Blog Topic 2: Docker

Imagine for a moment that you are Santa Claus and you need to deliver thousands of toys to little people across the globe. You have a problem though: the elves gave the toys some serious prerequisites that they expect the children to have already. Across millions of children in a million homes, you begin to realize that not every house is equipped for these highly specialized but awesome toys. So how do you solve the problem? Have each kid get their home set up properly for your one toy? Well, that could break things for another plaything and who knows if their parents could even afford it. However, what if you could provide each child with a special 'play compartment' that had everything they would need for their toy without affecting any of the other toys in the home? This would allow for greater reach of the toys' enjoyment – no longer a privilege that can afford a specialized setup.

This is essentially the issue that Docker sought to solve for software development. Programs have a ton of dependencies that they rely on to function. This poses a massive issue when you are working on an app. It's nearly impossible to ensure that your program functions on any platform without adversely affecting other applications. As such, developers were seeking a way to deliver a tidy package containing all of the dependences that didn't require complex hypervisors and virtual machines to create the proper environments. Docker provided the solution in the form of containers.

Containers revolutionized the industry. By creating an environment that is abstracted from the host operating system, containers introduced a simple way to create a standard unit of one's application, holding the application's code and all requisite dependencies, without impacting the other components of the system. Moreover, recovery was made far easier as one only had to issue the container rather than reforging the entire system in the exact same way. Functional recovery was as easy as importing the proper image and populating it with your information. The implementation was so widespread and integral to development that everyone from Google to Netflix to Saleforce adopted the platform and continues to use.

Docker was not made in a vacuum. Development of the software is still underway. The Docker corporation is primarily responsible for the maintenance and development of Docker. They have recently developed a couple of different 'flavors' of the platform, allowing for lighter weight or more secure implantations. As a major platform across the industry, many big names contribute as partners. For example, Red Hat, VMWare, and AWS are all verified publishers.

Despite its dominance, several competitors to Docker attempt to steal some of the market share. These platforms are typically reserved for niche use cases. Some are geared toward lightweight use cases, such as RunC. Others, like Podman, are targeted toward Linux systems and meant operate without certain resource, e.g. a daemon. However, with its overwhelming market share and dearth of applications already running on the platform, most companies still gravitate toward Docker.

For examples of applications using this platform, one doesn't need to stray too far from the first Google result. Github has a curated page from Docker containing sample

applications. For those interest, that information can be found here: https://github.com/dockersamples. In my limited experience, I have used Docker to run a network-wide ad-blocking application called Pihole. This minimized the amount of effort required to implement the application. I was able to pull the image and 'build' the container in approximately 10 lines of code. That project can be found here: https://github.com/pi-hole/docker-pi-hole. With thousands of free applications using the platform, anyone interested in Docker can get started quickly.

For those needing a bit more background or looking to develop on the platform, Docker is thoroughly documented. The company's official website, https://www.docker.com/, has everything from the history of the organization to 'Getting Started' information geared toward developers. For those looking for a course, every major online curriculum platform will have a course on Docker. My personal favorite, Udemy, has over 2500 relevant courses. Over 40 of these courses will prepare you to become a Docker Certified Associate. Again, one can find a near endless quantity of resources for the development and implementation of Docker containers.

Docker is an industry standard and was revolutionary at its outset. The platform is maintained actively with more and more partners joining every year. Even if a tech beginner is interested, thousands of projects will get you off the ground. For more advanced technophiles, documentation is plentiful and robust. Independent of someone's employment, this platform has had a massive impact on their life and will continue do so.