Containers & Kubernetes

A crash course
Bob Killen
rkillen@umich.edu
Senior Research Cloud Administrator
CNCF Ambassador

Github: @mrbobbytables
Twitter: @mrbobbytables
Jeffrey Sica
jsica@umich.edu
Senior Research Database Administrator
Github: @jeefy
Twitter: @jeefy
What is an application, really

- Code
- Execution environment
  - Language-level libraries (pip or npm)
  - OS-level libraries (libjpg or libmysql)
- Potentially external dependencies/services
  - Database (Postgres, MySQL etc)
  - Caching layer (redis or memcached)
What is a Container?

A means to easily isolate, package, deliver, and deploy code.

A “Shipping Container” for your code.

Image Source: HANJIN container by John A.
Containers contain like buckets, not like moats. They're designed to put things together, not isolate them.
"I like putting apps into containers because then I can pretend they're not my problem."

@sadoperator
What a Container is Not

● A VM*
  ○ Containers are not a VM. Docker is not a hypervisor.

● Persistent
  ○ Containers are ephemeral in nature.

● Secure by Default
  ○ A container is not a security panacea.
  ○ They do have the capability of being more secure, but effort is required.

* Some “Sandbox” containerizers spin up a container within a VM for each container.
Workloads are becoming **multi-tier** and more complex.

Application(s) and their associated infrastructure are more **dynamic**. Application infrastructure now must "react" to changes and events.

Developers can work locally and push globally.

Facilitates a “microservice” architecture.
Why Are Containers a Thing?

- Reproducible
- Portable
- Easy to “plug-in” and link components
What is a Container...Really?
What is a Container...Really?

A process
What is a Container...Really?

A process
(with some additional properties)
What is a Container...Really?

- **Isolated**
  - Kernel handles application process namespace separation.

- **Fast**
  - Containers boot in milliseconds.

- **Immutable***
  - Container images bundle an application, its dependencies, and other run-time requirements into an immutable, redistributable image.
  - Small compared to full fledged VM.
What is a Container...Really?

- **Cgroups (Control groups)**
  - Feature of the Linux Kernel.
  - Manages groups of processes and their resources.

- **Namespaces**
  - Restricts a process to certain aspects of the host.

- **Filesystem (image)**
  - Hierarchical layered filesystem.
  - Tar file(s) used for distribution.

<table>
<thead>
<tr>
<th>Linux Namespaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process ID Number (PID)</td>
</tr>
<tr>
<td>Network (NET)</td>
</tr>
<tr>
<td>Interprocess Communication (IPC)</td>
</tr>
<tr>
<td>Unix Time-sharing System (UTS)</td>
</tr>
<tr>
<td>User</td>
</tr>
<tr>
<td>Mount (MNT)</td>
</tr>
</tbody>
</table>
Container vs VM
## A Brief History

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1982</td>
<td>• Unix/BSD chroot</td>
</tr>
<tr>
<td>2000</td>
<td>• BSD Jails</td>
</tr>
<tr>
<td>2005</td>
<td>• Solaris Zones</td>
</tr>
<tr>
<td>2008</td>
<td>• Linux Containers (LXC)</td>
</tr>
<tr>
<td>2013</td>
<td>• LMCTFY (Google)</td>
</tr>
<tr>
<td></td>
<td>• Docker (formerly dotCloud)</td>
</tr>
<tr>
<td>2014</td>
<td>• rkt (CoreOS)</td>
</tr>
<tr>
<td>2015</td>
<td>• Open Container Initiative (OCI) Established</td>
</tr>
</tbody>
</table>
## Container “Components”

<table>
<thead>
<tr>
<th>Container Runtime</th>
<th>Image</th>
<th>Container</th>
</tr>
</thead>
<tbody>
<tr>
<td>Governing process that downloads and executes a container image.</td>
<td>Immutable bundle containing an application and its runtime dependencies.</td>
<td>Running instance of the image.</td>
</tr>
</tbody>
</table>

Image Source: [docker.com](https://docker.com)
What is Docker?

- Docker is **BOTH** a Company (Docker Inc.) and a large Open Source project.
- Available on multiple platforms and architectures.
- **The** “Standard” when people think of container.

Image Source: docker.com
What is Docker?

$ docker build -t <image_name> .
$ docker run -p 80:80 -v $(pwd):/workdir \
   --name <container_name> <image_name>
$

$ docker logs -f <container_name>
$ docker exec -i -t <container_name> /bin/bash
$ docker rm -f <container_name>
$ docker rmi <image_name>
Container Orchestration
Containers In Production

- Runtime
- Application Definition
- Networking
- Storage
- Scheduling Rules
- Health Checks
Container Orchestration

Kubernetes
What Does “Kubernetes” Mean?

Greek for “pilot” or “Helmsman of a ship”
What can Kubernetes REALLY do?

- Autoscale Workloads
- **Blue/Green** Deployments
- Fire off jobs and scheduled cronjobs
- Manage Stateless and Stateful Applications
- Provide native methods of service discovery
- Easily integrate and support 3rd party apps
Most Importantly...

Use the **SAME API** across bare metal and **EVERY** cloud provider!!!
Containers In Production

- Runtime            CRI
- Networking         CNI
- Storage           Volumes / CSI
- Application Definition    Specs
- Scheduling Rules      Affinity / Anti-Affinity
- Health Checks       Liveness Probes
Core Concepts

Kubernetes has several core building blocks that make up the foundation of their higher level components.

Namespaces

Pods
Labels

Services
Selectors
Pods

- **Atomic unit** or smallest "unit of work" of Kubernetes.
- Pods are **one or MORE containers** that share volumes, a network namespace, and are a part of a **single context**.
Pods

They are also ephemeral!
Services

- Unified method of accessing the exposed workloads of Pods.
- **Durable resource**
  - static cluster IP
  - static namespaced DNS name
Services

- Unified method of accessing the exposed workloads of Pods.
- **Durable resource**
  - static cluster IP
  - static namespaced DNS name

NOT Ephemeral!
What is an application, really

- Code
- Execution environment
  - Language-level libraries (pip or npm)
  - OS-level libraries (libjpg or libmysql)
- Potentially external dependencies/services
  - Database (Postgres, MySQL etc)
  - Caching layer (redis or memcached)
Tying it all together

- A container is a way to **package code and dependencies into a reproducible distributable environment**.
- Docker is the foremost “container runtime”.
- Kubernetes is a cloud-agnostic project that takes container workloads and runs them in production.
- Minikube is a Kubernetes project that allows developers to spawn a Kubernetes cluster locally.
Reference material

- Introduction to Containers
  Tutorial / Intro to Containers
- Introduction to Kubernetes
  Tutorial / Intro to Kubernetes
- Katacoda
  Online Tutorials of many different container related subjects.
Questions

Also open discussion. :)