Romin Irani Follow
My passion is to help developers succeed. ¯_(ツ)_/¯
Jul 27, 2015 · 4 min read

Docker Tutorial Series : Part 6 : Docker Private Registry

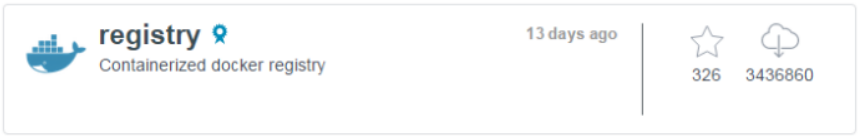
This is part 6 of the Docker Tutorial Series.

In this part we shall take a look at how you can host a local Docker registry. In an earlier part, we had looked at the Docker Hub, which is a public registry that is hosted by Docker. While the Docker Hub plays an important role in giving public visibility to your Docker images and for you to utilize quality Docker images put up by others, there is a clear need to setup your own private registry too for your team/organization.

Pull down the Registry Image

You might have guessed by now that the registry must be available as a Docker image from the Docker Hub and it should be as simple as pulling the image down and running that. You are correct!

A Docker Hub search on the keyword 'registry' brings up the following image as the top result:



Assuming that you have **boot2docker** running, execute the standard command to pull down the registry image.

```
docker@boot2docker:~$ docker pull registry
```

This will pull down the '**latest**' registry image and once it is pulled successfully, you should be able to see that in via the docker images command.

Run the local Registry

The registry Docker image is configured to start on port 5000 in the container, so we will expose the host port also as 5000.

You can launch the registry via the following command:

```
docker@boot2docker:~$ docker run -d -p 5000:5000 --name
localregistry registry
```

On successful launch, it will print the Container ID on the console.

To recap, we are starting a container named '**localregistry**' based on the '**registry**' image. The container is started in detached mode and the host:container port mapping has been done for both the port numbers to be 5000.

Check if the our container named '**localregistry**' has started via the **docker ps** command as shown below:

```
docker@boot2docker:~$ docker ps
```

The STATUS column should indicate that it is up.

Pull down a few images and push to local Registry

Now, let us pull down a few images first and then push them into the local Registry. Let us do that in 2 steps:

Step 1 : Pull down busybox and alpine Linux Images

Execute the pull commands for the following two images as shown below:

```
docker@boot2docker:~$ docker pull busybox
```

```
docker@boot2docker:~$ docker pull alpine
```

Once the images have been pulled down, verify that they are present in your Images list via the docker images command.

Quick Exercise:

If at this point, we try to pull the alpine image from our local registry, it should ideally respond with the image not found, shouldn't it ? Yes, it will. Remember that we have already started the registry container on port 5000. The result of this operation is shown below:

```
docker@boot2docker:~$ docker pull localhost:5000/alpine
Pulling repository localhost:5000/alpine
Error: image alpine:latest not found
```

Notice that the format for specifying the Image in a specific registry is as:

```
[REGISTRY_HOSTNAME:REGISTRY_PORT]/IMAGENAME
```

For public Docker Hub, we were not specifying the **[REGISTRY_HOSTNAME:REGISTRY_PORT]** option. But for our local registry we need to specify that so that the docker client will look there.

Step 2 : Push busybox and alpine Linux Images into the local Registry

Now, we will push the two images that we downloaded (**busybox** and **alpine**) into the local registry. Remember that these are two images that you downloaded directly from the Docker Hub. We have not modified it in any way. But you could create your own modified image (as we saw earlier in this series) and and push that into the local Registry too.

The step to push your image into the local Registry is done as follows:

- The first step is to take your image or containerLet us work with the alpine image that we have pulled earlier. The fully qualified name for this image is **alpine:latest** if you do a docker images command.Execute the following command to tag the alpine:latest image with the tag of the local registry to which we are going to push it.

- docker@boot2docker:~$ docker tag alpine:latest localhost:5000/alpine:latest

If you now run a docker images command, you will see both the **alpine** and the **localhost:5000/alpine** images listed.

The next step is to push this tagged image or container into the local registry.

This is done via the standard docker push command that you saw earlier. All we have to do is use the new tagged **localhost:5000/alpine** image. The command is given below:

```
docker@boot2docker:~$ docker push
localhost:5000/alpine:latest
The push refers to a repository [localhost:5000/alpine]
(len: 1)
Sending image list
Pushing repository localhost:5000/alpine (1 tags)
31f630c65071: Image successfully pushed
Pushing tag for rev [31f630c65071] on
{http://localhost:5000/v1/repositories/alpine/tags/lates
t}
```

Search for Images in local Registry

Now that we have pushed the alpine image into the local registry, we can even search for it. Just like how we used the docker search command to search for repositories in Docker Hub, we can use the same command for the local registry too, except that we will tag it with the registry host name and the port.

The syntax for docker search is of the following format:

```
docker search [my.registry.host]:[port]/library
```

To search for the alpine image, we can give the following command:

```
docker@boot2docker:~$ docker search
localhost:5000/alpine
NAME            DESCRIPTION   STARS     OFFICIAL
```

```
AUTOMATED
library/alpine                  0
```

You should now be in a position to create your own private registry. Other Docker clients can now pull images from this registry.

Further Reading

In this part, we looked at running a private registry. This was a registry that did not enforce any authentication on the part of the client. While this is useful from an understanding point of view, keep in mind that you should still follow the best practices for running a secure registry. It is not just about running a secure registry but also about making decisions around storage options for your private registry repositories. Towards this, I suggest to read up on this article.