httpd in docker

How to Setup a Simple Apache Web Server in a Docker Container

If you are a Linux system administrator who provides support for developers, chances are you've heard of **Docker**. If not, this software solution will make your life easier beginning today by helping you reduce operating costs and accelerate deployments – among other benefits.

But it's not magic. **Docker** as a platform leverages containers – packages of an application along with all the tools it needs to run to eliminate differences between environments.

In other words, containerized software will operate and can be managed consistently regardless of where it is installed. Additionally, containers are much easier to set up, start, stop, and maintain than good old virtual machines.

If you're interested in knowing more about the differences between these two technologies, the official Docker website provides a great explanation.

To illustrate, in this article we will explain how to install **Docker** on **CentOS/RHEL**, **Rocky/Alma Linux**, and **Debian/Ubuntu**, and spin up an Apache **2.4** container from **Docker Hub**.

Setting Up an Apache Container

One of the amazing things about the **Docker** ecosystem is that there are tens of standard containers that you can easily download and use.

In the following example, we will instantiate an **Apache 2.4** container named **tecmint-web**, detached from the current terminal. We will use an image called **httpd:2.4** from **Docker Hub**.

Our plan is to have requests made to our public IP address on port **8080** be redirected to port **80** on the container. Also, instead of serving content from the container itself, we will serve a simple web page from **/home/user/website**.

We do this by mapping **/home/user/website/** on the **/usr/local/apache2/htdocs/** on the container. Note that you will need to use **sudo** or login as **root** to proceed, and do not omit the forward slashes at the end of each directory.

```
# sudo docker run -dit --name tecmint-web -p 8080:80 -v /home/user
/website/:/usr/local/apache2/htdocs/ httpd:2.4
```

Pull Docker Apache Container

```
root@tecmint:~# docker run -dit --name tecmint-web -p 8080:80 -v /home/user/website/
Unable to find image 'httpd:2.4' locally
2.4: Pulling from library/httpd
e5ae68f74026: Pull complete
bc36ee1127ec: Pull complete
d3576f2b6317: Pull complete
f1aa5f54b226: Pull complete
aa379c0cedc2: Pull complete
Digest: sha256:fba8a9f4290180ceee5c74638bb85ff21fd15961e6fdfa4def48e18820512bb1
Status: Downloaded newer image for httpd:2.4
d217a487a197ae3c17f9979122dd71c7914db3a2f26391da48a7ea764634bf46
```

At this point, our **Apache** container should be up and running.

```
$ sudo docker ps
```

Check Apache Docker Container

Now let's create a simple web page named **docker.html** inside the **/home/user/website** directory.

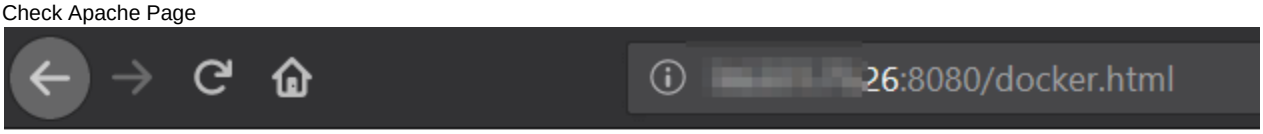
```
# vi /home/user/website/docker.html
```

Add the following sample HTML content to the file.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Learn Docker at Tecmint.com</title>
</head>
<body>
    <h1>Learn Docker With Us</h1>
</body>
</html>
```

Next, point your browser to **Server-IP:8080/docker.html** (where **Server-IP** is your host's public IP address). You should be presented with the page we created previously.

Check Apache Page



Learn Docker With Us

If you wish, you can now stop the container.

```
$ sudo docker stop tecmint-web
```

and remove it:

```
$ sudo docker rm tecmint-web
```

To finish cleaning up, you may want to delete the image that was used in the container (omit this step if you're planning on creating other **Apache 2.4** containers soon).

```
$ sudo docker image remove httpd:2.4
```

Note that in all the above steps we never had to install the webserver on our host.

Summary

In this article, we explained how to install **Docker** and manipulate a container. Unfortunately, these are just the basics – there are entire courses, books, and certification exams that cover **Dockers** (and containers in general) more in-depth.

If you want to learn more about **Docker**, we have already covered a 3-article series, that explains how to install Docker, run applications into containers, and automatically build docker images with dockerfile.