STIJN VERHOEVEN
AZURE TECHNICAL CONSULTANT

Cloud Migration – Part 2: Evolve from IaaS to PaaS

ARE YOUR APPLICATIONS READY?

Today

Application silos, built in isolation

Limited set of platforms and form factors

Overabundance of data

Servers and infrastructure to manage

Upfront capacity planning, fixed scale

Future

Multi-channel applications, covering all touchpoints

Many platforms, devices and form factors

Data-driven intelligence in applications

Focus on application functionality, not infrastructure

Elastic, unlimited scale

“Pets vs Cattle” (Yes, again)

Scale Up

- **Servers are like pets.**

  Pets are given names, are unique, lovingly hand raised and cared for. When they get ill, you nurse them back to health

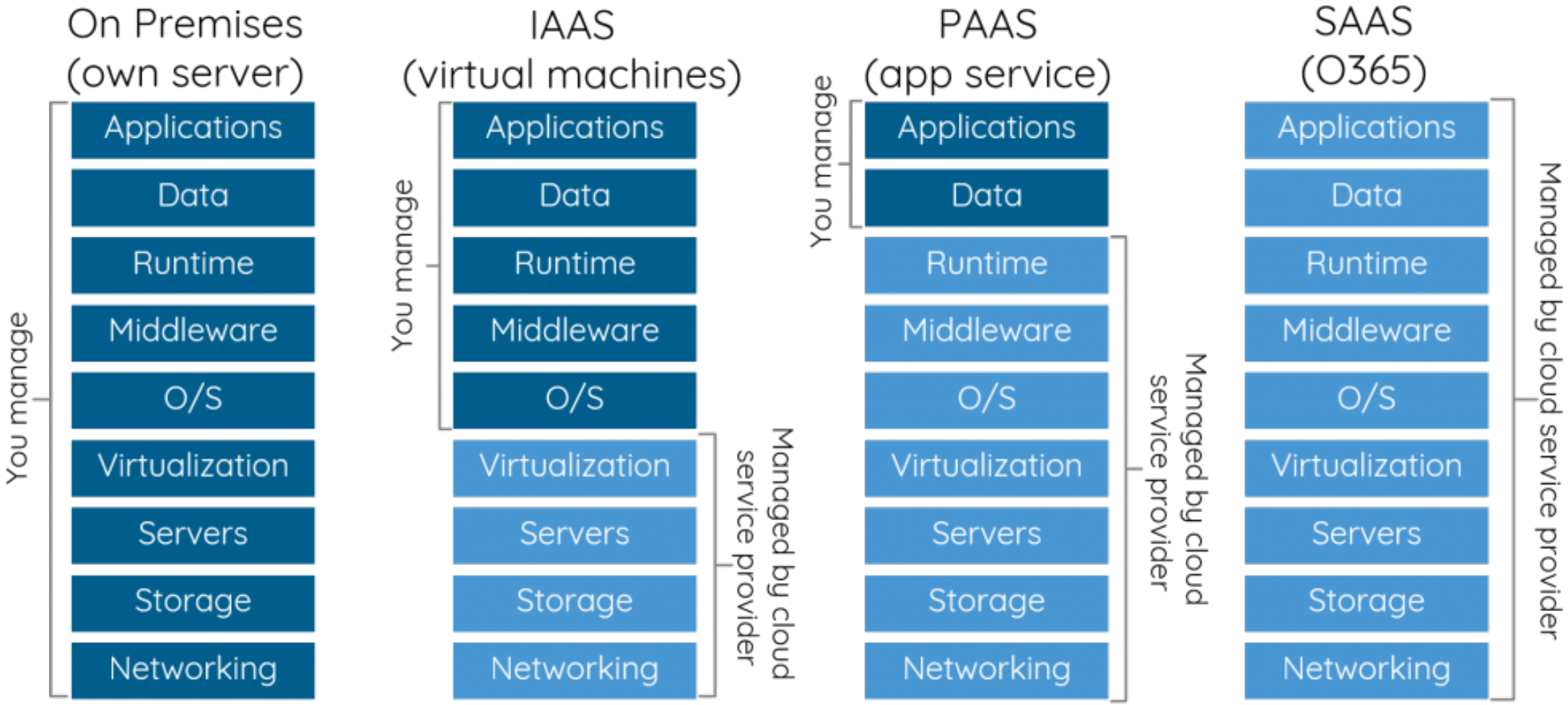
- **Servers are like cattle.**

  Cattle are given numbers and are almost identical to each other. When they get ill, you get another one.


How can I make sure that my existing applications can take maximum advantage of cloud capabilities?
I'm worried that I need to start from zero and rebuild for the cloud?
How do I get started, I have so many apps!

THE APPLICATION JOURNEY TO THE CLOUD

THE JOURNEY TO THE CLOUD



THE JOURNEY TO THE CLOUD

Migrate to the Cloud

Not sure which option is right for you? Dig deeper with the Choose the right Azure hosting option guide.

Option 1: Azure Virtual Machines

App compatibility ⓘ
Existing skill set ⓘ
Automated scale ⓘ
Automated patching ⓘ

Option 2: Azure App Service

App compatibility ⓘ
Existing skill set ⓘ
Automated scale ⓘ
Automated patching ⓘ

Option 3: Docker Containers

App compatibility ⓘ
Existing skill set ⓘ
Automated scale ⓘ
Automated patching ⓘ

THE JOURNEY TO THE CLOUD

IaaS/VM/Compute

Own your home

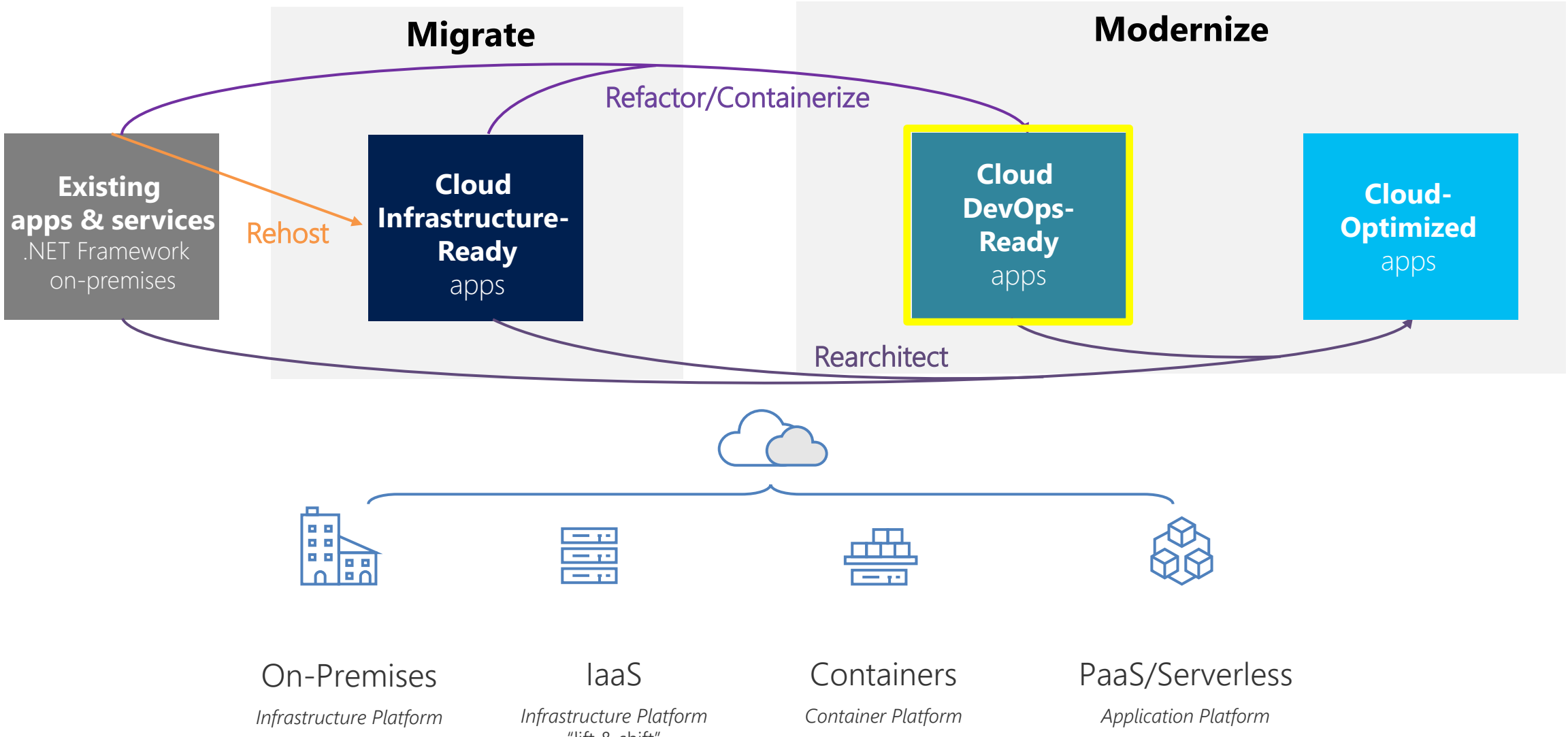
Platform as a Service

Bed and breakfast

Serverless

Hotel

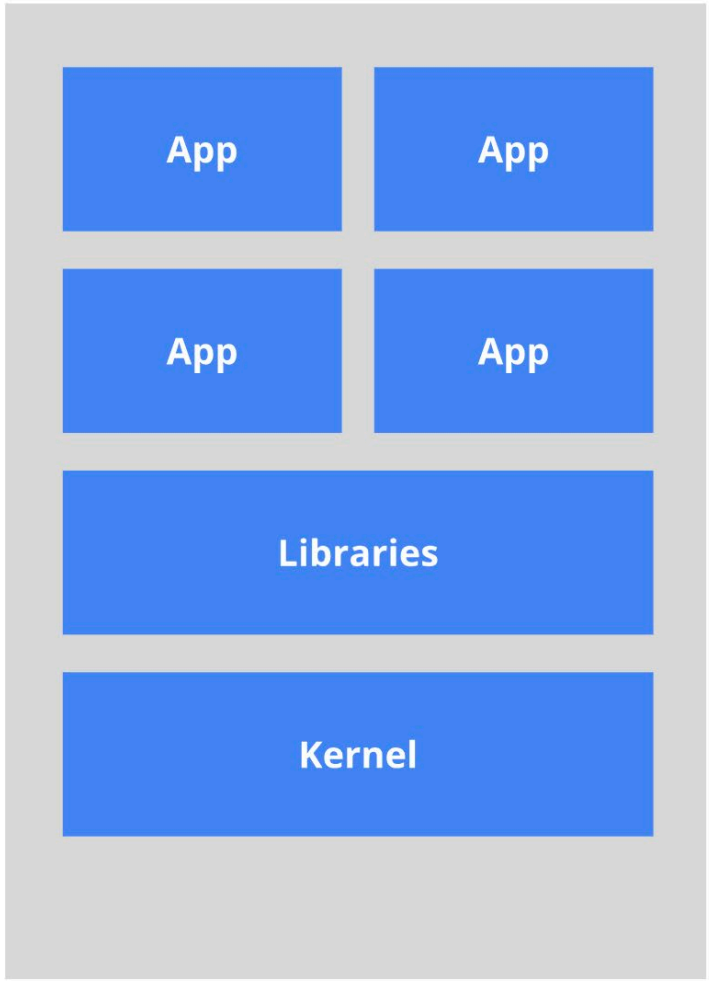
THE JOURNEY TO THE CLOUD



What are Containers

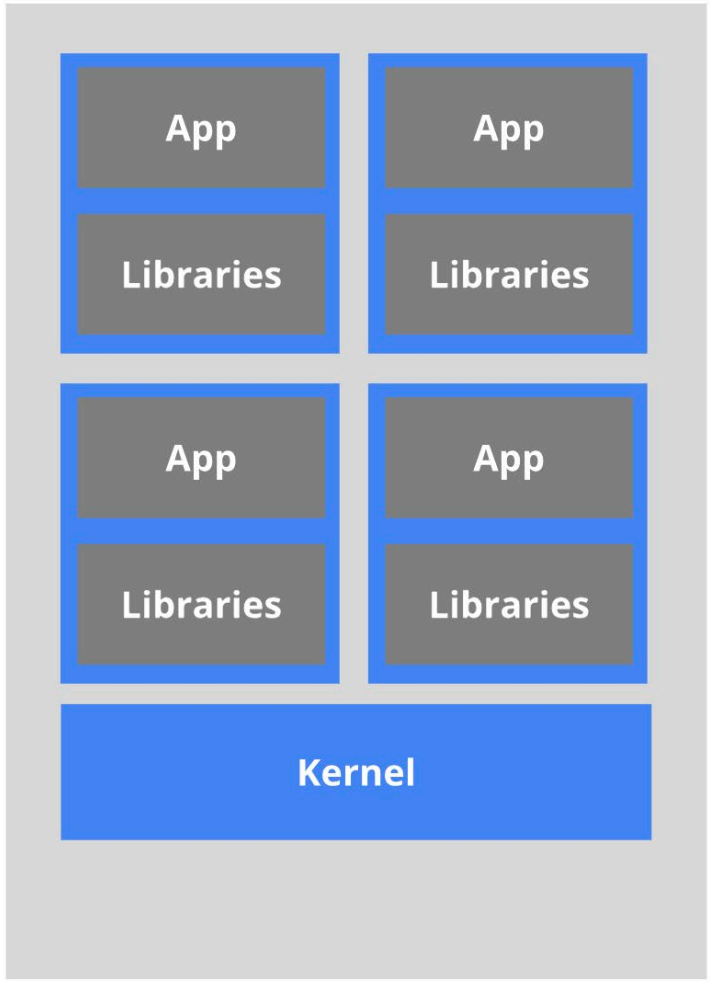
What are Containers

The old way: Applications on host



Heavyweight, non-portable
Relies on OS package manager

The new way: Deploy containers



Small and fast, portable
Uses OS-level virtualization

What are Containers

- What is a container?
  - Fully packaged applications
  - Including dependencies
  - Ready to deploy with a single call

What are Containers

- What is it not?
  - Virtual Machines
  - No hardware virtualization

Demo:

eShop Website running as Container on local dev machine

Moving from VMs to containers

- Containers = DevOps approach
  - Use CI/CD: Azure DevOps, Jenkins, …
- More cost effective, efficient and reliable
- Dramatic deployment and resilience benefits

Cost Savings

- Far more granular compute units
  - Use only what you need
- Dramatic utilization improvement
- Azure has per-second billing (when using Azure Container Instances)

Reliability

- Battle tested monitoring and orchestration tools
  - Orchestration Tools: Kubernetes, Openshift, …
  - Monitoring: Prometheus, Elasticsearch, …

- Incredible launch/restart/stop times

Security Wins

- Fine-grained “sandbox” per-process
  - Principle of least privilege for processes
- Quickly react to vulnerabilities, etc.
  - Easily build a new image with patch
- Open technologies get constant audits

Use Case:

Containerize eShop Website on Azure

THE JOURNEY TO THE CLOUD

Option 1: Azure Vi

App compatibility ⓘ

Existing skill set ⓘ

Automated scale ⓘ

Automated patching ⓘ

ocker Containers

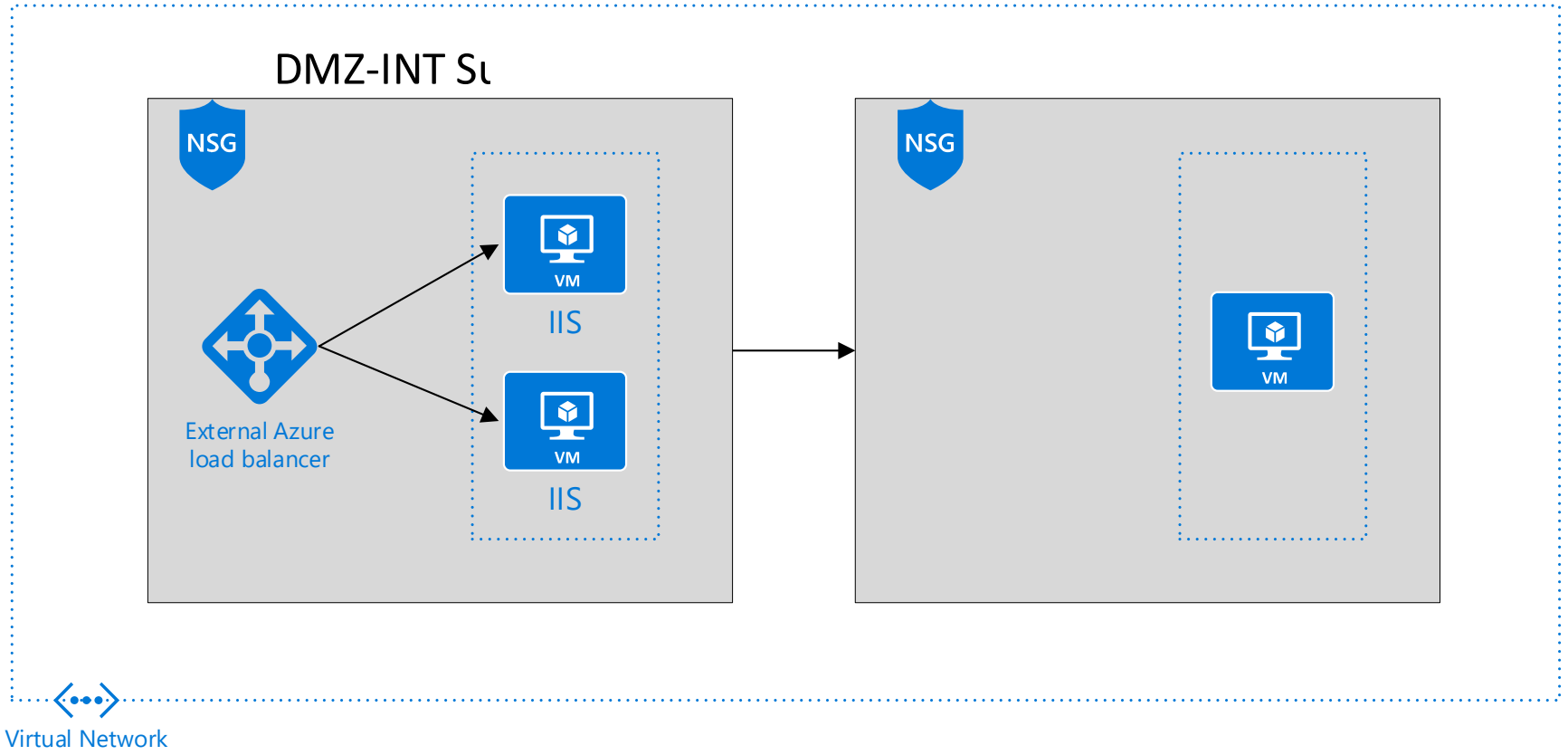
ⓘ

ng ⓘ

eShop Current Situation

- Situation after Lift & Shift:
  - ASP.NET Core website on IIS IaaS Webservers
  - Data plane: SQL 2017 IaaS VM
  - Exposed via Azure External Loadbalancer
  - Azure Automation in place for update and configuration management

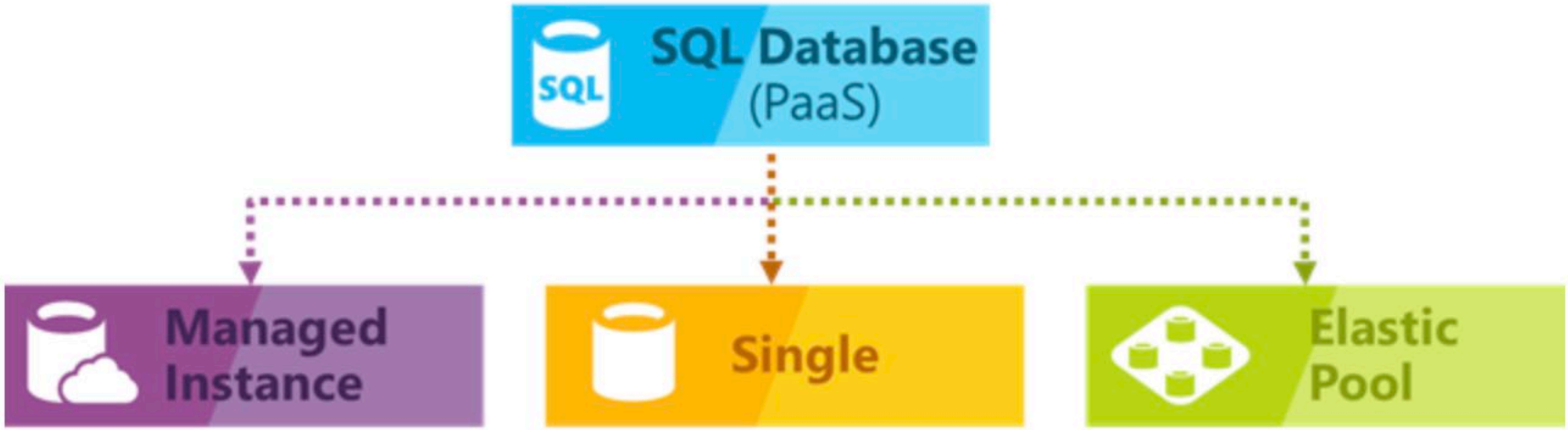
eShop Current Situation



Azure Automation:
Update Management
Change Tracking

Phase 1: Azure SQL

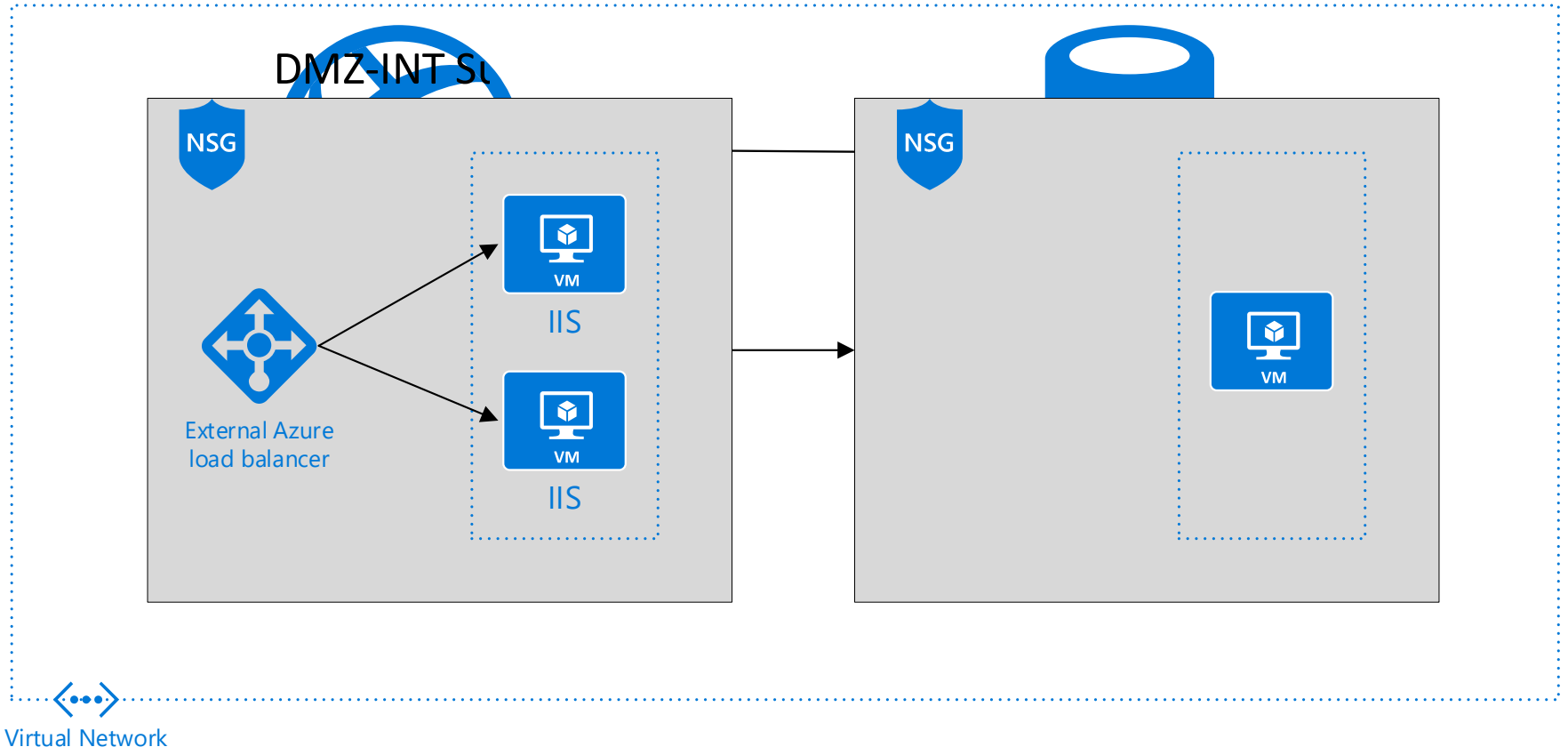
- Get rid of SQL IaaS VM
  - Azure SQL:
    - Patching handled by Microsoft
    - Dynamically scalable performance (Elastic Pools)
    - High Availability using Geo-Replication/Failover Groups



Phase 2: Azure WebApps for Containers

- Get rid of IIS IaaS Webservers
  - Azure Container Registry
  - Azure Webapps For Containers (Docker):
    - Security
    - Load Balancing
    - Auto Scaling
    - DevOps: Easy integration with Visual Studio, Azure DevOps, Github, Bitbucket, …
    - Deployment Slots: Test-Production
    - Serverless options using Azure Functions (App Service Plan)
    - Multiple languages supported

Demo:
Azure Container Registry
Azure WebApps for Containers



Azure Key Vault

Azure Automation:
Update Management
Change Tracking

Azure Automation:
Update Management
Change Tracking

PHASE 3 (OPTION B): CONTAINERS – KUBERNETES - AKS

- What is Kubernetes?

*Kubernetes provides a **container-centric** management environment. It orchestrates computing, networking, and storage infrastructure on behalf of user workloads. This provides much of the simplicity of Platform as a Service (PaaS) with the flexibility of Infrastructure as a Service (IaaS), and enables portability across infrastructure providers.**

*https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

PHASE 3 (OPTION B): CONTAINERS – KUBERNETES - AKS

Master Node 1

Master Node 2

Master Node 3

- kube-apiserver
- Etcd database
- Kube-scheduler
- Kube-controller-manager
- Cloud-controller-manager

Worker Node 1

Worker Node 2

Worker Node x

- Kubelet
- Kube-proxy
- Container Runtime (Docker, containerd, cri-o, rktlet, …)

Phase 2 (option b): Containers – Kubernetes - AKS

- Get rid of IIS IaaS Webservers:

  - Containers on Self-Hosted Kubernetes Cluster(s)
    - AKS-Engine
    - Rancher
    - …

  - Containers on Azure Kubernetes Services (AKS) aka ‘Kubernetes as a Service’
    - ‘PaaS’: Management plane is managed by Cloud Provider
    - IaaS for Worker Nodes
    - Still ‘lots of stuff’ to manage by DevOps Team
    - Integration with Log Analytics for Monitoring

Demo:
eShop Website On Kubernetes

WHAT'S NEXT ?

	Technical Track	Services & Management Track
12:30-13:30	Lunch	
13:30-14:15	Your data platform in the cloud: strategy and options *Brecht Vuylsteke*	DNA and added value of a chatbot *Dirk Gepts*