# Container technologies on IBM Z and LinuxONE and their Orchestration

**Wilhelm Mild**
IBM Executive IT Architect for Mobile, IBM Z and Linux
IBM R&D Lab, Germany

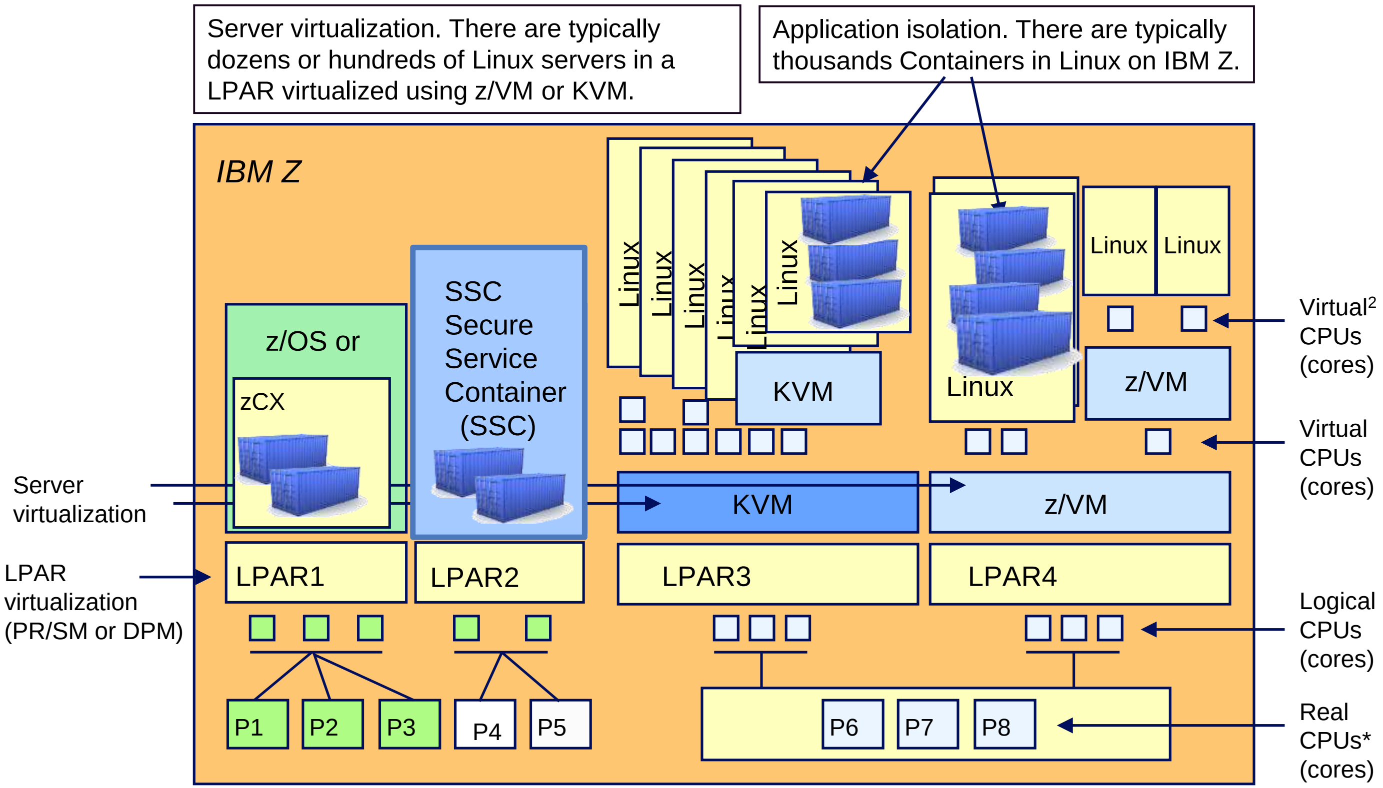
# Agenda

- **Container technologies and Ecosystem**

- **Container Orchestration**

Kubernetes

# IBM Z Virtualization options and Container




P1 – P8 are Central Processor Units (CPU -> core) or Integrated Facility for Linux (IFL) Processors (IFL -> core)

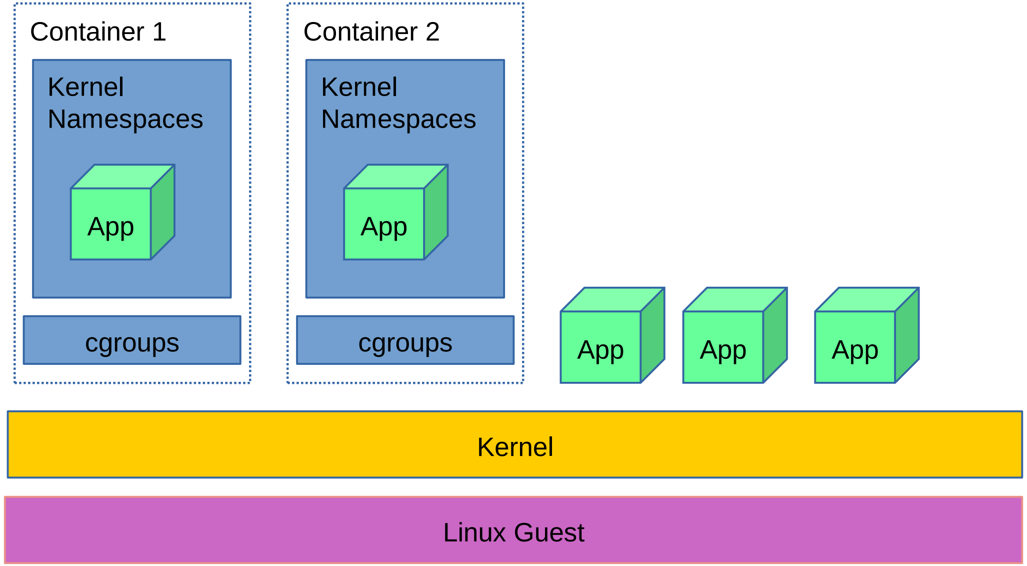
* - One shared Pool of cores per System only

Note: - LPARs can be managed by traditional PR/SM

# Linux Containers - based on control groups and namespaces for isolation

The goal was to offer a Linux distro and vendor neutral environment for the development of Linux container technologies.

- 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



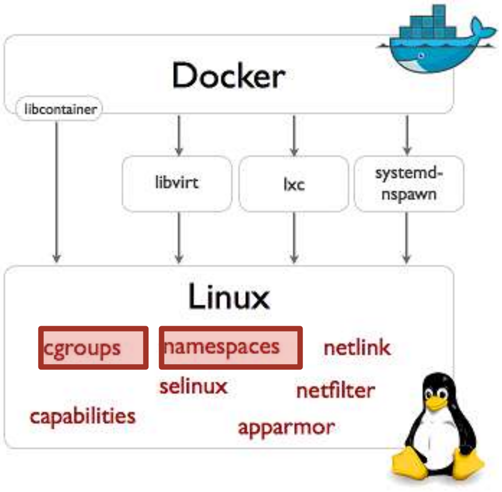
# Linux Containers vs. virtual server

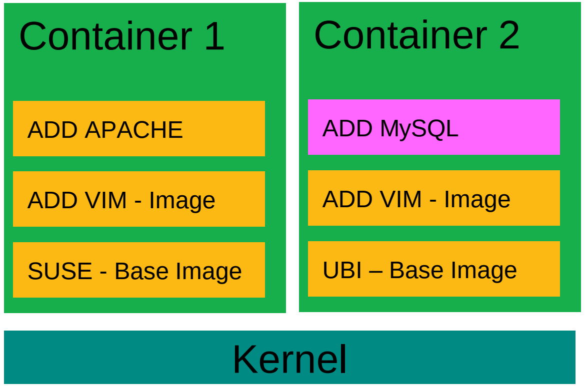
**Virtualization,** usually provides a **high level of isolation** and security as all communication between the guest and host is **through the hypervisor**.

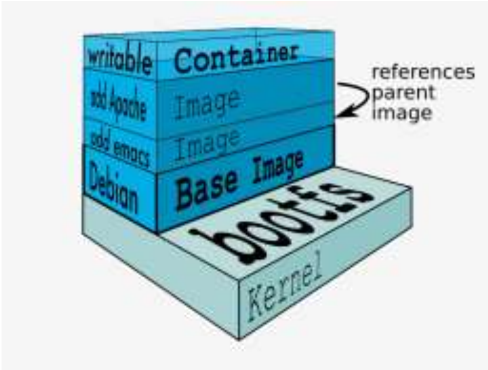
- ➢ It is usually slower and incurs some **overhead due to the infrastructure emulation**.

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

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



# Docker and Containers

In 2014, Docker teamed with Canonical, Google, Red Hat, and Parallels to create a standardized open-source program libcontainer that allows containers to work within Linux namespaces and control groups (cgroups) without needing administrator access.

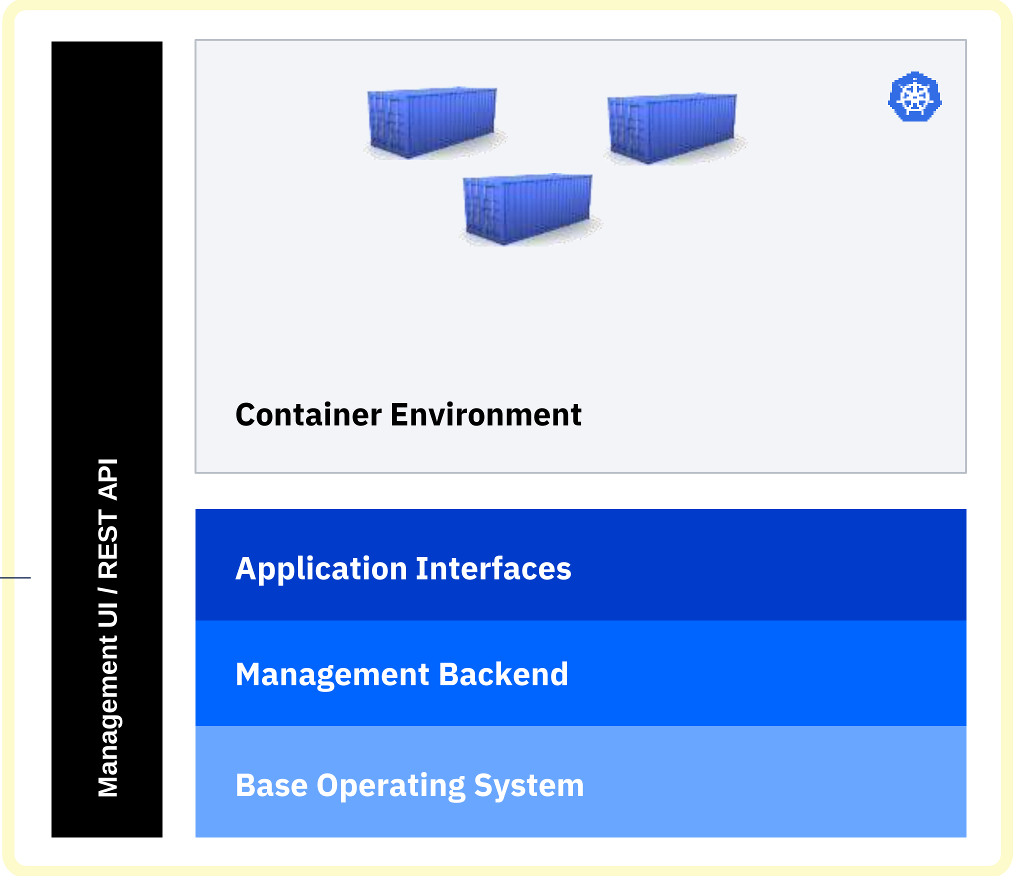
**Docker is one implementation of Linux containers and their management with the goal to:**

- Open, portable, light-weight run-time and packaging tool
- Easy build and ship complex application, without worrying about infrastructure differences or interference from other software stacks
- Quickly and reliably deploy and run applications on many infrastructures
- Essential for horizontally scaling apps in the cloud

# IBM Secure Service Container (SSC) – Hyper Protect services everywhere

- **SSC is a special LPAR** and provides simplified mechanism for fast deployment and management of packaged solution
- Provides tamper protection during installation and runtime
- Ensures confidentiality of data and code -at flight and at rest
- Management provided via Remote APIs (RESTful) and web interfaces only
- Enables containers to be delivered via distribution channels

IBM Secure Service Container Appliance




**Deploy your container workload in a highly secure environment**

# Enterprise IBM Hyper protect services based on Containers in SSC

**IBM Cloud Hyper Protect Crypto Services**
Infuse the highest level of security with data encryption and key management capabilities into your apps. http://ibm.biz/hpcrypto

**IBM Cloud Hyper Protect DBaaS**
Retain your data in a fully encrypted client database without the need for specialized skills.
http://ibm.biz/hpdbaas

IBM Db2® Analytics Accelerator is a high-performance component tightly integrated with Db2 for z/OS® for high-speed processing for complex Db2 queries and analytic workloads.
https://www.ibm.com/products/db2-analytics-accelerator

Hyper secure services are based on IBM Secure Service Containers,
a special type of Hyper protect LPAR in IBM Z.

**IBM Blockchain Platform**
Deploy Blockchain on IBM Cloud in a Hyper Secure environment on LinuxONE.
https://www.ibm.com/blockchain/platform

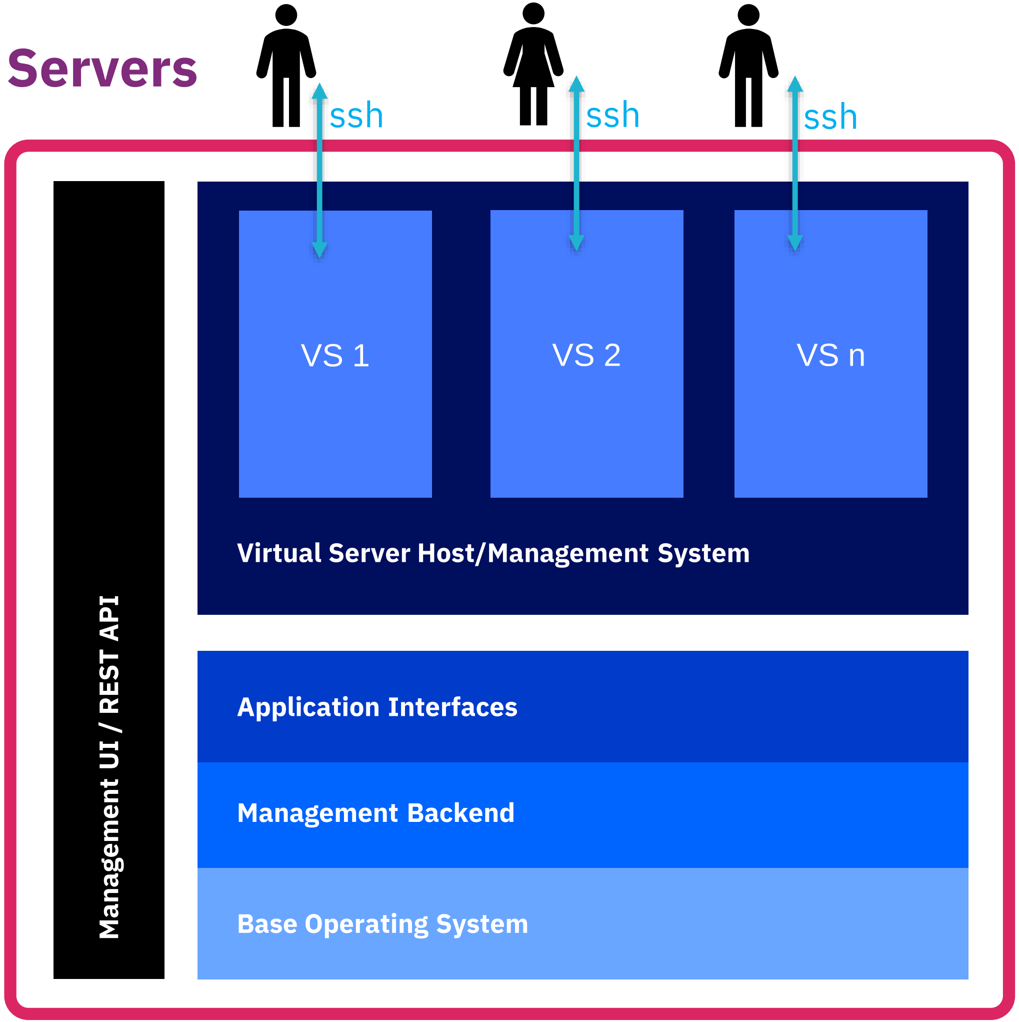
**IBM Hyper Protect Virtual Servers**
Create Linux VMs with own public ssh key to maintain exclusive access to code and data
http://ibm.biz/hpvserv

**IBM Secure Service Container for ICP**
Deploy Kubernetes workloads on a highly secure platform built for confidential computing.
https://www.ibm.com/support/knowledgecenter/SSUPZ7

https://www.ibm.com/cloud/hyper-protect-services

# IBM Cloud Hyper Protect Virtual Servers

- Rapidly provision a Virtual Server running on LinuxONE in the IBM Cloud
- Authentication is done via ssh keys
  → No password is exposed to IBM
- Our system administrators do not have access to the data within the Virtual Servers and the hosting OS
- Ubuntu Operating System
- Built on IBM Secure Service Container to enforce confidentiality
- Available now




[https://www.ibm.com/cloud/virtual-servers](https://www.ibm.com/cloud/virtual-servers)

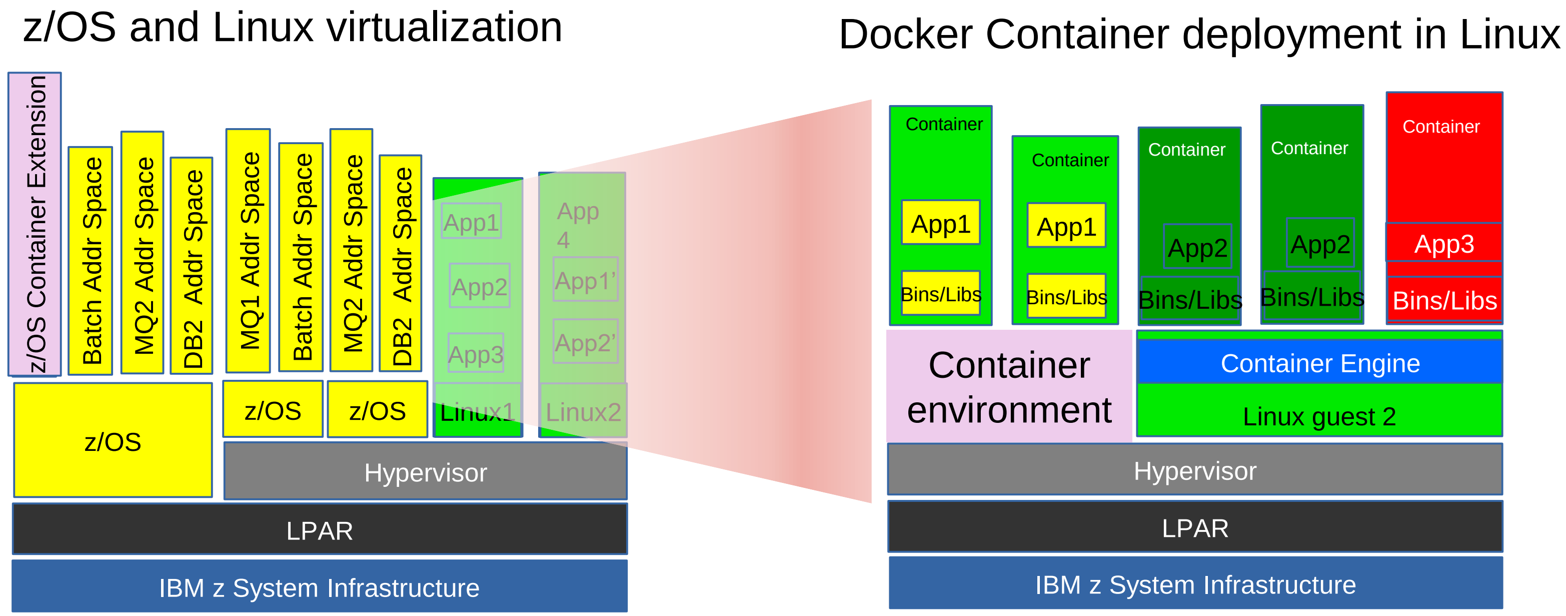
# IBM Cloud Hyper Protect Virtual Servers (HPVS)

IBM Cloud Virtual Server versions - classic infrastructure

| | | |
|---|---|---|
| Public virtual servers | Multitenant virtual server deployments can give you rapid scalability and higher cost-effectiveness, with predefined sizes. | Try public virtual servers |
| Transient virtual servers | Single-tenant virtual servers offer rapid provisioning and flexibility, enabling virtual server-placement control for workloads requiring physical isolation. | Try transient virtual servers |
| Reserved virtual servers | Reserve up to 20 specific virtual server instances for guaranteed capacity. | Try reserved virtual servers |
| Dedicated virtual servers | Single-tenant with rapid provisioning and flexibility, enabling virtual server placement control for workloads requiring physical isolation. | Try dedicated virtual servers |

https://www.ibm.com/cloud/virtual-servers

# Application isolation is long tradition 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

New

# Container in IBM z/OS Version 2 Release 4

- z/OS V2.4 introduced **IBM z/OS Container Extensions**,
  - **execute Linux® on IBM Z Docker container in z/OS, alongside existing z/OS applications and data.**

- **z/OS Container Extensions**:
  - enable application developers to develop and data centers **to operate popular open source packages, Linux applications, IBM software, and third-party software** together with z/OS applications and data-leveraging industry standard skills.

- Enables the **capability to integrate z/OS more easily into private and multicloud environments**
  - with improvements to deliver a more robust and highly available IBM Cloud™ Provisioning and Management for z/OS and cloud storage access for z/OS data

https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&supplier=897&letternum=ENUS219-344

# z/OS Container Extensions– A turn-key Virtual Docker Server

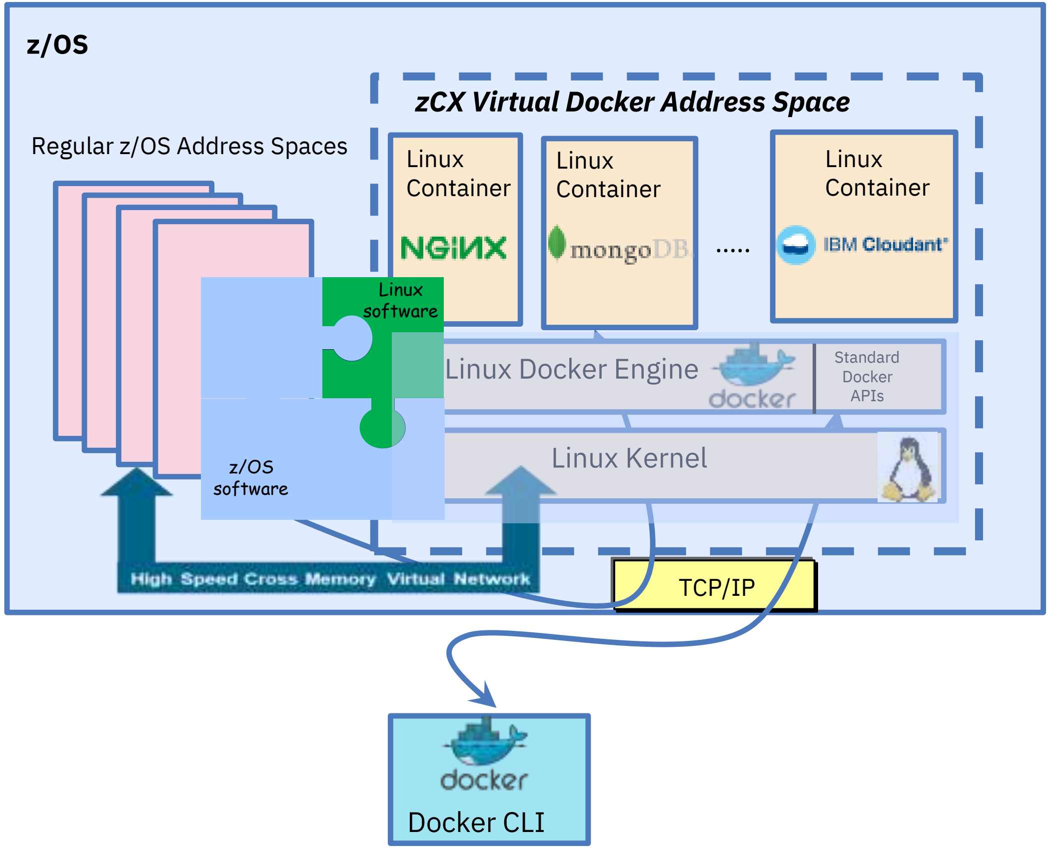
New

**Pre-packaged Docker Environment** provided by IBM
- Includes Linux and Docker Engine components
- Supported directly by IBM
- Can include clustering and registry capabilities
- Initial focus is on base Docker capabilities
- Competitive price/performance (Exploits zIIPs)

**Application developers can deploy software using Docker interface**
- Any software available as a Docker image (s390x) - growing ecosystem available on Docker Hub
- Any home-grown Linux on Z container images
- Using standard Docker interfaces

**Limited visibility into Linux environment**
- No root access
- Access as defined by Docker interfaces
- Limited Linux administrative overhead
- Secure virtual network - SAMEHOST

Also provides IBM and ISVs a means of delivering solutions into this environment
- Requires packaging of software as Docker images

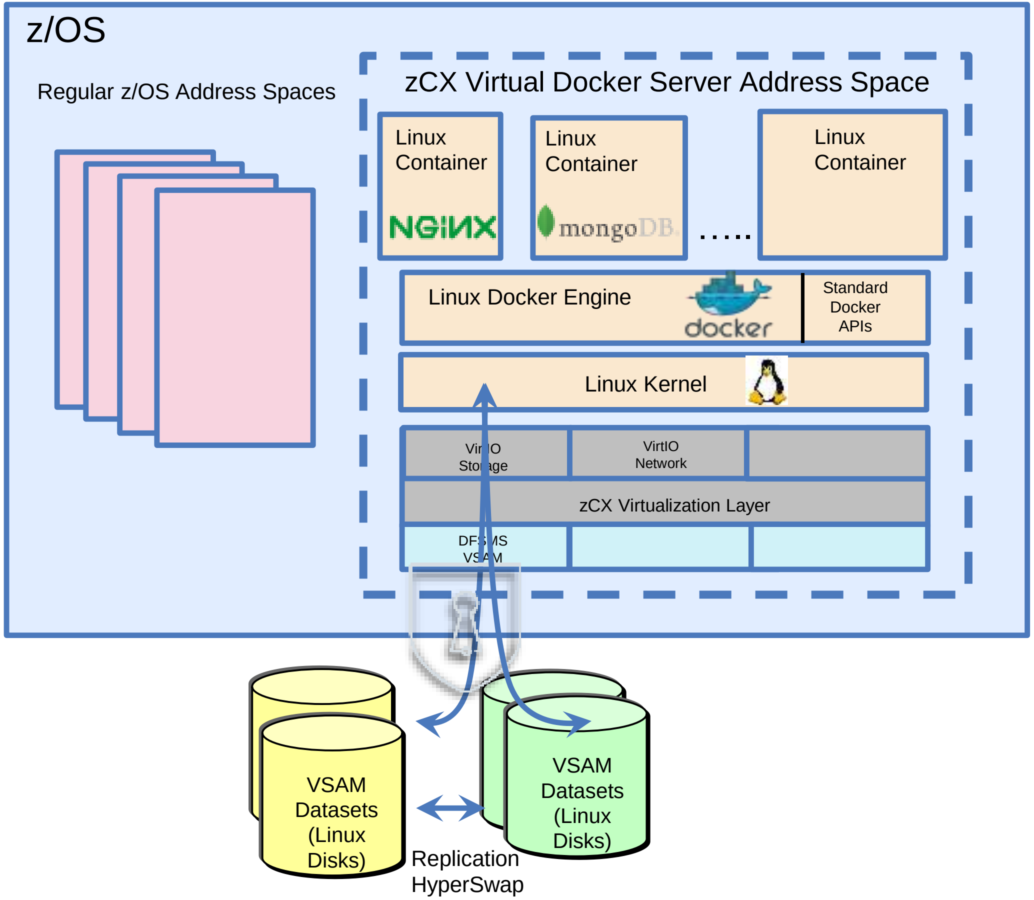


# IBM zCX – z/OS Storage Integration

**z/OS Linux Virtualization Layer:**

- Allows virtual access to z/OS Storage, Network
- Using virtio Linux interfaces
- Allows to support unmodified, open source Linux on Z containerized software

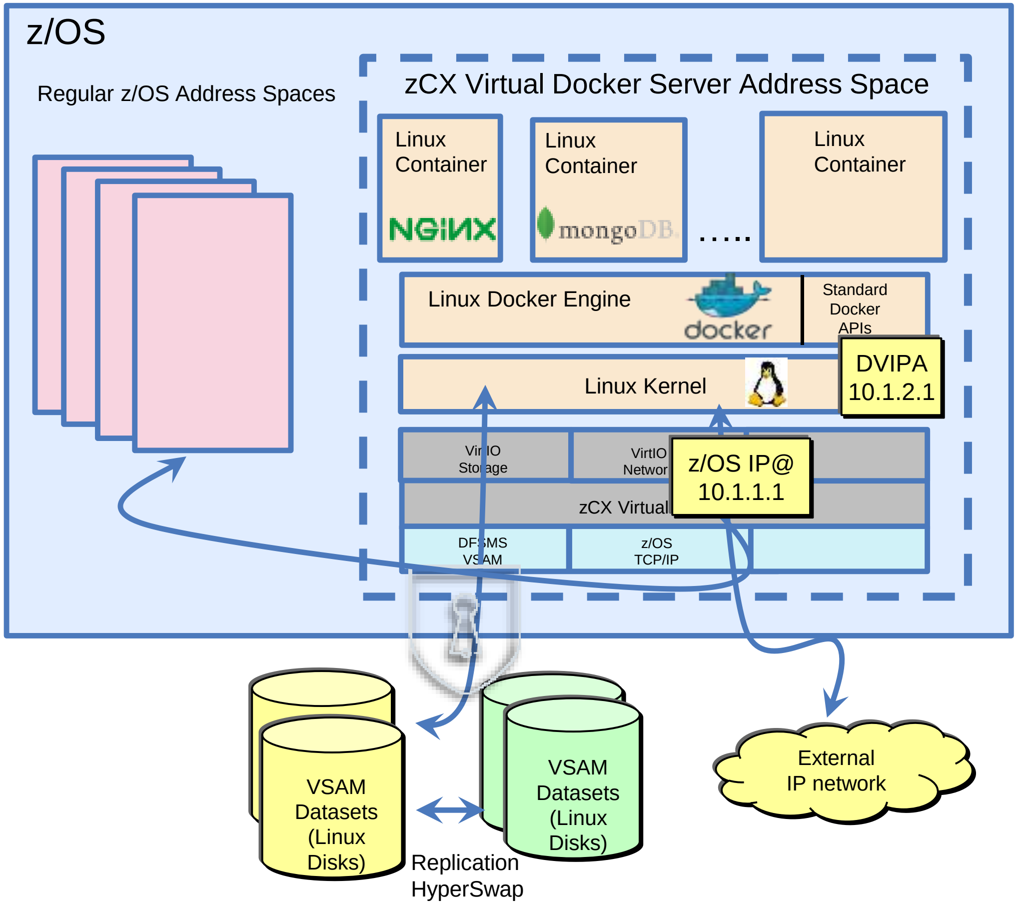
- **Linux storage/disk access (via z/OS owned and managed VSAM datasets)**
  - Leverages latest I/O enhancements (e.g. zHyperLinks, I/O fabric diagnostics, etc.)
  - Built-in host-based encryption
  - Replication technologies and HyperSwap



# IBM zCX – z/OS Network Integration

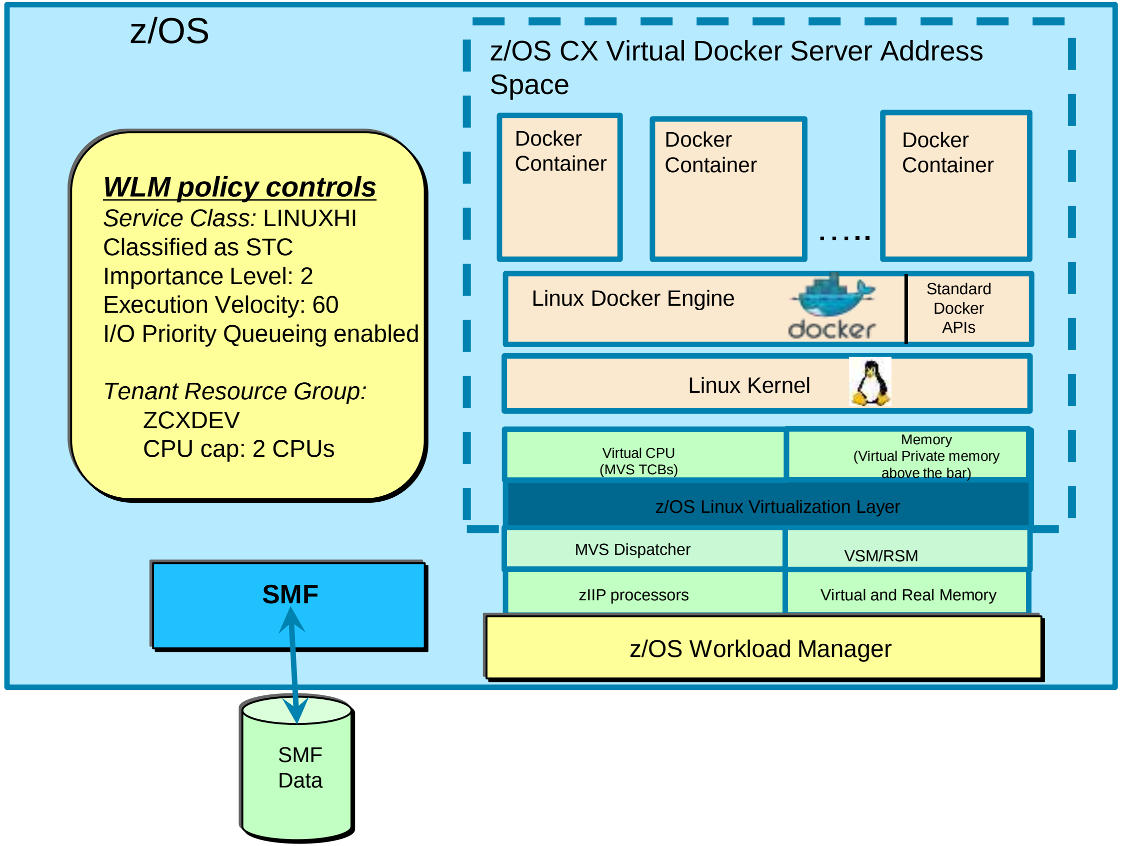
- **z/OS Linux Virtualization Layer:**
  - Allows virtual access to z/OS Storage, Network and Console
  - Using virtio Linux interfaces
    - Stable, well defined interfaces used to virtualize Linux
  - Allows us to support unmodified, open source Linux for z kernels

- **Linux network access via high speed** virtual ***SAMEHOST*** link to z/OS TCP/IP protocol stack
  - Each Linux Docker Server represented by a z/OS owned, managed and advertised Dynamic VIPA (DVIPA)
    - Allows restart of a CX instance in another system in the sysplex
  - Provide high performance network access across z/OS applications and Linux Docker containers – leveraging cross memory
    - All communications between zCX containers and z/OS applications over TCP/IP
  - External network access via z/OS TCP/IP
    - z/OS IP filters to restrict external access



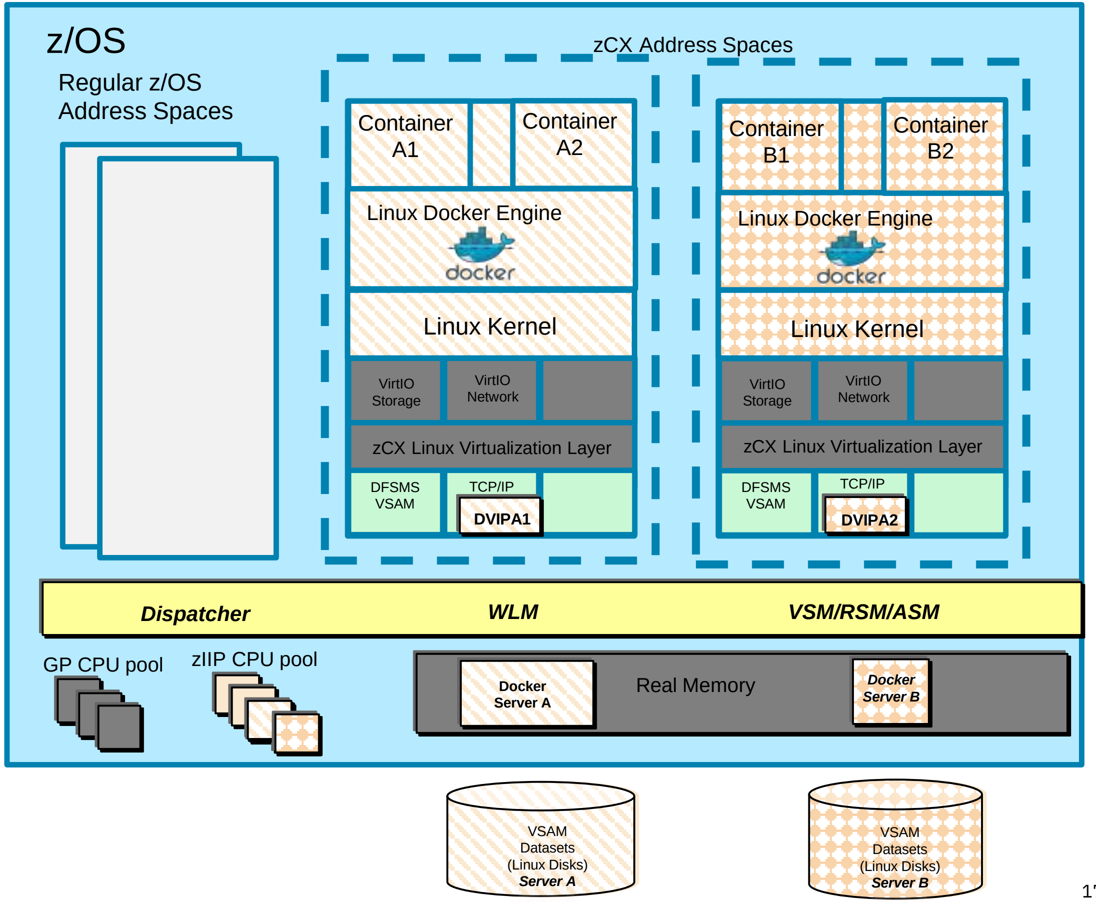
# IBM zCX - CPU, Memory and Workload Management

- Memory Management
  - Provisioned per zCX Docker Server address space
  - Private, above the 2GB bar Fixed Memory
  - Managed by VSM, RSM
- CPU Management
  - Virtual CPUs provisioned to each zCX Docker Server address space
    - Each virtual CPU is a dispatchable thread (i.e. MVS TCB) within the address space
    - zIIP CPU access via MVS dispatcher
  - A zCX instance can host multiple Docker Container instances
- Normal WLM policy and resource controls extend to zCX Docker Server address spaces
  - Service Class association, goals and Importance levels
  - Tenant Resource Group association
    - Optional caps for CPU and real memory
- Normal SMF data available
  - SMF type 30, 72, etc.
  - Enables z/OS performance management and capacity planning



# Deploying Multiple zCX Address Spaces

- Multiple zCX instances can be deployed within a z/OS system:
  - Isolation of applications (containers)
  - Different business/performance priorities (i.e. unique WLM service classes)
  - Capping of resources allocated for related workload (CPU, memory, disk, etc.)
- Each zCX address space:
  - Has specific assigned storage, network and memory resources
  - Shares CPU resources with other address spaces
    - But can influence resource access via configuration and WLM policy controls
- A new Hypervisor built using existing z/OS capabilities
  - The z/OS Dispatcher, WLM and VSM/RSM components manage access to CPU and memory
  - The zCX virtualization layer manages Storage, Network and Console access
    - Using dedicated resources
    - There is no communications across z/OS Linux virtualization layer instances
- Integrated z/OS Capacity Provisioning and Management
  - WLM, CPM, adding/removing CPU and Memory resources



# IBM zCX - Goals & Qualities of Service

### Integrated Disaster Recovery & Planned Outage Coordination

Using z/OS DR/GDPS to cover storage used by Linux automatically, integrated restart capabilities for site failures, etc.

Integrated Planned Outage Coordination

No need to coordinate with non-z/OS administrators when planning a maintenance window, moving workloads to alternate CECs, sites, etc.

### z/OS Storage Resilience

Eliminate single points of failure

Exploit z/OS VSAM which offers transparent encryption, and failure detection with HyperSwap

Configuration validation, I/O health checks,

Automatic exploitation zHyperLink and future z/OS Storage enhancements

### z/OS Networking Virtualization, Security & Availability

Support for VIPAs, Dynamic VIPAs allowing for non-disruptive changes, failover, and dynamic movement of the workload.

High speed and secure communications with Cross-Memory Virtual Network Interface (SAMEHOST)

### z/OS Workload Management, Capacity Planning & Chargeback

WLM: Service Class goals, Business Importance levels, ability to cap resource consumption (CPU and memory)

Capacity Provisioning Manager (CPM) support

SMF support for accounting and chargeback

# Why consider containers on IBM Z ?

- **Highest scalability on one footprint on IBM Z**

- **Most securable platform and containers profit from the capabilities**
- **Encryption performance with the Crypto accelerators and CPACF on each core**
- **New Linux software components and solutions in z/OS**
- **No software-level dependencies** between containers or to the host
- **Re-use** of same components in different Ops scenarios (test, QA, Prod)
- **Micro-services** implementation flexibility

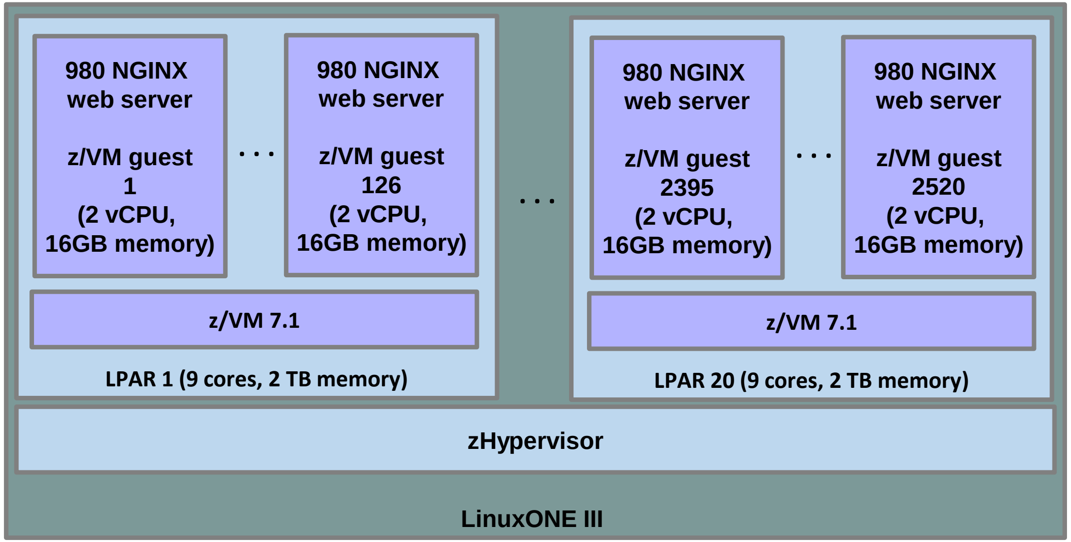
- **Portability and Multi-platform** deployment through generic build description

- **High Density** through lightweight container implementations in Linux kernel

- **Bridges Dev to Ops** with consistent tooling and environment

# Scale-out with Container under z/VM on LinuxONE III

**Scale-out to 2.4 million Docker containers in a single LinuxONE III system**



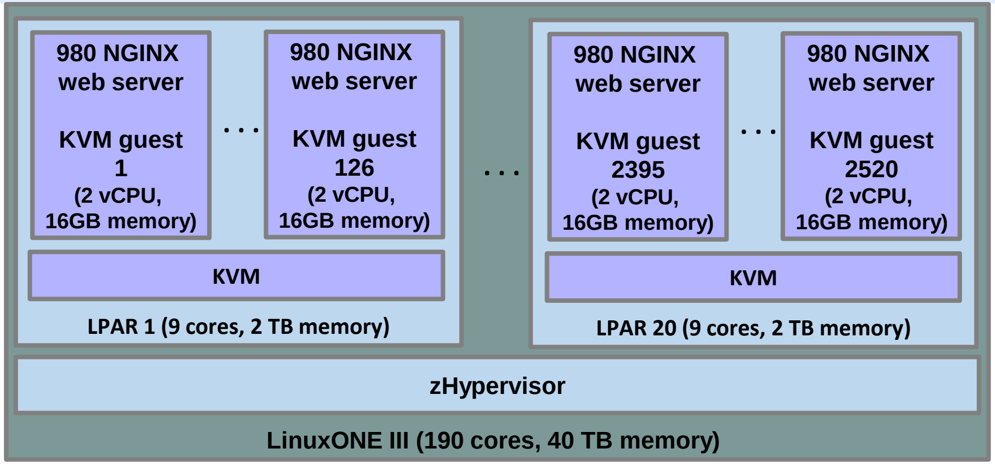

**DISCLAIMER:** Performance result is extrapolated from IBM internal tests running in a LinuxONE III LPAR with 1 dedicated core and 16 GB memory 980 NGINX Docker containers. Results may vary. Operating system was SLES12 SP4 (SMT mode). Docker 18.09.6 and NGINX 1.15.9 was used.

# Container Scale-out Performance

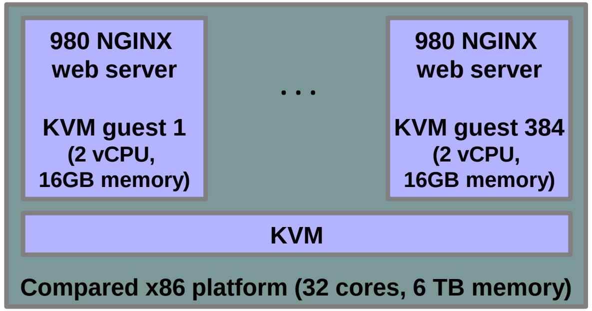
## Scale-out under KVM on LinuxONE III versus x86 Skylake

**Run up to 6.6x more Docker containers under KVM on a LinuxONE III system versus a compared x86 platform**

**DISCLAIMER:** Performance result is extrapolated from IBM internal tests running 980 NGINX Docker containers in a LinuxONE III LPAR and bare-metal on a x86 server. LinuxONE III measurement configuration: LPAR with 1 dedicated core, 16 GB memory, running SLES 12 SP4 (SMT mode), Docker 18.09.6, NGINX 1.15.9. x86 measurement configuration: 1 Intel® Xeon® Gold 6126 CPU @ 2.60 GHz with Hyperthreading turned on, 16 GB memory, running SLES 12 SP4, Docker 18.09.6, NGINX 1.15.9. Based on the measurement results it is extrapolated that a LinuxONE III server with 190 cores and 40 TB memory can run 2.469 million NGINX Docker containers if configured with 20 LPARs, each having 9 cores, 2 TB memory, and running a KVM 2.11.2 instance with 126 KVM guests, each configured with 2 vCPUs, 16 GB memory, and running 980 dockerized NGINX web server. Based on the measurement results it is extrapolated that a x86 server with 8 Intel® Xeon® Platinum 8156 processors (32 cores in total) and 6 TB memory can run 376 thousand NGINX Docker containers if configured with KVM 2.11.2 with 384 KVM guests, each configured with 2 vCPUs, 16 GB memory, and running 980 dockerized NGINX web server. Results may vary.

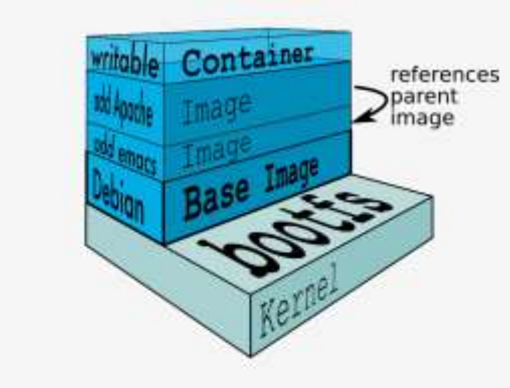



**2.4 million Docker container on LinuxONE III w/ 40 TB memory**
**versus**
**376 thousand Docker container on a x86 server w/ 6 TB memory**

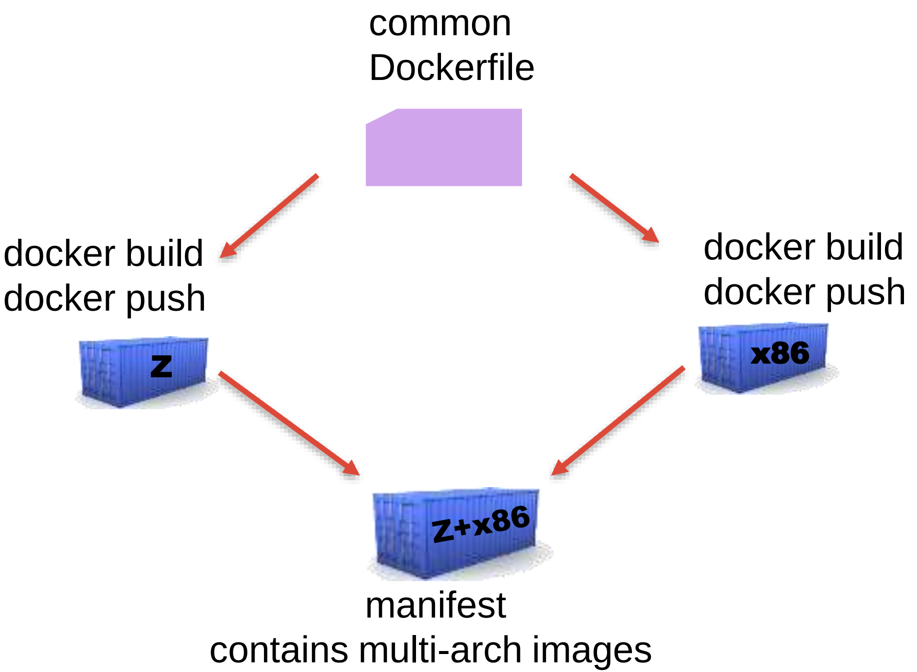


# Portability of Container & Multi Architecture support

- **Container user experience (CLI, REST API) is identical across platforms**
- **Container images are not portable**, the source code or a s390x binary must be build and available
- Micro-service architectures often have clean structure and simple individual components
- Containers are often created with Dockerfiles (build descriptions) containing:
  - Specification of the base image
    - If the same distribution is available on s390x, usually simple
    - If the base image is not available, some creativity is required
  - Additional steps to modify the image are often platform independent
    - Add packages (needs to match the base image)
    - Download files, Perform build
- Same Dockerfile can be used for multi-platform builds
- Multi-arch Registry support available using external tools ( i.e. manifest tool )
  - [http://containerz.blogspot.com.br/2016/07/multi-arch-registry.html](http://containerz.blogspot.com.br/2016/07/multi-arch-registry.html)



# Manifest tool - creates Multi – Architecture Container Images




```
image: webapp:latest
manifests:
-
image: webapp-s390x
platform:
architecture: s390x
os: linux
-
image: webapp-amd64
platform:
architecture: amd64
```

**Container images on Docker Hub are multi-arch**

- lots of images as s390x versions available

# Container Ecosystem evolved for IBM Z

| Container image | Container runtimes & orchestration | Container tools |
| --- | --- | --- |
| ubuntu, alpine, debian, fedora, OpenJDK, MariaDB FOUNDATION, NGINX, php | docker, cri-o (OCI-based Container engine), containerd, docker SWARM, Kubernetes, runC runtime, runQ runtime, kata | elasticsearch, logstash, Kibana, CRIU, notary, LINKERD, buildah, portainer.io, LinuxKIT |

Multi-arch support

# The OCI Initiative

https://www.opencontainers.org/

The Open Container Initiative (OCI) is a lightweight, open governance structure (project), formed under the auspices of the Linux Foundation, for the express purpose of **creating open industry standards around container formats and runtime**.

The **OCI was launched on June 22nd 2015** by Docker, CoreOS and other leaders in the container industry

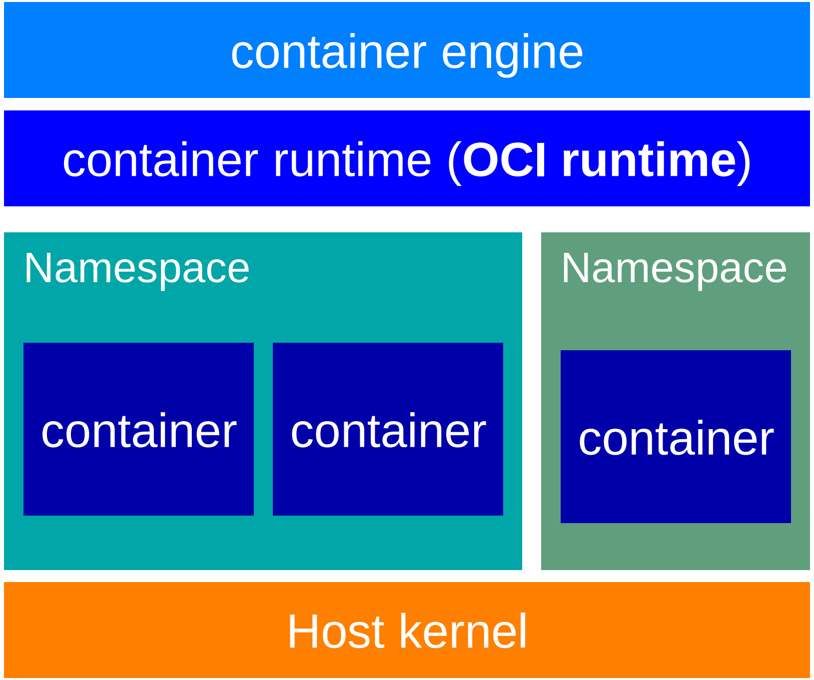
Two specifications:

**Image Specification** : define an OCI Image then it will be unpacked into an OCI Runtime filesystem bundle

**Runtime Specification**: how to run a "filesystem bundle" that is unpacked on disk.

**'runc'** implements the runtime specification

# Container components




A **Container engine** manages the container lifecycle

- Create the container image
- Run the container

**Different container engines exist**,
e.g. Docker, CRI-O, containerd, podman...

A **container Runtime** is a lower level component, typically used in a Container Engine

**runc** is one of the most used container runtimes

Other runtimes beside ‘runc’, e.g. ‘runq’ and ‘Kata’

# Container engines diversity

- A stable, core, performant core container runtime for the cloud
- Has a CRI implementation, and is a CNCF graduated project

- “all the runtime Kubernetes needs and nothing more”; RH created CRI implementation over runc and 2 open libraries; K8s incubator

- Intel Clear Containers + Hyper.sh combined project
- Lightweight virtualization (KVM/qemu) under cri-o and containerd

- Amazon open source project announced Nov 2018; lightweight virt.
- Uses Rust-based VMM instead of qemu; plugs into containerd

- CRI implementation over Sylabs Singularity runtime project
- Userbase traditionally from academia/HPC use cases

# Container in Red Hat & OpenShift

## Red Hat headed towards a world without any Docker

- **Cri-o** is only one component (the Kubernetes CRI runtime) of OpenShift
- RHEL will not deliver a modern Docker engine; Red Hat will replace it with:
  - **podman** (docker client clone); **skopeo** (registry); **buildah** (docker build..)

## Red Hat dedicated customers will not have Docker

# Project Atomic – new Container ecosystem

Available in **Fedora Atomic Host, CentOS Atomic Host, and Red Hat Atomic Host** editions depending on your platform and support needs.

**Container Tools**
Tools to create, deploy, manage, and secure containers.

- Build containers with Buildah
- Compose applications with Kompose
- Pull and move images with Skopeo
- Deploy using Kubernetes and CRI-O
- Manage container hosts with Cockpit
- Install, run and scan with Atomic CLI
- Sandbox apps with Bubblewrap

**Container Registries**
Get your containerized applications from the **CentOS Container Pipeline** and the **Fedora Layered Image Build Service**

- Trusted container content from the projects you already trust
- Learn more about Fedora Layered Images
- Learn more about CentOS Container

https://www.projectatomic.io/

# Alternative container engine

Red Hat provides an alternative container ecosystem tooling with cri-o

Available from RHEL 8 and Fedora 29:

- Container deployment: **podman**
- Container building: **buildah**
- Container Registry: **Quay**
- Manage container images and registry: **skopeo**
- Cri-o: container engine used in Red Hat Openshift V4

# Container build: Docker vs Podman

> docker build –f Dockerfile .

**Daemon**: all operations manages by a single deamon. Single point of failure.

**Root privileges**: all Docker operations have to be conducted by a user (or users) with the same full root authority

**Networking**: CNR and CNI plugins support

**User friendly**: straightforward to use and a lot of examples, documentation and tooling available

> podman build –f Dockerfile .

**Daemon less**: a podman instance pro container

**Run container rootless**: user without root privileges can start containers

**Networking**: only CNI plugin support

**User friendly**: goal to offer the same user experience as docker. Less documentation and not all the flags available for docker are available in podman

These tools are all building OCI compliant container imagers and can be used with different container runtimes.

You may wish to keep Docker around while you try out Podman.
There are some useful tutorials and an awesome demonstration available.

# Deployment: Podman vs. Docker

https://developers.redhat.com/blog/2019/02/21/podman-and-buildah-for-docker-users/

The claim is made:

- **if you have existing scripts that run Docker you can create a docker alias for podman and all your scripts should work**
  (alias docker=podman)

When you first type

‘**podman images**’ - you might be surprised that you don’t see any of the Docker images you’ve already pulled down – running it as user vs as root.

**Podman’s local repository is in /var/lib/containers instead of** /var/lib/docker

This isn’t an arbitrary change; this new storage structure is based on the Open Containers Initiative (OCI) standards.

https://github.com/containers/libpod/blob/master/docs/tutorials/podman_tutorial.md

Setup Podman / Buildah:

There are a few things to unpack:

- You install Podman instead of Docker. You do not need to start or manage a daemon process like the Docker daemon.
- The commands you are familiar with in Docker work the same for Podman.
- Podman stores its containers and images in a different place than Docker.
- Podman and Docker images are compatible.
- Podman does more than Docker for Kubernetes environments.

What is buildah and why might I need it?

- *Buildah* can be described as a *superset of podman* commands related to creating and managing container images and it has much finer-grained control over images.
- Dynamic mounts i.e. secrets, volumes, can only be made with buildah

https://github.com/containers/buildah/tree/master/docs/tutorials

You may wish to keep Docker around while you try out Podman.
There are some useful tutorials and an awesome demonstration available.

# Availability of Container Tools on IBM Z

**Docker** is available and supported in

Ubuntu 16.04 and later

RHEL 7.5 - 7.7 via *extras* repository

SLES 15 – SLES 15 SP1

**Podman** is available and supported in

RHEL 7.5 and later

SLES15 SP1

*Docker* as community edition:

Ubuntu 16.04 and later

Fedora 28 and later

Red Hat

| | docker | podman |
|---|---|---|
| RHEL 7.5 | 1.13 | 0.9.2 |
| RHEL 7.6 | 1.13 | 1.4.4 |
| RHEL 7.7 | 1.13 | 1.4.4 |
| RHEL 8 | - | 1.0.0.2 |
| RHEL 8.1 | - | 1.4.2 |

SUSE

| | docker | podman |
|---|---|---|
| SLES15 | 17.09 | - |
| SLES15 SP1 | 18.09.1 | 1.0.1 |

Ubuntu

| | docker | podman |
|---|---|---|
| 16.04 LTS | 18.09.7 | - |
| 18.04 LTS | 18.09.7 | - |

# Agenda

- ➢ **Container technologies and Ecosystem**

- ➢ **Container Orchestration**

Kubernetes

# Container Orchestration

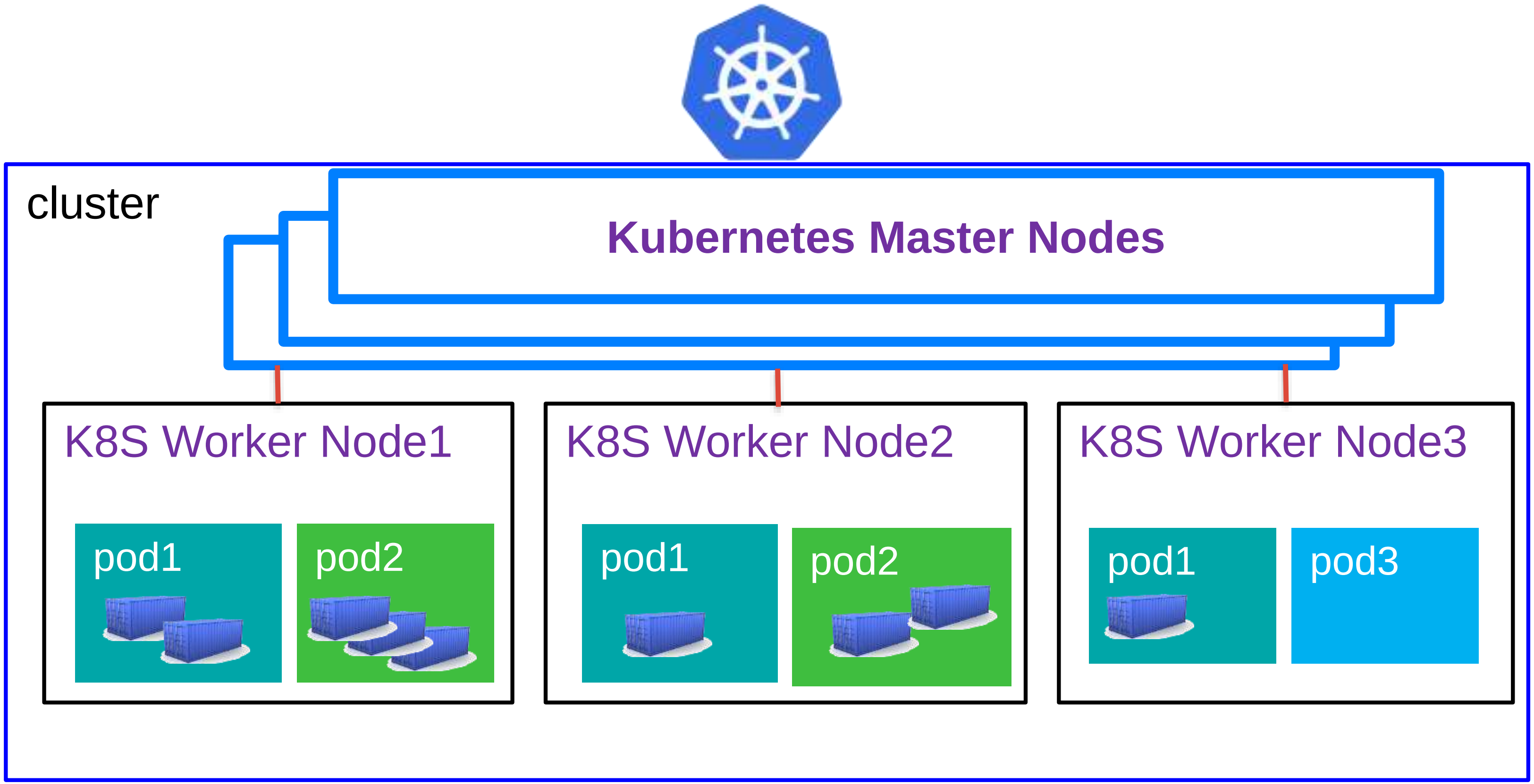
- Kubernetes and Docker Swarm build the base ecosystem
  - Based on identical source code
  - IBM Z binaries are built as part of the release process

- Kubernetes and Docker Swarm mixed architecture development and deployment

- Docker Hub Content (images) valid for both orchestrators
- Both products run on Linux on Z

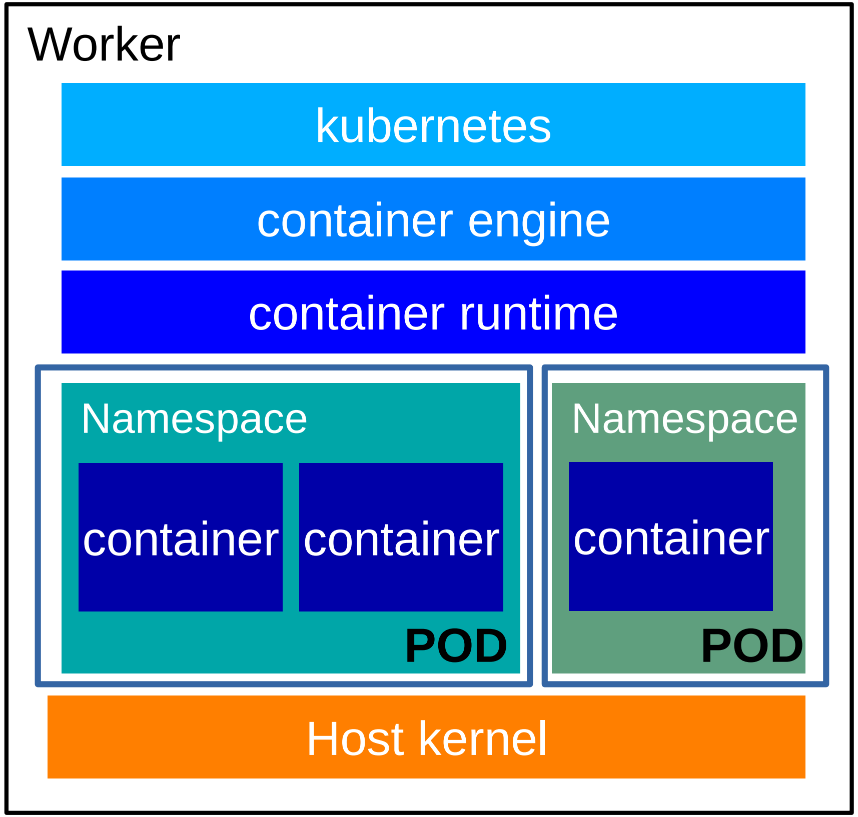
Kuernetes

# Kubernetes (K8S) – defines itself in a cluster format



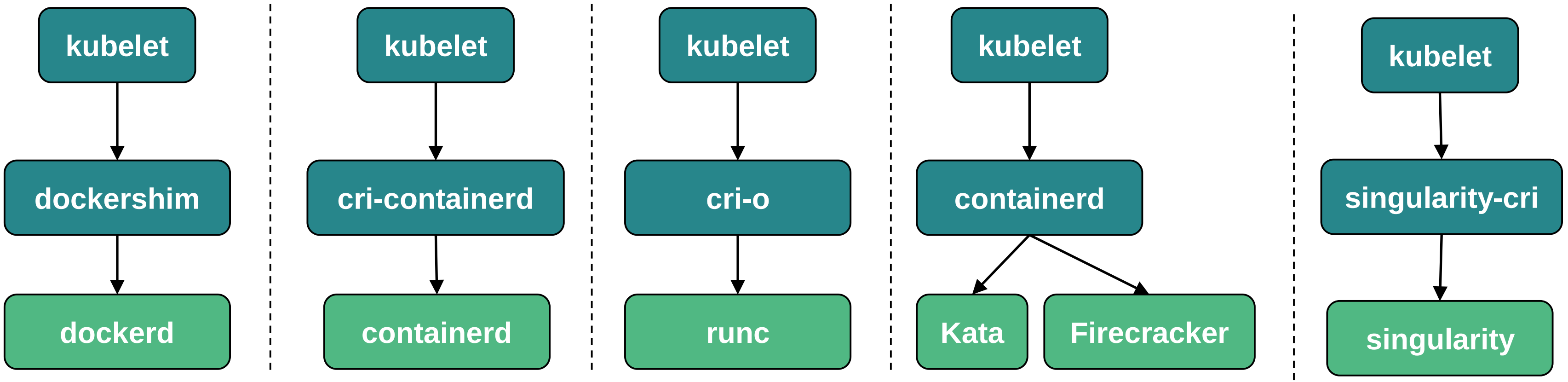

**Kubernetes is not running container – it orchestrates them**

# Kubernetes (k8s) - worker node architecture



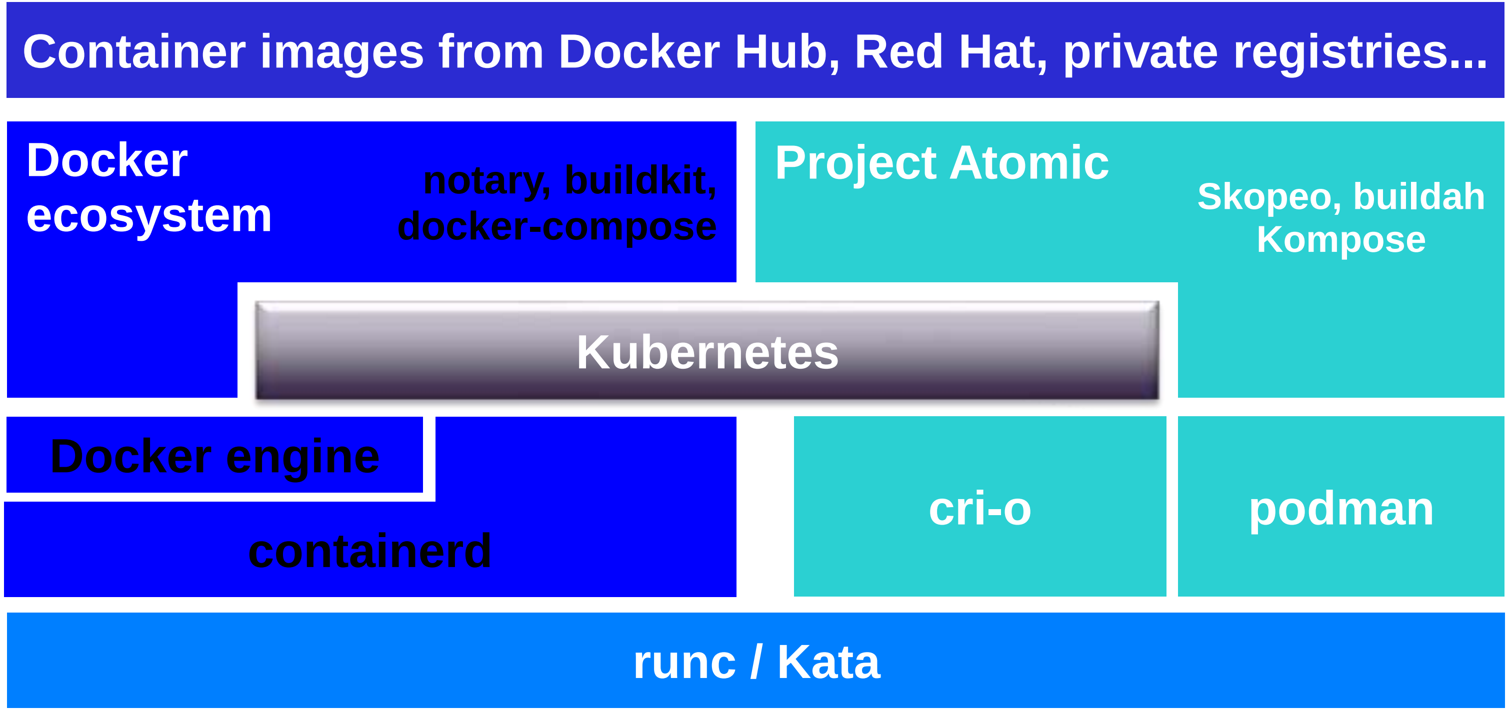

- **Kubernetes** is a container **orchestration** tool for automation, application deployment, scalability and container management
- It groups containers in a unit called **Pod**
- It deploys container using a container engine
- The *kubelet* is the primary “node agent” that runs on each node
- Easy to extend through its API
- Huge ecosystem around Kubernetes API

# Diversity of CRI Runtimes to Kubernetes today




**kubelet --container-runtime {string}**
**--container-runtime-endpoint {string}**

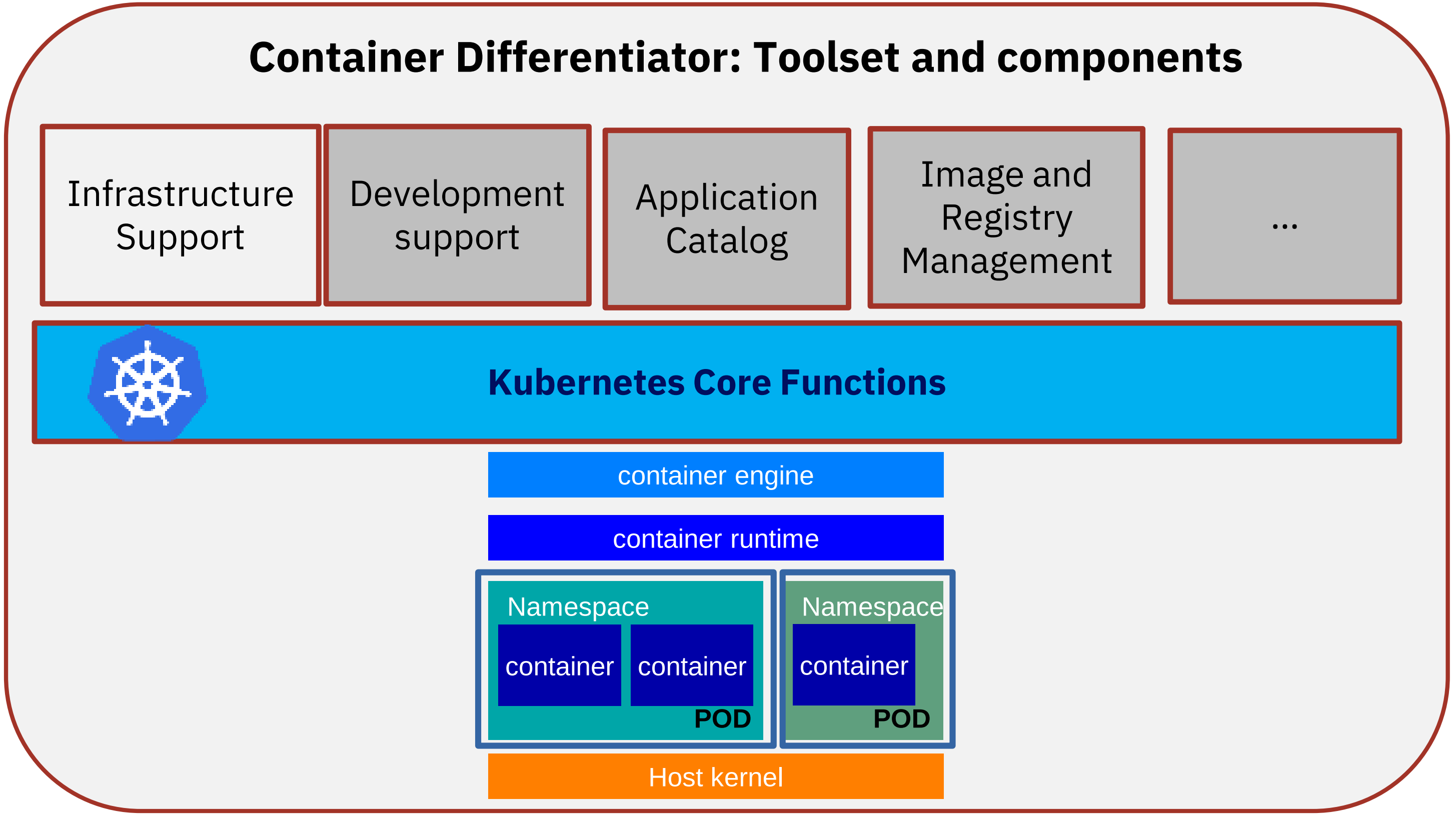
# Container orchestration ecosystem with K8S




[1] https://www.docker.com/why-docker
[2] https://www.projectatomic.io/

# Kubernetes APIs are used in all Orchestration products

## (i.e. OpenShift, Cloud Foundry, IBM Cloud Private)

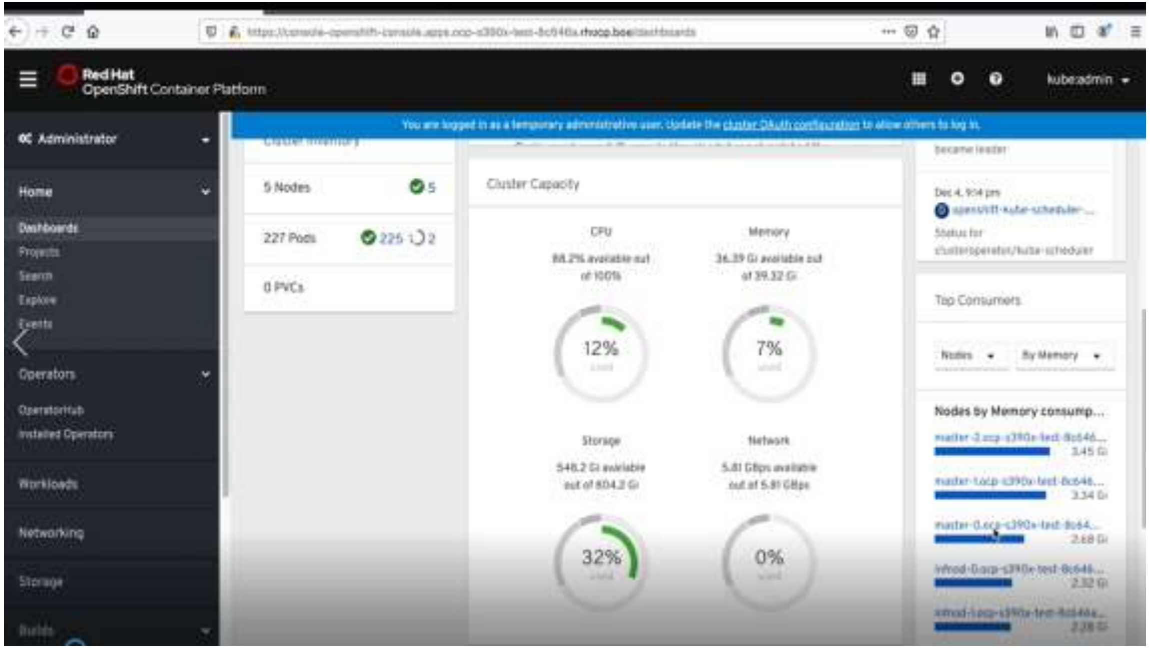


# Red Hat OpenShift Container Platform available for Linux on Z & LinuxONE

New

- OpenShift brings together the core open source technologies of Linux, containers and Kubernetes.

**Available:**
**Red Hat OpenShift V4 for IBM Z and LinuxONE**
Announced by Ross Mauri
Feb 13, 2020

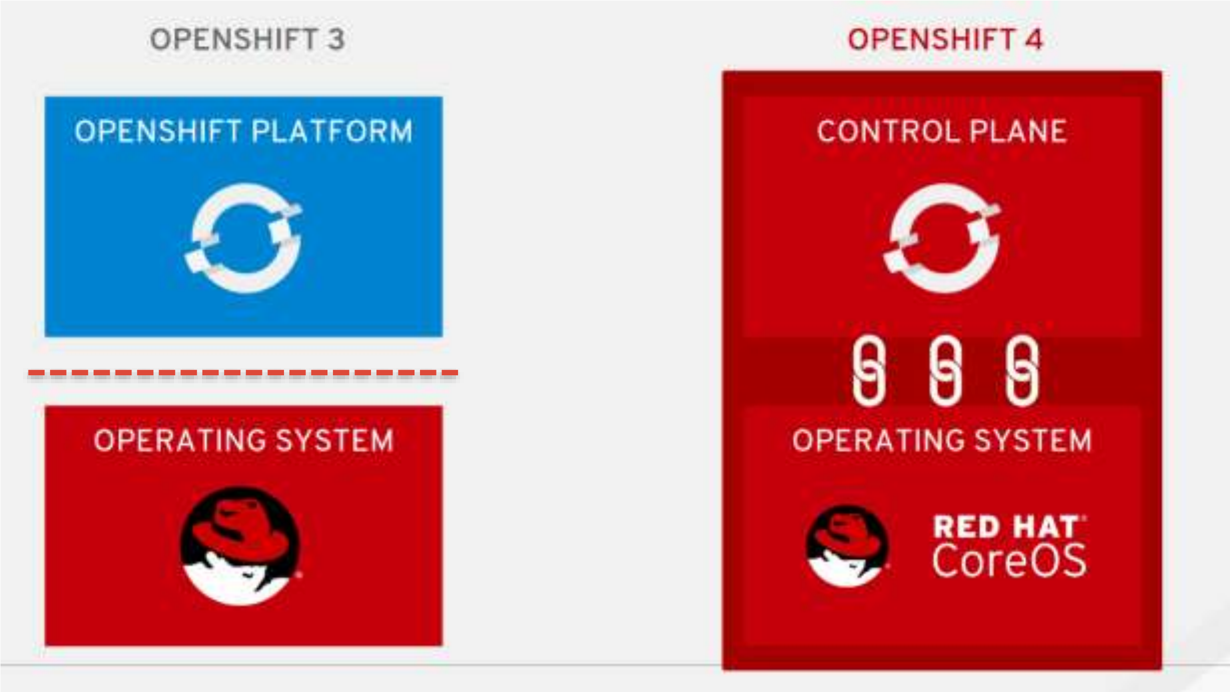



http://www.ibm.com/blogs/systems/red-hat-openshift-now-available-ibm-z-linuxone

https://developer.ibm.com/blogs/willie-tejada-redhat-openshift-ibmz/

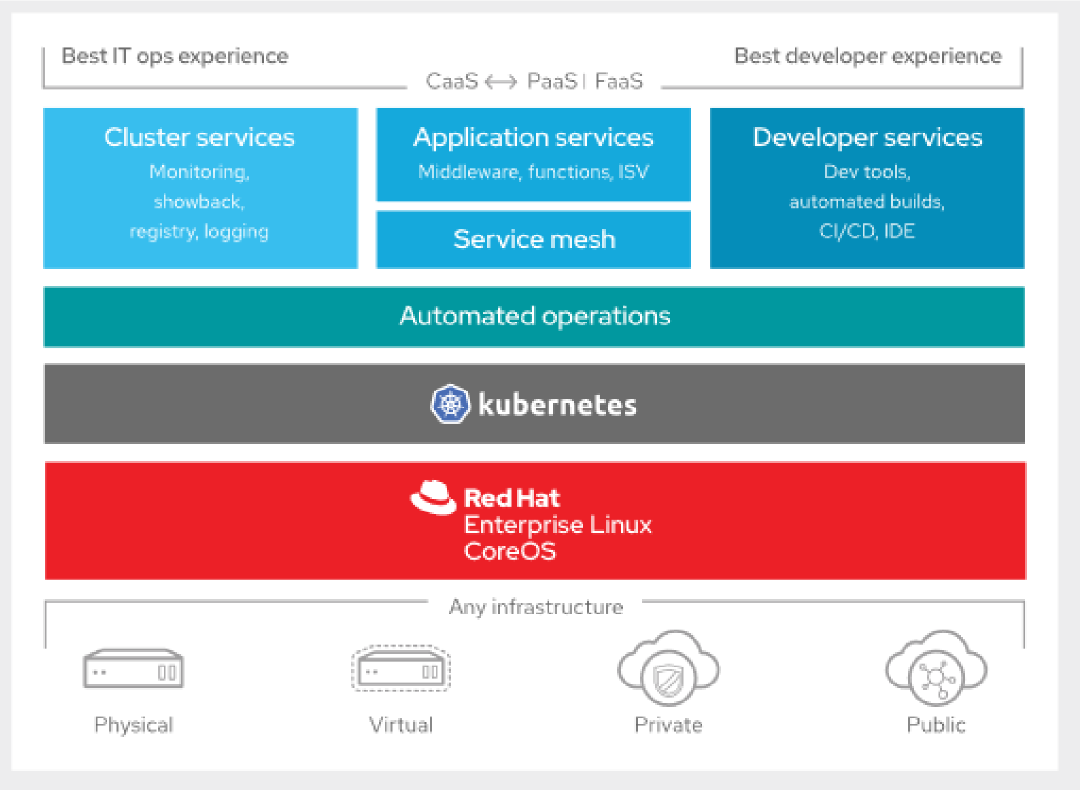
# Red Hat OpenShift - Immutable Infrastructure

Immutability = repeatability
Immutability = auditability




https://blog.openshift.com/wp-content/uploads/Red-Hat-OpenShift-4.0-Roadmap-Public-Feb-2019-Ali.pdf

# Red Hat OpenShift V4




https://www.redhat.com/cms/managed-files/cl-openshift-4-datasheet-f16726wg-201905-en.pdf

# Red Hat OpenShift V4

OpenShift is a layered system designed to expose Container images and Kubernetes concepts, with a focus on easy composition of applications by a developer.

https://docs.openshift.com/container-platform/4.1/architecture/architecture.html

**What Are the Layers?**

- The Container service provides the abstraction for creating container images.
- Kubernetes provides the cluster management and orchestrates containers
  - **Container Runtime Interface** (CRI) - how K8S talks with a container engine
  - **Container engines -** implement the CRI interface (OCI compliant)

**OpenShift Container Platform adds:**

- Source code management, builds, and deployments for developers
- Managing and promoting images at scale as they flow through your system
- Application management at scale
- Team and user tracking for organizing a large developer organization
- Networking infrastructure that supports the cluster

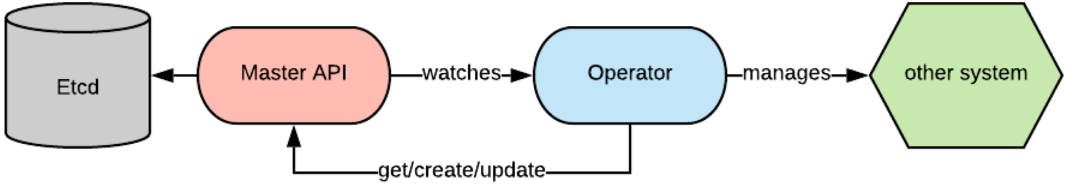
# OpenShift V4 characteristics

- RHEL CoreOS is the immutable container host version of Red Hat Enterprise Linux and features a RHEL kernel with SELinux enabled by default.
- It includes the kubelet, which is the Kubernetes node agent, and the CRI-O container runtime, which is optimized for Kubernetes.
- Every control plane machine in an OpenShift 4 cluster must run RHEL CoreOS.
- It includes a critical first-boot provisioning tool called CoreOS Ignition hat enables the cluster to configure the machines.
- Operating system updates are delivered as an Atomic OSTree repo embedded within a container image which is rolled out across the cluster via an operator.
- Actual operating system changes are made in-place on each machine as an atomic operation using rpm-ostree.

# Red Hat OpenShift V4 operators

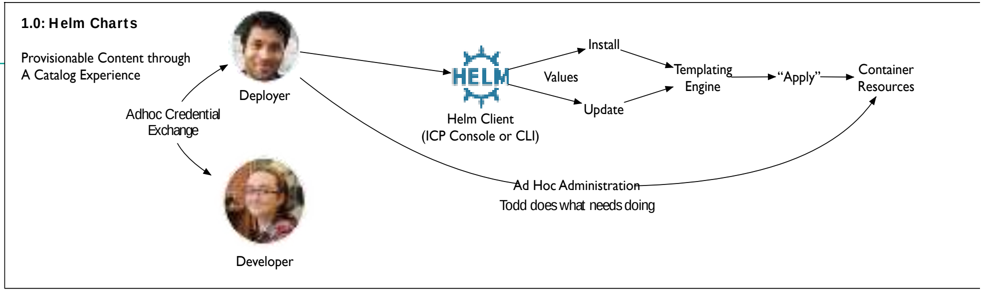
OpenShift uses operators to manage every aspect of the cluster. They enhance k8S Helm.

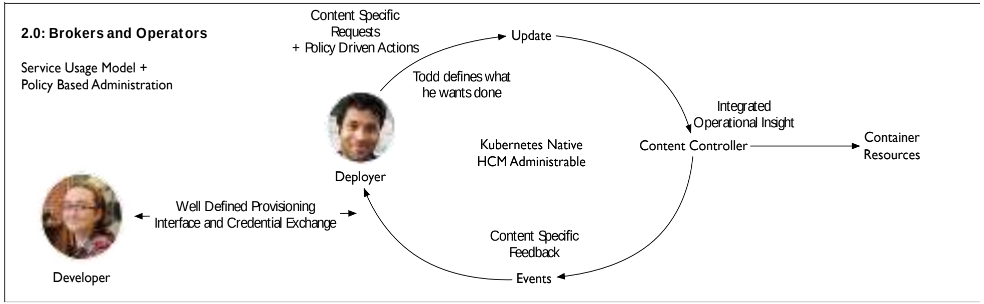
- includes operators that manage components like the api server, scheduler, and controller manager.
- Additional operators for components like the cluster-autoscaler, cluster-monitoring, web console, dns, ingress, networking, node-tuning, and authentication are included to provide management of the entire platform.
- Each operator exposes a custom resource API interface to define the desired state, observe the status of their rollout, and diagnose any issues that may occur.
- The operator lifecycle manager component is included to provide a framework for administrators to manage additional Kubernetes-native applications that are optionally integrated into the cluster as a day-2 task.
- Operators are implemented as a collection of controllers where each controller watches a specific resource type.



# Operators vs Helm

- **Helm** covers three stages of the content lifecycle:
  1. Initial deployment
  2. Upgrades
  3. Deletion
- **Operators** cover end-to-end lifecycle
  - Continuous monitoring of related resources
  - **Ability to invoke/trigger day-2 operations**
  - **Ability to model dependencies between operators and trigger actions accordingly**
  - Operators can lean on Helm charts for initial provisioning etc
- Operators are standard Kubernetes elements




**Operators complement and enhance Helm!**

# Red Hat OpenShift Deployment options

Red Hat OpenShift 4 (OCP) aims to deliver the automation experience of a native public cloud container platform while retaining the flexibility of a multi-cloud, enterprise-class solution.

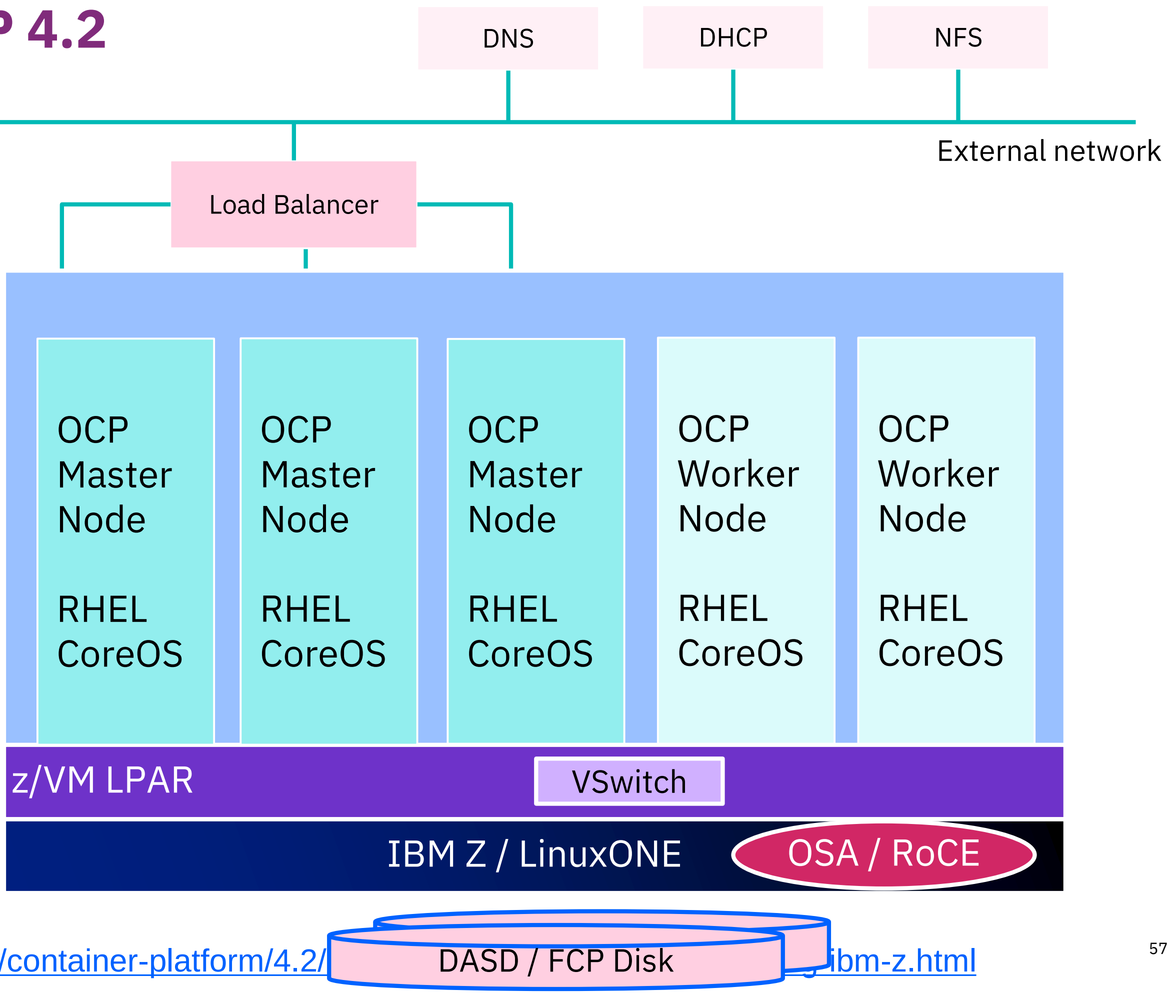
- **Installer Provisioned Infrastructure (IPI)**

On supported platforms, the installer is capable to provision the underlying infrastructure for the cluster.
Via the installer create all components, networking, machines, and operating systems for the cluster.

- **User Provisioned Infrastructure (UPI)**

For platforms or in scenarios where installer provisioned infrastructure would be incompatible, the platform administrator has to provision the infrastructure using the cluster assets generated by the install tool.
Once the infrastructure has been created, OpenShift 4 is installed, maintaining its ability to support automated operations and over-the-air platform updates.

# Minimum operational OCP 4.2 cluster on z/VM Layout



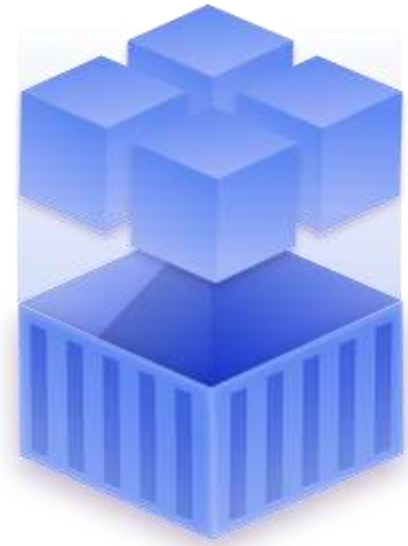

**Characteristics:**

- UPI deployment with z/VM
- Based on CoreOS only
- Automatic through deployment server for installation (temporary, i.e. z/VM guest)
- UBI – Universal Base Image, the signed base image to containerize aplications for OCP

https://docs.openshift.com/container-platform/4.2/ ... ibm-z.html

# IBM Cloud Paks – IBM Software in Container

*Enterprise-grade, modular middleware solutions giving clients an open, faster, more reliable way to move, build, and manage on the cloud*



Pre-integrated for cloud use cases

**IBM Certified Containers**
Containerized, security-compliant IBM middleware and Open Source components

**Common operational services**
Logging, monitoring, metering, persistent storage, security, identity access management, Docker registry/Helm

**Container platform**
Kubernetes-based and portable

Red Hat OpenShift

IBM Cloud | RED HAT OPENSHIFT | amazon webservices | Azure | openstack | Google Cloud

[https://www.ibm.com/cloud/paks/](https://www.ibm.com/cloud/paks/)

# Cloud Pak for Applications v4

*Migrate to cloud at your own rate and pace with flexible licensing to mix and match your entitlements over time*

**Run existing apps**

Protects customer investments: Continue to run existing apps – WebSphere, JBoss

**Modernize existing apps**

Refactor existing applications to containers on OpenShift

**Build new apps**

Develop and deploy new cloud native applications

# Cloud Paks – Pre-integrated for cloud use cases

*Today, IBM offers clients the first six Cloud Paks…*

## Cloud Pak for Applications

Developer & DevOps Tools
Modernization Toolkit
Frameworks and Runtimes

Build, deploy, and run applications

Red Hat OpenShift

## Cloud Pak for Data

Organize
Analyze
Collect

Collect, organize, and analyze data

Red Hat OpenShift

## Cloud Pak for Integration

API Lifecycle
Messaging and Events
App and Data Integration

Integrate applications, data, cloud services, and APIs

Red Hat OpenShift

## Cloud Pak for Automation

Content
Operational Intelligence
Workflow and Decisions

Transform business processes, decisions, and content

Red Hat OpenShift

## Cloud Pak for Multicloud Management

Multicluster
App and Infrastructure
Security and Compliance Management

Multicloud visibility, governance, and automation

Red Hat OpenShift

## Cloud Pak for Security

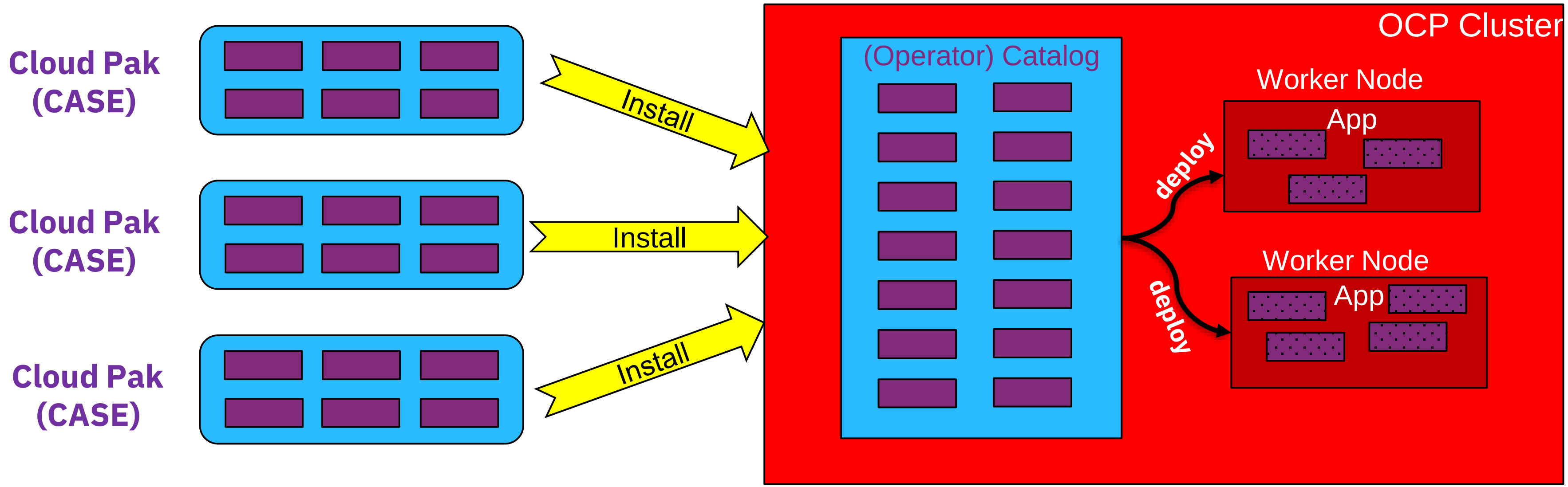
Federated Search and Investigation
Incident Response
Security Orchestration and Automation

Connect security data, tools, and teams

Red Hat OpenShift

IBM Cloud | aws | Azure | Google Cloud | Edge | Private | Systems

# From silos to an integrated set of capabilities




- Individual capabilities are added to a common catalog (Operator / Helm)
- Easily deployable from catalog to support applications
- Using dynamic service bindings

# IBM Cloud Pak for Application – Pre-integrated for cloud use cases

## Available NOW !

New

**IBM Cloud Pak for Applications V4.0**
**for IBM Z and LinuxONE delivers**
**Red Hat OpenShift Container Platform 4.2**

https://www-01.ibm.com/common/ssi/ShowDoc.wss?docURL=/common/ssi/rep_ca/4/897/ENUS219-574/index.html&request_locale=en

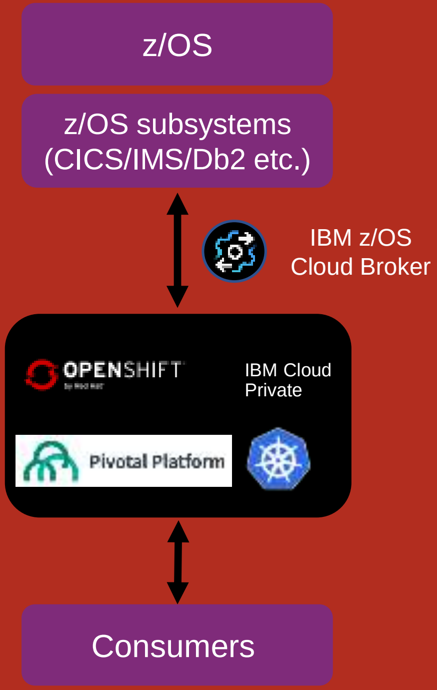
# IBM z/OS Cloud Broker

- **Connects z/OS services running on an IBM Z backend to a frontend private cloud platform** providing self-service access and consumption of these services to developers

Provides self-service access to managed IBM Z resources to all flavors of application developers

Centralization and automation of IBM Z operations to provide Z resources to agencies or clients in their hybrid cloud

Improve time to value through efficiencies in development and deployment




**Support for OpenShift Platform GA: 4Q 2019**

https://www.ibm.com/support/z-content-solutions/cloud-broker/

# Summary: Why Containers work -- Separation of Concerns

**Dan the Developer**

- **Worries about what's "inside" the container**
  - **His code**
  - **His Libraries**
  - **His Package Manager**
  - **His Apps**
  - **His Data**
- **All Linux servers look the same**

Oscar the Ops Guy

- Worries about what's "outside" the container
- Logging
- Remote access
- Monitoring
- Network config
- All containers start, stop, copy, attach, migrate, etc. the same way

## Build once…(finally) run anywhere*

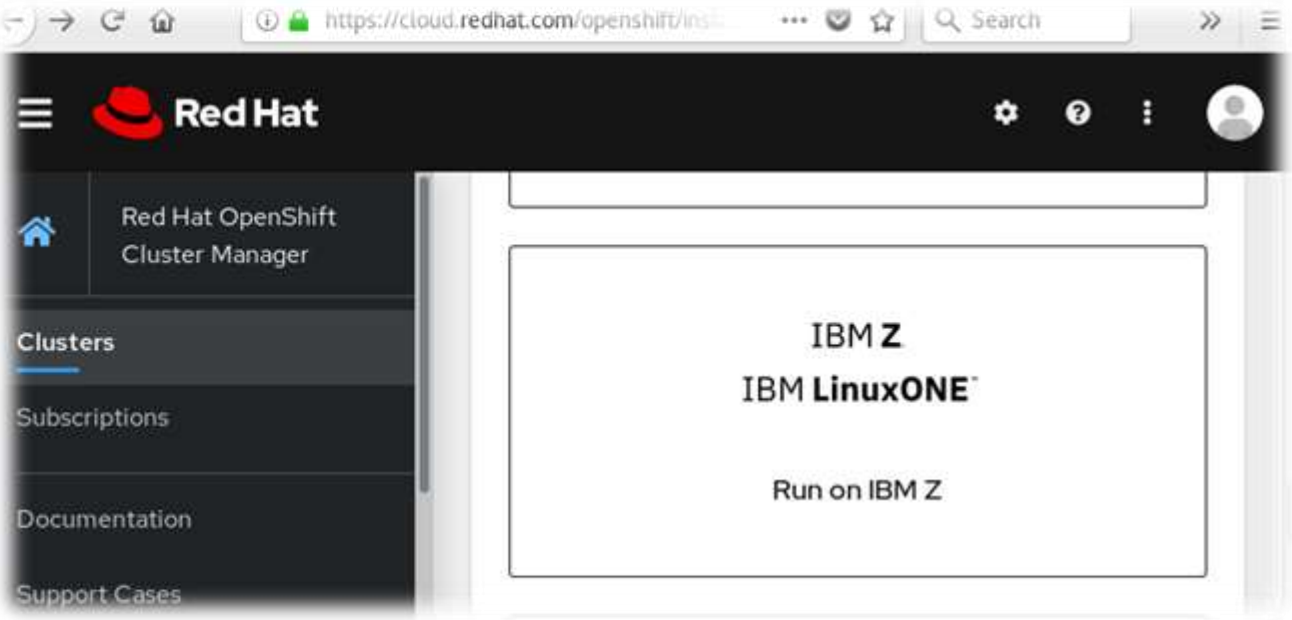
- A clean, safe, hygienic and portable runtime environment for your app.
- No worries about missing dependencies, packages and other pain points during subsequent deployments.
- Run each app in its own isolated container, so you can run various versions of libraries and other dependencies for each app without worrying
- Automate testing, integration, packaging…anything you can script
- Reduce/eliminate concerns about compatibility on different platforms, either your own or your customers.
- Cheap, zero-penalty containers to deploy services? A VM without the overhead of a VM? Instant replay and reset of image snapshots? That's the power of Docker

## Configure once…run anything

- Make the entire lifecycle more efficient, consistent, and repeatable
- Increase the quality of code produced by developers.
- Eliminate inconsistencies between development, test, production, and customer environments
- Support segregation of duties
- Significantly improves the speed and reliability of continuous deployment and continuous integration systems
- Because the containers are so lightweight, address significant performance, costs, deployment, and portability issues normally associated with VMs

# Where can I download OCP V4 for IBM Z ?

**try.openshift.com**
**cloud.redhat.com**



# Open-source containerized Software for Linux on IBM Z & IBM LinuxONE

https://www.ibm.com/developerworks/community/forums/html/topic?id=5dee144a-7c64-4bfe-884f-751d6308dbdf

The table provides up-to-date information on open source packages that have been ported and/or validated on corresponding distro versions by IBM.

| Package | Ubuntu 19/18.x | SLES 15.x | RHEL 7.x | Dockerfile/image | SLES 12.x |
|---|---|---|---|---|---|
| Akka | Latest | Latest | Latest | NA | Latest |
| Alfresco | 5.x | 5.x | 5.x | 5.x, image | 5.x |
| Anaconda | NA | NA | 4.x | NA | 4.x |
| Ansible | Distro, Latest | Latest | Latest | Latest, image | Latest |
| AntLR | Distro, 3.x, 4.x | NA | 3.x, 4.x | 4.x, image | 3.x, 4.x |
| Apache ActiveMQ | Distro, Latest | Latest | Latest | 5.x, image | Latest |
| Apache Camel | Latest | Latest | Latest | NA | Latest |
| Apache Cassandra | 2.x, 3.x | 3.x | 2.x, 3.x | 3.x, image | 2.x, 3.x |
| Apache Flume | 1.x | 1.x | 1.x | 1.x, image | 1.x |
| Apache Geode | Latest | Latest | Latest | 1.x, image | Latest |
| Apache HTTP | Distro, 2.4 | Distro,2.4 | Distro,2.4 | 2.x, Image | Distro,2.4 |
| Apache Ignite | Latest | Latest | Latest | 2.x, Image | Latest |
| Apache JMeter | Distro, Latest | Latest | Latest | 5.x, image | Latest |
| Apache Kafka | Latest | Latest | Latest | 2.x, image | Latest |
| Apache Maven | NA | NA | download | 3.x, Image | download |
| Apache Mesos | 1.x | 1.x | 1.x | 1.x, image | 1.x |
| Apache Spark | 2.x | 2.x | 2.x | 2.x, image | 2.x |
| Apache Solr | 9.x | 8.x | 8.x | 8.x, Image | 8.x |
| Apache Storm | 2.x | 2.x | 2.x | 2.x, Image | 2.x |
| Apache Tomcat | Distro, Latest | Distro,Latest | Distro,Latest | 9.x, Image | Distro,Latest |
| Apache Zeppelin | 0.8.x | 0.8.x | 0.8.x | 0.8.x, image | 0.8.x |
| Apache ZooKeeper | Distro, Latest | Latest | Latest | 3.x, Image | Latest |
| Apigility | 1.5.x | 1.5.x | 1.5.x | 1.5.x, image | 1.5.x |
| Beats | NA | NA | 7.x | 7.x, Image | 7.x |
| BIRT | NA | NA | NA | NA | NA |

# Questions?

***Wilhelm Mild***

*IBM Executive IT Architect*

The Open Group Open Certified
Distinguished Architect
IT Architecture
Chief/Lead IT Architect

IBM

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

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

# Notices and disclaimers

- 
- **U.S. Government Users Restricted Rights — use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.**
- Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed "as is" without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.
- IBM products are manufactured from new parts or new and used parts.
  In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply."
- **Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.**
- Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those
- customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.
- References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.
- Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.
- It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

# Notices and disclaimers continued

- Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of 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. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**
- The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.
- IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml