Kubectl Commands Documentation

May 2025

Contents

1 Introduction

This document provides a comprehensive reference for commonly used `kubectl` commands in Kubernetes. `kubectl` is the command-line tool for interacting with Kubernetes clusters. The commands are organized by category, with descriptions and examples to facilitate usage. This guide is intended for administrators, developers, and operators managing Kubernetes clusters.

2 Getting Started

2.1 Installing kubectl

To use `kubectl`, it must be installed on your system. Below is an example command to install `kubectl` on a Linux system using `curl`:

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/
    release/stable.txt)/bin/linux/amd64/kubectl" && chmod +x kubectl &&
    sudo mv kubectl /usr/local/bin/
```

2.2 Configuring kubectl

Configure `kubectl` to communicate with your cluster by setting up the kubeconfig file, typically located at `~/.kube/config`. Example:

```
kubectl config set-context --current --namespace=default
```

3 Cluster Management Commands

3.1 Cluster Information

View cluster details, such as the API server endpoint and cluster version:

```
kubectl cluster-info
```

3.2 Viewing Nodes

List all nodes in the cluster:

```
kubectl get nodes
```

3.3 Describing a Node

Get detailed information about a specific node:

```
kubectl describe node <node-name>
```

4 Resource Management Commands

4.1 Creating Resources

Create resources from a YAML or JSON file:

```
kubectl apply -f <filename>.yaml
```

4.2 Listing Resources

List resources like pods, services, or deployments:

```
kubectl get pods
kubectl get services
kubectl get deployments
```

4.3 Deleting Resources

Delete a specific resource:

```
kubectl delete pod <pod-name>
```

4.4 Scaling Deployments

Scale a deployment to a specified number of replicas:

```
kubectl scale deployment <deployment-name> --replicas=3
```

5 Pod and Container Management

5.1 Viewing Pod Details

Describe a pod to view its details, events, and status:

```
kubectl describe pod <pod-name>
```

5.2 Accessing Pod Logs

View logs for a pod:

```
kubectl logs <pod-name>
```

5.3 Executing Commands in a Pod

Run a command inside a pod's container:

```
kubectl exec -it <pod-name> -- /bin/bash
```

6 Namespace Management

6.1 Listing Namespaces

List all namespaces in the cluster:

```
kubectl get namespaces
```

6.2 Creating a Namespace

Create a new namespace:

```
kubectl create namespace <namespace-name>
```

7 Configuration and Secret Management

7.1 Creating a ConfigMap

Create a ConfigMap from a file or literal value:

```
kubectl create configmap <configmap-name> --from-literal=key=value
```

7.2 Creating a Secret

Create a Secret for sensitive data:

```
kubectl create secret generic <secret-name> --from-literal=password=
    secret
```

8 Troubleshooting and Debugging

8.1 Checking Events

View cluster events for troubleshooting:

```
kubectl get events
```

8.2 Debugging with Describe

Use `describe` to debug resource issues:

```
kubectl describe <resource-type> <resource-name>
```

8.3 Port Forwarding

Forward a local port to a pod:

```
kubectl port-forward <pod-name> 8080:80
```

9 Advanced Commands

9.1 Rolling Update

Perform a rolling update on a deployment:

```
kubectl set image deployment/<deployment-name> <container-name>=<new-
    image>
```

9.2 Tainting Nodes

Apply a taint to a node:

```
kubectl taint nodes <node-name> key=value:NoSchedule
```

9.3 Annotating Resources

Add annotations to a resource:

```
kubectl annotate pod <pod-name> description='example annotation'
```

10 Conclusion

This document covers the most commonly used `kubectl` commands for managing Kubernetes clusters. For more detailed information, refer to the official Kubernetes documentation at `https://kubernetes.io/docs/reference/kubectl/`. Always ensure you have the correct permissions and context configured before executing commands.