- Documentation
- Blog
- Partners
- Community
- Case Studies
- English
 - o Chinese
- v1.11
 - o <u>v1.11</u>
 - o <u>v1.10</u>
 - o <u>v1.9</u> o v1.8
 - o <u>v1.7</u>

Try Kubernetes

Get Started

Ready to get your hands dirty? Build a simple Kubernetes cluster that runs "Hello World" for Node.js.

Documentation

Learn how to use Kubernetes with the use of walkthroughs, samples, and reference documentation. You can even help contribute to the docs!

Community

If you need help, you can connect with other Kubernetes users and the Kubernetes authors, attend community events, and watch video presentations from around the web.

Blog

Read the latest news for Kubernetes and the containers space in general, and get technical how-tos hot off the presses.

Interested in hacking on the core Kubernetes code base?

View On Github

Explore the community

Twitter Github Slack Stack Overflow Forum Events Calendar

Tasks

- HOME
- SETUP
- CONCEPTS
- TASKS
- TUTORIALS
- REFERENCE

• CONTRIBUTE

Search

Tasks

Install Tools

Contailgured Poets Lands Contrainers

Adstrain Members and Pods

Ardendin Ratati Rites with plicable administration and Pods

Nicht Strate Constitution of the Constitution

Red Company Co

Welaffen Medical Control of the Control of the Residential Volume Cluster

Philipped Stability Stability Stability Stability Special Special Special Stability Special Stability Special Sp

Plantfactive Mailiague Apudates Risa Alpheli Bellebiolaitican Constroller

Asnifred Month School Black End Using a Service

Continue | Photel Office Resistant de la Continue of h

Spacifiyira Chilets Status APBiothie Life in wheat Are plication

1 of 6 09/09/18, 4:52 PM Cish Addl Cart Bloke hald langer (Frofigiles im a led us bele

Shater Pizicres De Brance picte rite travers ni Colme a Sagre Prod Posing a Shared Volume

Debrisdatie a DDRS Robe & Column File to Kubernetes Resources

Meditor, Netwarld Hoelbug

Exterior Knub Crimente Scotton ller d Melangering

Wisdi Consitrom Silverson at Rest

faltibed Britinidie Approximately 10 in Mylittiale (Glob River South) Definitions

Manage Cliffet Darmons of Charles Federated Services

Histall Service Catalog at 1 a Child Catalog at 1 a

Federation Remark Applicant Multiple Clisters Federation

Bellevicial abbiding Ederation

Dieducate de la Charle de la C

Relier filted Resident Kobeled i Faal Live Cluster

PaletapedGDandokaBenstyingeseftoxiCessteralDaemons
BedettaDadSBsackdorderwhile Respecting Application SLOs

Bedta Tallac PH Goriz Court ali Rg C Anteix celers (HPA)

Eet divalue destingnées a settiers e a rice la accodifici b false a

Confidence Access to Multiple Clusters

Stalling All and All Repition in Sept 18 esources

his page shows how to configure access to multiple clusters by using configuration files. After your clusters, wsers, lend-contexts are defined in one on more configuration files, you can quickly switch between clusters by using thankulestimentie view oftexticemmendation

Using sysctls in a Kubernetes Cluster

Note: A file that is used to configure access to a cluster is sometimes called a kubeconfig file. This is a generic way of referring to configuration files. It does not mean that there is a file named kubeconfig.

- Before you begin
- · Define clusters, users, and contexts
- Create a second configuration file
- Set the KUBECONFIG environment variable
- Explore the \$HOME/.kube directory
- Append \$HOME/.kube/config to your KUBECONFIG environment variable
- Clean up
- What's next

Before you begin

You need to have a Kubernetes cluster, and the kubectl command-line tool must be configured to communicate with your cluster. If you do not already have a cluster, you can create one by using Minikube, or you can use one of these Kubernetes playgrounds:

- Katacoda
- Play with Kubernetes

To check the version, enter kubectl version.

Define clusters, users, and contexts

Suppose you have two clusters, one for development work and one for scratch work. In the development cluster, your frontend developers work in a namespace called frontend, and your storage developers work in a namespace called storage. In your scratch cluster, developers work in the default namespace, or they create auxiliary namespaces as they see fit. Access to the development cluster requires authentication by certificate. Access to the scratch cluster requires authentication by username and password.

Create a directory named config-exercise. In your config-exercise directory, create a file named config-demo with this content:

apiVersion: v1 kind: Config preferences: {} clusters: cluster: name: development cluster: name: scratch users: name: developer - name: experimenter contexts: context: name: dev-frontend

2 of 6 09/09/18, 4:52 PM

```
- context:
  name: dev-storage
- context:
  name: exp-scratch
```

A configuration file describes clusters, users, and contexts. Your config-demo file has the framework to describe two clusters, two users, and three contexts.

Go to your config-exercise directory. Enter these commands to add cluster details to your configuration file:

```
kubectl config --kubeconfig=config-demo set-cluster development --server=https://1.2.3.4 --certificate-authority=fake-ca-file kubectl config --kubeconfig=config-demo set-cluster scratch --server=https://5.6.7.8 --insecure-skip-tls-verify
```

Add user details to your configuration file:

```
kubectl\ config\ --kubeconfig=config-demo\ set-credentials\ developer\ --client-certificate=fake-cert-file\ --client-key=fake-key-seefile\ kubectl\ config\ --kubeconfig=config-demo\ set-credentials\ experimenter\ --username=exp\ --password=some-password
```

Add context details to your configuration file:

```
kubectl config --kubeconfig=config-demo set-context dev-frontend --cluster=development --namespace=frontend --user=developer kubectl config --kubeconfig=config-demo set-context dev-storage --cluster=development --namespace=storage --user=developer kubectl config --kubeconfig=config-demo set-context exp-scratch --cluster=scratch --namespace=default --user=experimenter
```

Open your config-demo file to see the added details. As an alternative to opening the config-demo file, you can use the config view command.

kubectl config --kubeconfig=config-demo view

The output shows the two clusters, two users, and three contexts:

```
apiVersion: v1
clusters:
- cluster:
   certificate-authority: fake-ca-file
    server: https://1.2.3.4
  name: development
 cluster:
    insecure-skip-tls-verify: true
   server: https://5.6.7.8
  name: scratch
contexts:
- context:
   cluster: development
   namespace: frontend
   user: developer
  name: dev-frontend
- context:
    cluster: development
    namespace: storage
   user: developer
  name: dev-storage
 context:
    cluster: scratch
   namespace: default
    user: experimenter
name: exp-scratch
current-context: ""
kind: Config
preferences:
users:
 name: developer
 user:
   client-certificate: fake-cert-file
    client-key: fake-key-file
- name: experimenter
  user:
    password: some-password
   username: exp
```

Each context is a triple (cluster, user, namespace). For example, the dev-frontend context says, Use the credentials of the developer user to access the frontend namespace of the development cluster.

Set the current context:

```
kubectl config --kubeconfig=config-demo use-context dev-frontend
```

Now whenever you enter a kubectl command, the action will apply to the cluster, and namespace listed in the dev-frontend context. And the command will use the credentials of the user listed in the dev-frontend context.

To see only the configuration information associated with the current context, use the --minify flag.

```
{\tt kubectl\ config\ --kubeconfig=config-demo\ view\ --minify}
```

The output shows configuration information associated with the dev-frontend context:

```
apiVersion: v1
clusters:
- cluster:
```

3 of 6 09/09/18, 4:52 PM

```
certificate-authority: fake-ca-file
    server: https://1.2.3.4
  name: development
contexts:
 context:
    cluster: development
   namespace: frontend
   user: developer
  name: dev-frontend
current-context: dev-frontend
kind: Config
preferences: {}
users:
 name: developer
  user:
    client-certificate: fake-cert-file
    client-key: fake-key-file
```

Now suppose you want to work for a while in the scratch cluster.

Change the current context to exp-scratch:

```
kubectl config --kubeconfig=config-demo use-context exp-scratch
```

Now any kubectl command you give will apply to the default namespace of the scratch cluster. And the command will use the credentials of the user listed in the exp-scratch context.

View configuration associated with the new current context, exp-scratch.

```
kubectl config --kubeconfig=config-demo view --minify
```

Finally, suppose you want to work for a while in the storage namespace of the development cluster.

Change the current context to dev-storage:

```
kubectl config --kubeconfig=config-demo use-context dev-storage
```

View configuration associated with the new current context, dev-storage.

```
kubectl config --kubeconfig=config-demo view --minify
```

Create a second configuration file

In your config-exercise directory, create a file named config-demo-2 with this content:

```
apiVersion: v1
kind: Config
preferences: {}

contexts:
    cluster: development
    namespace: ramp
    user: developer
    name: dev-ramp-up
```

The preceding configuration file defines a new context named dev-ramp-up.

Set the KUBECONFIG environment variable

See whether you have an environment variable named KUBECONFIG. If so, save the current value of your KUBECONFIG environment variable, so you can restore it later. For example, on Linux:

```
export KUBECONFIG_SAVED=$KUBECONFIG
```

The KUBECONFIG environment variable is a list of paths to configuration files. The list is colon-delimited for Linux and Mac, and semicolon-delimited for Windows. If you have a KUBECONFIG environment variable, familiarize yourself with the configuration files in the list.

Temporarily append two paths to your Kubeconfig environment variable. For example, on Linux:

```
export KUBECONFIG=$KUBECONFIG:config-demo:config-demo-2
```

In your config-exercise directory, enter this command:

```
kubectl config view
```

The output shows merged information from all the files listed in your KUBECONFIG environment variable. In particular, notice that the merged information has the dev-ramp-up context from the config-demo-2 file and the three contexts from the config-demo file:

```
contexts:
    context:
    cluster: development
```

4 of 6 09/09/18, 4:52 PM

```
namespace: frontend
    user: developer
  name: dev-frontend
- context:
    cluster: development
    namespace: ramp
user: developer
  name: dev-ramp-up
 context:
    cluster: development
    namespace: storage
  user: developer
name: dev-storage
  context:
    cluster: scratch
    namespace: default
    user: experimenter
  name: exp-scratch
```

For more information about how kubeconfig files are merged, see Organizing Cluster Access Using kubeconfig Files

Explore the \$HOME/.kube directory

If you already have a cluster, and you can use kubectl to interact with the cluster, then you probably have a file named <code>config</code> in the \$HOME/.kube directory.

Go to \$HOME/.kube, and see what files are there. Typically, there is a file named config. There might also be other configuration files in this directory. Briefly familiarize yourself with the contents of these files.

Append \$HOME/.kube/config to your KUBECONFIG environment variable

If you have a \$HOME/.kube/config file, and it's not already listed in your KUBECONFIG environment variable, append it to your KUBECONFIG environment variable now. For example, on Linux:

```
export KUBECONFIG=$KUBECONFIG:$HOME/.kube/config
```

View configuration information merged from all the files that are now listed in your KUBECONFIG environment variable. In your config-exercise directory, enter:

kubectl config view

Clean up

Return your kubeconfig environment variable to its original value. For example, on Linux:

export KUBECONFIG=\$KUBECONFIG_SAVED

What's next

- Organizing Cluster Access Using kubeconfig Files
- kubectl config

Report a problem Edit on Github

Page last modified on June 22, 2018 at 11:20 AM PST by Apply templates to all concepts and tasks to fix double bullets in TOC (#9149) (Page history)

Home Blog Partners Community Case Studies

twitter Github Slack

Stack Overflow Forum Events Calendar

Get Kubernetes Contribute

© 2018 The Kubernetes Authors | Documentation Distributed under CC BY 4.0

Copyright © 2018 The Linux Foundation®. All rights reserved. The Linux Foundation has registered trademarks and uses trademarks. For a list of trademarks of The Linux Foundation, please see our <u>Trademark Usage page</u> ICP license: 京ICP备17074266号-3

- <u>Documentation</u>
- Blog
- Partners
- Community
- <u>Case Studies</u>
- English
- Chinese

5 of 6 09/09/18, 4:52 PM

- v1.11
 v1.10
 v1.9
 v1.8
 v1.7
 Get Started
 Documentation
 Community
 Blog
 Twitter
 Github
 Slack
 Stack Overflow
 Forum
 Events Calendar

6 of 6 09/09/18, 4:52 PM