Intro To Containers, Container Storage and Docker

Alex McDonald, NetApp (Moderator)
Keith Hudgins, Docker
Chad Thibodeau, Veritas Technologies

October 6, 2016
SNIA Presenters

Chad Thibodeau
Veritas Technologies

Keith Hudgins
Docker

Alex McDonald
Chair SNIA Cloud Storage Initiative
The material contained in this presentation is copyrighted by the SNIA unless otherwise noted.

Member companies and individual members may use this material in presentations and literature under the following conditions:

- Any slide or slides used must be reproduced in their entirety without modification
- The SNIA must be acknowledged as the source of any material used in the body of any document containing material from these presentations.

This presentation is a project of the SNIA.

Neither the author nor the presenter is an attorney and nothing in this presentation is intended to be, or should be construed as legal advice or an opinion of counsel. If you need legal advice or a legal opinion please contact your attorney.

The information presented herein represents the author's personal opinion and current understanding of the relevant issues involved. The author, the presenter, and the SNIA do not assume any responsibility or liability for damages arising out of any reliance on or use of this information.

NO WARRANTIES, EXPRESS OR IMPLIED. USE AT YOUR OWN RISK.
Agenda

1. Intro to Containers and how they enable DevOps & CI-CD
2. What is Docker?
3. Persistent Storage for Containers
4. Docker Data Center
5. Docker Case Studies
# Application Deployment History

<table>
<thead>
<tr>
<th>Monolithic Apps on Physical</th>
<th>Virtual Machine Abstraction</th>
<th>Stateless &amp; Horizontal Scalable Apps</th>
<th>Micro-services &amp; Containers</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Monolithic Apps on Physical" /></td>
<td><img src="image2" alt="Virtual Machine Abstraction" /></td>
<td><img src="image3" alt="Stateless &amp; Horizontal Scalable Apps" /></td>
<td><img src="image4" alt="Micro-services &amp; Containers" /></td>
</tr>
</tbody>
</table>

- **Monolithic Apps on Physical**
  - Web
  - App
  - DB

- **Virtual Machine Abstraction**
  - Web
  - App
  - DB

- **Stateless & Horizontal Scalable Apps**
  - Web
  - App
  - DB

- **Micro-services & Containers**
  - Application
  - Services

**Application Deployment History**

- **Build**
- **Ship**
- **Run**
Intro to Containers and how they enable DevOps & CI-CD
Poll Question #1

Are You Currently Using Containers in?

A. Test and Dev only
B. Test and Dev and Production
C. Neither
Containers 101

- Virtualization of application instead of hardware
- Runs on top of the core OS (Linux or Windows)
- Doesn’t require dedicated CPU, Memory, Network —managed by core OS
- Optimizes Infrastructure—speed and density

"Containerization seems poised to offer both a complement and a viable alternative to server virtualization"  

(1) IDC
## Containers vs VMs vs Bare-metal Servers

<table>
<thead>
<tr>
<th></th>
<th>Container</th>
<th>Virtual Machine</th>
<th>Bare-Metal x86 Server</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Underlying Platform</strong></td>
<td>OS on Virtual Machine or Bare-Metal x86 Server</td>
<td>Hypervisor on Bare-Metal x86 Server</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Performance: Speed and Consistency</strong></td>
<td>Average</td>
<td>Average</td>
<td>Fastest</td>
</tr>
<tr>
<td><strong>Provisioning Time</strong></td>
<td>Seconds</td>
<td>Minutes</td>
<td>Hours</td>
</tr>
<tr>
<td><strong>Tenant Isolation Enforcement</strong></td>
<td>OS Kernel</td>
<td>Hypervisor</td>
<td>Physical</td>
</tr>
<tr>
<td><strong>Ideal Application Types</strong></td>
<td>Mode 2</td>
<td>Mode 1 or Mode 2</td>
<td>Mode 1 or Mode 2</td>
</tr>
<tr>
<td><strong>Configuration and Reconfiguration Flexibility</strong></td>
<td>Highest</td>
<td>Medium</td>
<td>Lowest</td>
</tr>
<tr>
<td><strong>Host Consolidation Density</strong></td>
<td>Maximum</td>
<td>Average</td>
<td>None</td>
</tr>
<tr>
<td><strong>Application Portability</strong></td>
<td>Application Packaging/Manifest*</td>
<td>VM Image, VM Migration Tools</td>
<td>Backup and Restore, ISO Images</td>
</tr>
<tr>
<td><strong>Granularity</strong></td>
<td>Extremely Small</td>
<td>Average</td>
<td>Largest</td>
</tr>
</tbody>
</table>

*While application portability is somewhat easier in container environments that are leveraging a container management and orchestration solution, portability should not be assumed to be universal — differences in the underlying host OS below the containers could still present some interoperability challenges.*

Source: Gartner (September 2015)
Driving Factors for Containers

- Density & Performance
- Licensing Costs
- Shift to DevOps
- Cloud-native Applications (Scale-out)
- Faster Exploration & Deployment (CI/CD)

“Containerization seems poised to offer both a complement and a viable alternative to server virtualization” - IDC
In 2018 more than 50% of new Workloads will be deployed into containers in at least one stage of the life application cycle (1)

(1) Gartner
Poll Question #2

Does Your Organization Practice DevOps or Have a DevOps Department?

A. Practicing DevOps
B. Planning on Practicing DevOps
C. No DevOps Plans
Driving force behind modern app initiatives

- **Microservices**: 3 out of 4 initiatives revolve around applications.
- **Docker**: 80% of cloud strategies are centered around Docker.
- **DevOps**: 44% of organizations are looking to adopt DevOps.
Scenario: Continuous Integration and Delivery

1. Development
   - Version control

2. Test
   - QA / QE

3. Stage / Production
   - Sysadmin
   - Developer
Scenario: Enabling Transformation to Microservices

Before

<table>
<thead>
<tr>
<th>App A</th>
<th>App B</th>
</tr>
</thead>
<tbody>
<tr>
<td>App Service</td>
<td>App Service</td>
</tr>
<tr>
<td>Auth</td>
<td>Auth</td>
</tr>
<tr>
<td>App Reg</td>
<td>App Reg</td>
</tr>
<tr>
<td>Marketplace</td>
<td>Marketplace</td>
</tr>
<tr>
<td>Logging</td>
<td>Logging</td>
</tr>
<tr>
<td>…more</td>
<td>…more</td>
</tr>
</tbody>
</table>

Common services in monoliths are turned into base applications stored in the Trusted Registry available to all app teams.

After

<table>
<thead>
<tr>
<th>Application Teams</th>
<th>Trusted Registry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization</td>
<td>App Registration</td>
</tr>
<tr>
<td>Session Management</td>
<td>Marketplace Integration</td>
</tr>
<tr>
<td>Logging</td>
<td>…more</td>
</tr>
</tbody>
</table>

Teams request into central IT maintained portal/registry to provision infrastructure and pull base images.

Universal Control Plane

Portability

Monoliths are now microservices applications. Each app has its own containers based on the same base image.
What is Docker?
Poll Question #3

Are You Familiar with Docker?

A. Yes
B. No
What is Docker?

Docker is a platform for developing, shipping and running applications using container technology.

The Docker Platform consists of multiple products/tools

- Docker Engine
- Docker Hub
- Docker Trusted Registry
- Docker Machine
- Docker Compose
- Docker for Windows/Mac
- Docker Datacenter
Docker Adoption

- Docker Hub - “Pull” is one download of a container image
- 650,000 registered users
- 5 Billion pulls since 2013
- Growing by 150% per month
Persistent Storage for Containers
Stateful vs Stateless

“Stateful container apps represent the next big IT challenge\(^{(1)}\)"

“Persistent storage among top issues for container enterprise-readiness in production\(^{(2)}\)"

“Stateful Database applications such as Redis, MySQL, MongoDB among most pulled images on Docker Hub\(^{(2)}\)"

---

### Importance of Container Orchestration Abilities

<table>
<thead>
<tr>
<th></th>
<th>Extremely Important</th>
<th>Very Important</th>
<th>Moderately Important</th>
<th>Slightly Important</th>
<th>Not At All Important</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long-running applications</td>
<td>68%</td>
<td>18%</td>
<td>5%</td>
<td>5%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>Load balancing</td>
<td>59%</td>
<td>10%</td>
<td>11%</td>
<td>8%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>Persistent storage</td>
<td>38%</td>
<td>32%</td>
<td>14%</td>
<td>16%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Batch applications</td>
<td>27%</td>
<td>16%</td>
<td>32%</td>
<td>22%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>Non-containered workloads</td>
<td>11%</td>
<td>8%</td>
<td>25%</td>
<td>28%</td>
<td>28%</td>
<td></td>
</tr>
</tbody>
</table>

\(^{(1)}\) The New Stack Survey, March 2016. When evaluating container orchestration tools, how important following requirements? n=107. Due to rounding, figures may not equal 100%.
Persistent Storage—Why

Data Accessibility
Run containers anywhere without worries about where data is located

Deployment Cycles
Traditional storage approaches slow-down innovation

Data Availability
Data needs to be always on no matter what happens

Storage Costs
Storage defined by Software using any commodity HW or Cloud
Storage Services for Containers

- Persistent Storage Management
- Scale-Out
- Snapshots
- I/O Acceleration

- Quality of Service
- Encryption
- Disaster Recovery
- Heterogeneous Support
Poll Question #4

Do you have storage challenges in running containerized applications in production?

A. Persistent Storage
B. High Availability
C. Quality-of-Service
D. All of these
Docker Storage Types

- **Registry**: Cold storage of container images
- **Graph**: Active storage of running container images
- **Volume**: Persistent block storage for data
Docker Registry Storage

- Config data stored via standard Docker volumes
- Images stored via driver
- Native filesystem (We don’t care what’s beneath - NFS or iSCSI is best!)
- Drivers available for cloud object storage for images (S3, Swift, GCS)
Docker Volume Storage

- This is where persistent data lives
- Extremely pluggable
- Network attached storage is extremely useful here
Docker Data Center (DDC)
Docker Datacenter CaaS workflow

Developers

BUILD
Development Environments

IT Operations

SHIP
Secure Content & Collaboration

RUN
Deploy, Manage, Scale
Developers

- Freedom to create and deploy apps fast
- Define app needs

IT Operations

- Quickly and flexibly respond to changing needs
- Standardize, secure, and manage

Frictionless portability across teams, environments, infrastructure
Docker Datacenter delivers agility

- **Easy to setup and use**
  - Fastest time to value for ops
  - Developer self service from library of images

- **Native Docker solution**
  - Full support for Docker API
  - Integrated Docker Engine, Swarm, Compose

- **Extend the existing Docker developer experience to deployment**
  - Deploy Compose apps directly in UCP
Docker Datacenter delivers portability

- **Compute, network and storage**
  - Network and volume plugins ensure apps work without recoding to new environments
  - Move across public, private and hybrid cloudility

- **Seamless dev to prod workflow**
  - Same code in dev runs in prod w/o changes
  - Eliminate the “works on my machine” issues
  - Full support for Docker API
Docker Datacenter delivers control

- **Management ease at scale**
  - View and manage apps, containers, nodes, volumes, networks, images, users and groups
  - Built in HA, backup/restore
  - Point and click GUI or CLI support

- **Integrated security and enterprise controls**
  - Content Trust for image signing and verification
  - Secure access with RBAC and LDAP/ADP
  - Out of the box TLS

- **Extend and integrate DDC to your systems and processes**
Docker Case Studies:
Brief overviews of recent success stories for Docker customers
Case Study: ADP

Before

App A
- App Service
  - Auth
  - App Reg
  - Marketplace
  - Logging
  - …more

App B
- App Service
  - Auth
  - Session
  - App Reg
  - Logging
  - …more

Common services in monoliths are turned turned into base applications stored in the Trusted Registry available to all app teams.

After

Application Teams

- Teams request into central IT maintained portal/registry to provision infrastructure and pull base images

Universal Control Plane

- Monoliths are now micro services applications. Each app has it’s own containers based on the same base image.
Case Study: SA Home Loans

SA Home Loans uses Docker Datacenter to convert Monoliths to Microservices

Goal

• Convert monolithic .Net applications (built in Mono) into microservices

Result

• Evaluated Docker running small-scale postgres services across 2 nodes
• Docker’s enterprise-class networking and security capabilities were key but impressed with the ease-of-use of Docker Native orchestration - Swarm
• Currently running Docker Datacenter across 4 nodes as they are working to Dockerize all enterprise-class applications in the next few months
Challenge
• Migrate away from monolithic application
• Long and cumbersome application development cycles

Solution
• Build a new developer platform (IAE Common Service Platform) with Docker Trusted Registry and commercially supported Docker Engine on AWS

Benefit
• Improved customer centric services Reduced time-to-market
• Improve security and reduce attack surface area
Container Future Areas of Exploration
Build | Ship | Run

Build

Copy Data Management

Data Insight

Ship

Disaster Recovery

High Availability

Run

Scalability

Resiliency

Portability
But Wait, There’s More!

- Please join us for part-two of this Webcast!

  “Containers: Best Practices & Data Management Services”
  December 7, 2016, 10:00 am PT

- Register at:
  https://www.brighttalk.com/webcast/663/227349

- Stay updated! Join our Containers opt-in email List
  http://eepurl.com/ciMk0P
More SNIA Webcasts on Containers

- The SNIA Ethernet Storage Forum (ESF) is hosting a Containers Webcast next month

  “The State of Storage in the Container World”
  November 17, 2016, 10:00 am PT

- Register at: https://www.brighttalk.com/webcast/663/225901
After This Webcast

- Please rate this Webcast. We value your feedback
- This Webcast and a copy of the slides will be on the SNIA Cloud Storage website and available on-demand
  - [http://www.snia.org/forum/csi/knowledge/webcasts](http://www.snia.org/forum/csi/knowledge/webcasts)
- A Q&A from this webcast, including answers to questions we couldn't get to today, will be on the SNIACloud blog
- Follow us on Twitter @SNIACloud
Thank you.

Keith.Hudgins@docker.com
Chad.Thibodeau@veritas.com