Evolution of The Container Technology

Preparation

Christopher Christensen

Summary

Docker Inc. calls Docker a platform to develop, deploy and run applications with containers. It use Linux containers which is called containerization. Applications are deployed on to these containers. Some advantages of containers are listed as follows (Docker Inc. 2019):

- Flexible
- Lightweight
- Interchangeable
- Portable
- Scalable
- Stackable

Containers

Docker runs an image that launches a container. These images are executable packages that have all configurations, code, runtime, libraries and environment variables that you need to provide in order to run your application. Each container then represents a runtime instance of that image (Docker Inc. 2019).

Containers vs. VMs

The differences are listed in this table (Wong 2016).

- Containers
  - Relies on underlying OS to provide basic services
  - Virtual-Memory support for isolation
  - Lower overhead
  - Container systems typically target environments where thousands of containers are in play
  - Service isolation between containers, therefore can have limited resource access
  - Abstract OS

- Virtual Machines
  - VM runs software on the bare metal hardware, whilst isolating it from the real hardware
  - VM are run on hypervisors that use their own OS
  - Use hardware VM support
  - Larger overhead
  - Abstract machines that use device drivers

Container History

- Same concepts have already existed in the past
- When technologies advance these (older) concepts resurface
- 1979 `chroot`: change the root directory of a running process
- 2000 FreeBSD `jail` command
- 2001 Linux VServer ($\sim$ `jail` mechanism)
- 2004 Solaris Zones (comes closest to containers today)
- 2006 Open VZ
- 2006 process containers (Google): Isolate and limit resources (control groups)
- 2008 cgroups merged into Linux kernel
  - LXC project (the beginning of Docker)
- 2011 Warden (CloudFoundry)
- 2013 Google open-sourced LMCTFY

Further Notes from Talk

- Many creative people, but too many technologies
- That is why Docker wants to create environments where "anything" can run no matter the technology
- Docker doesn't lay as a layer under the Bins/Libs but stands "beside" them and only orchestrates the containers
- used to be 3-tier applications (front- / backend, database)
- nowadays microservices
- To run desktop applications over a container you need to use the namespaces

OCI - Open Container Initiative

- Established in 2015
- There are now 2 Specifications
  - `runtime`
  - `image`

Images

- Standardized unit
- Read-only filesystem containing relevant binaries / libraries / ...
- configuration information

Questions

1. Why has Docker become so popular?
   - Lightweight (no GuestOS)
   - DockerInc popularized it
   - Easy Setup
2. What are the limits of Docker?
   -
3. What are the downsides of Docker?
   - Changes inside a container are not persisted (but this is where images are great)
   - with microservices: how do services find eathother, etc.
   - orchestrating can be complex, and services such as Kubernetes for instance Kubernetes have a steep learning curve
4. Where is Docker not a smart idea?
   - can get very complex when using with microservices (besides the benefits)
   - databases (maybe in the beginning and without experience)
   - otherwise not much (at current state)
5. Are there any vulnerabilities to consider when using Docker?
   - not same stage as vms, but they are secure.
   - depends on the installations or dependencies that you have in your containers, whereas with VMs you have a lot of things you don't need that might have vulnerabilities you are unaware of.

Sources

- Docker Inc., *Get Started, Part 1: Orientation and setup*, 2019, docs.docker.com
- William G. Wong, *What's the Difference Between Containers and Virtual Machines?*, 2016, electronicdesign.com