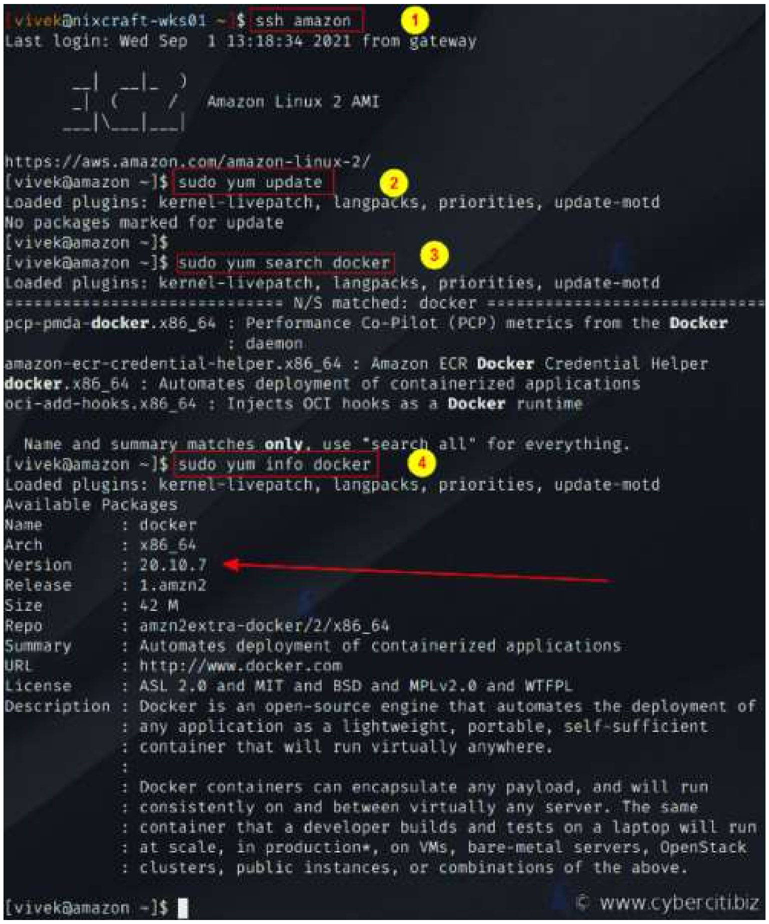
Installing Docker on Amazon Linux 2

The procedure to install Docker on AMI 2 (Amazon Linux 2) running on an EC2 instance is as follows:

1) Login into the remote AWS server using the ssh command or using putty:
   $ ssh ec2-user@ec2-ip-address-dns-name-here
2) Apply pending updates using the yum command:
   $ sudo yum update
3) Search for Docker package:
   $ sudo yum search docker
4) Get version information:
   $ sudo yum info docker

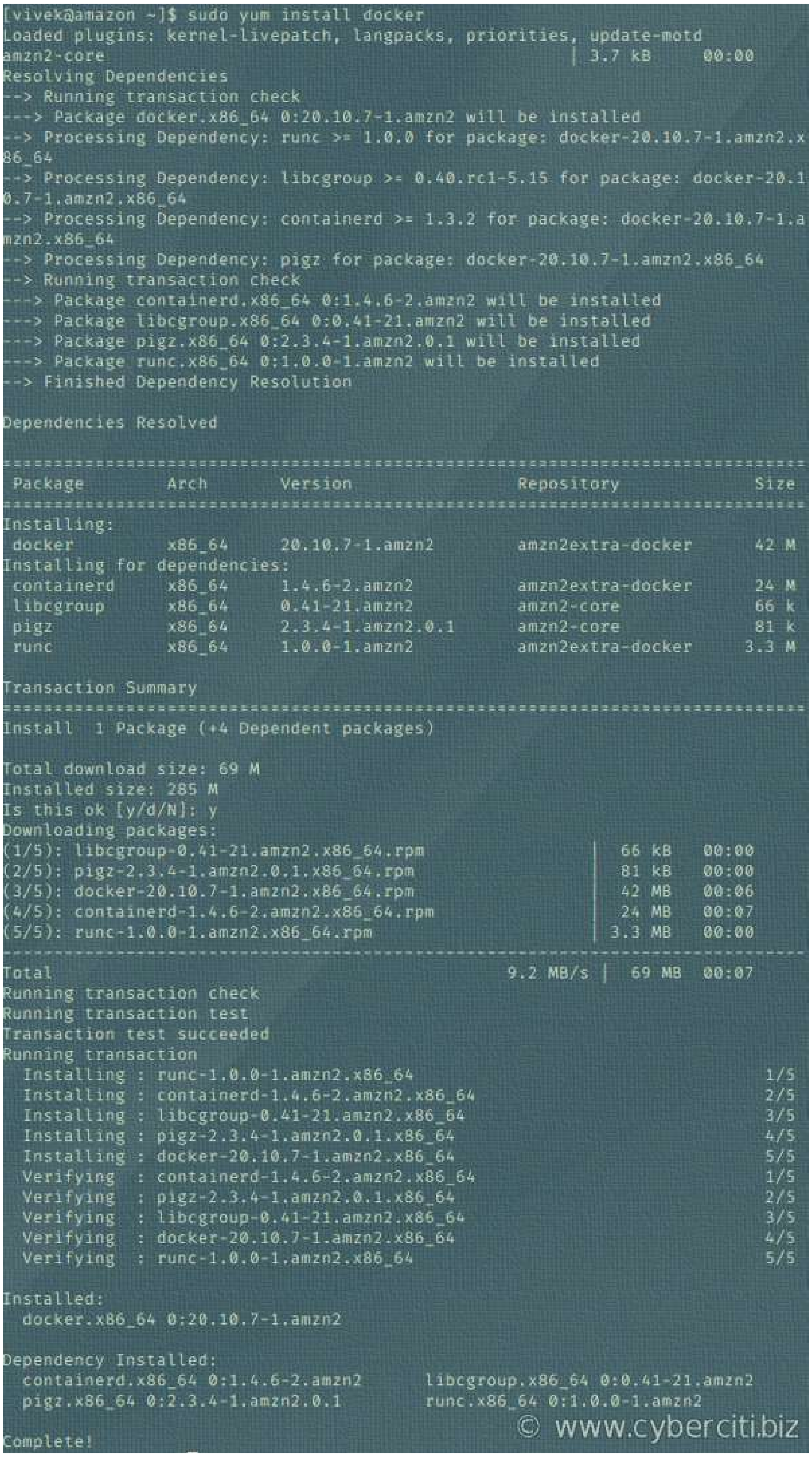


```
[vivek@nixcraft-wks01 ~ ]$ ssh amazon
Last login: Wed Sep  1 13:18:34 2021 from gateway

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[vivek@amazon ~]$ sudo yum update
Loaded plugins: kernel-livepatch, langpacks, priorities, update-motd
No packages marked for update
[vivek@amazon ~]$
[vivek@amazon ~]$ sudo yum search docker
Loaded plugins: kernel-livepatch, langpacks, priorities, update-motd
============================= N/S matched: docker ==============================
pcp-pmda-docker.x86_64 : Performance Co-Pilot (PCP) metrics from the Docker
                       : daemon
amazon-ecr-credential-helper.x86_64 : Amazon ECR Docker Credential Helper
docker.x86_64 : Automates deployment of containerized applications
oci-add-hooks.x86_64 : Injects OCI hooks as a Docker runtime

  Name and summary matches only, use "search all" for everything.
[vivek@amazon ~]$ sudo yum info docker
Loaded plugins: kernel-livepatch, langpacks, priorities, update-motd
Available Packages
Name        : docker
Arch        : x86_64
Version     : 20.10.7
Release     : 1.amzn2
Size        : 42 M
Repo        : amzn2extra-docker/2/x86_64
Summary     : Automates deployment of containerized applications
URL         : http://www.docker.com
License     : ASL 2.0 and MIT and BSD and MPLv2.0 and WTFPL
Description : Docker is an open-source engine that automates the deployment of
            : any application as a lightweight, portable, self-sufficient
            : container that will run virtually anywhere.
            :
            : Docker containers can encapsulate any payload, and will run
            : consistently on and between virtually any server. The same
            : container that a developer builds and tests on a laptop will run
            : at scale, in production*, on VMs, bare-metal servers, OpenStack
            : clusters, public instances, or combinations of the above.

[vivek@amazon ~]$
```

© www.cyberciti.biz

Getting Docker version

5) Install docker, run:
$ sudo yum install docker



```
[vivek@amazon ~]$ sudo yum install docker
Loaded plugins: kernel-livepatch, langpacks, priorities, update-motd
amzn2-core                                               | 3.7 kB     00:00
Resolving Dependencies
--> Running transaction check
---> Package docker.x86_64 0:20.10.7-1.amzn2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.7-1.amzn2.x
86_64
--> Processing Dependency: libcgroup >= 0.40.rc1-5.15 for package: docker-20.1
0.7-1.amzn2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.7-1.a
mzn2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.7-1.amzn2.x86_64
--> Running transaction check
---> Package containerd.x86_64 0:1.4.6-2.amzn2 will be installed
---> Package libcgroup.x86_64 0:0.41-21.amzn2 will be installed
---> Package pigz.x86_64 0:2.3.4-1.amzn2.0.1 will be installed
---> Package runc.x86_64 0:1.0.0-1.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

==============================================================================
 Package         Arch        Version              Repository             Size
==============================================================================
Installing:
 docker          x86_64      20.10.7-1.amzn2      amzn2extra-docker      42 M
Installing for dependencies:
 containerd      x86_64      1.4.6-2.amzn2        amzn2extra-docker      24 M
 libcgroup       x86_64      0.41-21.amzn2        amzn2-core             66 k
 pigz            x86_64      2.3.4-1.amzn2.0.1    amzn2-core             81 k
 runc            x86_64      1.0.0-1.amzn2        amzn2extra-docker     3.3 M

Transaction Summary
==============================================================================
Install  1 Package (+4 Dependent packages)

Total download size: 69 M
Installed size: 285 M
Is this ok [y/d/N]: y
Downloading packages:
(1/5): libcgroup-0.41-21.amzn2.x86_64.rpm                 |  66 kB   00:00
(2/5): pigz-2.3.4-1.amzn2.0.1.x86_64.rpm                  |  81 kB   00:00
(3/5): docker-20.10.7-1.amzn2.x86_64.rpm                  |  42 MB   00:06
(4/5): containerd-1.4.6-2.amzn2.x86_64.rpm                |  24 MB   00:07
(5/5): runc-1.0.0-1.amzn2.x86_64.rpm                      | 3.3 MB   00:00
------------------------------------------------------------------------------
Total                                            9.2 MB/s |  69 MB   00:07
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : runc-1.0.0-1.amzn2.x86_64                                  1/5
  Installing : containerd-1.4.6-2.amzn2.x86_64                            2/5
  Installing : libcgroup-0.41-21.amzn2.x86_64                             3/5
  Installing : pigz-2.3.4-1.amzn2.0.1.x86_64                              4/5
  Installing : docker-20.10.7-1.amzn2.x86_64                              5/5
  Verifying  : containerd-1.4.6-2.amzn2.x86_64                            1/5
  Verifying  : pigz-2.3.4-1.amzn2.0.1.x86_64                              2/5
  Verifying  : libcgroup-0.41-21.amzn2.x86_64                             3/5
  Verifying  : docker-20.10.7-1.amzn2.x86_64                              4/5
  Verifying  : runc-1.0.0-1.amzn2.x86_64                                  5/5

Installed:
  docker.x86_64 0:20.10.7-1.amzn2

Dependency Installed:
  containerd.x86_64 0:1.4.6-2.amzn2       libcgroup.x86_64 0:0.41-21.amzn2
  pigz.x86_64 0:2.3.4-1.amzn2.0.1         runc.x86_64 0:1.0.0-1.amzn2

Complete!
```

© www.cyberciti.biz

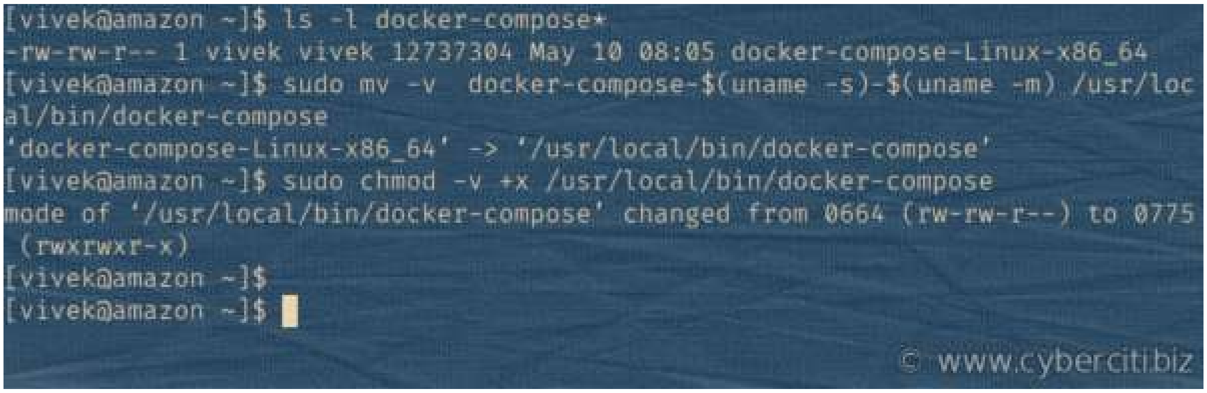
Amazon Linux 2: Install docker command

6) Add group membership for the default ec2-user so you can run all docker commands without using the sudo command:

```
$ sudo usermod -a -G docker ec2-user
$ id ec2-user
# Reload a Linux user's group assignments to docker w/o logout
$ newgrp docker
```

7) For docker-compose, try any one of the following commands:

```
# Get pip3
sudo yum install python3-pip

# Then run any one of the following
sudo pip3 install docker-compose # with root access

OR

pip3 install --user docker-compose # without root access for security reasons

OR

wget https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)
sudo mv docker-compose-$(uname -s)-$(uname -m) /usr/local/bin/docker-compose
sudo chmod -v +x /usr/local/bin/docker-compose
```



```
[vivek@amazon ~]$ ls -l docker-compose*
-rw-rw-r-- 1 vivek vivek 12737304 May 10 08:05 docker-compose-Linux-x86_64
[vivek@amazon ~]$ sudo mv -v  docker-compose-$(uname -s)-$(uname -m) /usr/loc
al/bin/docker-compose
'docker-compose-Linux-x86_64' -> '/usr/local/bin/docker-compose'
[vivek@amazon ~]$ sudo chmod -v +x /usr/local/bin/docker-compose
mode of '/usr/local/bin/docker-compose' changed from 0664 (rw-rw-r--) to 0775
 (rwxrwxr-x)
[vivek@amazon ~]$
[vivek@amazon ~]$ 
```

How to install docker-compose in Amazon Linux

8) Enable docker service at AMI boot time:

```
$ sudo systemctl enable docker.service
```

9) Start the Docker service:

```
$ sudo systemctl start docker.service
```

Verification

Now that both required software is installed, we need to make sure it is working. Hence, type the following commands.

Finding status

Get the docker service status on your AMI instance, run:

```
$ sudo systemctl status docker.service
```

Outputs:

```
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2021-09-08 05:03:52 EDT; 18s ago
     Docs: https://docs.docker.com
  Process: 3295 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
  Process: 3289 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
 Main PID: 3312 (dockerd)
    Tasks: 9
   Memory: 39.9M
   CGroup: /system.slice/docker.service
           └─3312 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/c...

Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local systemd[1]: Started Docker Applicatio...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Hint: Some lines were ellipsized, use -l to show in full.
```

Getting docker version info on Amazon Linux

- The docker-compose is installed in the ‘/usr/local/bin’ directory and may not be on your PATH settings. To see current PATH settings run the echo command/printf command:

  ```
  $ echo "$PATH"
  ```

- To add /usr/local/bin/ path to your bash startup file such as ~/.profile or ~/.bash_profile using the export command:

  ```
  $ export PATH=$PATH:/usr/local/bin
  ```

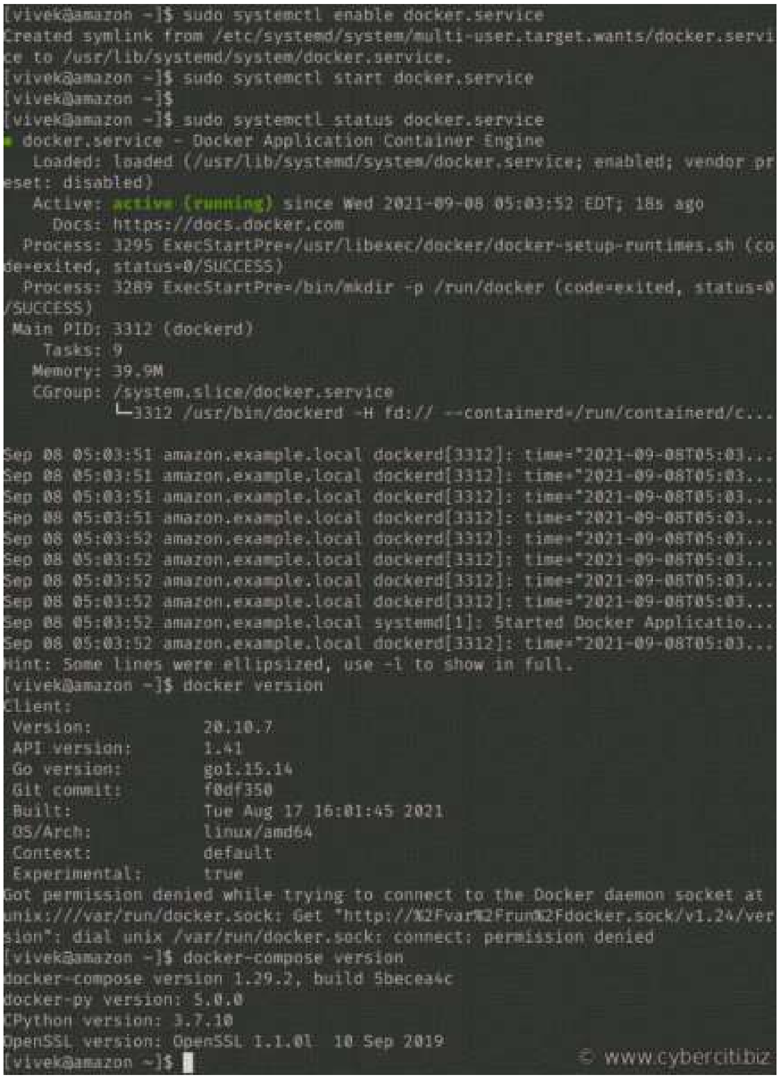
- Another option for developers is to use the find command to locate the docker-compose file as follows:

  ```
  $ sudo find / -name "docker-compose" -ls
  ```

See docker version:

$ docker version

Also verify that docker-compose install was a success on AMI 2 by running the following command:
$ docker-compose version



```
[vivek@amazon ~]$ sudo systemctl enable docker.service
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.servi
ce to /usr/lib/systemd/system/docker.service.
[vivek@amazon ~]$ sudo systemctl start docker.service
[vivek@amazon ~]$
[vivek@amazon ~]$ sudo systemctl status docker.service
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor pr
eset: disabled)
   Active: active (running) since Wed 2021-09-08 05:03:52 EDT; 18s ago
     Docs: https://docs.docker.com
  Process: 3295 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (co
de=exited, status=0/SUCCESS)
  Process: 3289 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0
/SUCCESS)
 Main PID: 3312 (dockerd)
    Tasks: 9
   Memory: 39.9M
   CGroup: /system.slice/docker.service
           └─3312 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/c...

Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:51 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Sep 08 05:03:52 amazon.example.local systemd[1]: Started Docker Applicatio...
Sep 08 05:03:52 amazon.example.local dockerd[3312]: time="2021-09-08T05:03...
Hint: Some lines were ellipsized, use -l to show in full.
[vivek@amazon ~]$ docker version
Client:
 Version:           20.10.7
 API version:       1.41
 Go version:        go1.15.14
 Git commit:        f0df350
 Built:             Tue Aug 17 16:01:45 2021
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true
Got permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/ver
sion": dial unix /var/run/docker.sock: connect: permission denied
[vivek@amazon ~]$ docker-compose version
docker-compose version 1.29.2, build 5becea4c
docker-py version: 5.0.0
CPython version: 3.7.10
OpenSSL version: OpenSSL 1.1.0l  10 Sep 2019
[vivek@amazon ~]$
```

© www.cyberciti.biz

Getting docker version on AMI using the ssh client

How to control docker service

Use the systemctl command as follows:

```
sudo systemctl start docker.service #<-- start the service
sudo systemctl stop docker.service #<-- stop the service
sudo systemctl restart docker.service #<-- restart the service
sudo systemctl status docker.service #<-- get the service status
```

Creating your first Docker project

Make a new project folder using the mkdir command and cd into it using the cd command. For instance:

```
$ mkdir static-website-1
$ cd static-website-1
```

Use the echo command as follows to create a new index.html for our project:

```
$ echo 'Docker Training Apache static site' > index.html
```

Make a new Dockerfile using a text editor such as nano command or vim command:

```
$ vim Dockerfile
```

Append the following config for your Amazon Linux container:

```
FROM rockylinux/rockylinux:latest

MAINTAINER DevOpsTrainer
LABEL Remarks="Linux test image for installing static webpage with Apache2"

# Install apache2 with less
RUN yum -y update && \
yum -y install httpd && \
yum clean all

# Sample index.html for test
COPY index.html /var/www/html/index.html

# Port and set entry point for container
EXPOSE 80
ENTRYPOINT /usr/sbin/httpd -DFOREGROUND
```

Build docker:

```
$ sudo docker build -t staticsite01 .
```

Outputs:

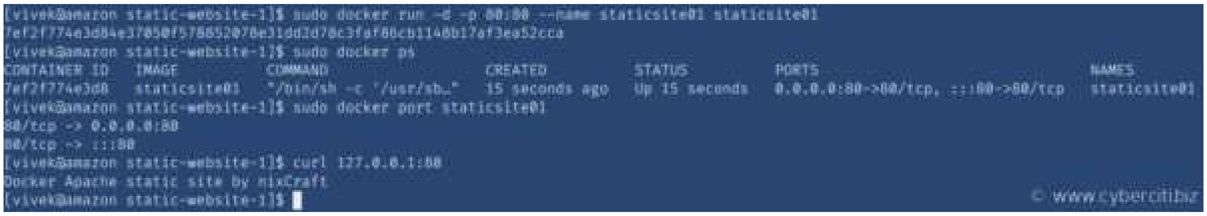
```
Sending build context to Docker daemon  3.072kB
Step 1/7 : FROM rockylinux/rockylinux:latest
latest: Pulling from rockylinux/rockylinux
ecce7a433753: Pull complete
Digest: sha256:98dcf3fbe75741058c16ece621f5917e0ff52d9333073e6389c5de8efaa3d5c4
Status: Downloaded newer image for rockylinux/rockylinux:latest
 ---> 86f02aa837b3
Step 2/7 : MAINTAINER nixCraft
 ---> Running in 7f4f35c8d95a
Removing intermediate container 7f4f35c8d95a
 ---> e40cd8411b69
Step 3/7 : LABEL Remarks="CentOS 8 test image for installing ng with Apache2"
 ---> Running in 31bf348db2fb
Removing intermediate container 31bf348db2fb
 ---> 28accfe0f9ff
Step 4/7 : RUN yum -y update && yum -y install httpd && yum clean all
 ---> Running in f588730a294f
Rocky Linux 8 - AppStream                       6.2 MB/s |  10 MB     00:01
Rocky Linux 8 - BaseOS                          6.7 MB/s | 7.7 MB     00:01
Rocky Linux 8 - Extras                           59 kB/s |  12 kB     00:00
Dependencies resolved.
================================================================================
 Package                 Arch      Version                    Repo       Size
================================================================================
Upgrading:
 gzip                    x86_64    1.9-13.el8_5               baseos    166 k
 libreport-filesystem    x86_64    2.9.5-15.el8.rocky.6.3     baseos     20 k
 openssl-libs            x86_64    1:1.1.1k-6.el8_5           baseos    1.5 M
 vim-minimal             x86_64    2:8.0.1763-16.el8_5.13     baseos    574 k
 zlib                    x86_64    1.2.11-18.el8_5            baseos    101 k
Installing dependencies:
 openssl                 x86_64    1:1.1.1k-6.el8_5           baseos    708 k
Installing weak dependencies:
 openssl-pkcs11          x86_64    0.4.10-2.el8               baseos     65 k

Transaction Summary
================================================================================
Install  2 Packages
Upgrade  5 Packages

Total download size: 3.1 M
.....
..
.....
  rocky-logos-httpd-85.0-3.el8.noarch

Complete!
27 files removed
Removing intermediate container f588730a294f
```

List images:

```
$ sudo docker images
REPOSITORY            TAG      IMAGE ID        CREATED          SIZE
staticsite01          latest   51c5c08cf14d   3 minutes ago   232MB
rockylinux/rockylinux  latest   86f02aa837b3   6 weeks ago      205MB
```

Run docker:

```
$ sudo docker run -d -p 80:80 --name staticsite01 staticsite01
$ sudo docker ps
$ sudo docker port staticsite01
$ curl 127.0.0.1:80
```



```
[vivek@amazon static-website-1]$ sudo docker run -d -p 80:80 --name staticsite01 staticsite01
7ef2f774e3d84e37050f578852070e31dd2d70c3faf80cb1140b17af3ea52cca
[vivek@amazon static-website-1]$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                                 NAMES
7ef2f774e3d8   staticsite01   "/bin/sh -c '/usr/sb…"   15 seconds ago   Up 15 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   staticsite01
[vivek@amazon static-website-1]$ sudo docker port staticsite01
80/tcp -> 0.0.0.0:80
80/tcp -> :::80
[vivek@amazon static-website-1]$ curl 127.0.0.1:80
Docker Apache static site by nixCraft
[vivek@amazon static-website-1]$ 
```

© www.cyberciti.biz

Summing up

That is all for now. You learned how to install Docker on AMI 2 and deploy Apache 2 as the Docker container for a static website. Use the following command to get an overview of available commands:

```
$ docker help
$ docker --help
```

For specific client examples please see the man page for the specific Docker command using the man command. For instance:

```
$ man docker-build
$ man docker-run
```