Container Image

Go to Terms and Definitions

Discussion

A Container Image is an immutable Image Artifact that packages the files, software, configuration, and metadata required to instantiate a containerized workload.

A Container Image may contain:

- Application files
- Executable software
- Runtime libraries
- Operating-system files
- Configuration files
- Environment defaults
- Startup instructions
- Dependency information
- Image metadata
- One or more filesystem layers

A container runtime uses a Container Image to create one or more container instances.

Multiple container instances may originate from the same Container Image while maintaining separate runtime state.

A Container Image differs from a container:

- A Container Image is the immutable image Artifact used to create a container
- A container is a runtime instance created from a Container Image

A Container Image differs from a Machine Image:

- A Container Image packages the content required to instantiate a containerized workload within a container runtime
- A Machine Image packages the content required to instantiate a virtual machine

A Container Image may use layered content. A change to image content produces a new Container Image rather than modifying an existing immutable image in place.

A Container Image may participate in:

- Image building
- Image signing
- Image verification
- Compliance assessment
- Image promotion
- Repository publication

- Offline transfer
- Containerized-workload deployment

Definition

immutable image artifact that packages the files, software, configuration, and metadata required to instantiate a containerized workload

Source

Dido Solutions, Inc. and Jackrabbit Consulting, Inc.

Note

A Container Image may be identified by:

- A repository name
- A tag
- A content digest
- A Revision identifier
- A release identifier
- A combination of these identifiers

A mutable tag does not make the referenced Container Image mutable. A tag may be reassigned to another Container Image, while each content-addressed image remains distinguishable by its digest.

A Container Image does not by itself establish that the image is:

- Approved
- Signed
- Verified
- Free from vulnerabilities
- Compliant with an applicable Baseline
- Authorized for deployment
- Compatible with a particular container runtime
- Suitable for a specified purpose

Separate requirements govern those determinations.

The term does not prescribe a particular image format, container runtime, image registry, orchestration platform, operating system, or provider implementation.

Example

Crucible builds a Container Image containing an application, its runtime libraries, configuration defaults, startup instructions, and identifying metadata. A container runtime later uses the Container Image to instantiate multiple containerized workloads.
