Virtualization and Docker

I use Virtualization and Docker to better utilize the computing power of my server. While Virtualization and Docker are similar in function, I originally had a lot of trouble trying to learn Docker by abstracting it in terms of Virtualization. Once I realized this I made the choice to just create a new "Object" in my head for Docker instead of trying to make it fit into virtualization. Also, both technologies can and should be used concurrently to get the most from a computer given the current software and hardware landscape.

Most of the services I host are based in docker containers now, but I still have about a dozen VMs for various things, including one larger VM that is my docker host. I have assigned the docker host about 50% of all available compute on the main server, and this VM hosts several dozen different containers for different purposes.

Naturally this could all be accomplished by using complete VMs for each different service that docker runs. This would require a lot more overhead, since each VM would need to be provisioned and configured from scratch for each service. Docker can be thought of in some ways as a platform that automates the creation of VMs while leaving out the bits that aren't needed. That said, it isn't actually this, it is just a murky analogy.

Docker can also be thought of as an application like Adobe Acrobat Reader, where PDF files are the containers. The PDF file can be moved from system to system and always look the same. Similarly, a properly configured docker container can be moved from system to system by just moving the files around. Docker also uses a routing network stack, and the containers have their own networks, and ports need to be opened to the host for services to be visible.

A docker container can also be thought of as an application that runs and quits. Specifically, a docker container is built using components and command just to run a final single command and close out. Usually, the command or process is placed in some kind of looping script to keep it running. The point of this is to ensure a consistent platform under which to run the command. In other words, a docker container can be thought of as all the components needed to run one specific command. This command could be something as simple as echo-ing a string to the terminal, or hosting an entire cloud storage platform. The important part is that the bulk of the configuration of a docker container is instructions on how to build the container with all the bits needed for the single command to run successfully.

Given all these different analogies that are applicable to docker, you can see why trying to abstract this as like virtualization but different is not entirely adequate to describe how it works. I find it helpful to just think of a docker container as a whole new thing in my mind instead. One of the things I did on my path to understanding docker was to migrate my password generating website over to docker. That site was originally just some php code connecting to a MySQL server all running on my old web server VM. To get this into docker, I had to configure a container that would have all the web hosting components, the database components, and commands to import the

database into the container. Once I made it out the other end of that process, I had a much stronger understanding of docker, and as time has passed I have learned more nuances of the platform.

Revision #3
Created 5 April 2024 18:10:06 by Matt Anderson
Updated 11 July 2025 05:08:21 by Matt Anderson