3)Virtual Machine

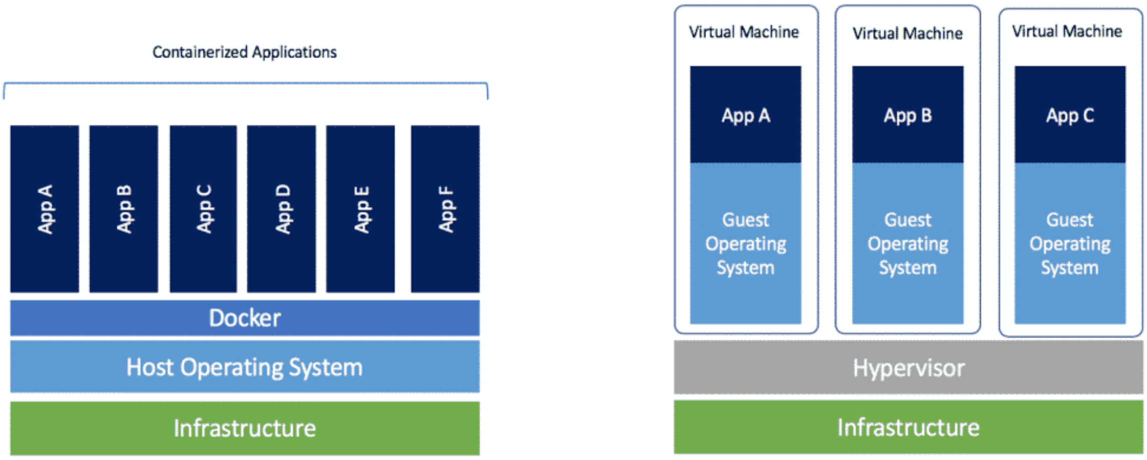
What is a virtual machine? Before delving into the explanations of virtual machines or Docker, we need to understand their common concept, which is virtualization. Virtualization is the process of converting the physical resources of a computer, such as storage, RAM, and processors, into virtual entities. This transformation leads us to the concept of virtual machines, also known as VMs. Virtual machines are essentially combinations of virtual components of a computer, meaning they are virtual computers running on a host's physical hardware. Multiple virtual machines can run on a single physical hardware layer, requiring the management and allocation of physical resources for each individual virtual machine. This need introduces us to a key concept for understanding VMs: hypervisors. Hypervisors act as intermediaries between the hardware and VMs running on the same host, ensuring successful distribution and allocation of resources for each VM.

Advantages of virtual machines:

1. **Operating System Flexibility:** Virtual machines can be used for applications that run only on specific operating systems since each VM operates with its own dedicated operating system.
2. **Isolated Testing Environment:** The isolated environment of each VM allows for testing of potentially harmful or malicious software without risking harm to the host system.
3. **Cost Savings:** Virtual machines can be a cost-saving option as multiple VMs can run on a single infrastructure, reducing physical footprint and associated costs such as electricity and storage.
4. **High Availability and Redundancy:** In case of hardware failures or issues with the host system, virtual machines can be quickly and easily transferred to a different hypervisor on another hardware, minimizing downtime and ensuring continuity of operations.

Docker

What is Docker? Docker is a software platform that utilizes virtualization but operates at the operating system level rather than the hardware level like virtual machines (VMs). Docker facilitates virtualization within the operating system, as depicted in the diagram.



What sets Docker apart? Docker is a containerization platform designed for developing, shipping, or running applications. Consider this scenario: you've developed an application on your computer and want to share it with a friend for feedback. However, your friend may have a different operating system or lack the necessary libraries and environment to run your app. Here's where Docker comes in handy. You can package your application code along with its required libraries and dependencies into a Docker container and then share it with your friend. This way, your friend doesn't need the exact same environment as yours to run the app. Docker containers encapsulate the app and all its dependencies, providing flexibility and portability for running, testing, or developing apps across different environments.

When discussing Docker, it's important to highlight some key concepts. The Docker engine is crucial—it manages all containers, Docker images, and other Docker objects, essentially acting as the brain of Docker. The Dockerfile is another essential concept; it serves as a blueprint for configuring the environment in which your application runs, helping to build Docker images. A Docker image is a template of your application, including necessary libraries, operating systems, configurations, or dependencies. Once you have your Docker image, you can upload it to a platform like DockerHub to share your app. However, simply having an image is not enough to run your app. You need to deploy it into a stable, portable, and isolated environment known as containers, and then run these containers to execute your app.

Advantages of Docker:

1. **Shared Kernel Performance**: Unlike virtual machines that copy the entire operating system, containers share the host's kernel. This shared kernel approach leads to a significant performance boost.

2. **Dependency Management**: Containers help mitigate the effects of "dependency hell," where applications struggle with conflicting dependencies. Each container provides an isolated environment, ensuring that dependencies are managed independently.

3. **Portability and Lightweight**: Containers are highly portable and lightweight compared to virtual machines. They encapsulate only the necessary components for an application, making them easier to deploy and manage across different environments.

4. **Automation and Standardization**: Docker containers are ideal for automated and standardized tasks. They allow developers to package applications and their dependencies consistently, ensuring that applications run reliably across various platforms and environments.