Virtual machines and containers

Software deployment pains:

* A program depends on it’s environment, which includes
  + A specific operating system
  + Ho
* It is easy to forget which parts of the environment the program depends on

Security concerns:

- Operating systems are not designed to isolate resources for multiple users

- VMs and containers provide isolation:

- Computing resources

- Network (isolate traffic to different customers)

How containers work:

* A container engine runs as a normal application on top of an underlying OS
* Each container is a normal operating system **process**, thus enjoying memory isolation. (This process can spawn children.)

How VMs work:

* A virtual machine manager runs on top of the underlying system. The virtual machines running on top of each think they run on their own hardware
* Thus, each VM needs it’s own OS. Different VMs may have different OS

Characteristics of VMs:

* **Fidelity:**Software on the VMM executes identically to its execution on hardware, barring timing effects.
* **Performance:**An overwhelming majority of guest instructions are executed by the hardware without the intervention of the VMM.
* **Safety:**The VMM manages all hardware resource

A comparison of different types of vmm

Description automatically generated

Comparison:

* Containers:
  + A container image takes 10s of MBs
  + Fast
  + The underlying OS can be anything
  + The simulated OS is usually Linux, although this is not a theoretical limitation
  + Typical example: Docker
* VMs:
  + A VM image takes 10s of GBs
  + Slow (especially to boot)
  + The underlying OS can be anything
  + Since the VM simulates hardware, one can install any actual OS on top of it

Docker concepts:

* Container: a running group of processes, isolated from other process on the computer; in particular. They have their own file system
* Image: a file system from which a container can be started
* Dockerfile: a list of instructions on how to create an image, by applying changes to an existing image
* Engine: a service for managing images and containers
* Orchestrator: a tool for deploying containers