**Kubernetes and Read Hat OpenShift**

**Why Linux?**

The integration between Red Hat technology and Cloud Pak for Data is not accidental. Some elements used in cloud computing already existed in Linux before containerization became a big deal. Linux isolates processes and uses namespaces (a function which allows to isolate resources). Linux kernel introduced control groups (limiting resources for a container) in 2008.

The illustration below shows, in a simplified way, layers of technologies used to run Cloud Pak for Data in its current form. It starts with the already mentioned control groups to show that the concept of containers did not come from nowhere (actually the concept of containerization is decades old). In order to understand the distinction between containers, Kubernetes and Red Had OpenShift, examine the picture below or read the text underneath. The whole idea lies in isolating applications – the solution provided by the Linux kernel was used to create containers. When the number of containers grows, things such as managing communication between services or setting resource limits become too difficult to be managed manually and that is where Kubernetes applies. On top of Kubernetes, Red Hat OpenShift provides enterprise-ready solutions such as remote management, monitoring and auditing.

At first Linux introduced control groups which enable the isolation of resources. Then Docker Engine accelerated the adoption of containerization. As managing many containers is challenging, manual work required for maintaining containers is replaced by Kubernetes to allow high availability. At the end, Red Hat OpenShift provides the enterprise-level Kubernetes container platform to manage hybrid cloud and multicloud deployments.11

## Containerization

A **container**is an **executable package of software** that includes everything needed to run it. Red Hat OpenShift Container Platform makes use of containers.

Think of a container as a fully packaged application with everything you need to run the application, such as the required libraries and binaries which you can ship anywhere, and run it anytime. The concept of containers is similar to that of shipping containers. Everything that is necessary for the application to run resides within the "shipping container". For example, when the application needs to run on Windows, it can be shipped there. When it needs to run on Linux, it can be shipped there.

Let's take the concept of a container, as an application with all the things necessary to run that application, and combine it with the concept of **microservices**.

You can think of a microservice as a simple entity, performing a single function, or a service. Different microservices can connect to the single user interface.

Consider a microservice developed inside a container. This container performs a single function. In keeping up with the analogy of shipping containers where each container is now a microservice.

Combine many of microservices together and now you have a ship full of containers. The goal of application design using this framework is that each service can operate **independently**of one another. Each container, or service, can be updated independently without any disruption to the entire application and platform.

## What is Kubernetes?

Containers are great, but imagine having hundreds or thousands of them working together. Imagine the dependencies between them and how you would need to manage everything. **Kubernetes**, another open source tool, helps with the management and control of all these containers. Kubernetes is used by the Red Hat for OpenShift Container Platform to organize and govern these containers, and it offers a lot of features.

A very important concept in Kubernetes is operators. An operator is an application that manages other applications and does not have user functionality. It adds the important functional value to the Red Hat OpenShift Container Platform. With operators you can deploy applications and their components, without the need for manual upgrades of operating system and control plane applications. They enable simplified, cluster-wide management of critical components.

 Later in this learning path you will learn how to use operators to install Cloud Pak for Data.

**Intelligent scheduling**

Intelligent scheduling automatically places containers into the scheduling queue for resource consumption based on their resource requirements and other constraints, while not sacrificing their availability. Mixing critical workloads requires dedicated resources with best-effort workloads that operate under minimal resource requirements in order to increase utilization of nodes and save platform resources.

**Self-healing**

Self-healing restarts containers that fail, replaces and reschedules containers when nodes crash, deletes containers that don't respond to your user-defined health check, and does not advertise them to clients until they are ready to serve.

**Horizontal scaling**

Scale your application up and down with a simple command, with a UI, or automatically based on CPU usage.

**Service discovery and load balancing**

There is no need to modify your application to use an unfamiliar service discovery mechanism. Kubernetes gives containers their own IP addresses and a single DNS name for a set of containers, and can load-balance across them.

**Automated rollout and rollback**

Kubernetes progressively rolls out changes to your application, while monitoring application health to ensure that it does not kill all of your instances at the same time. If something goes wrong, Kubernetes will rollback the change for you.

**Secret configuration and management**

Deploy and update secrets (objects that are intended to hold sensitive information, such as passwords, tokens, and SSH keys) and application configuration without rebuilding your image and without exposing secrets in your stack configuration.

## Red Hat OpenShift Container Platform architecture

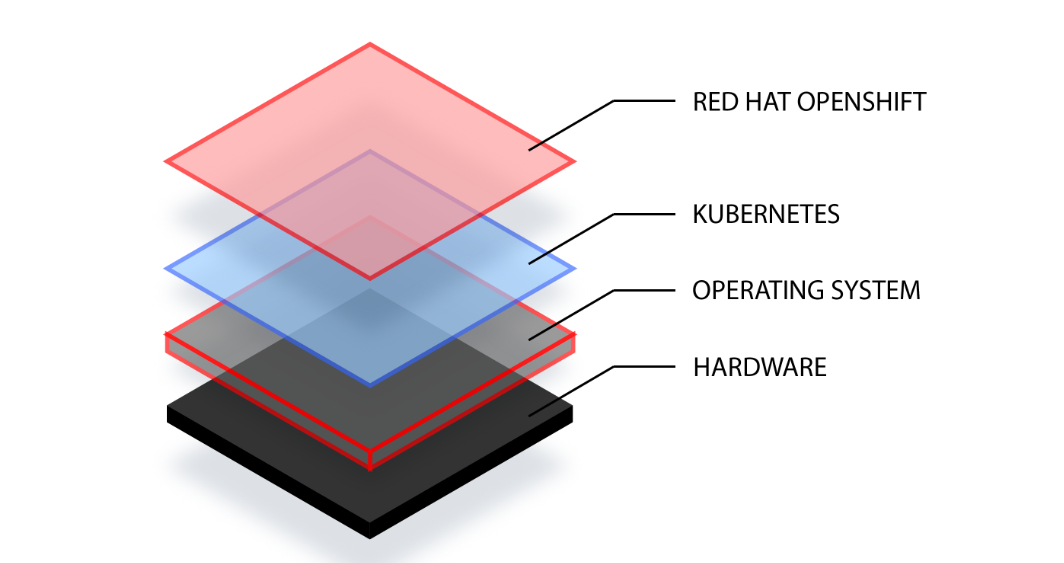
OpenShift is a platform powered by Kubernetes that allows you to run containerized applications and workloads regardless of where you choose to run them.

You can take advantage of public or private resources for running OpenShift. That includes bare metal or virtualized hardware, whether it is on-premises or on a cloud provider.

On top of that there is an operating system – which in the case of IBM Cloud Pak for Data is Red Hat Enterprise Linux CoreOS or Red Hat Enterprise Linux.

Kubernetes lies on that operating system.

Finally, there is OpenShift. It's a layer built on top of Kubernetes, and makes working with it much easier. OpenShift adds more capabilities to Kubernetes, like deploying applications and making day-to-day operations easier by building a web console and a command-line interface (CLI).



## OpenShift Routes and services

Recall that a pod is the smallest deployable unit of computing that can be made up of one or more containers. A service is a combination of IP and port which allows communication with a pool of pods. A route is a mechanism to expose applications to the outside world.

OpenShift **route**exposes a service as a host name, such as www.example.com, so that the external client can reach it by name. A **router** uses the services selector to find the service.

A router **detects relevant changes** in the internal IP address of its services and adapts accordingly.

## OpenShift use scenarios

Below you can find examples of two different types of engineers that really benefit from using OpenShift.

1. Developers.

Responsibilities - writing applications, deploying them into a cluster, creating changes.

The developer creates a project and an application. Once the code gets pushed into that GitHub, Jenkins (which is another tool for developers) will create a container image out of that source code and put it into a registry, which comes built-in in OpenShift. Once the image gets built and pushed into that registry, OpenShift will push it into the actual cluster. Whenever a change is detected with that image the software will allow you to push those with no downtime to applications.

2. Operations Engineers, Site Reliability Engineers

Responsibilities - **maintaining a high availability of a cluster,** making sure the **applications are available,**and the **infrastructure is healthy**.

OpenShift web console and the CLI are great ways to make sure that availability number meets expectations. If another host is needed to distribute the load, the process of scaling up a new host and adding it into the cluster can be quite painstaking. OpenShift takes advantage of something called Ansible Playbooks which can automate all of those different tasks that an operations engineer might have to do.

What is the Red Hat OpenShift Container Platform infrastructure?

Red Hat OpenShift Container Platform assigns each host different role. Roles define the function of a machine within a cluster. There are two types of roles defined by Red Hat OpenShift Container Platform:

* Control plane (master) role types
* Compute (worker) role types

Machines with the control plane role manage workloads on the compute (worker) machines.

Compute machines are where the actual workloads run.

Red Hat Enterprise Linux CoreOS is the only supported operating system for the control plane (master) nodes. For Compute (or worker) nodes, Red Hat Enterprise Linux CoreOS is the default, but Red Hat Enterprise Linux (RHEL) is also supported. More maintenance is required if you mix and match Red Hat Enterprise Linux CoreOS and RHEL.

A node provides the runtime environments for containers. Each node in a cluster has the required services to be managed by the control plane node. Nodes also have the required services to run pods such as a kubelet.

The illustration below shows that Control Plane consists of master machines. Pods, containers and kubelet run on compute nodes.

**Kubelet** is the primary node agent for launching and monitoring containers.

## How does the Red Hat OpenShift Container Platform relate to IBM Cloud Pak for Data?

Recall from "Overview of IBM Cloud Pak for Data" course that IBM Cloud Pak for Data is fundamentally **modular**. Cloud Pak for Data is composed of integrated **microservices**that run on a multi-node Red Hat OpenShift cluster, which manages resources elastically and runs with minimal downtime.

Because Cloud Pak for Data runs on top of Red Hat OpenShift, you can set up Cloud Pak for Data on an on-premises, private cloud cluster or any public cloud infrastructure that supports Red Hat OpenShift.

As Red Hat OpenShift  is an enterprise-ready Kubernetes container platform with full-stack automated operations it enables to manage hybrid cloud and multicloud deployments.