Docker Components / Terminologies

There area a number of Docker specific jargon that we need to clarify before diving into installation and usage examples. Below are commonly used terminologies in Docker ecosystem.

* **Docker daemon**: This is also called Docker Engine, it is a background process which runs on the host system responsible for building and running of containers.
* **Docker Client**: This is a command line tool used by the user to interact with the Docker daemon.
* **Docker Image**: An image is an immutable file that’s essentially a snapshot of a container. A docker image has a file system and application dependencies required for running applications.
* **Docker container**: This is a running instance of a docker image with an application and its dependencies. Each container has a unique process ID and isolated from other containers. The only thing containers share is the Kernel.
* **Docker registry**: This is an application responsible for managing storage and delivery of Docker container images. It can be private or public.

Install Docker CE on RHEL 8 / CentOS 8

So far we have covered docker introduction and terminologies. We should be ready to install Docker CE on RHEL 8 / CentOS 8. We will start with the installation of Docker then Docker Compose.

There are two editions of Docker available.

* **Community Edition (CE)**: ideal for individual developers and small teams looking to get started with Docker and experimenting with container-based apps.
* **Enterprise Edition (EE)**: Designed for enterprise development and IT teams who build, ship, and run business-critical applications in production at scale.

The Docker Enterprise Edition requires an active license to use. In this guide, we will install Docker CE on RHEL 8. Let’s add Docker repository before we can install it.

**sudo curl** [**https://download.docker.com/linux/centos/docker-ce.repo**](https://download.docker.com/linux/centos/docker-ce.repo) **-o /etc/yum.repos.d/docker-ce.repo**

This command will download Docker repository file to /etc/yum.repos.d/docker-ce.repo. Let’s update RPM index cache.

$ **sudo yum makecache**   
Updating Subscription Management repositories.  
Updating Subscription Management repositories.  
Docker CE Stable - x86\_64 6.1 kB/s | 14 kB 00:02   
Red Hat Enterprise Linux 8 for x86\_64 - AppStream Beta (RPMs) 1.8 kB/s | 4.1 kB 00:02   
Red Hat Enterprise Linux 8 for x86\_64 - BaseOS Beta (RPMs) 3.5 kB/s | 4.1 kB 00:01   
Metadata cache created.

Finally install Docker CE by running the command below in your terminal.

sudo dnf -y install docker-ce --nobest

Start and enable Docker Service to start at boot.

sudo systemctl enable --now docker

The docker service status should indicate running.

$ **systemctl status docker**  
 ● docker.service - Docker Application Container Engine  
 Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)  
 Active: active (**running**) since Sun 2019-03-10 09:52:54 EAT; 29s ago  
 Docs: [https://docs.docker.com](https://docs.docker.com/)  
 Main PID: 2875 (dockerd)  
 Tasks: 21 (limit: 11510)  
 Memory: 48.3M  
 CGroup: /system.slice/docker.service  
 ├─2875 /usr/bin/dockerd  
 └─2882 docker-containerd --config /var/run/docker/containerd/containerd.toml  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.261477553+03:00" level=info msg="ClientConn switching balancer to \"pick\_first\"">  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.261506651+03:00" level=info msg="pickfirstBalancer: HandleSubConnStateChange: 0xc>  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.261707466+03:00" level=info msg="pickfirstBalancer: HandleSubConnStateChange: 0xc>  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.261722124+03:00" level=info msg="Loading containers: start."  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.637205789+03:00" level=info msg="Default bridge (docker0) is assigned with an IP >  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.835734889+03:00" level=info msg="Loading containers: done."  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.857760401+03:00" level=info msg="Docker daemon" commit=d7080c1 graphdriver(s)=ove>  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.858095579+03:00" level=info msg="Daemon has completed initialization"  
 Mar 10 09:52:54 rhel8.local dockerd[2875]: time="2019-03-10T09:52:54.915887172+03:00" level=info msg="API listen on /var/run/docker.sock"  
 Mar 10 09:52:54 rhel8.local systemd[1]: Started Docker Application Container Engine.

The docker group is created, but no users are added to the group. Add your user to this group to run docker commands without sudo.

$ **sudo usermod -aG docker $USER**

$ **id $USER**

uid=1000(jmutai) gid=1000(jmutai) groups=1000(jmutai),10(wheel),984(docker)

Logout and Login again to use Docker without sudo. The version of Docker installed can be checked with:

**$ newgrp docker**

$ **docker version**

Client:

Version: 18.06.3-ce

API version: 1.38

Go version: go1.10.3

Git commit: d7080c1

Built: Wed Feb 20 02:26:51 2019

OS/Arch: linux/amd64

Experimental: false

Server:

Engine:

Version: 18.06.3-ce

API version: 1.38 (minimum version 1.12)

Go version: go1.10.3

Git commit: d7080c1

Built: Wed Feb 20 02:28:17 2019

OS/Arch: linux/amd64

Experimental: false

#### Pull Test docker image

The last step is to test your installation by downloading a test docker container.

$ **docker pull alpine**

Using default tag: latest

latest: Pulling from library/alpine

8e402f1a9c57: Pull complete

Digest: sha256:644fcb1a676b5165371437feaa922943aaf7afcfa8bfee4472f6860aad1ef2a0

Status: Downloaded newer image for alpine:latest

List downloaded images.

$ **docker images**  
 REPOSITORY TAG IMAGE ID CREATED SIZE  
 alpine latest 5cb3aa00f899 2 days ago 5.53MB  
 hello-world latest fce289e99eb9 2 months ago 1.84kB

Verify that Docker CE is working correctly by running the alpine container from downloaded image.

$ **docker run -it --rm alpine /bin/sh**  
/ # **apk update**  
fetch <http://dl-cdn.alpinelinux.org/alpine/v3.9/main/x86_64/APKINDEX.tar.gz>  
fetch <http://dl-cdn.alpinelinux.org/alpine/v3.9/community/x86_64/APKINDEX.tar.gz>  
v3.9.2-1-g592d872fb8 [<http://dl-cdn.alpinelinux.org/alpine/v3.9/main>]  
v3.9.2-2-ge7dc3349a9 [<http://dl-cdn.alpinelinux.org/alpine/v3.9/community>]  
OK: 9754 distinct packages available  
/ # **exit**

## Install Docker Compose on RHEL 8 / CentOS 8

We have Docker installed and running, let’s now change gear to Docker Compose. Docker Compose is a tool used to define and run multi-container Docker applications.

The application services are configured using a Compose file. Unlike running docker commands, a single command is all that’s required to create and start all the services from your configuration. Docker Compose is a great tool for development, testing, and staging environments, as well as CI workflows.

### Download Docker Compose on RHEL 8 / CentOS 8

Follow our separate guide on installation of latest Docker Compose on Linux.

[Install Docker Compose on Linux](https://computingforgeeks.com/how-to-install-latest-docker-compose-on-linux/)

For the sake of keeping this guide brief, we won’t dive into Docker compose usage. I’ll recommend you go through Official[Docker documentation](https://docs.docker.com/) and [Docker Compose documentation](https://docs.docker.com/compose/" \t "_blank) to learn more.

## Install Docker UI – Optional

If you need Docker management UI which allows you to easily manage your different Docker hosts and containers, please give Portainer a try. It is easy to install and use.

[How to Install Portainer Docker UI manager](https://computingforgeeks.com/install-docker-ui-manager-portainer/)

## Monitoring Docker containers

Monitoring Docker containers can be achieved by using Monitoring tools such as Netdata or Ctop or Prometheus and Grafana. Below guides should be helpful.