Docker Deployment Guide

Docker Deployment Guide

3/31/2025

Docker Deployment Guide

> **Warning**
> The following content has been deprecated and is maintained for reference only.

Genesys products are built and deployed with modern development principles and technologies such as Microservices, Docker, DevOps, and Automation. Product development follows the *cloud first* approach using the tools, principles, and methodologies that are applied for cloud services that includes full deployment and upgrade automation, full regression, and a blue-green upgrade model.

This is a generic guide that can be used by the product teams for Docker deployment. The product teams need to follow the instructions provided in this guide as mentioned in each of the chapters. The gist of the instructions is summarized below.

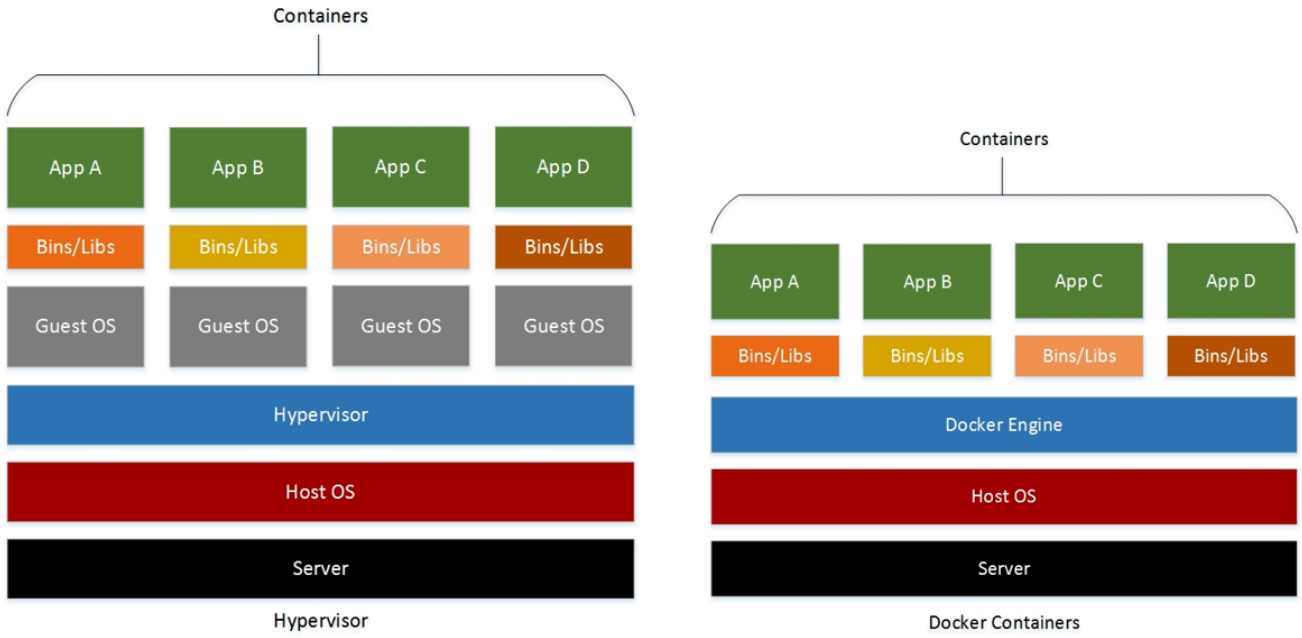
1. Install the Docker Engine CE. For details, refer to Installation of Docker Engine CE.
2. After installing the Docker Engine CE, set up the Docker Engine to use the Bintray Repository provided by Genesys. For details refer to Pulling Repositories from Bintray.
3. Deploy Docker containers in High Availability (HA) models. For details, refer to High Availability.

> **Important**
> The product team decides the HA approach and the orchestration platform to be used.

4. Implement the Blue-Green deployment model. For details, refer to Blue-Green Deployment Model. This topic explains about Blue-Green deployment with Kubernetes.

Introduction to Docker

Docker containers share the OS and kernel of the host system. Therefore, they are easily scalable and lightweight when compared to virtual machines. Each Genesys component is represented as an individual microservice. Each microservice is executed in a Docker container using N+1 horizontal scaling model principles.



Benefits of Product Dockerization

Continuous Deployment and Testing

Docker containers are configured to maintain all configurations and dependencies internally. Therefore, you can use the same container from development to production after ensuring there are no discrepancies or manual intervention.

Environment Standardization and Version Control

Docker containers help developers with easy version control and collaboration by storing the container images in a registry.

Isolation

Docker ensures that each application uses only those resources (CPU, memory, and disk space) that are assigned to it.

Security

Docker ensures that applications running on containers are completely segregated and isolated from each other.

Other Capabilities

Docker containers, when implemented on top of any container orchestration platform (like Kubernetes), will generate the following capabilities:

- Auto-scaling of containers based on demand

- Fault tolerance/self-healing
- High Availability
- Blue-Green deployment of individual containers