Guided Exercise: The Kubernetes and OpenShift Command-line Interfaces

Access an OpenShift cluster by using the command-line to get information about cluster services and nodes.

Outcomes

- Use the OpenShift web console to locate the installation file for the `oc` OpenShift command-line interface.
- Get and use a token from the web console to access the cluster from the command line.
- Identify key differences between the `kubectl` and `oc` command-line tools.
- Identify the main components of OpenShift and Kubernetes.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command ensures that all resources are available for this exercise.

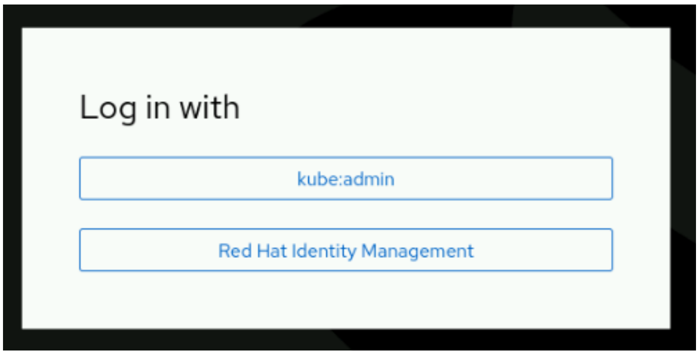
```
[student@workstation ~]$ lab start cli-interfaces
```

Instructions

1. Log in to the OpenShift web console as the `developer` user. Locate the installation file for the `oc` OpenShift command-line interface (CLI).

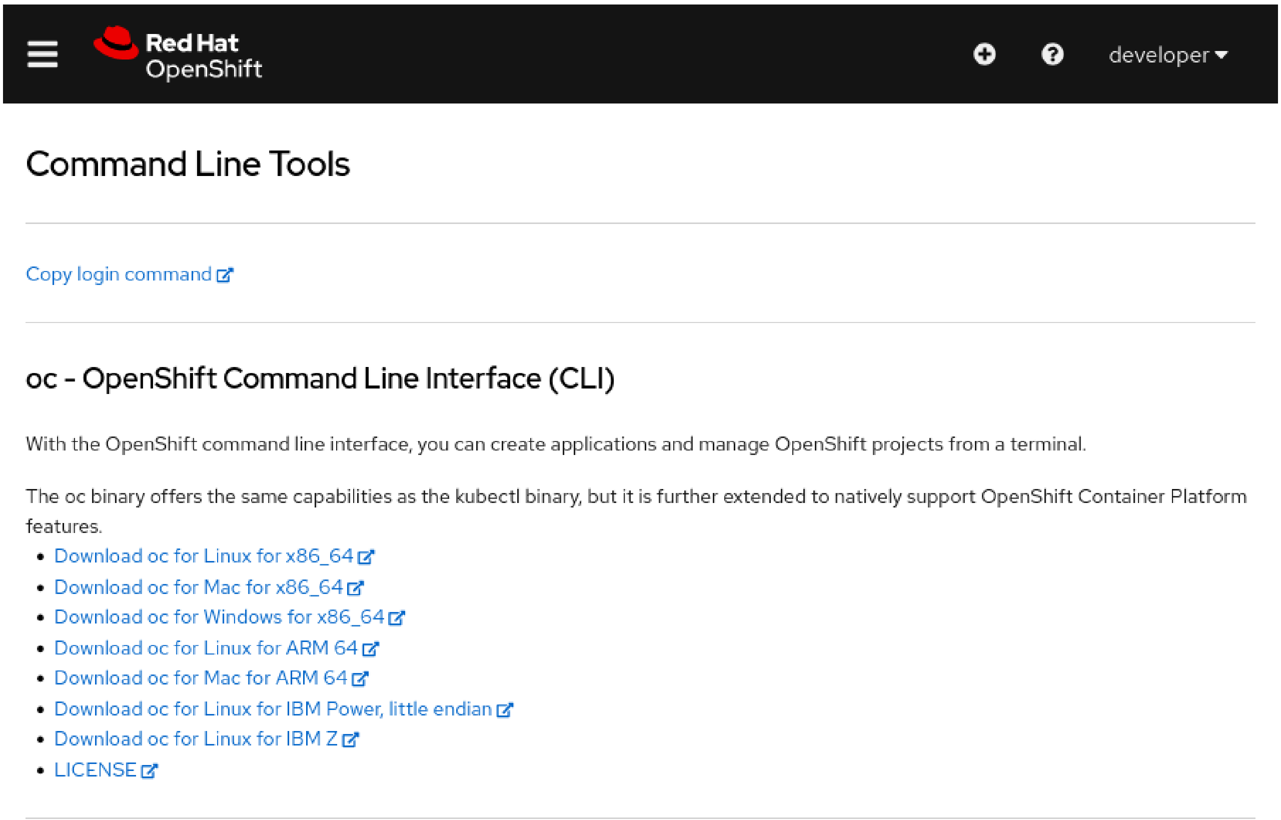
   Open a web browser and go to `https://console-openshift-console.apps.ocp4.example.com`.

   Click **Red Hat Identity Management** and log in as the `developer` user with the `developer` password.



Log in with

   Locate the installation file for the `oc` CLI. From the OpenShift web console, select **Help** → **Command line tools**. The **Help** menu is represented by a `?` icon.



Command Line Tools

Copy login command

oc - OpenShift Command Line Interface (CLI)

With the OpenShift command line interface, you can create applications and manage OpenShift projects from a terminal.

The oc binary offers the same capabilities as the kubectl binary, but it is further extended to natively support OpenShift Container Platform features.

- Download oc for Linux for x86_64
- Download oc for Mac for x86_64
- Download oc for Windows for x86_64
- Download oc for Linux for ARM 64
- Download oc for Mac for ARM 64
- Download oc for Linux for IBM Power, little endian
- Download oc for Linux for IBM Z
- LICENSE

The `oc` binary is available for multiple operating systems and architectures. For each operating system and architecture, the `oc` binary also includes the `kubectl` binary.

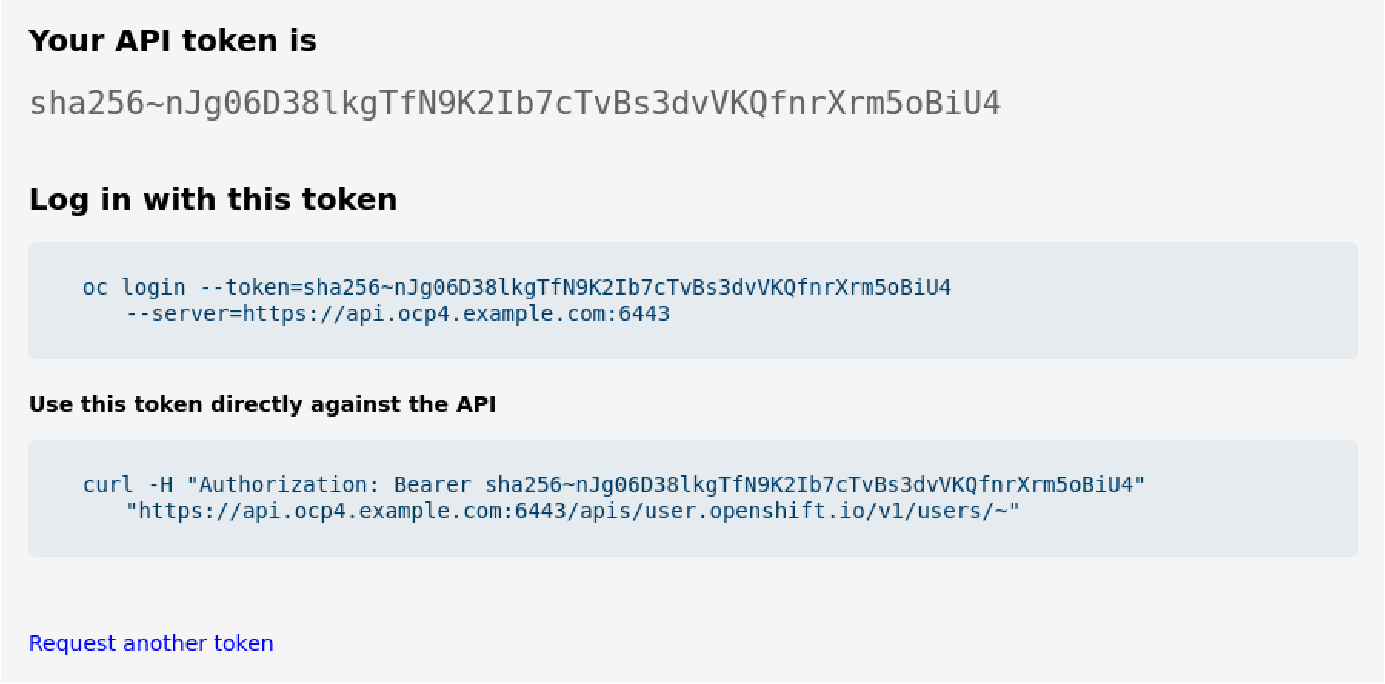
> NOTE
>
> You do not need to download or install the `oc` and `kubectl` binaries, which are already installed on the `workstation` machine.

2. Download an authorization token from the web console. Then, use the token and the `oc` command to log in to the OpenShift cluster.

   From the **Command Line Tools** page, click the **Copy login command** link.

   The link opens a login page. Click **Red Hat Identity Management** and log in as the `developer` user with the `developer` password.

   A web page is displayed. Click the **Display token** link to show your API token and the login command.



Your API token is

sha256~nJg06D38lkgTfN9K2Ib7cTvBs3dvVKQfnrXrm5oBiU4

Log in with this token

```
oc login --token=sha256~nJg06D38lkgTfN9K2Ib7cTvBs3dvVKQfnrXrm5oBiU4
   --server=https://api.ocp4.example.com:6443
```

Use this token directly against the API

```
curl -H "Authorization: Bearer sha256~nJg06D38lkgTfN9K2Ib7cTvBs3dvVKQfnrXrm5oBiU4"
   "https://api.ocp4.example.com:6443/apis/user.openshift.io/v1/users/~"
```

Request another token

Copy the `oc login` command to your clipboard. Open a terminal window and then use the copied command to log in to the cluster with your token.

```
[student@workstation ~]$ oc login --token=sha256-fypX...Ot6A \
  --server=https://api.ocp4.example.com:6443
Logged into "https://api.ocp4.example.com:6443" as "developer" using the token provided.
...output omitted...
```

3. Compare the available commands for the `kubectl` and `oc` commands.

   Use the `help` command to list and review the available commands for the `kubectl` command.

```
[student@workstation ~]$ kubectl help
kubectl controls the Kubernetes cluster manager.

 Find more information at: https://kubernetes.io/docs/reference/kubectl/

Basic Commands (Beginner):
  create          Create a resource from a file or from stdin
  expose          Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
  run             Run a particular image on the cluster
  set             Set specific features on objects

Basic Commands (Intermediate):
  explain         Get documentation for a resource
  get             Display one or many resources
  edit            Edit a resource on the server
  delete          Delete resources by file names, stdin, resources and names, or by resources and label selector
...output omitted....
```

   Notice that the `kubectl` command does not provide a `login` command.

   Examine the available subcommands and options for the `kubectl create` command by using the `--help` option.

```
[student@workstation ~]$ kubectl create --help
Create a resource from a file or from stdin.

 JSON and YAML formats are accepted.

Examples:
  # Create a pod using the data in pod.json
  kubectl create -f ./pod.json
...output omitted....
Available Commands:
  clusterrole           Create a cluster role
  clusterrolebinding    Create a cluster role binding for a particular cluster role
  configmap             Create a config map from a local file, directory or literal value
  cronjob               Create a cron job with the specified name
  deployment            Create a deployment with the specified name
...output omitted...
Options:
    --allow-missing-template-keys=true:
        If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to
        golang and jsonpath output formats.

    --dry-run='none':
        Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without
        sending it. If server strategy, submit server-side request without persisting the resource.
...output omitted....
Usage:
  kubectl create -f FILENAME [options]

Use "kubectl create <command> --help" for more information about a given command.
Use "kubectl options" for a list of global command-line options (applies to all commands).
```

You can use the `--help` option with any `kubectl` command. The `--help` option provides information about a command, including the available subcommands and options, and the command syntax.

List and review the available commands for the `oc` binary by using the `help` command.

```
[student@workstation ~]$ oc help
OpenShift Client

This client helps you develop, build, deploy, and run your applications on any
OpenShift or Kubernetes cluster. It also includes the administrative
commands for managing a cluster under the 'adm' subcommand.

Basic Commands:
  login             Log in to a server
  new-project       Request a new project
  new-app           Create a new application
  status            Show an overview of the current project
  project           Switch to another project
  projects          Display existing projects
  explain           Get documentation for a resource
...output omitted....
```

The `oc` command supports the same capabilities as the `kubectl` command. The `oc` command provides additional commands to natively support an OpenShift cluster. For example, the `new-project` command creates a project, which is a Kubernetes namespace, in the OpenShift cluster. The `new-app` command is unique to the `oc` command. It creates applications by using existing source code or prebuilt images.

Use the `--help` option with the `oc create` command to view the available subcommands and options.

```
[student@workstation ~]$ oc create --help
Create a resource from a file or from stdin.

 JSON and YAML formats are accepted.

Examples:
  # Create a pod using the data in pod.json
  oc create -f ./pod.json
...output omitted...

Available Commands:
  build                  Create a new build
  clusterresourcequota   Create a cluster resource quota
  clusterrole            Create a cluster role
  clusterrolebinding     Create a cluster role binding for a particular cluster role
  configmap              Create a config map from a local file, directory or literal value
  cronjob                Create a cron job with the specified name
  deployment             Create a deployment with the specified name
...output omitted....
Options:
    --allow-missing-template-keys=true:
        If true, ignore any errors in templates when a field or map key is missing in the template. Only applies to
        golang and jsonpath output formats.

    --dry-run='none':
        Must be "none", "server", or "client". If client strategy, only print the object that would be sent, without
        sending it. If server strategy, submit server-side request without persisting the resource.
...output omitted...
Usage:
  oc create -f FILENAME [options]
....output omitted....
```

The `oc create` command includes the same subcommands and options as the `kubectl create` command, and provides additional subcommands for OpenShift resources. For example, you can use the `oc create` command to create OpenShift resources such as a deployment, a route, and an image stream.

4. Use the `oc login` command with the `--web` option to log in as the `admin` user with `redhatocp` as the password. Identify the components and Kubernetes resources of an OpenShift cluster by using the terminal. Regular cluster users, such as the `developer` user, cannot list resources at a cluster scope. Unless otherwise noted, all commands are available for the `oc` and `kubectl` commands.

   In a terminal, use the `oc login --web` command.

   ```
   [student@workstation ~]$ oc login --web
   ```

   In the browser tab that opens, click **Red Hat Identity Management** and log in as the `admin` user with `redhatocp` as the password. You get the `access token received successfully; please return to your terminal` message. Close the browser tab.

   Return to the terminal window. Identify the cluster version with the `version` command.

   ```
   [student@workstation ~]$ oc version
   Client Version: 4.18.6
   Kustomize Version: v5.4.2
   Server Version: 4.18.6
   Kubernetes Version: v1.31.6
   ```

   Use the `cluster-info` command to identify the URL for the Kubernetes control plane.

   ```
   [student@workstation ~]$ oc cluster-info
   Kubernetes control plane is running at https://api.ocp4.example.com:6443

   To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
   ```

   Identify the supported API versions by using the `api-versions` command.

```
[student@workstation ~]$ oc api-versions
admissionregistration.k8s.io/v1
admissionregistration.k8s.io/v1beta1
apiextensions.k8s.io/v1
apiregistration.k8s.io/v1
apiserver.openshift.io/v1
apps.openshift.io/v1
apps/v1
...output omitted....
```

List cluster operators with the `get clusteroperator` command.

```
[student@workstation ~]$ oc get clusteroperator
NAME                       VERSION  AVAILABLE  PROGRESSING  DEGRADED  SINCE ...
authentication             4.18.6   True       False        False     5d5h
baremetal                  4.18.6   True       False        False     41d
cloud-controller-manager   4.18.6   True       False        False     41d
cloud-credential           4.18.6   True       False        False     41d
cluster-autoscaler         4.18.6   True       False        False     41d
config-operator            4.18.6   True       False        False     41d
console                    4.18.6   True       False        False     29d
control-plane-machine-set  4.18.6   True       False        False     41d
csi-snapshot-controller    4.18.6   True       False        False     41d
dns                        4.18.6   True       False        False     32m
etcd                       4.18.6   True       False        False     41d
image-registry             4.18.6   True       False        False     41d
ingress                    4.18.6   True       False        False     41d
...output omitted...
```

Use the `get` command to list pods in the `openshift-api` project. Specify the project with the `-n` option.

```
[student@workstation ~]$ oc get pods -n openshift-apiserver
NAME                         READY   STATUS    RESTARTS   AGE
apiserver-68c9485699-ndqlc   2/2     Running   6          18d
```

Use the `oc status` command to retrieve the status of resources in the `openshift-authentication` project.

```
[student@workstation ~]$ oc status -n openshift-authentication
Warning: apps.openshift.io/v1 DeploymentConfig is deprecated in v4.14+, unavailable in v4.10000+
In project openshift-authentication on server https://api.ocp4.example.com:6443

https://oauth-openshift.apps.ocp4.example.com (passthrough) to pod port 6443 (svc/oauth-openshift)
  deployment/oauth-openshift deploys quay.io/openshift-release-dev/ocp-v4.0-art-dev@sha256:64e6...de42
    deployment #7 running for 2 weeks - 1 pod
    deployment #6 deployed 2 weeks ago
    deployment #4 deployed 2 weeks ago
    deployment #5 deployed 2 weeks ago
    deployment #3 deployed 2 weeks ago
    deployment #2 deployed 2 weeks ago
    deployment #1 deployed 2 weeks ago
...output omitted...
```

Use the `explain` command to list the description and available fields for `services` resources.

```
[student@workstation ~]$ oc explain services
KIND:     Service
VERSION:  v1

DESCRIPTION:
     Service is a named abstraction of software service (for example, mysql)
     consisting of local port (for example 3306) that the proxy listens on, and
     the selector that determines which pods will answer requests sent through
     the proxy.

FIELDS:
   apiVersion   <string>
     APIVersion defines the versioned schema of this representation of an
     object. Servers should convert recognized schemas to the latest internal
     value, and may reject unrecognized values.
...output omitted...
```

Use the `get` command to list cluster nodes.

```
[student@workstation ~]$ oc get nodes
NAME       STATUS   ROLES                         AGE   VERSION
master01   Ready    control-plane,master,worker   18d   v1.31.6
```

A single node exists in the cluster.

Finish

On the `workstation` machine, use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish cli-interfaces
```