Lab 3.1 - Creating a New Cluster Using MicroK8s

If you already have an existing cluster, you may opt to skip this section and move on.

Note: The following instructions assume running on a Linux system using Ubuntu 18.04.4 LTS.

MicroK8s is an open source project led by Ubuntu which enables lightweight ephemeral Kubernetes clusters which are particularly useful for test purposes. The project homepage is located at https://microk8s.io/ and the GitHub repo at https://github.com/ubuntu/microk8s.

Step 1

Run the following command to install MicroK8s:

```
$ sudo snap install microk8s --classic --channel=1.18/stable
```

Step 2

Add your user to the group used by MicroK8s, give yourself ownership of the `~/.kube` directory, and re-initialize your terminal session:

```
$ sudo usermod -a -G microk8s $USER
$ sudo chown -f -R $USER ~/.kube
$ sudo su - $USER
```

Step 3

Next, run the following command to check if the Kubernetes cluster is ready:


```
$ microk8s status --wait-ready
```

This will hang in the terminal until the cluster is ready (or there is an error). Once done, the output should look similar to the following:

```
microk8s is running
addons:
cilium: disabled
dashboard: disabled
...
```

If the output contains "`microk8s is running`", your cluster should be ready to use.

Step 4

As part of MicroK8s starting, your environment should be automatically authenticated against the new cluster. Just to be extra safe, you can run the following command:

```
$ microk8s config > ~/.kube/config
```

This outputs the MicroK8s "kubeconfig" to the file path `~/.kube/config`, which is the default location for the YAML-based Kubernetes authentication file. Many Kubernetes-related tools (such as Helm) use this file to authenticate against the Kubernetes API.

This default is overridden by the KUBECONFIG environment variable. If this environment variable is set in your environment, you have two (2) options:

1. Unset the variable for the current terminal session:
   ```
   $ unset KUBECONFIG
   ```
2. Override the contents of the file at KUBECONFIG (**Warning**: Make sure you're not overwriting some other existing kubeconfig file first):
   ```
   $ microk8s config > "${KUBECONFIG}"
   ```

Step 5

One of the nice things about MicroK8s is that it ships with its own version of kubectl, the Kubernetes CLI. You can verify cluster access with the following command:

```
$ microk8s kubectl cluster-info
```

Expected output should look like the following:


```
Kubernetes master is running at https://127.0.0.1:16443
```

Notice the “127.0.0.1” address indicating the cluster is running locally.

Step 6

For the rest of the course you will see references to “kubectl”. Unless you’ve installed “kubectl” separately, you’ll need to run **`microk8s kubectl`**. In order to create and activate a simple alias for this, you should run the following:

```
$ echo "alias kubectl='microk8s kubectl'" >> "${HOME}/.bash_aliases"
$ source "${HOME}/.bash_aliases"
```

You can now use the **`kubectl`** command, which will run **`microk8s kubectl`** behind the scenes.

You are now ready to move on.

