Building Docker Images Without Being Root In The Image

I love being sudo (superuser, root) in Linux. It makes me feel powerful and unrestrained! But being sudo, and requiring sudo all the time, is NOT always a good thing. I noticed that Mosh from "Code with Mosh" does NOT use sudo in his Docker commands. At a certain point in the progression of his course, I cannot do what he does in the containers he runs *from his images*.

He makes a great case too for setting up Docker *images* to NOT require root permission for general security reasons. I agree with him. I am pretty sure that he is working in Mac OS. But DANG, most Linux users use sudo with Docker command lines. Thus, I thought that I might need to determine how to use Docker commands WITHOUT sudo. This may be hard and confusing. I want to find the straight path and bring some light into this situation to burn away the fog.

In your search to use Docker in Linux without sudo, you might come across "Post-installation steps for Linux" on the main Docker site. What they explain there does not make it so that you are NOT root. It only makes it so that you do not need to do "type" sudo. At least they warn us about this. There are other links that will also lead you down this "don't have to type sudo" path. This page on the AskUbuntu area of StackExchange explains things very well with regard to this issue.

But the REAL concern is that our containers do NOT operate as root. There's a difference between who the user is on your computer that is running Docker, and who the user is in a container. So you create a non-root user in the image for the container and make sure that user can do what's needed in the container. How did this become a concern? When trying to `RUN npm install` when building the image, it complains about not having adequate permissions. What must be recognized is that this happens when trying to run `npm install` on your local computer too. IF you do not setup nodejs operations for non-sudo use, you will encounter the same issues each time you `init` a new node instance.

Thus, the trick in this exercise is to accomplish an `npm install` in the image build without being root. I don't want to reveal the bulk of Mosh's OUTSTANDING video course in this document, because his course is worth what he charges. However, I believe this post will help those who've purchased his course around this issue that seems to only occur in Linux - at least for me.

Getting Node Packager Manager (NPM) To Install Without Being Root

A good guide is provided by John Papa that is called Node and npm without sudo. PLEASE look through that guide and apply it to your own Linux box OR your own Linux virtual machine. If you follow that guide, you will be able to initialize new `npm` instances for your nodejs projects without being sudo / root. This is a good to have setup. However, doing so on your own machine that you control and are using for development is not so crucial. I mean, it's a great practice to get into, but we need this most in the actual released application.

What we *really* need to figure out is *HOW to do this during the build of a Docker image.* We follow Mosh's Docker course directions up to the point of the `RUN npm install` statement. What do we need to do before that RUN command in the Dockerfile? Something pretty close to what John Papa did in the article previously shared.

Having successfully done this on my Linux machine for `npm install`, I know it can be done. However, base images in Docker are NOT all the same. The exact implementation may change a bit from base image to base image. However, the nice thing is that they are still some form of Linux. So with a bit of patience, we can find the needed changes for each implementation.

I'll be trying to implement this philosophy of being root as little as possible moving forward. What I mean is that other things may need to be done that don't require root in our images. When I find ways to do this with each new step, I will point those out specifically.

Summary

Being more security minded with respect to the way that we build our images is a good mentality to have. Going forward with making images, this will require a bit of research for each new base image and each new package we use in our images. Thus, following this good philosophy will be an ongoing practice that we'll want to help each other do.