Red Hat

Developing and deploying containers using Red Hat and AWS solutions

Accelerate your container practice with Red Hat and AWS

The convergence of cloud technology advancements, microservices, and market disruptions has created a sense of urgency around digital transformation. Businesses want to move faster and be more agile and lean. Applications are key to the success of these digital initiatives, but time is of the essence. It's now common that applications need to be deployed in just a few weeks.

Application containers can help accelerate application delivery. The key is the container image, an unchangeable, static file that includes executable code so it can then run as an isolated process on IT infrastructure. The image consists of system libraries, system tools, and other platform settings that a software program needs to run on a containerization platform. The image shares the operating kernel of its host machine.

Containers introduce consistency to the process because the development and deployment environments are identical. The result is a much faster release cycle because each stage validates against a standard, immutable representation of the application.

Linux® containers are ideal for most digital transformation needs because they take advantage of the process isolation in Linux alongside the namespaces to create isolated processes. Red Hat® Enterprise Linux is a standard for running Linux containers in enterprise environments, with numerous options for building containers. With these options, developers can easily spin up containers, but they also need to manage deployment and orchestration.

This paper covers how Red Hat container tools and Amazon Web Services (AWS) developer tools help developers containerize and deliver applications.

The benefits of Linux application containers

Linux application containers help IT meet business demands faster. Developers segment applications into modular microservices that are then deployed as Linux containers. After containerization, these application components have small footprints and can be deployed faster.

Linux container applications are highly portable across architectures because they are abstracted from the hardware and the operating system. Containers make it easier to deploy and run applications across different environments, from a developer's laptop to production clusters. They also decouple application upgrades and rollbacks from the servers where they run. Containers offer operational simplicity, modularity, and flexibility, reducing the burden on IT administrators. The benefits of modularity and flexibility increase when orchestration systems, like Kubernetes or Red Hat OpenShift, are used to manage containers.

Linux containers can decrease infrastructure costs. The small, lightweight nature of containers allows them to be moved easily across bare-metal systems as well as public, private, hybrid, and multicloud environments that run the same operating system. Containers help speed development of new apps, optimize existing apps, and connect all apps, providing they are compatible with the underlying operating system (OS).

Compared to virtual machines (VMs), containers are best used for:

- Building cloud-native apps.
- Packaging microservices.
- Adopting DevOps or continuous integration/continuous delivery (CI/CD) practices.
- Moving scalable IT projects across diverse IT footprints that share the same operating system.

By contrast, VMs can run more operations than a single container, which is why they are traditionally used for monolithic workloads. However, that expanded functionality makes VMs less portable because of the dependence on the OS, application, and libraries.

Compared to containers, VMs are best used for:

- Housing traditional, legacy, and monolothic workloads.
- Isolating risky development cycles.
- Provisioning infrastructural resources, like networks, servers, and data.
- Running a different OS inside another OS—for example, running Unix on Linux.

Decreasing the number of VMs and operating systems running in an environment significantly reduces licensing requirements. Linux containers can further reduce these costs. Red Hat and AWS tools and platforms can help you build cloud-native containerized applications and adopt DevOps and CI/CD practices for greater agility and reduced time-to-market.

Application container development on Red Hat Enterprise Linux

Red Hat solutions allow you to build application containers on Red Hat Enterprise Linux. A lightweight, open standards-based container toolkit is fully supported and included with Red Hat Enterprise Linux 8. These tools also link with the AWS software development kits (SDKs) and cloud development kit (CDK), allowing you to build and test enterprise applications.

After developing containers, they need to be deployed. If using Kubernetes for orchestration, you might encounter issues related to complexity, scalability, service discovery, and more that can impede your progress. Red Hat OpenShift is designed to abstract the potential challenges related to Kubernetes and container deployment and management so you can focus on creating code. For Red Hat Enterprise Linux, a small, nimble set of tools makes it easy to develop containers.

Red Hat Enterprise Linux 8 and container development

Red Hat Enterprise Linux simplifies container development with fewer repositories and more developer tools. In addition to these tools, Red Hat provides base images to act as the foundation for your own images. Some of these base images target use cases ranging from business applications built using Node.js, PHP, Java™, and Python to infrastructure functions like logging, data collection, and authentication.

Red Hat Enterprise Linux 8 introduces new features for command-line tools like Buildah, Podman, runC, and Skopeo, the universal base image (UBI), the repository in Red Hat Quay, and the Supplemental repository, all of which lessen the complexity of developing containers.

This diagram shows the architecture of a Red Hat Enterprise Linux 8 container:

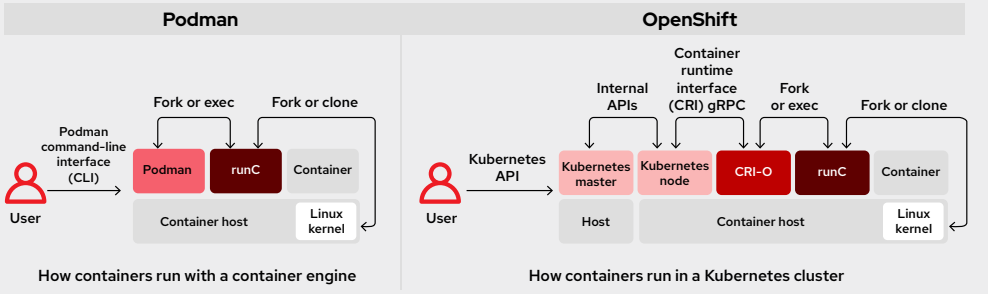

Figure 1. How Red Hat Enterprise Linux 8 containers run with a container engine or in a Kubernetes cluster

Buildah

With Buildah, you can build a container from scratch without installing a daemon or Docker. A good use case for a scratch build is to consider development images versus staging or production images of a Java application. During development, a container image for a Java application might require the Java compiler and Maven as well as other tools. But in production, you might only need the Java runtime and your packages.

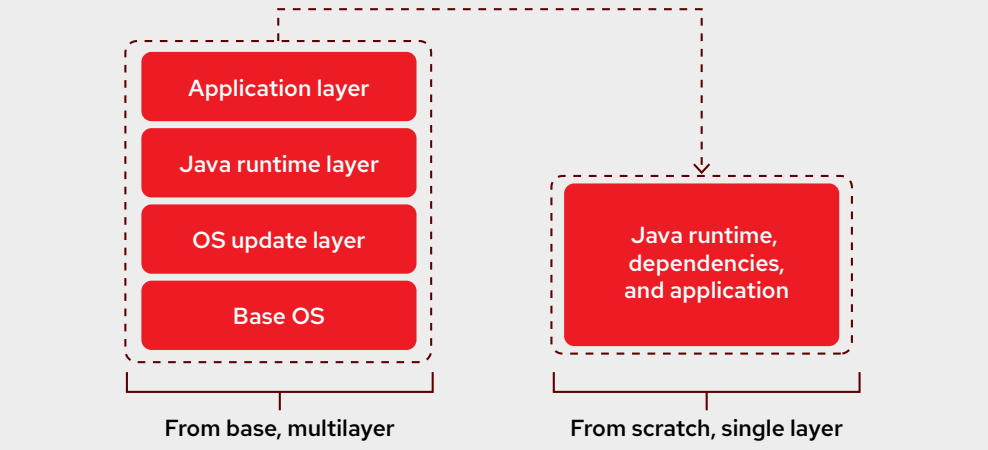

Figure 2. Layers of Java applications built from a base image or from scratch

Since Buildah is daemonless, it is easier to run it in a container without setting up special infrastructure on the host.

Buildah provides just the basic requirements needed to create or modify Open Container Initiative (OCI)-compliant Linux container images, making integration into existing build pipelines easier. For example, Buildah can assemble containers that do not include package managers (DNF/YUM), if they are not required by the final image. Buildah lets you build these containers simply and with consistent security, but it can also reduce overhead (and therefore image size) and extend customization to what you need in your cloud-native applications.

The most powerful use for Buildah is to write Bash scripts for creating your images, similar to writing a Dockerfile.

Podman

With Podman, you can manage containers without a separate daemon, and it is also compatible with the Docker command-line interface (CLI). Podman allows direct interaction with the image registry, with the container and image storage, and with the Linux kernel through the runC container runtime process.

You can run podman without root privileges and without a daemon that provides root capabilities, so there needs to be a separate place where podman can write images. Podman uses a repository in the user's home directory to avoid making /var/lib/containers world writeable or applying other practices that might lead to security problems. This also makes sure that every user has separate sets of containers and images and can use Podman concurrently on the same host without interfering with each other. When users are finished with their work, they can push changes to a common registry to share their image with others.

runC

Buildah and Podman use runC, the OCI runtime, to launch containers. You can build and run an image, or you can run docker-formatted images with runC. This Go language-based tool reads a runtime specification, configures the Linux kernel, and eventually creates and starts container processes. Although the low-level code of runC can be shared with Docker, runC does not depend on any of the components of the Docker platform. It supports Linux namespaces, live migration, and has portable performance profiles. It also provides full support for Linux security features like SELinux, control groups (cgroups), secure computing mode (seccomp), and others.

Skopeo

Skopeo inspects images in and transports them to any of the places where an OCI image can be stored. Before Skopeo was released, to inspect an image, you had to pull the whole image, even if you only wanted to inspect some metadata. Skopeo shows all image properties, including layers, so you don't have to pull the image to the host. If your inspection indicates a need to copy a container image from one location to another, you can do it with Skopeo.

Skopeo also allows you to delete an image from a repository and sync an external image repository to an internal registry for air-gapped or disconnected network deployments. When required by the repository, Skopeo can pass the appropriate credentials and certificates for authentication.

Red Hat Universal Base Image

Traditionally, Linux containers need to be built for each destination platform. The Red Hat Universal Base Image (UBI) addresses this issue. A UBI is an enterprise-grade base container image on which developers can build and deliver their applications. For Red Hat Enterprise Linux 8, all Red Hat base images are available as UBIs. With UBI, you can build and redistribute container images based on Red Hat Enterprise Linux without a Red Hat subscription, and users do not need a subscription, either.

Based on time-tested Red Hat Enterprise Linux base images, the UBI is designed to be a foundation for cloud-native and web application use cases developed in containers while eliminating the extra work of creating CentOS-based container images for community projects or for those who prefer self support.

You can build a containerized application using UBI, push it to your choice of registry server, transition it to production, and easily share it with others. Because it's freely redistributable, you can even deploy it on other vendor platforms. This capability is useful if you are using Ubuntu or another Linux platform, and it helps you get familiar with Red Hat Enterprise Linux, even though you might be on a different platform.

Repository in Red Hat Quay

You can also create an image repository in Red Hat Quay, a private container registry that stores, builds, and deploys container images. It analyzes your images for security vulnerabilities, identifying potential issues that can help you mitigate security risks. There are two ways to create a repository in Red Hat Quay: using a push (from Docker or Podman) and with the Red Hat Quay user interface (UI). You can also access public repositories in Red Hat Quay.

Supplementary repository

The Supplementary repository includes proprietary-licensed packages that are not included in the open source Red Hat Enterprise Linux repositories. Although these packages are not supported by Red Hat Enterprise Linux 8, they offer the flexibility of accessing other software as part of containerizing applications.

Red Hat Enterprise Linux on Amazon Elastic Compute Cloud (EC2)

Red Hat Enterprise Linux on Amazon Elastic Compute Cloud (Amazon EC2) is the combination of Amazon EC2 compute and Red Hat Enterprise Linux. Red Hat maintains the base Red Hat Enterprise Linux images for Amazon EC2. Updates are received as soon as they are available from Red Hat. Your computing environment and security remain reliable, and Red Hat Enterprise Linux certified apps maintain supportability. Amazon EC2 running Red Hat Enterprise Linux offers a virtual development environment and a dependable platform for delivering a broad range of applications, including containerized applications. Its distributed architecture lets you deploy applications with greater flexibility and agility.

Red Hat Enterprise Linux on EC2 gives you access to the AWS SDKs for common languages, including Go, which is considered a core container language, along with C++, Java, and Rust, all of which can be used to containerize applications. You can also combine the AWS SDK with container building blocks that are included in Red Hat Enterprise Linux. In addition, you can take advantage of the Red Hat Universal Base Image (UBI) and application streams to build, share, and collaborate on your containerized application where you want.

As you can see, Red Hat Enterprise Linux 8 container development is a viable option for developing, building, and managing individual containers and container images. However, using Red Hat OpenShift® makes container development and deployment even more scalable and efficient.

Red Hat OpenShift on AWS: Combining container deployment and development

If you are building cloud-native apps, packaging microservices, using DevOps or CI/CD practices, or moving scalable IT projects across a diverse IT footprint that shares the same operating systems, containers are ideal. Because they are not tied to operating systems, applications, and libraries, they are portable and lightweight. However, they also present challenges. If you decide to take advantage of them, you have to address how to scale, deploy, and replace them, how to manage persistent storage, and how to deal with service discovery.

Also, if you are only using a container runtime, the complexity can be an impediment. Any expansion in operations requires adding containers, quickly leading to a chain of separate processes. This can create so many containers on a server that performance is adversely affected. It is also easy to lose track of the storage that is decoupled from containers (volumes). An orchestration layer, therefore, is critical for container deployment. Kubernetes is the orchestrator of choice, but trying to deploy your own native Kubernetes can be complex and problematic. You have the option of using Red Hat Quay with Kubernetes to manage your images and the configuration details you need to get a complete application up and running. However, Red Hat OpenShift removes almost all the Kubernetes complexity so that you can get up and running on Kubernetes as fast as possible.

Development tools available in Red Hat OpenShift:

- CodeReady Workspaces allows remote development teams to provision and share environments with the click of a button, enabling faster starts and low-latency interactions.
- The Quarkus framework lets you build Kubernetes-native Java applications.
- Preview support for Buildpacks and Kaniko are available alongside S2I and Dockerfile builds through Buildah.
- Helm simplifies working with charts and releases.

Deploying containers on Red Hat OpenShift

Red Hat OpenShift is designed to simplify deploying and managing your container platform. With integrated platform monitoring and automated maintenance operations and upgrades built in, it is an administrator-friendly enterprise Kubernetes container platform.

Red Hat OpenShift caters to an "write once, run anywhere" strategy that allows you to run workloads where it is most meaningful, on-premise or in public, private, or hybrid cloud environments. On AWS, Red Hat OpenShift is integrated and packaged as another service, listed in the AWS console, so you can create and consistently manage OpenShift clusters with less time and effort. You also get on-demand billing, a single invoice, and the option of contacting AWS for support. You can deploy a Red Hat OpenShift cluster on AWS with default settings or custom AWS settings. You can also deploy a cluster on AWS infrastructure that you provisioned yourself. You can modify the provided AWS CloudFormation templates to meet your needs.

Developing containers on Red Hat OpenShift

Red Hat OpenShift includes the source-to-image framework (S2I), which allows a move straight from application code to container, significantly reducing container development complexity. You simply write images that take application source code as an input and produce a new image that runs the assembled application as output. There are two ways to build in OpenShift, through Dockerfiles and S2I.

User-supplied Dockerfiles invoke Buildah, which is included in the OpenShift builder, to create the image. Buildah also tags and pushes the resulting image. Alternatively, you can use S2I to generate a Dockerfile following the same flow as the Buildah-based Dockerfile build. The resulting builder image contains the specific intelligence required to produce that executable image (also known as a build artifact).

As S2I demonstrates, Red Hat OpenShift is as much a platform for container development as it is for deployment. Developers can build on a Kubernetes platform instead of coding to the specifics of existing AWS infrastructures. In addition, Red Hat OpenShift allows you to approach application development with containers in many ways, so you use the right one for different situations. It addresses the needs of developers who are unfamiliar with Kubernetes and just want to code, as well as expert Kubernetes developers seeking maximum flexibility.

Red Hat OpenShift supports the languages, databases, and tools that developers already use. It also makes it easy to access application development services, like those offered by AWS.

AWS developer tools for containers on Red Hat OpenShift

Red Hat OpenShift container development can be even easier when using AWS tools. These tools can also be used on platforms other than Red Hat OpenShift, including Red Hat Enterprise Linux.

AWS CodeArtifact is a fully managed artifact repository service that makes it easy for organizations of any size to more securely store, publish, and share software packages used in their software development process.

AWS Cloud9 is an integrated development environment (IDE) for writing, running, and debugging code in a browser-based shell that allows you to install additional software, do a git push, or enter commands.

AWS Cloud Development Kit uses the familiarity and expressive power of programming languages for modeling applications, accelerating onboarding to AWS because there are few new things to learn.

Amazon CodeGuru provides visualizations and recommendations on how to fix performance issues and the estimated cost of running inefficient code, helping developers prioritize remediation. It integrates into your existing software development workflow on Red Hat OpenShift, making intelligent suggestions for improving code quality and identifying an application's most expensive lines of code.

More development tools in Red Hat OpenShift:

- Odo provides a way for developers to iterate on code with its CLI. It supports Kubernetes and Red Hat OpenShift, an open model for tools through a standard definition, and rapid iterative Java development using Quarkus.
- Serverless support of Knative serving and eventing allows developers to build serverless and event-driven applications that include Strimzi (for running Apache Kafka on OpenShift) and Service Mesh.
- Tekton in Red Hat OpenShift pipelines, OpenShift plug-ins for GitHub actions, Jenkins, and GitLab runner support offers continuous integration capabilities.

Start building, deploying, and managing containers today

Building application containers doesn't have to be a challenge. Red Hat container tools and AWS developer tools can help. Deploying and orchestrating containers—the last steps of container implementation—can be the most challenging part of the process. Red Hat OpenShift removes most of the Kubernetes complexity, allowing you to deploy and manage containers at scale, no matter how they were built.

The value of Red Hat OpenShift is that it is not just a deployment solution. Developers have an array of choices for developing their containers on the platform and then deploying them. Red Hat OpenShift is available on AWS in both self-managed and fully managed options. Red Hat OpenShift allows you to deploy and manage your own OpenShift implementations on AWS. Red Hat OpenShift Service on AWS and Red Hat OpenShift Dedicated cater to fully Red Hat Cloud Services so you can focus on building and bringing applications to markets.

No matter how you develop your containers or how you deploy them, Red Hat and AWS offer solutions to streamline and accelerate the process so you can create and deploy the applications your business needs at a faster pace.

See how the Red Hat and AWS partnership helps you build modern infrastructure with Red Hat Enterprise Linux or learn more about:

- Red Hat Enterprise Linux documentation
- Podman
- Developer tools on AWS

About Red Hat

Red Hat is the world's leading provider of enterprise open source software solutions, using a community-powered approach to deliver reliable and high-performing Linux, hybrid cloud, container, and Kubernetes technologies. Red Hat helps customers develop cloud-native applications, integrate existing and new IT applications, and automate and manage complex environments. A trusted adviser to the Fortune 500, Red Hat provides award-winning support, training, and consulting services that bring the benefits of open innovation to any industry. Red Hat is a connective hub in a global network of enterprises, partners, and communities, helping organizations grow, transform, and prepare for the digital future.

facebook.com/redhatinc
@RedHat
linkedin.com/company/red-hat

North America
1 888 REDHAT1
www.redhat.com

Europe, Middle East, and Africa
00800 7334 2835
europe@redhat.com

Asia Pacific
+65 6490 4200
apac@redhat.com

Latin America
+54 11 4329 7300
info-latam@redhat.com

redhat.com
#F30136_1021