Microsoft

Kubernetes Technical Briefing

Module 2:
Azure Kubernetes Service

Agenda

- Azure Kubernetes Service Overview
- Regions and Availability Zones
- Node Pools / Cluster Autoscaler
- Cluster Authentication
- Kubernetes RBAC and Azure AD
- Networking
- AKS Integrations

Introduction to Azure Kubernetes Service

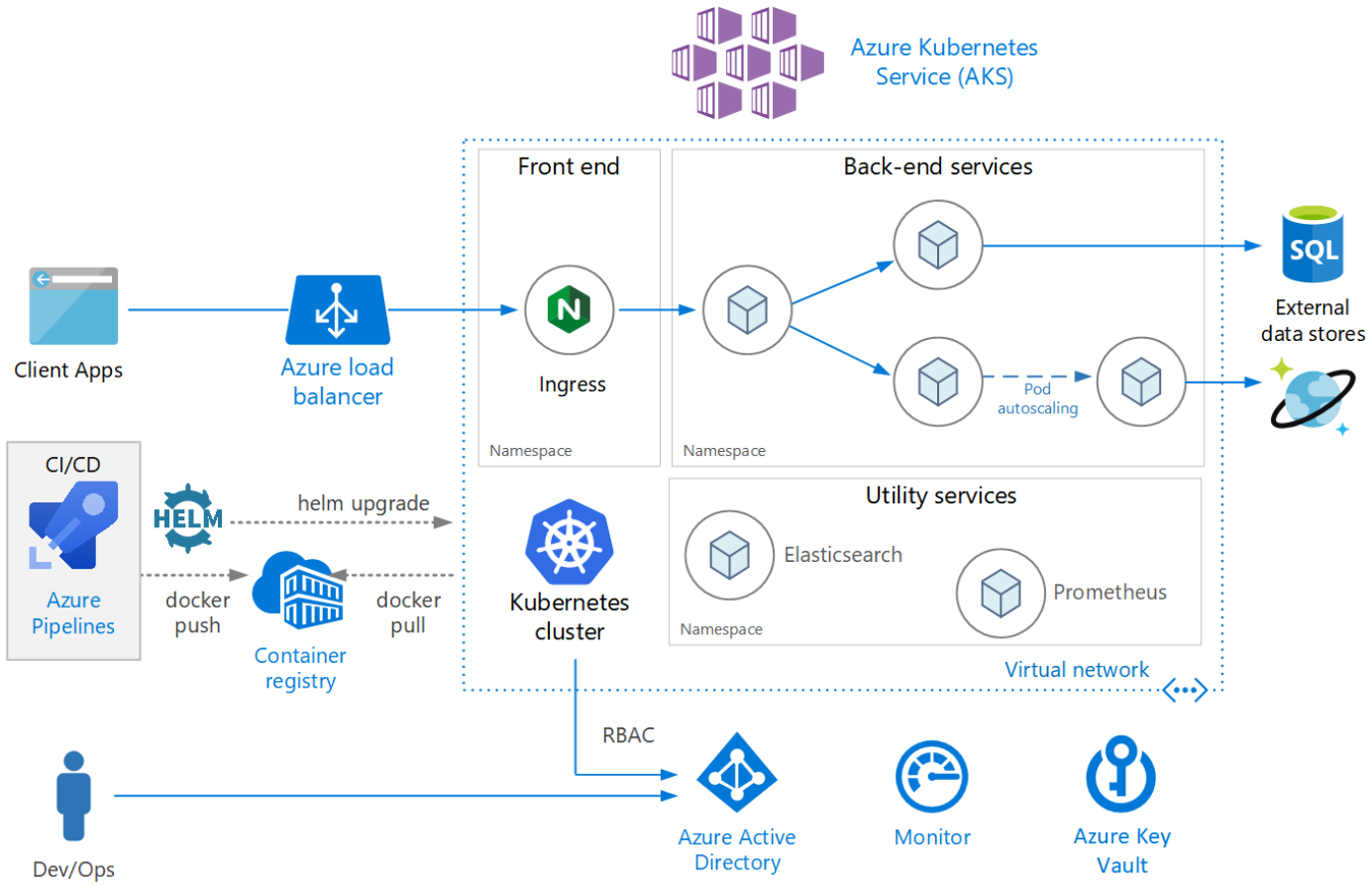
1

Cloud-Hosted, Managed Kubernetes

- ***Self-hosting*** Kubernetes can be extremely difficult if you do not have the necessary expertise, especially in a mixed OS environment.
- A ***cloud-hosted***, managed Kubernetes solution, it doesn't require a lot about your network configurations or have a lot of experience with infrastructure.
- Your cloud provider will deploy and keep everything running and updated for you.
- When you're ready to scale up to more machines and higher availability, you'll find that a hosted solution will make it very easy for you to quickly adjust your infrastructure.
- In addition to maintaining your cluster, having a cloud-hosted Kubernetes solution allows your cluster to easily integrate with other cloud services with ease (database, queues, persistent storage, etc.).

Azure Kubernetes Service (AKS)

- Azure Kubernetes Service is the Microsoft solution for hosting Kubernetes as a managed **PaaS** service.
- AKS is highly integrated with other Azure Services like Azure Container Registries (ACR), Azure Disk, Azure File Shares, Azure SQL, Azure AD, Azure Monitor, Azure Key Vault, and many more.



Azure Kubernetes Service (AKS)

- AKS automates provisioning, upgrading, monitoring, and scaling with, a simpler development-to-production experience, and enterprise-grade security and governance.
- By providing Kubernetes as a managed service, AKS abstracts the complexity and operational overhead of managing Kubernetes.
- Unlike some cloud providers, Azure doesn't charge an hourly fee per instance AKS running. An Azure Kubernetes Service instance is completely ***free***!
- The only costs of running an AKS cluster are based the resources the cluster uses (nodes, public IPs, etc.).
- You have full control over the cluster's cost based on the size and type resources you choose to add to it and how available do you need your cluster to be.

Installing AKS

- There are multiple ways to create/upgrade Azure Kubernetes Service:
  - **Azure Portal** – A web-based UI that allows you create AKS clusters with the most commonly used settings (not all settings are available in the Azure Portal). Good for training and creating development clusters.
  - **Azure CLI** – A cross-platform command-line interfaces that allows you to configure ALL AKS features and settings. Use when creating production clusters.
  - **Bicep/Terraform** – Infrastructure as code tools that lets you *declaratively* define the configuration of your clusters. Use to deploy multiple clusters and supporting services.

Kubernetes Version

- Kubernetes in an open-source application hosted on GitHub.
- Major and minor releases are released frequently.
- When creating Azure Kubernetes Service cluster, you need to decide which version of Kubernetes you want to install as your control plane.
- The default version selected in the Azure Portal is **2** releases back from the latest stable (non-preview) release.
- You can upgrade your cluster to a newer version later.
- The nodes must be running a version of Kubernetes that is equal to or less than the current version on the control plane.

Regions and Availability Zones

54 regions worldwide

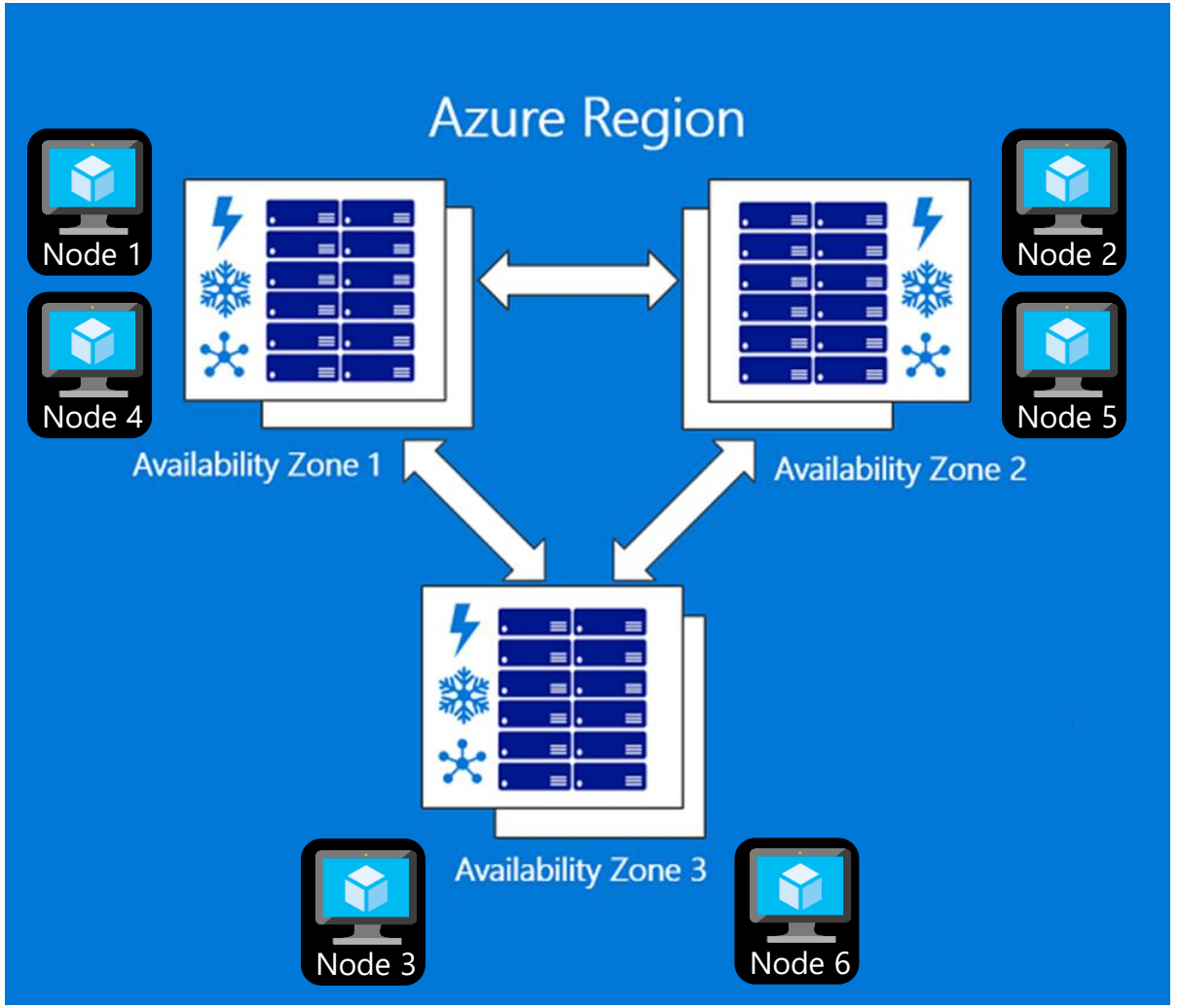
140 available in 140 countries

AKS is Available in a Limited Number of Regions

- **Azure Region** - A set of datacenters deployed within a latency-defined perimeter and connected through a dedicated regional low-latency network.
- AKS with Availability Zones is available in the following regions:
  - Australia East
  - Brazil South
  - Canada Central
  - Central US
  - East US
  - East US 2
  - France Central
  - Germany West Central
  - Japan East
  - North Europe
  - Southeast Asia
  - South Central US
  - UK South
  - US Gov Virginia
  - West Europe
  - West US 2

Availability Zones

- **Availability Zone** - Unique physical locations within a region. Each zone is made up of one or more datacenters equipped with independent ***power, cooling, and networking***.
- Zone-redundant services distribute your applications across Availability Zones to protect from single-points-of-failure.



Node Pools

3

Cluster Node Pools

- A ***Node Pool*** in a set of Nodes (VMs) with the exact **same configuration**.
- In AKS, each Node Pool maps to a ***Virtual Machine Scale Set***.
- Every AKS cluster contains at least one Node Pool.
- A default Node Pool is created when a new AKS cluster is provisioned.
- The default Node Pool is designated as a ***System*** Node Pool, for running Kubernetes utilities.
  - System Nodes should not be confused with Master Nodes, which control the Kubernetes cluster and are hidden.
  - You have no control over the configuration and number of Mater Nodes.
- While you can schedule your workloads on system nodes, it's recommended that you create a separate ***User*** Node Pool to run your Pods.

Types of Node Pools

AKS supports several types of Node Pools:

- **System Node Pool** – Used by system utilities like ***CoreDNS*** and ***Metrics Server***. System Node Pools can only use the Linux OS.
- **Linux User Node Pool** – Pool of VMs that can run user workloads requiring the Linux OS.
- **Windows User Node Pool** - Pool of VMs that can run user workloads requiring the Window 2019 OS.
- **Spot Node Pool** – A User Node Pool that consists of Spot VM Instances. Azure can reclaim these VMs at any time if it needs their compute power. Spot VMs are up to 90% cheaper than regular VMs. These are great for "expensive" workloads that are resilient enough to tolerate being deleted and recreated as Nodes are removed and reattached.
- **Virtual Nodes** – Dynamic nodes created as wrappers for Azure Container Instances. One node is created for each Pod it hosts. See limitations.

Use Labels and Node Selectors to Schedule Workloads

Leverage well-known, prepopulated node labels and node selectors to ensure your workloads are scheduled in the correct node pools.

System Node Pool

```
nodeSelector:
  kubernetes.azure.com/mode: system
```

Linux User Node Pool

```
nodeSelector:
  kubernetes.azure.com/mode: user
  kubernetes.io/os: linux
```

Windows Node Pool

```
nodeSelector:
  kubernetes.io/os: windows
```

User Spot Node Pool

```
nodeSelector:
  kubernetes.azure.com/mode: user
  kubernetes.azure.com/scalesetpriority: spot
  kubernetes.io/os: linux
```

Virtual Nodes

```
nodeSelector:
  kubernetes.io/role: agent
  kubernetes.io/os: linux
  type: virtual-kubelet
tolerations:
- key: virtual-kubelet.io/provider
  operator: Exists
```

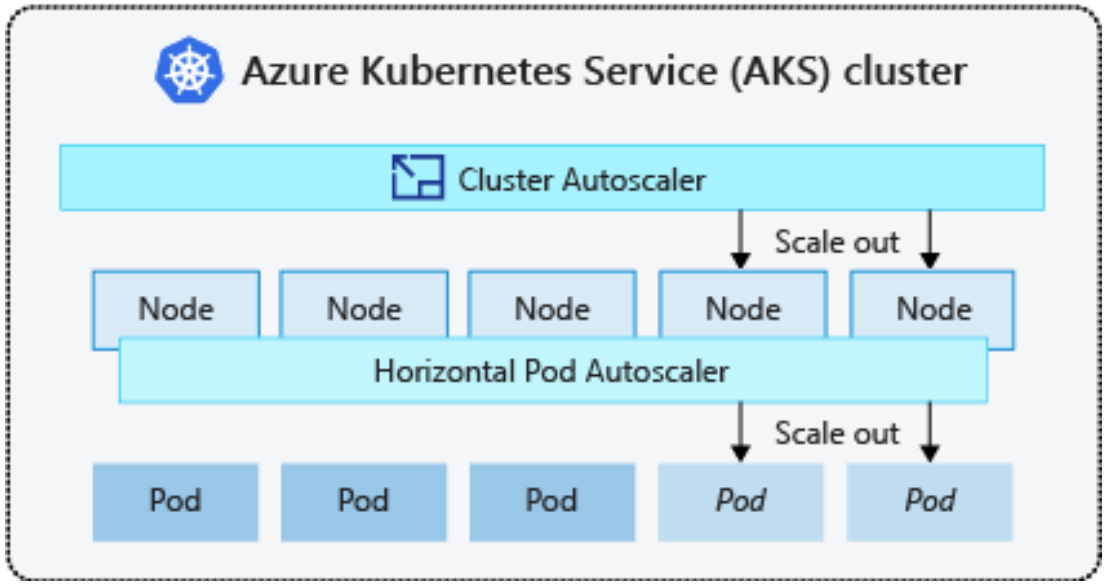
You can also automatically add custom labels to every node in a node pool and implement your own node selection criteria.

Right-Sizing VM Sizes

- The number and size of VMs in a clusters will depend on your system requirements.
- Use a *smaller* VM size for the ***System Node Pool*** to minimize costs. For example, using DS2_v2 nodes with the default 512 GB OS disk are enough for most system workloads.
- For ***User Node Pools***, use *larger* VM sizes to pack the maximum number of pods on a node. This will minimize the footprint of services that run on all nodes, such as monitoring and logging.
- Deploy at least 2-3 nodes per node pool. That way, the workload will have a high availability pattern with multiple replicas.
- Use the ***Cluster Autoscaler*** to have AKS automatically adjust node counts based on demand instead of always having enough nodes available to handle peak loads.
- When planning capacity for your cluster, assume that your workload can consume up to 80% of each node; the remaining 20% is reserved for AKS services.
- If the node sizes/counts don't meet your needs, you can reconfigure them at any time.

Cluster Autoscaler

- The **Cluster Autoscaler** watches for pods in your cluster that can't be scheduled because of resource constraints.
- When demand goes up, the number of nodes in a node pool is increased to meet the application demand.
- When demand go downs, the autoscaler automatically deletes nodes from the node pool.
- Each node pool can have different scaling options or manual scaling.



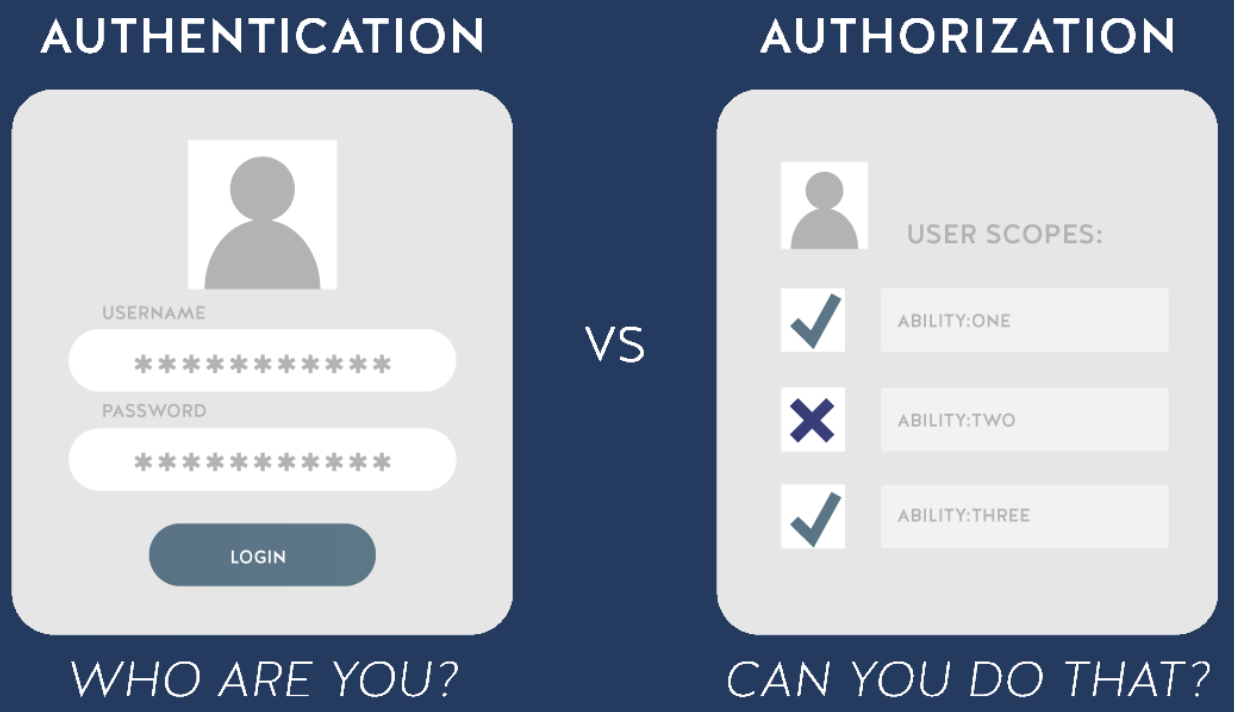
Cluster Autoscaler Customization

In addition to Min/Max number of nodes, the **Cluster Autoscaler** can be configured for fine-grain control of its operations. Here are just some of the options available:

Setting	Description	Default value
scan-interval	How often cluster is reevaluated for scale up or down	10 seconds
scale-down-delay-after-add	How long after scale up that scale down evaluation resumes	10 minutes
scale-down-delay-after-delete	How long after node deletion that scale down evaluation resumes	scan-interval
scale-down-unneeded-time	How long a node should be unneeded before it is eligible for scale down	10 minutes
expander	Type of node pool expander to be used in scale up. Possible values: most-pods, random, least-waste, priority	Random
Many more…		

4

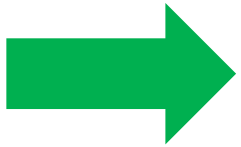
Authentication & Authorization



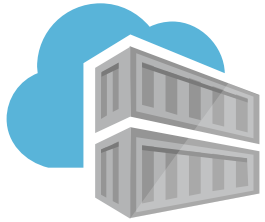
Authentication and Authorization from Azure Kubernetes Service to Azure Resources

An Azure Kubernetes Service cluster to be given permissions to access other Azure resources:

Azure Kubernetes Service

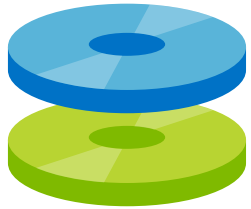


Container Registries



Key Vault

Managed Disks



Storage Accounts

Monitor

Type of Service Principals

There are 2 types of Service Principals in Azure:

- **Application** - This type of service principal is the local representation of a global ***application*** object in a single tenant or directory. The service principal object defines what the app *can do in the specific tenant, who can access the app, and what resources the app can access*.
- **Managed Identity** - Managed identities eliminate the need for developers to manage credentials. Managed identities provide an identity for applications to use *when connecting to* ***resources*** *that support Azure AD authentication*. Service principals representing managed identities can be granted access and permissions but cannot be updated or modified directly.

Azure Role Based Access Control

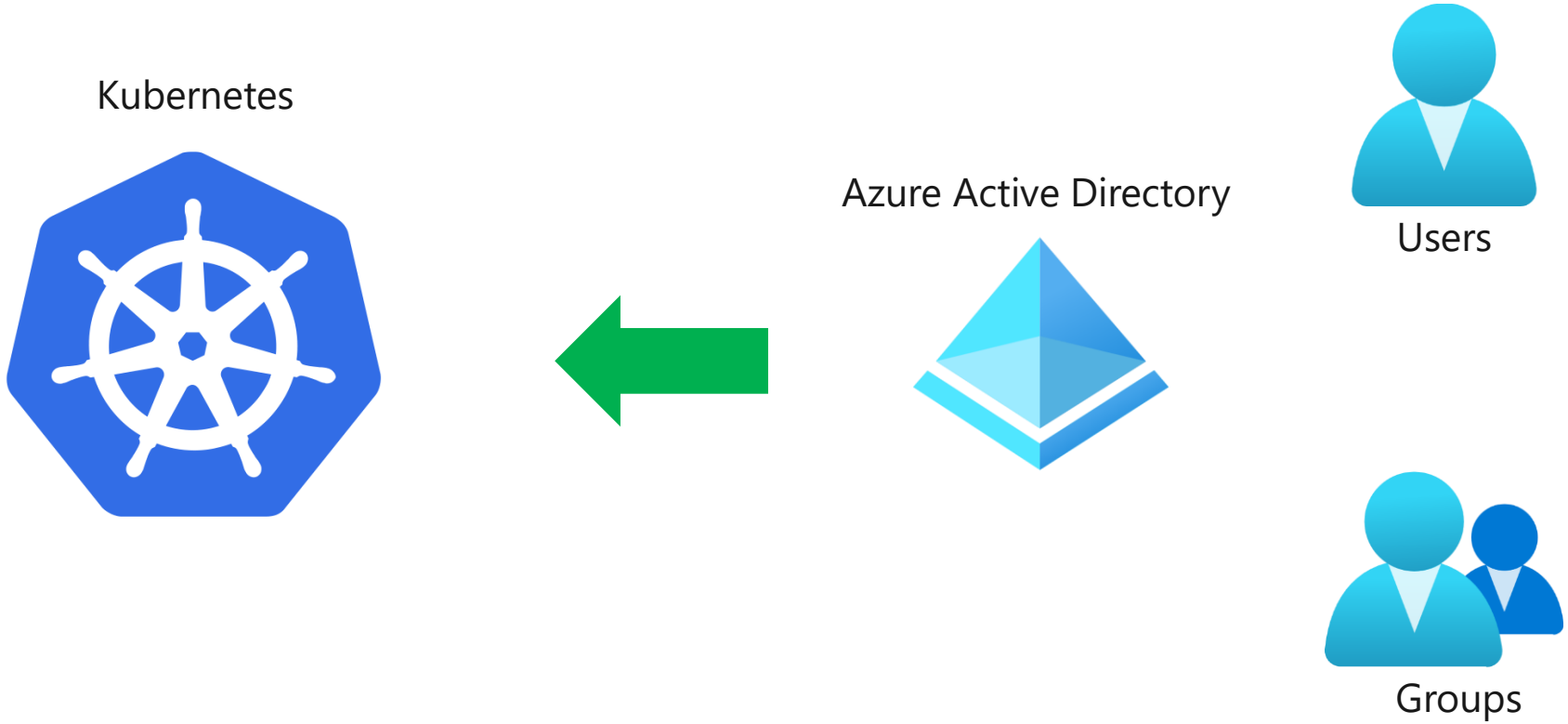
- Azure uses ***Role-Based Access Control*** (RBAC) to associate permissions with Service Principals.
- Roles common to most resources:
  - **Owner** – Can perform any action, including assign RBAC to other objects
  - **Contributor** – Can perform any action *except* assigning RBAC to others.
  - **Reader** – Can only perform read actions of a resource.
- There are also resource-specific roles available to assign to AKS:
  - **AcrPull** – The cluster's ability to pull images from the specified Azure Container Registries.
  - **Key Vault Secrets User** – The cluster's ability to read secret contents from a Key Vault.
  - **Monitoring Metrics Publisher** - The cluster's ability to send metrics to Azure Monitor.
  - **Network Contributor** - The cluster's ability to control the spoke virtual network.
- If none of the built-in RBAC roles match requirements, you can create custom roles to meet your needs.

Azure Service Principals and Azure RBAC Demo

Manage service principals in the Azure Portal

Authentication and Authorization into Kubernetes

External users/groups need to be given permissions to access a Kubernetes cluster.



Kubernetes User Access

- Kubernetes does not have objects which represent normal user accounts.
- Normal users cannot be added to a cluster through an API call.
- Kubernetes supports several strategies to support **external** authentication.
  - Static Passwords or Tokens
  - X.509 Certificates
  - ***Single Sign-On using OpenID***
  - Authentication Proxy
  - Webhook Token Authentication
- Azure Active Directory can be used to provide authentication to AKS clusters with OpenID Connect.
- You can use Azure AD Users and Groups to control access to AKS clusters.

Kubernetes RBAC

- Kubernetes also uses Role-based access control (RBAC) to regulate access to internal resources based on the roles bound to users/groups/service accounts.
- Kubernetes RBAC is enabled by default. Disabling RBAC is **not recommended**.
- Use the `--enable-aad` option to have Azure AD manage AKS authentication.
- Once enabled, Azure AD integration cannot be disabled.
- Kubernetes RBAC works with Azure AD as follows:
  - A user sends the `az aks get-credentials` command to get the credentials of the cluster.
  - Azure AD will authenticate the user's identity against the Azure roles that are allowed to get cluster credentials.
  - The user's token and email is passed to the cluster.
  - The user/group is bound to a Kubernetes **Role**, which defines the scope and actions that are allowed on various Kubernetes resources.

Azure RBAC

- You also have the option of use Azure RBAC, instead of Kubernetes RBAC, to manage authorization within the cluster.
- There are 4 built-in Azure AD RBAC roles available:

Role	Description
Azure Kubernetes Service RBAC Reader	Allows read-only access to see most objects in a namespace. It doesn't allow viewing roles or role bindings. This role **doesn't allow viewing Secrets**, since reading the contents of Secrets enables access to ServiceAccount credentials in the namespace.
Azure Kubernetes Service RBAC Writer	Allows read/write access to most objects in a namespace. This role **doesn't allow viewing or modifying roles or role bindings**. However, this role **allows accessing Secrets** and running Pods as any ServiceAccount in the namespace.
Azure Kubernetes Service RBAC Admin	Allows admin access, intended to be granted within a namespace. Allows read/write access to most resources in a namespace (or cluster scope), including the ability to create roles and role bindings within the namespace. This role **doesn't allow write access to resource quota or to the namespace itself.**
Azure Kubernetes Service RBAC Cluster Admin	Allows **super-user access** to perform any action on any resource. It gives full control over every resource in the cluster and in all namespaces.

- Use the `--enable-azure-rbac` option to use Azure AD RBAC authorization.

Networking

5

Azure Virtual Networks

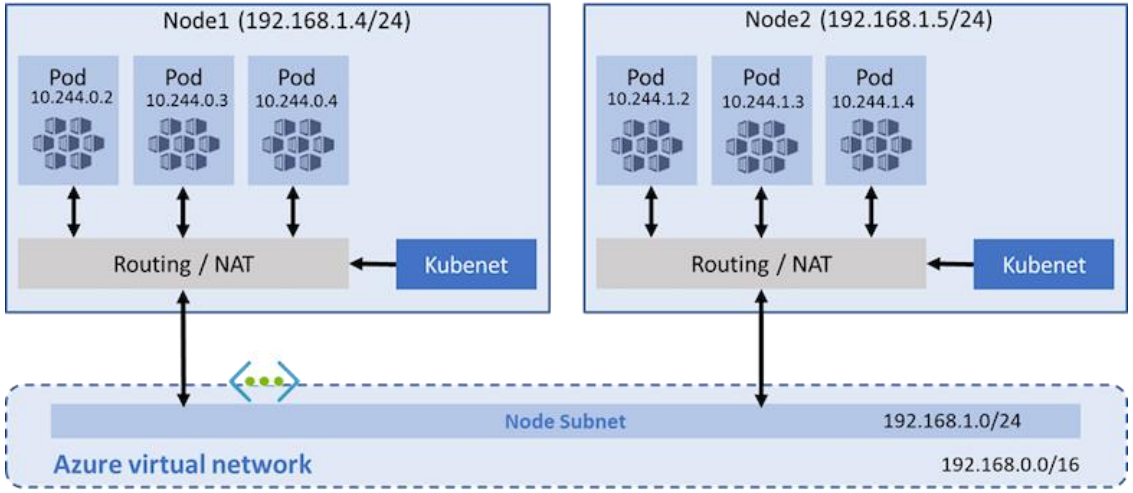
In AKS, you can deploy a cluster that uses one of the following two network models:

- ***Kubenet*** networking (Basic)
  - The network resources are typically created and configured as the AKS cluster is deployed.

- ***Azure Container Networking Interface (CNI)*** networking
  - The AKS cluster is connected to existing virtual network resources and configurations.

Kubenet (basic) Networking

- Uses **Kubenet** network plugin (default configuration).
- Azure virtual network and subnet are created automatically.
- Pods receive IP addresses from an internal address space, logically different from the Nodes' Azure virtual network subnet.
- Network address translation (NAT) is configured so Pods can reach resources on the Azure virtual network.
- The source IP address of traffic is translated to the node's primary IP address.
- Only works with Linux nodes.

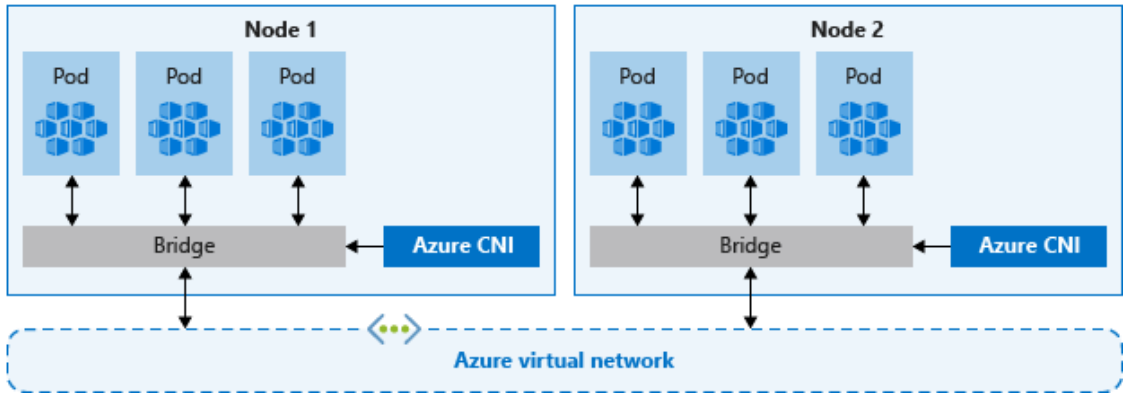


Drawbacks

- 2 different IP CIDRs to manage
- NAT Performance impact
- Peering or On-Premise connectivity is difficult to achieve

Advanced Networking (Azure CNI)

- **Azure CNI** allows a Kubernetes cluster to be integrated with an existing VNET.
- Pod get IP addresses from the subnet and can be accessed directly from the VNET.
- IP addresses must be unique across the network space.
- Each node has a configuration for maximum number of pods (***--max-pods***).
  - The equivalent number of IP addresses are reserved up front for that node.
- Requires planning and sometimes leads to IP address exhaustion.



Advantages

- Single IP CIDR to manage.
- Good Performance. No NAT needed.
- Peering and On-Premise connectivity is available out of the box.

Planning IP Addresses

- The address space of the virtual network should be large enough to hold all subnets.
- Account for all entities that will receive traffic. IP addresses for those entities will be allocated from the subnet address space. Consider these points:

- ***Upgrade:***
  - During an upgrade process, AKS creates a node that temporarily hosts the pods, while the upgrade node is cordoned and drained. That temporary node is assigned an IP address from the cluster subnet.
  - For pods, you might need additional addresses depending on your strategy. For rolling updates, you'll need addresses for the temporary pods that run the workload while the actual pods are updated.
  - If you use the ***Recreate*** strategy, old pods are removed, and then new ones are created. So, addresses associated with the old pods are reused.

Planning IP Addresses

- ***Scalability:***
  - Take into consideration the node count of all system and user nodes and their maximum scalability limit. Suppose you want to scale out by 400%. You'll need four times the number of addresses for all those scaled-out nodes.
  - Using **Azure CNI**, each pod can be contacted directly. So, each pod needs an individual address. Pod scalability will impact the address calculation. That decision will depend on your choice about the number of pods you want to grow.

- ***Azure Private Link addresses***
  - Factor in the addresses that are required for communication with other Azure services over Private Link. Addresses assigned for the links to Azure Container Registry and Key Vault.
  - Certain addresses are reserved for use by Azure. They can't be assigned.

For more information about planning IP for an AKS cluster, see Plan IP addressing for your cluster.

AKS Integrations

AKS Network Policy

- Network policy is a Kubernetes feature available in AKS that lets you control the traffic flow between pods.
- Azure provides two ways to implement network policy. You choose a network policy option when you create an AKS cluster. The policy option can't be changed after the cluster is created:
  - Azure's own implementation, called **Azure Network Policies**.
  - **Calico Network Policies**, an open-source network and network security solution.
- Both implementations use Linux IPTables to enforce the specified policies.
- Policies are translated into sets of allowed and disallowed IP pairs, which are then programmed as IPTable filter rules.

Use Azure Policy to Secure AKS Cluster

- To improve the security of your Azure Kubernetes Service (AKS) cluster, you can apply and enforce built-in security policies on your cluster using ***Azure Policy***.
- Azure Policy helps to enforce organizational standards and to assess compliance at-scale.
- After installing the Azure Policy Add-on for AKS, you can apply individual policy definitions or groups of policy definitions called initiatives (sometimes called policysets) to your cluster.

Azure Policy built-in definitions for AKS

Below is a *very* partial list of some of the built-in Azure Policy definitions for AKS.

Name	Description	Effect(s)
Kubernetes cluster pods should use specified labels	Use specified labels to identify the pods in a Kubernetes cluster. This policy is generally available for Kubernetes Service (AKS), and preview for AKS Engine and Azure Arc enabled Kubernetes. For more information, see https://aka.ms/kubepolicydoc.	audit, deny, disabled
Kubernetes cluster containers CPU and memory resource limits should not exceed the specified limits	Enforce container CPU and memory resource limits to prevent resource exhaustion attacks in a Kubernetes cluster. This policy is generally available for Kubernetes Service (AKS), and preview for AKS Engine and Azure Arc enabled Kubernetes. For more information, see https://aka.ms/kubepolicydoc.	audit, deny, disabled
Kubernetes cluster containers should only listen on allowed ports	Restrict containers to listen only on allowed ports to secure access to the Kubernetes cluster. This policy is generally available for Kubernetes Service (AKS), and preview for AKS Engine and Azure Arc enabled Kubernetes. For more information, see https://aka.ms/kubepolicydoc.	audit, deny, disabled
Kubernetes cluster containers should only use allowed images	Use images from trusted registries to reduce the Kubernetes cluster's exposure risk to unknown vulnerabilities, security issues and malicious images. This policy is generally available for Kubernetes Service (AKS), and preview for AKS Engine and Azure Arc enabled Kubernetes. For more information, see https://aka.ms/kubepolicydoc.	audit, deny, disabled

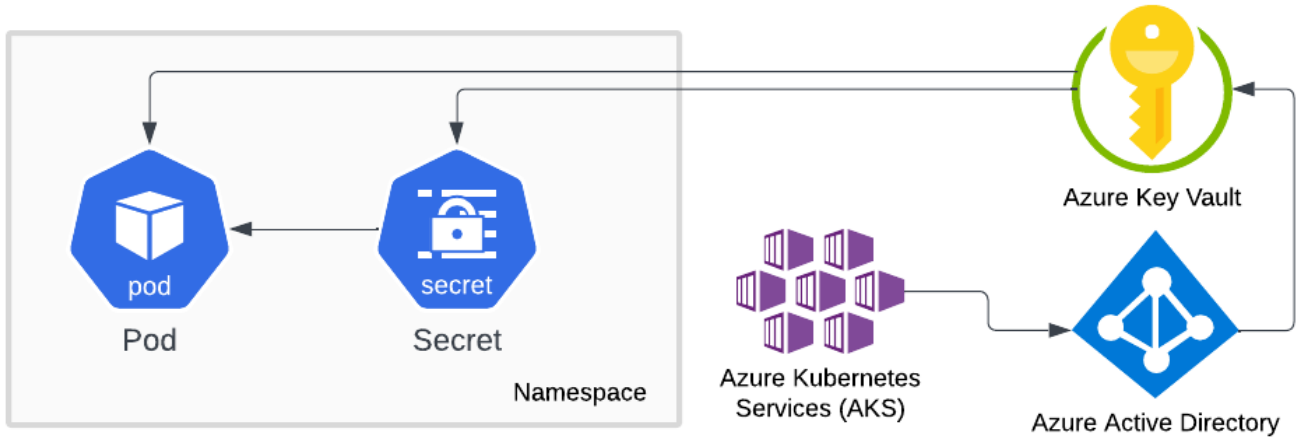
See Azure Policy built-in definitions for AKS for a complete list of AKS policy and initiative definitions.

Azure Policy Walkthrough

Explore Kubernetes Specific Azure Policies

Azure Key Vault

- Azure Kubernetes Service supports integration with ***Azure Key Vault*** to facilitate best practices when accessing sensitive information.
- A Secret Store Provider can be enabled during the creation of a cluster.
- AKS uses a managed identity to connect to Key Vault and dynamically create Kubernetes Secrets and/or inject secrets direct into containers.

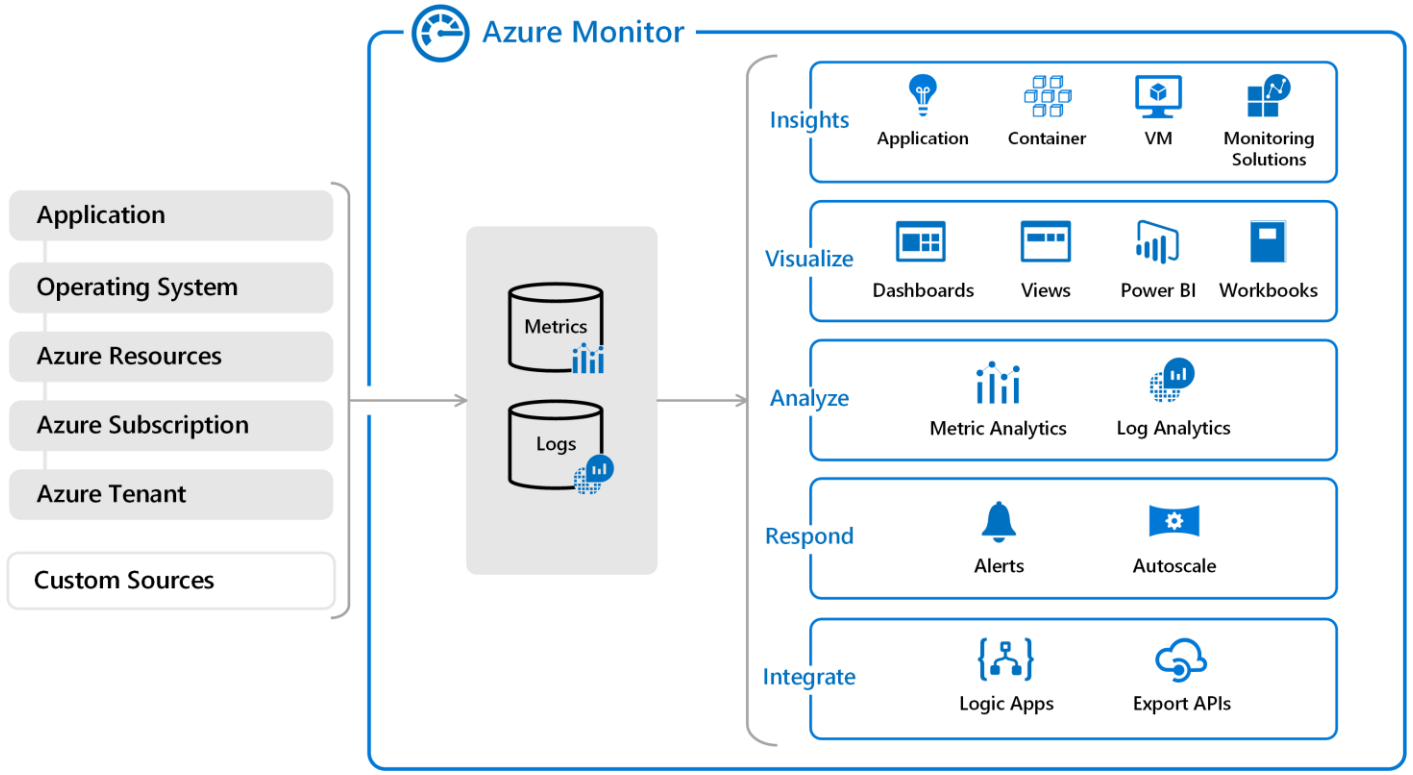


Azure Container Registry

- When running containers in AKS, you can pull their images from any container registry, if you have a Docker Secret configured in your cluster.
- Azure offers a private container registry as a service called Azure Container Registry (ACR).
- Build, store, secure, scan, replicate, and manage container images and artifacts with a fully managed, geo-replicated instance of ACR.
- Benefits of using ACR instead of a public registry:
  - You can import images from public registries
  - You can block unauthorized access to your images.
  - You won't have public facing dependencies (in case a public image is deleted).
  - You can access image pull logs to monitor activities and triage connectivity issues.
  - Take advantage of integrated container scanning and image compliance using ***Microsoft Defender for Containers***.

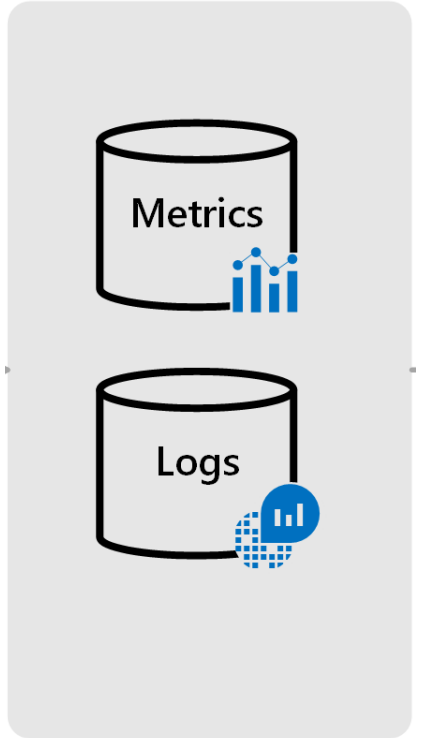
Azure Monitor

- ***Azure Monitor*** helps you maximize the availability and performance of your applications and services.
- It delivers a comprehensive solution for collecting, analyzing, and acting on telemetry from your cloud and on-premises environments.
- Most Azure resources can send their telemetry to Azure Monitor.



Log Analytics Workspace

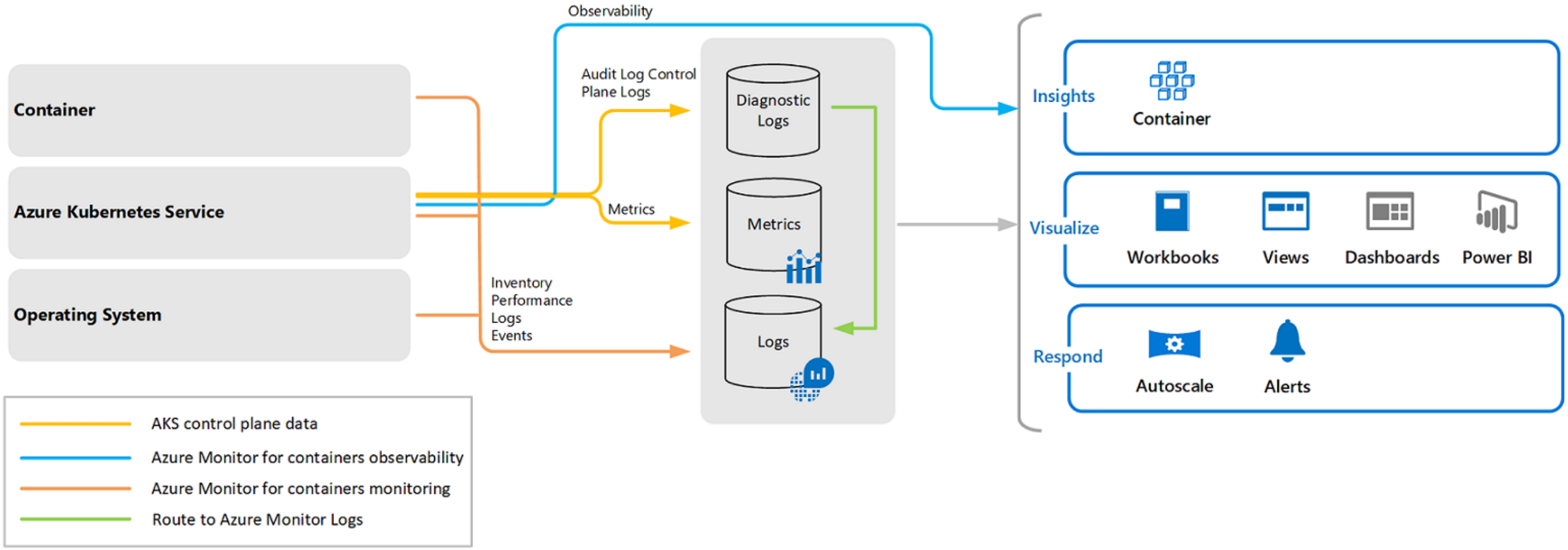
- A ***Log Analytics Workspace*** is a centralized databases that stores metrics and logs from AKS clusters.
- You can create a Log Analytics Workspace when you create an AKS cluster or leverage an existing instance.
- Log Analytics Workspace instances are identified by a **Workspace ID**.
- OMS Agents (Pods created on each Node by a DaemonSet) monitor nodes and send telemetry data to a Log Analytics Workspace instance*.



***Note:** Node metrics are not available from Virtual Nodes.

Azure Container Insights

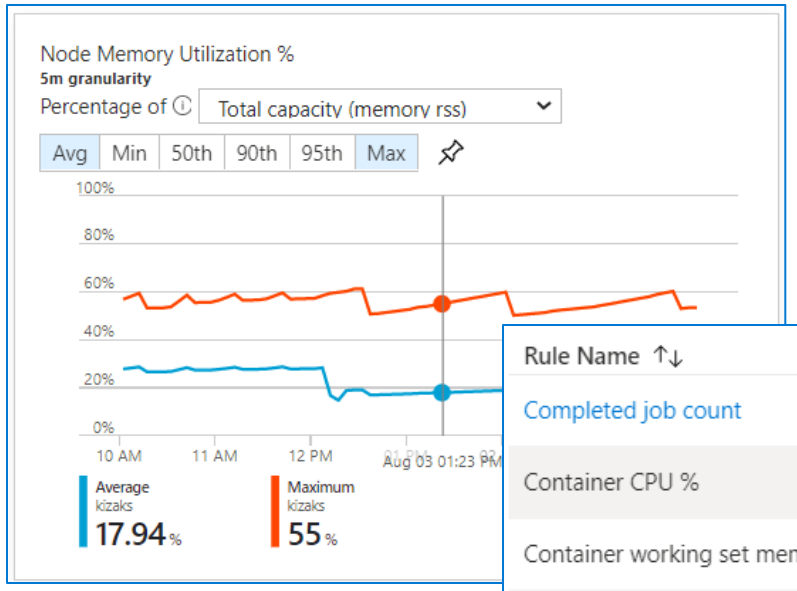
- Azure ***Container Insights*** is a part of Azure Monitor and integrates with AKS.



- See [How to query logs from Container Insights] for sample Kusto queries.

Azure Container Insights Dashboard

Container Insights allows you observe many aspects of your cluster and create alerts for critical events.



Name	Status	95th %	95th	Containers	UpTime	Controller
aks-agentpool-35127589-vmss000002	Ok	6%	116 mc	17	22 days	-
Other Processes	-	0%	39 mc	-	-	-
konnectivity-agent-5fbbb799b6-ptm72	Ok	2%	2 mc	1	18 days	konnectivity-agen...
clusterinfo-6fb57fff9b-wz9dc	Ok	2%	32 mc	1	22 days	clusterinfo-6fb57f...
					1 day	omsagent
					6 days	stats-queue-5c9d...
					18 days	keda-operator-m...

Rule Name	Condition	Action Groups	Status
Completed job count	Number of completed jobs(more than 6 hours ...	No action group assigned	Disabled
Container CPU %	Average container CPU is greater than 95%	No action group assigned	Disabled
Container working set memory %	Average container working set memory is grea...	No action group assigned	Disabled
Failed Pod counts	Number of Pods in Failed state are greater tha...	No action group assigned	Disabled
Node CPU %	Average node CPU is greater than 80%	No action group assigned	Disabled
Node Disk Usage %	Average disk usage for a node is greater than ...	No action group assigned	Disabled
Node NotReady status	Number of nodes in not ready state are greate...	No action group assigned	Disabled

Azure Monitor Demo

Create a Log Analytics Workspace to Send AKS Metrics and Logs to.

Azure Kubernetes Service Demo

Create an Azure Kubernetes Service Instance using the Azure Portal

PRO TIP: Stop/Start an AKS Cluster

- AKS workloads may not need to run continuously (for example a development cluster that is used only during business hours).
- To optimize costs during these periods, you can completely turn off (stop) an AKS cluster, which will delete all its Nodes.
- Stopping a cluster saves on all the compute costs, while maintaining all objects and cluster state for when it starts up again.
- **Example**: To stop/start a cluster:

```
az aks stop/start --name mycluster --resource-group mygroup
```

NOTE: Repeatedly starting/stopping an AKS cluster may result in errors. Once a cluster is stopped, wait at least 15-30 minutes before starting it up again.

Lab – Module 2

Create an Azure Kubernetes Service Cluster

Microsoft

Thank you