

Install Docker + Kubernetes on CentOS 7.9

Install a single node of Kubernetes using minikube on CentOS 7.9. Ensure your system has a min of 2 CPUs required for Kubernetes.

User: istio

Password: lumada



But first, let's update the package database:

Right mouse and select

New Folder	Shift+Ctrl+N
Paste	Ctrl+V
Select All	Ctrl+A
<input checked="" type="checkbox"/> Keep aligned	
Organize Desktop by Name	
Change Background	
Open Terminal	

```
$ sudo yum check-update
```

```
$ sudo yum -y update
```

Pre-requisites

Hostnames

Before you start, it is advised to check what your current hostname is. Type the following command in the console to find out:

```
$ hostnamectl
```

```
$ sudo nano /etc/hosts
```

Add to the 127.0.0.1 line

```
bookinfo.local and test.bookinfo.local
```

set hostname as localhost

```
$ hostnamectl set-hostname localhost
```

Later you will need to map the external loadbalancer IP address to localhost

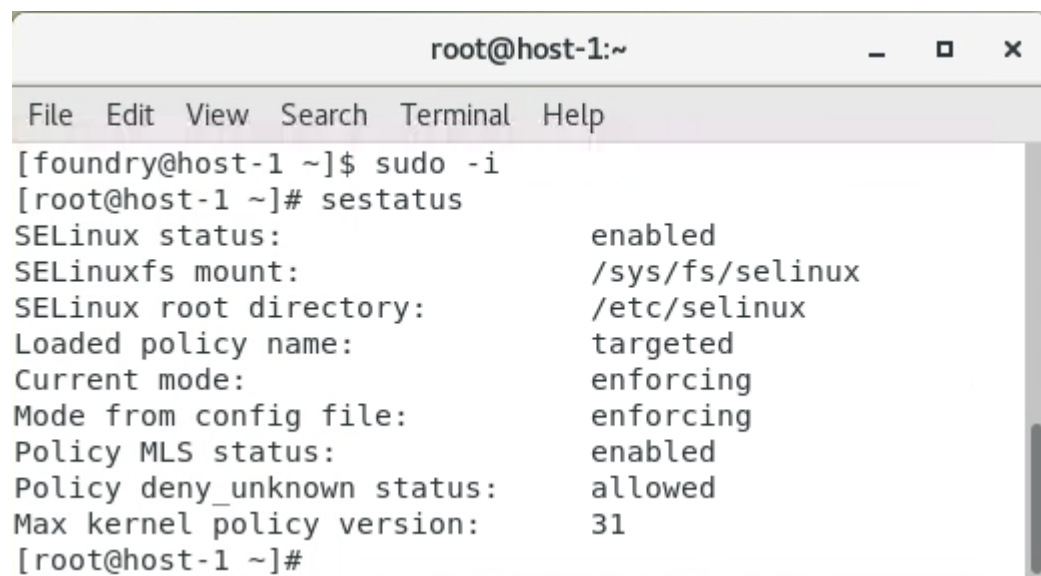
Check the SELinux Status

To view the current SELinux status and the SELinux policy that is being used on your system, use the `sestatus` command:

Switch to Root user

```
$ sudo -i
```

```
# sestatus
```

A screenshot of a terminal window titled 'root@host-1:~'. The terminal shows the output of the 'sestatus' command. The output lists various SELinux settings: status is 'enabled', SELinuxfs mount is '/sys/fs/selinux', SELinux root directory is '/etc/selinux', loaded policy name is 'targeted', current mode is 'enforcing', mode from config file is 'enforcing', policy MLS status is 'enabled', policy deny_unknown status is 'allowed', and max kernel policy version is '31'.

```
root@host-1:~  
File Edit View Search Terminal Help  
[foundry@host-1 ~]$ sudo -i  
[root@host-1 ~]# sestatus  
SELinux status:                enabled  
SELinuxfs mount:                /sys/fs/selinux  
SELinux root directory:        /etc/selinux  
Loaded policy name:            targeted  
Current mode:                  enforcing  
Mode from config file:         enforcing  
Policy MLS status:             enabled  
Policy deny_unknown status:    allowed  
Max kernel policy version:     31  
[root@host-1 ~]#
```

```
# setenforce 0
```

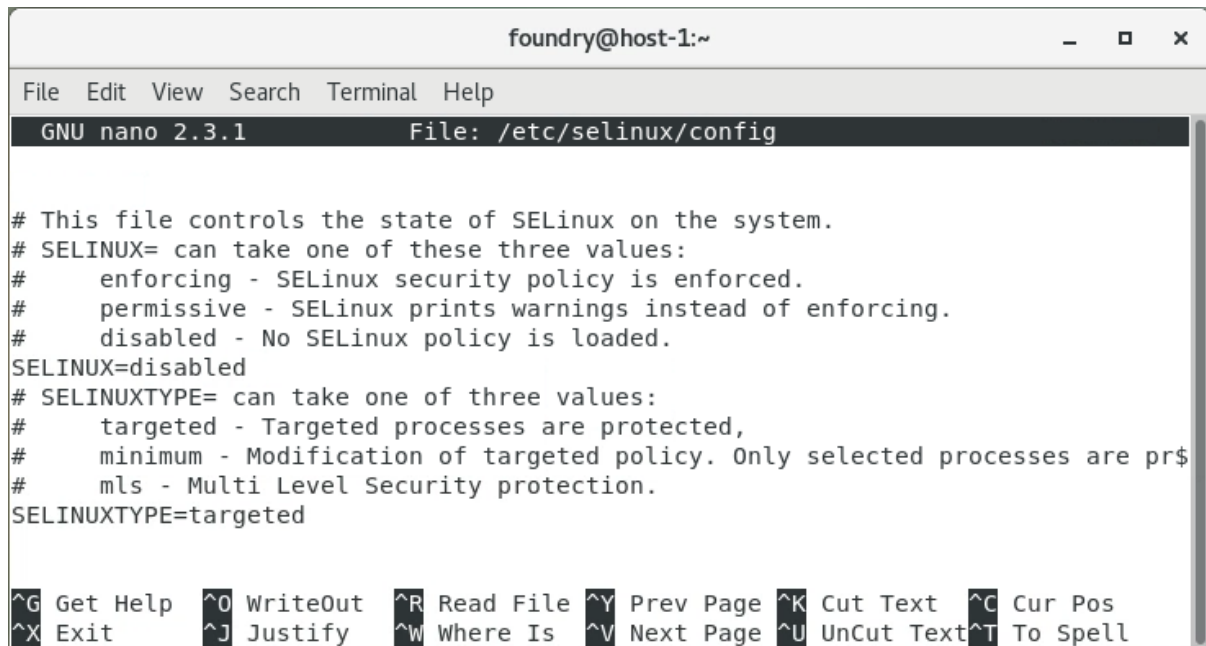
```
# sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g'  
/etc/sysconfig/selinux
```

```
# systemctl stop firewalld
```

```
# systemctl disable firewalld
```

Open the `/etc/selinux/config` file and set the SELINUX mod to disabled:

```
$ sudo nano /etc/selinux/config
```



```
foundry@host-1:~
File Edit View Search Terminal Help
GNU nano 2.3.1 File: /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#   targeted - Targeted processes are protected,
#   minimum - Modification of targeted policy. Only selected processes are protected.
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Disable SWAP

```
$ sudo swapoff -a
```

Reboot Server

```
$ reboot
```

Install Git

You will need to install Git to access the GitHub repository

```
$ sudo yum -y install git
```

Create a directory for course-materials

```
$ sudo mkdir /opt/course-materials/Istio
```

Now that we have git installed, we need to configure it so that it links to a repository.

```
# git config
```

Add name and email address for commits

```
root@host [~]# git config --global user.name "User Name"
```

```
root@host [~]# git config --global user.email "yourname@domain.com"
```

View the configuration information

```
root@host [~]# git config --list
```

```
exit
```

Install Visual Code

To install the stable 64-bit VS Code from a yum repository:

```
$ yum check-update  
  
$ sudo yum -y update  
  
$ sudo rpm --import  
https://packages.microsoft.com/keys/microsoft.asc  
  
$ sudo sh -c 'echo -e "[code]\nname=Visual Studio  
Code\nbaseurl=https://packages.microsoft.com/yumrepos/vscode\nenable  
d=1\nngpgcheck=1\nngpgkey=https://packages.microsoft.com/keys/microsof  
t.asc" > /etc/yum.repos.d/vscode.repo'  
  
$ sudo yum install code  
  
$ code
```

Install the following extensions:

Docker 1.9.x

Kubernetes 1.2.x

vscode istio snippets 0.1.0

JWT

Install Docker

The purpose of the install script is for a convenience for quickly installing the latest Docker-CE releases on the supported Linux distros. It is not recommended for deployment to production systems.

```
$ curl -fsSL https://get.docker.com -o get-docker.sh
```

```
$ sudo sh get-docker.sh
```

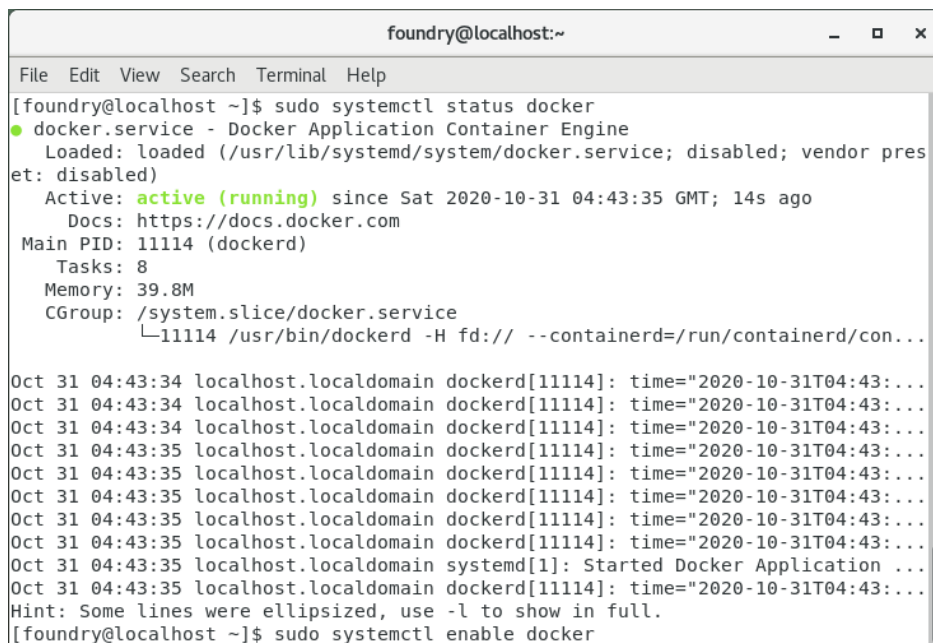
After installation has completed, start the Docker daemon:

```
$ sudo systemctl start docker
```

Verify that it's running:

```
$ sudo systemctl status docker
```

The output should be like the following, showing that the service is active and running:

A terminal window titled 'foundry@localhost:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command '[foundry@localhost ~]\$ sudo systemctl status docker' and its output. The output indicates that the 'docker.service' is 'active (running)' since Saturday, 2020-10-31 at 04:43:35 GMT. It lists the main PID as 11114 (dockerd), 8 tasks, 39.8M memory, and the CGroup path. Below this, a series of log entries from 'localhost.localdomain' show the docker daemon starting and running. The last line shows the command '[foundry@localhost ~]\$ sudo systemctl enable docker'.

Lastly, make sure it starts at every server reboot:

```
$ sudo systemctl enable docker
```

Executing Docker Command Without Sudo

By default, running the docker command requires root privileges — that is, you must prefix the command with `sudo`. It can also be run by a user in the docker group, which is automatically created during the installation of Docker. If you attempt to run the docker command without prefixing it with `sudo` or without being in the docker group, you'll get an output like this:

Output

```
docker: Cannot connect to the Docker daemon. Is the docker daemon running on this host?.
```

```
See 'docker run --help'.
```

If you want to avoid typing `sudo` whenever you run the docker command, add your username to the docker group:

```
$ sudo usermod -aG docker $(whoami)
```

You will need to log out of the Droplet and back in as the same user to enable this change.

If you need to add a user to the docker group that you're not logged in as, declare that username explicitly using:

```
$ sudo usermod -aG docker username
```

The rest of this article assumes you are running the docker command as a user in the docker user group. If you choose not to, please prepend the commands with `sudo`.

Start & Stop Docker Services

```
$ sudo systemctl start docker.service ## <-- Start docker ##
```

```
$ sudo systemctl stop docker.service ## <-- Stop docker ##
```

```
$ sudo systemctl restart docker.service ## <-- Restart docker ##
```

```
$ sudo systemctl status docker.service ## <-- Get status of docker ##
```

Using the Docker Command

With Docker installed and working, now's the time to become familiar with the command line utility. Using docker consists of passing it a chain of options and subcommands followed by arguments. The syntax takes this form:

docker [option] [command] [arguments]

To view all available subcommands, type:

```
$ docker
```

Output

Output

attach	Attach to a running container
build	Build an image from a Dockerfile
commit	Create a new image from a container's changes
cp	Copy files/folders between a container and the local
filesystem	
create	Create a new container
diff	Inspect changes on a container's filesystem
events	Get real time events from the server
exec	Run a command in a running container
export	Export a container's filesystem as a tar archive
history	Show the history of an image
images	List images
import	Import the contents from a tarball to create a filesystem
image	
info	Display system-wide information
inspect	Return low-level information on a container or image
kill	Kill a running container
load	Load an image from a tar archive or STDIN
login	Log in to a Docker registry
logout	Log out from a Docker registry
logs	Fetch the logs of a container
network	Manage Docker networks
pause	Pause all processes within a container
port	List port mappings or a specific mapping for the CONTAINER
ps	List containers
pull	Pull an image or a repository from a registry
push	Push an image or a repository to a registry
rename	Rename a container
restart	Restart a container
rm	Remove one or more containers
rmi	Remove one or more images
run	Run a command in a new container
save	Save one or more images to a tar archive
search	Search the Docker Hub for images
start	Start one or more stopped containers
stats	Display a live stream of container(s) resource usage
statistics	
stop	Stop a running container
tag	Tag an image into a repository
top	Display the running processes of a container
unpause	Unpause all processes within a container
update	Update configuration of one or more containers
version	Show the Docker version information
volume	Manage Docker volumes
wait	Block until a container stops, then print its exit code

Install Kubernetes

- Install kubectl
- Install minikube

Install Kubectl

```
$ curl -LO https://storage.googleapis.com/kubernetes-release/release/\$\(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt\)/bin/linux/amd64/kubectl
```

Make the kubectl binary executable.

```
$ chmod +x ./kubectl
```

Move the binary in to your PATH.

```
$ sudo mv ./kubectl /usr/local/bin/kubectl
```

Test to ensure the version you installed is up-to-date:

```
$ kubectl version --client
```

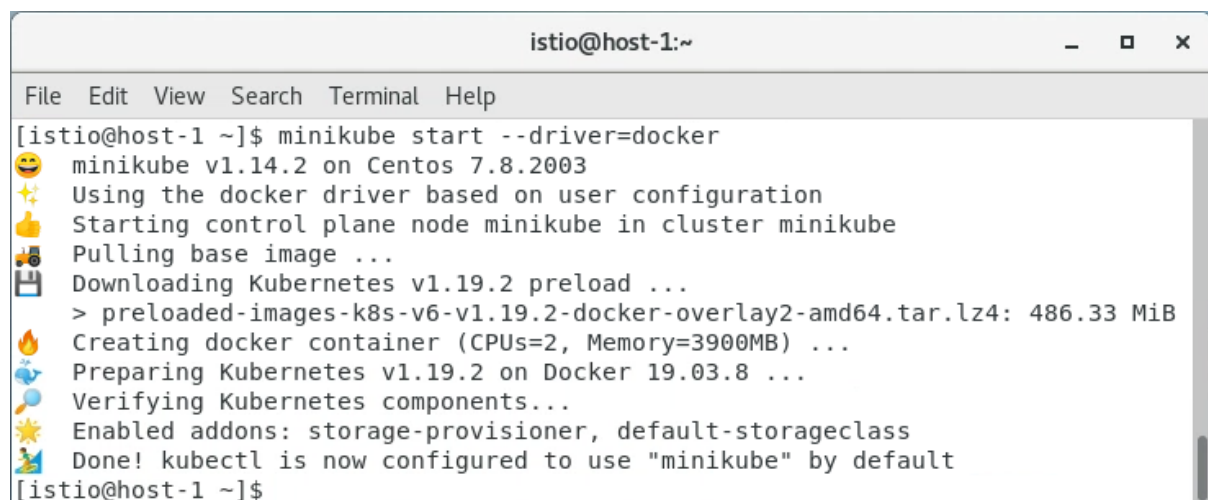
Install Kubernetes - Minikube

```
$ curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
```

```
$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
```

```
$ sudo usermod -aG docker $USER && newgrp docker
```

```
$ minikube start --driver=docker
```



```
istio@host-1:~  
File Edit View Search Terminal Help  
[istio@host-1 ~]$ minikube start --driver=docker  
😄 minikube v1.14.2 on Centos 7.8.2003  
🌟 Using the docker driver based on user configuration  
👍 Starting control plane node minikube in cluster minikube  
🚚 Pulling base image ...  
📦 Downloading Kubernetes v1.19.2 preload ...  
> preloaded-images-k8s-v6-v1.19.2-docker-overlay2-amd64.tar.lz4: 486.33 MiB  
🔥 Creating docker container (CPUs=2, Memory=3900MB) ...  
🌐 Preparing Kubernetes v1.19.2 on Docker 19.03.8 ...  
🔍 Verifying Kubernetes components...  
🌞 Enabled addons: storage-provisioner, default-storageclass  
🎉 Done! kubectl is now configured to use "minikube" by default  
[istio@host-1 ~]$
```

If minikube fails to start, see the drivers page for help setting up a compatible container or virtual-machine manager.

To make docker the default driver:

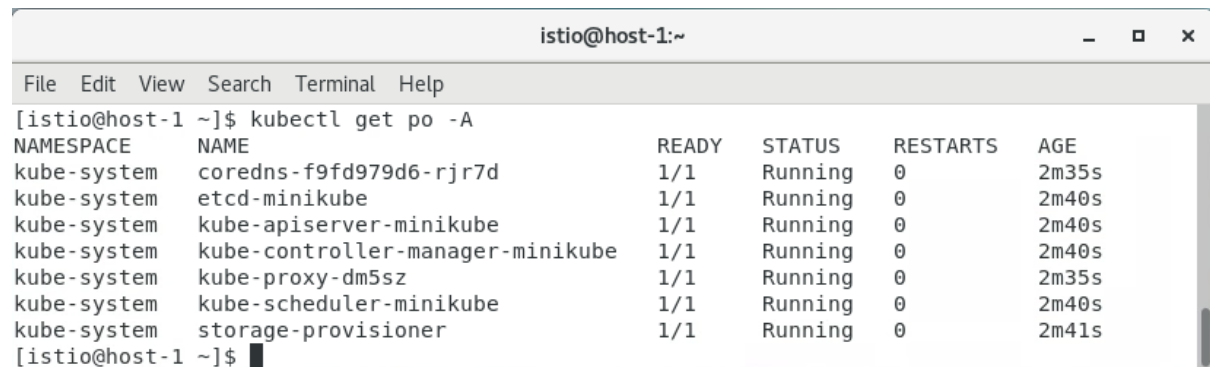
```
$ minikube config set driver docker
```

```
$ minikube stop
```

```
$ minikube delete
```

Ensure services are up and running:

```
$ kubectl get po -A
```

A terminal window titled 'istio@host-1:~' showing the output of the command 'kubectl get po -A'. The output is a table with columns: NAMESPACE, NAME, READY, STATUS, RESTARTS, and AGE. The table lists several pods in the kube-system namespace, all in a 'Running' state.

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-f9fd979d6-rjr7d	1/1	Running	0	2m35s
kube-system	etcd-minikube	1/1	Running	0	2m40s
kube-system	kube-apiserver-minikube	1/1	Running	0	2m40s
kube-system	kube-controller-manager-minikube	1/1	Running	0	2m40s
kube-system	kube-proxy-dm5sz	1/1	Running	0	2m35s
kube-system	kube-scheduler-minikube	1/1	Running	0	2m40s
kube-system	storage-provisioner	1/1	Running	0	2m41s

Set a loadbalancer

```
$ minikube tunnel
```

To find the routable IP, run this command and examine the EXTERNAL-IP column:

```
$ kubectl get services balanced
```

Increase the default memory limit (requires a restart):

```
$ minikube config set memory 16384
```

```
$ minikube config set cpus 4
```

Browse the catalog of easily installed Kubernetes services:

```
$ minikube addons list
```

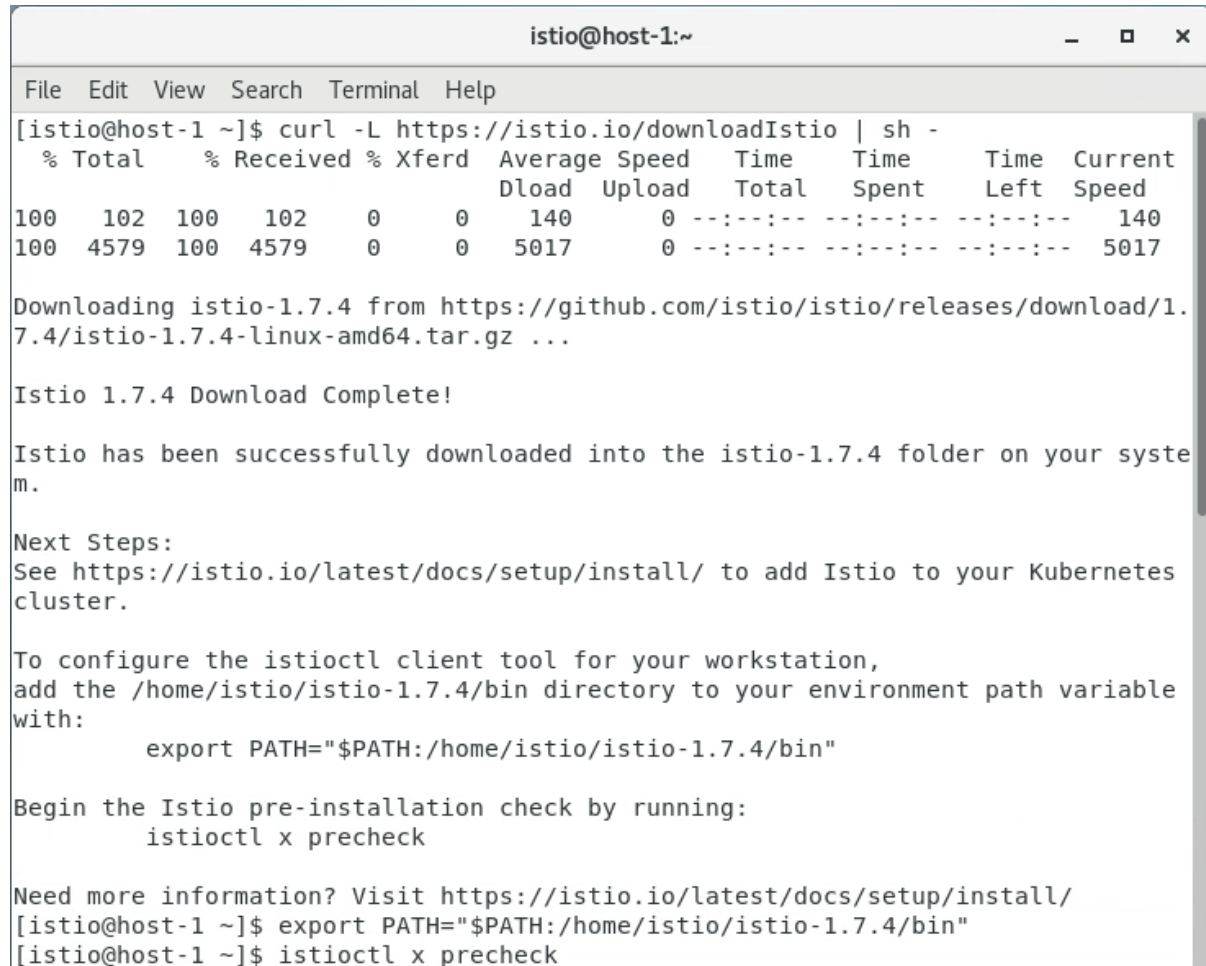
```
$ minikube dashboard
```

For further details: <https://minikube.sigs.k8s.io/docs/commands/>

Install Istio

Go to the [Istio release page](https://istio.io/latest/docs/setup/install/) to download the installation file for your OS, or download and extract the latest release automatically (Linux or macOS):

```
$ curl -L https://istio.io/downloadIstio | sh -
```



The screenshot shows a terminal window titled "istio@host-1:~". The terminal output displays the execution of the command `curl -L https://istio.io/downloadIstio | sh -`. It shows a progress bar for downloading the Istio 1.7.4 release for Linux AMD64. The download is complete, and the terminal provides instructions for the next steps, including adding the Istio bin directory to the PATH and running the pre-check.

```
istio@host-1:~  
File Edit View Search Terminal Help  
[istio@host-1 ~]$ curl -L https://istio.io/downloadIstio | sh -  
% Total % Received % Xferd Average Speed Time Time Time Current  
Dload Upload Total Spent Left Speed  
100 102 100 102 0 0 140 0 --:--:-- --:--:-- --:--:-- 140  
100 4579 100 4579 0 0 5017 0 --:--:-- --:--:-- --:--:-- 5017  
  
Downloading istio-1.7.4 from https://github.com/istio/istio/releases/download/1.7.4/istio-1.7.4-linux-amd64.tar.gz ...  
  
Istio 1.7.4 Download Complete!  
  
Istio has been successfully downloaded into the istio-1.7.4 folder on your system.  
  
Next Steps:  
See https://istio.io/latest/docs/setup/install/ to add Istio to your Kubernetes cluster.  
  
To configure the istioctl client tool for your workstation,  
add the /home/istio/istio-1.7.4/bin directory to your environment path variable with:  
    export PATH="$PATH:/home/istio/istio-1.7.4/bin"  
  
Begin the Istio pre-installation check by running:  
    istioctl x precheck  
  
Need more information? Visit https://istio.io/latest/docs/setup/install/  
[istio@host-1 ~]$ export PATH="$PATH:/home/istio/istio-1.7.4/bin"  
[istio@host-1 ~]$ istioctl x precheck
```

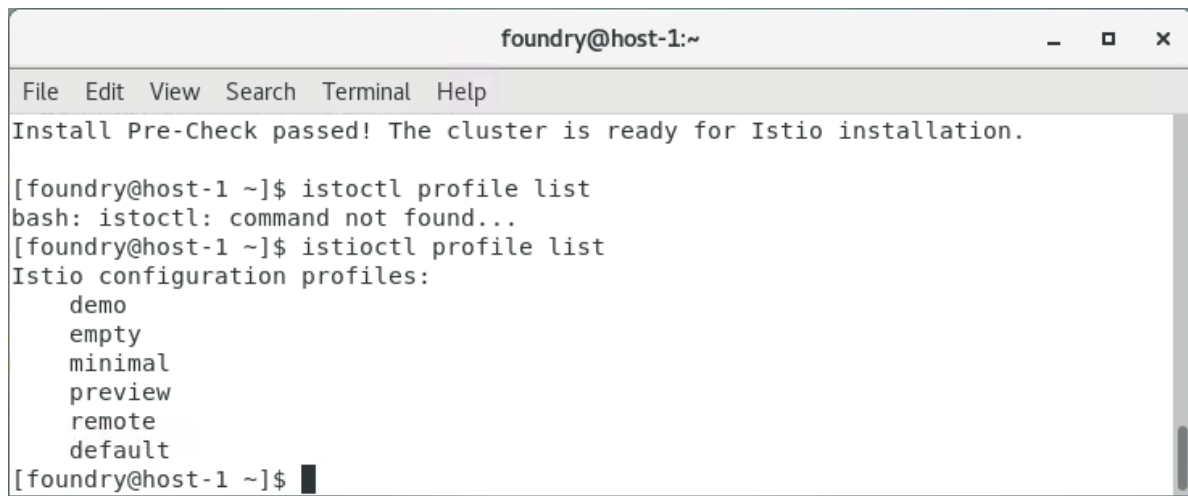
Add the istioctl client to your path

```
$ cd istio-1.7.4  
$ sudo nano ~/.bash_profile  
$ export PATH="$PATH:/home/istio/istio-1.7.4/bin"  
$ source ~/.bash_profile
```

Run the check

```
$ istioctl x precheck
```

```
$ istioctl profile list
```



```
foundry@host-1:~  
File Edit View Search Terminal Help  
Install Pre-Check passed! The cluster is ready for Istio installation.  
  
[foundry@host-1 ~]$ istioctl profile list  
bash: istioctl: command not found...  
[foundry@host-1 ~]$ istioctl profile list  
Istio configuration profiles:  
  demo  
  empty  
  minimal  
  preview  
  remote  
  default  
[foundry@host-1 ~]$
```