Install Docker + Kubernetes on CentOS 7.8

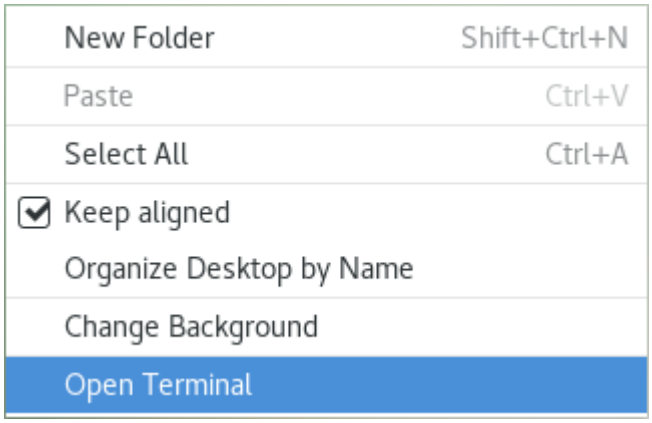
Install a single node of Kubernetes using minikube on CentOS 7.8. Ensure your system has a min of 2 CPUs required for Kubernetes.

User: foundry

Password: lumada

But first, let's update the package database:

Right mouse and select



$ sudo yum check-update

$ sudo yum -y update

Pre-requisites

Hostnames

Check Existing Hostname

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

bookinf.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 locahost

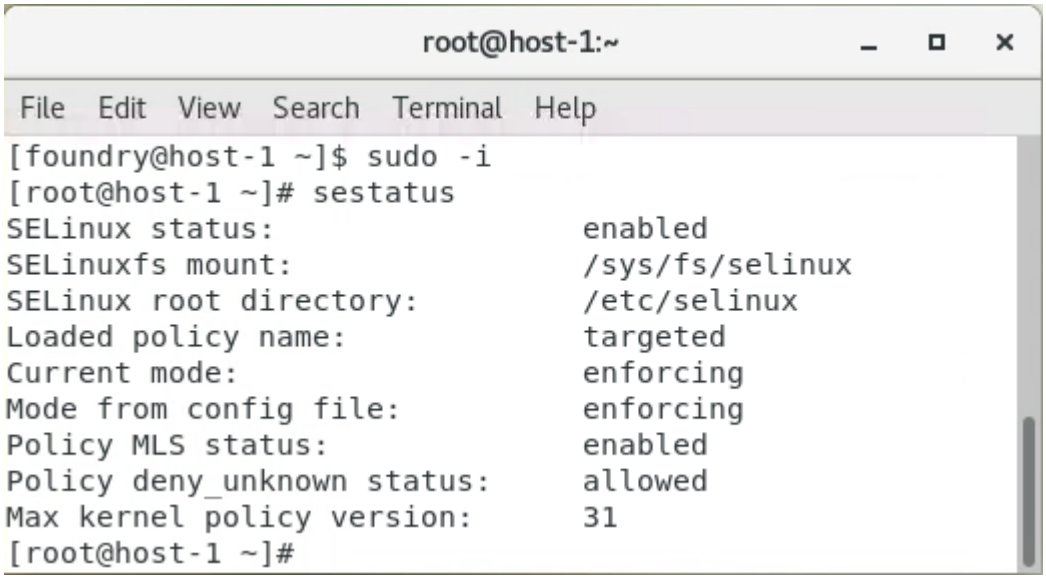
Check the SELinux Status

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

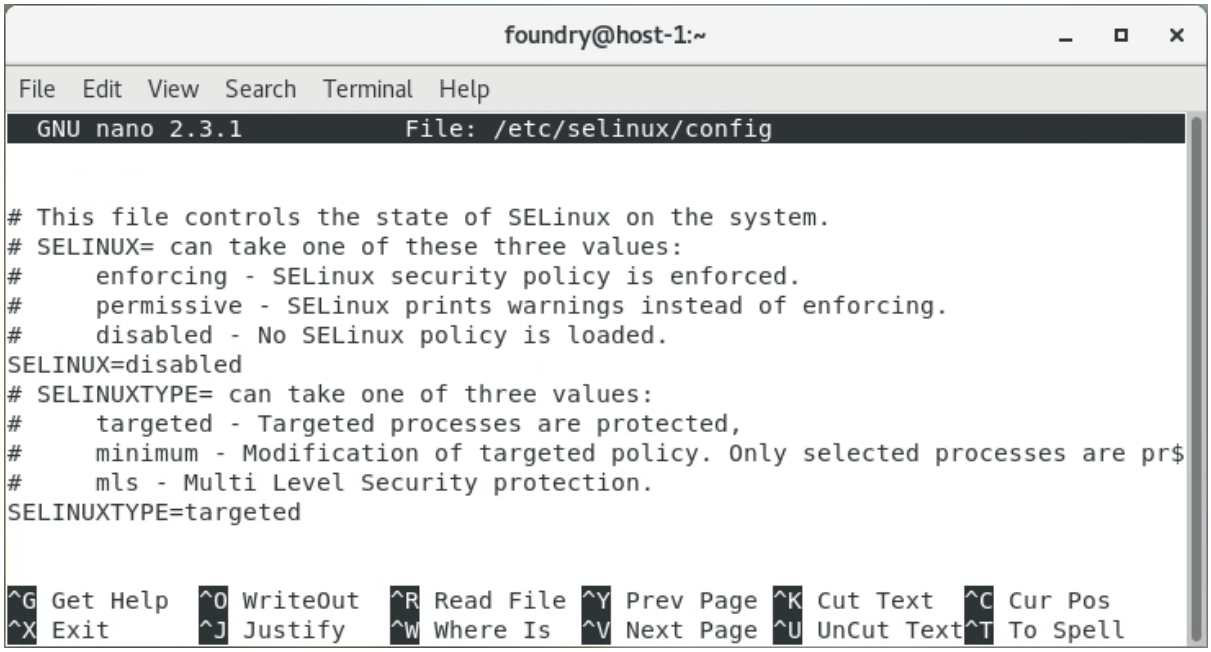
sestatus



setenforce 0

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

$ sudo nano /etc/selinux/config



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\nenabled=1\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" > /etc/yum.repos.d/vscode.repo'

$ sudo yum install code

$ code

Install the following extensions:

Docker 1.7.0

Kubernetes 1.2.1

vscode istio snippets 0.1.0

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

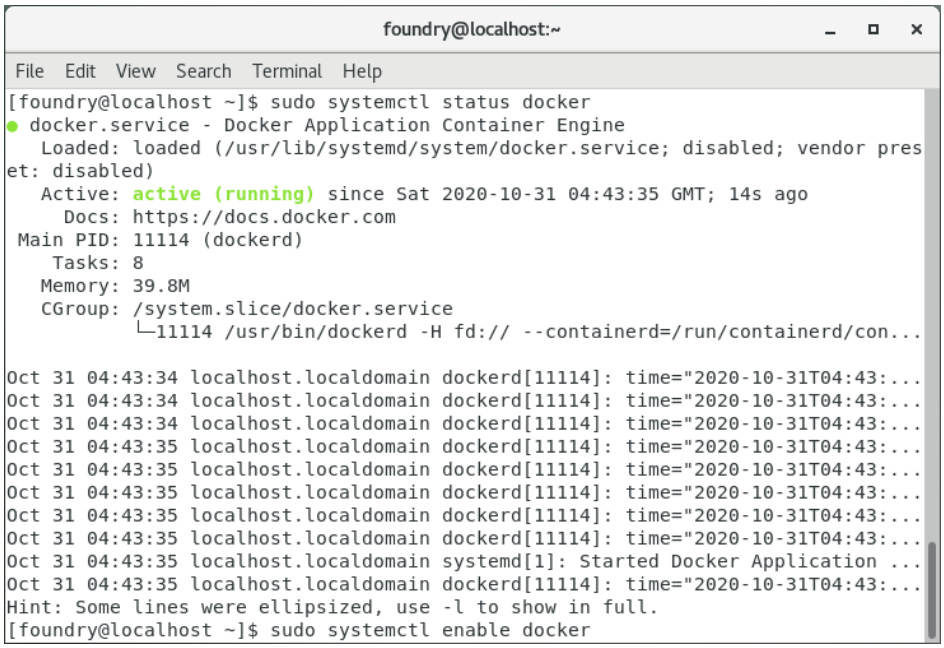
After installation has completed, start the Docker daemon:

$ 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:



Lastly, make sure it starts at every server reboot:

$ sudo systemctl enable docker

Executing Docker Command Without Sudo (Optional)

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 kind
- Install minikube

Install kubectl on Linux

Download the latest release with the command:

$ 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.

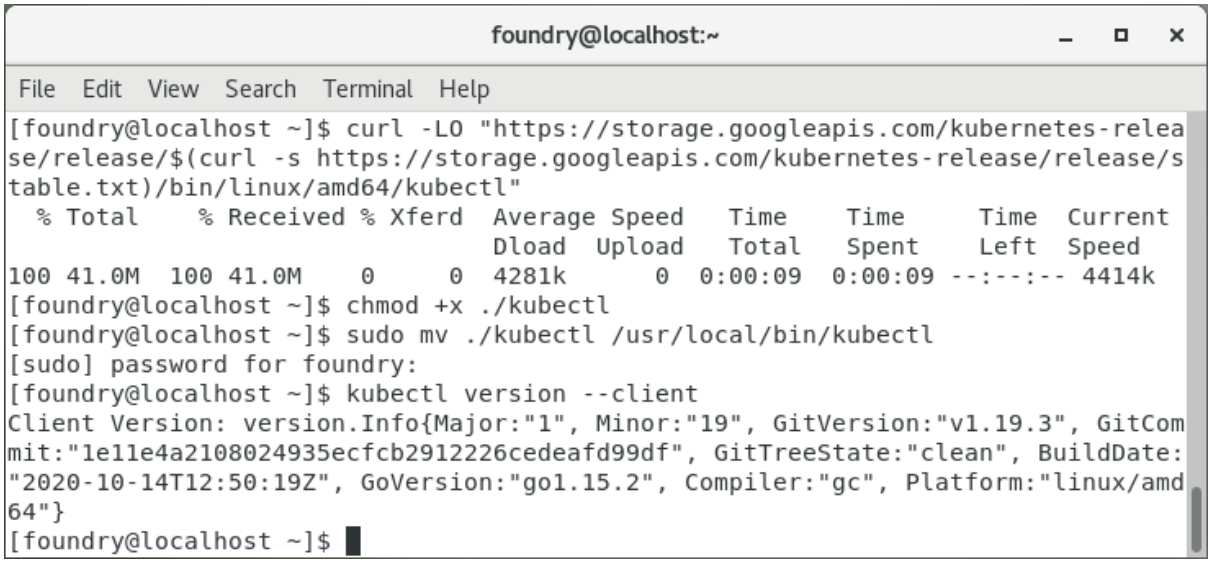
$ chmod +x ./kubectl

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 -o json



Install Kubernetes - Minikube

$ wget https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

$ chmod +x minikube-linux-amd64

$ sudo mv minikube-linux-amd64 /usr/local/bin/minikube

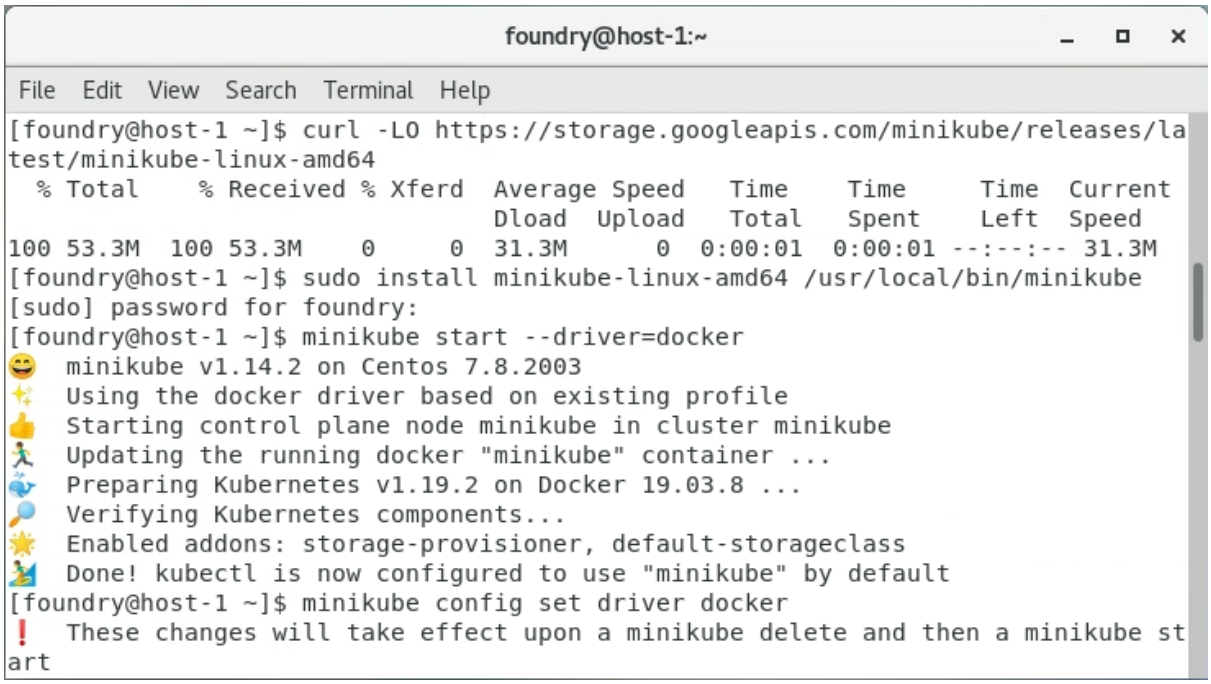
From a terminal with administrator access (but not logged in as root), run:

$ minikube start --driver=docker

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

Set a loadbalancer

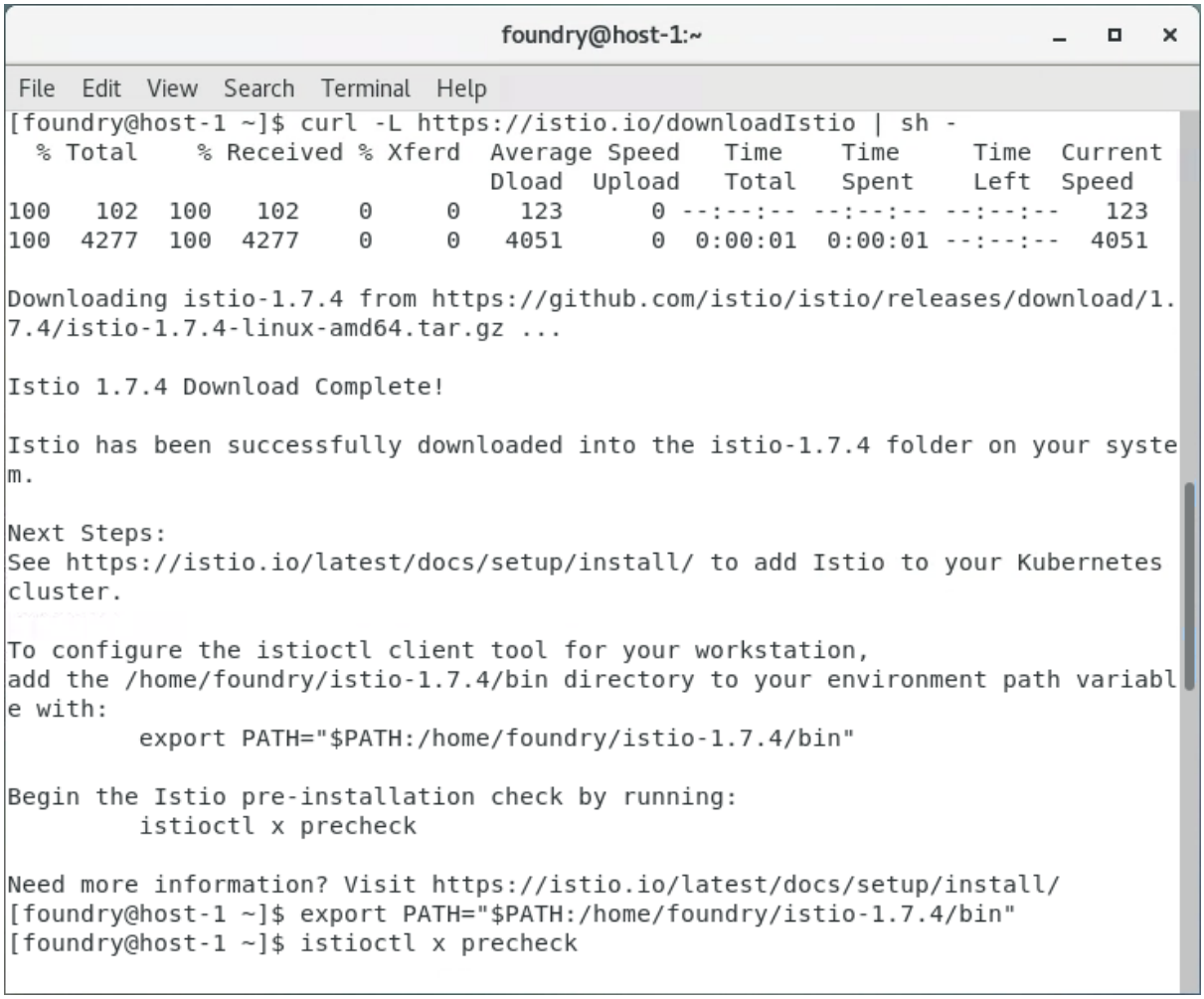
$ minikube tunnel

Install Istio

Download Istio

Go to the Istio release page 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 -



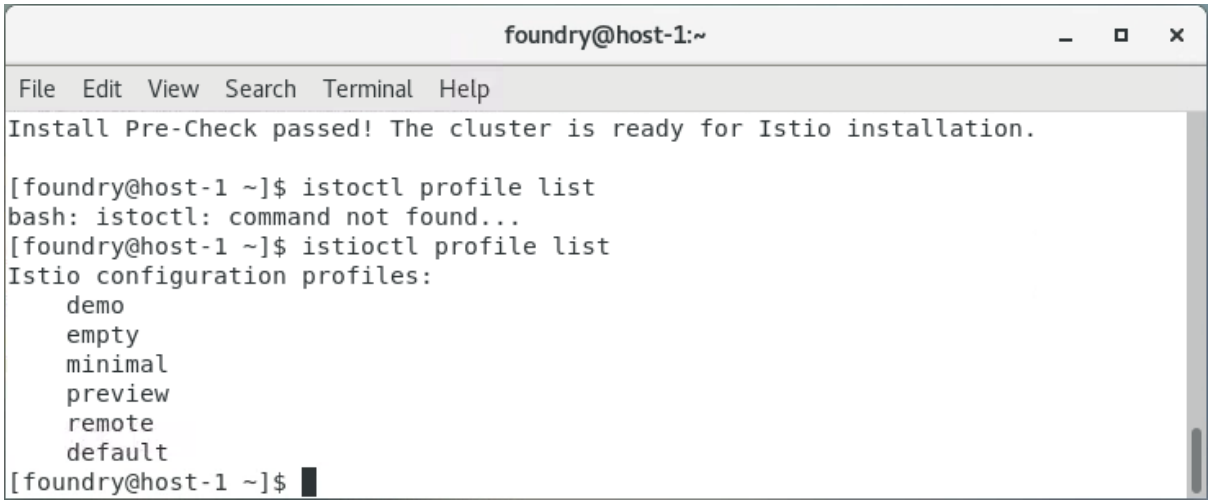
Add the istioctl client to your path

$ export PATH=”$PATH:/home/foundry/Istio-1.7.4/bin”

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 ~]$ istoctl profile list
bash: istoctl: command not found...
[foundry@host-1 ~]$ istioctl profile list
Istio configuration profiles:
demo
empty
minimal
preview
remote
default
[foundry@host-1 ~]$