Practical Container Scenarios in Azure

Anthony E. Nocentino
aen@centinosystems.com

Anthony E. Nocentino

- **Consultant and Trainer**
- **Founder and President of Centino Systems**
  - Specialize in system architecture and performance
  - Masters Computer Science
  - Microsoft MVP - Data Platform - 2017 - 2018
  - Linux Foundation Certified Engineer
  - Friend of Redgate - 2015-2019
- **email**: aen@centinosystems.com
- **Twitter**: @nocentino
- **Blog**: www.centinosystems.com/blog
- **Pluralsight Author**: www.pluralsight.com

Centino systems

Agenda

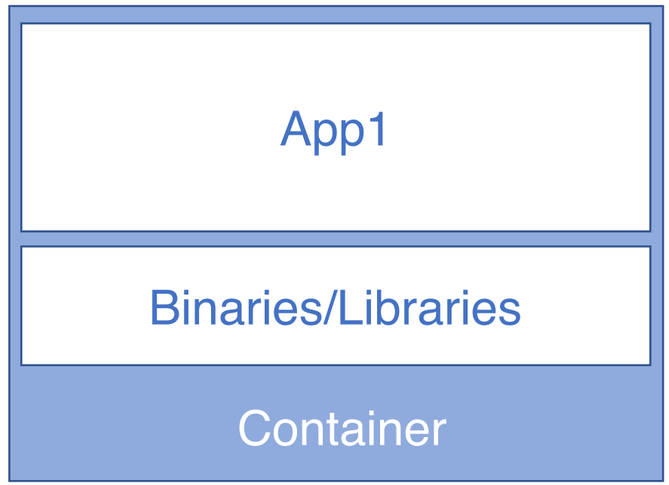
- **Container Fundamentals**
- **Creating a Container Image**
- **Working with Azure Container Registry**
- **Deploying our Application in Azure Kubernetes Service**

Containerizing Apps and Data Centers

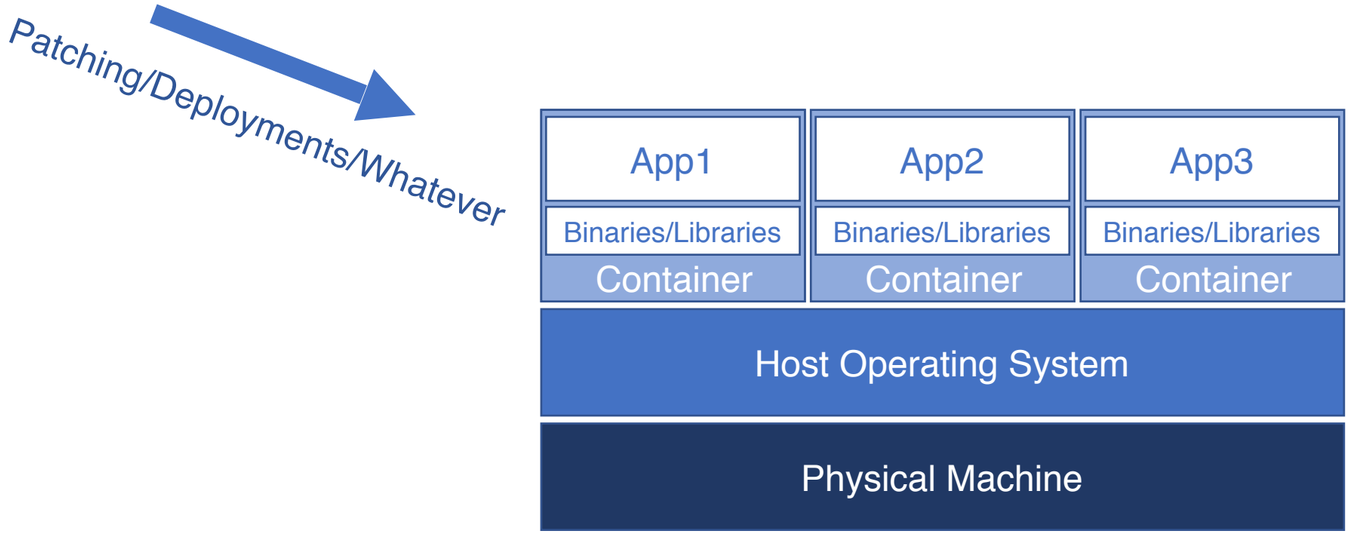
- Reducing development time
- Deployment automation – speed and consistency
- Enables DevOps and CI/CD scenarios
- Rethink how you deploy - it’s the application service, not the server

Container Fundamentals

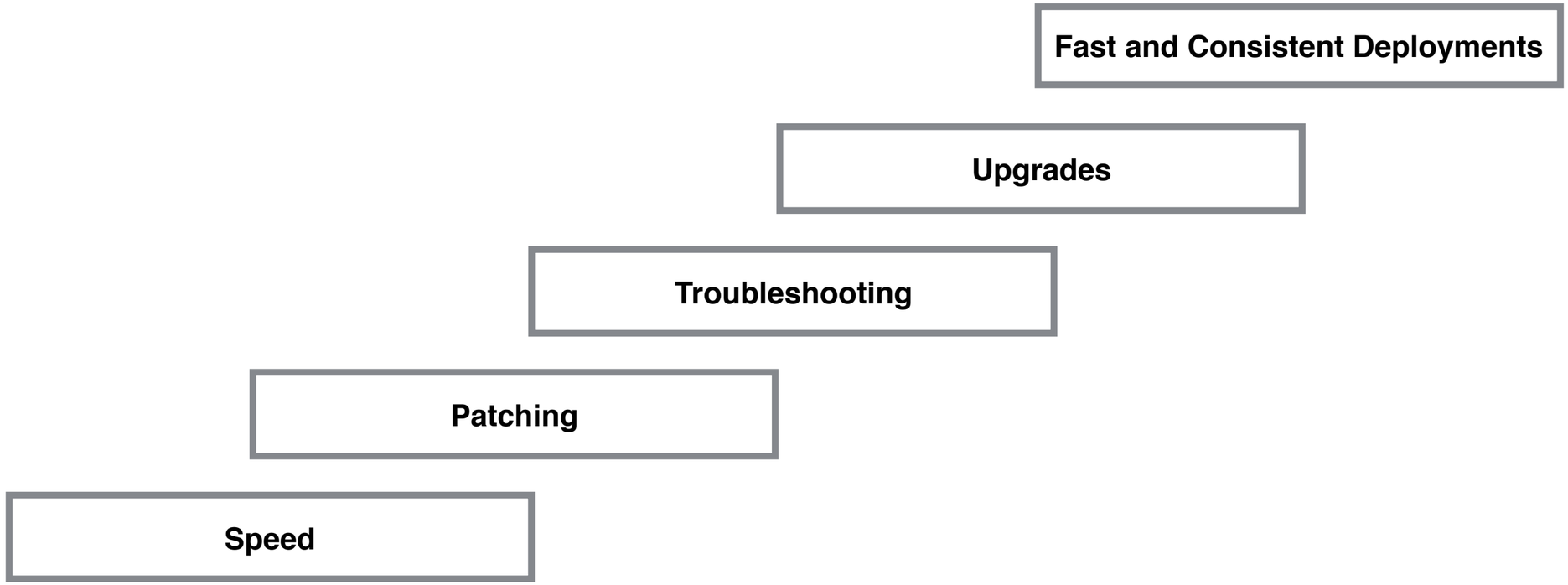
- Operating system virtualization
  - Shared kernel and system resources
- Container…contain…
  - Binaries, libraries and file system
- One app inside the container
  - This is the unit of work
- Containers are ephemeral



Containers


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

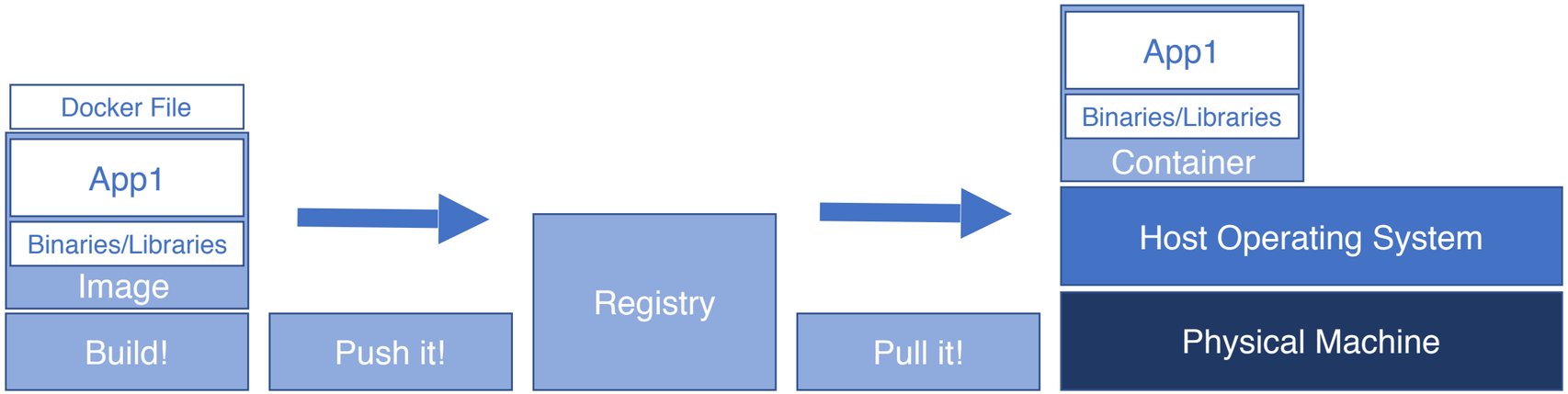
What do Containers Bring to the Table?



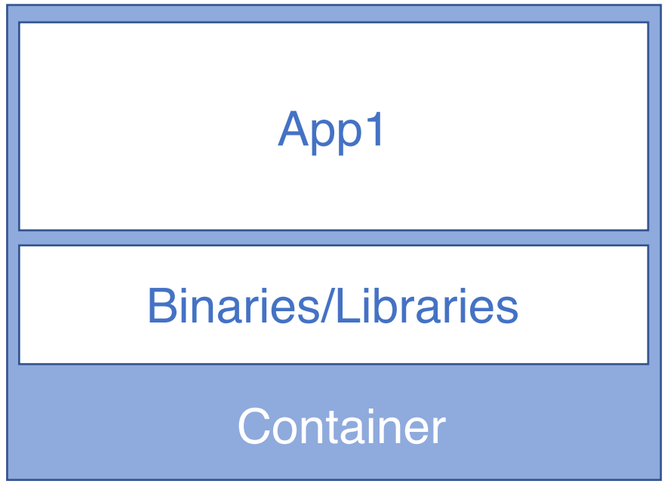
Services, we care about getting work done!

Getting/Creating Containers

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



Docker Files



- Describes the commands to build an **image**

```
FROM mcr.microsoft.com/dotnet/core/aspnet:2.2
COPY ./myWebApp/bin/Release/netcoreapp2.2/publish app/
ENTRYPOINT ["dotnet", "app/myWebApp.dll"]
EXPOSE 80
```

```
docker build -t mywebappimage .
```

https://docs.docker.com/engine/reference/builder/

Container Registries

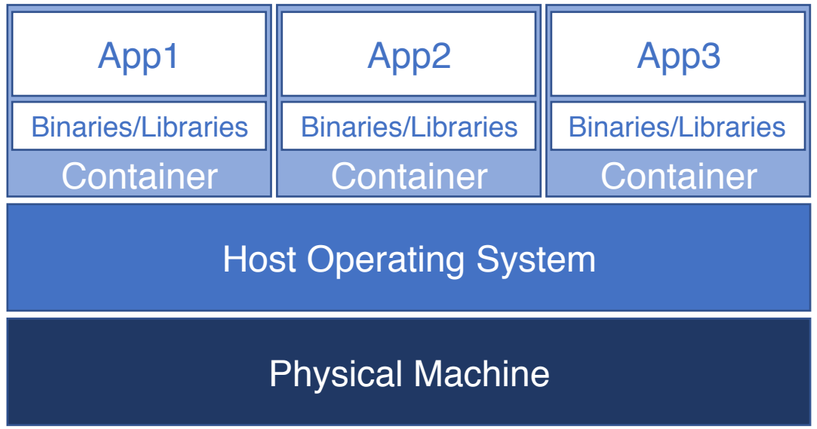
- Store container images
- Public or private
- Secured
  - Transport - HTTPS
  - Image digests - hash of image
- Key component of building a CI/CD pipeline
- Images are organized by tags
- Docker Hub
- Azure Container Registry
  - mcr.microsoft.com

Demo!

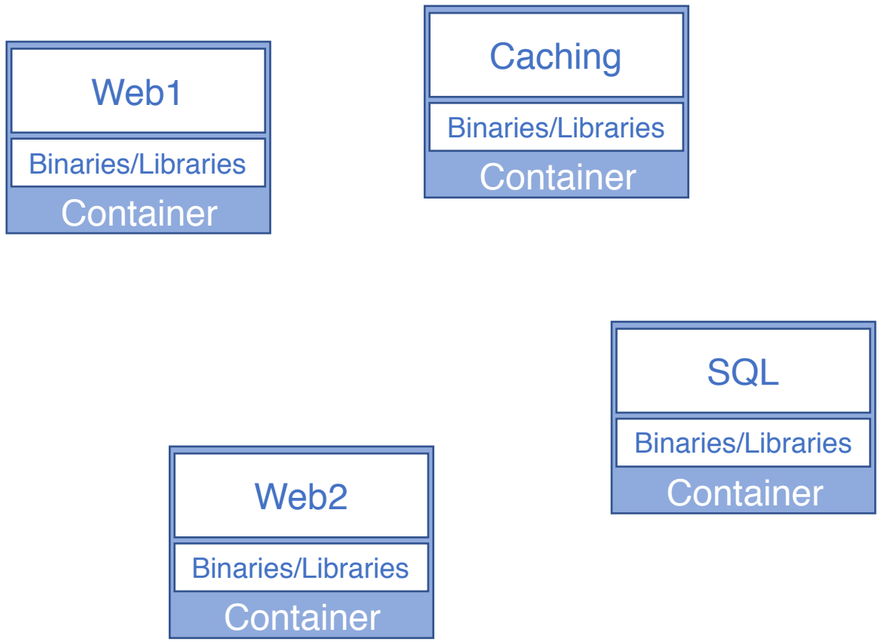
- Creating a container based application
- Building it in Azure Container Registry

Container Based Application Deployment

- Single-tier applications - anything written by IBM
- Multi-tier applications - Service oriented, Client/Server…
- Micro-services - smaller, more easily changed units



Modern Application Deployment



- Where do I run the application?
- How do I scale the application?
- How do I consistently deploy?
- How do I provide consistent services in a loosely connected system?

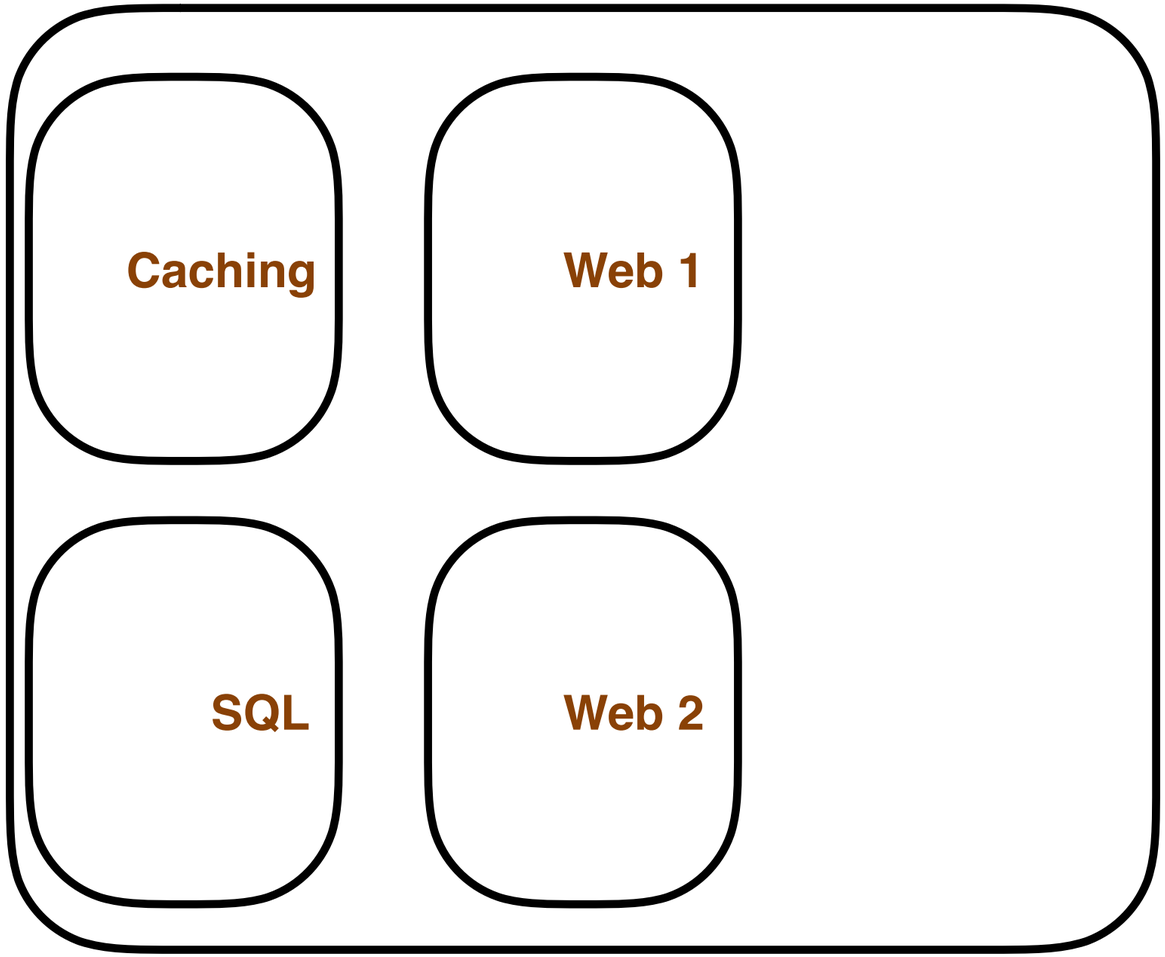
Container Orchestration

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

Container Orchestrators

- Docker Swarm
- Kubernetes
- Red Hat OpenShift
- Managed Services
  - Azure Kubernetes Services (AKS)
  - Google Kubernetes Engine (GKE)
  - Amazon Elastic Container Service for Kubernetes (EKS)

Kubernetes Cluster



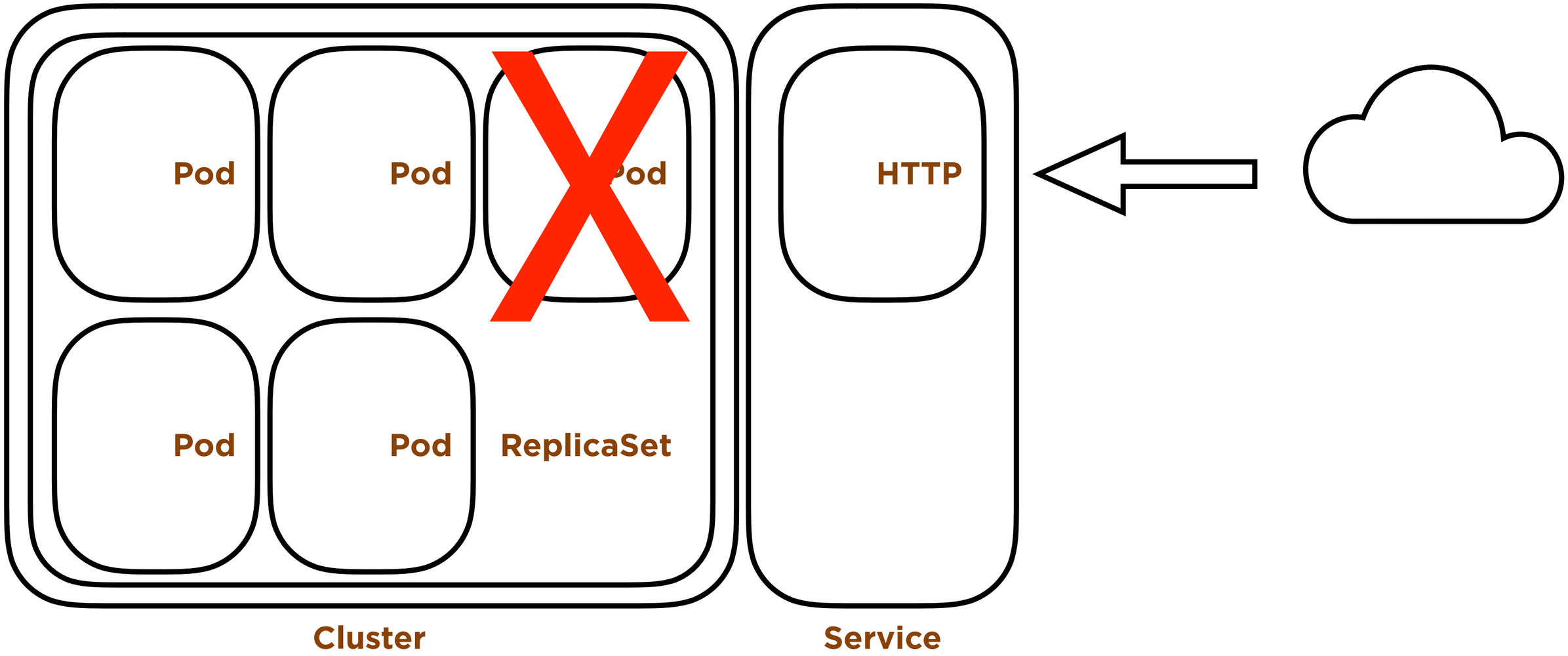
Azure Kubernetes Service

- Managed Cluster
- Upgrades handled in Azure (CLI/Portal)
- Define a number of Nodes (Agents)
- Nodes are in Availability Sets

Kubernetes API

- **API Objects** - Represent resources in your system
  - Really an API to the resources in your cluster…
  - **Pods** - your container based applications
  - **Controllers** - maintain desired state
  - **Services** - persistent access to your apps
  - **Storage** - persistent storage for your data
  - …and more

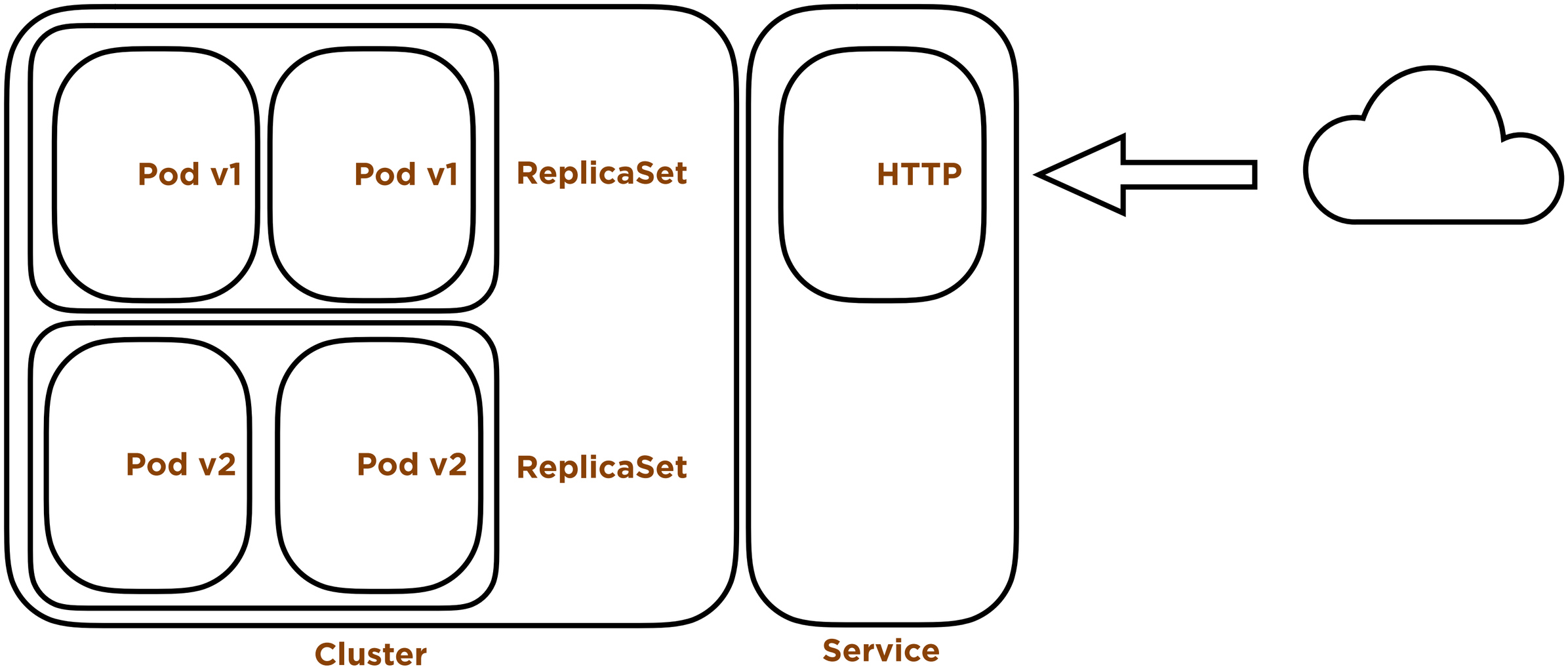
Services and ReplicaSets



Using Deployments

- **`Deployments`** are used to provide declarative updates to Pods and **`ReplicaSets`**
- We define the state and use the Deployment Controller to move towards that state
- **`Deployments`** are made of **`ReplicaSets`** and manage the transition between the **`ReplicaSets`**
- Scaling
  - Manually
  - Automatically based on resource consumption

Controller Operations - `Deployment`



Application Deployment in Kubernetes

- Imperative
  - **`kubectl run mywebapp --image=centinosystems.azurecr.io/mywebappimage`**
- Declarative
  - Define our desired state in code
  - Manifest
  - YAML or JSON
  - **`kubectl apply -f deployment.yaml`**

Demos!

Declaratively Deploying Applications in AKS

- **`Deployments`**
- **`Services`**

Scaling our application from 1 to 50 Replicas

What's Next?

- Building a Data Tier
  - Database Service
  - Database Connections
- Production Ready App Tier
  - Connection Strings in Azure Key Vault
  - SSL Termination (AppGW, Ingress…etc)
- DevOps
  - Automatically build container image
  - Automatically deploy to Kubernetes using a `Deployment`
  - Azure DevOps

More Resources

- **Docker for Windows/Mac**
- **Minikube**
- **Managed Service Providers**
  - Azure Kubernetes Service **(AKS)**
    - https://docs.microsoft.com/en-us/azure/aks/kubernetes-walkthrough
- **Pluralsight!**
  - https://app.pluralsight.com/profile/author/anthony-nocentino
  - **Kubernetes Installation and Configuration Fundamentals**
  - **Managing the Kubernetes API Server and Pods**

Review

- **Container Fundamentals**
- **Creating a Container Image**
- **Working with Azure Container Registry**
- **Deploying our Application in Azure Kubernetes Service**

Need more data or help?

http://www.centinosystems.com/blog/talks/

Links to resources
Demos
Presentation
Pluralsight

aen@centinosystems.com
@nocentino
www.centinosystems.com

Solving tough business challenges with technical innovation

Centino systems

Thank You!