KUBERNETES VS OPENSHIFT: WHAT'S THE DIFFERENCE?

When it comes to container orchestration, Kubernetes and OpenShift are the prominent choices one cannot ignore. While we compare the two, keep in mind that they rather complement each other; in fact, Kubernetes is a significant component of the OpenShift platform.

Additionally, thanks to their robust and scalable configurations, both technologies offer the possibility of:

- Large-scale application development
- Management
- Deployment

In this article, we'll analyze these two popular container orchestration platforms, and explore their fundamental differences.

(This article is part of our Kubernetes Guide, which you can navigate using the right-hand menu.)

Kubernetes

Google developed Kubernetes as an open-source and portable Container-as-a-Service platform that allows an organization to administer services for: automated deployment, scaling, and management of application workloads. Adopted by the Cloud Native Computing Foundation (CNCF), Kubernetes provides out-of-the-box functionalities like:

- Process automation

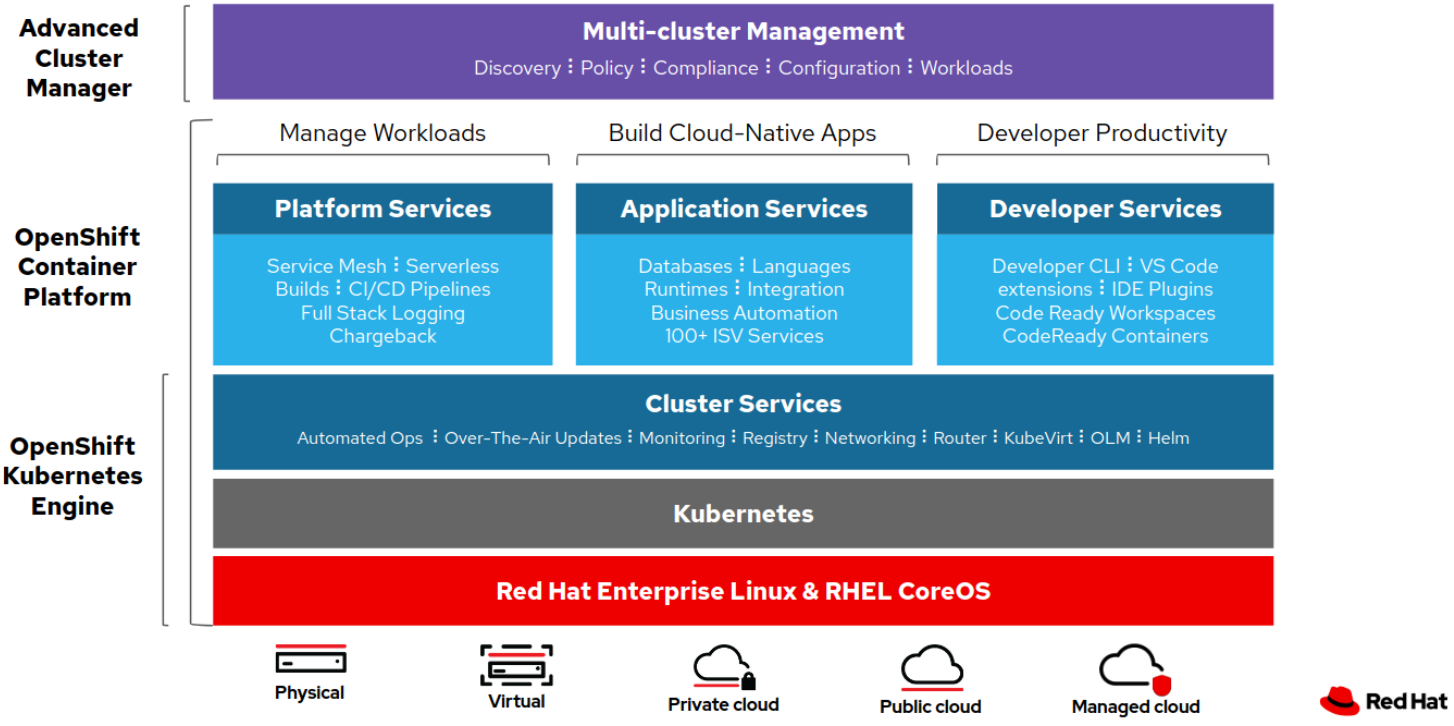
- Load balancing
- Self-monitoring
- Storage orchestration

OpenShift

Red Hat's OpenShift is a cloud-based container platform that functions as both a Platform-as-a-Service and Container Orchestration Engine. At its core, OpenShift is an open-source tool that leverages the Kubernetes platform to manage Docker containers for consistent:

- Workload management
- Self-monitoring
- Centralized policy provisioning

With OpenShift, developers can deploy containerized applications in an Integrated Development Environment (IDE), while employing Kubernetes to manage them.



(*Image source*)

Comparing Kubernetes and OpenShift

Both Kubernetes and OpenShift run on an Apache 2.0 license and help in large-scale application management and deployment. There are, however, fundamental differences in the way each of these technologies delivers their functionality.

Let's compares the main features of each.

Kubernetes Project vs OpenShift Product

Kubernetes was created as an open-source framework project. K8s is managed through an ongoing collaboration between users of the global developer's community. Though this means that the support is do-it-yourself, you also benefit from peer knowledge and collaboration within the open-source community.

OpenShift is available both as:

- A commercial product (OpenShift Container Platform)
- Public cloud (OpenShift Online and OpenShift Dedicated)

The OpenShift Container Platform is developed, managed, and administered by developers of Red Hat Inc., and comes with a paid subscription for administration and infrastructure management. This means that users receive dedicated support, with periodical upgrades. On the other hand, the open-source version of OpenShift, also known as OKD, is a community edition platform that is restricted to 'self-support'.

Compatibility & Installation

With the growing popularity and use cases of Kubernetes, the list of tools compatible with Kubernetes has grown exponentially. This remains an important consideration among developers which offers essentially the freedom of choice when it comes to platform, complexity, and upgrades.

More importantly, a wider acceptance of Kubernetes is also for the fact that Kubernetes has Managed Services on all three major public cloud platforms:

- GKE for Google's GCP
- AKS for Azure
- EKS for Amazon AWS

On the contrary, OpenShift offers limited installation options as you can only install it on three Linux distributions:

- For OpenShift 3, you will need to use either Red Hat Atomic or Red Hat Enterprise Linux (RHEL).
- To install OpenShift 4, you will need RedHat CoreOS.
- Installing the open-source version (OKD) requires CentOS or RHEL.

Security

Kubernetes lacks built-in capabilities for authentication and authorization. This necessitates the manual creation of authentication procedures, such as token bearing for security. Where the security protocol in Kubernetes is not well-defined, traffic within a Kubernetes cluster by-default also lacks encryption. As a result, an out-of-box Kubernetes instance is usually considered more susceptible to attack vectors.

OpenShift has strict and well-defined security policies. For instance, OpenShift restricts the running of Docker Containers as simple images. To maintain a minimum-security level on OpenShift, you will be required to have specific administrator privileges. Moreover, OpenShift offers an integrated server for easier authentication and authorization.

User Interface

The Kubernetes web interface is complex and not recommended for novices. To access the Kubernetes Web GUI, you will need to install the Kubernetes dashboard and forward your local machine's port address to the cluster server using **kube-proxy**. Since the dashboard lacks a login page, you will also have to create bearer tokens that facilitate authorization and authentication.

Conversely, OpenShift has an intuitive web console with a one-touch login page. The console gives you a simple, form-based interface that lets you easily modify, add, and delete resources. It is also easier to visualize your cluster roles, projects, and servers.

Deployment

Kubernetes is an open-source framework that can be deployed on:

- Any major public cloud platform: Azure, AWS, or GCP
- Any Linux distribution, including Ubuntu and Debian

To deploy Kubernetes containers, you'll need Kubernetes Objects that can easily handle concurrent updates.

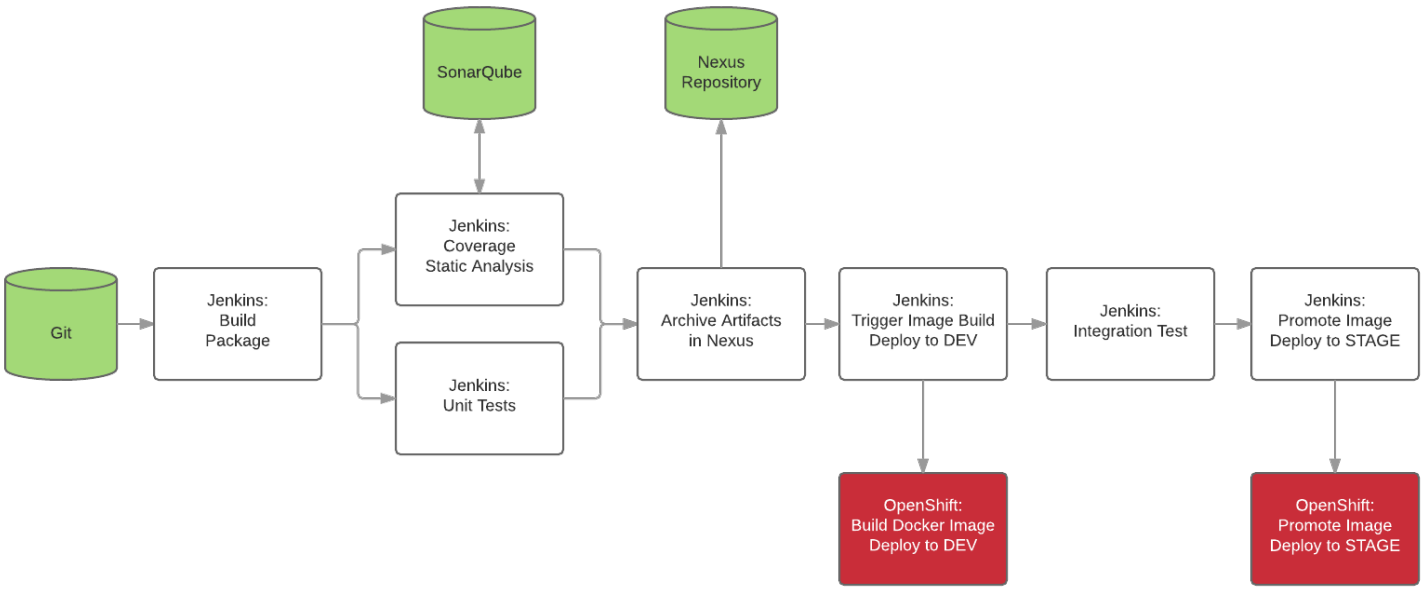
To deploy OpenShift, you'll need RHEL, Red Hat CoreOS, or CentOS Linux distributions. Deployment is performed using the **DeploymentConfig** command, which cannot be implemented with controllers, rather can be used through dedicated pod logics.

Though **DeploymentConfig** does not support concurrent updates like Kubernetes objects, OpenShift deployment, however, has certain advantages including versioning and automated deployment triggers.

Continuous Delivery/Continuous Integration

Kubernetes does not offer a comprehensive CI/CD solution out-of-the-box. You may, however, pair it with tools like automated monitoring, testing, and CI servers to help you create an entire CI/CD pipeline. Additionally, third party plugins, like CircleCI can help build faster CI/CD pipelines in Kubernetes seamlessly.

While also not a complete CI/CD solution, OpenShift has an integrated, certified Jenkins container that acts as a CI server. The following diagram shows a typical CI/CD pipeline formed out of Jenkins (for CI/CD engine), Git, Nexus Repository, and SonarQube (for SAST):



(*Image source*)

Image Registry Management

You can configure your own Docker registry in Kubernetes, but the platform lacks an integrated image registry. Kubernetes, however, does allow you to pull images from a private registry to create your own pods.

On the contrary, with an in-built image registry, OpenShift pairs seamlessly with DockerHub or Red Hat. Developers can easily search for and manage container images using image streams.

Updates

Kubernetes supports multiple upgrades occurring simultaneously. Upgrading is simple, as you only have to invoke the **kubeadm upgrade** command to get the recent version. It is always recommended that you backup the existing installation files before upgrading to a newer version.

OpenShift does not support multiple, concurrent updates automatically. You will need to access the Red Hat Enterprise Linux package management system through which you can install the most recent version of OpenShift.

Summarizing K8s vs OpenShift

Both Kubernetes and OpenShift are excellent options for large-scale deployment of containerized applications. While Kubernetes is popular with most organizations due to its flexible deployment options, OpenShift offers dedicated support and has plenty of inbuilt components that simplify app containerization, making it popular with both Agile and DevOps methodologies.

Kubernetes is unarguably the preferred choice for high-demand applications that require constant updates; OpenShift offers an all-inclusive solution that is perfect for companies that require constant and dedicated support.

Do you have a use case to share that prefers either of these or any other orchestration platforms? Let us know!

Additional resources

For more on this topic, explore these resources:

- BMC DevOps Blog
- BMC Multi-Cloud Blog
- Virtual Machines (VMs) vs Containers: What's The Difference?
- Containers in the Multi-Cloud
- Deploying vs Releasing Software: What's The Difference?