Contents

[Container Product/Services 1](#_Toc55100735)

[Find the Azure service for your container needs 1](#_Toc55100736)

[Azure Container Instance 2](#_Toc55100737)

[Container Image 2](#_Toc55100738)

[Container groups 2](#_Toc55100739)

[Azure Container Registry 3](#_Toc55100740)

[Azure Kubernetes Service (AKS) 3](#_Toc55100741)

[Pods 4](#_Toc55100742)

[Replica Sets 6](#_Toc55100743)

[Deployments 6](#_Toc55100744)

[BCDR in Azure Kubernetes Service (AKS) 7](#_Toc55100745)

[Plan for multiregion deployment 7](#_Toc55100746)

[Use Azure Traffic Manager to route traffic (DNS routing) 7](#_Toc55100747)

[Application routing with Azure Front Door Service 7](#_Toc55100748)

[Interconnect regions with global virtual network peering 7](#_Toc55100749)

[Enable geo-replication for container images 8](#_Toc55100750)

[Remove service state from inside containers 8](#_Toc55100751)

[Create a storage migration plan 8](#_Toc55100752)

[AKS Networking Concepts 9](#_Toc55100753)

[Azure Monitor for containers overview 9](#_Toc55100754)

[Azure Service Fabric 9](#_Toc55100755)

# Container Product/Services

<https://azure.microsoft.com/en-in/product-categories/containers/>

## Find the Azure service for your container needs

|  |  |
| --- | --- |
| IF YOU WANT TO | USE THIS |
| Simplify the deployment, management and operations of Kubernetes | [Azure Kubernetes Service (AKS)](https://azure.microsoft.com/en-in/services/kubernetes-service/) |
| Quickly create powerful cloud apps for web and mobile | [App Service](https://azure.microsoft.com/en-in/services/app-service/) |
| Easily run containers on Azure without managing servers | [Container Instances](https://azure.microsoft.com/en-in/services/container-instances/) |
| Cloud-scale job scheduling and compute management | [Batch](https://azure.microsoft.com/en-in/services/batch/) |
| Develop microservices and orchestrate containers on Windows or Linux | [Service Fabric](https://azure.microsoft.com/en-in/services/service-fabric/) |
| Store and manage container images across all types of Azure deployments | [Container Registry](https://azure.microsoft.com/en-in/services/container-registry/) |
| Run fully managed OpenShift clusters, jointly operated with Red Hat | [Azure Red Hat OpenShift](https://azure.microsoft.com/en-in/services/openshift/) |

# Azure Container Instance

<https://www.youtube.com/watch?v=jAWLQFi4USk>

<https://www.youtube.com/watch?v=oTM3xDpTrBE>

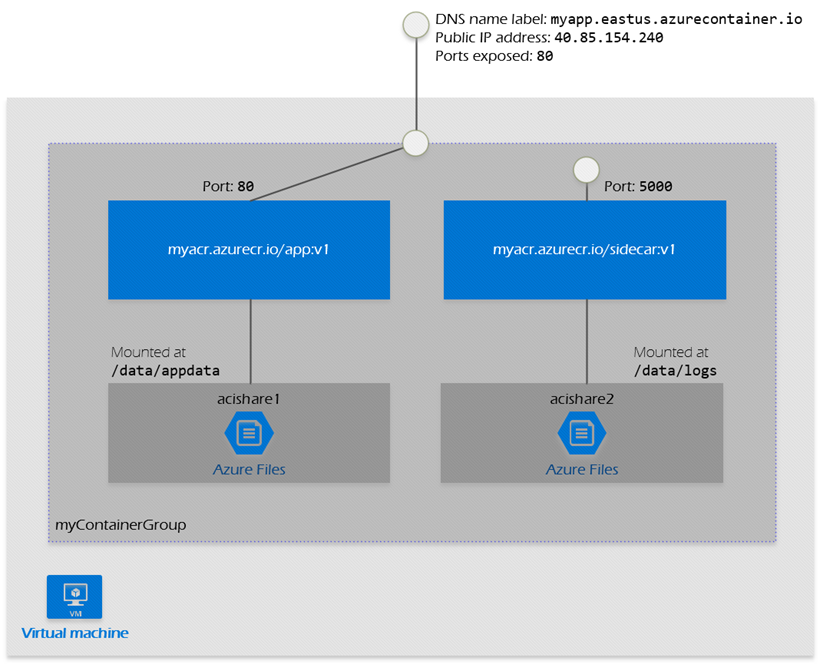
## Container Image

<https://www.youtube.com/watch?v=Bc7FpteQSAk>

## Container groups

<https://docs.microsoft.com/en-us/azure/container-instances/container-instances-container-groups>

A container group is a collection of containers that get scheduled on the same host machine. The containers in a container group share a lifecycle, resources, local network, and storage volumes. It's similar in concept to a pod in [Kubernetes](https://kubernetes.io/docs/concepts/workloads/pods/).



This example container group:

* Is scheduled on a single host machine.
* Is assigned a DNS name label.
* Exposes a single public IP address, with one exposed port.
* Consists of two containers. One container listens on port 80, while the other listens on port 5000.
* Includes two Azure file shares as volume mounts, and each container mounts one of the shares locally.

# Azure Container Registry

<https://www.youtube.com/playlist?list=PLpbcUe4chE7-4XhZurHEErjPqJzpGS5zp>

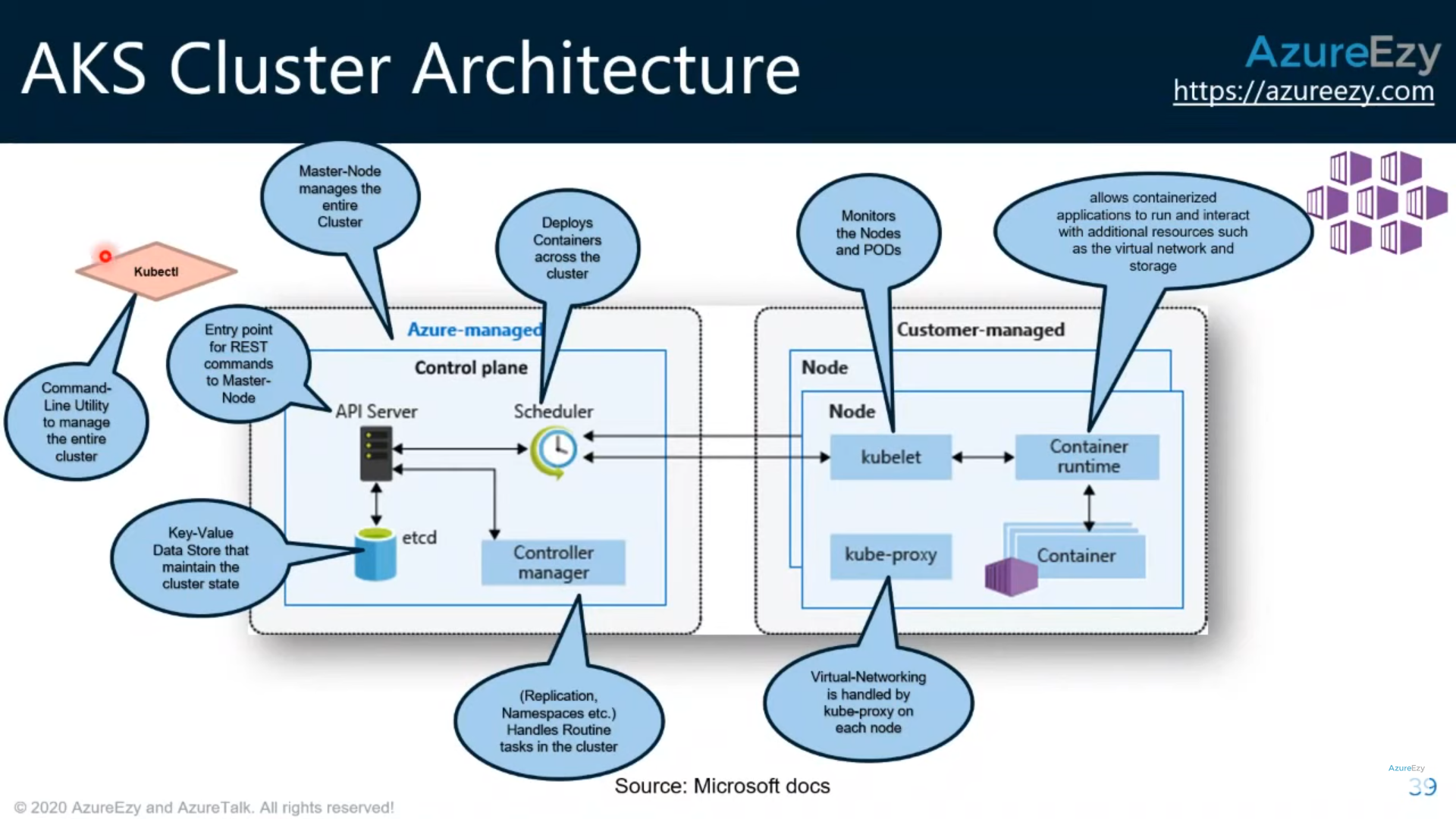
<https://github.com/HoussemDellai/azure-acr>

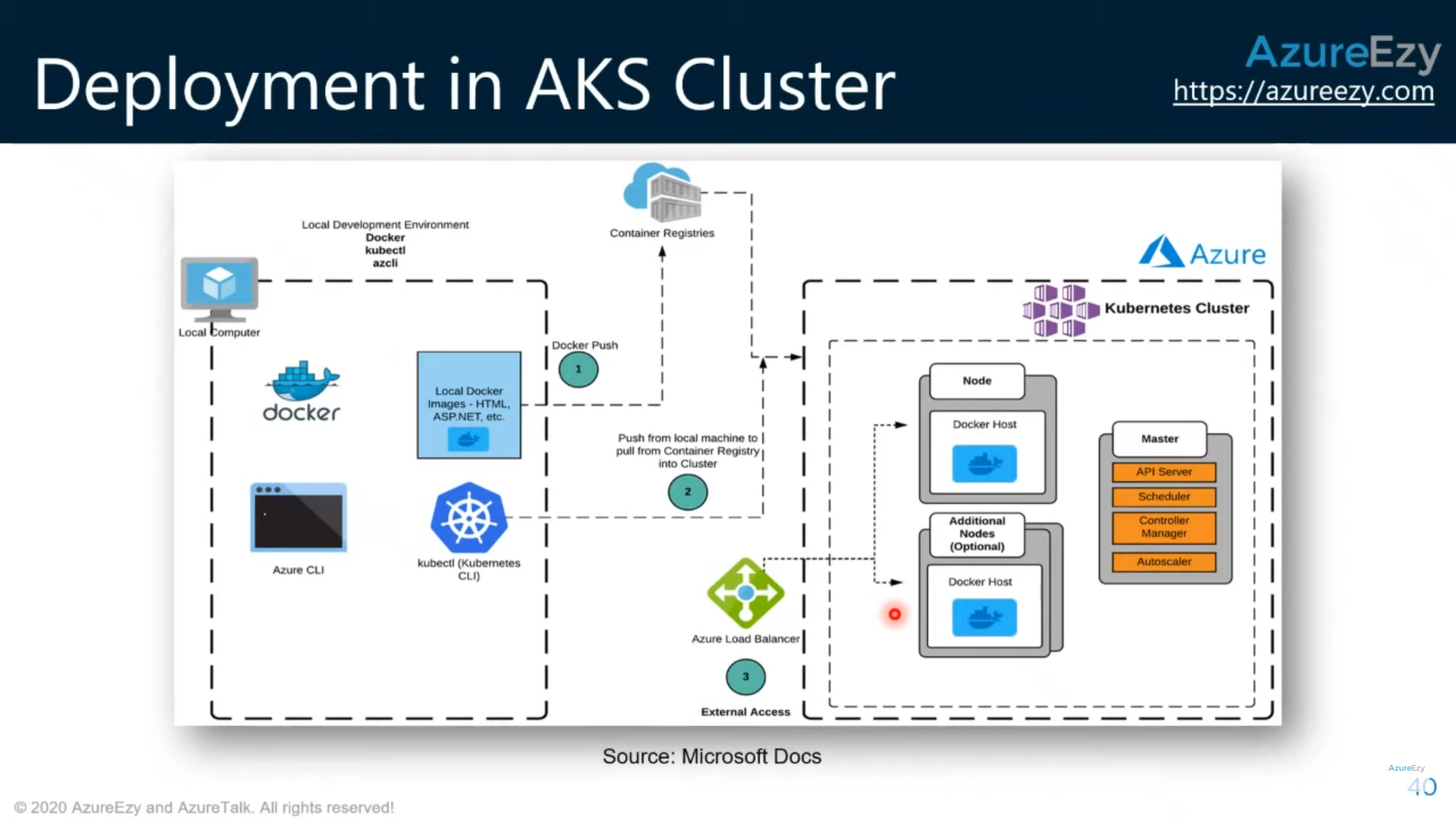
# Azure Kubernetes Service (AKS)

<https://docs.microsoft.com/en-us/azure/aks/intro-kubernetes>

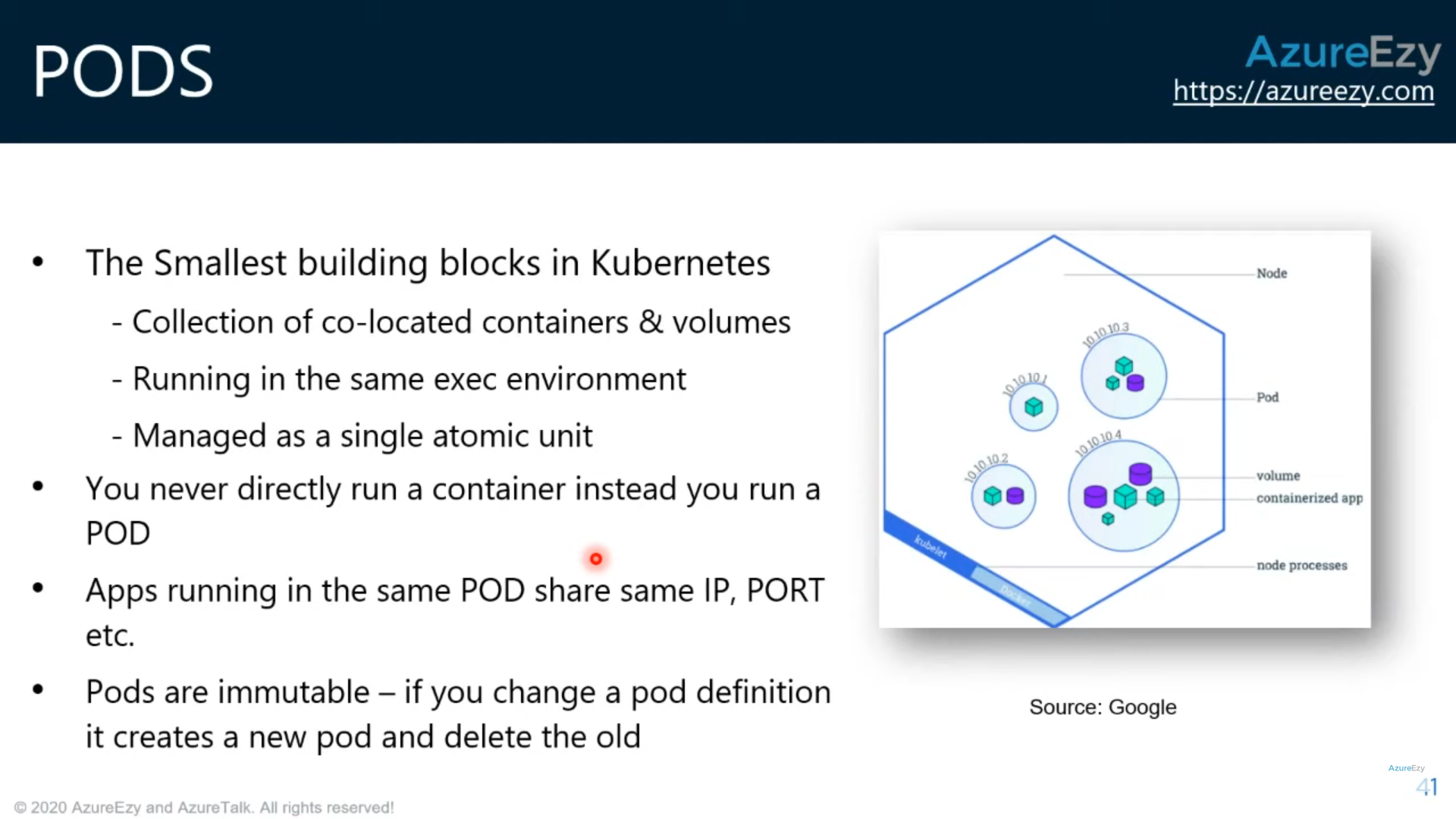
<https://www.youtube.com/playlist?list=PLp_fsLj4v7gTA8n2UngXfOc_5puPPmFdK>

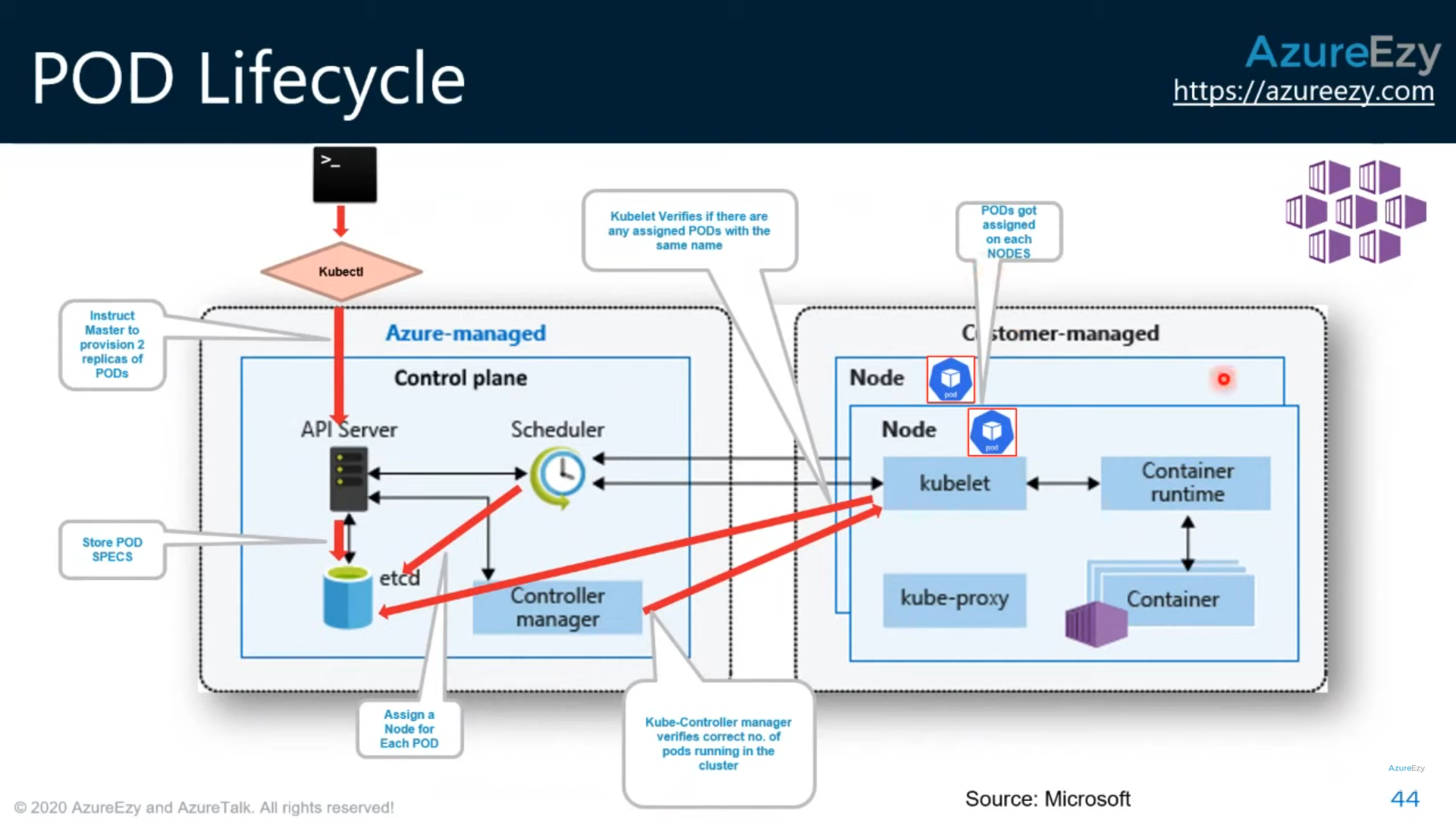
<https://www.youtube.com/playlist?list=PLfeid9nyFRy-arSxDJrq3-Rz5CP2OKJkr>



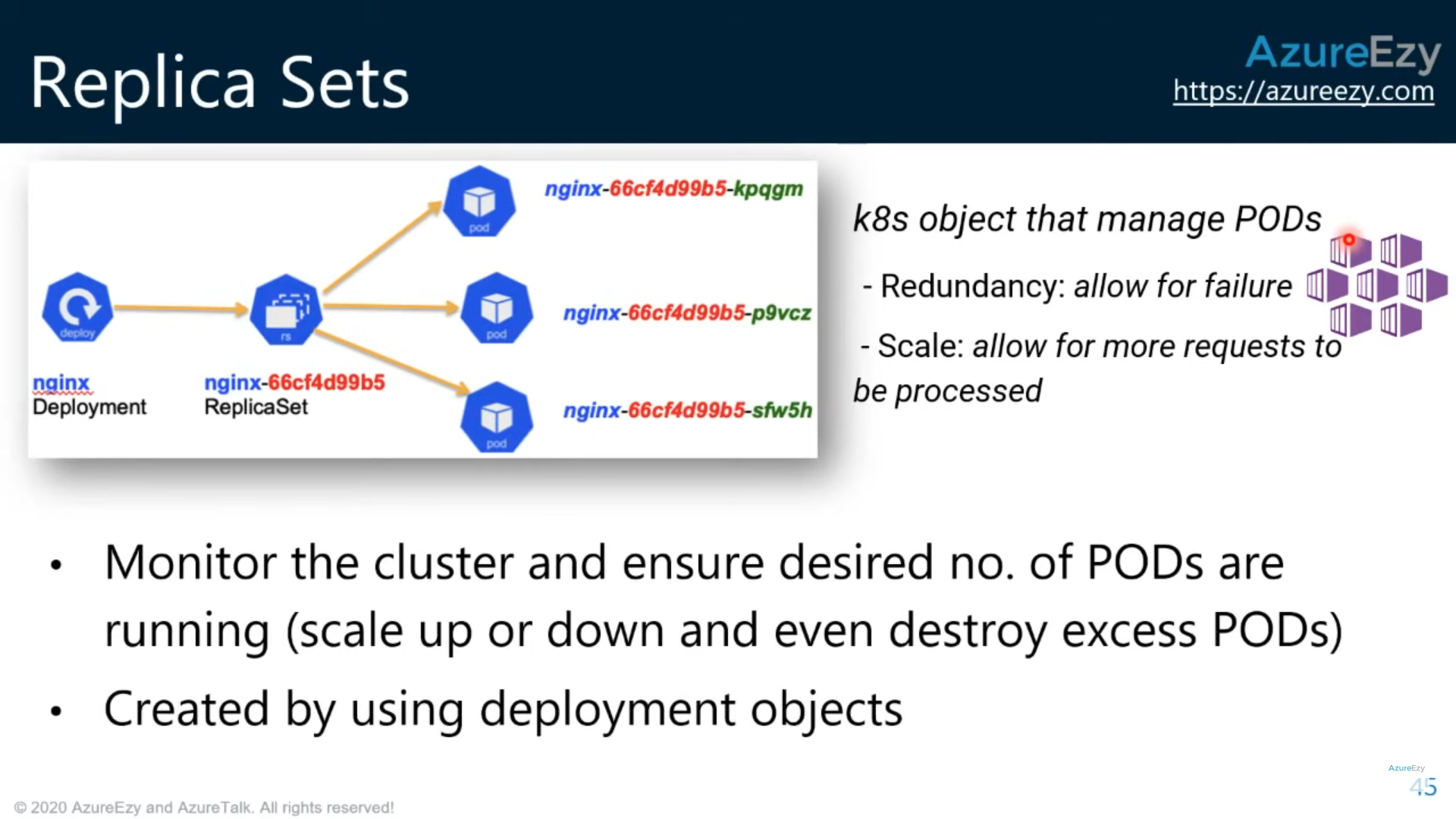


## Pods

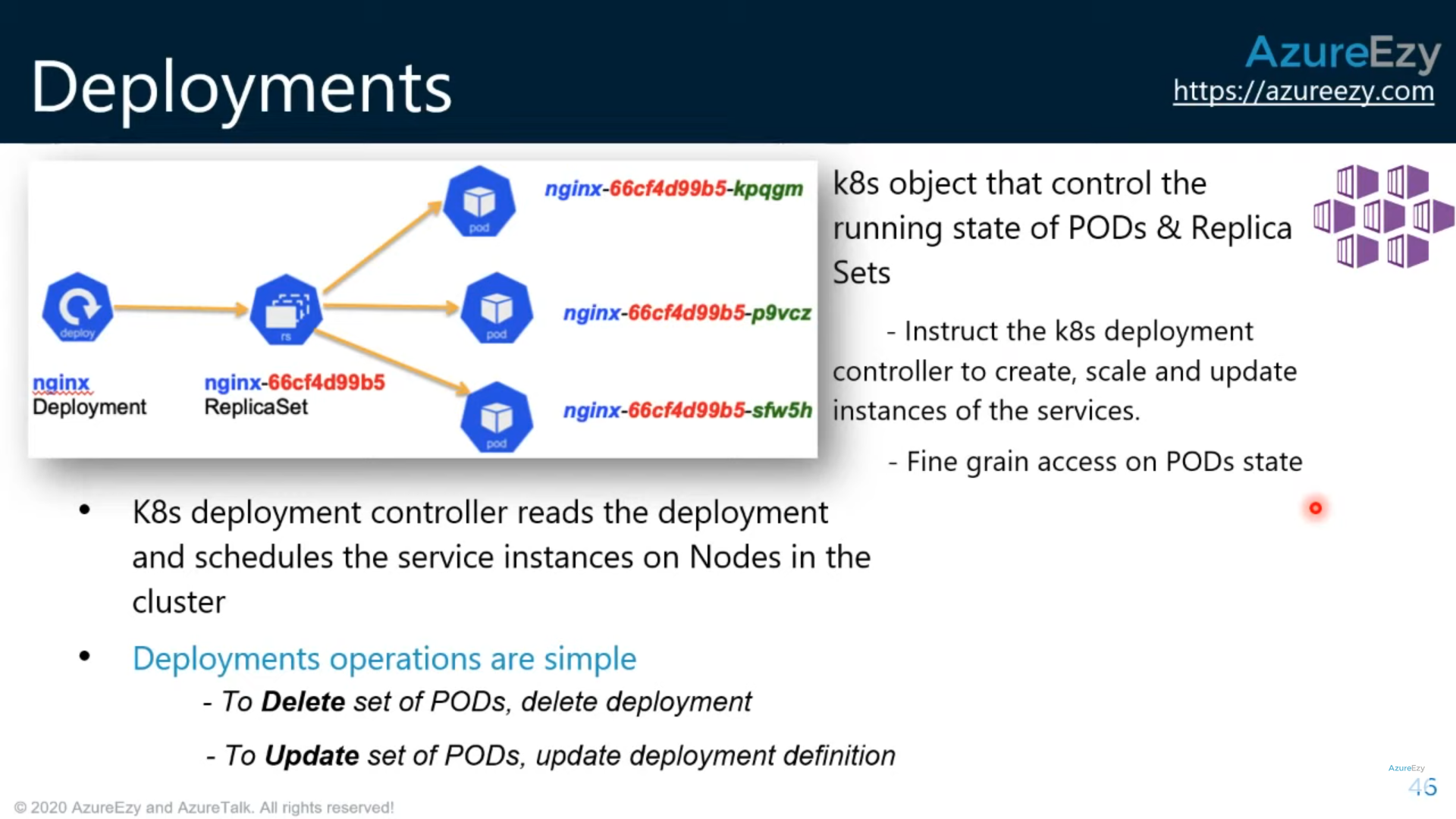




## Replica Sets



## Deployments



<https://github.com/CloudA2Z-Code/AZ_Azure_Webinars/blob/master/AZ_104_Webinar_Commands>

## BCDR in Azure Kubernetes Service (AKS)

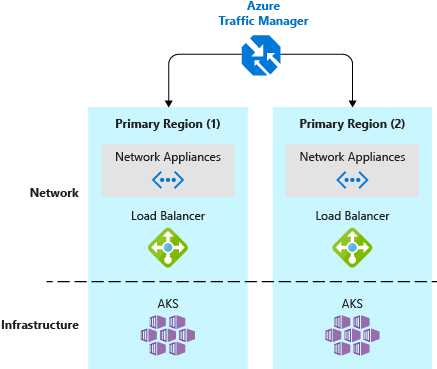
<https://docs.microsoft.com/en-us/azure/aks/operator-best-practices-multi-region>

### Plan for multiregion deployment

When you plan where to deploy your AKS cluster, consider:

* [**AKS region availability**](https://docs.microsoft.com/en-us/azure/aks/quotas-skus-regions#region-availability): Choose regions close to your users. AKS continually expands into new regions.
* [**Azure paired regions**](https://docs.microsoft.com/en-us/azure/best-practices-availability-paired-regions): For your geographic area, choose two regions that are paired with each other. Paired regions coordinate platform updates and prioritize recovery efforts where needed.
* **Service availability**: Decide whether your paired regions should be hot/hot, hot/warm, or hot/cold. Do you want to run both regions at the same time, with one region *ready* to start serving traffic? Or do you want one region to have time to get ready to serve traffic?

### Use Azure Traffic Manager to route traffic (DNS routing)



### Application routing with Azure Front Door Service

### Interconnect regions with global virtual network peering

### Enable geo-replication for container images

**Best practice**: Store your container images in Azure Container Registry and geo-replicate the registry to each AKS region.

Geo-replication is a feature of **Premium SKU container registries**.

### Remove service state from inside containers

**Best practice**: Where possible, don't store service state inside the container. Instead, use an Azure platform as a service (PaaS) that supports multiregion replication.

### Create a storage migration plan

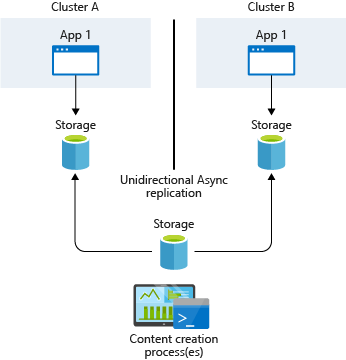
**Best practice**: If you use Azure Storage, prepare and test how to migrate your storage from the primary region to the backup region.

Here are two common ways to replicate storage:

* Infrastructure-based asynchronous replication
* Application-based asynchronous replication

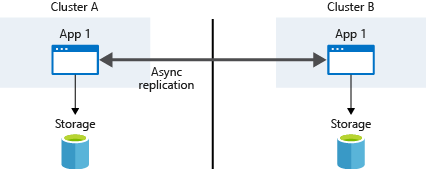
**Infrastructure-based asynchronous replication:**

The typical strategy is to provide a common storage point where applications can write their data. This data is then replicated across regions and then accessed locally.Common storage solutions such as [Gluster](https://docs.gluster.org/en/latest/Administrator%20Guide/Geo%20Replication/), [Ceph](https://docs.ceph.com/docs/master/cephfs/disaster-recovery/), [Rook](https://rook.io/docs/rook/v1.2/ceph-disaster-recovery.html), and [Portworx](https://docs.portworx.com/scheduler/kubernetes/going-production-with-k8s.html#disaster-recovery-with-cloudsnaps) provide their own guidance about disaster recovery and replication.



**Application-based asynchronous replication:**

Kubernetes currently provides no native implementation for application-based asynchronous replication. Typically, the applications themselves replicate the storage requests, which are then written to each cluster's underlying data storage.



## AKS Networking Concepts

<https://docs.microsoft.com/en-us/azure/aks/concepts-network>

# Azure Monitor for containers overview

<https://docs.microsoft.com/en-us/azure/azure-monitor/insights/container-insights-overview>

# Azure Service Fabric

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-overview>

**Azure Service Fabric** is more about just Azure Infrastructure and orchestrator. It is neutral on infrastructure. **That's why it can run on premise as well as on any cloud (including AWS and Azure).** This is a great option for teams who are looking to build an application using microservices architecture on premise. When the decisions on whether to move to cloud and which cloud are finalized, **they simply port the application from on premise to cloud**.

Unlike ACS, Service Fabric provides a prescriptive guidance on how application should be written. It provides a full blown programming model. This model proposes either **Reliable Services** or **Actor Model** to write application. Application written using this programming model, **can either be stateful or stateless**. It is more fault-tolerant and easy to scale. Multiple such services forms a microservices architecture based application. It gives more control (and ownership!) of the underlying infrastructure to developers.

## Event analysis and visualization with Application Insights

<https://docs.microsoft.com/en-us/azure/service-fabric/service-fabric-diagnostics-event-analysis-appinsights>

Part of Azure Monitor, Application Insights is an extensible platform for application monitoring and diagnostics.