LAMP using Docker Part I

published by admin on Sat, 04/11/2015 - 20:33

Putting the LAP in LAMP

Assuming you have Docker installed and have a docker user within a docker group, you're ready to set up a Linux-Apache-MySQL-PHP (LAMP) Server.

As root, launch a container from the ubuntu image (not base):

$ sudo docker run -i -t ubuntu \bin\bash

In the launched container, install PHP and Apache2.

apt-get install php5
apt-get install apache2

Assuming everything goes well, exit the container and view the list of containers with "sudo docker ps -a".

Find the container ID for the container you just used by the Image Name, Command executed (/bin/bash), and the time Created. Save the container as a new image using:

$ sudo docker commit [Hex Container ID] LAMP

Now, using the lower credentialled user "docker", you can launch a container of the LAMP image and shunt Port 80 from the container to the host:

$ sudo su - docker
$ sudo docker run -i -t -p :80 LAMP /bin/bash # service apache2 start

You can observe the IP Address in the Apache2 output - or - open another terminal and run:

$ sudo docker inspect [Hex Container ID]

Now, launch a browser and go to the container's IP Address. If Apache is running, you'll get the "It Works!" message in your browser.

Keeping The Container Running

When you exit the container launched with "/bin/bash", the container closes fully and can only be recalled by looking for its ID via:

$ sudo docker ps -a

and starting it via:

$ sudo docker start [Hex Container ID] $ sudo docker attach [Hex Container ID]

Unfortunately, you then need to manually start apache2 and keep the related terminal open. Alternatively, you can start - and keep running - a container from the LAMP image using:

$ sudo docker run -d -p 80:80 LAMP /usr/sbin/apache2ctl -D FOREGROUND

Now, if you execute:

$ sudo docker ps

without the "-a" flag (which calls all containers including those turned off), then you'll see something akin to:
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
db5eedad266b LAMP:latest /usr/sbin/apache2ctl 2 seconds ago Up 2 seconds 0.0.0.0:80->80/tcp hungry_hawkings

Going to 0.0.0.0 in a browser should now show the "It Works!" Apache page. All without having to keep the terminal open.

There's Got To Be A More Elegant Solution

And, of course, there is. The awesome guys at Docker.io provide a construct called Dockerfile. Using the above guidance, a Dockerfile can be generated.

```
# Apache-PHP of a LAMP Server.
# Version 0.1
FROM ubuntu:12.04
MAINTAINER Last Mile Synergy, LLC

RUN dpkg-divert --local --rename --add /sbin/initctl
RUN ln -s /bin/true /sbin/initctl
RUN apt-get update && apt-get install -y apt-utils && apt-get install -y apache2 && apt-get install -y php5 && apt-get autoclean && apt-get autoremove

ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2

EXPOSE 80

CMD ["/usr/sbin/apache2", "-D", "FOREGROUND"]
```

Save the file as Dockerfile in its own directory. At this time, that is the only filename recognized by Docker.

One note about the above Dockerfile: in order to execute the upstart initialization files, it is necessary to do some minor linking within the image. Thus the need for the first two RUN commands.

Now, cd to the directory where your Dockerfile is and execute the Docker command to build an image from your Dockerfile.

$ sudo docker build -rm -t apache-php .

When the build is complete, all of the intermediate containers should be deleted (due to the "rm" attribute) and a new image named "LAP" should appear:

REPOSITORY TAG IMAGE ID CREATED VIRTUAL SIZE
apache-php latest fc78709edde1 16 minutes ago 245.4 MB

Now, let's fire up a running container from our new image:

$ sudo docker run -d apache-php

Check out the running containers using:

$ sudo docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
8fa284e3d230 apache-php:latest /usr/sbin/apache2 -D 10 seconds ago Up 9 seconds 80/tcp
stoic_poincare

Now, when you inspect the running container, you can find the IP address:

$ sudo docker inspect 8fa284e3d230

"NetworkSettings": {
"IPAddress": "172.17.0.33",
"IPPrefixLen": 16,
"Gateway": "172.17.42.1",
"Bridge": "docker0",
"PortMapping": null,
"Ports": {
"80/tcp": null
}
}

Open a browser and go to http://172.17.0.33 [1] to see "It Works!"

Source URL: http://blackhillsystems.com/?q=node/49

Links
[1] http://172.17.0.33