Application-centric packaging with Docker & Linux containers

Daniel Riek,
Sr Director Systems
Design & Engineering

Lars Herrmann,
Sr Director Product &
Business Strategy
Enterprise IT - Roles

- Developer
- Sysadmin
- Application Owner
- Infrastructure Owner
- Line of Business
- IT
Enterprise IT - Roles

- Developer
- Sysadmin
- Application Owner
- DevOps
- Infrastructure Owner
- Line of Business
- IT
Evolution: Traditional Enterprise OS

Traditional application deployment

- Single userspace runtime shared between applications.
- Environment and life cycle defined by host OS.
- Trend to isolate apps on hardware level.
- Managed by IT, very limited delegation.
- Stable, long maintenance, few updates, hardware-centric.
- Very limited flexibility.
- Resources generally underutilized.
Evolution: Virtualization & IaaS

Application deployment via virt & IaaS

- Application isolation per VM.
- Guest environment and lifecycle defined by application.
- Application and runtime abstracted from hardware.
- Higher flexibility at cost of increased redundancy and overhead.
- Complex multi-level management of host and VM layers
- Delegation along the Host / VM boundary.
Evolution: Application-Centric IT & PaaS

App delivery using Docker containers

- Application packaged with individual runtime stack using Docker and deployed into containers.
- Multi-instance, multi-version, maximal flexibility, minimal overhead.
- Delegation along the container boundaries.
- Shared services provided by host / container environment.
- Standardized hardened container host, clustering, orchestration.
CONTINUOUS DELIVERY WITH CLEAR RESPONSIBILITIES WITHIN DEVOPS

DEVELOPMENT FOCUS

APPS, CODE, DEPENDENCIES (LIBRARIES), DATA, AND PACKAGING

OPERATIONS FOCUS

MONITORING, NETWORK CONFIGURATION, REMOTE ACCESS, AND LOGGING
Application-Centric Packaging
Tech Details – Containers & Docker

• Linux Containers are a combination of kernel features: namespaces, control groups, SELinux.

• Containers provide lightweight isolation of process, network, filesystem spaces.

• Docker builds on Linux Containers, adds an API, an image format and a delivery and sharing model.

• Docker provides aggregate packaging to bind application and its runtime dependencies for deployment into a Linux Container.
Containers vs. Virt?

- Generally complementary concepts
- Virtualization: vertical abstraction
- Containerization: horizontal segmentation
- Containers used to replace virtualization where container paradigms more applicable:
  - Horizontal application isolation
  - Lightweight delegation
  - “Application Virtualization”
  - Density
- Containers on top of Virt/Cloud common.
Tech Details - Layering

• New images can be created by adding layers.
• Layering model allows for specialization.
• Base image and select number of platform layers provided by Red Hat.
• ISV images to enable RHEL ecosystem.
• Stack optimized for individual application with minimal packaging per layer.
Tech Details – Static Linking, Sharing

- Layers are overlays in a single inheritance tree.
- A layer is statically linked to its parent.
- Docker layers can be created interactively or built in a reproducible way out of a Dockerfile.
- Simple distribution of images through Registry / Index model.
Red Hat Enterprise Linux Atomic Host

**IT IS RED HAT ENTERPRISE LINUX**

- Inherits the complete hardware ecosystem, military-grade security, stability and reliability Red Hat Enterprise Linux is known for.

**OPTIMIZED FOR CONTAINERS**

- Minimal environment, tuned for running Linux Containers while maintaining compatibility with Red Hat Enterprise Linux.
- Easy to deploy, update, and rollback using image-based technology.

**MINIMAL FOOTPRINT**

**SIMPLIFIED MAINTENANCE**
RHEL Atomic Container Host

- Minimal optimized container host with atomic updates.
- Systemd for process management.
- Generic container orchestration primitives.
- Integrated with OpenShift Geard for cross-node PaaS orchestration.
- Shared services and management agents deployed as privileged containers.
Tech Details – Architectural Components

• Atomic and Docker integration across Red Hat product portfolio.
• On-premise Registry, advanced content management provided by Satellite.
• To be integrated with upstream Indexes
• Next generation OpenShift architected around Docker.
• OpenStack convergence.
Benefits of Linux Containers in the Docker Format

- Deployment Flexibility
- Operational Efficiency
- Simplified Maintenance
- Lowered Deployment Costs
But ...

- Application compatibility
- Trust, Provenance
- Tools
- Skills
- Ecosystem of solutions and support
Red Hat's Journey to Containers

START

MARCH 2002: Red Hat Enterprise Linux introduced

FEBRUARY 2005: Red Hat Enterprise Linux 4 launches with SELinux

2006: Google starts work on cgroups (control groups)

MARCH 2007: Red Hat Enterprise Linux 5 launches; 1st commercial Linux with integrated virtualization

NOVEMBER 2012: OpenShift Enterprise launches; 1st enterprise PaaS based on Linux Containers

MAY 2011: OpenShift Online launches

NOVEMBER 2010: Red Hat Enterprise Linux 6 launches with cgroups and Linux (Namespace) Containers

MARCH 2010: systemd project launches

MARCH 2013: Docker project launches

JULY 2013: Red Hat Enterprise Linux OpenStack Platform launches with kernel namespaces

SEPTEMBER 2013: Red Hat announces intent to bring Docker technology to OpenShift

NOVEMBER 2013: Red Hat Enterprise Linux 6.5 launches with support for Linux Containers

MARCH 2014: Red Hat launches industry’s first Container Certification program for ISVs

DECEMBER 2013: Red Hat Enterprise Linux 7 beta launches with new features for Linux Containers

NOVEMBER 2013: Docker 0.7 released with support for Fedora

APRIL 2014: RED HAT UNVEILS NEW CONTAINER INNOVATIONS

#redhat #rhsummit

Application-Centric Packaging
20
Containers in the Open Hybrid Cloud

RED HAT OPEN HYBRID CLOUD

OPENSHIFT - PaaS

CLOUDFORMS - Hybrid Cloud Management

JBOSST MIDDLEWARE
JBOSST MIDDLEWARE
JBOSST MIDDLEWARE
THIRD PARTY MIDDLEWARE
JBOSST MIDDLEWARE

RED HAT ENTERPRISE LINUX
RED HAT ENTERPRISE LINUX
RED HAT ENTERPRISE LINUX
THIRD PARTY OPERATING SYSTEM
RED HAT ENTERPRISE LINUX

RHEL Docker Portability

PHYSICAL SYSTEMS

RHEV
Datacenter Virtualization Management

RHEV HYPervisor
KVM

OPENSTACK
IaaS

RHEV HYPervisor
KVM

THIRD PARTY
VIRT/IaaS

CERTIFIED
PUBLIC CLOUD PROVIDERS

RED HAT STORAGE - Hybrid Cloud Storage
Container Certification

• Red Hat announced certification program for container images.
• Expands existing Red Hat certification into the container space.
• Ensures stable, end-to-end supportable stack for applications.
• Enables the whole Red Hat ISV ecosystem to benefit from the advantages of aggregate Application-centric packaging with Docker.
# Application-Centric Packaging Benefits

<table>
<thead>
<tr>
<th>PORTABLE AND RELIABLE APPLICATION DEPLOYMENTS</th>
<th>RAPID AND EFFICIENT APPLICATION DELIVERY</th>
<th>SIMPLIFIED APPLICATION DELIVERY LIFECYCLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Includes dependencies to work the same across multiple hosts</td>
<td>• Built, delivered, and patched in seconds</td>
<td>• Consistency across dev, test, and production environments</td>
</tr>
<tr>
<td>• Portable across the Red Hat portfolio in the Open Hybrid Cloud</td>
<td>• Run instantly, without restart</td>
<td>• Better patching via updating only what’s changed + rollback</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LIGHTWEIGHT FOOTPRINT AND MINIMAL OVERHEAD</th>
<th>ISOLATED AND SECURE DEPLOYMENTS</th>
<th>FINE-GRAINED CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Includes “just enough” host application, and dependencies</td>
<td>• Isolate and secure apps without hypervisor overhead</td>
<td>• Limit resource usage for each application instance</td>
</tr>
<tr>
<td>• Rapid delivery and scale out</td>
<td>• Modify apps without impacting the rest of the server</td>
<td>• Applications can be upgraded, rolled back, or removed in seconds</td>
</tr>
</tbody>
</table>
References

Booths and Demos

• Infrastructure Booth, Pod 1
• Emerging technologies Booth, Container Pod 31
• Developer area, OpenShift Origin and Project Atomic Booths
• Container lab Wed, Labs 1, 3:50 pm – 5:50 pm (Sold Out)

Website: http://www.projectatomic.io/
Application-centric packaging with Docker & Linux containers

Daniel Riek
Lars Herrmann
Path To Application Optimized Infrastructure

<table>
<thead>
<tr>
<th>APP ENABLING</th>
<th>APP AWARE</th>
<th>APP OPTIMIZED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifecycle</td>
<td>Elasticity</td>
<td>Minimal footprint</td>
</tr>
<tr>
<td>Security</td>
<td>Software-defined Infra</td>
<td>Atomic updating</td>
</tr>
<tr>
<td>Stability</td>
<td>Scheduling</td>
<td>Containers</td>
</tr>
<tr>
<td></td>
<td>Management</td>
<td>Orchestration</td>
</tr>
</tbody>
</table>

**Repositories and Tools**

- **RED HAT® ENTERPRISE LINUX®**
- **RED HAT® ENTERPRISE LINUX® OPENSTACK PLATFORM**
- **RED HAT® CLOUD INFRASTRUCTURE**
- **RED HAT® ENTERPRISE LINUX® ATOMIC HOST**
- **OPENSHIFT® by Red Hat®**
Layering Example

• Layering follows single-inheritance model.
• Each layer is statically linked to the underlying layers, down to the base.