Securing Containers

What's so different?

Joel Lathrop
JOEL@DIDACTIC-SECURITY.COM

Triangle InfoSeCon | October 2018

DIDACTIC SECURITY

Software development

Automation

DevOps

Consulting

Forensics

Pen testing

Malware reversing

Cryptography

Didactic Security

"Containers don't contain."

– Dan Walsh

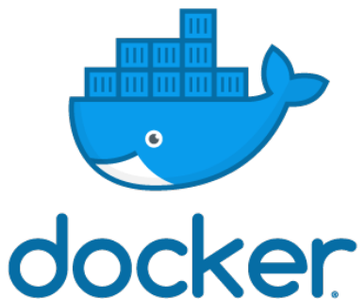
"… but they sure are handy!"

– your developers

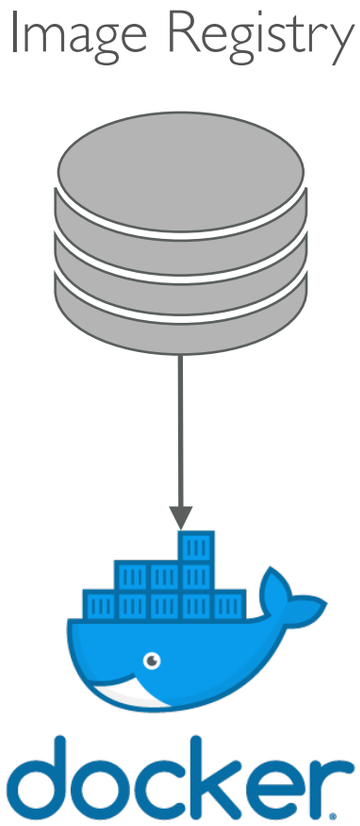
Goals

- What is a container?
- What are the security implications?
- How do I respond?

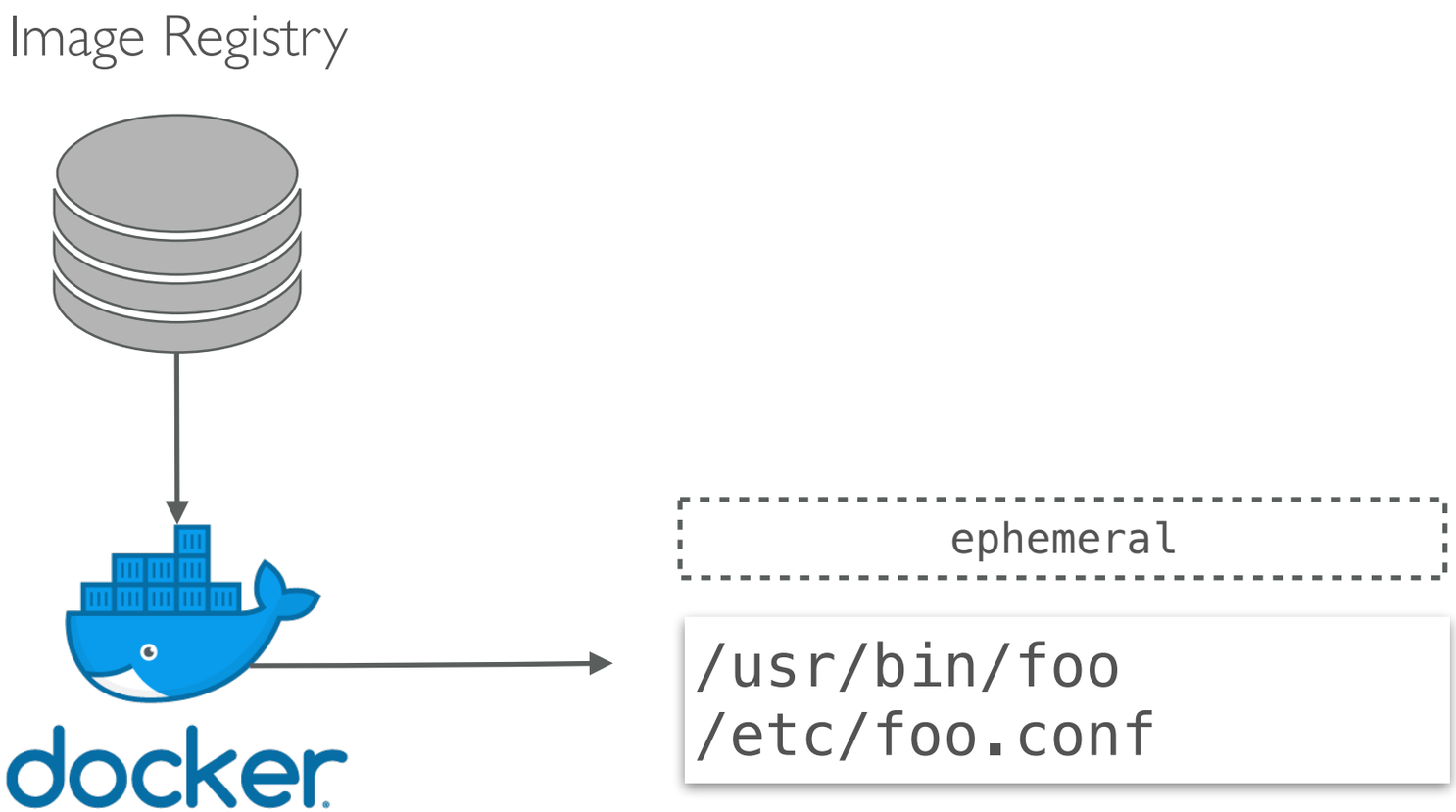
Anatomy



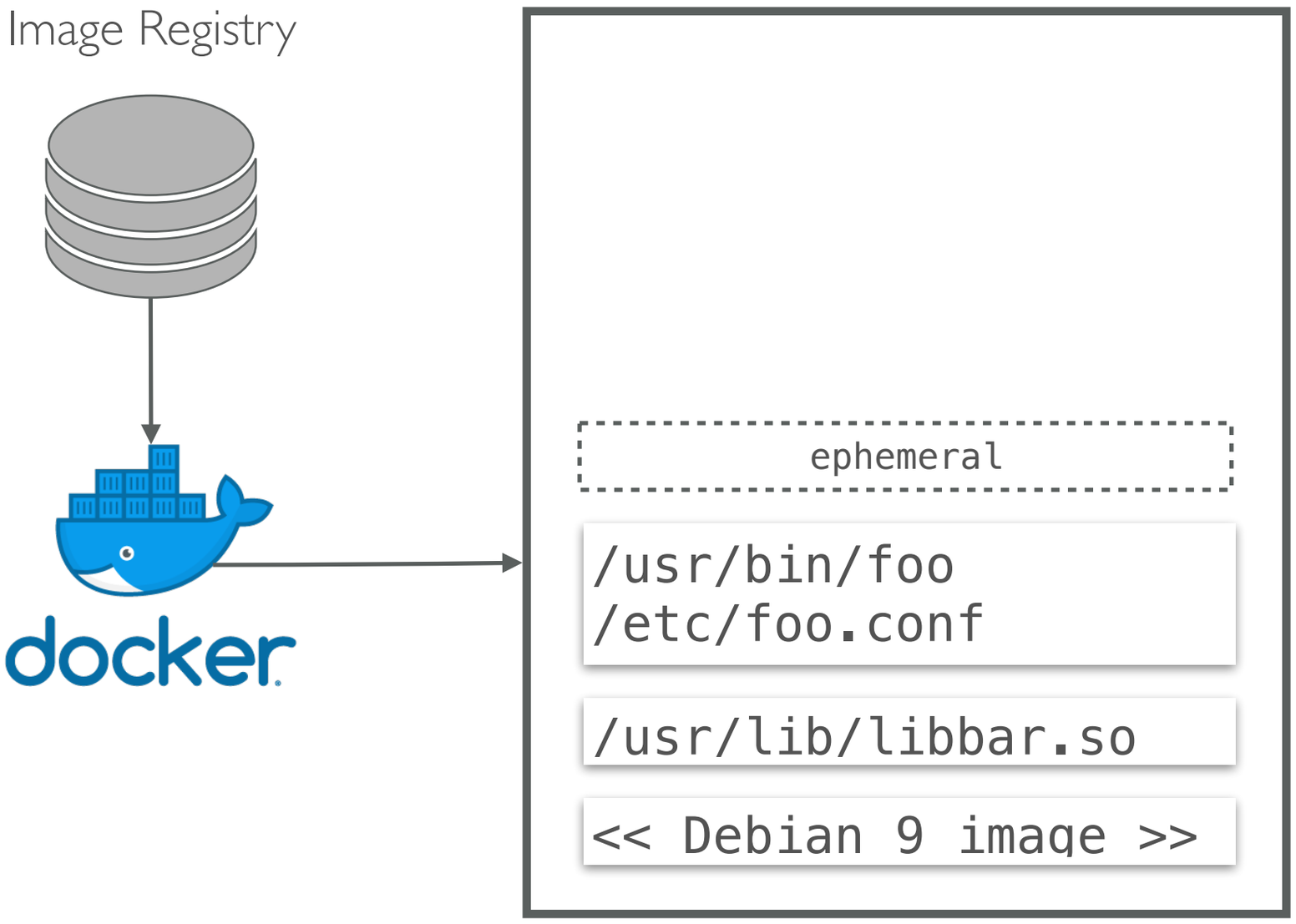
Anatomy



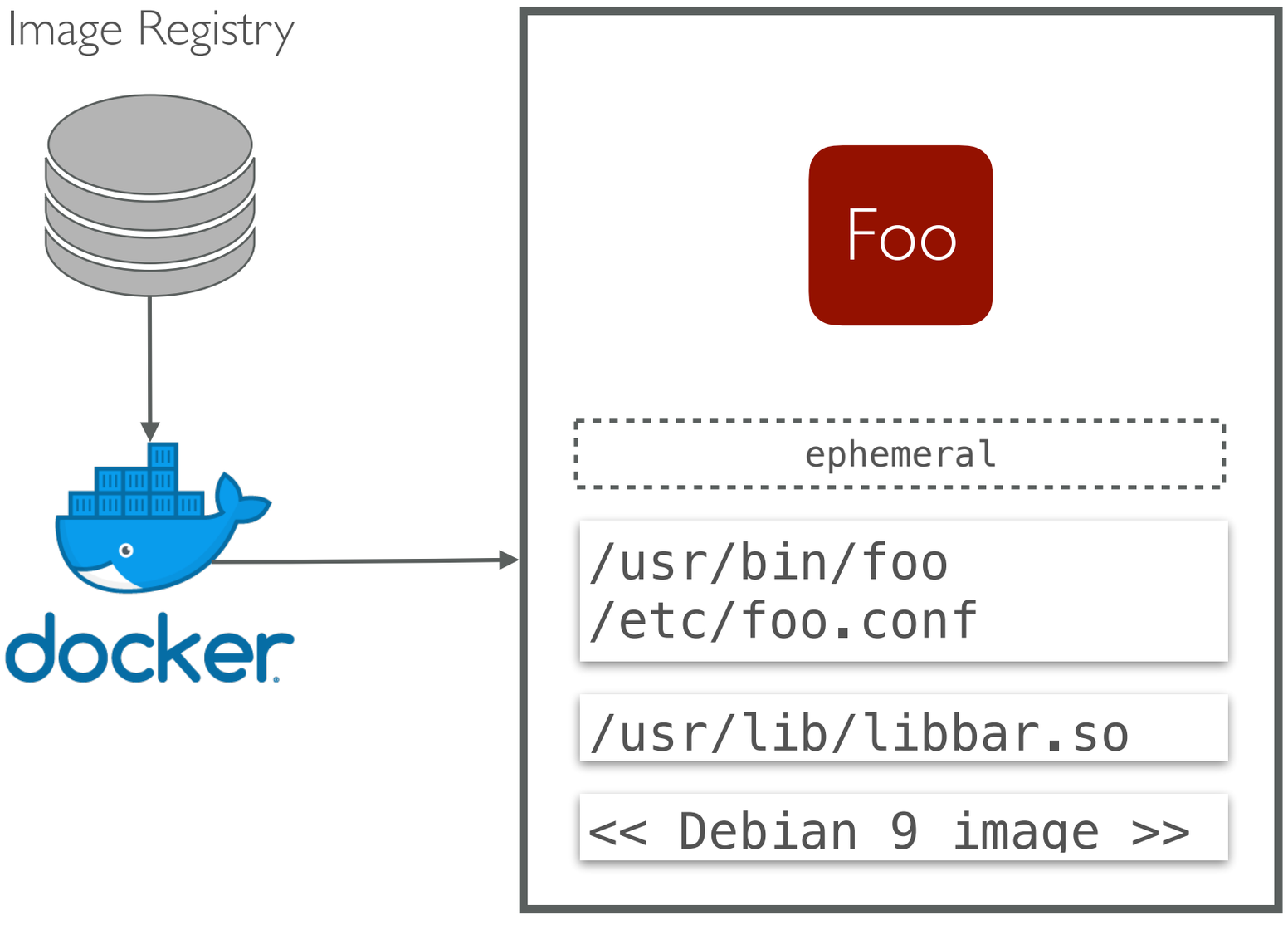
Anatomy



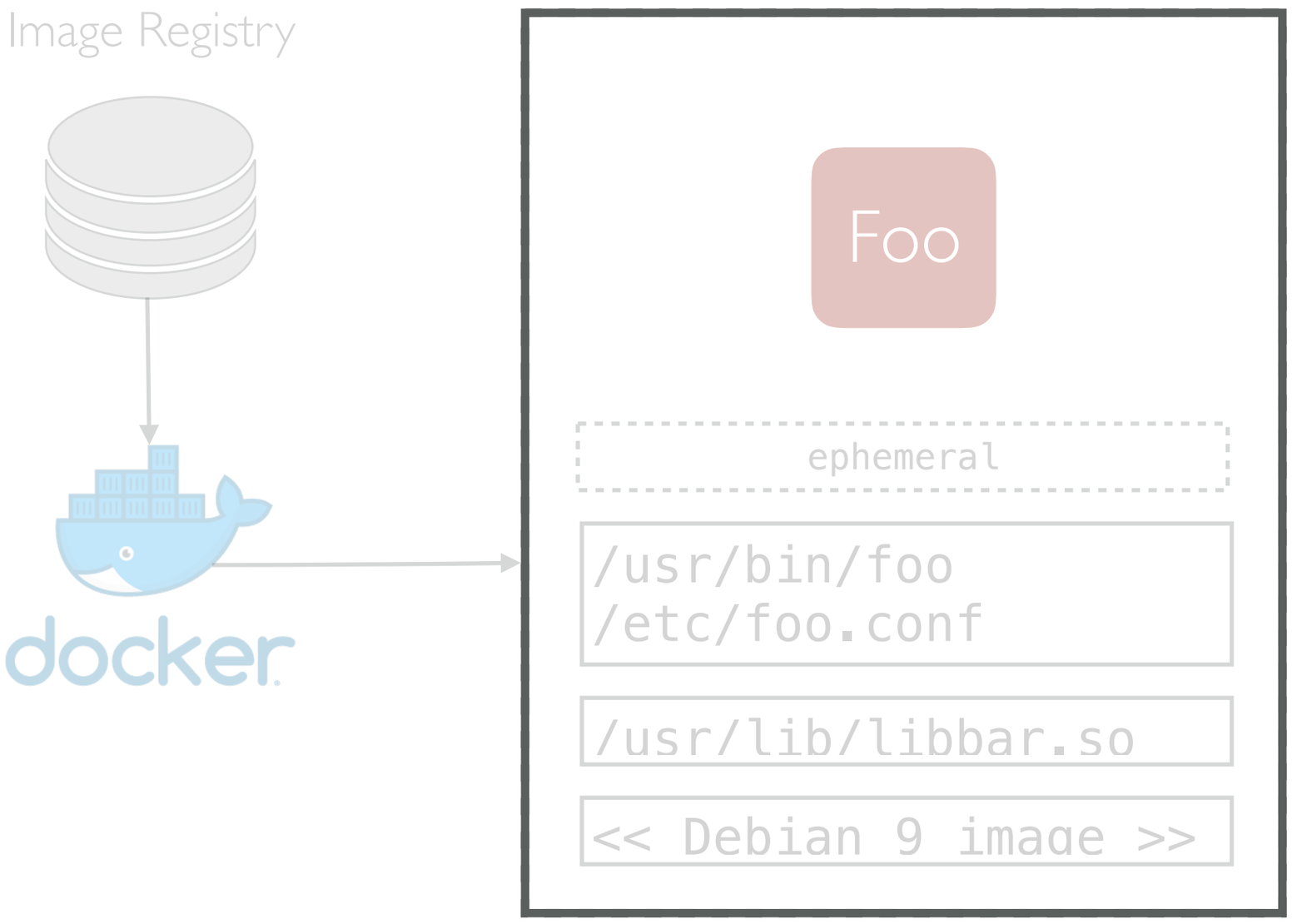
Anatomy



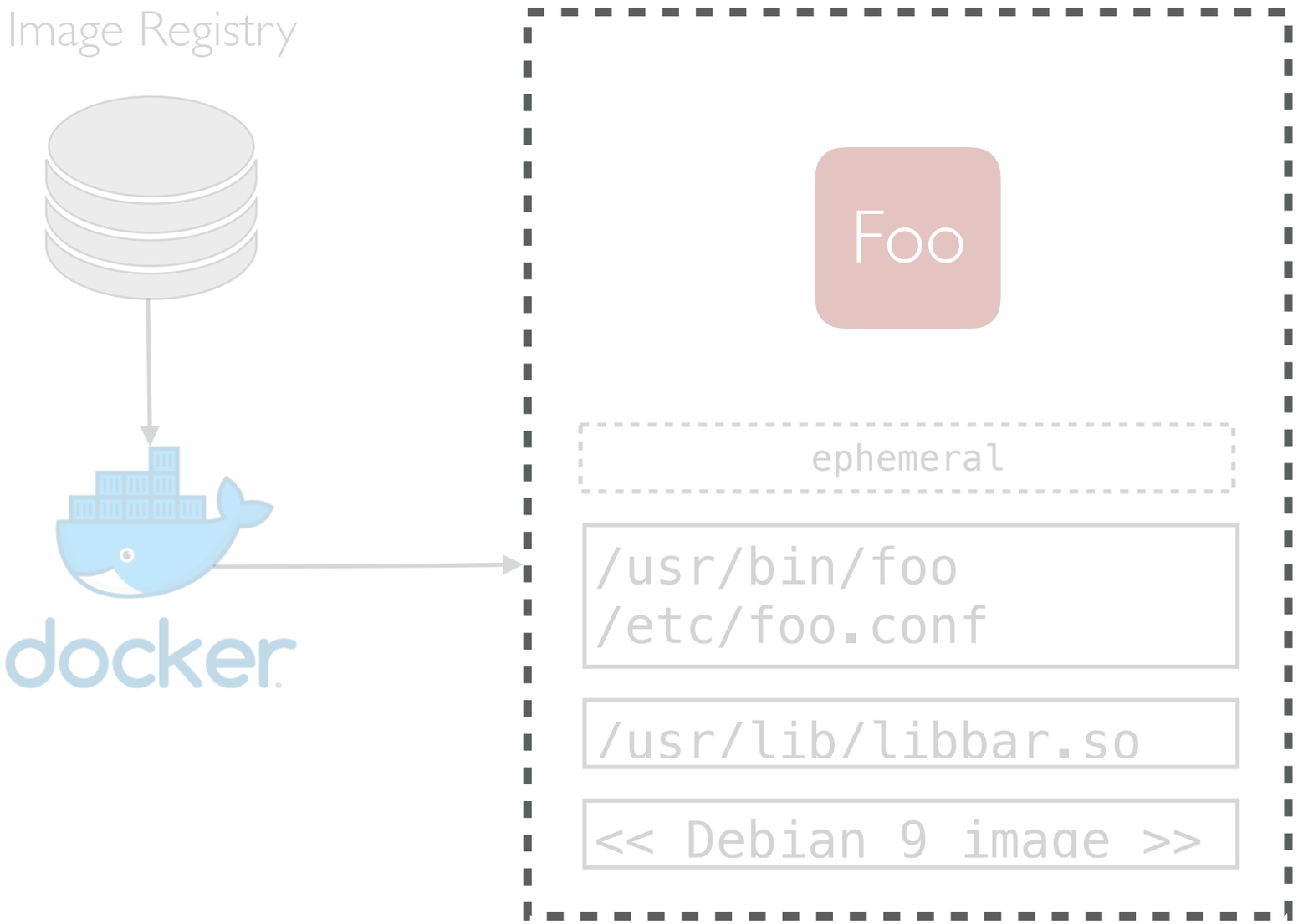
Anatomy



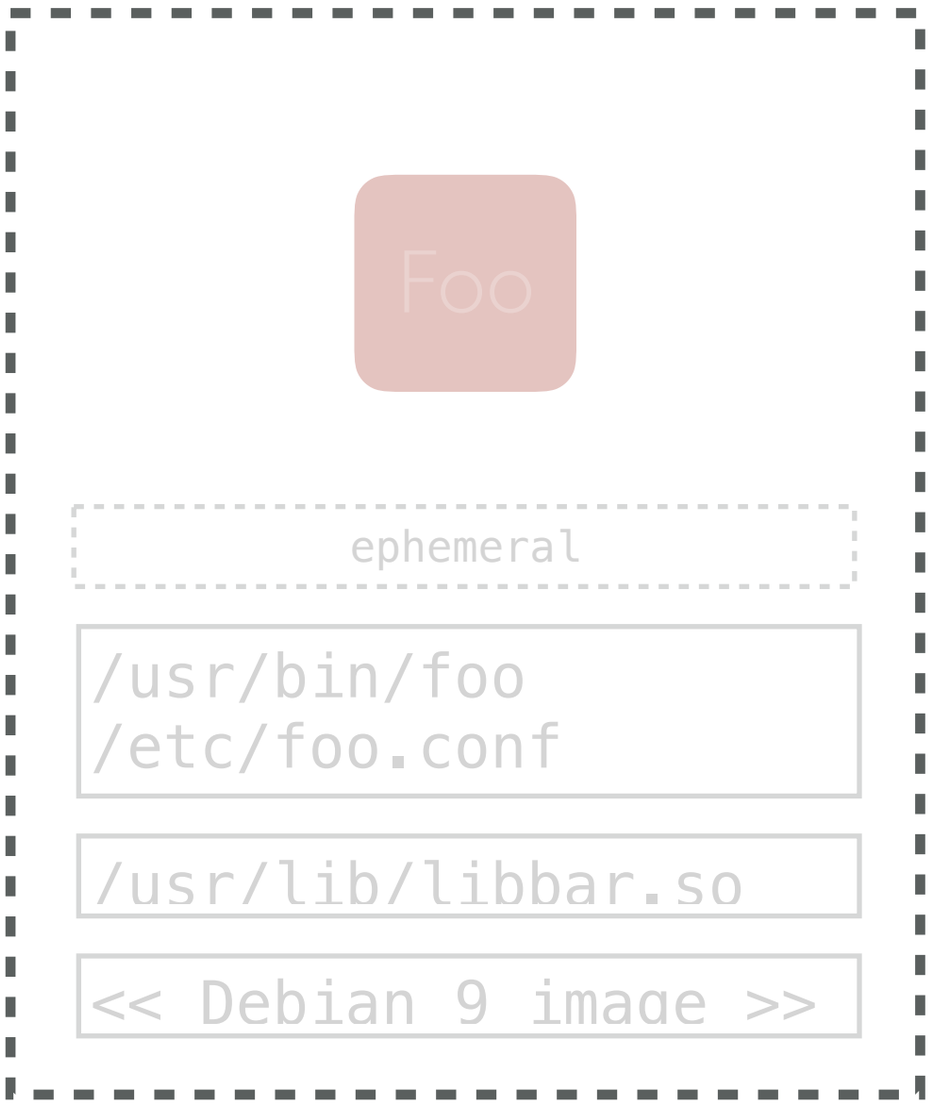
Kernel Escapes



Kernel Escapes



Kernel Escapes



- Kernel design is open by default
  - Therefore, trying to close holes
  - Not everything is namespaced
- Kernel bugs → Container escapes

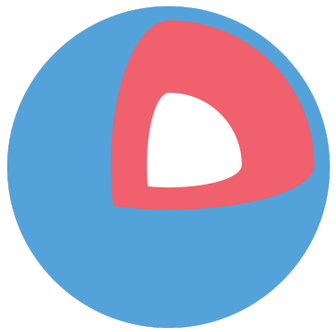
Mindset: Container Isolation

A container is **not** like a VM.

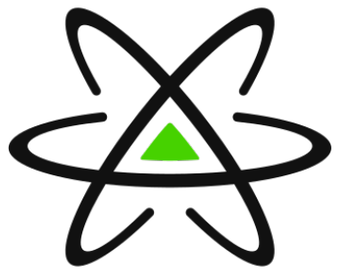
A container is like an **application package**
running in a *slightly leaky* sandbox.

Policy: Operating Systems

Use a **container-focused** host operating system.



CoreOS



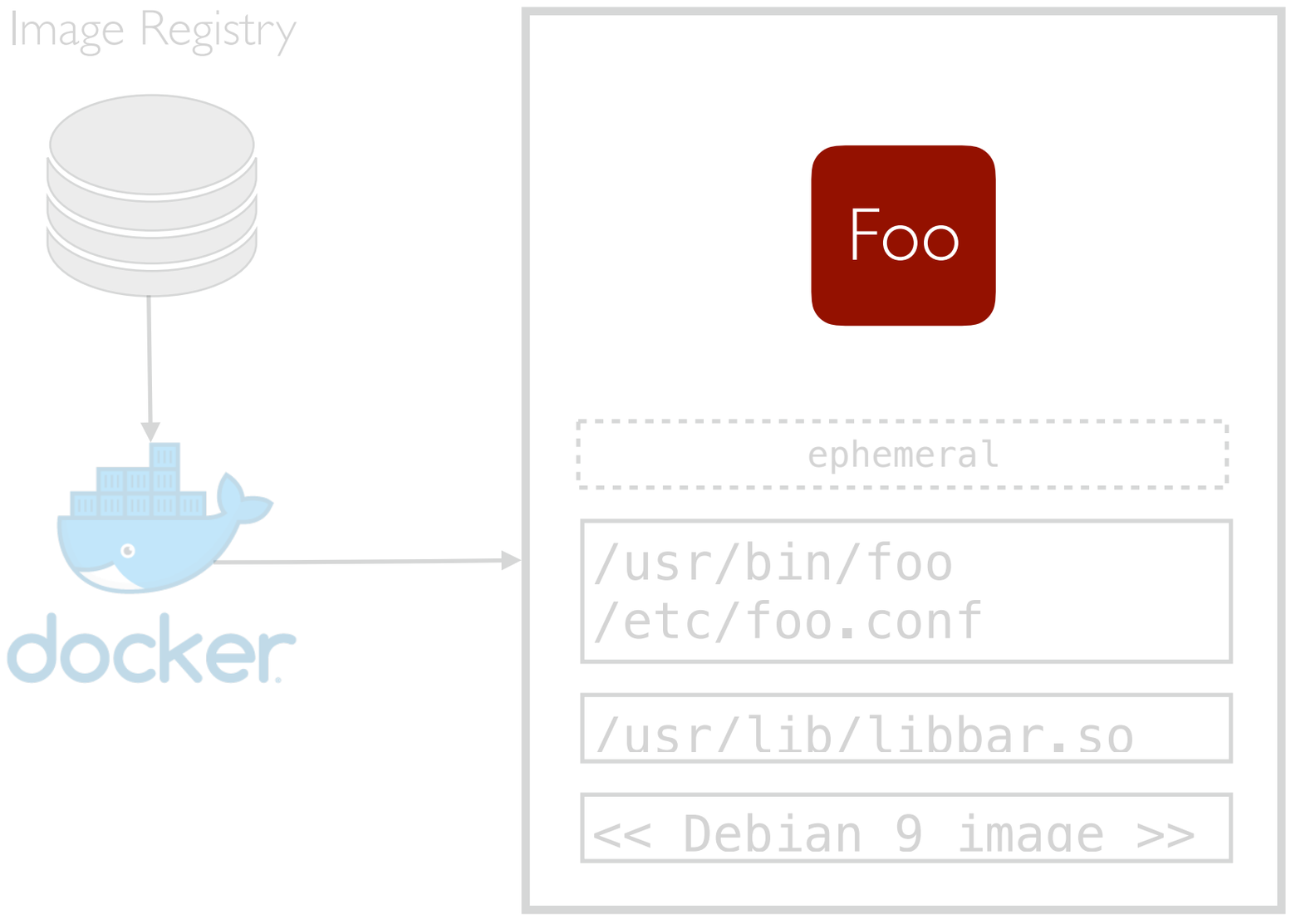
Project Atomic

Clear Linux

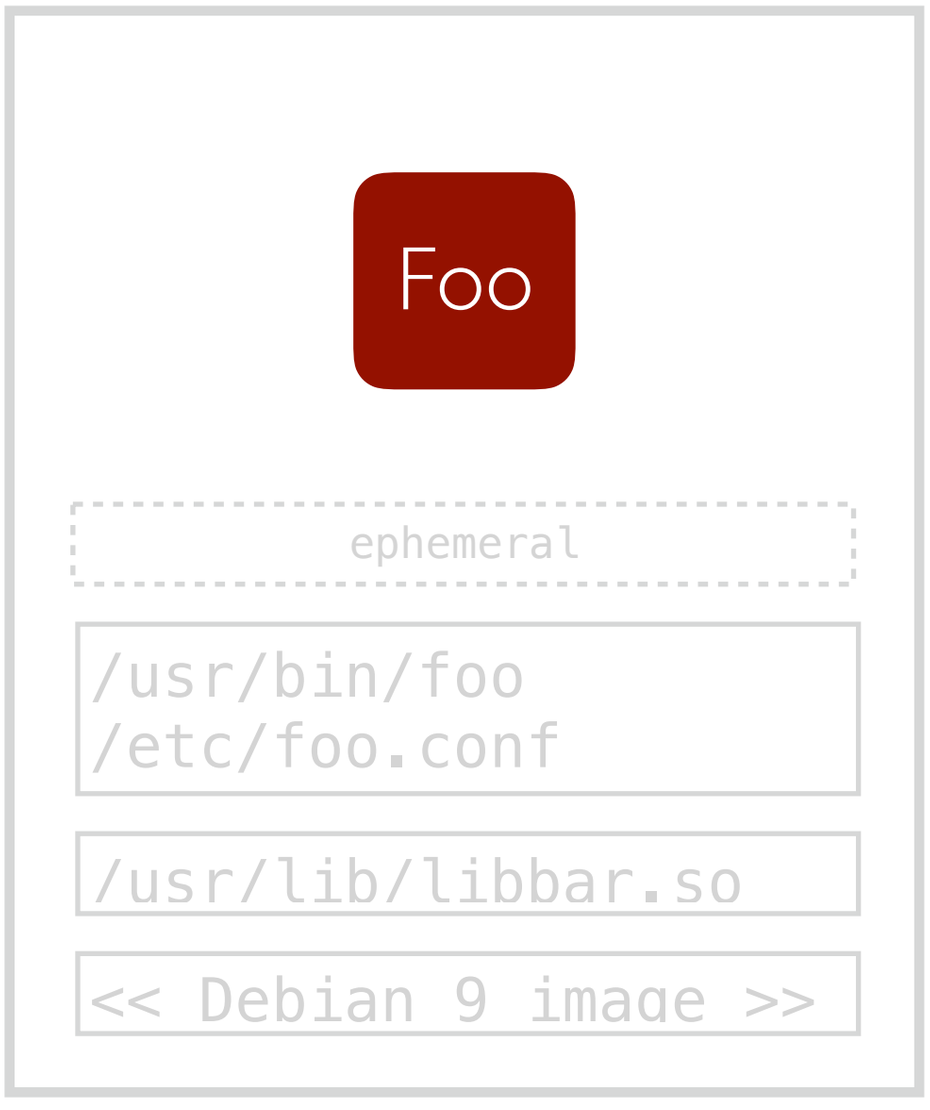
etc. etc.

(Google for more)

Over-Privileged Containers



Over-Privileged Containers



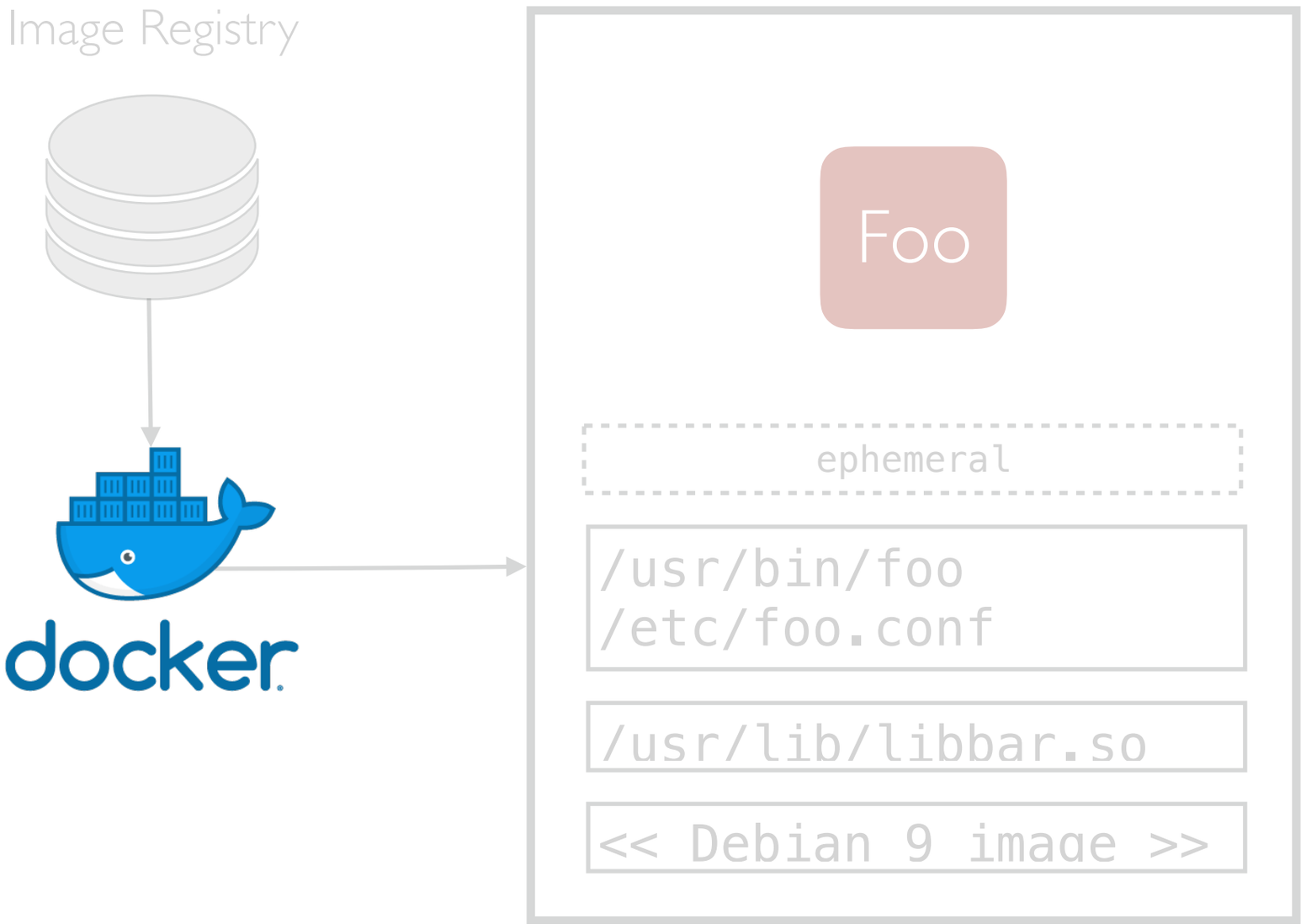
- The `--privileged` option
- Exposing the Docker socket
- Over-reaching volume mounts
- Unlimited resource allocation

Mindset: Least Privilege

Don't give a container what
you wouldn't give a regular application.

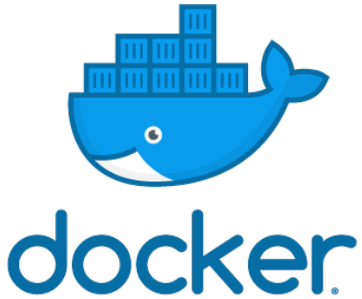
- Don't run as `root` inside the container
- Drop as many privileges as you can
- Minimize what volume mounts expose
- Set resource constraints

Daemon Dangers



Daemon Dangers

- Underlying socket API
  - No authorization required!
  - Can be exposed to the network
- Run a container ⇒ `root` on the host



Mindset: Least Privilege

Minimize access to the Docker daemon.

- Avoid exposing to the network
- Think of `docker run` like `sudo`

Image Distribution

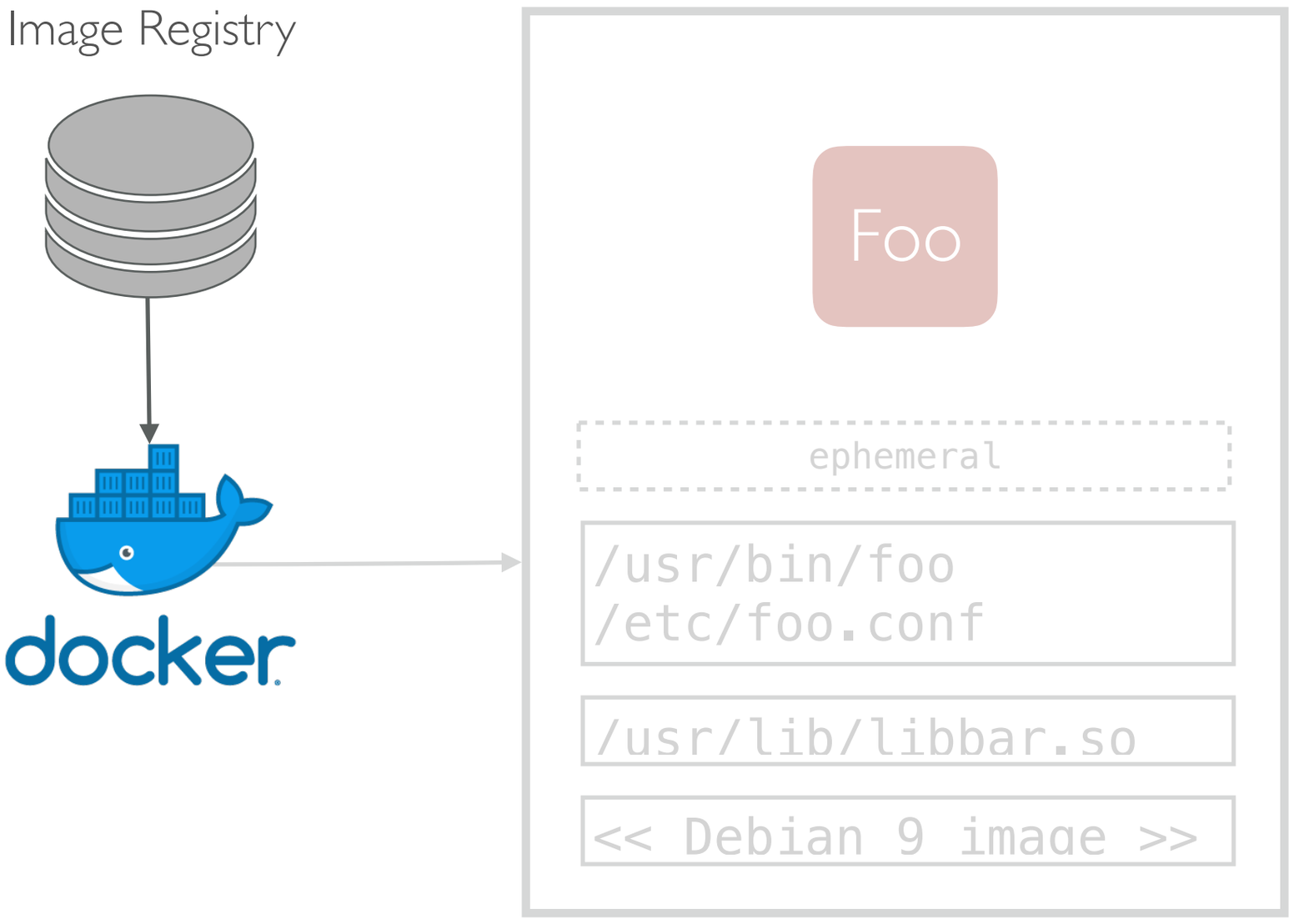
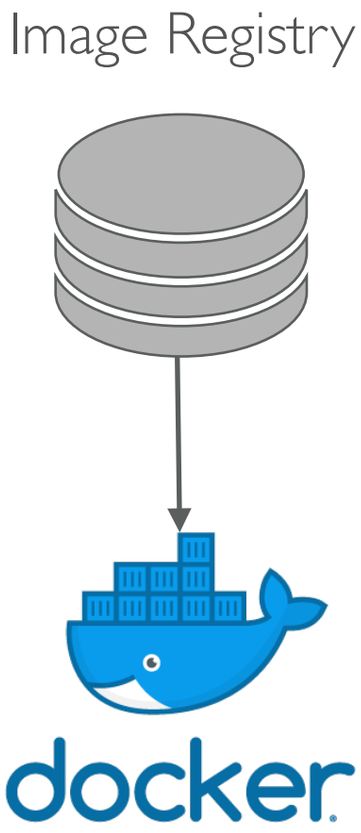


Image Distribution



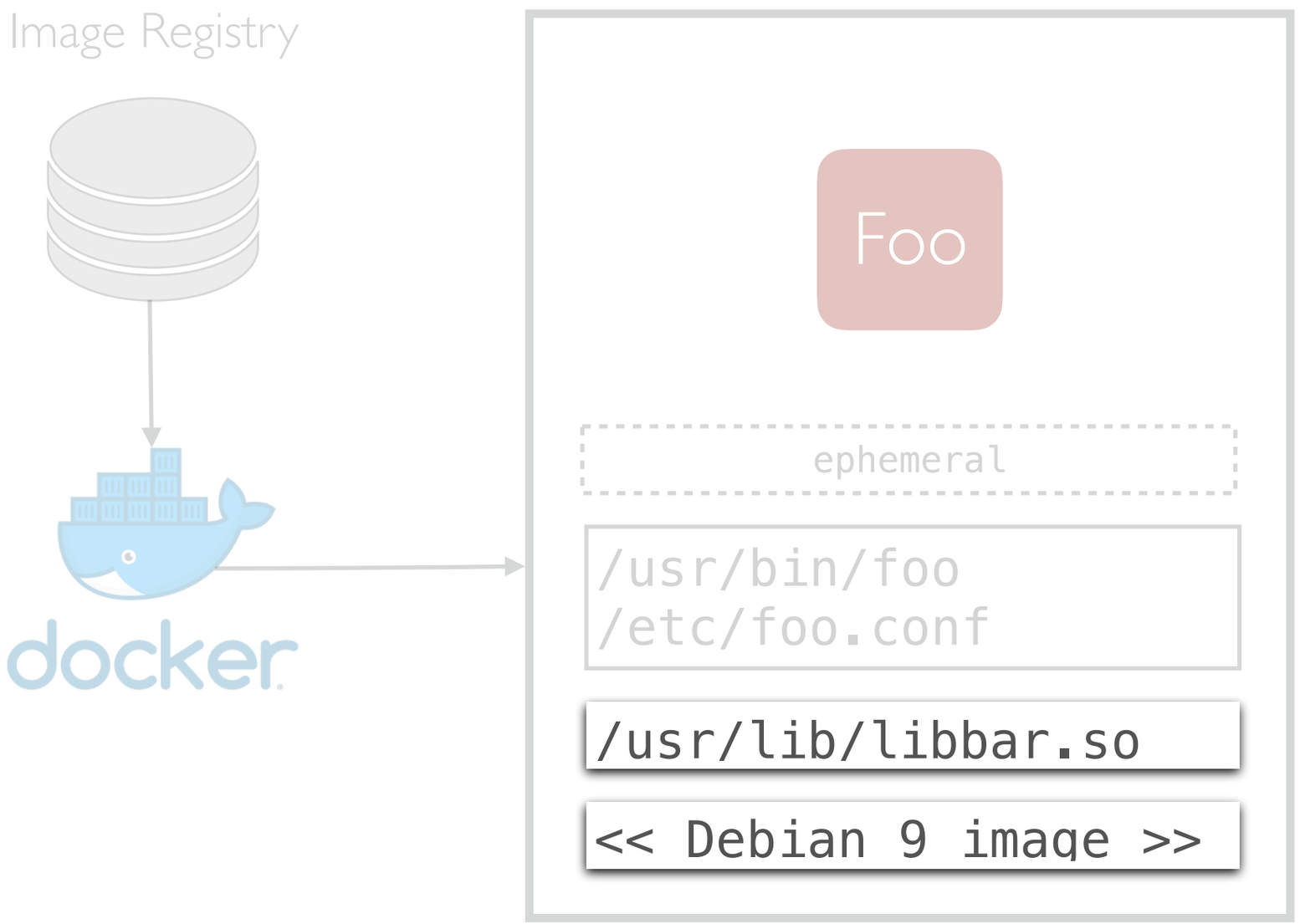
- Malicious source?
- Tampering in the registry?
- Tampering in transit?
- Tampering on the host system?

Policy: Signed Artifacts

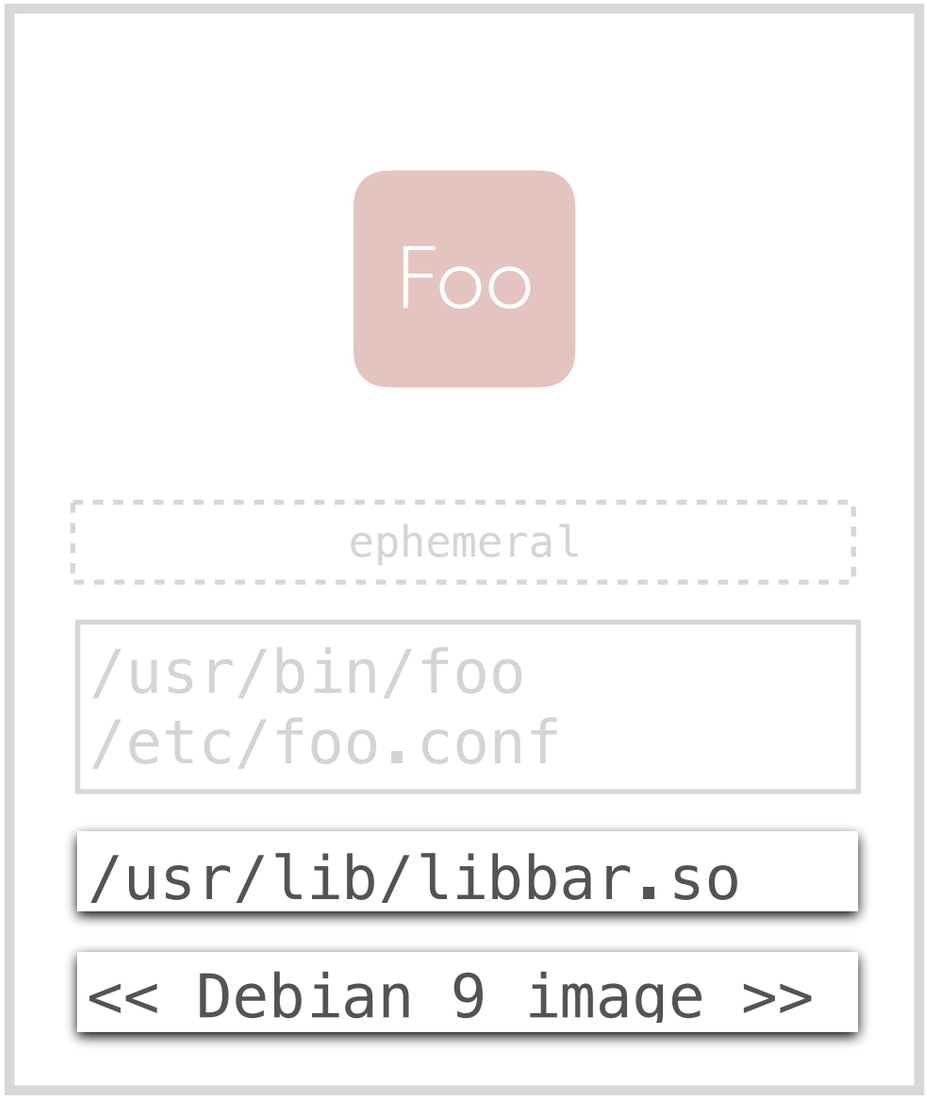
We already do this for regular software updates.

- Unfortunately, competing approaches
- Some implementations raise issues
  - Docker Content Trust uses ToFU
- Consider running an in-house registry

Vulnerable Dependencies



Vulnerable Dependencies



- In direct dependencies
- In the base image
- Cannot idiomatically patch at run-time

Mindset: Re-build Images

The old way was…

- Install a patch
- Restart the application

The new way is…

- Rebuild the image
- Re-deploy the container

Mindset: Inventory & Scans

Simpler to scan **images**, not hosts.

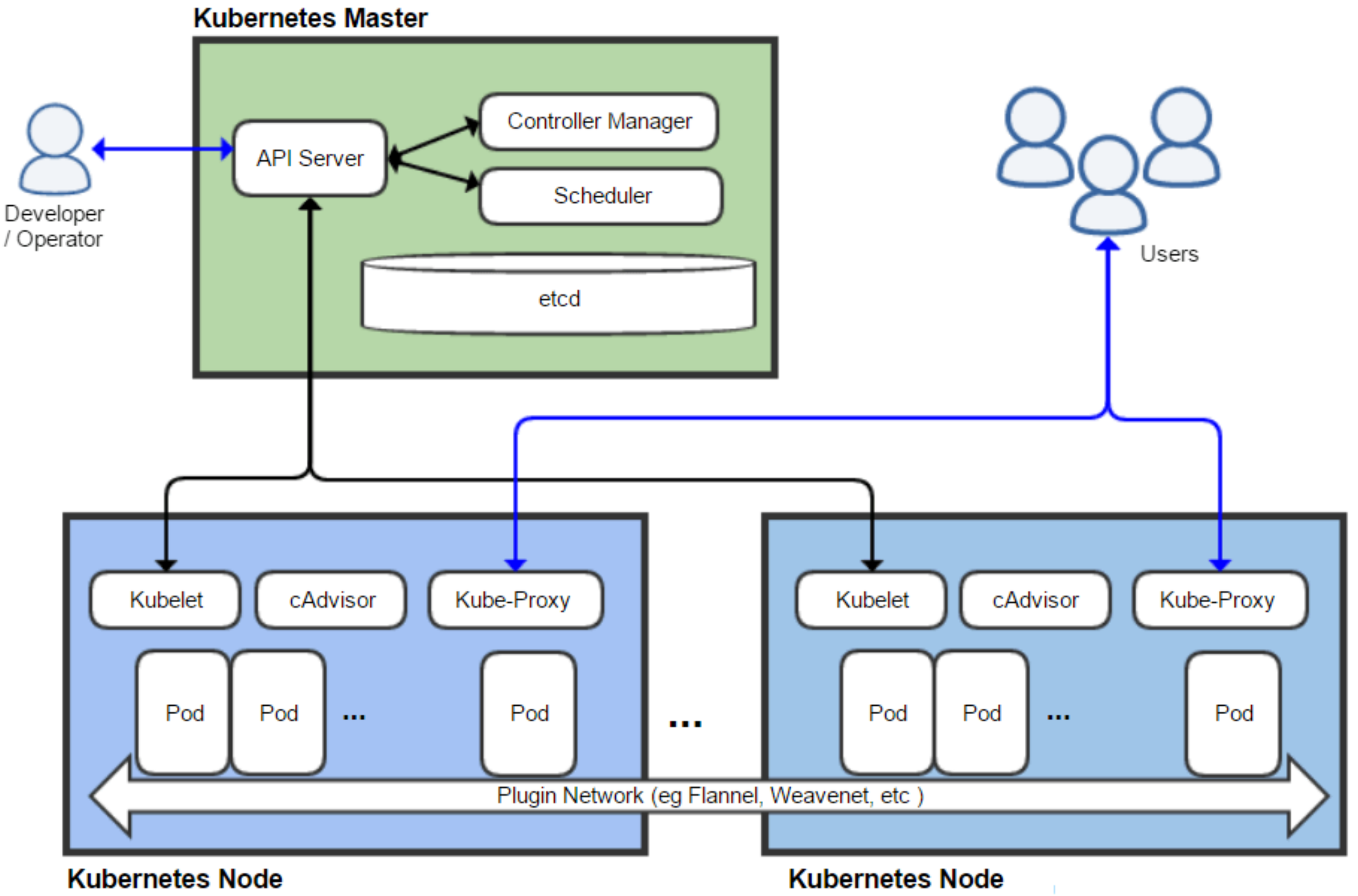
(Run a private image registry.)

Asset tracking of where images are running.

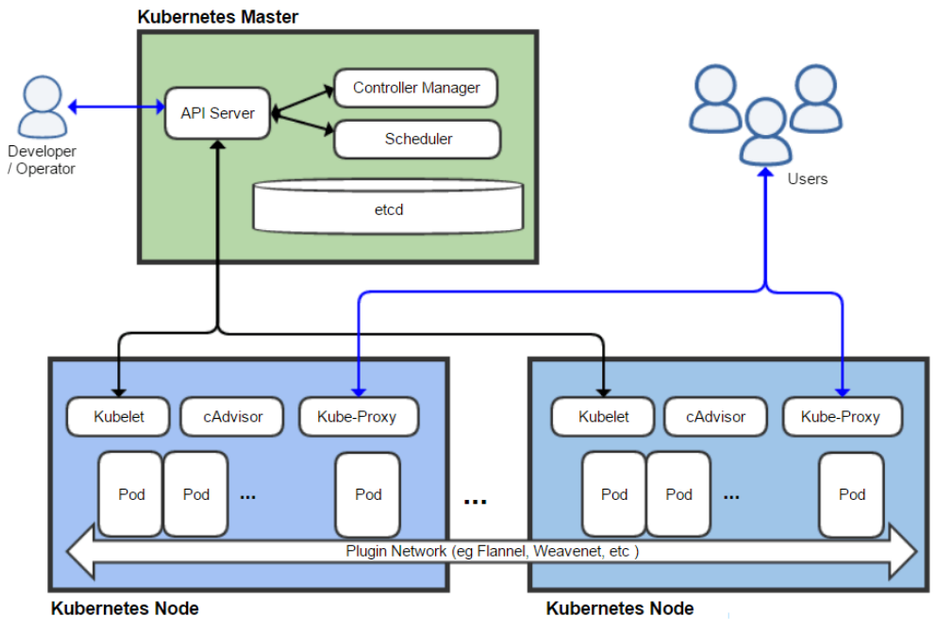
Summary

- *"Container's don't contain."*
  - They're not VM's
  - It helps if you use a container-focused host OS
- Don't give a container what you wouldn't give a regular host application
- Restrict access to the Docker daemon
- Verify image signatures / Have a plan for distribution
- Scan images for vulns & re-deploy from fresh images
  - Use your own private image registry

kubernetes



Hardening Basics



CC BY-SA: Wikipedia user Khtan66

- API Server: TLS + Auth + RBAC
- etcd: Mutual TLS auth
- kubelet: Disable anonymous auth

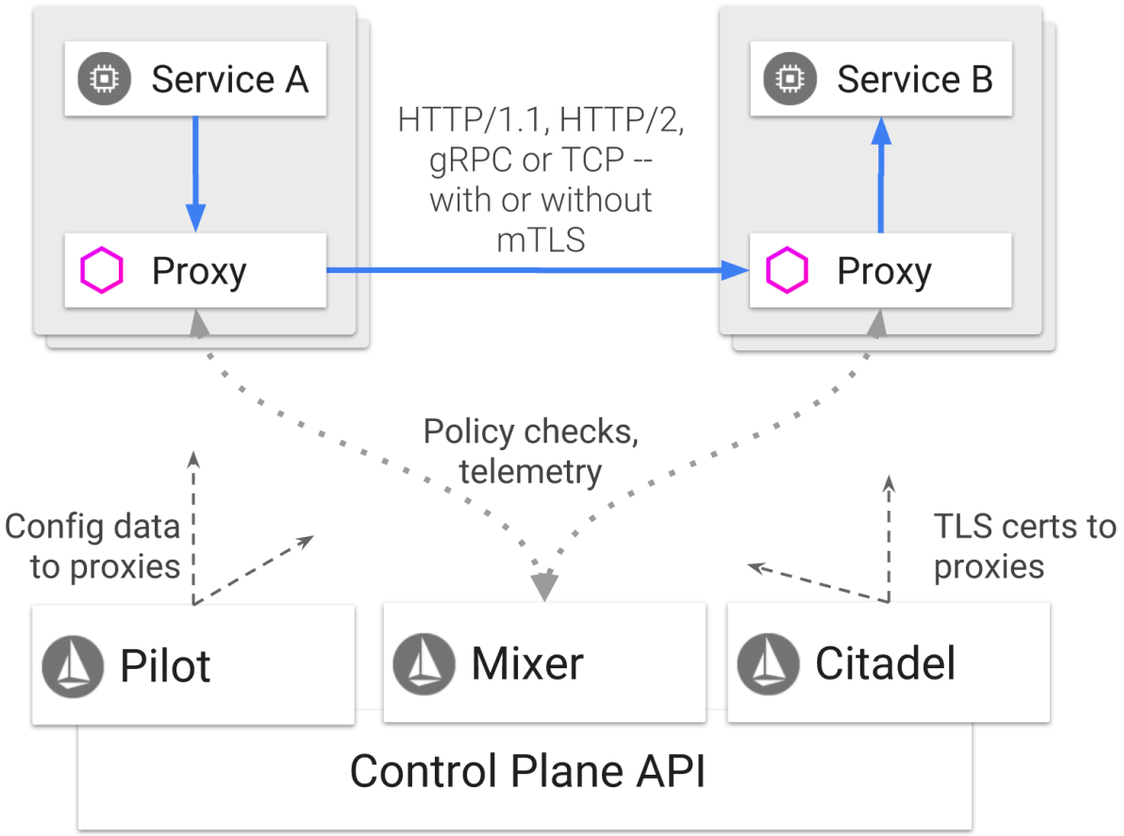
Read the docs: *"Securing a Cluster"*

Test your config: kube-bench

Didactic Security

Istio

Implications



Source: https://istio.io/docs/concepts/what-is-istio/

- Separates **data plane** from **control plane**
- Pro: Mutual TLS everywhere
- Con: Single point of failure

Cloud Native Landscape

2018-09-28T19:04:23Z b7f0fad

See the interactive landscape at l.cncf.io

Greyed logos are not open source

App Definition and Development

Database

CNCF Incubating: Vitess

CNCF Sandbox: TiKV

Streaming & Messaging

CNCF Incubating: NATS

CNCF Sandbox: cloudevents

Source Code Management

Application Definition & Image Build

CNCF Incubating: HELM

CNCF Sandbox: Telepresence

Continuous Integration & Delivery

Orchestration & Management

Scheduling & Orchestration

CNCF Graduated: kubernetes

Coordination & Service Discovery

CNCF Incubating: CoreDNS

Remote Procedure Call

CNCF Incubating: gRPC

Service Proxy

CNCF Incubating: envoy

API Gateway

Service Mesh

CNCF Incubating: LINKERD

Runtime

Cloud-Native Storage

CNCF Incubating: ROOK

Container Runtime

CNCF Incubating: containerd

CNCF Incubating: rkt

Cloud-Native Network

CNCF Incubating: CNI

Provisioning

Automation & Configuration

Container Registries

CNCF Sandbox: HARBOR

Security & Compliance

CNCF Incubating: Notary

CNCF Incubating: TUF

CNCF Sandbox: Open Policy Agent

Key Management

CNCF Sandbox: spiffe

CNCF Sandbox: SPIRE

Cloud

Public

Platform

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Certified Kubernetes - Installer

Non-Certified Kubernetes

PaaS/Container Service

Observability and Analysis

Monitoring

CNCF Graduated: Prometheus

CNCF Sandbox: cortex

CNCF Sandbox: OpenMetrics

Logging

CNCF Incubating: fluentd

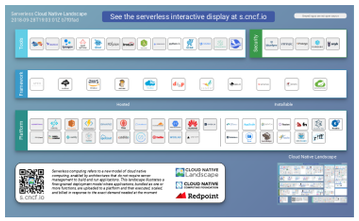
Tracing

CNCF Incubating: Jaeger

CNCF Incubating: OpenTracing

Chaos Engineering

Serverless



This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path

l.cncf.io

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint Amplify PARTNERS

Special

Kubernetes Certified Service Provider

Kubernetes Training Partner

DIDACTIC SECURITY

Questions?