kubectl Commands Reference Guide

Complete guide to Kubernetes command-line tool

Generated: July 2025

Table of Contents

1. Introduction to kubectl

kubectl is the command-line interface for running commands against Kubernetes clusters. It allows you to deploy applications, inspect and manage cluster resources, and view logs.

> **Tip:** kubectl communicates with the Kubernetes API server to perform operations on your cluster.

Installation Verification

```
kubectl version --client
```

Displays the kubectl client version installed on your system.

2. Basic Commands

Cluster Information

```
kubectl cluster-info
```

Displays cluster information including master and services URLs.

```
kubectl get nodes
```

Lists all nodes in the cluster with their status.

```
kubectl get namespaces
```

Shows all available namespaces in the cluster.

Resource Listing

```
kubectl get all
```

Lists all resources in the current namespace.

```
kubectl get all --all-namespaces
```

Lists all resources across all namespaces.

3. Resource Management

Creating Resources

> **`kubectl create -f <filename>`**

> Creates resources from a YAML or JSON file.

> **`kubectl apply -f <filename>`**

> Applies configuration changes to resources. Creates if doesn't exist, updates if exists.

```
kubectl apply -f deployment.yaml
kubectl apply -f https://example.com/config.yaml
```

Deleting Resources

> **`kubectl delete -f <filename>`**

> Deletes resources defined in the specified file.

> **`kubectl delete <resource-type> <resource-name>`**

> Deletes a specific resource by type and name.

```
kubectl delete pod my-pod
kubectl delete service my-service
kubectl delete deployment my-deployment
```

Describing Resources

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

Shows detailed information about a specific resource.

```
kubectl get <resource-type> -o yaml
```

Displays resource information in YAML format.

4. Pod Operations

Pod Management

```
kubectl get pods
```

Lists all pods in the current namespace.

```
kubectl get pods -o wide
```

Lists pods with additional information including node placement and IP addresses.

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

Displays logs from a specific pod.

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

Follows (streams) logs from a pod in real-time.

Pod Interaction

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

Opens an interactive shell session inside a pod.

```
kubectl exec <pod-name> -- <command>
```

Executes a command inside a pod without opening a shell.

```
kubectl exec my-pod -- ls -la
kubectl exec my-pod -- cat /etc/hostname
```

Port Forwarding

```
kubectl port-forward <pod-name> <local-port>:<pod-port>
```

Forwards a local port to a port on a pod for testing purposes.

```
kubectl port-forward my-pod 8080:80
```

5. Service Management

Service Operations

```
kubectl get services
```

Lists all services in the current namespace.

```
kubectl expose deployment <deployment-name> --port=<port>
--type=<service-type>
```

Creates a service to expose a deployment.

```
kubectl expose deployment nginx --port=80 --type=LoadBalancer
kubectl expose deployment api --port=3000 --type=ClusterIP
```

Service Types

Service Type	Purpose	Access
ClusterIP	Internal cluster communication	Cluster-internal only
NodePort	External access via node IP	External on specific port
LoadBalancer	External access with load balancing	External with cloud LB

6. Configuration & Context

Context Management

```
kubectl config get-contexts
```

Lists all available contexts (cluster configurations).

```
kubectl config current-context
```

Shows the currently active context.

```
kubectl config use-context <context-name>
```

Switches to a different context.

Namespace Management

```
kubectl config set-context --current --
namespace=<namespace>
```

Sets the default namespace for the current context.

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

Creates a new namespace.

7. Troubleshooting Commands

Debugging Pods

```
kubectl get events
```

Shows recent events in the current namespace, useful for debugging.

```
kubectl get events --sort-by=.metadata.creationTimestamp
```

Shows events sorted by creation time.

```
kubectl top nodes
```

Shows resource usage (CPU/Memory) for nodes.

```
kubectl top pods
```

Shows resource usage for pods.

Resource Status

```
kubectl get pods --field-selector=status.phase=Failed
```

Lists only failed pods.

```
kubectl get pods --field-selector=status.phase=Pending
```

> Lists pods that are stuck in pending state.

> **Warning:** Always check events and logs when troubleshooting pod issues. They often contain crucial error information.

8. Advanced Operations

Scaling

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

Scales a deployment to the specified number of replicas.

```
kubectl scale deployment nginx --replicas=5
```

Rolling Updates

```
kubectl rollout status deployment/<deployment-name>
```

Shows the status of a deployment rollout.

```
kubectl rollout history deployment/<deployment-name>
```

Shows rollout history for a deployment.

```
kubectl rollout undo deployment/<deployment-name>
```

Rolls back a deployment to the previous revision.

Resource Quotas and Limits

```
kubectl describe quota
```

Shows resource quotas for the current namespace.

```
kubectl describe limits
```

Shows resource limits for the current namespace.

9. Common Flags & Options

Flag	Purpose	Example
-n, --namespace	Specify namespace	kubectl get pods -n kube-system
-o, --output	Output format	kubectl get pods -o yaml
-w, --watch	Watch for changes	kubectl get pods -w
--all-namespaces	All namespaces	kubectl get pods --all-namespaces
--dry-run=client	Test without executing	kubectl create deployment test --dry-run=client
-f, --filename	Specify file	kubectl apply -f deployment.yaml
--force	Force operation	kubectl delete pod --force

Output Formats

```
-o json       # JSON format
-o yaml       # YAML format
-o wide       # Additional columns
-o name       # Only resource names
-o jsonpath   # Custom output using JSONPath
```

10. Best Practices

Safety Practices

> **Always verify your context:** Run `kubectl config current-context` before executing commands to ensure you're working on the correct cluster.

> **Use dry-run:** Test commands with `--dry-run=client` flag before actual execution.

> **Be cautious with:** `kubectl delete` commands, especially with `--all` or `--force` flags.

Efficiency Tips

- Use aliases for frequently used commands
- Set up shell autocompletion for kubectl
- Use labels and selectors for bulk operations
- Monitor resource usage regularly
- Keep YAML manifests in version control

Common Aliases

```
alias k=kubectl
alias kgp='kubectl get pods'
alias kgs='kubectl get services'
alias kgd='kubectl get deployments'
alias kdp='kubectl describe pod'
alias kl='kubectl logs'
```

Autocompletion Setup

```
# Bash
echo 'source <(kubectl completion bash)' >> ~/.bashrc

# Zsh
echo 'source <(kubectl completion zsh)' >> ~/.zshrc
```

This guide covers the most commonly used kubectl commands. For complete documentation, visit the official Kubernetes documentation.

Remember: Always test commands in a development environment before using them in production.