Microsoft Azure

Kubernetes on Azure

Learn about Kubernetes benefits, challenges, and enhancements made possible from a managed platform. Get the most out of Azure Kubernetes Service (AKS) with top scenarios, Azure capabilities, and tools.

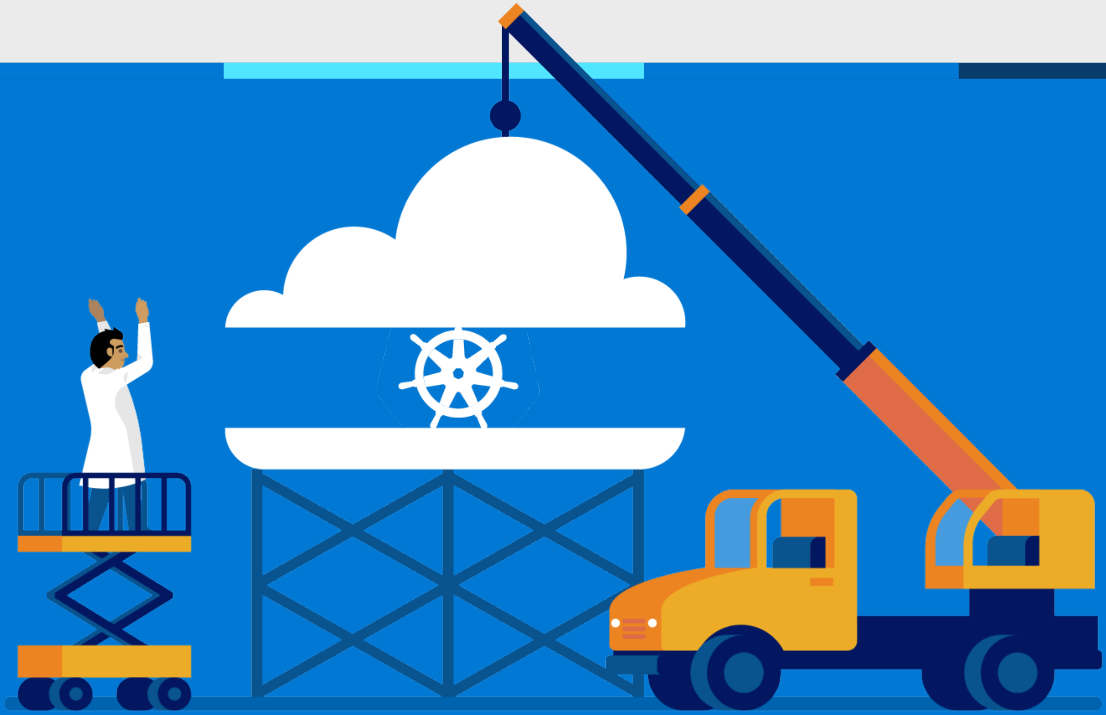
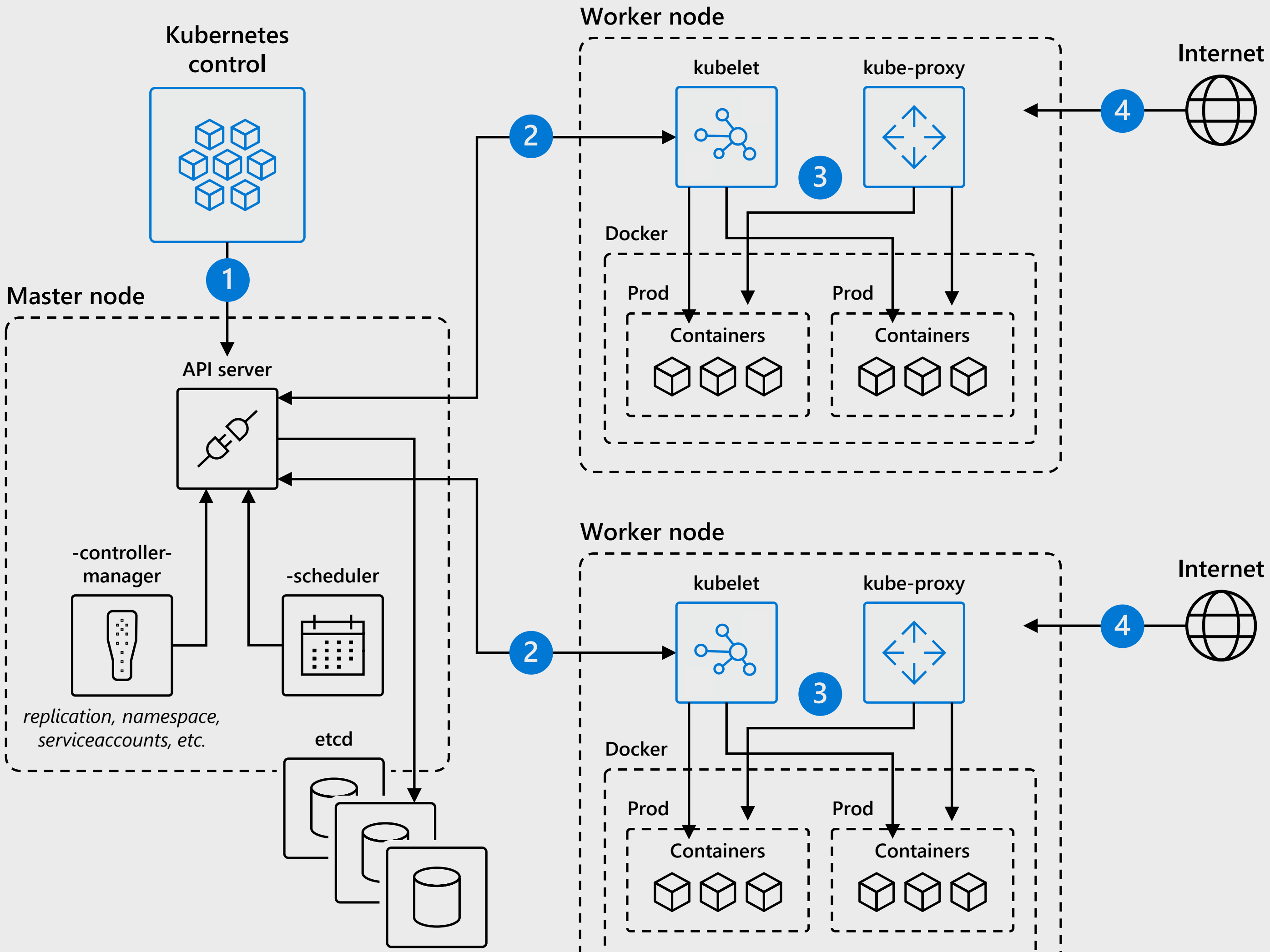


Table of contents

How Kubernetes works

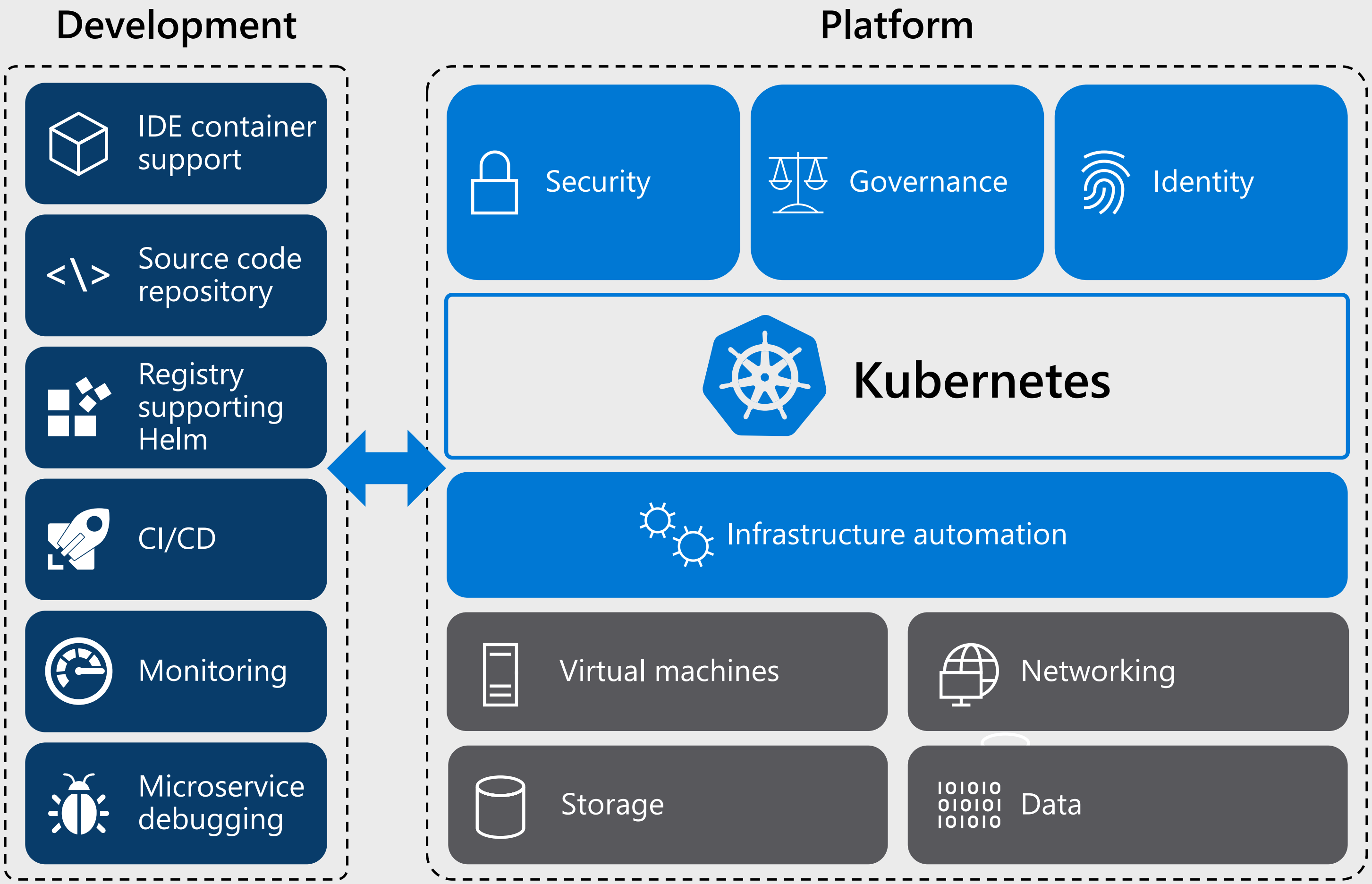
1. Kubernetes users communicate with API server and apply desired state
2. Master nodes actively enforce desired state on worker nodes
3. Worker nodes support communication between containers
4. Worker nodes support communication from the Internet


Kubernetes control
1
Master node
API server
-controller-manager
-scheduler
replication, namespace, serviceaccounts, etc.
etcd
Worker node
kubelet
kube-proxy
2
3
4
Internet
Docker
Prod
Containers
Prod
Containers
Worker node
kubelet
kube-proxy
2
3
4
Internet
Docker
Prod
Containers
Prod
Containers

But Kubernetes on its own is not enough

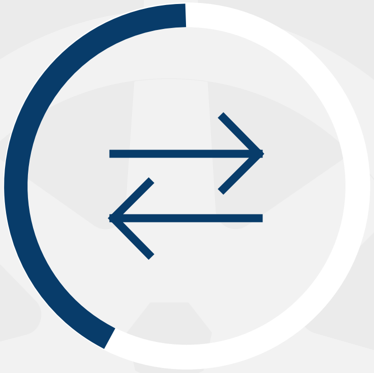
- Save time from infrastructure management and roll out updates faster without compromising security
- Unlock the agility for containerized applications using:
  - Infrastructure automation that simplifies provisioning, patching, and upgrading
  - Tools for containerized app development and CI/CD workflows
  - Services that support security, governance, and identity and access management

Learn more at
aka.ms/k8slearning

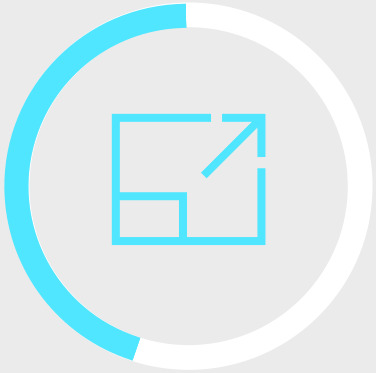

Development
Platform
IDE container support
Source code repository
Registry supporting Helm
CI/CD
Monitoring
Microservice debugging
Security
Governance
Identity
Kubernetes
Infrastructure automation
Virtual machines
Networking
Storage
Data

What’s behind the Kubernetes growth?

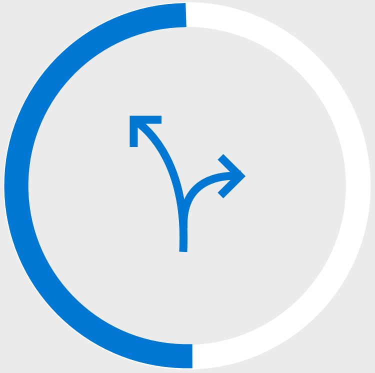
The perceived developer benefits of Kubernetes



42%
portability



45%
scalability



50%
agility

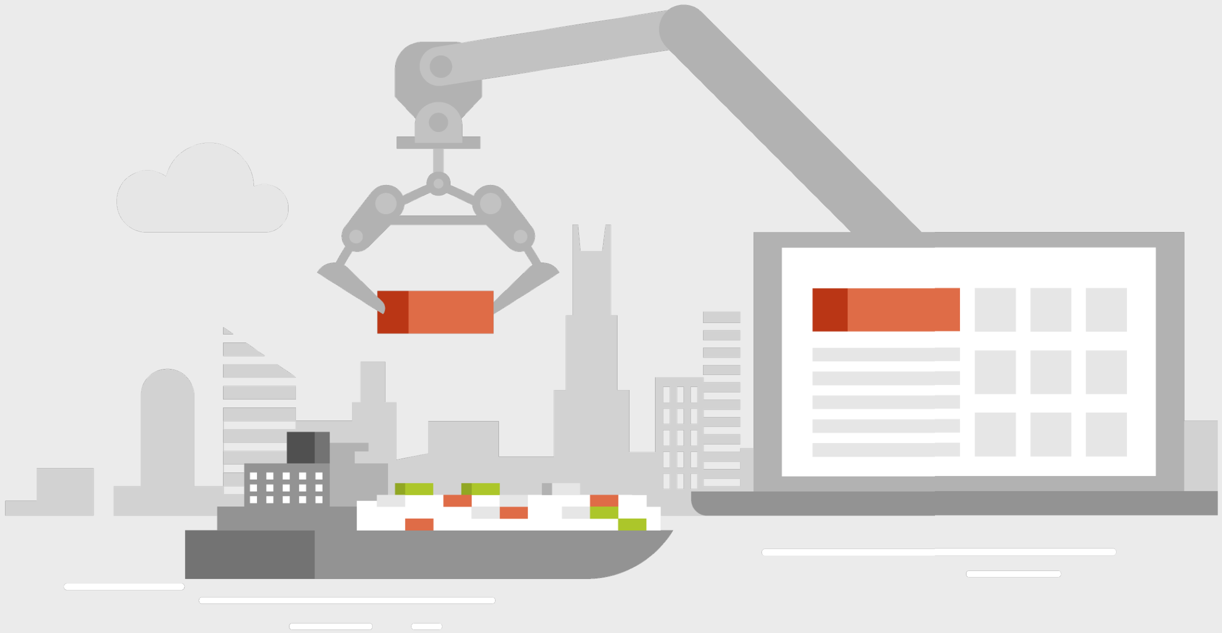
Azure Kubernetes Service (AKS) momentum

AKS **usage grew 30x** since it was made generally available in June 2018

Trusted by thousands of customers

J.B. HUNT
Hafslund
MAERSK
AMBIT ENERGY
BOSCH
ELASTACLOUD
webjet.com.au
WhiteSource
falkonry
Aporeto
Munich RE welt hunger hilfe
FINASTRA
Altair
SOTETSU
xerox
SIEMENS Healthineers
NETSHOES
eLynx Technologies
varian
NAMOGOO
きりんカルテシステム株式会社
Tryg
equinor
nyris

Infrastructure automation



Kubernetes gives you the knobs to schedule and deploy containers across clusters, scale to your desired state, and manage the Kubernetes lifecycle to keep your apps up and running.

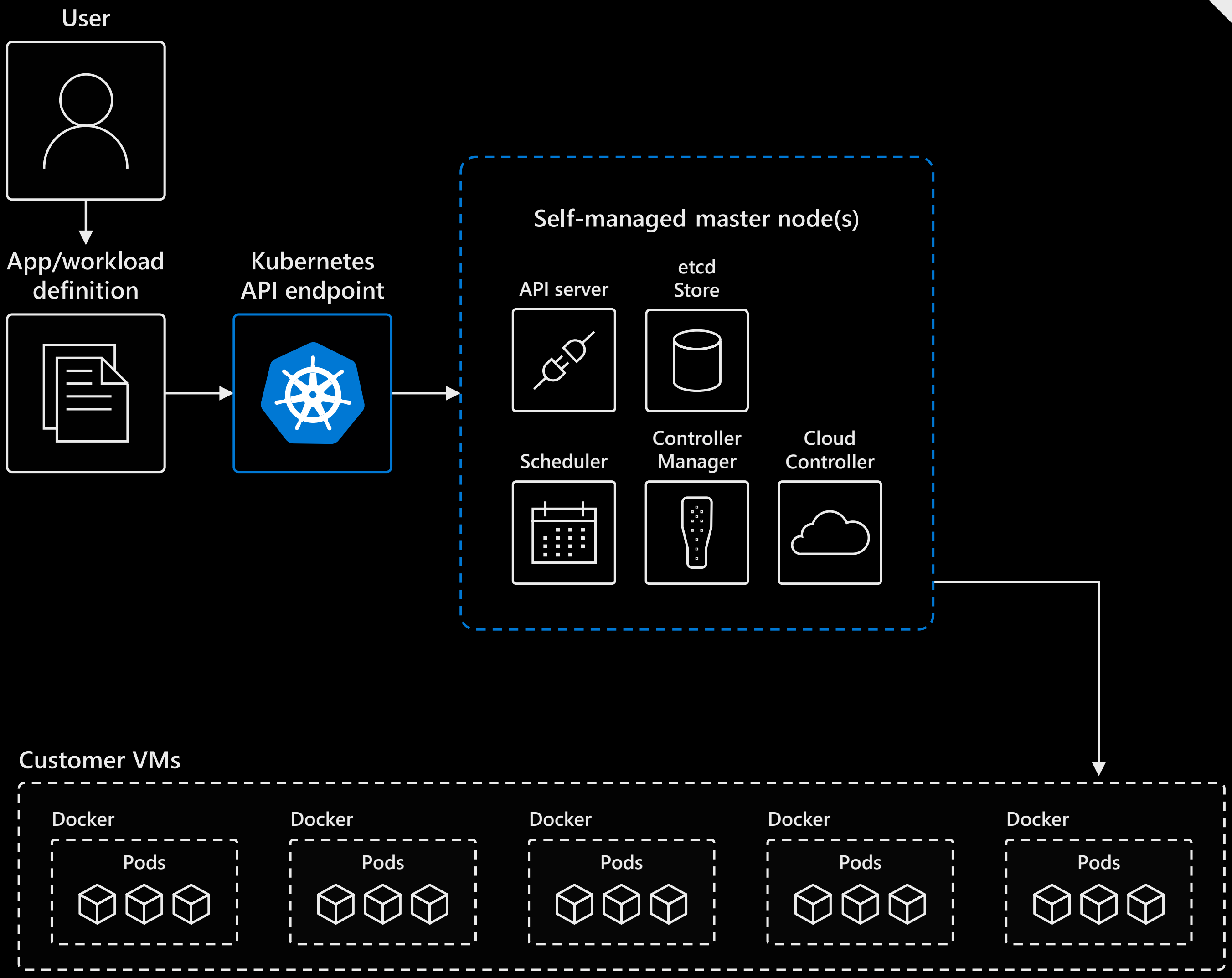
As your applications move to production, they often span multiple containers, deployed across a cluster of servers—increasing the complexity of operating the knobs and taking up time you could be spending delivering value to your customers.

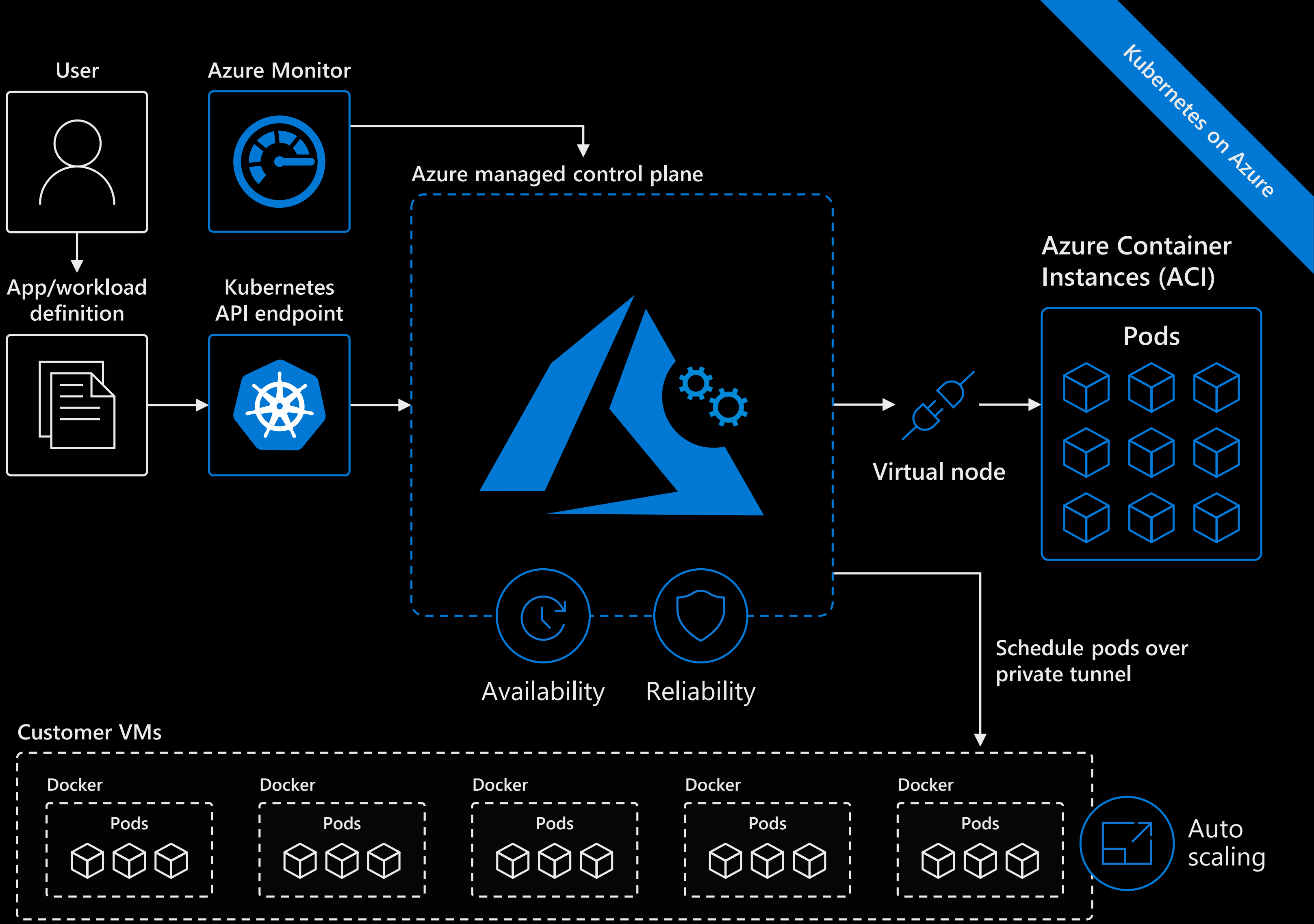
A fully managed Kubernetes service, like Azure Kubernetes Service (AKS), **automates provisioning, upgrading, monitoring, and scaling for compute resources**.

Manage Kubernetes with ease

- Automated provisioning, upgrades, and patches
- High reliability and availability
- Serverless scaling
- API server monitoring
- Delivered at no charge

Kubernetes out-of-the-box




Kubernetes on Azure
User
Azure Monitor
Azure managed control plane
App/workload definition
Kubernetes API endpoint
Azure Container Instances (ACI)
Pods
Virtual node
Availability
Reliability
Schedule pods over private tunnel
Customer VMs
Docker
Pods
Docker
Pods
Docker
Pods
Docker
Pods
Docker
Pods
Auto scaling

“Thanks to AKS, we can now spin up new demo environments in 10 minutes instead of 24 hours. Moving Docushare Flex from virtual machines to containers in Azure allows us to provision environments faster, empowering our sales and partner network.”

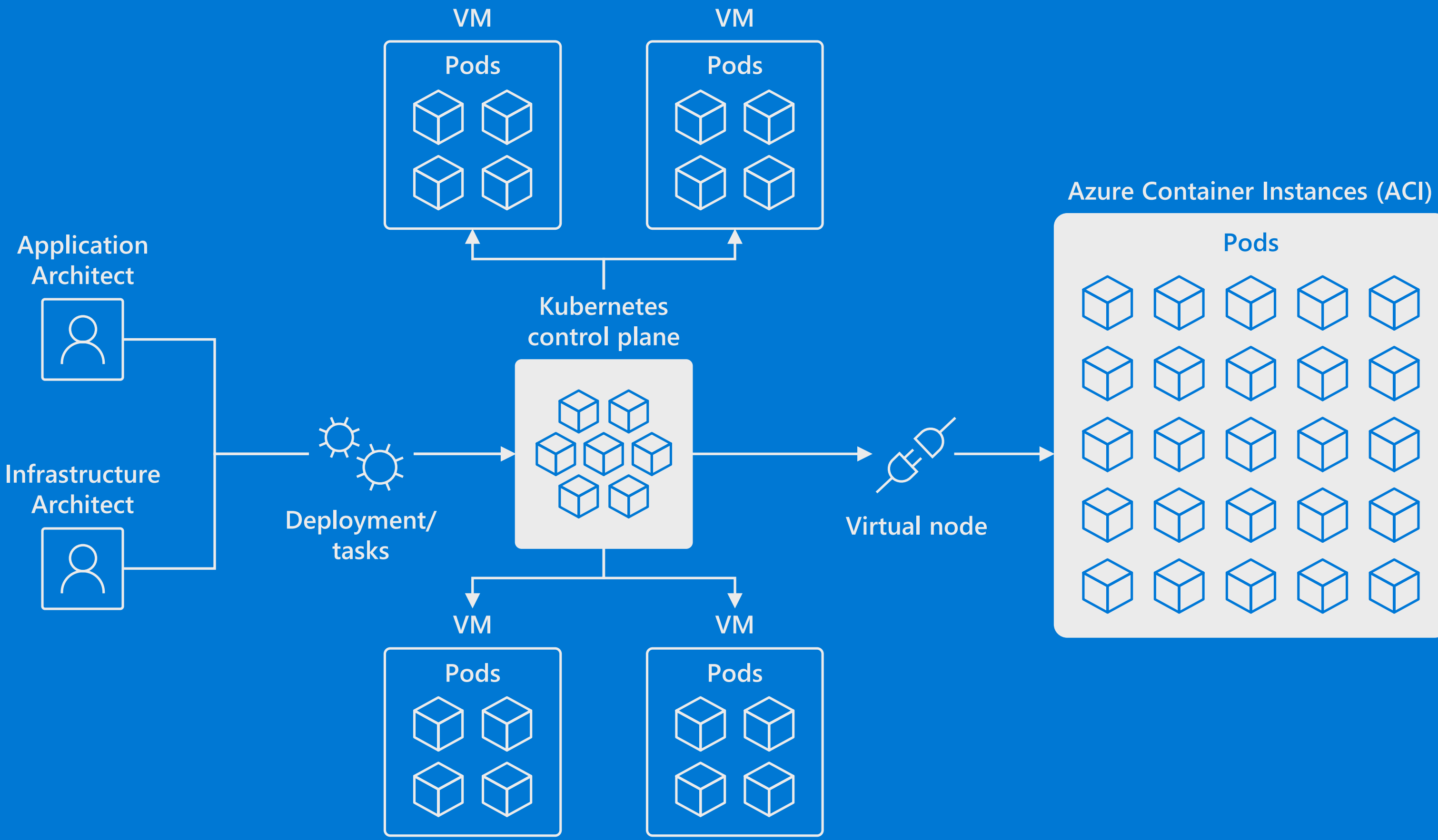
— Robert Bingham, Director of DocuShare Cloud Operation
Xerox

Capability

Virtual node

- Elastically provision capacity in seconds
- No infrastructure to manage
- Built on open-sourced Virtual Kubelet technology, a sandbox project from CNCF

Learn more at
aka.ms/aksbook/virtualnode

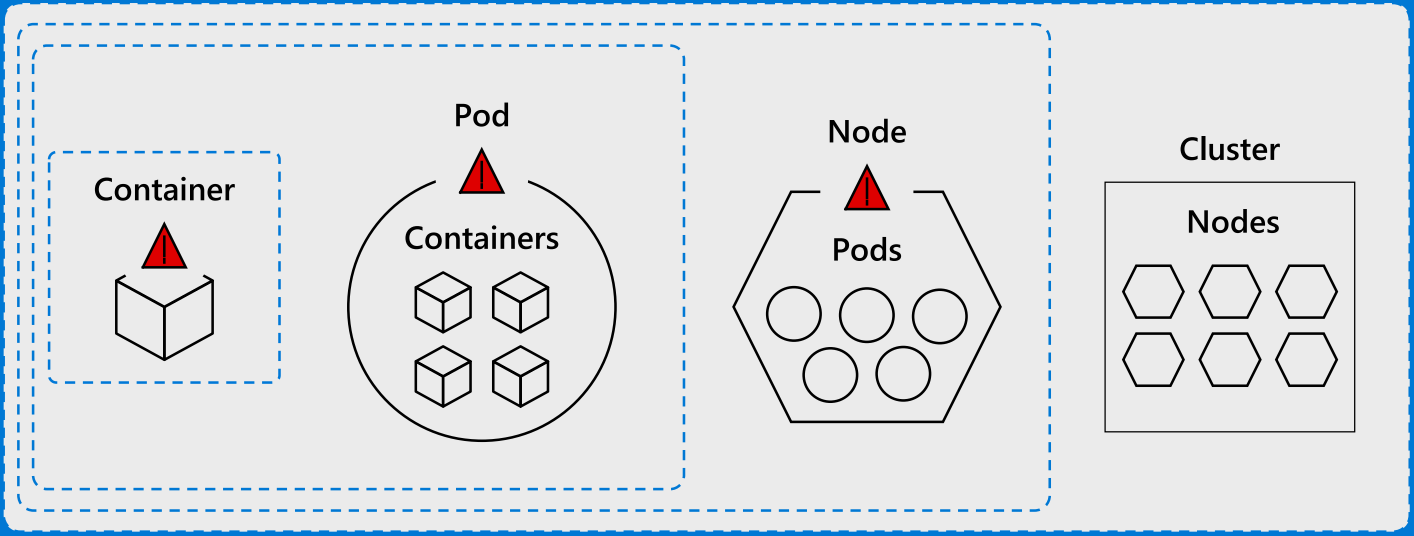

VM
Pods
VM
Pods
Azure Container Instances (ACI)
Pods
Application Architect
Kubernetes control plane
Infrastructure Architect
Deployment/ tasks
Virtual node
VM
Pods
VM
Pods

Auto scale

- Efficiently scale and run apps without downtime—all out of the box
- Automatically add or remove instances based on resource utilization

Learn more at
aka.ms/aksbook/autoscale

Out-of-the-box cluster autoscaling



Automatically spin up more... → Pods → Containers → Clusters

⚠ = exhausted

End-to-end developer experience

Kubernetes API itself doesn’t include development tools. To run an application in a Kubernetes cluster, a developer may use a code editor to write code and perhaps a source code control repository to manage it; a Docker client to help with containerization; Helm for packaging; and kubectl, or a YAML configuration to deploy containers to Kubernetes.

In a real-world scenario, the picture becomes much more complicated. As containers, environments, and the teams that work with them multiply, release frequency increases—along with developmental and operational complexity. For example, the need to merge code effectively with the option to rollback; testing the application in a way that mimics the production environment but doesn’t impact the production environment; and, quickly identifying and addressing any issues without downtime. The last thing you want to have on top of this complexity is a fragmented tool chain.

A managed Kubernetes platform designed for developers can **integrate seamlessly with your favorite IDE, CI/CD, and monitoring tools and automate these workflows to support your Kubernetes app development**. An IDE that directly supports Kubernetes deployment can help you set up the most complex microservices development environment and connect with your private container registry. With built-in CI/CD and a pre-configured deployment strategy, you can accelerate the move from code to container to Kubernetes cluster in minutes by automating those tasks. Finally, a complete view from container health monitoring to centralized logging can be auto-configured with your developer portal to prevent resource bottlenecks, trace malicious requests, and keep your Kubernetes applications healthy.

Accelerate containerized development

Develop

- Native containers and Kubernetes support in IDE
- Remote debugging and iteration for multi-containers
- Effective code merge
- Automatic containerization

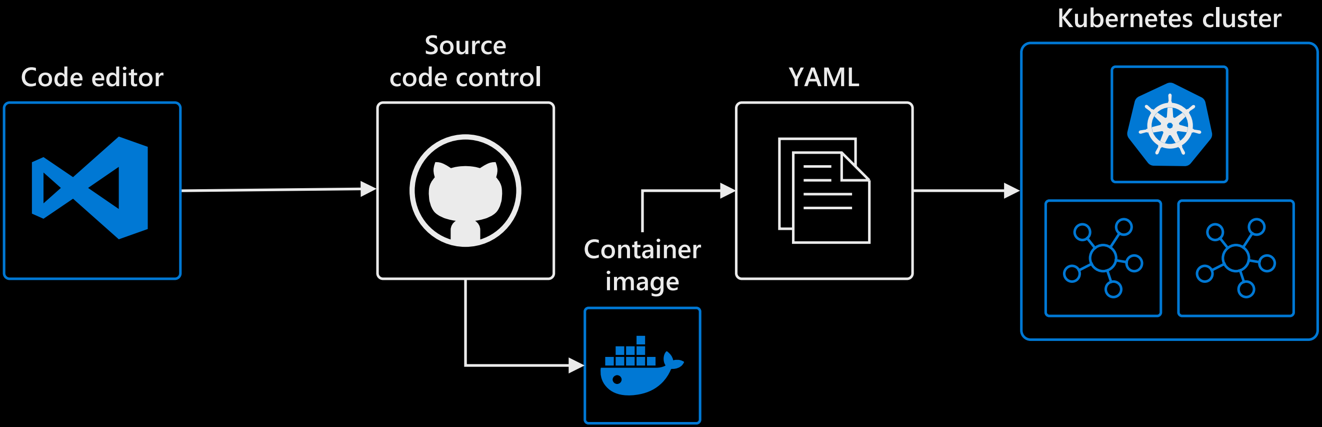
Deliver

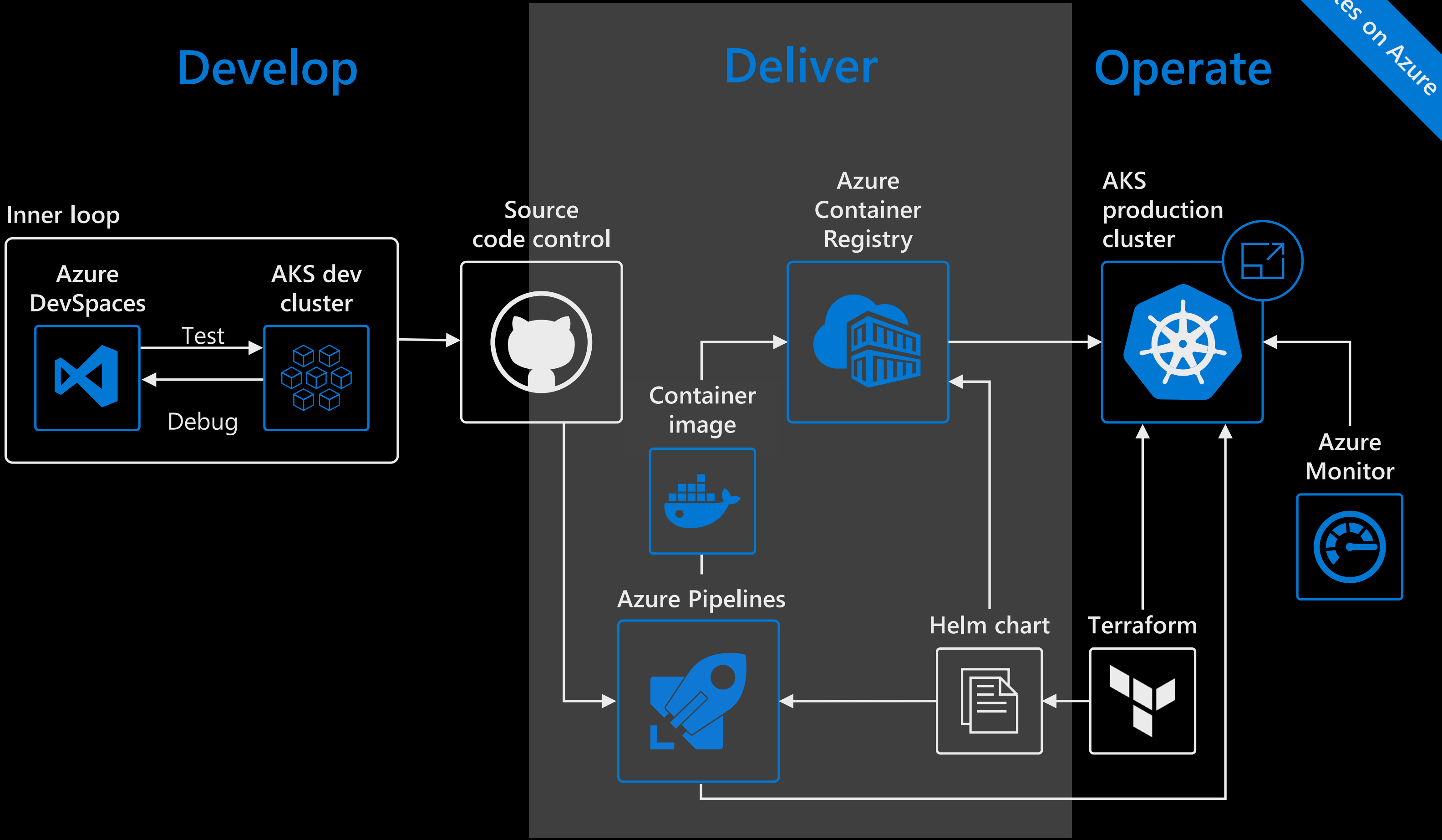
- CI/CD pipeline with automated tasks in a few clicks
- Pre-configured canary deployment strategy
- In depth build and delivery process review and integration testing
- Private registry with both container image and Helm chart management

Operate

- Out-of-box control plane telemetry, log aggregation, and container health
- Declarative resource management
- Auto scaling

Kubernetes out-of-the-box




Develop
Deliver
Operate
Inner loop
Azure DevSpaces
AKS dev cluster
Test
Debug
Source code control
Container image
Azure Container Registry
Azure Pipelines
Helm chart
Terraform
AKS production cluster
Azure Monitor

“We are building our own new applications using microservices—and AKS is our choice for orchestrating their workloads.”

— Ståle Heitmann, Chief Technology Officer
Hafslund Nett

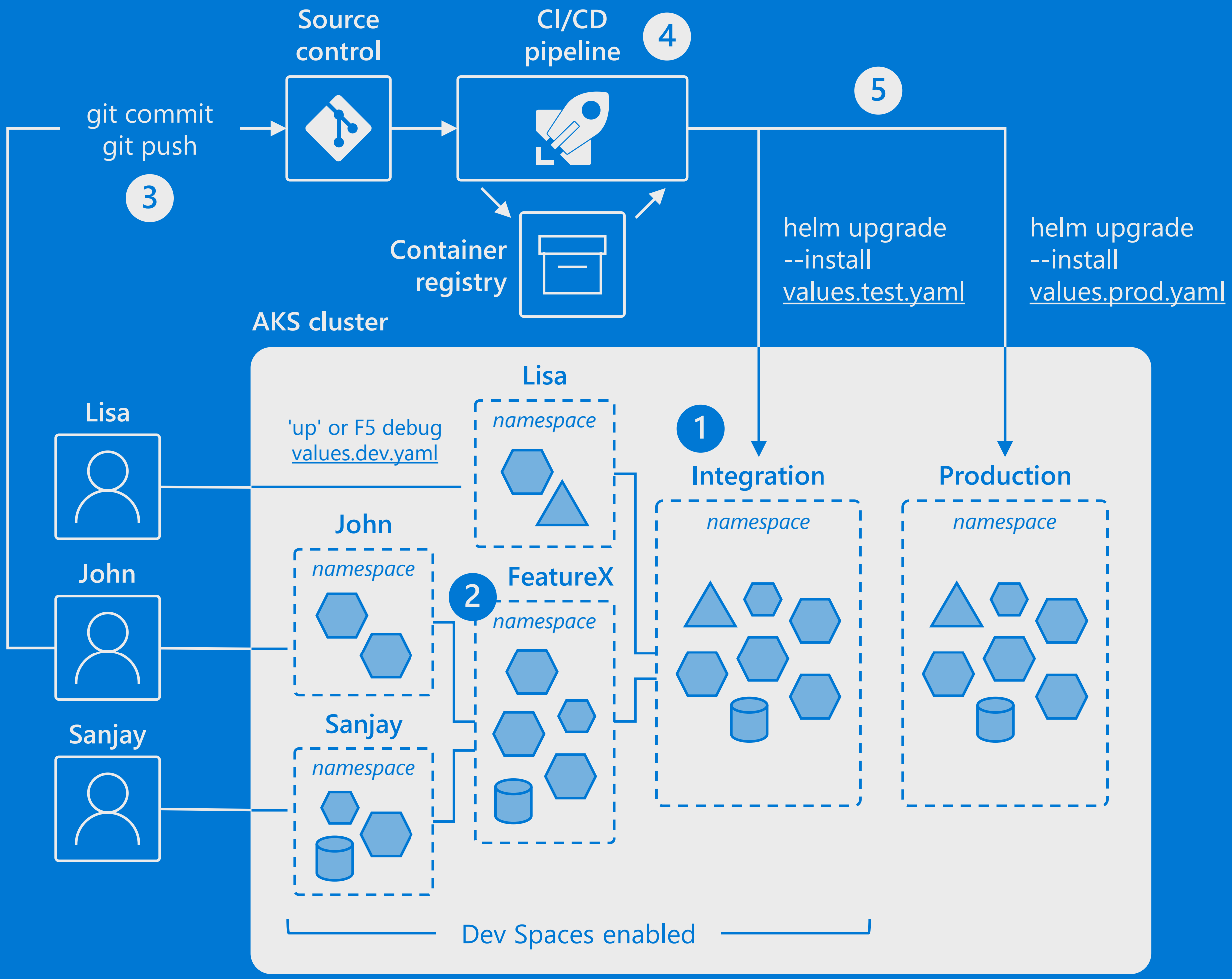
Hafslund
Nett

Azure Dev Spaces

1. “Integration” dev space runs full baseline version of app
2. John and Sanjay collaborate on FeatureX
3. Code committed to the master source control
4. CI/CD pipeline triggered to deploy into “Integration
5. Helm assets used in later environments by CD system

Dev Spaces is enabled per Kubernetes namespaces and can be defined as anything. Any namespace in which Dev Spaces is ***not*** *enabled runs unaffected.*

Learn more at
aka.ms/aksbook/devspaces

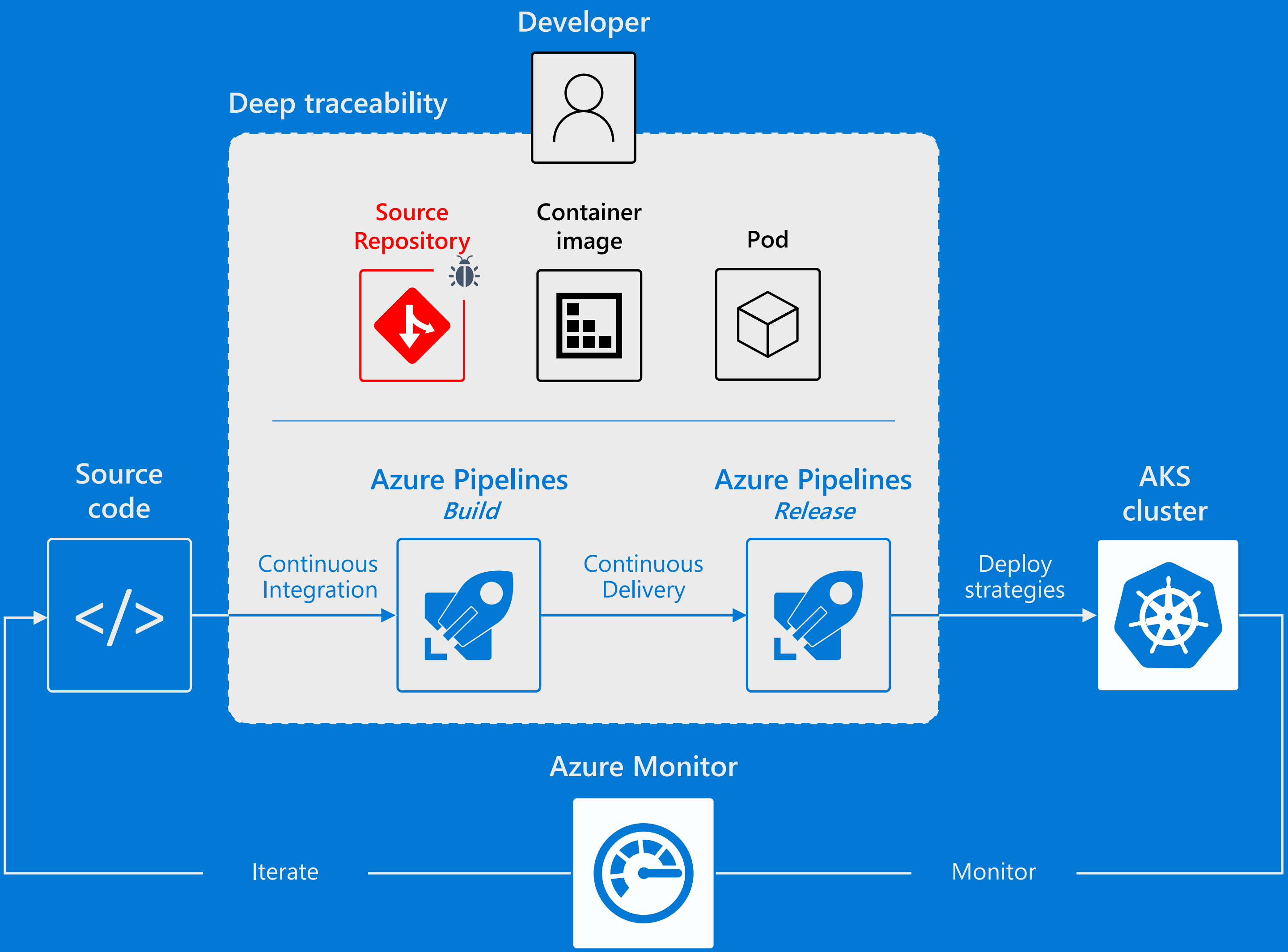

Source control
CI/CD pipeline
4
5
git commit
git push
3
Container registry
helm upgrade --install values.test.yaml
helm upgrade --install values.prod.yaml
AKS cluster
Lisa
'up' or F5 debug values.dev.yaml
Lisa
namespace
1
Integration
Production
namespace
namespace
John
John
namespace
2
FeatureX
namespace
Sanjay
Sanjay
namespace
Dev Spaces enabled

Capability

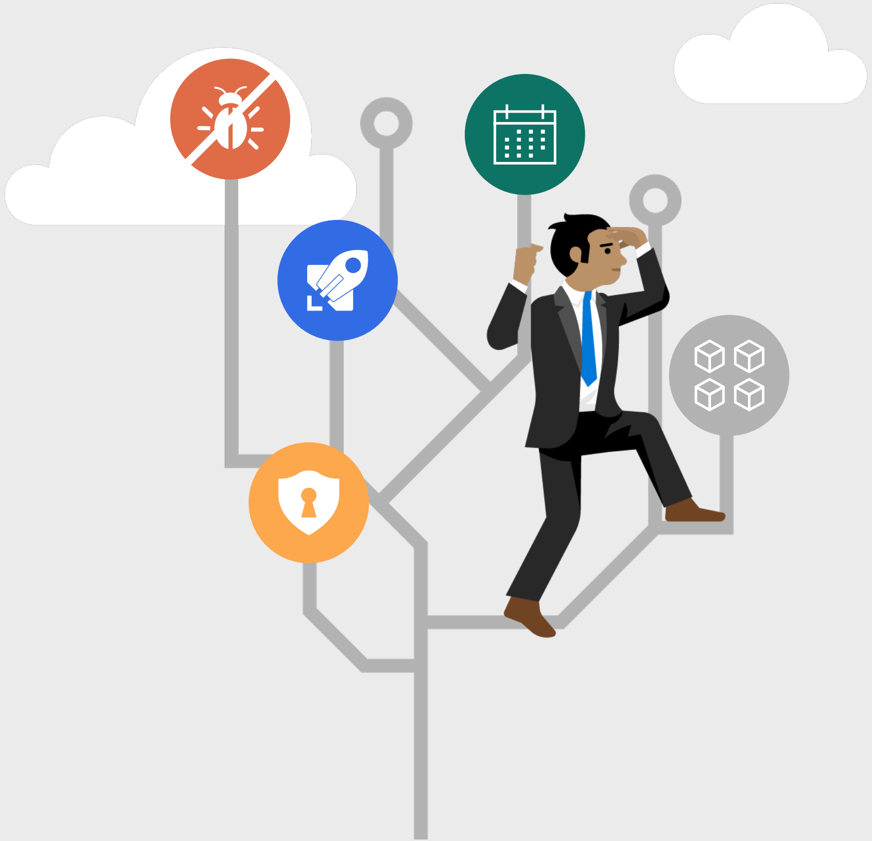
Azure Pipelines for AKS

- Add a full CI/CD pipeline to your AKS cluster with automated routine tasks and multiple deployment strategies—all set up in just a few clicks
- Detect failures early and optimize your pipelines in a heartbeat with deep traceability into your deployments and source code

Learn more at
aka.ms/aksbook/pipelines


Developer
Deep traceability
Source Repository
Container image
Pod
Source code
Azure Pipelines
Build
Azure Pipelines
Release
AKS cluster
Continuous Integration
Continuous Delivery
Deploy strategies
Azure Monitor
Iterate
Monitor

Balancing agility and security



Kubernetes provides built-in capabilities like namespaces and admission controller to help with isolation and privilege management for your Kubernetes resources. But to achieve hardened security and meet compliance requirements, your applications need more in-depth defense and dynamic control that goes beyond Kubernetes itself.

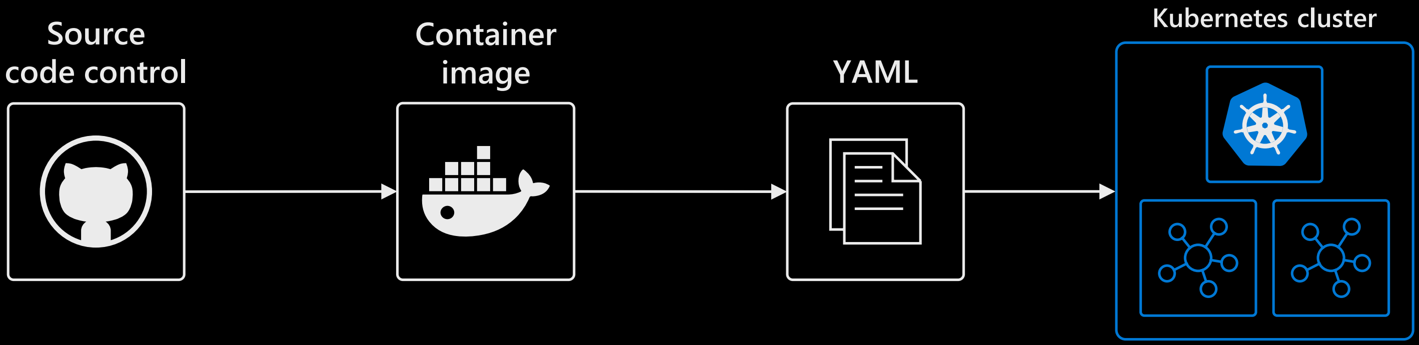
As container images become the new deployment format, the ecosystem around their security and controls is starting to emerge. Still, enforcing security and compliance without hindering agility is challenging and prone to error. The complexity lies in both development and infrastructure operations. For example, how do you embed policy requirements of your organization while the images are getting built and deployed as part of the CI/CD workflows?

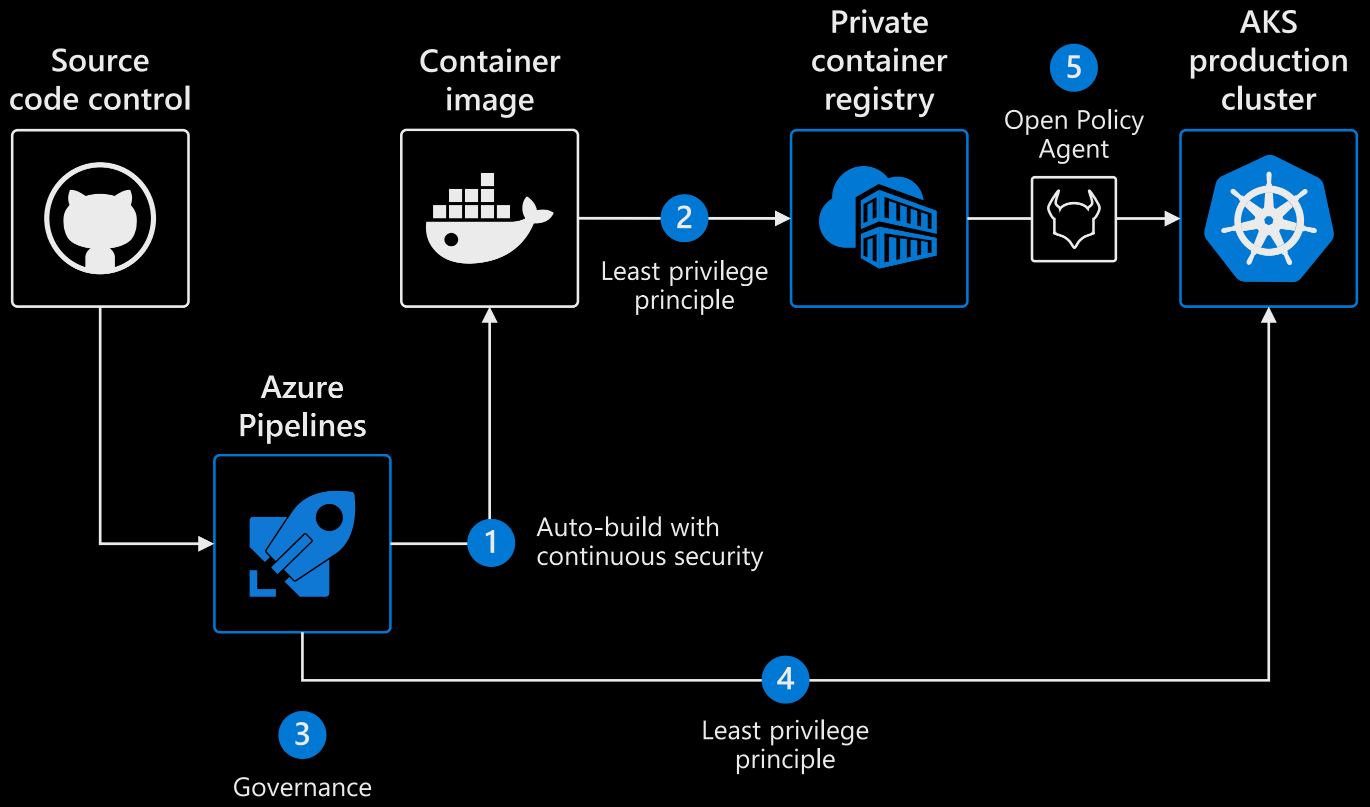
An enterprise-grade platform designed for developers can provide cloud services that offer deep, real-time observability for your build and release pipelines, and apply compliance audit and reconfigurations easily—all as part of the DevOps workflow.

Put guardrails around the development process

1. **Auto-build with continuous security**: Enforce pre-defined policies to build your pipeline
2. **Least privilege principle**: Only build pipelines that have the key/permission to push image into registry
3. **Governance**: Add policy audit to your pipeline—non-compliant releases will be flagged for review and action
4. **Least privilege principle**: Only the release pipeline has permission to create new pods or new applications in your Kubernetes environment
5. **Open Policy Agent**: Only images from trusted registries will get deployed and executed in the cluster

Kubernetes out-of-the-box




Source code control
Container image
Private container registry
5
Open Policy Agent
AKS production cluster
2
Least privilege principle
Azure Pipelines
1
Auto-build with continuous security
4
3
Governance
Least privilege principle

“Using Kubernetes on Azure satisfies our objectives for efficient software development. It aligns well with our digital plans and our choice of open-source solutions for specific programming languages.”

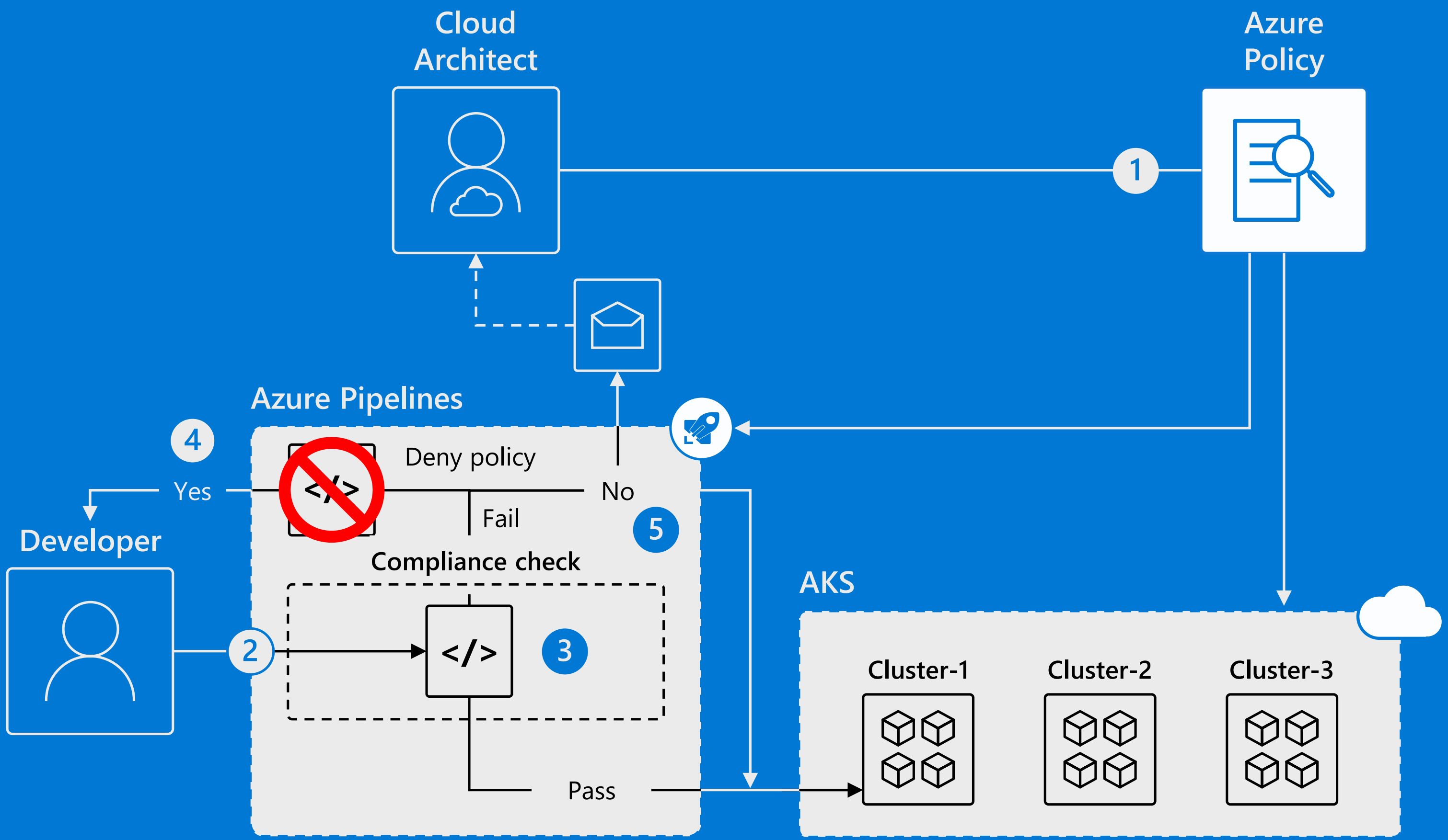
— Rasmus Hald, Head of Cloud Architecture
A.P. Moller - Maersk

MAERSK

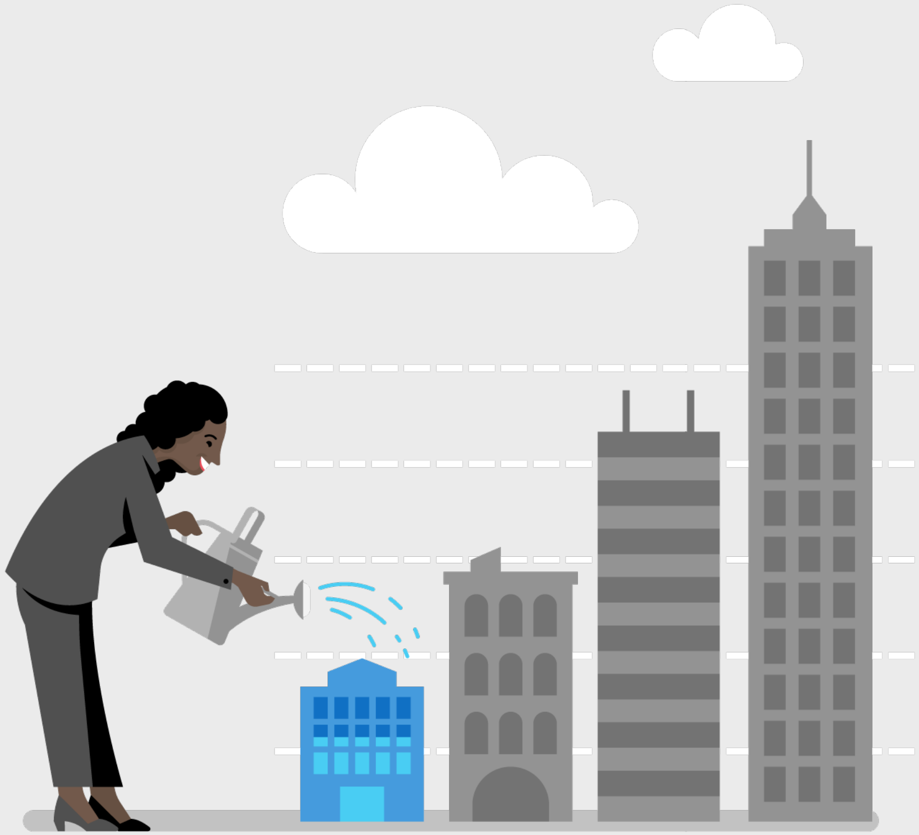
Azure Pipelines to deliver; Azure Policy to enforce

1. Cloud architect assigns a policy across clusters; policy can be set to block non-compliance (deny) or generate non-compliance warnings (audit)
2. Developer makes code change that kicks off an Azure Pipelines build
3. Azure Pipelines evaluates the request for policy compliance
4. If policy is set to deny, Azure Pipelines rejects the build attempt if any non-compliance is identified
5. If policy is set to audit, a non-compliance event is logged and the build is allowed to proceed

Learn more at
aka.ms/aksbook/policy


Cloud Architect
Azure Policy
1
Azure Pipelines
4
Yes
Deny policy
No
Fail
5
Developer
Compliance check
2
</>
3
Pass
AKS
Cluster-1
Cluster-2
Cluster-3

Secure cluster setup



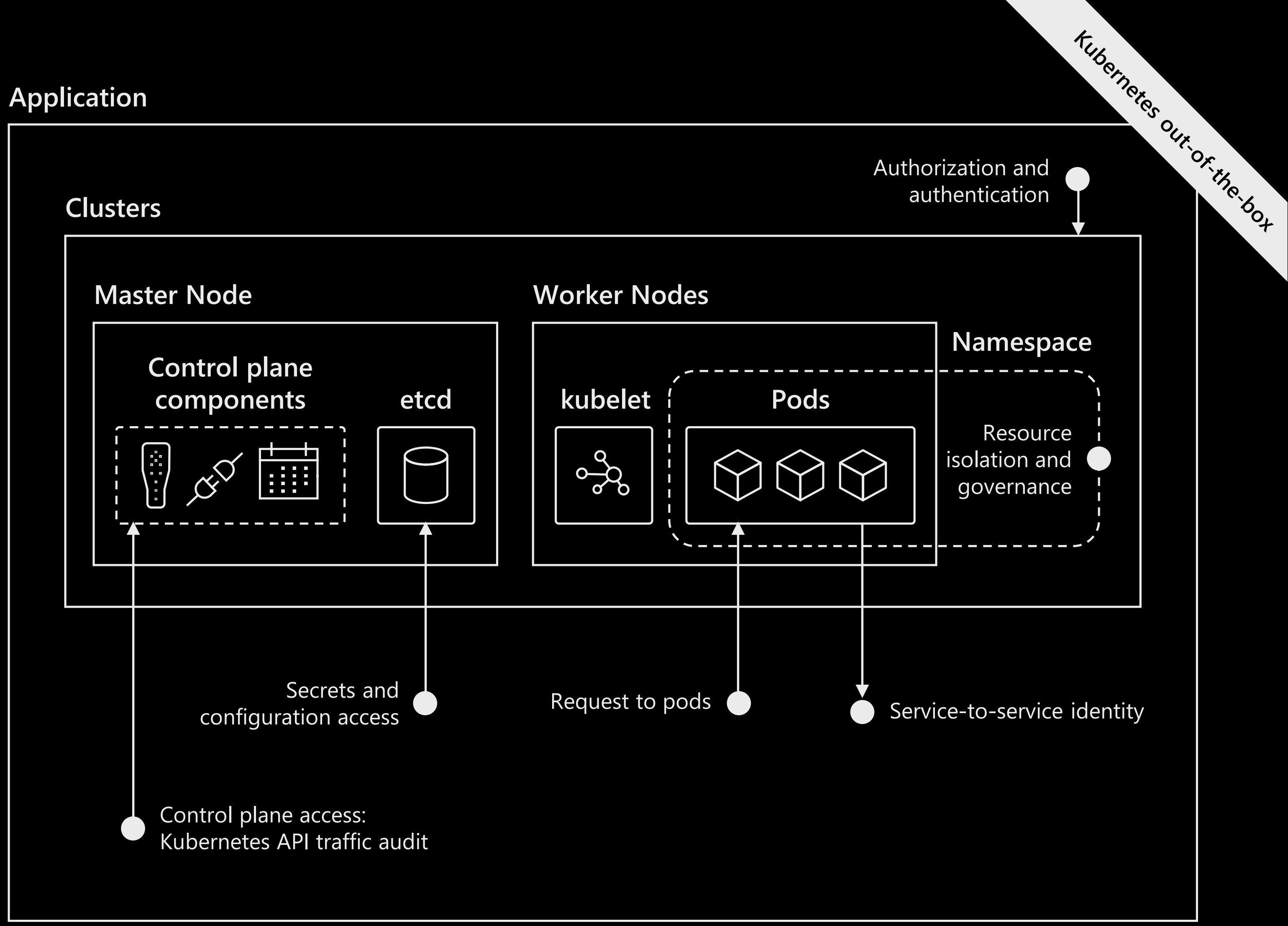
As a cloud-native container orchestration tool, Kubernetes provides various access points to its users. These include the API server and kubectl to access it via the command line, kubelet for interacting with the container runtime, and etcd storage for state and cluster information, just to name a few.

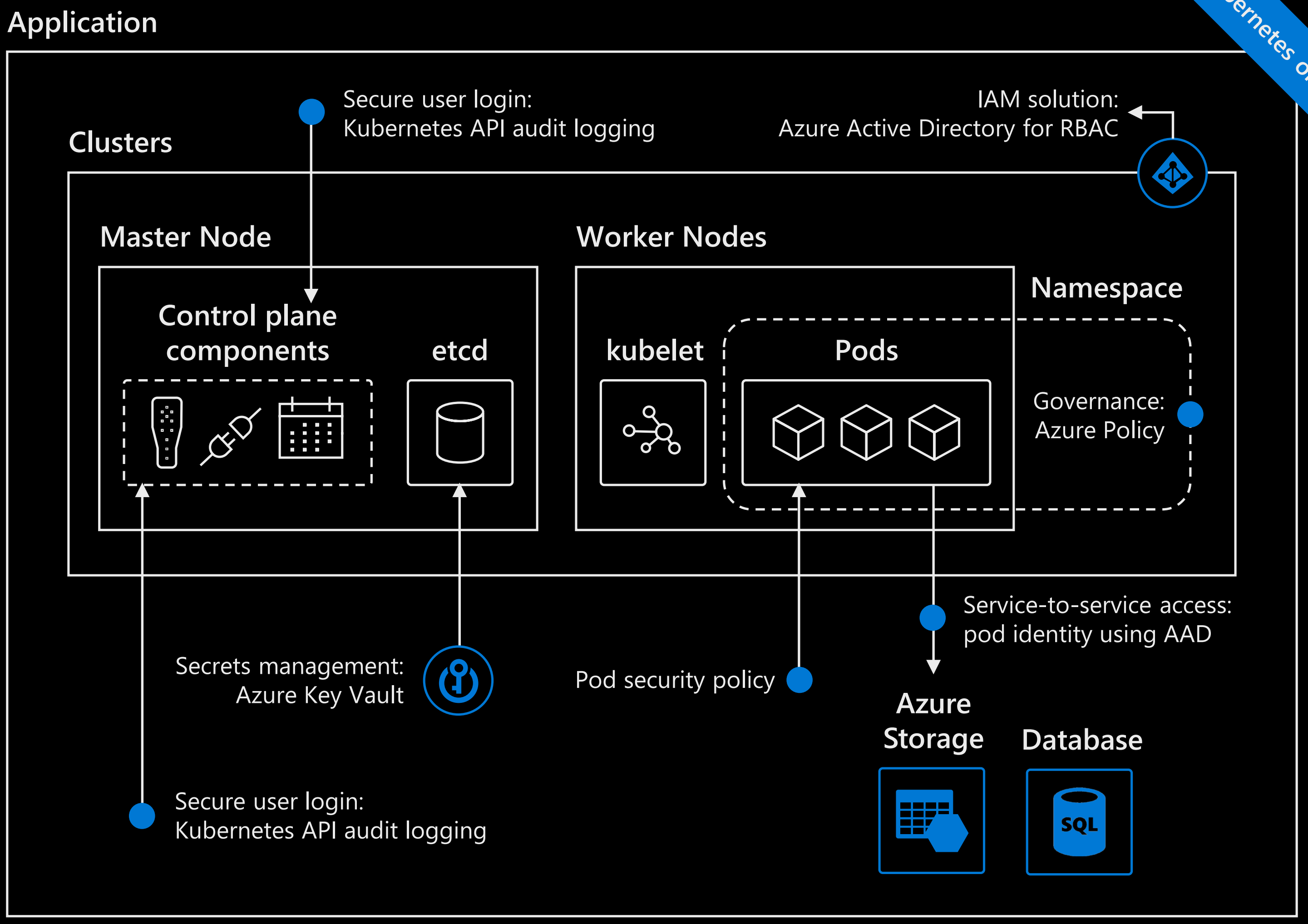
Malicious access to any of the above can be severely problematic. While you can use Kubernetes settings and associated best practices to manage security, production systems demand hardened security that goes beyond configurations and settings.

To secure your cluster, you want to **build on a secure, enterprise-grade platform that can easily incorporate solutions for identity and access management (IAM), secrets management, and policy enforcement** without introducing a steep learning curve for your team. For example, you can use Azure Active Directory to get fine-grained identity and access control to Kubernetes resources from cluster to containers, while Azure Policy can provide rules enforcement across multiple clusters.

Hardened security for Kubernetes resources

- Get secure login and fine-grained identity and access control to Kubernetes resources from cluster to containers
- Securely store and centrally manage secrets outside the cluster using **Azure Key Vault Flex Volumes**
- Validate requests to pods and define conditions required for pods to run in cluster using **Pod Security Policy**
- Enforce and synchronize access control with other services required for the application with identity for Kubernetes pods in the same IAM solution
- Record, monitor, investigate API calls for suspicious activities using audit logging
- Audit and enforce rules defined in **Azure Policy** across multiple clusters in real-time—powered by Open Policy Agent


Kubernetes out-of-the-box
Application
Clusters
Authorization and authentication
Master Node
Worker Nodes
Namespace
Control plane components
etcd
kubelet
Pods
Resource isolation and governance
Secrets and configuration access
Request to pods
Service-to-service identity
Control plane access: Kubernetes API traffic audit


Application
Clusters
Secure user login:
Kubernetes API audit logging
IAM solution:
Azure Active Directory for RBAC
Master Node
Worker Nodes
Namespace
Control plane components
etcd
kubelet
Pods
Governance:
Azure Policy
Service-to-service access:
pod identity using AAD
Secrets management:
Azure Key Vault
Pod security policy
Azure Storage
Database
SQL
Secure user login:
Kubernetes API audit logging

“Using Azure Kubernetes Service puts us into a position to not only deploy our business logic in Docker containers, including the orchestration, but also… to easily manage the exposure and control and meter the access.”

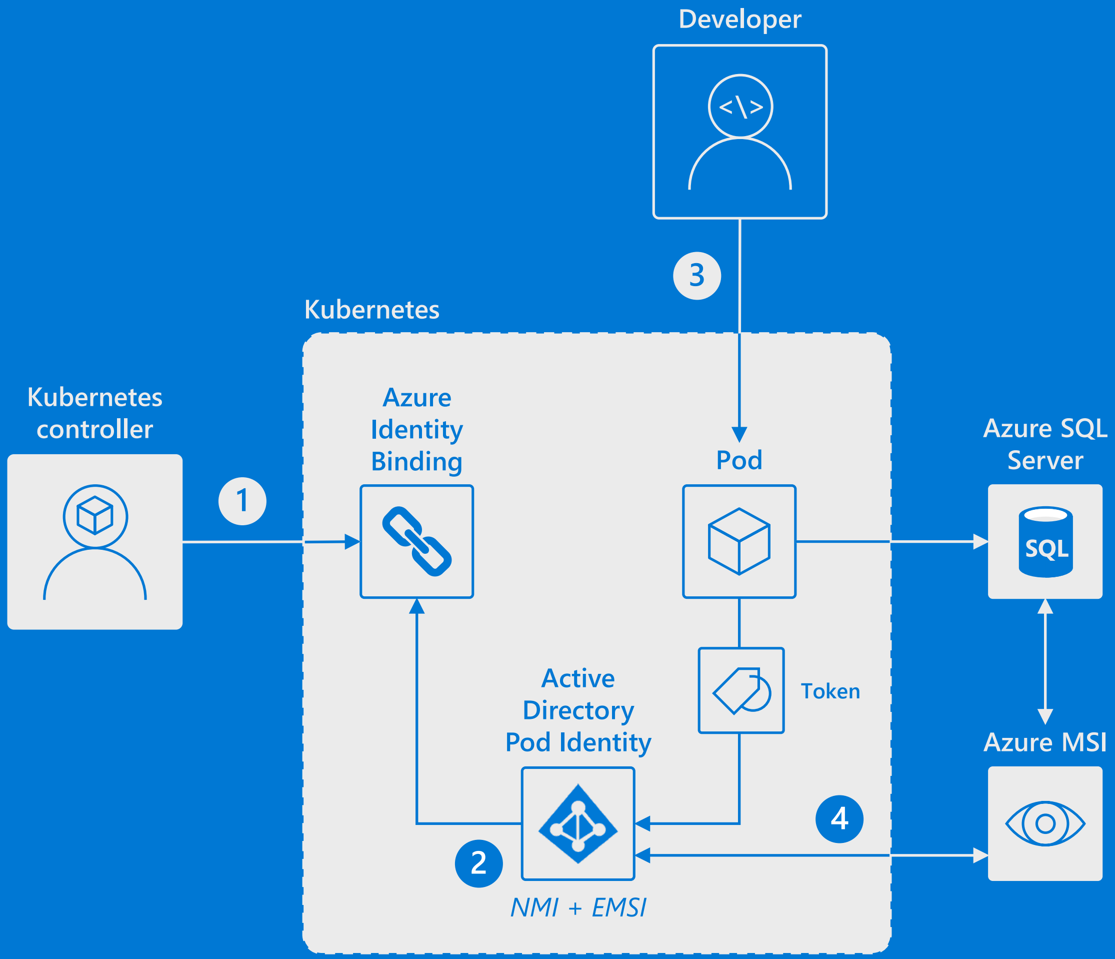
— Thomas Gossler, Lead Architect, Digital Ecosystem Platform
Siemens Healthineers

SIEMENS
Healthineers

Pod identity

1. Kubernetes operator defines an identity map for K8s service accounts
2. Node Managed Identity (NMI) watches for mapping reaction and syncs to Managed Service Identify (MSI)
3. Developer creates a pod with a service account, and pod uses standard Azure SDK to fetch a token bound to MSI
4. Pod uses access token to consume other Azure services; services validate token

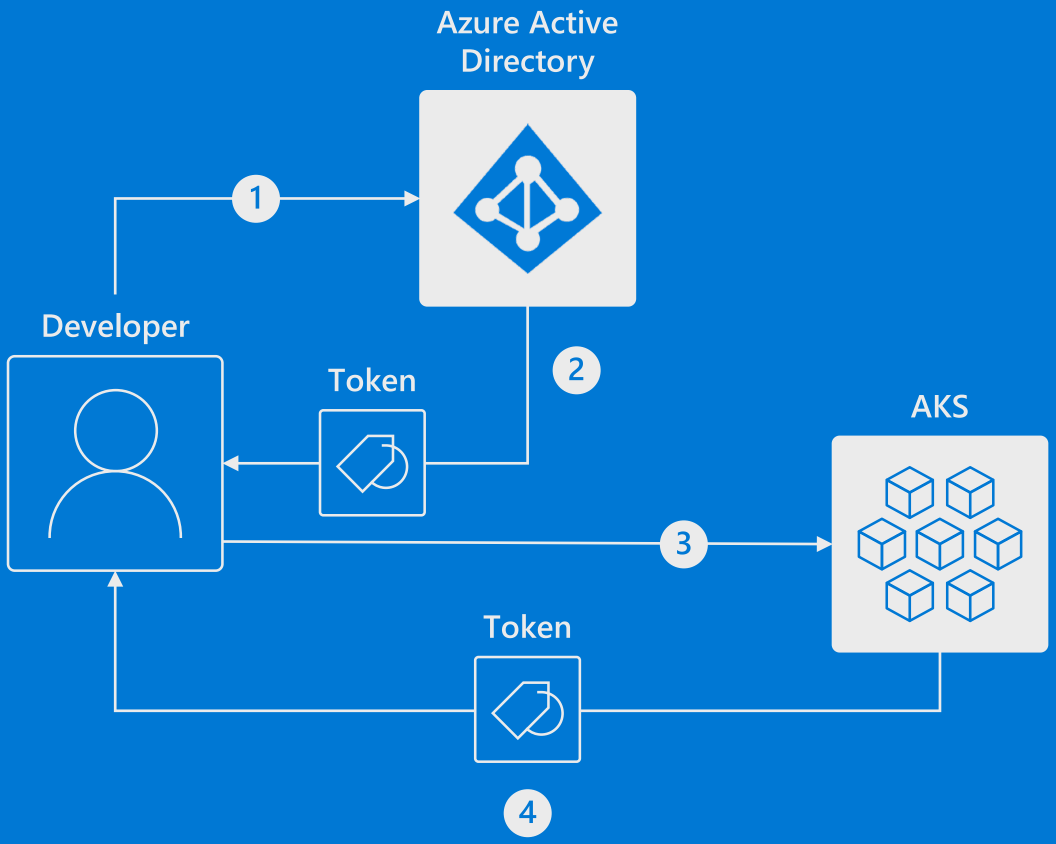
Learn more at
aka.ms/aksbook/podidentity


Developer
3
Kubernetes
Kubernetes controller
1
Azure Identity Binding
Pod
Azure SQL Server
SQL
Active Directory Pod Identity
Token
Azure MSI
4
2
NMI + EMSI

Identity and access management through AAD and RBAC

1. A developer authenticates to the AAD token issuance endpoint and requests an access token
2. The AAD token issuance endpoint issues the access token
3. The access token is used to authenticate to the secured resource
4. Data from the secured resource is returned to the web application

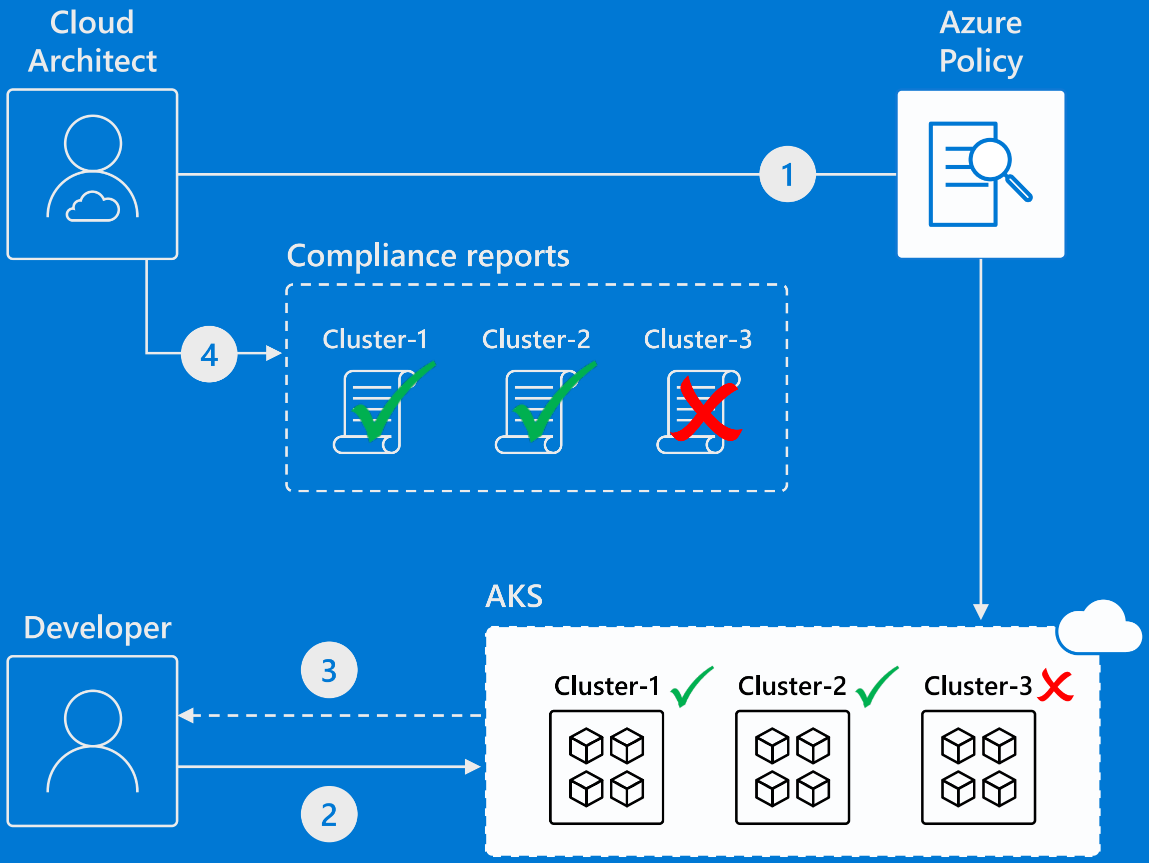
Learn more at
aka.ms/aksbook/aad


Azure Active Directory
1
Developer
Token
2
AKS
3
Token
4

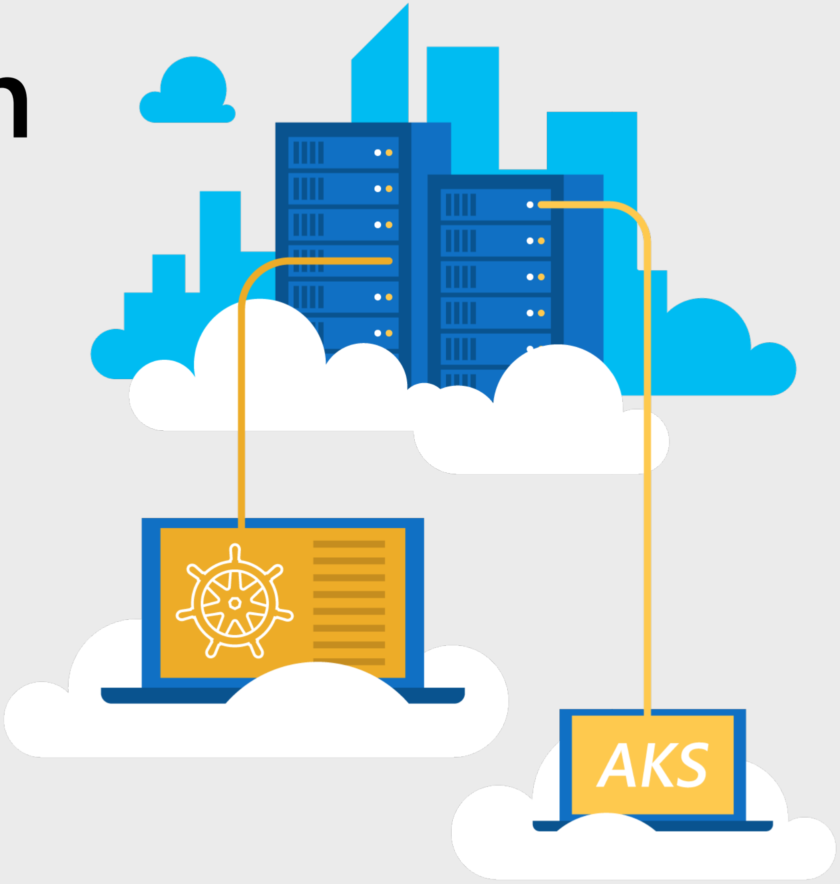
Azure Policy for Kubernetes clusters

1. Cloud architect assigns a deployment policy across cluster(s)
2. Developer uses standard Kubernetes API to deploy to the cluster
3. Real-time deployment enforcement (acceptance/denial) provided to developer based on policy
4. Cloud architect obtains compliance report for the entire environment and can drill down to individual pod level

Learn more at
aka.ms/aksbook/policy


Cloud Architect
Azure Policy
1
Compliance reports
4
Cluster-1
Cluster-2
Cluster-3
AKS
Developer
3
2
Cluster-1
Cluster-2
Cluster-3

Network segmentation



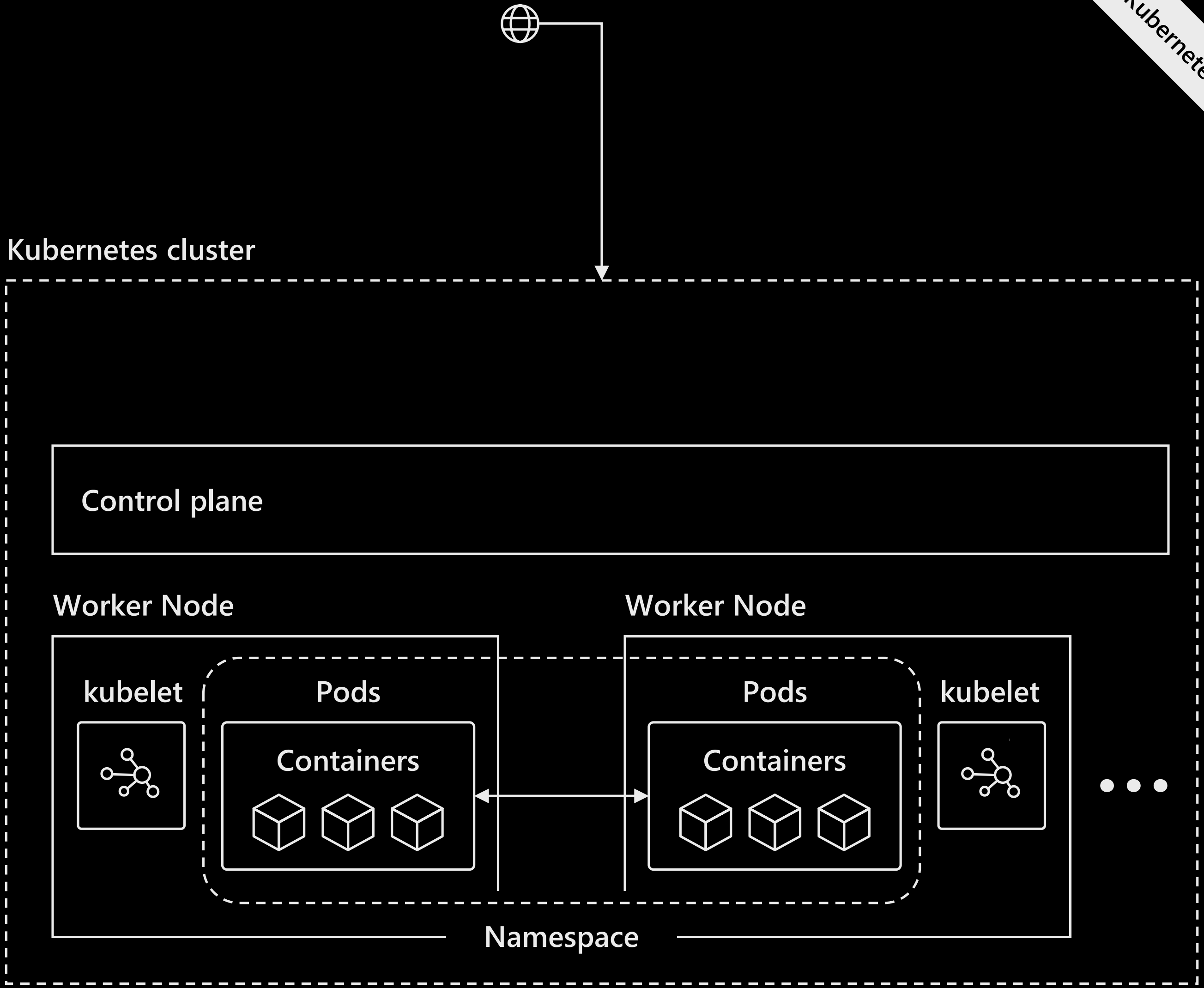
As you decompose your application into microservices, the complexity of management and networking—both within the cluster and to external services—increases.

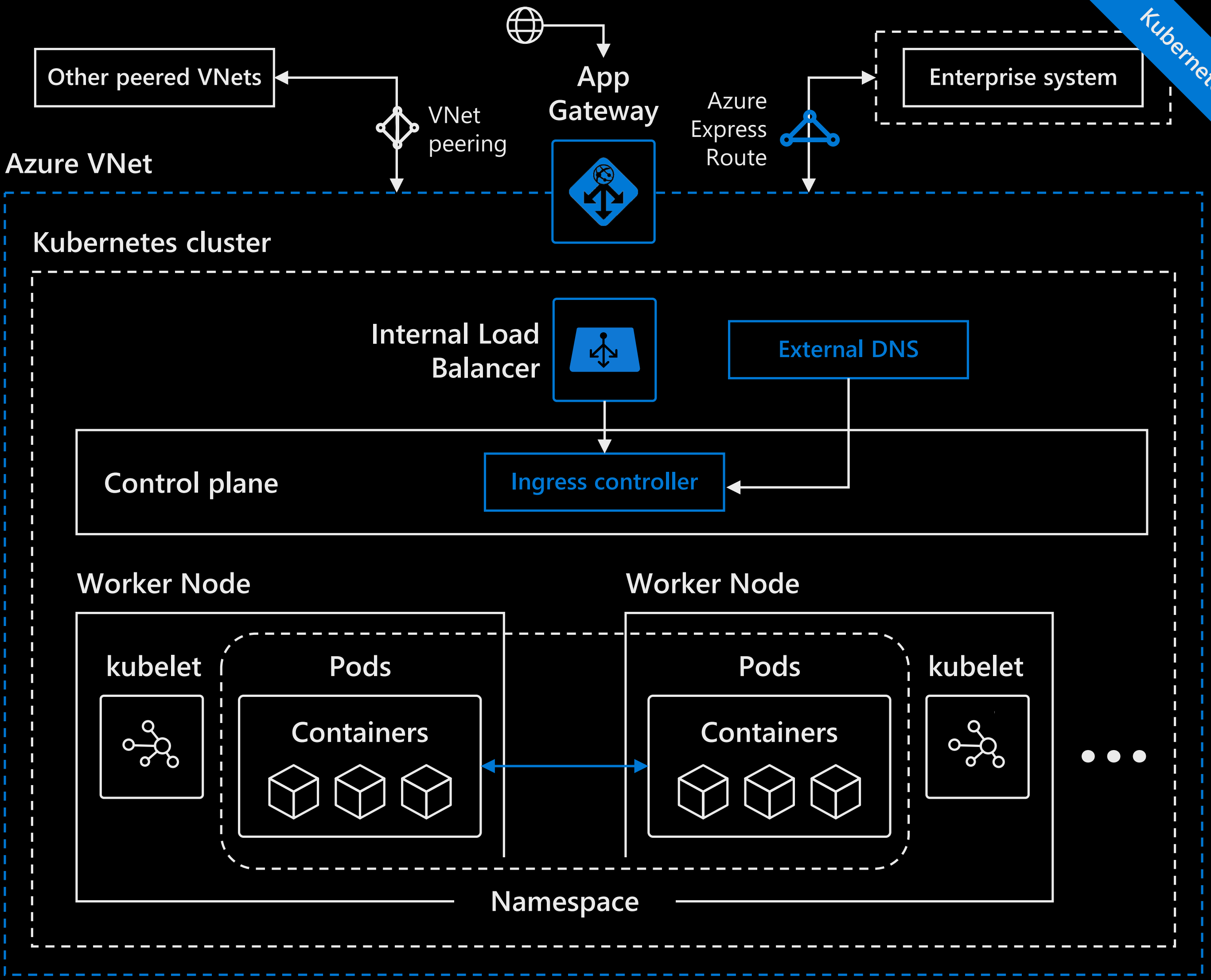
Kubernetes applications are distributed by nature. Native components such as ingress, kube-proxy, and namespaces assist service discovery, load balancing, and segmentation, but are insufficient for the secure communication paths required by your workloads in production.

Capabilities like virtual networks, network policy, and application gateways can help set up a solid foundation for secure networking. An enterprise-grade platform can also offer you hybrid networking capabilities that help utilize your existing technology investment.

Secure communication path

- Create an isolated environment using Azure Virtual Network to allow only authenticated IPs to access your network
- Protect against threats and intrusions using App Gateway with WAF
- Secure communication paths between namespaces (and nodes) using network policy
- Connect to on-premises infrastructure using Azure Express Route
- Secure connection between VNets with VNet peering


Kubernetes cluster
Control plane
Worker Node
Worker Node
kubelet
Pods
Containers
Pods
Containers
kubelet
Namespace



Kubernetes on Azure

“Azure support for Docker, Kubernetes, Puppet, Terraform, Cassandra, and other open source tools has become very important to us and has really accelerated our move into Azure.”

— Robert Rudduck, Director of Architecture and DevOps, Ambit Energy

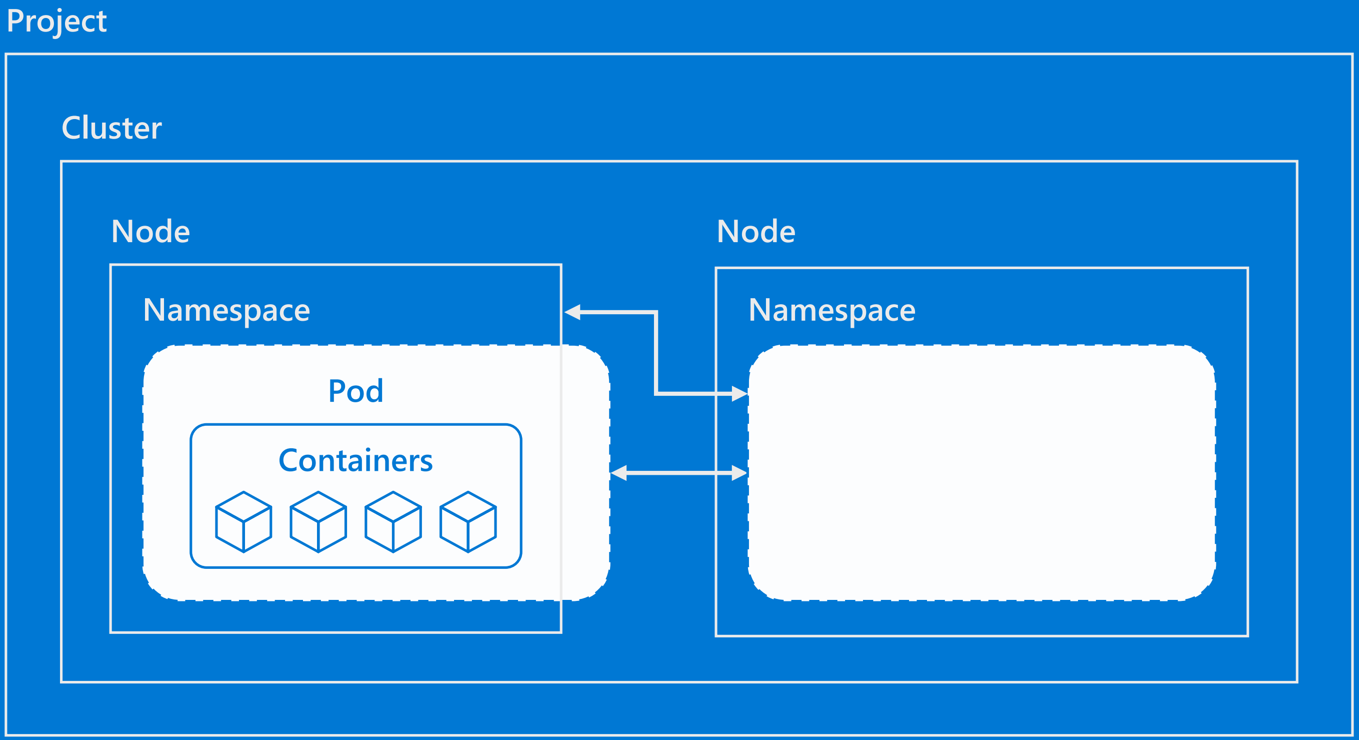
AMBITENERGY®

Capability

Network policy

- Secure communication paths between namespaces and nodes
- Better controls with user-defined network policy
- All powered by Calico, an open source project

Learn more at
aka.ms/aksbook/networkpolicy


Project
Cluster
Node
Node
Namespace
Namespace
Pod
Containers

Top AKS scenarios

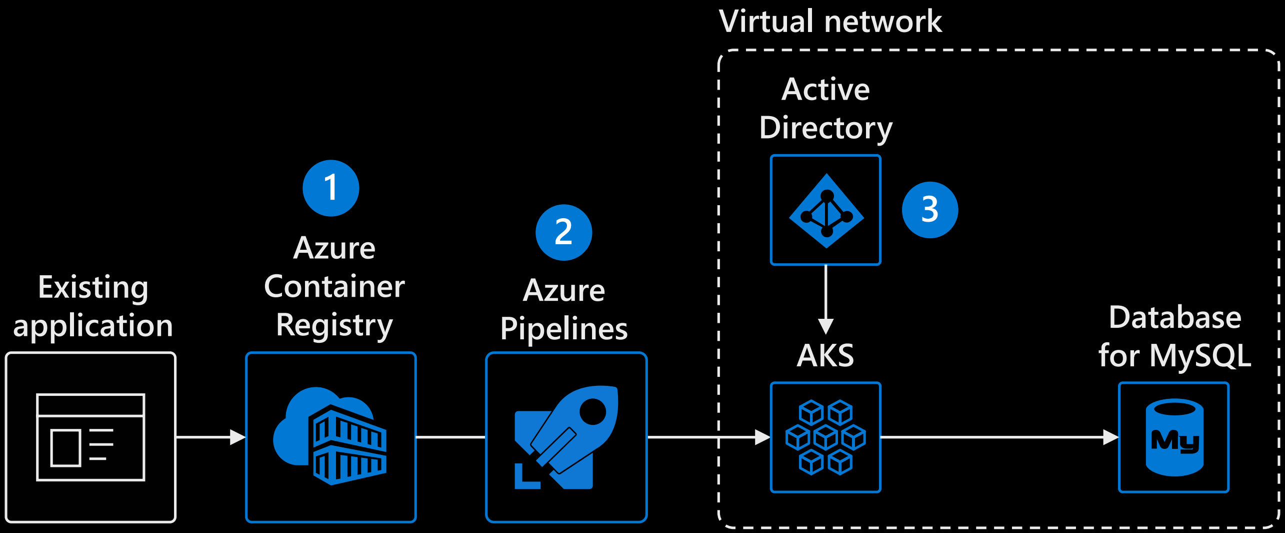
Lift and shift to containers	Microservices	Machine learning	IoT	DevSecOps
Cost saving Without refactoring your app	**Agility** Faster application development	**Performance** Low latency processing	**Portability** Build once, run anywhere	**Security** Deliver code faster and securely at scale

App modernization without code changes

Capabilities

1. Use Azure Container Registry to store container images and Helm charts for your modernized applications, replicated globally for low latency image serving
2. Integrate AKS with Azure Pipelines or other Kubernetes ecosystem tooling to enable continuous integration/continuous delivery (CI/CD)
3. Enhance security with Azure Active Directory and RBAC to control access to AKS resources

Learn more at
aka.ms/aksbook/liftandshift



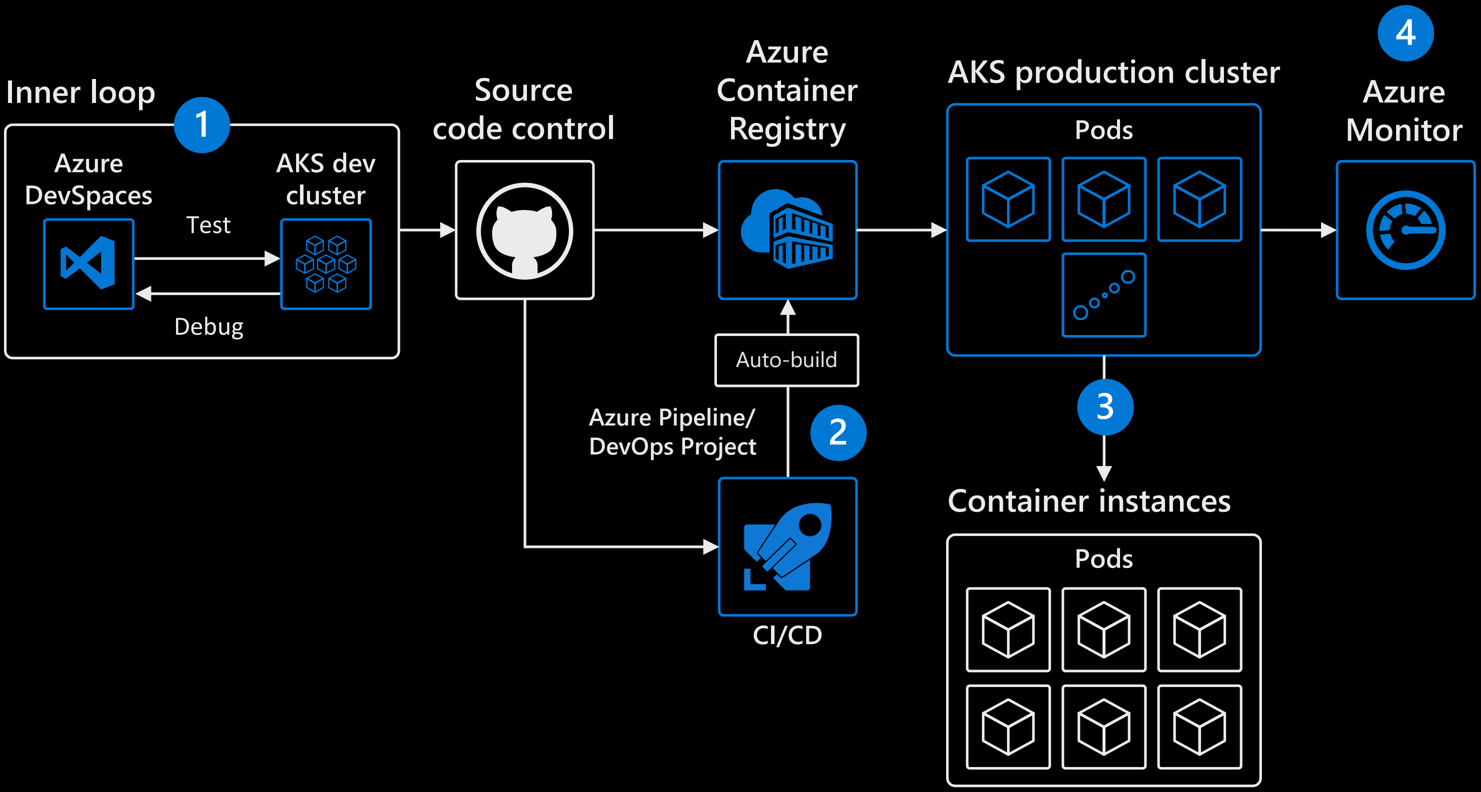
Virtual network
Active Directory
3
1
Azure Container Registry
2
Azure Pipelines
Existing application
AKS
Database for MySQL

Microservices for faster app development

Capabilities

1. Use Azure Dev Spaces to iteratively develop, test, and debug microservices targeted for AKS clusters.
2. Azure DevOps has native integration with Helm and helps simplifying continuous integration/continuous delivery (CI/CD)
3. Virtual node—a Virtual Kubelet implementation—allows fast scaling of services for unpredictable traffic.
4. Azure Monitor provides a single pane of glass for monitoring app telemetry, cluster-to-container level health analytics.

Learn more at
aka.ms/aksbook/microservices

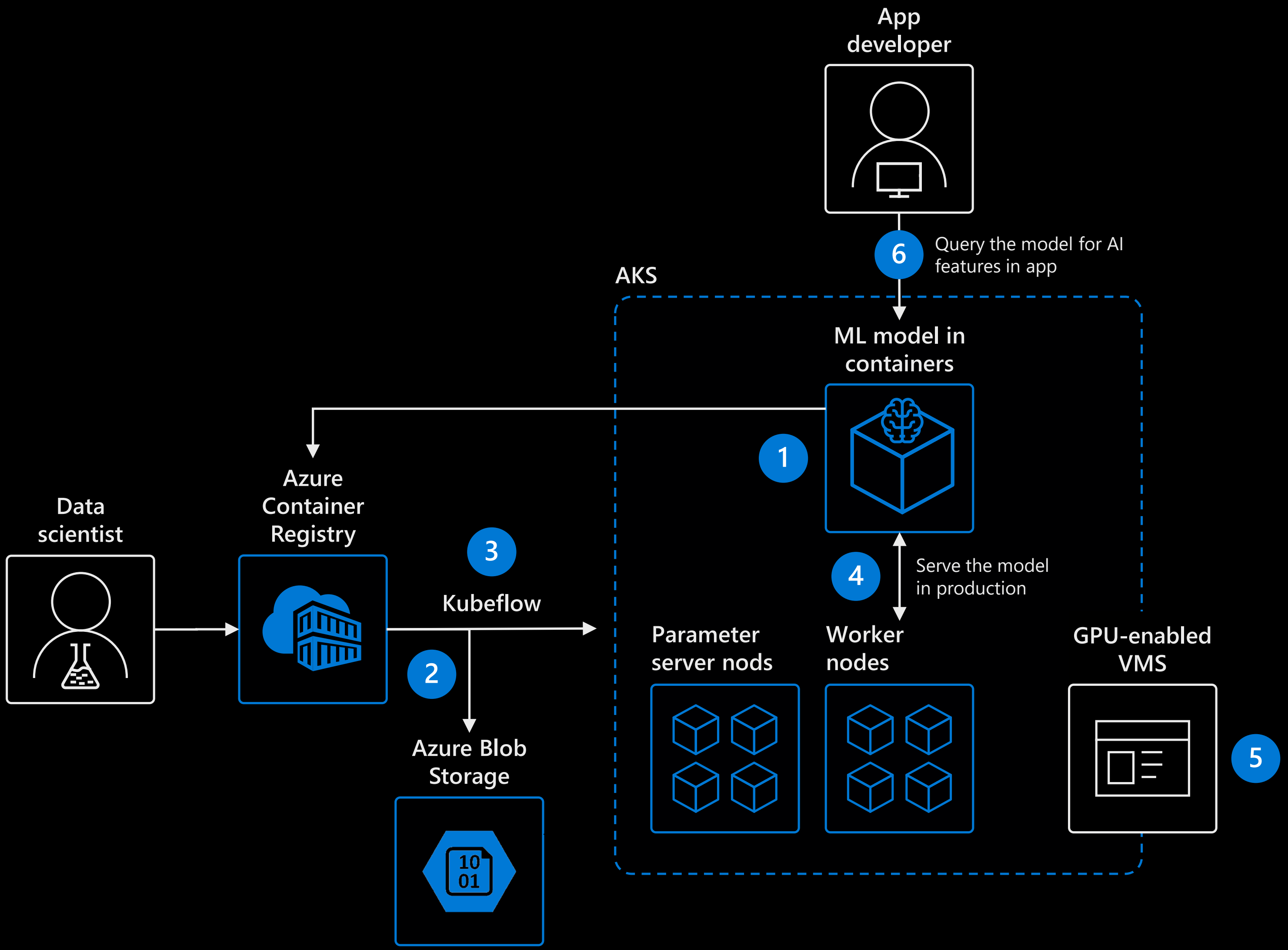

Inner loop
1
Azure DevSpaces
AKS dev cluster
Test
Debug
Source code control
Azure Container Registry
Auto-build
Azure Pipeline/ DevOps Project
2
CI/CD
AKS production cluster
Pods
3
Container instances
Pods
4
Azure Monitor

Data scientist in a box

Capabilities

1. Package ML model into a container and publish to Azure Container Registry
2. Azure Blob Storage hosts training data sets and trained model
3. Use Kubeflow to deploy training job to AKS, distributed training job to AKS includes Parameter servers and Worker nodes
4. Serve production model using Kubeflow, promoting a consistent environment across test, control and production
5. AKS supports GPU-enabled VM
6. Developer can build features querying the model running in AKS cluster

Learn more at
aka.ms/aksbook/ml



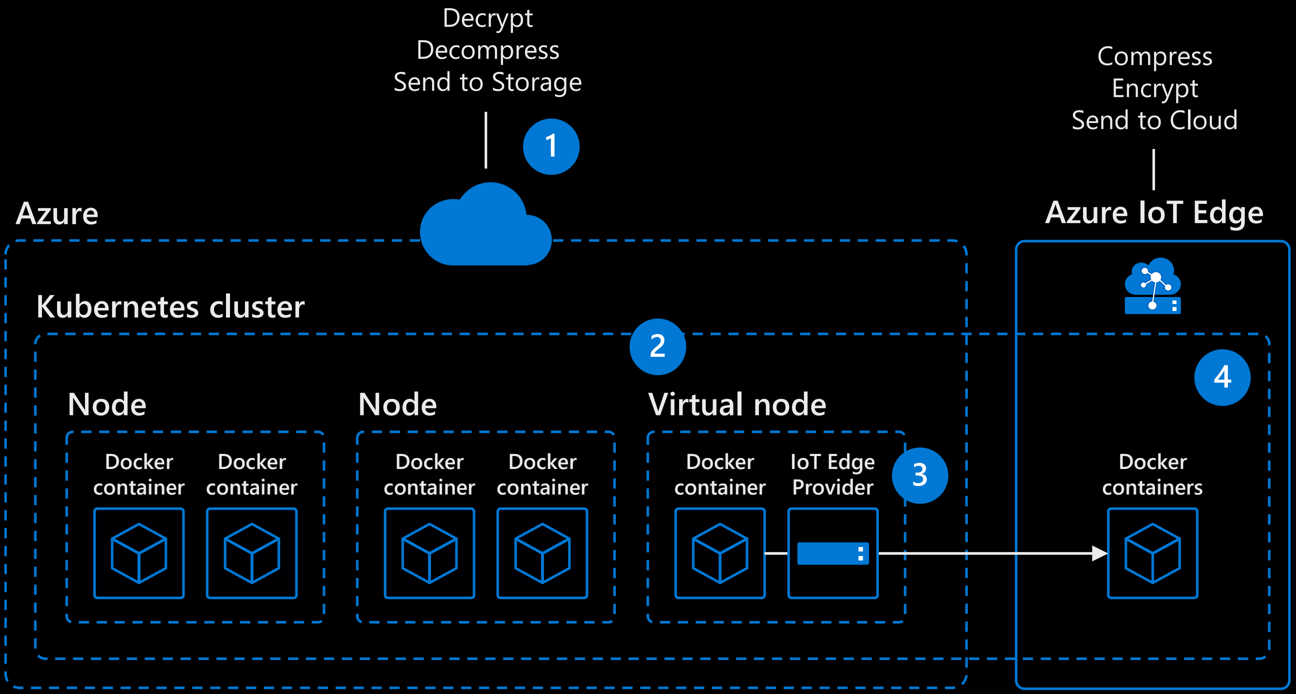
App developer
6
Query the model for AI features in app
AKS
ML model in containers
1
Azure Container Registry
Data scientist
3
Kubeflow
2
Azure Blob Storage
4
Serve the model in production
Parameter server nods
Worker nodes
GPU-enabled VMS
5

Scalable Internet of Things solutions

Capabilities

1. **Azure IoT Edge** encrypts data and send to Azure, which then decrypts the data and sends to storage
2. **Virtual node**, an implementation of Virtual Kubelet, serves as the translator between cloud and Edge
3. **IoT Edge Provider in virtual node** redirects containers to IoT Edge and extend AKS cluster to target millions of edge devices
4. Consistent update, management, and monitoring as one unit in AKS using single pod definition

Learn more at
aka.ms/aksbook/iot

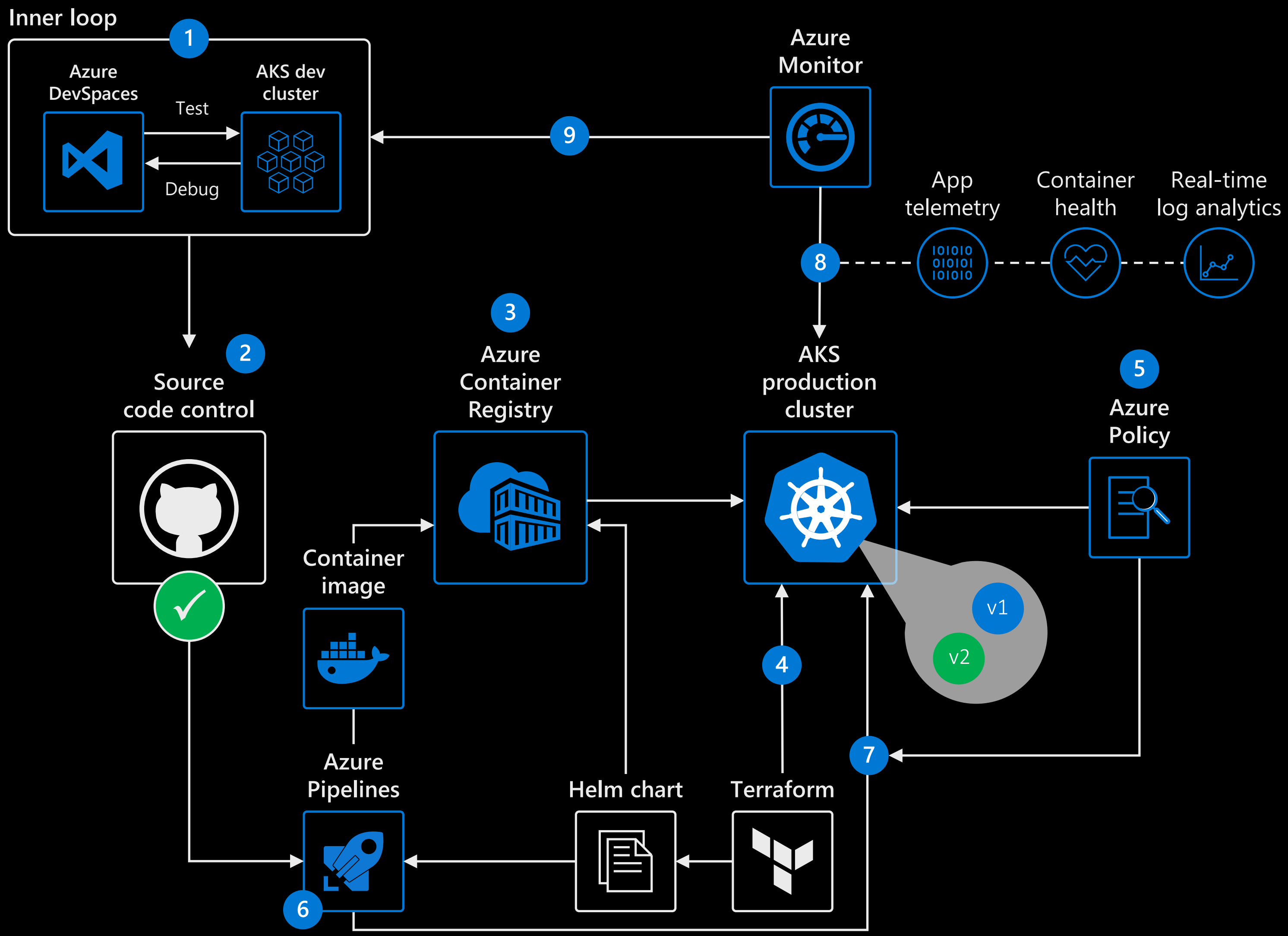

Decrypt
Decompress
Send to Storage
1
Compress
Encrypt
Send to Cloud
Azure
Azure IoT Edge
Kubernetes cluster
2
4
Node
Node
Virtual node
Docker container
Docker container
Docker container
Docker container
Docker container
IoT Edge Provider
3
Docker containers

DevSecOps

Capabilities

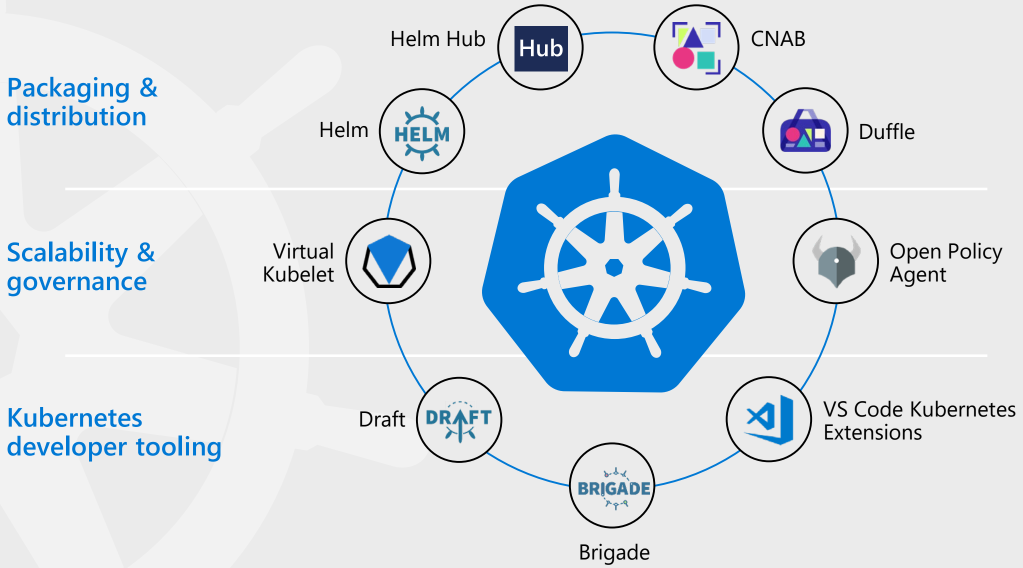
1. Developers rapidly iterate, test, and debug different parts of an application together in the same Kubernetes cluster
2. Code is merged into a GitHub repository, after which automated builds and tests are run by Azure Pipelines
3. Container image is pushed to the Azure Container Registry
4. Kubernetes clusters are provisioned using tools like Terraform; Helm charts, installed by Terraform, define the desired state of app resources and configurations
5. Operators enforce policies to govern deployments to the AKS cluster
6. Release pipeline automatically executes pre-defined deployment strategy with each code change
7. Policy enforcement and auditing is added to CI/CD pipeline using Azure Policy
8. App telemetry, container health monitoring, and real-time log analytics are obtained using Azure Monitor
9. Insights used to address issues and fed into next sprint plans

Learn more at
aka.ms/aksbook/devsecops


Inner loop
1
Azure DevSpaces
Test
Debug
AKS dev cluster
9
Azure Monitor
8
App telemetry
Container health
Real-time log analytics
2
Source code control
3
Azure Container Registry
AKS production cluster
5
Azure Policy
Container image
v1
v2
4
7
Azure Pipelines
Helm chart
Terraform
6

Microsoft's contributions to the community

Microsoft brings knowledge from working with diverse customers to the Kubernetes community, giving developers access to the latest Microsoft learnings and technologies, and making Kubernetes itself enterprise-friendly and easier to use.



Best support for your needs

Learning path
aka.ms/LearnKubernetes

What is Kubernetes
aka.ms/aks/k8sLearning

Hear from experts
aka.ms/aks/videos

Case studies
aka.ms/aks/casestudy

Azure Kubernetes
aka.ms/aks/page

Try for free
aka.ms/aks/trial

© 2019 Microsoft Corporation. All rights reserved. This document is for informational purposes only. Microsoft makes no warranties, express or implied, with respect to the information presented here.

Microsoft Azure