Demystifying a Red Hat OpenShift environment with z/OS or VSE knowledge

Wilhelm Mild
IBM Executive IT Architect
IBM R & D Lab Germany
wilhelm.mild@de.ibm.com

Virtualization vs. application isolation

Virtualization and application isolation

Virtualization:

- Infrastructure oriented
- Virtual server resource management
- Several applications per server
- Isolation in virtual servers
- Persistence via Virtual server
- Security in virtual server

Application isolation:

- Service oriented
- Application resources managed inside a virtual server
- Solution decomposed into several application units / modules
- Isolation and persistence on application level depends on the implementation
- Security based on access security in OS and Virtualization layer

Virtualization vs. application isolation

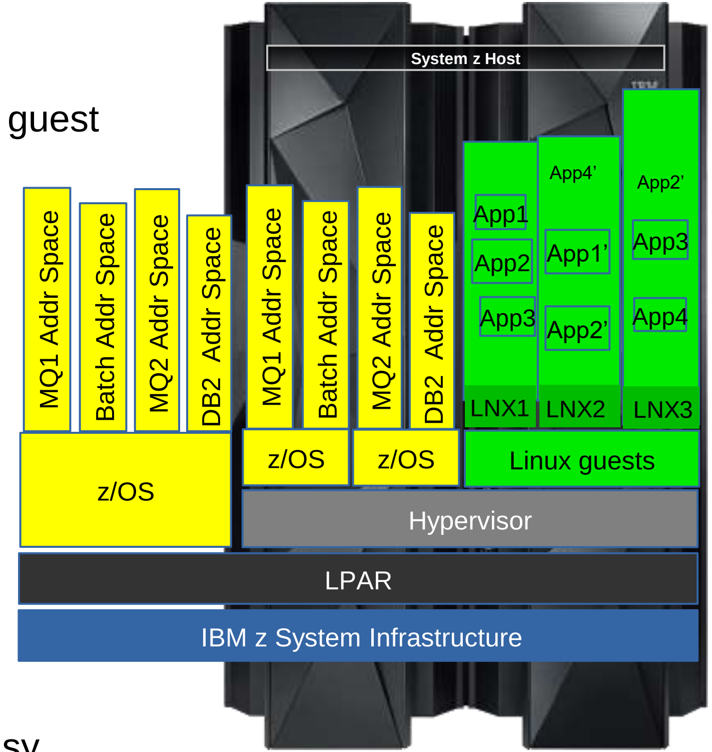
z/OS and Linux applications

In z/OS:

- LPAR or z/VM Hypervisor virtualizes z/OS & VSE Systems as guest
- applications are running in address spaces / partitions
- address space / partition characteristics:
  - isolate applications
  - enable application control, horizontal scalability
- applications can share data
- address space priorities allow application prioritization
- WLM the main component for workload and priorities

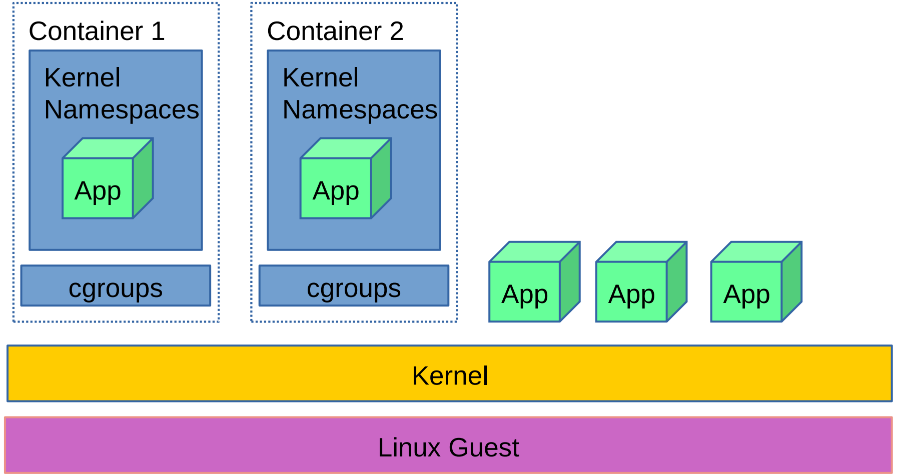
In Linux:

- LPAR or z/VM & KVM Hypervisors virtualization is for Linux guest systems not for pure application isolation
- applications are running as processes
- no hard isolation between processes
- multiple start of a single application on the same kernel not easy
- *Linux Containers – concept to improve application isolation their scalability and flexibility*



Linux control groups and namespaces demystified

- To simplify:
  - “**cgroups**” will allocate & control resources in your container
    - CPU
    - Memory
    - Disk I/O throughput
  - “**namespace**” will isolate
    - process IDs
    - Hostnames
    - User IDs
    - network access
    - interprocess communication
    - filesystems



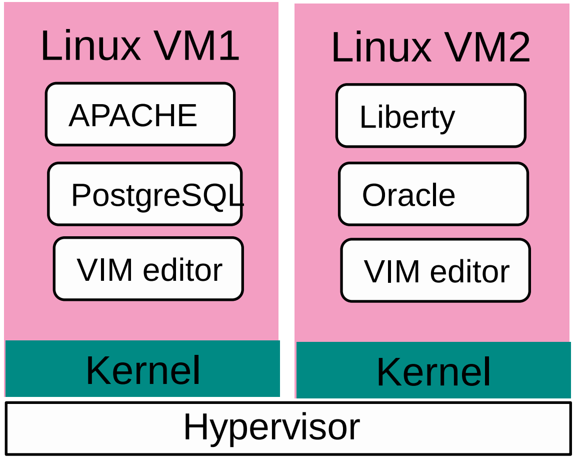
Linux applications in virtual servers vs. Containerized applications

Virtualization, usually provides a **high level of isolation** and security as all communication between the guest and host is **through the hypervisor**. Linux on IBM zSystems runs in LPAR or Virtual machines.

- ➢ It is also usually slower than containers and incurs some **overhead due to the infrastructure emulation**.

Containers, reduce the virtualization overhead, the level of virtualization called "**container virtualization**" was introduced which allows to run **multiple isolated user space instances on the same kernel**.

- ➢ **Containers is a layered approach and uses copy-on-write filesystems**



Linux VM1

APACHE

PostgreSQL

VIM editor

Kernel

Linux VM2

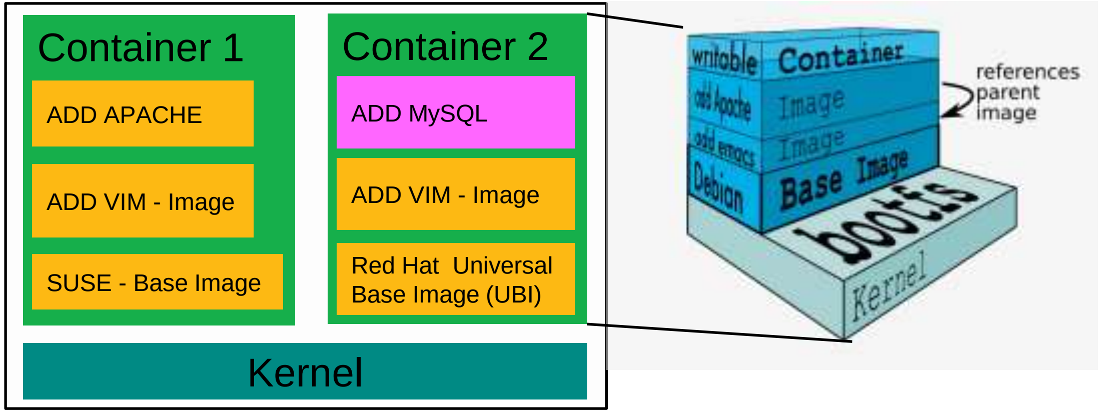
Liberty

Oracle

VIM editor

Kernel

Hypervisor



Container 1

ADD APACHE

ADD VIM - Image

SUSE - Base Image

Container 2

ADD MySQL

ADD VIM - Image

Red Hat Universal Base Image (UBI)

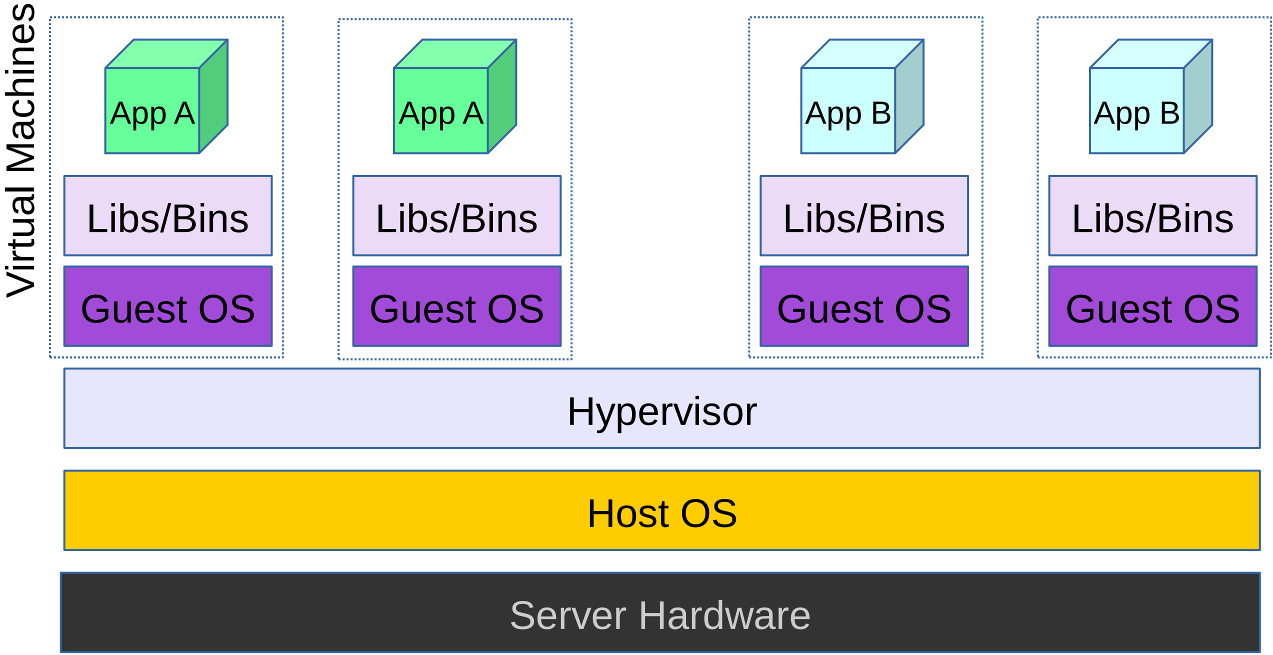
Kernel

Containers are orchestrated with other tools than VMs

Linux Virtual Machines & applications

Virtualization - Infrastructure oriented

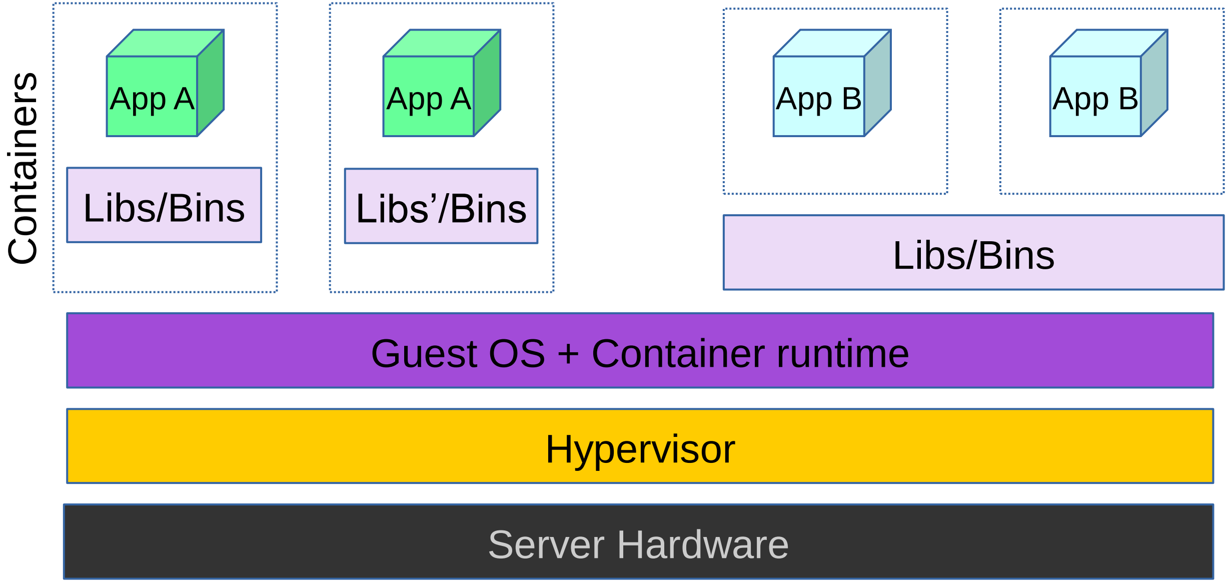
- ➢ Customers have virtualized their servers to gain efficiencies
- ➢ Focus is on virtual server resource management
- ➢ Few applications per Guest VM / Operating System instance
- ➢ Provides application isolation
  - An application failure does not adversely affect other applications residing in other guest VMs
- ➢ Provides persistence across server restarts



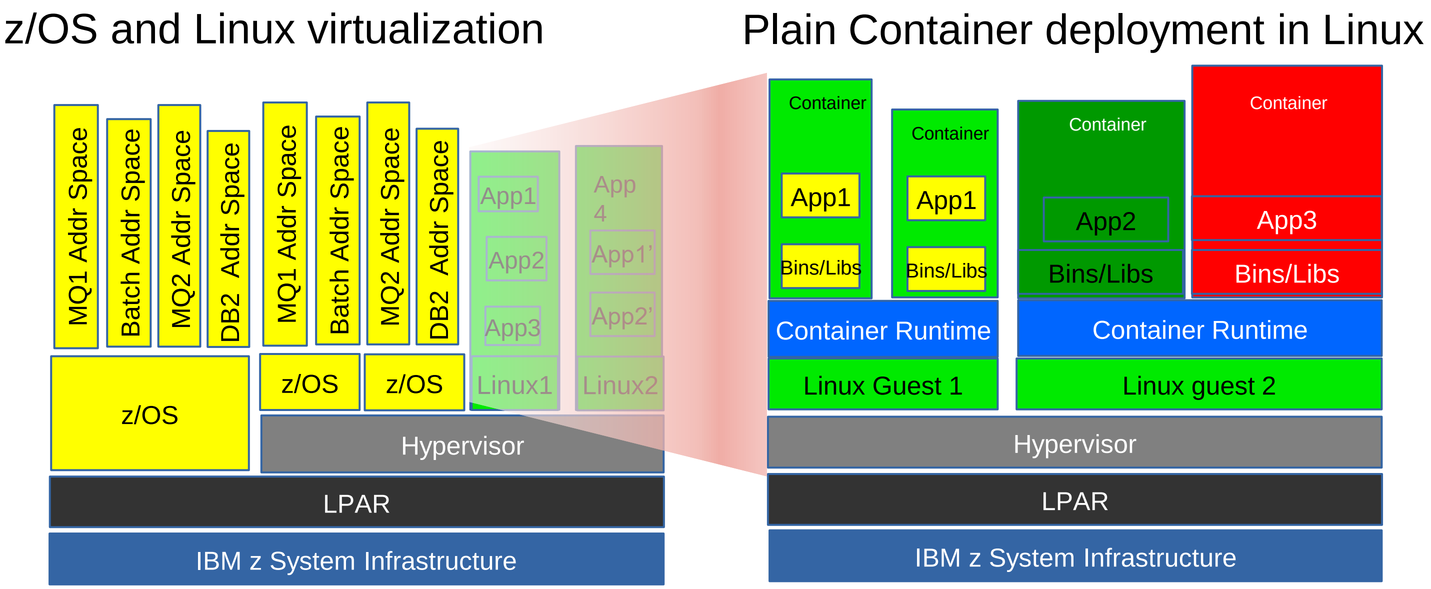
Linux Applications in Container

Containers - Service oriented

- Application-centric - infrastructure resources are assumed to be already in place
- Focus is on application isolation / management
- Containers need a Guest VM or Operating environment
- Have specific DevOps advantages
- Provide a very dynamic application deployment model



Virtualization vs. Container application isolation in IBM Z



Virtualization:

- Infrastructure oriented
- Virtual server resource management
- Several applications per server
- Isolation per virtual server

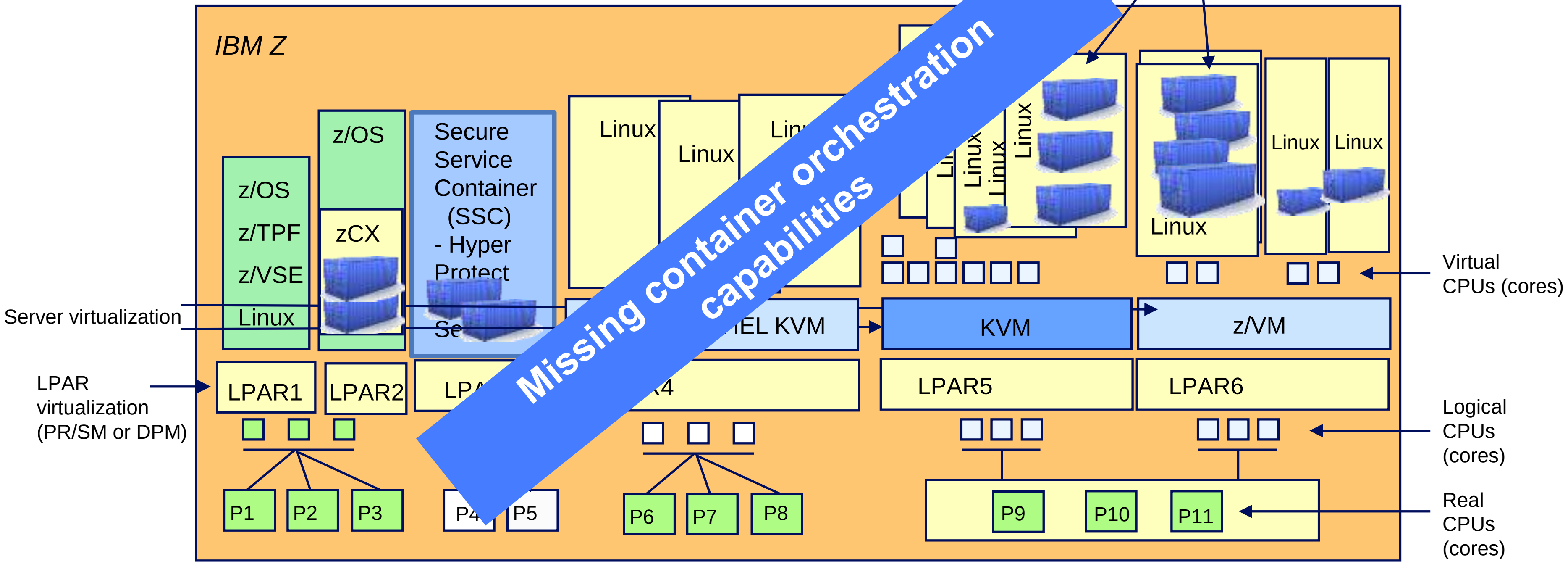
Containers:

- Service oriented
- Application management via container
- Solution decomposed into several units
- Dynamic, isolation in container

IBM zSystems Virtualization and Container options

Server virtualization. There are typically dozens or hundreds of Linux servers in a LPAR virtualized using z/VM or KVM or SSC.

Application isolation.
There ar typically thousands of
Cont n Linux on IBM Z.



Missing container orchestration capabilities

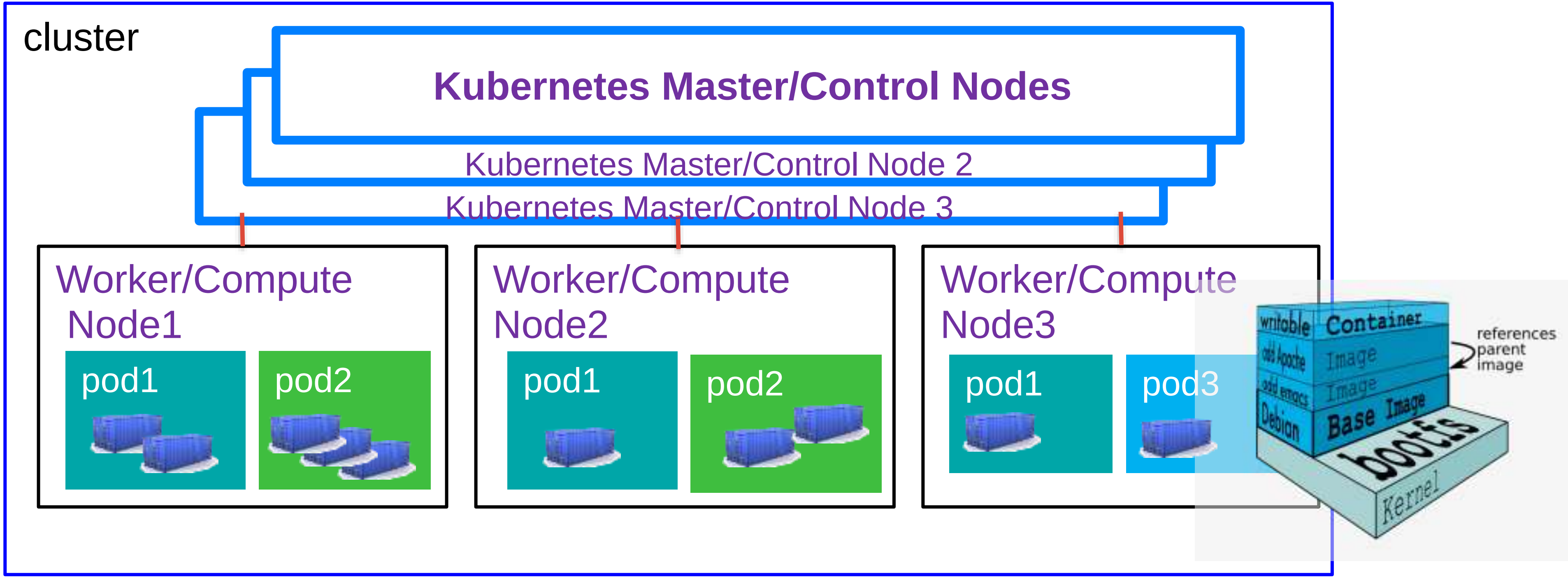
P1 – P11 are General Purpose Processor Units (GPU -> core) or Integrated Facility for Linux (IFL) Processors (IFL -> core)

Note: - LPARs can be managed by traditional PR/SM in IBM Z and additional with Dynamic Partition Manager (DPM) in LinuxONE

© Copyright IBM Corporation 2020

Kubernetes (K8S) – container orchestration

- defines itself in a cluster format for HA per design



cluster

Kubernetes Master/Control Nodes

Kubernetes Master/Control Node 2

Kubernetes Master/Control Node 3

Worker/Compute Node1

pod1

pod2

Worker/Compute Node2

pod1

pod2

Worker/Compute Node3

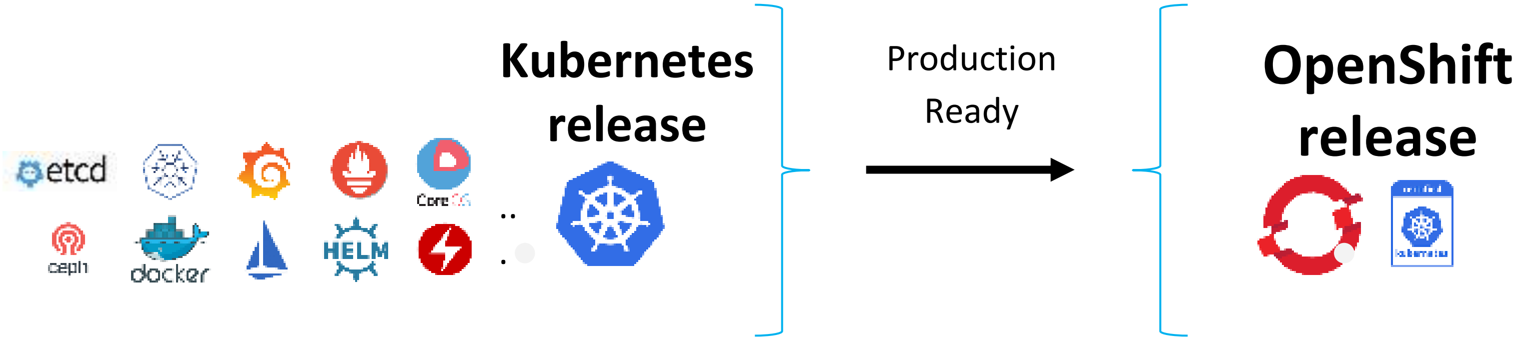
pod1

pod3

Kubernetes is not running container – it orchestrates them

Kubernetes is THE container Orchestration tool

Red Hat OpenShift is trusted enterprise Kubernetes



Kubernetes release

Production Ready

OpenShift release

- Hundreds of defect and performance fixes
- 200+ validated integrations
- Certified container ecosystem
- Over 9-years enterprise life-cycle management
- Red Hat is one of the leading Kubernetes contributor since day 1

Red Hat OpenShift Container Platform (RHOCP)

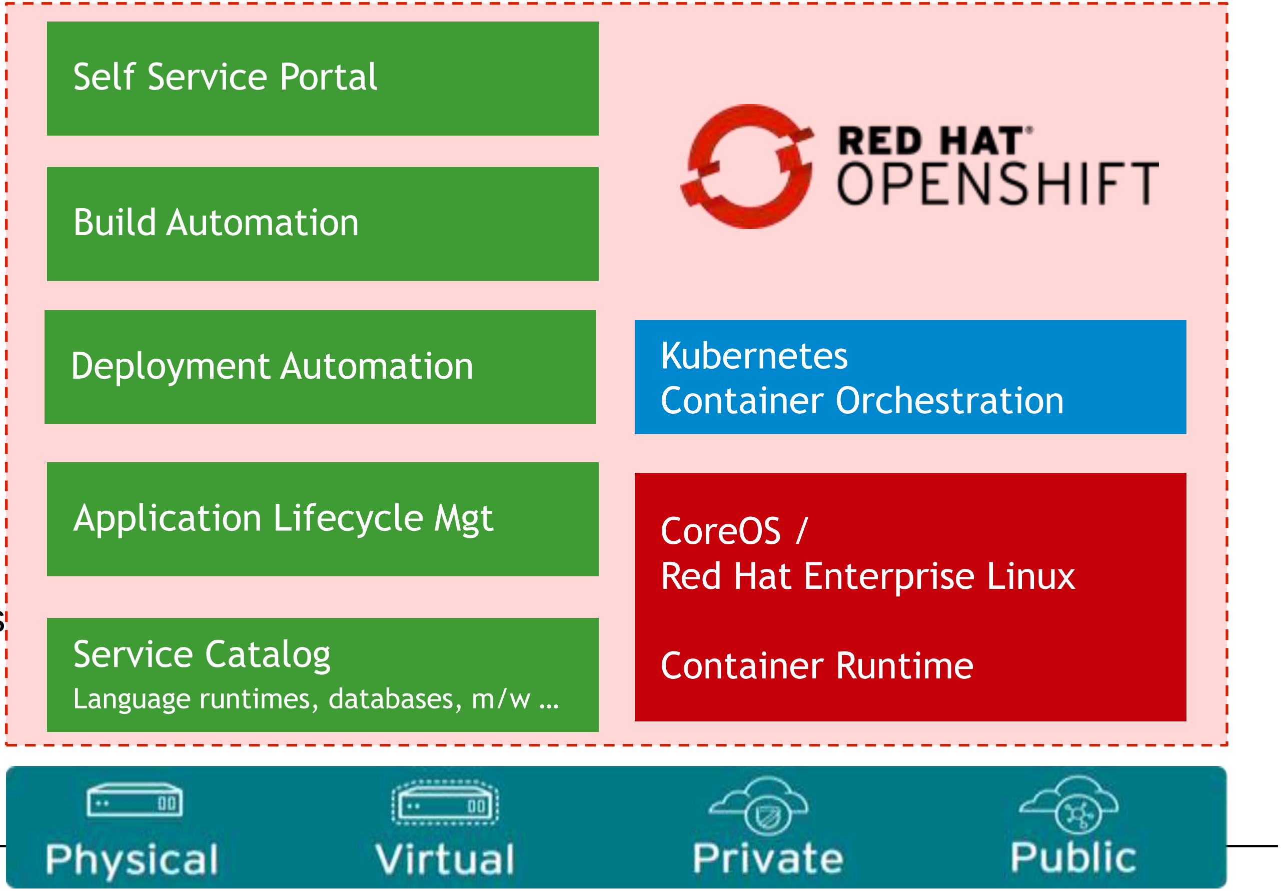
to Build, Deploy, Manage Containerized, Cloud Native Apps that can Run Anywhere

Red Hat OpenShift
- The enterprise
  Kubernetes Platform

- Runs on IBM zSystems, IBM Power, x86 and public clouds

- Is THE platform for
  Life cycle management
  of containerized applications

- Has capabilities for extensions
  to manage Clusters on
  different Architectures
  with RHACM



Self Service Portal

Build Automation

Deployment Automation

Application Lifecycle Mgt

Service Catalog
Language runtimes, databases, m/w …

RED HAT OPENSHIFT

Kubernetes
Container Orchestration

CoreOS /
Red Hat Enterprise Linux

Container Runtime

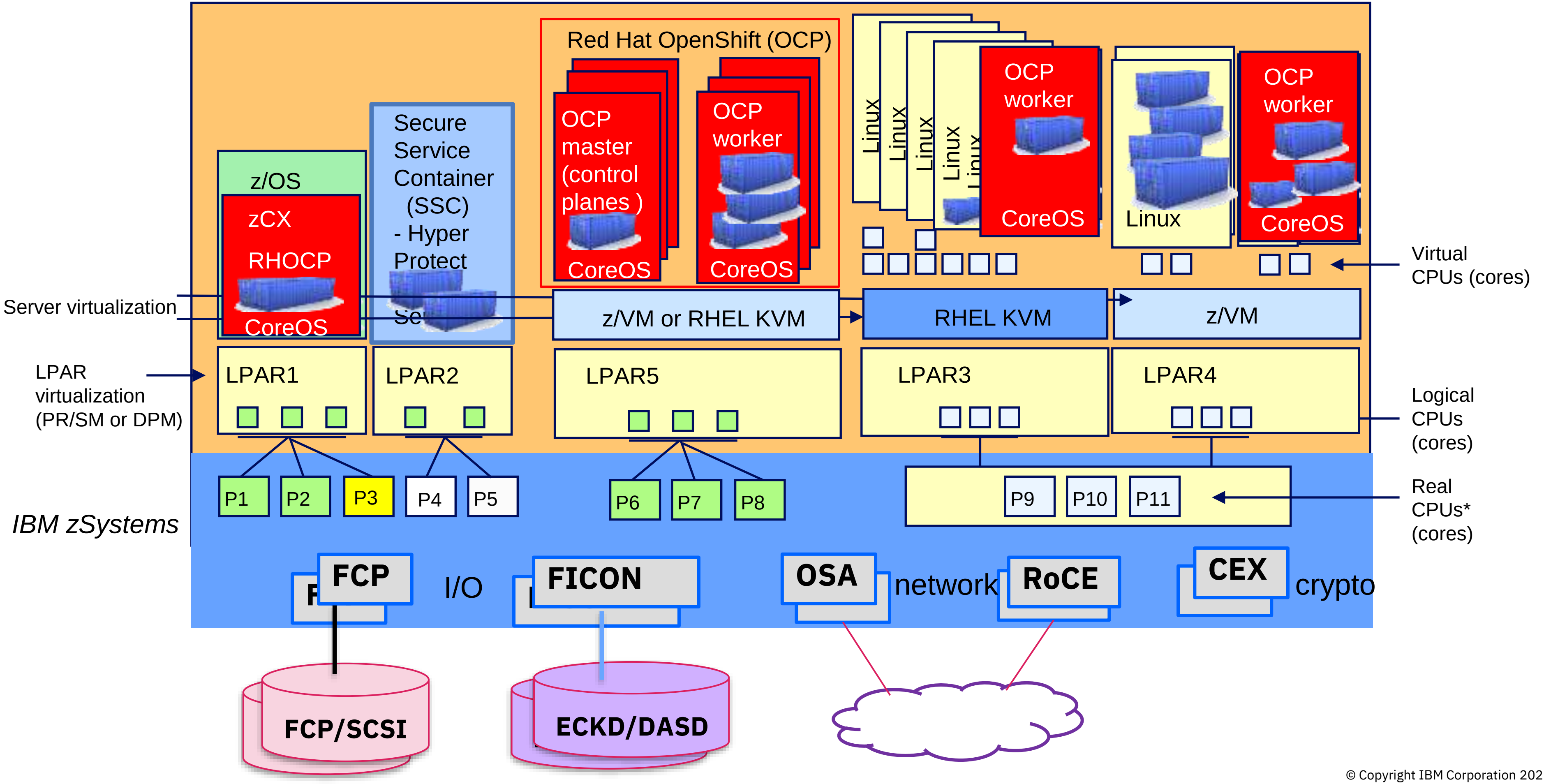
Physical

Virtual

Private

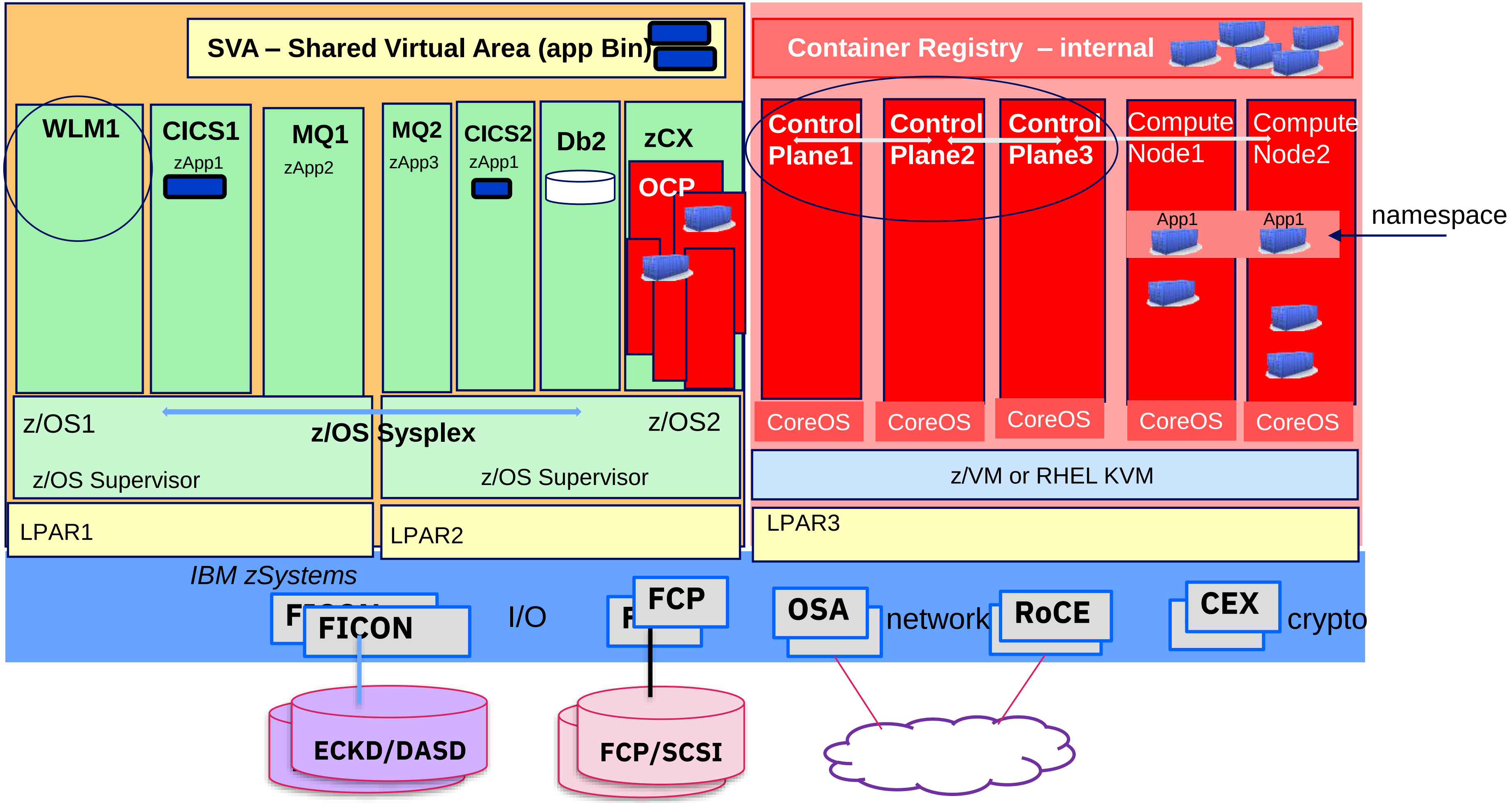
Public

IBM zSystems Virtualization and Container options



© Copyright IBM Corporation 2022

IBM zSystems Virtualization and Container options



SVA – Shared Virtual Area (app Bin)

WLM1

CICS1
zApp1

MQ1
zApp2

MQ2
zApp3

CICS2
zApp1

Db2

zCX
OCP

z/OS1

z/OS Sysplex

z/OS2

z/OS Supervisor

z/OS Supervisor

LPAR1

LPAR2

Container Registry – internal

Control Plane1

Control Plane2

Control Plane3

Compute Node1
App1

Compute Node2
App1

namespace

CoreOS

CoreOS

CoreOS

CoreOS

CoreOS

z/VM or RHEL KVM

LPAR3

IBM zSystems

FICON

I/O

FCP

OSA

network

RoCE

CEX

crypto

ECKD/DASD

FCP/SCSI

© Copyright IBM Corporation 2023

Container and OpenShift on IBM zSystems and LinuxONE

Container and OpenShift to z/OS Terminology:

- **Openshift Cluster** is a Kubernetes cluster -> set of **3 Control Plane Nodes** represented by Virtual machines in z/VM or RH KVM and a number of min 2 **Compute Nodes**, VMs as well
  - z/OS Basic Sysplex - a cluster of tightly-coupled independent instances of the z/OS operating system environments with multiple Address Spaces – for high availability of z/OS applications in the Sysplex
- **Openshift Control Plane Nodes** – control and manage the status of the cluster and applications, controlling restart of failing applications
  - z/OS WLM in the Sysplex – controls prios, workload resources and dispatcher
- **Openshift Compute Nodes** – separate workload Nodes in the OCP cluster
  - z/OS Address Spaces across the Sysplex
- **Openshift Pod –** a group of containers, running in an Openshift Compute Node. Multiple Pods run per Node, one or multiple containers per pod
  - z/OS Address Space with multiple Applications running in it
- **Container Registry:** The library which contains the container images that can be instantiated in the OpenShift Cluster.
  - z/OS – a Shared Virtual Area (SVA) containing reentrant application binaries

Container and OpenShift on IBM zSystems and LinuxONE

Container and OpenShift to z/OS Terminology:

- **OpenShift or Kubernetes Namespace** – a logical group of pods which represent an application entity
  - z/OS application deployed highly available in a Sysplex
- **Container Image**: the image from which containers are instantiated. An image represents an App logic as read-only, layered – like a golden image. Images can be created with container dev tools.
  - Binary image (Phase), reentrant, in SVA, that can be instantiated in multiple Address Spaces at the same time
- **Container**: a running instance, from a container image.
  - z/OS – an running application in an Address Space (MQ App, DB2, COBOL app.)
- **Openshift High Availability –** per design Kubernetes and Openshift are implemented with high availability concepts for the container software and the cluster
  - z/OS Sysplex and GDPS build an unmatched entity for availability from infrastructure to the application level.

z/OS CICS is unique – > there is no transactional control component in OpenShift

*

Kubernetes and Openshift are designed primarily for stateless applications.

Globally: What is a container application related to z/OS concepts

- A Container is a form of isolation for individual applications or workloads
  - z/OS: Can be compared with a z/OS Address Space (new container tool: zospt tool)

- It is a flexible way of deployment of isolated applications
  - z/OS: starting an application in a certain Address Space

- The container is build via a Containerfile (Dockerfile), which contains instructions to include necessary components in the container image that is built
  - z/OS: Use of JCL, binary modules and link books to build applications

- Based on a Container image, a container can be instantiated or transferred
  - z/OS: Binary member copy – is not containing application dependencies like IP or runtime lib.

- A Container (runtime) is not persistent
  - z/OS: A workload in an Address Space can be HA in a Parallel Sysplex but is not persistent to itself

- A Container can be used as base to build another container image

Everything should be made as simple as possible,
but no simpler

– Albert Einstein

Questions?

Wilhelm Mild

IBM Executive IT Architect

The Open Group Open Certified Distinguished Architect

IT Architecture Chief/Lead IT Architect

IBM Deutschland Research
& Development GmbH
Schönaicher Strasse 220
71032 Böblingen, Germany

Office: +49 (0)7031-16-3796
wilhelm.mild@de.ibm.com

Trademarks & Disclaimer

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

IBM, the IBM logo, BladeCenter, Calibrated Vectored Cooling, ClusterProven, Cool Blue, POWER, PowerExecutive, Predictive Failure Analysis, ServerProven, System p, System Storage, System x , z Systems, WebSphere, DB2 and Tivoli are trademarks of IBM Corporation in the United States and/or other countries. For a list of additional IBM trademarks, please see http://ibm.com/legal/copytrade.shtml.

The following are trademarks or registered trademarks of other companies: Java and all Java based trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries or both Microsoft, Windows,Windows NT and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both. Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
UNIX is a registered trademark of The Open Group in the United States and other countries or both. Linux is a trademark of Linus Torvalds in the United States, other countries, or both. Cell Broadband Engine is a trademark of Sony Computer Entertainment Inc. InfiniBand is a trademark of the InfiniBand Trade Association.
Other company, product, or service names may be trademarks or service marks of others.

NOTES: Linux penguin image courtesy of Larry Ewing (lewing@isc.tamu.edu) and The GIMP

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Users of this document should verify the applicable data for their specific environment. IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
Information is provided “AS IS” without warranty of any kind. All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices are suggested US list prices and are subject to change without notice. Starting price may not include a hard drive, operating system or other features. Contact your IBM representative or Business Partner for the most current pricing in your geography. Any proposed use of claims in this presentation outside of the United States must be reviewed by local IBM country counsel prior to such use. The information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any

Notice Regarding Specialty Engines

Any information contained in this document regarding Specialty Engines ("SEs") and SE eligible workloads provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zIIPs, zAAPs, and IFLs). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the "Authorized Use Table for IBM Machines" provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html ("AUT").

No other workload processing is authorized for execution on an SE.

IBM offers SEs at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.