments.
* Implement an **Ingress Controller** for advanced traffic routing.

This guide equips you with the fundamental knowledge to manage Services in Kubernetes effectively. 🚀

### Study Material: Creating and Using Services in Kubernetes

---

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

In Kubernetes, a \*\*Service\*\* is an abstraction that exposes your application (running in Pods) to the network. Services allow you to connect to your application using a single IP address, even if it’s running across multiple Pods. There are different types of Services, such as \*\*ClusterIP\*\*, \*\*NodePort\*\*, and \*\*LoadBalancer\*\*. In this section, we’ll focus on creating a \*\*ClusterIP\*\* Service.

---

### \*\*Step 1: Create a ClusterIP Service\*\*

A \*\*ClusterIP\*\* Service exposes your application internally within the Kubernetes cluster. It assigns a virtual IP address that can be used to access your application from inside the cluster.

---

#### \*\*Expose a Deployment Using ClusterIP:\*\*

1. Ensure you have a Deployment running. For example, let’s assume you have an Nginx Deployment:

```bash

kubectl create deployment nginx-deployment --image=nginx

```

2. Expose the Deployment using a ClusterIP Service:

```bash

kubectl expose deployment nginx-deployment --port=88 --target-port=80

```

- `--port=88`: The external port exposed by the Service.

- `--target-port=80`: The internal port where the application is running (Nginx runs on port 80 by default).

3. Verify the Service:

```bash

kubectl get services

```

You’ll see a Service named `nginx-deployment` with a \*\*ClusterIP\*\* type and an IP address like `10.96.x.x`.

---

### \*\*Step 2: Explore the Service Details\*\*

Let’s explore the details of the Service to understand how it works.

---

#### \*\*Describe the Service:\*\*

1. Describe the Service to see detailed information:

```bash

kubectl describe service nginx-deployment

```

You’ll see:

- \*\*ClusterIP\*\*: The virtual IP address assigned to the Service.

- \*\*Port\*\*: The external port (e.g., `88`).

- \*\*TargetPort\*\*: The internal port (e.g., `80`).

- \*\*Endpoints\*\*: The IP addresses of the Pods managed by the Service.

---

### \*\*Step 3: Access the Service\*\*

A \*\*ClusterIP\*\* Service is only accessible from within the Kubernetes cluster. Let’s test connectivity.

---

#### \*\*Access the Service from Inside the Cluster:\*\*

1. SSH into the Minikube node:

```bash

minikube ssh

```

2. Use `curl` to connect to the Service’s ClusterIP and port:

```bash

curl <cluster-ip>:88

```

You’ll see the Nginx welcome page.

3. Exit the SSH session:

```bash

exit

```

---

#### \*\*Access the Service from Outside the Cluster:\*\*

1. Try accessing the Service from your local machine:

```bash

curl <cluster-ip>:88

```

You’ll see \*\*no response\*\* because ClusterIP Services are not accessible from outside the cluster.

---

### \*\*Step 4: Key Takeaways\*\*

- \*\*ClusterIP\*\* Services expose your application internally within the Kubernetes cluster.

- They provide a single virtual IP address to access multiple Pods.

- Use `kubectl expose` to create a Service for a Deployment.

- ClusterIP Services are not accessible from outside the cluster. For external access, use \*\*NodePort\*\* or \*\*LoadBalancer\*\* Services.

---

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

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

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

| `kubectl expose deployment <name> --port=<p1> --target-port=<p2>` | Create a ClusterIP Service for a Deployment. |

| `kubectl get services` | List all Services. |

| `kubectl describe service <name>` | Show detailed information about a Service. |

| `minikube ssh` | SSH into the Minikube node. |

| `curl <cluster-ip>:<port>` | Test connectivity to a Service from inside the cluster. |

---

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

Now that you’ve created a ClusterIP Service, you can:

- Create a \*\*NodePort\*\* Service to expose your application externally.

- Use a \*\*LoadBalancer\*\* Service for cloud environments.

- Explore advanced Service configurations like \*\*Ingress\*\*.

---

With this knowledge, you’re ready to expose and manage your applications in Kubernetes! 🚀