EN.605.609.81.SP25 DevOps and Secure Software Development

Assignment #7

Akin Williams

March 10, 2025

1. Vagrant vs. Docker

Vagrant and Docker are both tools that help manage environments in development workflows, but they do so in fundamentally different ways.

Vagrant is essentially a virtual machine manager. It can use type 1 and type 2 hypervisors to create and manage virtual machines that represent complete operating systems. The benefits of Vagrant are its ability to create an isolated environment that meets exact specifications down to the kernel level. The disadvantages of Vagrant are that each virtual machine consumes significant resources (disk, memory). Vagrant uses scripted configurations to set up a virtual machine. It is used primarily when a specific environment needs to be mirrored or when multiple operating systems are used for testing.

Docker employs a higher level of virtualization to create containers that run applications and dependencies on a host operating system. Docker containers are not fully isolated; instead, they share their host operating system's kernel. Docker containers are much lighter on resources; however, application performance, and perhaps behavior, may depend on the host system. Docker also uses scripted configurations to create images that are used to run containerized applications. It is used when fast environment provisioning and fewer resources are more important than 100% faithful environment replication.

For most use cases, I prefer Docker over Vagrant. The trade-off between Vagrant's kernel-level fidelity and Docker's speed and ease of use weighs heavily in Docker's favor. Docker's ability to share host resources means that provisioning many containers on the same system is quite feasible and easy to manage, compared to Vagrant where resource allocation must be managed carefully in order to avoid exceeding the host resource capacity. Also, spinning up containers takes considerably less time than entire virtual machines. As a virtualization solution, Vagrant would only make sense if there were features in the guest system that were not available in the host, or if there were a legitimate reason for having multiple operating systems on the same machine.