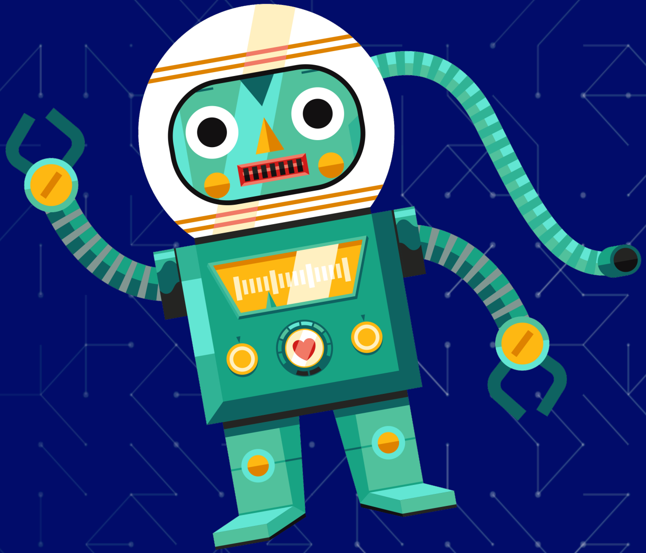


TECHORAMA

TECHORAMA

Supercharging your DevOps pipelines with Docker Containers

Steven Follis
Docker Inc.

Howdy!

I’m Steven Follis:

- Solutions Engineer @ Docker Inc.
- Working with customers with both greenfield & brownfield applications
- Will be smuggling as much chocolate back to The States as possible post-Techorama

docker

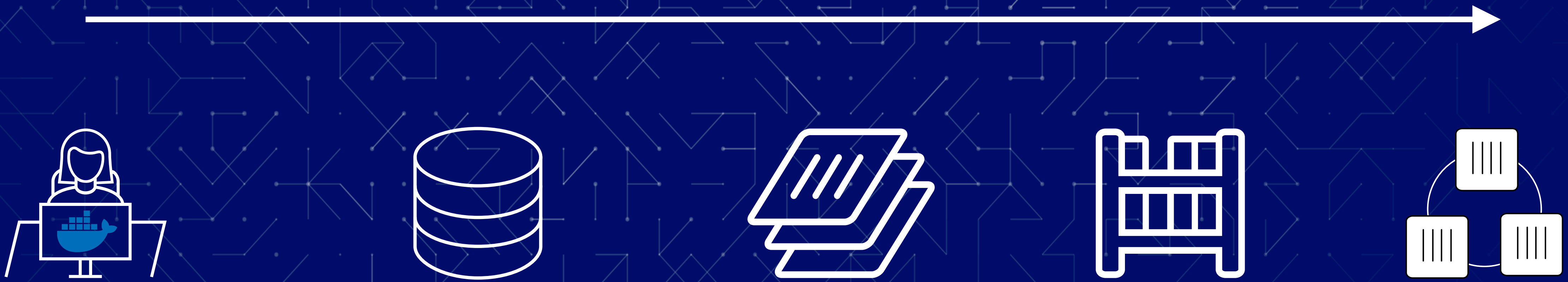
Today

Build

Share

Run

Building a software supply chain with Docker



Develop Code with Docker Desktop

Commit Code to SCM

Build Docker Image

Push image to Registry

Deploy to cluster

Building containers across the ecosystem

circleci CODESHIP Travis CI Azure DevOps

Hosted SaaS

cloudbees circleci Bamboo GitLab

Supported On-Premises

Open Source

And More!

Building Docker Containers

Traditional software builds

Traditional Builds

- VM-based
- One size fits all
- SDK versions
- Dedicated infrastructure
- Managed by dedicated teams
- Difficult to scale

Container Builds

- Container-based
- Customizable per application
- Easily adjust SDKs, etc.
- Leverage clusters
- Manageable by developers
- Easily scaled

Containerized Build Agents

Connect to host’s Docker Daemon

Linux: Mounted Socket

```
docker run -v /var/run/docker.sock:/var/run/docker.sock
```

Windows: Named Pipe Mount

```
docker run -v \\.\pipe\docker_engine:\\.\pipe\docker_engine
```

Interact with host’s Daemon via Docker CLI within container

Update Build Agent’s Dockerfile to add libraries, dependencies, etc.

Optimizing Multi-Stage Builds

Stage 1: Build

```
$ docker image ls \
  --format '{{.Repository}}:{{.Tag}} -- \
  {{.Size}}' | grep mcr
mcr.microsoft.com/dotnet/core/sdk:2.1 -- 1.74GB
mcr.microsoft.com/dotnet/core/aspnet:2.1 -- 253MB
mcr.microsoft.com/dotnet/core/runtime:2.1 -- 180MB
```

Stage 2: Runtime

Building for different architectures

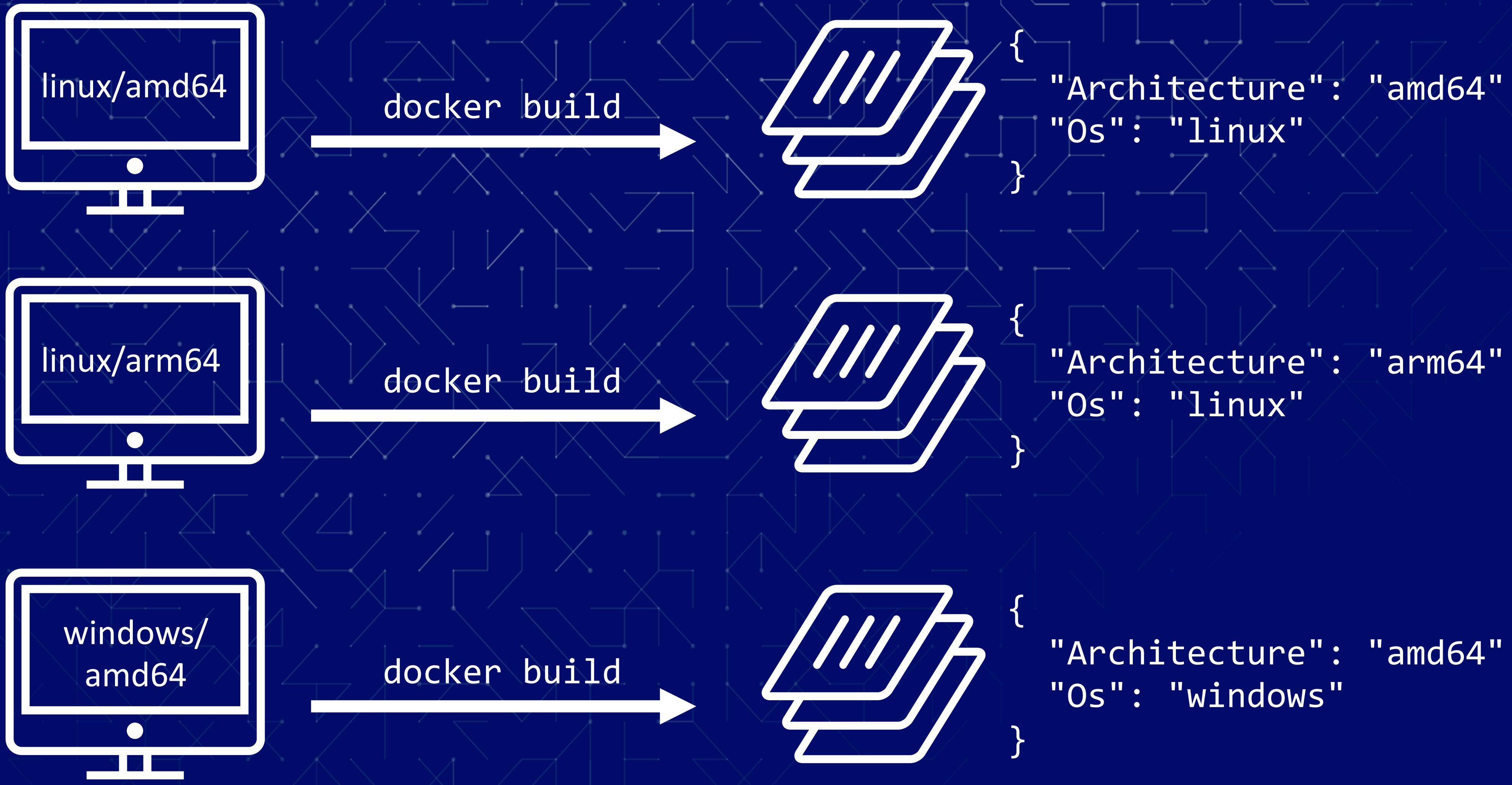
```
docker run hello-world
```

Laptop

Server

IoT Device

Multi-Architecture Builds



linux/amd64

docker build

```
{
    "Architecture": "amd64"
    "Os": "linux"
}
```

linux/arm64

docker build

```
{
    "Architecture": "arm64"
    "Os": "linux"
}
```

windows/ amd64

docker build

```
{
    "Architecture": "amd64"
    "Os": "windows"
}
```

Multi-Architecture Builds

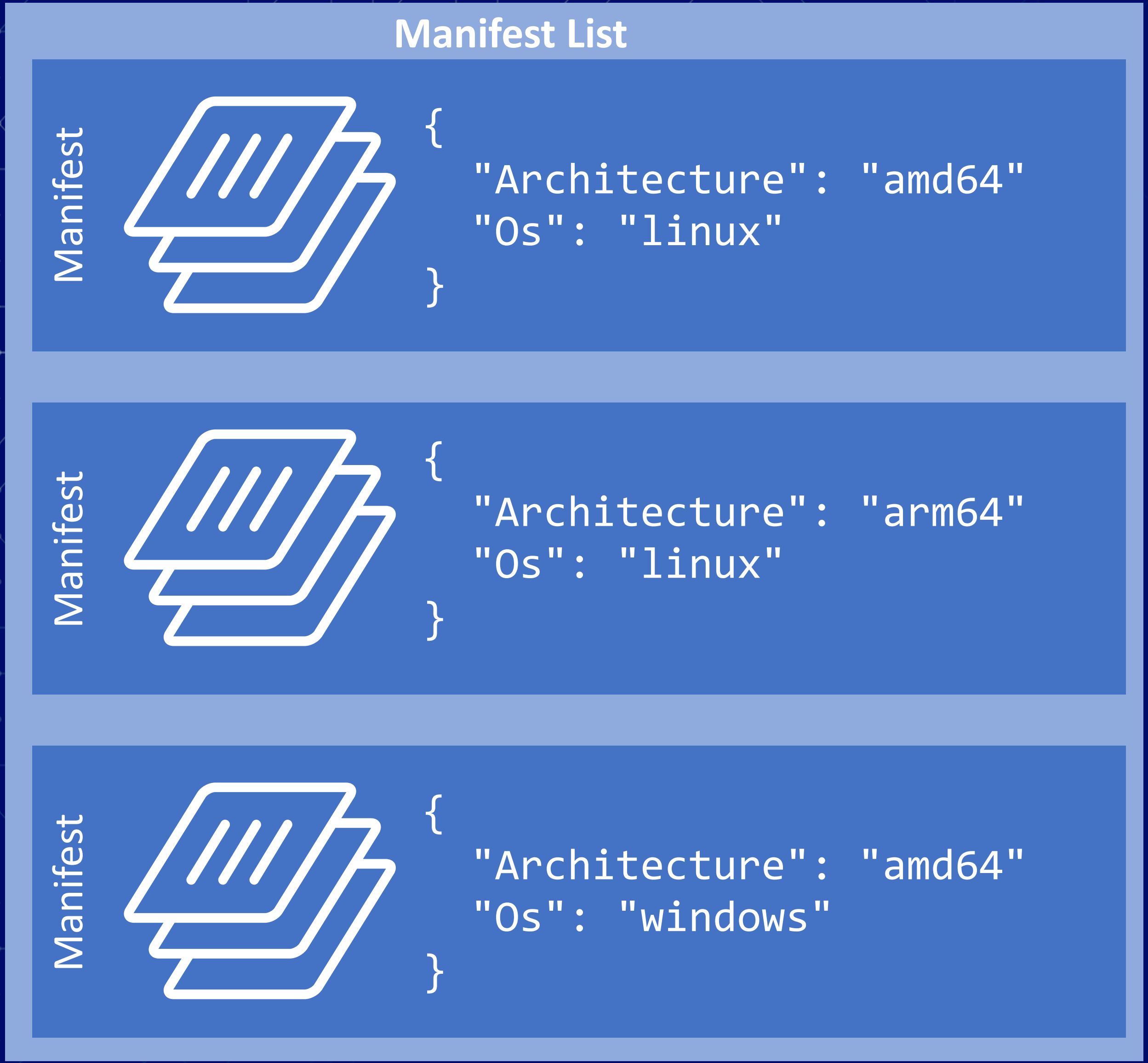
```
$ docker manifest create \
      techorama/app:multi-arch \
      techorama/app:linux \
      techorama/app:windows

$ docker manifest push \
      techorama/app:multi-arch
```

Requires Experimental CLI:

```
~/.docker/config.json
{
  ...
  "experimental": "enabled"
}
```

or

```
$DOCKER_CLI_EXPERIMENTAL="enabled"
```



Manifest List

Manifest

```
{
  "Architecture": "amd64"
  "Os": "linux"
}
```

Manifest

```
{
  "Architecture": "arm64"
  "Os": "linux"
}
```

Manifest

```
{
  "Architecture": "amd64"
  "Os": "windows"
}
```

Choosing a base image

Docker Hub includes various image types

- Certified = built & tested for best practices and security by vendor + Docker
- Official = curated set of open source images from Docker Inc.

Private Registry

- Your organization’s registry
- Control access per repository with role-based access controls

Derivative images can introduce dependencies

- Ex. microsoft/mssql-server-windows-express

Registry Tips

Use immutable Tags

Ditch :latest tag

Scan for vulnerabilities

New build system coming with BuildKit

Improves `docker build` performance, storage, & extensibility

- Run build steps in parallel when possible
- Safely mount in volumes, secrets, and more
- More precise cache support
- Output artifacts such as binaries

Enable today with `export DOCKER_BUILDKIT=1`

- Opt-in feature since Docker v18.09
- Linux support today, Windows coming soon

DCSF Talk: https://www.docker.com/dockercon/2019-videos?watch=open-source-summit-build-kit

Extending builds with buildx

Docker CLI Plugin for extending BulidKit capabilities

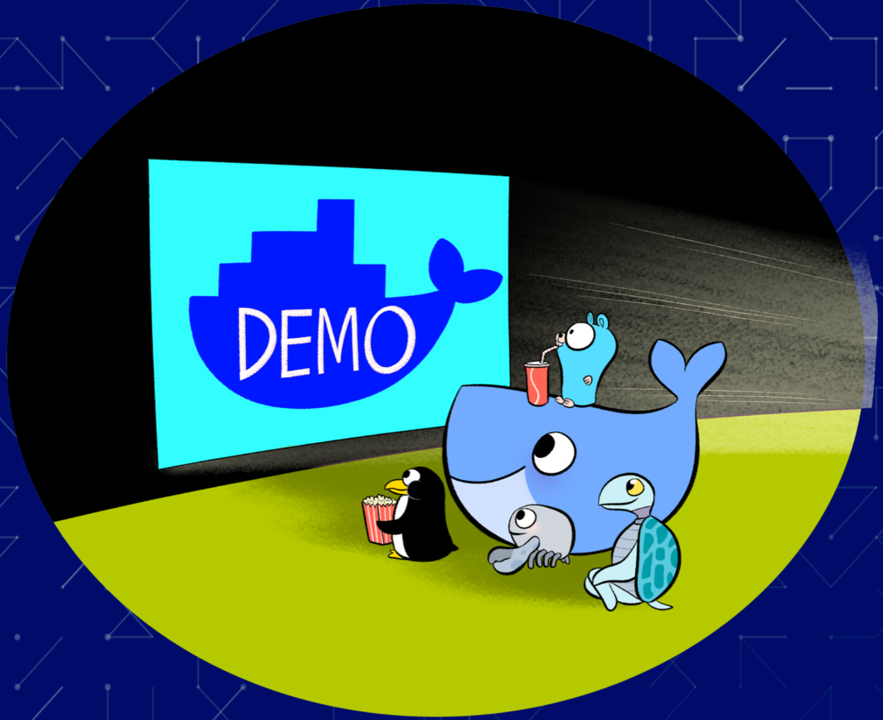
Native support for Multi-Architecture builds

- Emulation via QEMU
- Native on remote servers

```
$ docker buildx build \
  --platform linux/amd64,linux/arm64,linux/arm/v7 \
  .
```

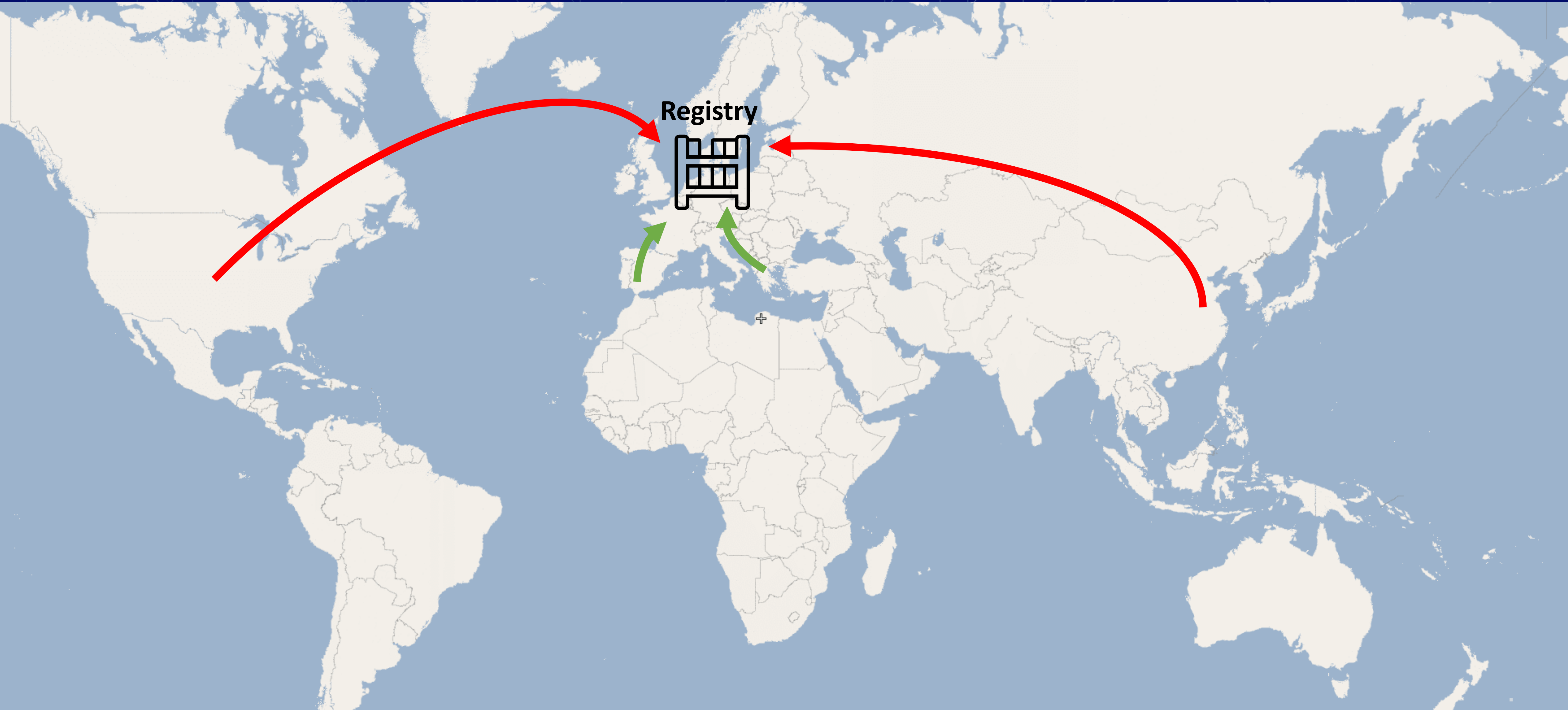
Follow development at **https://github.com/docker/buildx**

TECHORAMA

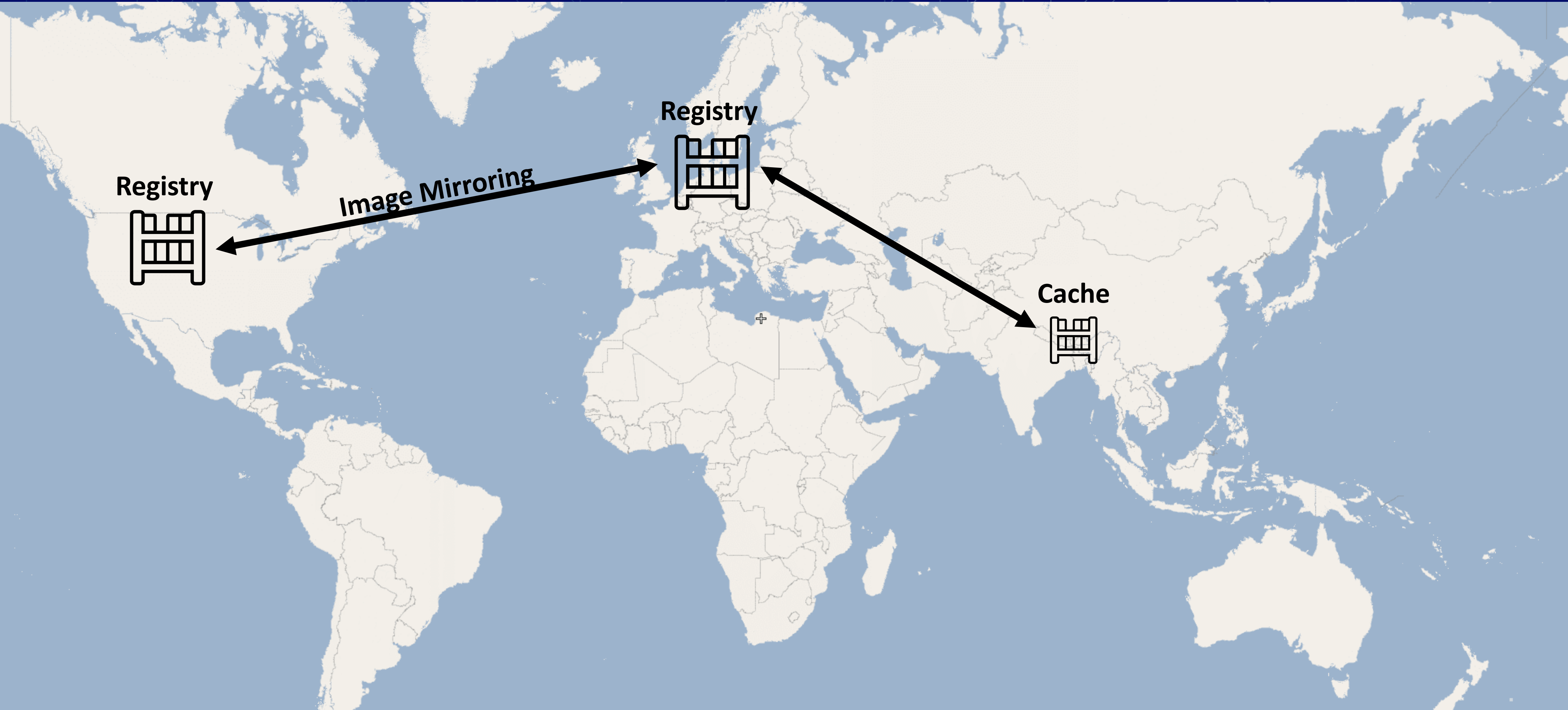


Share Containers

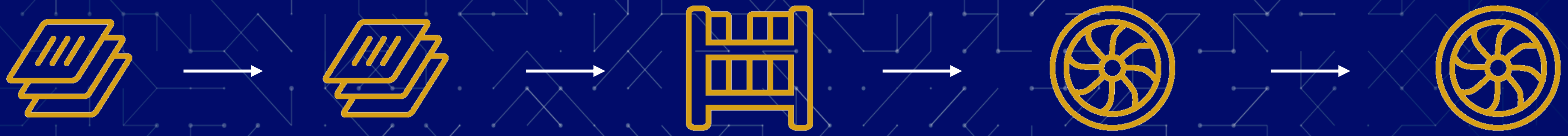
Multi-region registries



Multi-region registries



Signing images with Content Trust



User Builds or Pulls Docker Image

User Signs the Digest with RBAC Public Key

User Pushers Image (with signing data) to a Registry with a Notary Server

User Enables Content Trust on Cluster or Engine

Container Signing Metadata is checked against RBAC Engine Before Container Starts

Run Containers

v3 Alpha
now available

Helm

Kubernetes-specific package manager

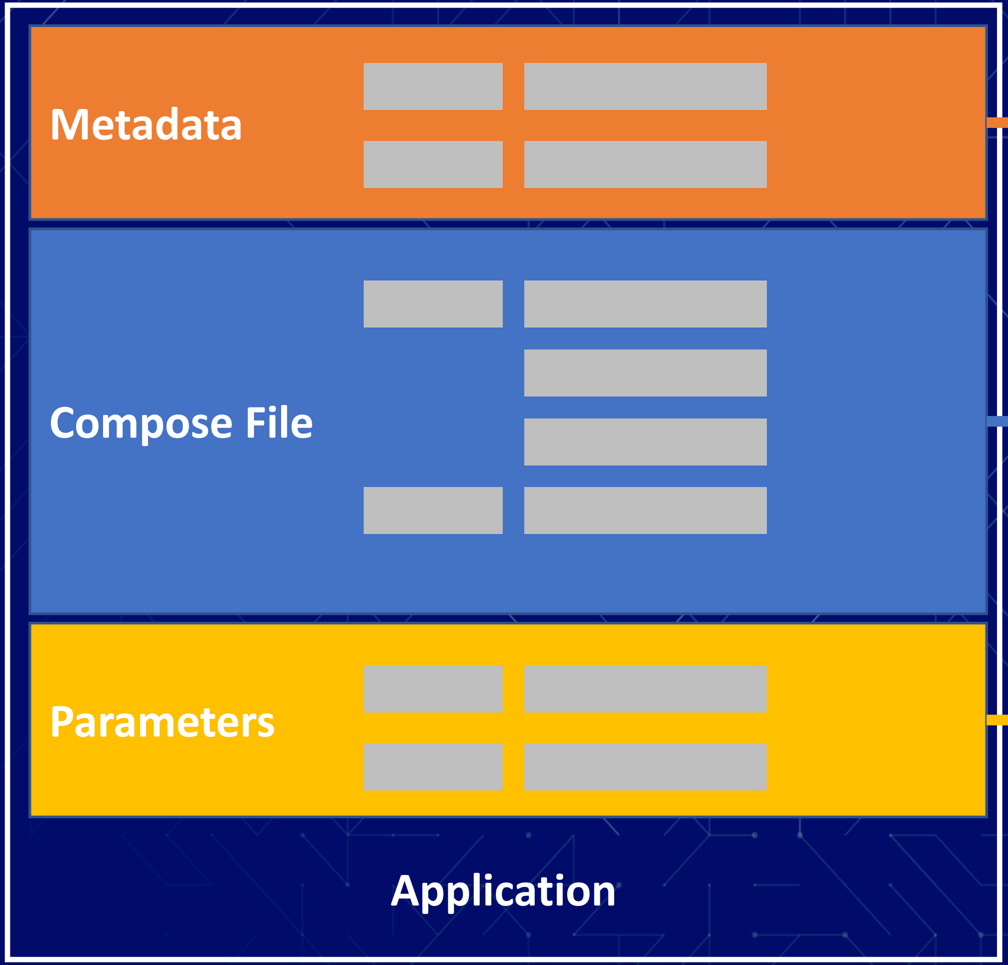
Streamlines the acquisition, deployment, and lifecycle management of 1st & 3rd party applications

Cloud Native Computing Foundation (CNCF) incubation project

Helm

```
foo/
  |
  |- .helmignore        # Contains patterns to ignore when packaging Helm charts.
  |
  |- Chart.yaml         # Information about your chart
  |
  |- values.yaml        # The default values for your templates
  |
  |- charts/            # Charts that this chart depends on
  |
  |- templates/         # The template files
  |
  |- templates/tests/   # The test files
```

Docker App

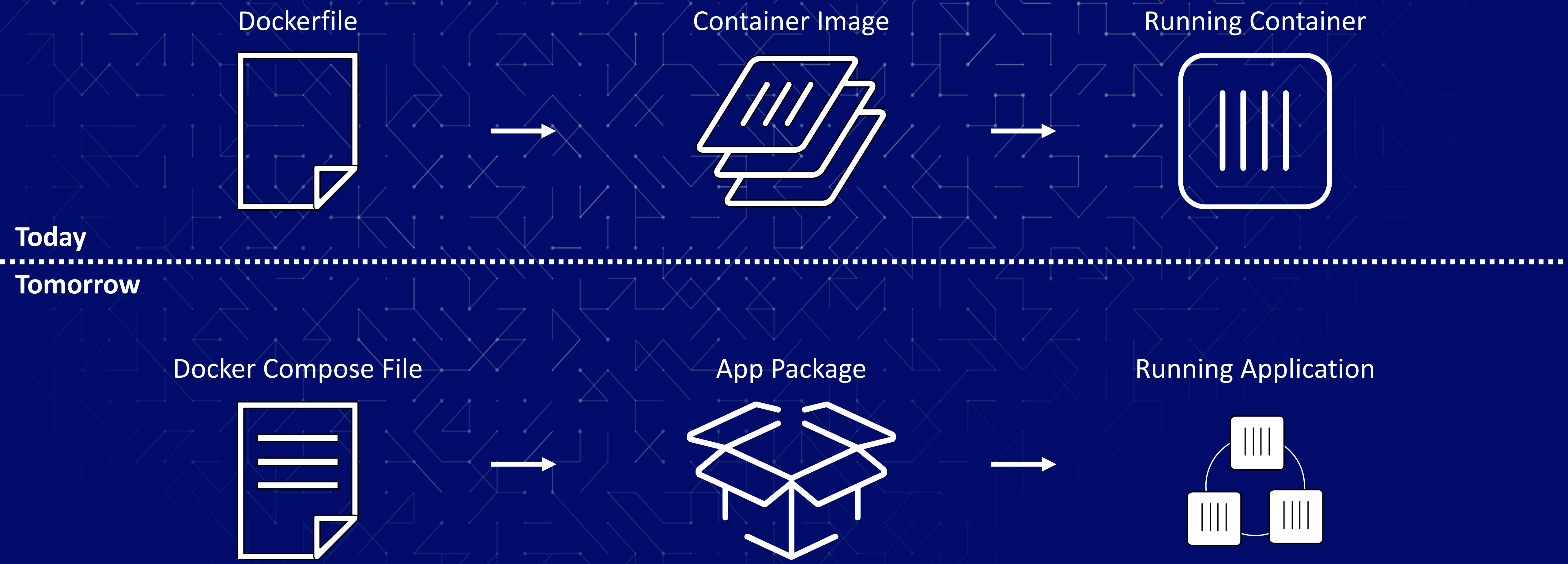


Name, Version, Description

Services, Volumes, Networks, Secrets

Install parameters

Docker App



Dockerfile → Container Image → Running Container

Today

Tomorrow

Docker Compose File → App Package → Running Application

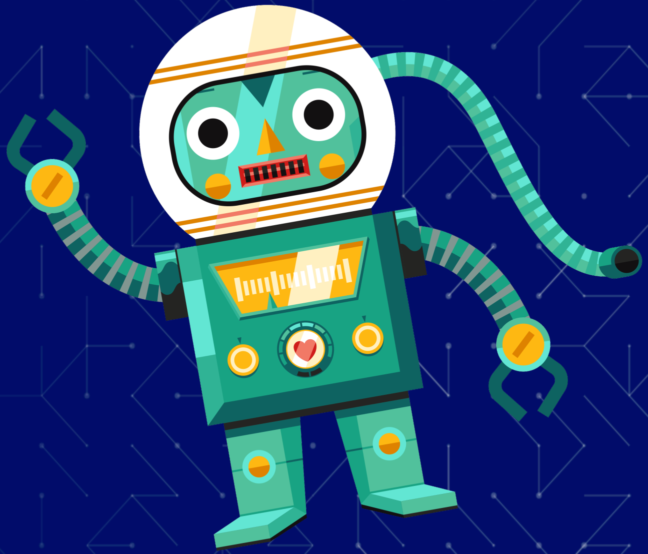
Summary

1. Refine build artifacts to save time and increase efficiency

2. Sign images & distribute near to your team

3. Use automation tooling for more consistent lifecycle management

https://tinyurl.com/techoramadocker



TECHORAMA

https://tinyurl.com/techoramadocker