About BoxBoat

BoxBoat Technologies was founded to empower and implement DevOps to build, ship, and run Docker distributed applications. We are technologists at heart who believe in the power of containerization and the increased scale, flexibility, and resource utilization it brings to the development workflow. Deliver software faster with BoxBoat, a Docker Inc. and CNCF Kubernetes Authorized Consulting Partner.

Will Kinard
CTO
It’s all about containers

What Docker containers are
How containers can benefit your organization
People are really doing this!
Container orchestration with Kubernetes and Swarm
Docker Open Source Project

GitHub (github.com/docker, github.com/moby/moby)
- 2900+ Contributors
- 10,000+ Active Forks
- 34,000+ Stars

Docker Hub (hub.docker.com)
- 26B+ Image Downloads
- 1,000,000+ Dockerized Applications
- Exponential growth
What is a [Docker] Container?

- Method to run applications in isolation
- Isolation includes namespacing pid, network, users, restricting root, cpu and memory limits, and providing separate filesystem
- Many of the technologies are old, but haven’t been packaged in an easy to use toolset before Docker

“Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server.” (https://www.docker.com/what-docker)
Container Technology Isn’t New...

<table>
<thead>
<tr>
<th>Technology</th>
<th>Year</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chroot</td>
<td>1979</td>
<td>Early container technology</td>
</tr>
<tr>
<td>Linux Namespaces</td>
<td>2002</td>
<td>Process isolation</td>
</tr>
<tr>
<td>Google Process Containers</td>
<td>2006</td>
<td>Process aggregation for resource management</td>
</tr>
<tr>
<td>Linux Control Groups</td>
<td>2007</td>
<td>Process containers renamed and merged into kernel 2.6.24</td>
</tr>
<tr>
<td>LXC Linux Containers</td>
<td>2008</td>
<td>Userland tooling</td>
</tr>
<tr>
<td>DotCloud → Docker Inc.</td>
<td>2013</td>
<td>Introduction of Docker Open Source Project</td>
</tr>
</tbody>
</table>

...Its Implementation Is.
Virtual Machines

- Application
- Binaries and libraries
- Entire guest operating system

Containers

- Include application and all dependencies
- Share kernel with other containers
- Run as an isolated process not tied to any specific infrastructure

They’re different, not mutually exclusive
We still need Virtual Machines (...for now)

Virtual Machine: Host Virtualization

Containers are used in partnership with current IaaS stacks.

Docker and other container platforms still need a host.

Container: Application Virtualization

Take advantage of the streamlined process for VM based IaaS and gain efficiencies in:

- Higher density workloads
- Scale
- Portability
- Security
Portability is Empowering

Let’s relate the virtual abstract to the physical world:

- Shipping industry provides several modes of transporting goods
- Goods differ in shape, size, weight, and requirements
- Transport mediums must accommodate for freight discrepancy
Portability is Empowering

Let’s relate the virtual abstract to the physical world:

- The Shipping container provides an ideal standardization
- If it fits into a container, it can be shipped
- Transport mediums simply accommodate the container
Portability is Empowering

Now back to the abstract:

• Linux containers are the standardized medium for “shipping” or deploying software

• Here the analogy breaks – Linux containers also provide a standardized medium for running software
Container Delivery is becoming the Standard

Software vendors can now finally control client environments!

- Host Operating System package requirements are now just Docker (or container runtime alternatives)
- Vendors ship pre-installed software (Docker Images)
- Container orchestrated environments provide upsell features in high availability and replication
**Application Development**

**Problem:** Code Migration/Deployment

- Environment replication is difficult
- Developer on-boarding is tedious
- Feedback cycle from QA is slow
- Artifact management creep

**Solution:** Single Artifact Deployment

- Container replicates exact environment
- Deploy and migrate in sub-seconds
- Automated and self-documenting
- Track promotion cryptographically

Begin with “Known Trusted State”

Version control

Developer

QA / QE

Sysadmin

End with Approved Image
Application Modernization

**Problem: Legacy Applications**
- Brittle and difficult to upgrade
- Lengthy deployment cycle times
- Aging platform components and dependencies

**Solution: Microservice Architecture**
- Technology Diversity
- Deployment flexibility (On-Premise, Cloud, Hybrid)
- Cluster orchestration & high availability

Slider from ~2000 to Today:
- Monolithic
  - Slow changing
  - Big Servers
- Loosely Coupled Services
  - Rapidly updated
  - Many Small Servers
Production Operations

**Problem:** Inefficiency of VMs

- OS duplication for isolated work loads
- Lengthy boot and replication times
- Hardware, Storage, and Hypervisor costs $$
- Customer self-servicing leads to VM creep

**Solution:** Docker CaaS

- Consolidated storage in UFS and COW
- Deployment flexibility (On-Premise, Cloud, Hybrid)
- High Availability and Scaling
- Policy driven architecture using PKI
What about Orchestration?

Marathon, Swarm, Kubernetes, Oh My!

“I want to run multiple containers, across multiple hosts”

This has been a disparate ecosystem, but quickly converging.
Why Kubernetes?

The Market has Spoken

Flexibility, Stability, and Contribution

Swarm workloads still have a place in this world
A survey of 750 hiring managers by the Linux Foundation and Dice reported that 57 percent are seeking employees with container skills, up from 27 percent in last year’s survey.¹

¹https://thenewstack.io/steady-docker-adoption-leads-to-jump-in-hiring/
Thank You

Will Kinard
CTO BoxBoat Technologies

will@boxboat.com
@boxboat

www.boxboat.com/blog