Docker: Images/Cache/Container Lifecycle Management

Ashish Desai

May 2019

https://www.linkedin.com/in/logmaster

Bio

Fidelity's mission is to inspire better futures and deliver better outcomes for the customers and businesses we serve. With assets under administration of $7.3 trillion, including managed assets of $2.6 trillion as of February 28, 2019, we focus on meeting the unique needs of a diverse set of customers: helping more than 30 million people invest their own life savings, 22,000 businesses manage employee benefit programs, as well as providing more than 13,500 financial advisory firms with investment and technology solutions to invest their own clients' money. Privately held for more than 70 years, Fidelity employs more than 40,000 associates who are focused on the long-term success of our customers.
For more information about Fidelity Investments, visit https://www.fidelity.com/about.

I represent, Enterprise Cloud Computing, the centralized Cloud organization for Fidelity Investments: Our mission is to power the next generation of Fidelity digital services through innovation, collaboration, and transparency

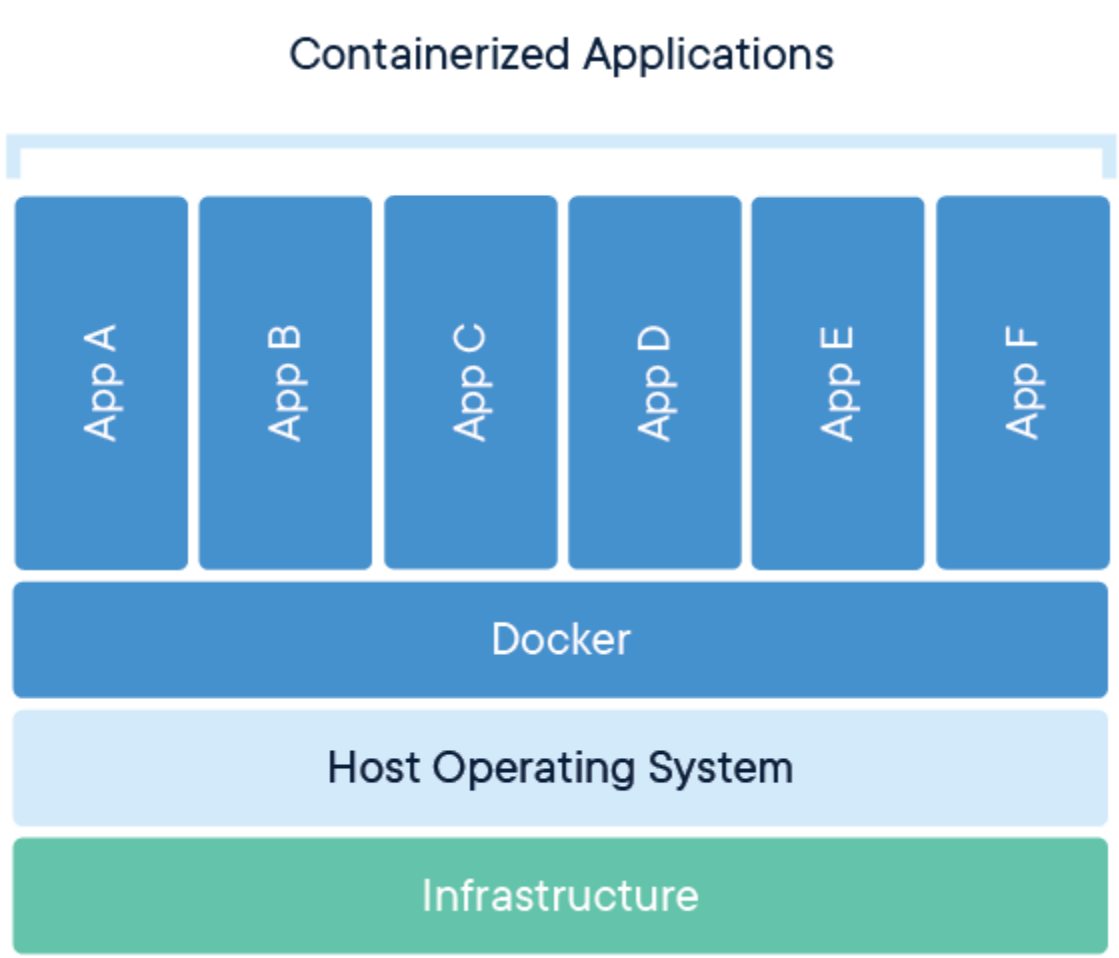
Agenda

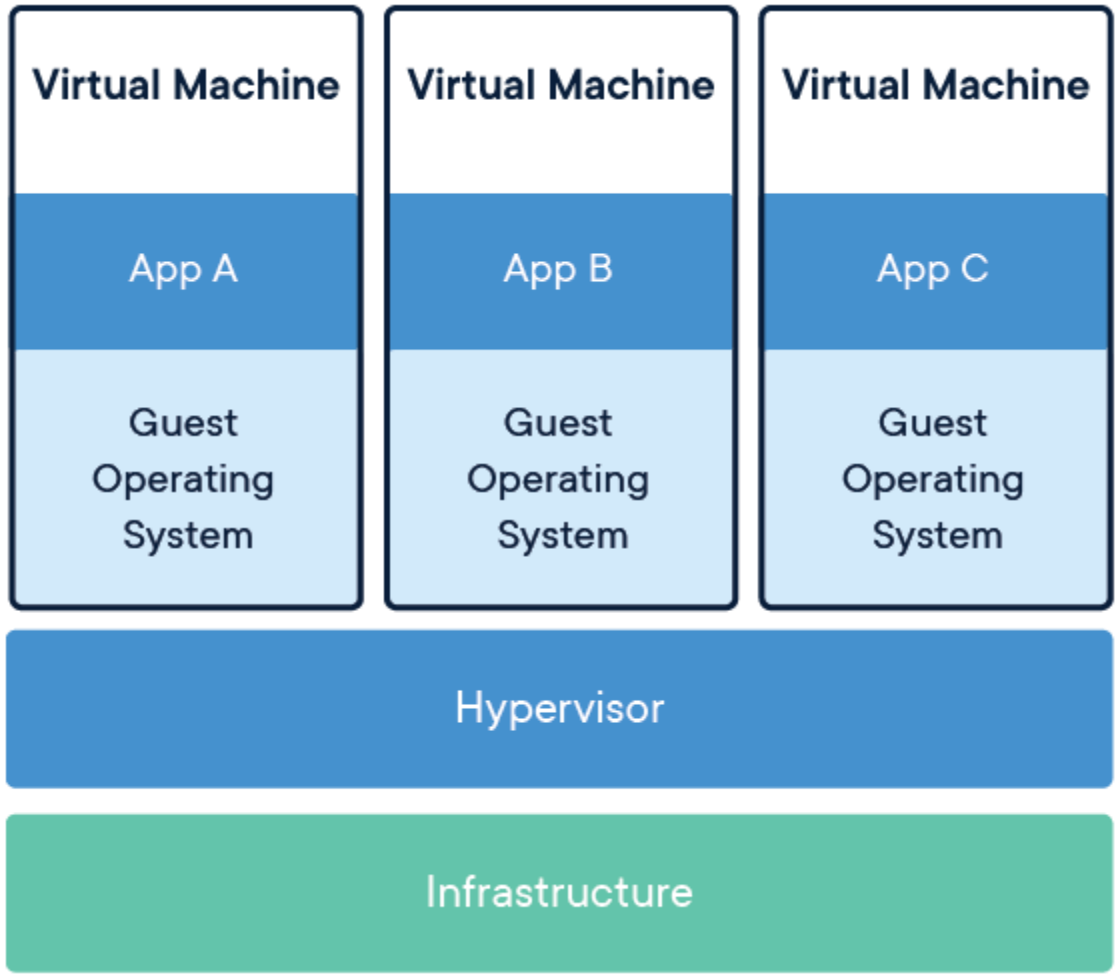
- What is Docker
- Lifecycle
- Challenges
- Proposal
- Policy

What is Docker?

Docker	An open source system to run “containers” on Linux using kernel “cgroup” feature
Dockerfile	Defines the contents of a Docker Image (files, network port, process to run)
Image	The components of a Docker container stored on disk
Container	A running instance of a “Docker Image”
Host	VM where multiple containers run
Host Cache	A local cache on a docker host, contains “Docker Images” that were started as containers, are running or have run in the past
Registry	Registry for storing multiple “Docker Images”

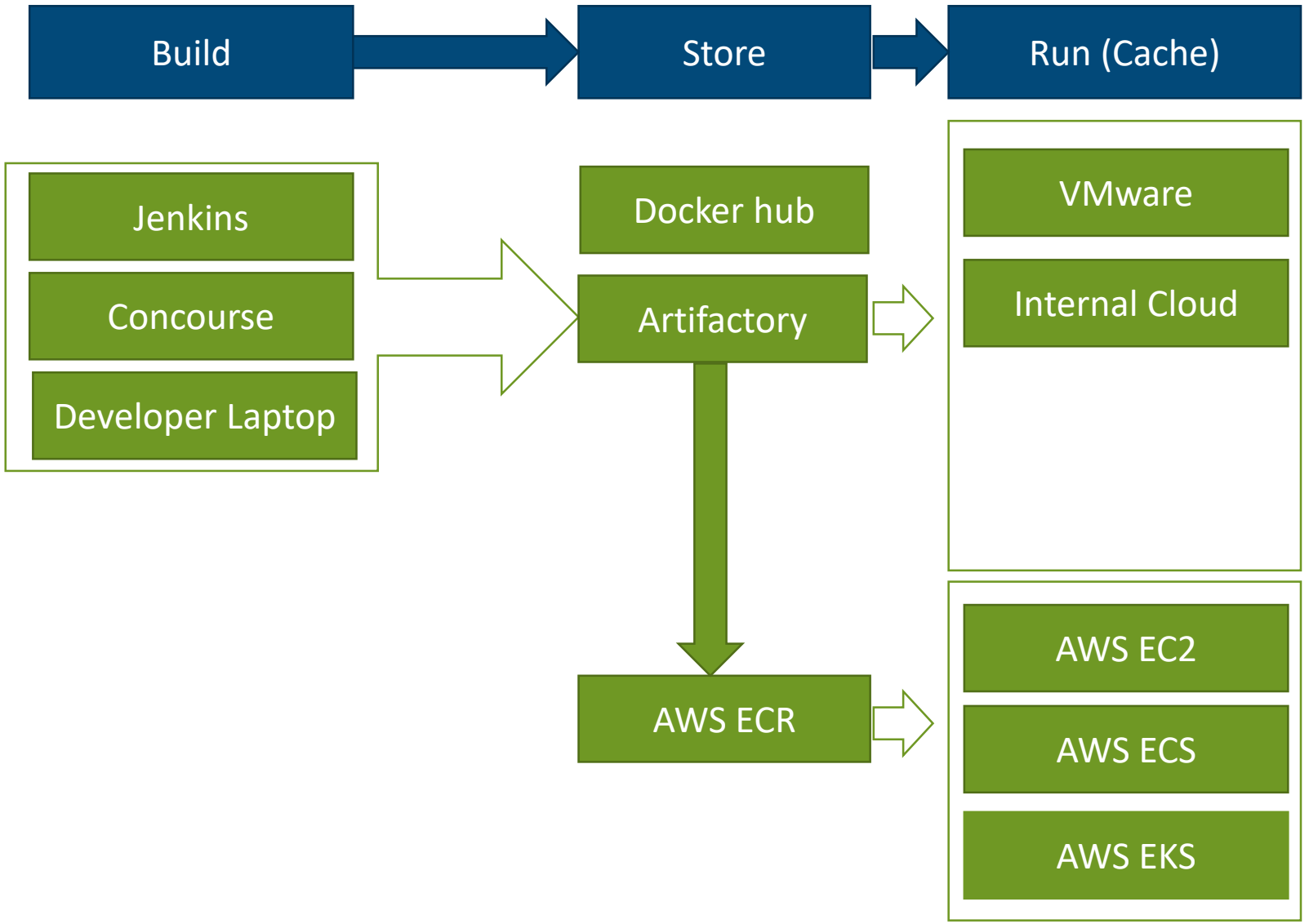
Docker vs Traditional VM





https://www.docker.com/resources/what-container

Lifecycle

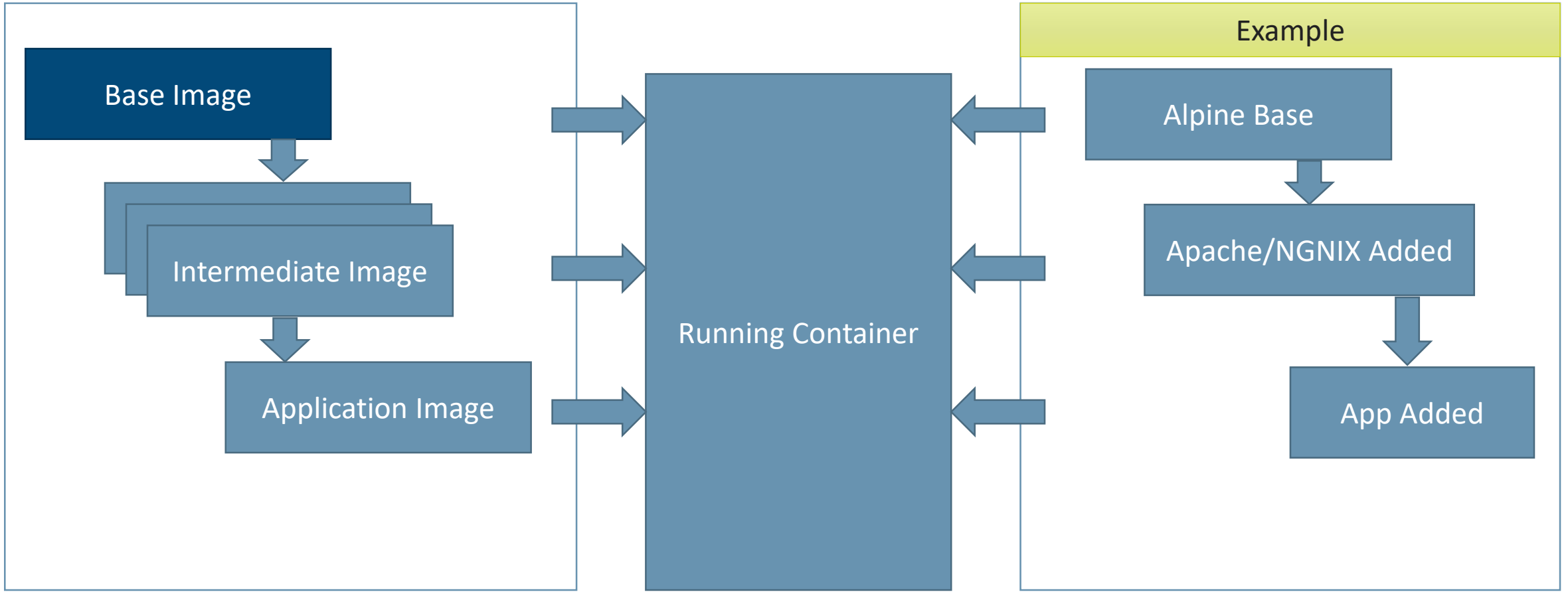


Build
Create a docker image given a Dockerfile manifest

Store
Push Docker image to Docker Registry

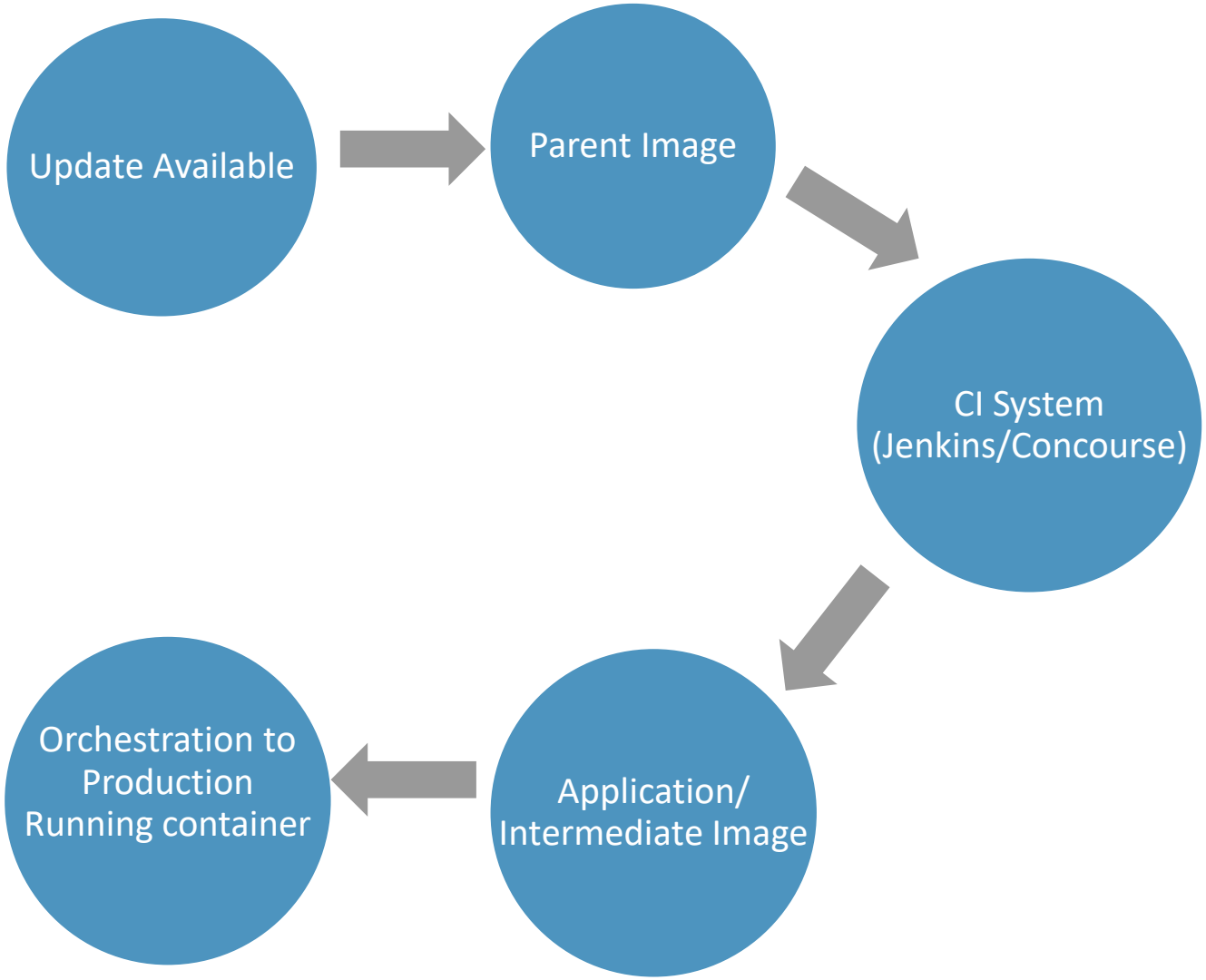
Run
Pull Docker image from Docker Registry and create a running container (Store image in cache)

Lifecycle (Part 2)



"Base", "Intermediate" or "Application image can be run as a container

Container lifecycle model



Challenges

Build	Store	Cache	Run
What is the Base Image	What is the source Dockerfile	What is the source repository	
Age of Base Image	Age of Docker Image	Age of Docker images	Age of running container
Commands in Dockerfile			References to on disk storage volume
Output Size	Size of Image	Size of Image	Size of Running container
Ownership of Dockerfile	Ownership of Image	Ownership of Cache cleanup	Ownership of Start/Stop operation

Potential Solutions

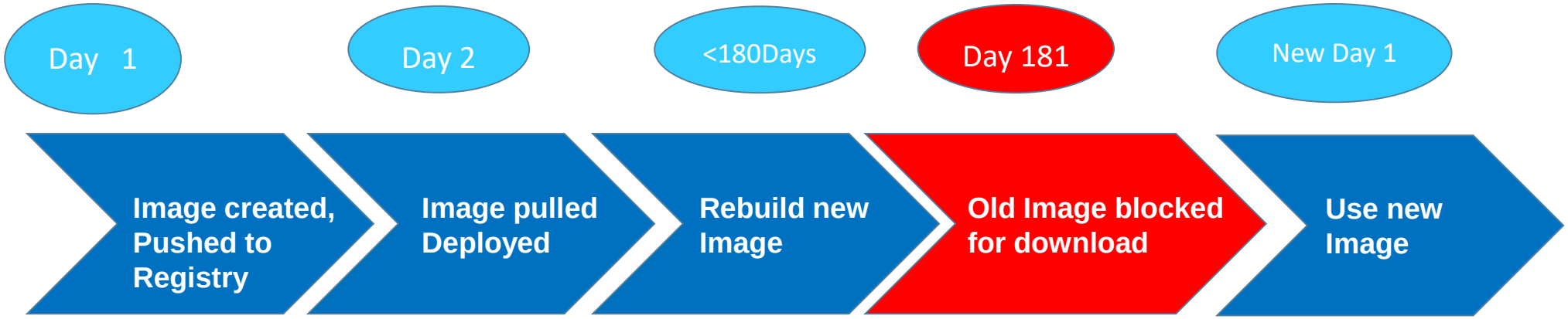
A Docker governance policy

- ❑ Mandate where the image can be built and are stored
- ❑ Mandate the age of an image
- ❑ Constrain set of base images used
- ❑ Central admin delete images in Registry after X days from creating time
- ❑ Central admin cleanup old images from host cache
- ❑ Central admin kill running containers after time X

Docker: Images/Container Lifecycle Management Roadmap

Year 1 Q1 Inventory	Year 1 Q2 Policy Definition	Year 1 Q3/Q4 Monitor & Report	Year 2 Q1 Image Compliance	Year2 Q2 Container Compliance
No defined policy Lack of control	Review policy with Business Teams Development Teams	Auditing and reporting processes defined and documented	BUs have formal strategy to comply	
Collect Inventory	Design reporting process	Regular Compliance reporting defined		
Draft Lifecycle management policy	Policy Approved	Define enforcement process	Compliance enforced for images	Compliance enforced for containers
		Awareness across BUs	Image Mgmt achieved	Container Mgmt achieved

Image Lifecycle



- Build
- Development
- Testing

- Production

- Rebuild image based on updated parent image (Recommended monthly)

- Old image will be blocked for download regardless of vulnerability status

Questions?

Thank You

https://www.linkedin.com/in/logmaster