How To Install and Use Docker on Ubuntu 16.04

Posted May 20, 2016 70.5k views Docker Ubuntu Ubuntu 16.04

Introduction

Docker is an application that makes it simple and easy to run application processes in a container, which are like virtual machines, only more portable, more resource-friendly, and more dependent on the host operating system. For a detailed introduction to the different components of a Docker container, check out The Docker Ecosystem: An Introduction to Common Components.

There are two methods for installing Docker on Ubuntu 16.04. One method involves installing it on an existing installation of the operating system. The other involves spinning up a server with a tool called Docker Machine that auto-installs Docker on it.

In this tutorial, you'll learn how to install and use it on an existing installation of Ubuntu 16.04.

Prerequisites

To follow this tutorial, you will need the following:

- 64-bit Ubuntu 16.04 Droplet
- Non-root user with sudo privileges Initial Setup Guide for Ubuntu 16.04 explains how to set this up.)

Note: Docker requires a 64-bit version of Ubuntu as well as a kernel version equal to or greater than 3.10. The default 64-bit Ubuntu 16.04 Droplet meets these requirements.

All the commands in this tutorial should be run as a non-root user. If root access is required for the command, it will be preceded by `sudo`. Initial Setup Guide for Ubuntu 16.04 explains how to add users and give them sudo access.

Step 1 — Installing Docker

The Docker installation package available in the official Ubuntu 16.04 repository may not be the latest version. To get the latest and greatest version, install Docker from the official Docker repository. This section shows you how to do just that.

But first, let's update the package database:

```
sudo apt-get update
```

Now let's install Docker. Add the GPG key for the official Docker repository to the system:

```
sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys 58118E89F3A912897C070ADBF76221572C52609D
```

Add the Docker repository to APT sources:

```
echo "deb https://apt.dockerproject.org/repo ubuntu-xenial main" | sudo tee /etc/apt/sources.list.d/docker.list
```

Update the package database with the Docker packages from the newly added repo:

```
sudo apt-get update
```

Make sure you are about to install from the Docker repo instead of the default Ubuntu 16.04 repo:

```
apt-cache policy docker-engine
```

You should see output similar to the follow:

Output of apt-cache policy docker-engine

```
docker-engine:
  Installed: (none)
  Candidate: 1.11.1-0~xenial
  Version table:
     1.11.1-0~xenial 500
        500 https://apt.dockerproject.org/repo ubuntu-xenial/main amd64 Packages
     1.11.0-0~xenial 500
        500 https://apt.dockerproject.org/repo ubuntu-xenial/main amd64 Packages
```

Notice that `docker-engine` is not installed, but the candidate for installation is from the Docker repository for Ubuntu 16.04. The `docker-engine` version number might be different.

Finally, install Docker:

```
sudo apt-get install -y docker-engine
```

Docker should now be installed, the daemon started, and the process enabled to start on boot. Check that it's running:

```
sudo systemctl status docker
```

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

```
Output
```

```
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2016-05-01 06:53:52 CDT; 1 weeks 3 days ago
     Docs: https://docs.docker.com
 Main PID: 749 (docker)
```

Installing Docker now gives you not just the Docker service (daemon) but also the `docker` command line utility, or the Docker client. We'll explore how to use the `docker` command later in this tutorial.

Step 2 — Executing the Docker Command Without Sudo (Optional)

By default, running the `docker` command requires root privileges — that is, you have to 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`.