Containers for Grownups: Migrating Traditional & Existing Applications

Almost all software was designed before containers

Scott McCarty
Container Strategist
Why Migrate
Container Journey

- Evaluate Technology
- Experiment
- Quick Win
- Inventory Applications
- Determine Technology
- Containerize
The Journey

Single Node

Traditional Development

FIND  RUN  BUILD
The Journey

Single Node+

Traditional Development

FIND  RUN  BUILD  SHARE
The Journey
Multi Node

Traditional Development

Cloud Native

FIND  RUN  BUILD  SHARE  INTEGRATE  DEPLOY
The Journey
Can start anywhere

Traditional Development

Cloud Native

FIND
RUN
BUILD
SHARE
INTEGRATE
DEPLOY

RHEL (Podman/Buildah/Skopeo)
Quay
OpenShift (Kubernetes)
Customer Needs
Mapping customer needs to solutions

<table>
<thead>
<tr>
<th>Capability</th>
<th>Technology</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Node</td>
<td>Linux &amp; Container Tools</td>
<td>Red Hat Enterprise Linux</td>
</tr>
<tr>
<td>Multi Node</td>
<td>Linux &amp; Kubernetes</td>
<td>OpenShift</td>
</tr>
</tbody>
</table>
There are Different Options
Each has a different level of effort

<table>
<thead>
<tr>
<th>LIFT &amp; SHIFT</th>
<th>REFACTOR</th>
<th>REWRITE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move the application as-is. Do the minimal amount necessary to containerize</td>
<td>Leave much of the application as-is. Build new strategic portions.</td>
<td>Start from scratch - adopt the latest software development patterns (microservices, etc)</td>
</tr>
</tbody>
</table>
The Problem

Almost all applications, even new ones, are built on technologies that were designed and developed before Linux Containers.

*Image: Cargo Designed for a Ship from 1921*
Gaining migration skills will help your team make good strategic decisions about what should and should not be containerized.
LISTEN TO YOUR APPLICATION

We have seen this before...

UNIX TO LINUX
Will my app run?

PHYSICAL TO VIRTUAL
Will my app run?

APPS TO CONTAINERIZED APPS
Will my app run?
Guidance & Case Studies
APPLICATION CHARACTERISTICS
These are the things you need to think about..

Architectural  Security  Performance

Migration from process isolation to containers
Architecture

1. Code: mysqld
2. Configuration: /etc/my.cnf
3. Data: /var/lib/mysql
4. Other stuff :-)
# Level of Effort

<table>
<thead>
<tr>
<th></th>
<th>EASY</th>
<th>MODERATE</th>
<th>DIFFICULT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Code</strong></td>
<td>Completely Isolated (single process)</td>
<td>Somewhat Isolated (multiple processes)</td>
<td>Self Modifying (e.g. Actor Model)</td>
</tr>
<tr>
<td><strong>Configuration</strong></td>
<td>One File</td>
<td>Several Files</td>
<td>Anywhere in Filesystem</td>
</tr>
<tr>
<td><strong>Data</strong></td>
<td>Saved in Single Place</td>
<td>Saved in Several Places</td>
<td>Anywhere in Filesystem</td>
</tr>
<tr>
<td><strong>Secrets</strong></td>
<td>Static Files</td>
<td>Network</td>
<td>Dynamic Generation of Certificates</td>
</tr>
<tr>
<td><strong>Network</strong></td>
<td>HTTP, HTTPS</td>
<td>TCP, UDP</td>
<td>IPSEC, Highly Isolated</td>
</tr>
<tr>
<td><strong>Installation</strong></td>
<td>Packages, Source</td>
<td>Installer and Understood Configuration</td>
<td>Installers (install.sh)</td>
</tr>
<tr>
<td><strong>Licensing</strong></td>
<td>Open Source</td>
<td>Proprietary</td>
<td>Restrictive &amp; Proprietary</td>
</tr>
<tr>
<td><strong>Recoverability</strong></td>
<td>Easy to Restart</td>
<td>Fails Sometimes</td>
<td>Fails Often</td>
</tr>
</tbody>
</table>

Scott McCarty  Twitter: @fatherlinux  Blog: bit.ly/fatherlinux
The Tenancy Scale

- Process
- Container
- Virtual Server
- Physical Server
- Rack
- Data Center
# Performance Characteristics

<table>
<thead>
<tr>
<th></th>
<th>Bare Metal</th>
<th>+Containers</th>
<th>+Virtualization</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Intensive</td>
<td>Fast</td>
<td>Fast</td>
<td>Fast</td>
</tr>
<tr>
<td>Memory Intensive</td>
<td>Fast</td>
<td>Fast</td>
<td>Fast</td>
</tr>
<tr>
<td>Disk I/O Latency</td>
<td>Fast</td>
<td>Fast</td>
<td>Medium</td>
</tr>
<tr>
<td>Disk I/O Throughput</td>
<td>Fast</td>
<td>Fast</td>
<td>Fast</td>
</tr>
<tr>
<td>Network Latency</td>
<td>Fast</td>
<td>Fast</td>
<td>Medium</td>
</tr>
<tr>
<td>Network Throughput</td>
<td>Fast</td>
<td>Fast</td>
<td>Fast</td>
</tr>
<tr>
<td>Deployment Speed</td>
<td>Slow</td>
<td>Fast</td>
<td>Medium</td>
</tr>
<tr>
<td>Uptime (Live Migration)</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Alternative OS</td>
<td>Yes</td>
<td>Some</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Use Cases & Demos
PROBLEM

- WordPress websites needed at large scale
- Individual server doesn’t scale with load
- Multi-server states hard to synchronize
- Development environment difficult to replicate

SOLUTION

- Containerized service
- Site code distributed in images
- Single-service containers are easy to swap
- Development & production built from images

THE RESULT

- Enterprise-level - ~8,800 sites and ~29,000 users
- Quickly scalable, cloudburst-able
- Interchangeable front-end containers = automated sites with custom DNS and HTTPS
- Developers have taken ownership of deployment, can update, roll-back, clone w/o sysadmins
### Inventory: Red Hat IDM in a Container

#### PROBLEM
- Dozens of daemons, their own data volumes
- Libraries share config files
- Installer logic expects single machine
- Knowledge embedded in init/systemd
- Extensive initial setup needed

#### SOLUTION
- Build IDM (aka FreeIPA) in single container
- Simplified install, upgrade, rollback
- Minimize data volumes
- Template based data volume population

#### THE RESULT
- Better software delivery
- Better technical demarcation between vendor and customer
- Extremely portable
- Environments can be setup, tested, torn down in minutes
- Same pattern for other products and services: OpenShift, OpenStack (yes, OSP), sssd, etc.
Inventory: Tools

**PROBLEM**
- Can’t install software on prod servers
- Can’t introduce risk on prod servers
- Can’t access tools quickly

**SOLUTION**
- Build tools containers
- Stage tools containers in registry server
- Completely self contained

**THE RESULT**
- Use tools and delete them easily (rhel-tools)
- Scan network
- Instrument kernel
- Use any tool you like
Remember, Containers Are Just Child Processes
Questions?
Further Reading & Citations

- Do Linux distributions still matter with containers? [https://goo.gl/t4LLWw](https://goo.gl/t4LLWw)
- A Practical Introduction to Docker Terminology: [http://red.ht/2beXHDD](http://red.ht/2beXHDD)
- Container Tidbits: When Should I Break My Application into Multiple Containers? [http://red.ht/22xKw9i](http://red.ht/22xKw9i)
- Architecting Containers Part 4: Workload Characteristics and Candidates for Containerization: [http://red.ht/1SBw9ql](http://red.ht/1SBw9ql)
- Architecting Containers Series: [http://red.ht/2aXijVF](http://red.ht/2aXijVF)
- Red Hat Connect for Technology Partners: [https://connect.redhat.com/](https://connect.redhat.com/)
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHatNews
FUTURE OPPORTUNITIES

- Do Linux distributions still matter with containers? [https://goo.gl/t4LLWw](https://goo.gl/t4LLWw)
- Let Red Hat help you analyze your application portfolio [http://red.ht/2ic4TX3](http://red.ht/2ic4TX3)
- Check out Architecting Containers Series [http://red.ht/1SBw9qI](http://red.ht/1SBw9qI)
- Learn more about Red Hat Summit at [http://redhat.com/summit](http://redhat.com/summit)