Docker and Oracle Everything You Wanted To Know

June, 2017

Umesh Tanna
Principal Technology Sales Consultant
Oracle Sales Consulting Centers (SCC)
Bangalore
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Program Agenda

1. Docker Containers – What and Why
2. Oracle Linux and Docker Containers
3. Oracle Cloud and Docker Containers
4. Introducing Oracle Container Cloud Service (OCCS)
5. OCCS Functional Walk-through
6. Q&A
Virtual Machines vs. Containers

**Virtual Machines**
- Each virtual machine (VM) includes the app, the necessary binaries and libraries and an **entire guest operating system**.

**Containers**
- Containers include the app & all of its dependencies, but **share the kernel** with other containers.
- Run as an isolated process in userspace on the host OS
- Not tied to any specific infrastructure – containers run on any computer, infrastructure and cloud.
The history of Unix containers

While Docker has been playing a key role in adoption of the Linux container technology, they did not invent the concept of containers.

- UNIX V7 added chroot (1979)
- FreeBSD Jails (2000)
- Linux vserver (2001)
- Process Containers (2006)
- AIX (6.1) WPARS (2007)
- LCX (2008)
- Warden (2011)
- LMCTFY (2013)
- Docker (2013)
- rkt (2014)
- OCI (2015)
Docker Engine

- Container execution and admin
- Uses Linux Kernel namespaces and control groups
- Namespaces provide for isolated workspace
Docker Images

• An image is a collection of files and some meta data
• Images are comprised of multiple layers, multiple layers referencing/based on another image
• Each image contains software you want to run
• Every image contains a base layer
• Docker uses a copy on write file system
• Layers are read only
Why Containers?

A container is packaged as an entire runtime environment: the service/app plus all dependencies, libraries, & configuration files needed to run it
Portable across environments & lightweight (share the OS)

Developers care because:

• Quickly create ready-to-run packaged applications, low cost deployment and replay
• Automate testing, integration, packaging
• Reduce / eliminate platform compatibility issues (“It works in dev!”)
• Support next gen applications (microservices)

IT cares because:

• Improve speed and frequency of releases, reliability of deployments
• Makes app lifecycle efficient, consistent and repeatable – configure once, run many times
• Eliminate environment inconsistencies between development, test, production
• Improve production application resiliency and scale out / in on demand
Key Container Use Cases

• Developer productivity a top use case today

• Building out CI/CD pipelines
  – Consistent container image moves through pipeline
  – Preventing “it worked in dev” syndrome

• Application modernization and portability are also key adoption drivers (Prem <-> cloud)

The Need for Unified Container Management

Core Docker Architecture

- Docker client – Command Line Interface (CLI) for interfacing with the Docker
- Dockerfile – Text file of Docker instructions used to assemble a Docker Image
- Image – Hierarchies of files built from a Dockerfile, the file used as input to the docker build command
- Container – Running instance of an Image using the docker run command
- Registry – Image repository

Advanced Functions

- Orchestration, Monitoring, Operations, Service Discovery
- Docker Environment Provisioning

Fragmented Market Solutions

- Kubernetes
- Swarm, Docker Data Center, Docker Cloud
- Consul, ETCD, Docker Networking
- etc
Oracle Linux and Docker

- Docker Engine binary is built and maintained by Oracle
- Supports full container lifecycle: build, ship, distribute, and deployment of applications
- Oracle Linux Base images available
  - OL 5: 5.11
  - OL 6: 6.6, 6.7
  - OL 7: 7.0, 7.1, 7.2
- Support for common usage (i.e.: yum package manager)
- Support for btrfs for Docker filesystem
- Small image size

Support for Docker Running on Oracle Linux (Doc ID 1921163.1)
## Installing Docker on Oracle Linux

Last Updated: 09 May 2017

### Oracle Linux 7 Documentation

<table>
<thead>
<tr>
<th>Oracle Linux 7 Documentation</th>
<th>eLab</th>
<th>HTML</th>
<th>PDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Linux 7 Licensing Information User Manual</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Installation Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Administrator's Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Update 3 Release Notes</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Update 2 Release Notes</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Update 1 Release Notes</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Release Notes</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux 7 Security Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
</tbody>
</table>

### Oracle Linux Documentation

<table>
<thead>
<tr>
<th>Operating System Release 2.0 for Oracle Linux Release Notes</th>
<th>eLab</th>
<th>HTML</th>
<th>PDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Linux Docker User's Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux OTrace Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux OTrace Tutorial</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux OProfile User's Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Oracle Linux Distribution Linux Network User's Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Software Collection Library 2.3 for Oracle Linux Release Notes</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Spacewalk 2.6 for Oracle Linux Client Life Cycle Management Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Spacewalk 2.6 for Oracle Linux Concepts Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Spacewalk 2.6 for Oracle Linux Installation Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Spacewalk 2.6 for Oracle Linux Release Notes</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
<tr>
<td>Spacewalk 2.4 for Oracle Linux Client Life Cycle Management Guide</td>
<td>eLab</td>
<td>HTML</td>
<td>PDF</td>
</tr>
</tbody>
</table>
Installing Docker on Oracle Linux
2.2 Installing and Configuring the Docker Engine on Oracle Linux 7

**Note**
Docker version 1.9 and later require that you configure the system to use the Unbreakable Enterprise Kernel Release 4 (UEK R4) and boot the system with this kernel.

From version 1.8 of Docker onward, using the Docker configuration files in /etc/systemd/system/docker.service.d after adding or modifying a drop-in file while the Docker service is running, run the command `systemctl daemon-reload` to tell systemd to reload the configuration for the service.

To install and configure the Docker Engine on an Oracle Linux 7 system:

1. If you want to install Docker, configure the system to use the Unbreakable Enterprise Kernel Release 4 (UEK R4) and boot the system with this kernel:
   a. If your system is registered with ULN, disable access to the ul7afia64_uede4 channel and enable access to the ul7afia64_uede3 channel.

   ```bash
   [ul7afia64]
   baseurl=http://yum.oracle.com/repo/OracleLinux/UL7/UNR/64/ulafia
   gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ulafia
   enabled=0
   
   [ul7afia3]
   baseurl=http://yum.oracle.com/repo/OracleLinux/UL7/UNR/64/ulafia
   gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-ulafia
   enabled=0
   
   You can download the latest version of this file from http://yum.oracle.com/public-yum.7.repo.
   
   b. Run the following command to upgrade the system to UEK R4:
   ```bash
   # yum update
   
   c. Reboot the system, selecting the UEK R4 kernel if this is not the default boot kernel.

7 Steps installation is documented. However, getting started for learning purpose steps 4, 5 and 6 may not be required. However, please follow the appropriate section depending whether you are installing docker in Linux 6 or Linux 7.
Oracle Cloud and Docker Containers

Compute CS

DIY Container Management

DIY Container

Container CS

Oracle Managed Container Service

OCCS

Application Container CS

Docker-based Cloud Polyglot Platform

ACCS
Docker Page On http://developer.oracle.com/docker
Oracle on Docker

With the increasing popularity of Docker for not only development, but also for deployment, we are making available on the Docker Store our flagship products such as Oracle Database, Oracle WebLogic Server, Oracle Coherence, Oracle Java JRE and more as Docker images. These will join Oracle’s official Oracle Linux and MySQL images which are available on the Docker Hub. Developers may also engage with us by looking at the Dockerfiles published as open source software on GitHub.
Oracle on GitHub

Oracle Dockerfiles on GitHub

Developers can find the sources used to build all of the images that are available on Docker Hub and on the Docker Store right on the Docker-images project hosted on the Oracle GitHub organization. Our engineers are constantly improving these Dockerfiles to get the latest and greatest features of Docker, as well as adding samples and other Oracle products for faster use and adoption. If you have ideas, suggestions, or issues using the images or building from these sources, let us know.
Announcing: Official Oracle Docker Images

- Oracle Container Registry server to host pre-built Docker images for Oracle products
  container-registry.oracle.com

- Dockerfiles and samples on Oracle’s GitHub
  github.com/oracle/docker-images

  - Oracle Coherence
  - Oracle Database
  - Oracle HTTP Server
  - Oracle Java
  - Oracle Linux
  - Oracle NoSQL Database
  - Oracle Tuxedo
  - Oracle WebLogic
Introducing Oracle Container Cloud Service!

**What is It?**

- Hosted container service allows customers to bring and run their own Docker containers
- **Core and Advanced Docker Management** to orchestrate container placement and policies

**What Problems Does it Solve?**

- Building & maintaining Docker environments & management is complex, costly and time consuming
- Lack of example container applications and best practices to get started
- Enterprises need Docker management from dev/test through to production

**Key Benefits**

- Enables developers to get started and deploy containers quickly, DevOps teams with Docker management, visibility and control.
- Integrates with Continuous Integration & Deployment Pipelines to automate new releases.
- Supports polyglot development, open source software and microservice architectures
# Key Container CS Terminology Covered in Demo

<table>
<thead>
<tr>
<th>Container CS Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Service</strong></td>
<td>A service is a template for how to run a container from a single image and associated Docker runtime information (e.g. mapped ports, storage volumes etc...)</td>
</tr>
<tr>
<td><strong>Stack</strong></td>
<td>A Stack is a template for how to run a multi-service (container) based application (e.g. an “ELK” stack). It is very similar (and compatible with) Docker Compose format.</td>
</tr>
<tr>
<td><strong>Deployment</strong></td>
<td>One or more running containers that have been created from deploying a Service or a Stack</td>
</tr>
</tbody>
</table>
# Container CS Key Features

## Cloud & Container Native

- **Docker Compatible** (Docker Engine, Docker Compose)
  - Convert between Docker run, UI, YAML
- **One Click Deploy** (of example application stacks)
- **Registry Integration** (connect to public & private registries such as Docker Hub)
- **Build CI/CD Pipelines** (via “WebHooks”)
- **Service Discovery** – Built in to enable multi-container cross host deployments

## Enterprise Operations Ready

- **Operations Dashboards**
  - Key health/performance stats & container logs
- **Container Health Checks** (portping for container processes)
- **Service Scaling** (scale out services via UI or API)
- **Orchestration & Scheduling Policies** (least CPU/Memory, tag based placement, others)
- **Self-Healing Deployments** (maintain the desired state of a deployed multi-container app)

## Collaboration Built-In

- **Elegant Interface** - “Quickstart” Wizards to get users started
- **Best Practices Included** (example stacks covering monitoring, logging, load balancing)
  - Run & Deploy Oracle Certified Docker images
- **API Ready** – Drive common container operations and deployments via REST API
- **Resource Pools** (partition compute resources across dev, test, staging etc...)

---

Copyright © 2017, Oracle and/or its affiliates. All rights reserved. |
Container CS Provides Out of the Box Functionality

- Create Instances as Needed
- Define Resource Pools
- Add Private Registries
- Edit Create New Services
- Compose Application Stacks
- Deploy Stacks with 1 Click
- Automated Deployment
- Multi-Host, Easy Scale Out
- Built in Service Discovery
- Integrated Health Checks
- Unified Dashboard
- Monitoring and Auditing
Container Cloud Service Differentiation

1. Easy Onboarding and Use
2. Example Application Stacks
3. Part of Rich App Dev Portfolio
4. Planned Container Hybrid Cloud with Cloud @ Customer
1. Easy Onboarding and Use

- Intuitive complete UI for Developers and DevOps teams
- Quickstart Wizards for rapid setup
- Dashboards give at a glance operational views
- Docker Compatible (Docker Engine and tooling like Docker Compose)
2. Example Application Stacks

- Example “Stacks” enable quick customer ramp without complex orchestration

- Service Discovery Built-In to enable multi-host app deployments

- Complete functioning examples:
  - Load Balancing
  - Logging
  - Monitoring
  - Open Source (WordPress, Redis, others)
  - WebLogic Multi-Host
3. Part of Rich App Dev Portfolio
Unique in Blending Traditional, Cloud Native and Low Code with End to End PaaS

Oracle Platform for Cloud Application Development

**BACK-END SERVICES**
- Application Container
- Java
- Functions
- Mobile
- SDKs (3rd Party, JET)
- App Builder

**COMMON SERVICES**
- Developer Cloud (CICD)
- Container Cloud (Docker, Orchestration)
- APM/Log Analytics
- API Management
- Identity

**DATA SERVICES**
- Database
- MySQL
- NoSQL
- Integration
- SOA
- Internet of Things

**INTEGRATION SERVICES**

**FRONT-END TOOLING**
- SDKs (3rd Party, JET)
- App Builder

**INFRASTRUCTURE AS A SERVICE**
- Compute (VM, Bare Metal)
- Network
- Storage
4. Planned Container Hybrid Cloud with Cloud @ Customer

- Same IaaS and PaaS software
- Same updates as Oracle Cloud
- Same subscription and pay-as-you-go pricing
- Single vendor for the entire solution
Oracle Container Cloud Service Architecture

OCCS Manager Node

SSH Access / Console UI / REST API

Image Registry (Public/Private)

OCCS Worker Node

OCCS Worker Node

OCCS Worker Node

OCCS Instance
How Does **Container CS** fit into the DevOps Story?

- IaaS-centric target for deployment of **Docker-ized** apps
- Complements Oracle PaaS Applications with a **Bring your Own Container** Service
- Run ANY **Containerized** Application including **Open Source**
The Docker DevOps Cycle

Seen in the DevOps Sales Play – featuring:

- Oracle Developer Cloud Service
  - Now Includes Docker Image Build Function
- Oracle Container Cloud Service
  - Deploy, Operate, Monitor
The New Docker Pipeline

Using Docker with Continuous Delivery in Oracle Cloud [CON4824]
Functional Walkthrough
Oracle Container Cloud Service
Configure Instances as Needed

- Create instances of Oracle Container CS quickly. Use these either long term or short term. Teams can have their own instances.
- Access in Oracle Cloud My Services
Control & Pool Resources

- Create Resource Pools - Groups of Hosts - to match your needs for full Lifecycle Deployments
- Launch your applications into Resource pools

Pools can align with your DevOps cycle

Pools can be shared resources
Add Private Registries

- The Docker Public registry is defined by default
- Add registries to access your organization’s images

Users can add as many registry definitions as needed.
Application Deployment

Services (container templates)

- Select from one of many pre-configured Services
- Edit or Create New Services via UI, Docker Run or Docker Compose YAML
- Leverage images from Docker Hub or your Private Registry

Compose via UI, Docker Run or YAML
Services (container templates)

- Copy and Paste an existing “Docker Run” command or Docker Compose based YAML
- Switch between views (UI, Run or YAML) as needed
Stacks (application templates)

- Select from one of many example application Stacks included
- Stacks are sets of services that are ready to run as containerized applications

Stacks are deployable in 1 Click or Further Edited and Saved as a New Stack
Stacks (application templates)

- Easily edit existing Stacks to change such things as orchestration defaults

Drag and Drop Services

Easily Edit YAML
Deploy a Stack in One Click

- Stacks can be quickly deployed with 1 click

Choose Resource Pool to Deploy To

Preset Orchestration based on YAML (can be changed at deployment time)

1 Click Stack Deploy
Orchestrate Easily

- Deployment Status and Health is clearly shown in the UI
- Easily scale running Deployments when needed

Green = Healthy Deployment

Easy Scale Up/Down

Integrated HTTP, TCP and Basic Auth Health checks
Scale Out in a Click

- Container CS provides a super-easy method to scale containers up or down

 Simply change the quantity of Containers to total needed

Automatic deployment across hosts in resource pool per the orchestration policy for that service
Container Orchestration

Built-in Service Discovery

- Container CS provides integrated Service Discovery, so application containers know where to find dependent resources / services

Automatic registration of host ip and port for service
Healthy Deployments

• Container CS supplements the reported container status with integrated Health checks
Dashboard – Unified View

- Views across all Docker Resources
- Quickly Identify Problem Areas

Unhealthy App Deployment

Over Utilization of Resources
Easily View Container Logs

- Container CS provides the user with an easy way to view container logs.
- No need to navigate a CLI and learn Docker commands
Auditing / Monitoring

- Container CS provides built-in Monitoring and Event Logging

Integrated Host Monitoring

Integrated Event Log
Questions?