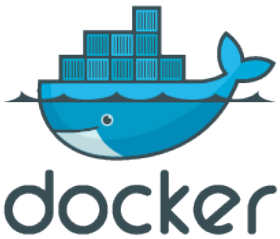
What is Dockerengine?



Overview

Docker Engine is the heart of Docker, a technology that allows you to create and run small, lightweight packages called **containers**. These containers are like tiny virtual machines but much more efficient. They contain everything an application needs to run, including the code, system libraries, and settings, so it behaves the same on any computer.

Docker 101

https://www.youtube.com/embed/rIrNIzy6U_g

Docker Engine consists of the following major components:

- **Docker Daemon**: This is the background service that handles container management on a Docker host. It listens for Docker API requests and manages the lifecycle of Docker containers, including starting, stopping, and monitoring.
- **REST API**: This API allows external tools and programs to communicate with the Docker Daemon, making it possible to manage Docker resources programmatically.
- **Docker CLI**: The Command-Line Interface (CLI) provides users with the ability to interact with the Docker Daemon using commands. It allows for creating and managing containers, images, networks, and volumes from the terminal.

Docker Engine can run on any Linux-based operating system, including distributions like **Debian**, **Ubuntu**, and **CentOS**, as well as other systems like **Windows** and **macOS** using platform-specific adaptations. On Linux systems, Docker containers share the host's kernel, making them lightweight and highly efficient.

How Does Docker Engine Work?

At its core, Docker Engine uses a **client-server** model. You, the user, interact with Docker by typing commands (using the CLI) or through other software (via the REST API). The **Docker Daemon** (the server part) listens to these requests and manages all the containers on your system.

- **Images and Containers**: Containers are created from something called **images**. Think of an image as a template or blueprint for a container. When you run an image, it becomes a container that can actually perform tasks.
- **Layers and File System**: Docker Engine makes things more efficient by building containers in layers, where each layer represents a change or addition to the image. This way, Docker doesn't need to rebuild everything from scratch each time you make changes.
- **Isolation and Resources**: Docker Engine uses special features in the Linux kernel (the core of the operating system) to isolate containers from each other, ensuring that one container's actions don't affect another. It also controls how much CPU, memory, and other resources each container can use.

Key Features of Docker Engine

- **Lightweight**: Containers don't need their own operating system; they use the host system's resources. This makes them much smaller and faster than virtual machines.
- **Portability**: Once you create a container, it will run the same way on any system that has Docker, no matter where it is. This makes it easy to move your application from your computer to a cloud server or any other environment.
- **Fast**: Containers start up almost instantly because they don't have to load a whole operating system. This makes them ideal for quick testing and development.
- **Isolation**: Each container has its own environment, meaning that your application and its dependencies won't interfere with other applications on the same system.

More Capabilities of Docker Engine

- **Networking**: Docker Engine allows containers to communicate with each other and the outside world through networks. You can connect containers together or expose them to the internet easily.
- **Storage**: Docker Engine can manage data that needs to persist even when containers are restarted or deleted. It does this using **volumes** (for storing data outside of containers) or **bind mounts** (which link folders on the host system to containers).

- **Orchestration Support**: For larger applications, Docker Engine works well with tools like **Docker Swarm** and **Kubernetes**. These tools help manage and automate the running of many containers at once, often across multiple servers.

Interested on More?

Check Out "Docker Vs VM's"

Revision #6
Created 11 September 2024 09:24:23 by aeoneros
Updated 8 October 2024 21:59:30 by aeoneros