Container Fundamentals and Introducing Kubernetes

Anthony E. Nocentino
aen@centinosystems.com

Course Overview

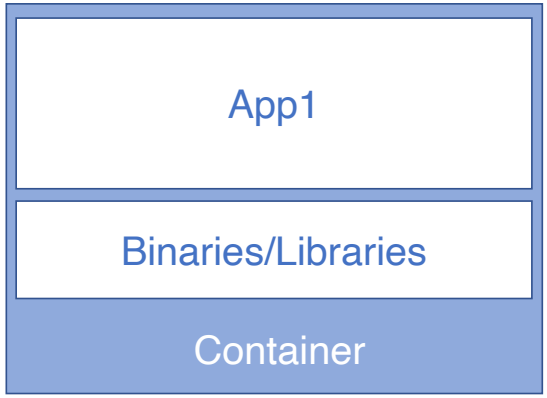
- **Module 0** - Introduction
- **Module 1** - Container Fundamentals and Introducing Kubernetes
- **Module 2** - Kubernetes Architecture and API Objects
- **Lunch @ 12:00-12:45**
- **Module 3** - Interacting With Your Cluster
- **Module 4** - Deploying Applications in Kubernetes
- **Module 5** - Building and Deploying Container-based Applications in Kubernetes

Agenda

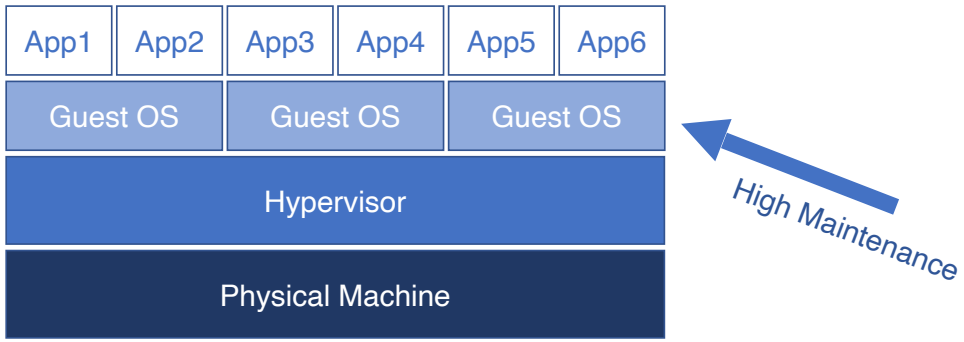
- **Container and Linux Fundamentals**
  - Container Fundamentals
  - Container Based Application Deployment
  - The Need for Container Orchestrators
- **Introducing Kubernetes and its Architecture**
  - What is Kubernetes

Container Fundamentals

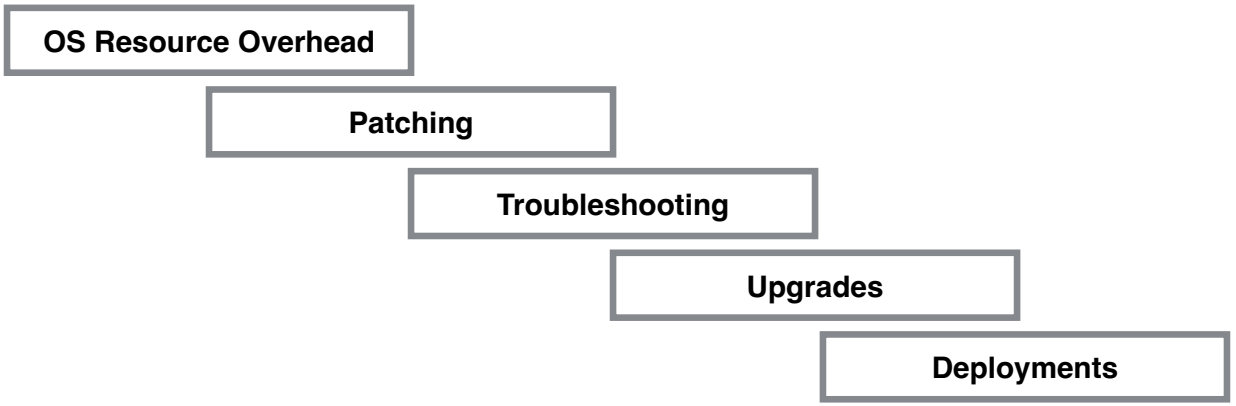
- Operating system virtualization
  - Shared kernel and system resources
- Container…contain…
  - Binaries, libraries and file system
- One app inside the container
  - This is the unit of work
- Containers are ephemeral
  - Let's start off with a comparison…



Virtual Machines

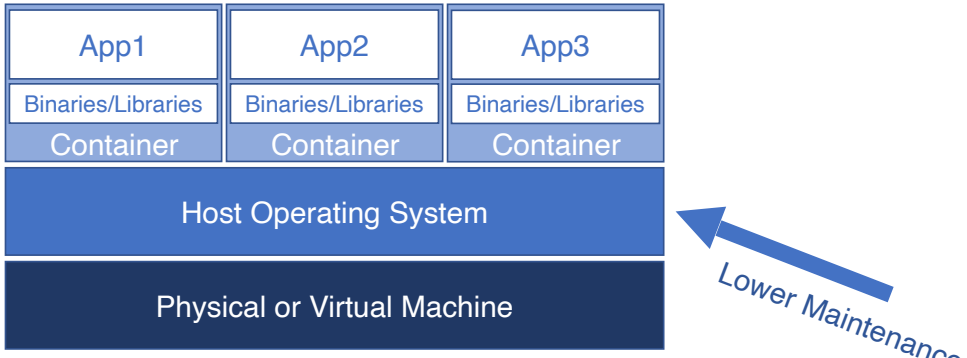


What's so Hard About Virtual Machines?

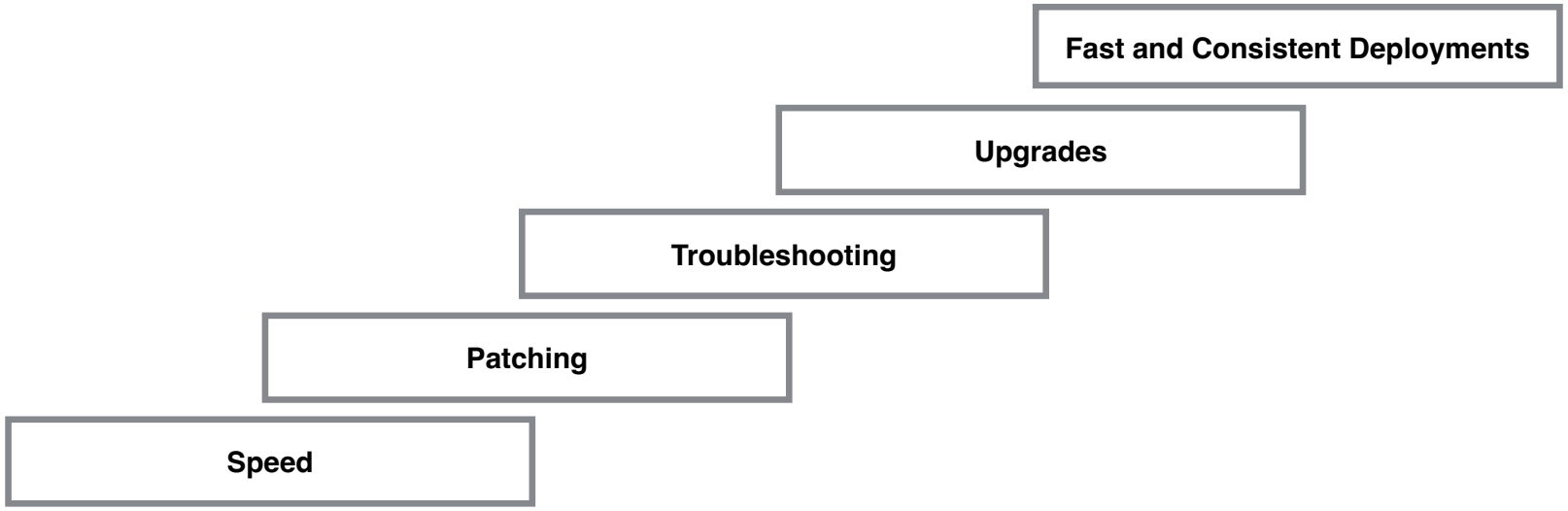


Does any of this move your business forward?

Containers

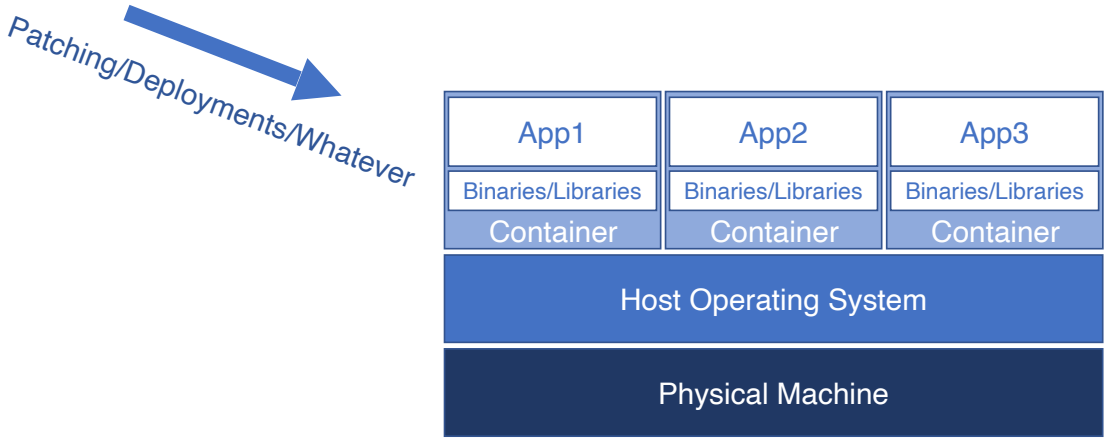

App1
App2
App3
Binaries/Libraries
Binaries/Libraries
Binaries/Libraries
Container
Container
Container
Host Operating System
Physical or Virtual Machine
Lower Maintenance

What do Containers Bring to the Table?



Services, we care about getting work done!

Containers

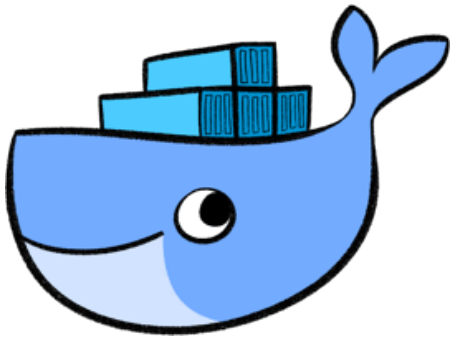

Patching/Deployments/Whatever
App1
App2
App3
Binaries/Libraries
Binaries/Libraries
Binaries/Libraries
Container
Container
Container
Host Operating System
Physical Machine

Containerizing Apps and Data Centers

- Reducing development time
- Deployment automation – speed and consistency
- Enables DevOps and CI/CD scenarios
- Orchestration
- Rethink how you deploy - it’s the service, not the server

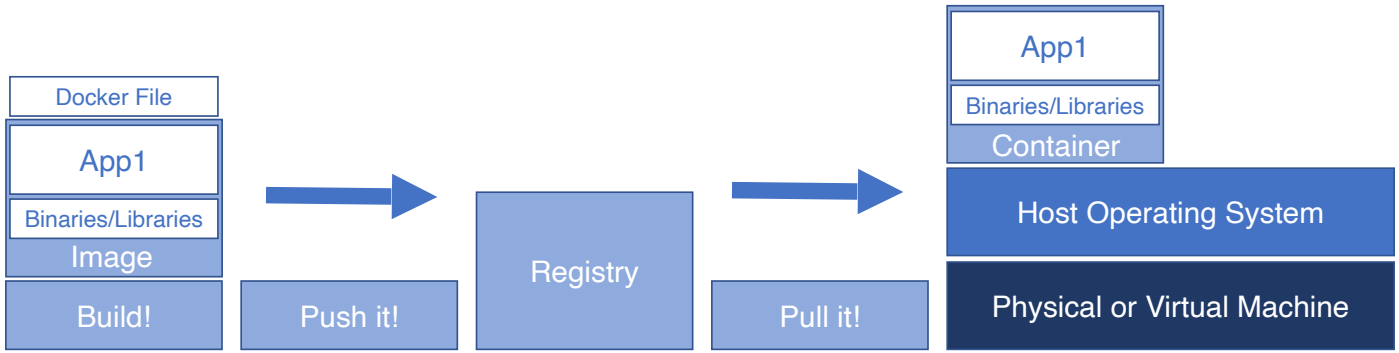
The Container Universe

- Docker
  - Linux
  - Windows
  - Mac
- Docker Inc.
- Other Container Runtimes
  - containerd
  - CoreOS
  - Windows
  - chroot…chwhat?



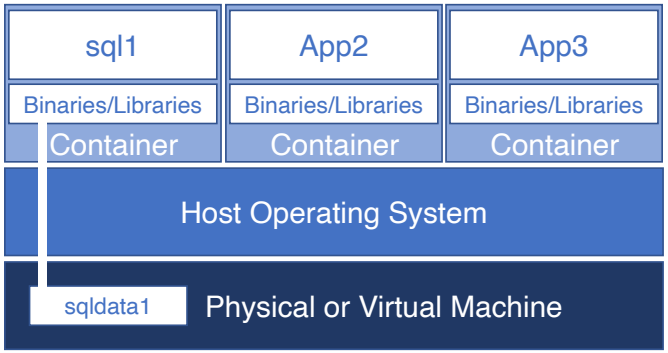
Getting Containers

- **Images** – code, runtimes, libraries, environment variables
- **Registries** – where images live. Docker Hub, Azure Container Registry, internal
- **Docker Files** – defines the container image



Data Persistency in Containers

- If your container is alive so is your data, don't delete the container
- Docker Data Volumes
  - Docker managed resource
  - Independent of the container
- https://docs.docker.com/storage/



Running SQL Server in Containers

- Why run SQL Server on a Container?
- Same reasons…
  - Deployments, upgrades, patching, speed…agility
  - What if the unit of persistency IS the database…NOT the Server!
- Only Linux is available
- Windows is no longer available
- Active Directory authentication available now

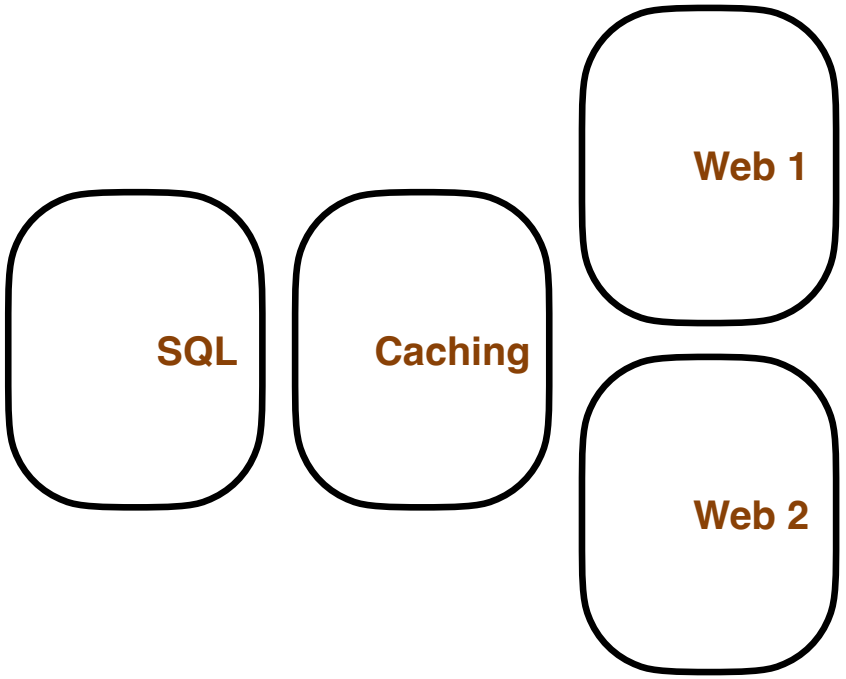
Hands on lab time…

- Pull an Image
- Run a Container
- Access our application
- Connect to the Container
- Persisting data with a Container

Let's Move on…

- **Introducing Kubernetes and its Architecture**

Modern Application Deployment



- Where do I run the application?
- How do I scale the application?
- How do I consistently deploy?
- How do I or my applications access the services?

What is Kubernetes?

- Container Orchestrator
- Infrastructure Abstraction
- Desired State

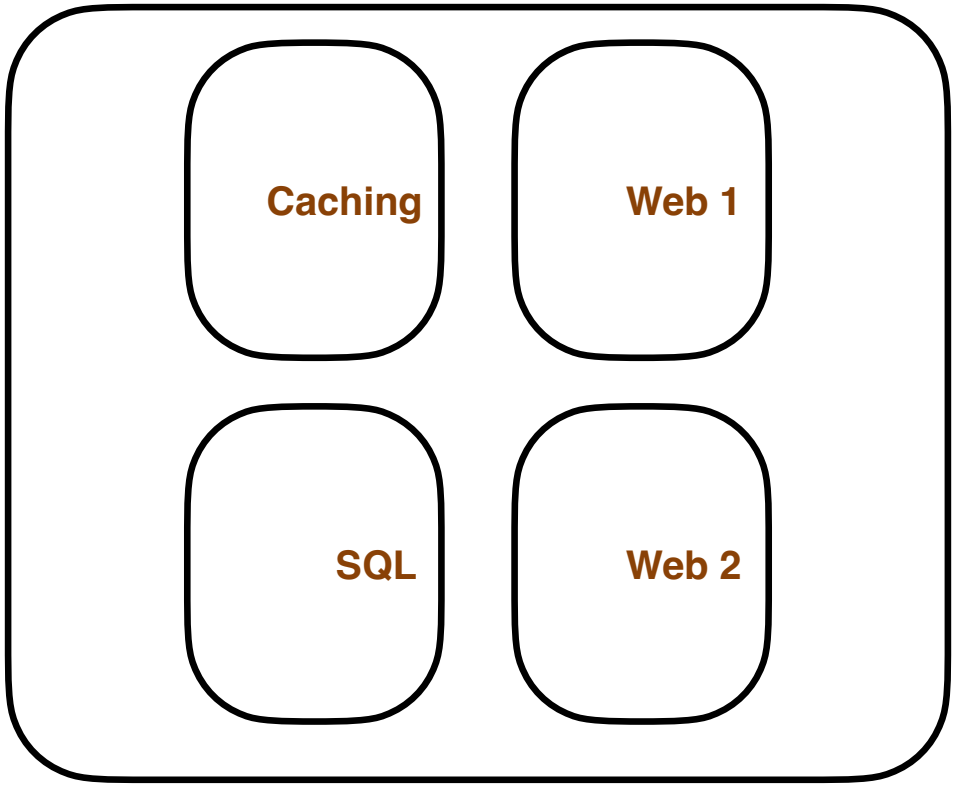
Kubernetes Principles

- Desired State
- Declarative Configuration
- Controllers/Control Loops
- The API Server

Kubernetes Benefits

- Workload placement
- Managing state, starting things up and keeping things up
- Networking and Services
- Load balancing services
- Persistent storage
- Declarative model

Kubernetes Cluster



Review

- **Container and Linux Fundamentals**
  - Container Fundamentals
  - Container Based Application Deployment
  - The Need for Container Orchestrators
- **Introducing Kubernetes and its Architecture**
  - What is Kubernetes