Microsoft

Containers on Azure

Gold Sponsors

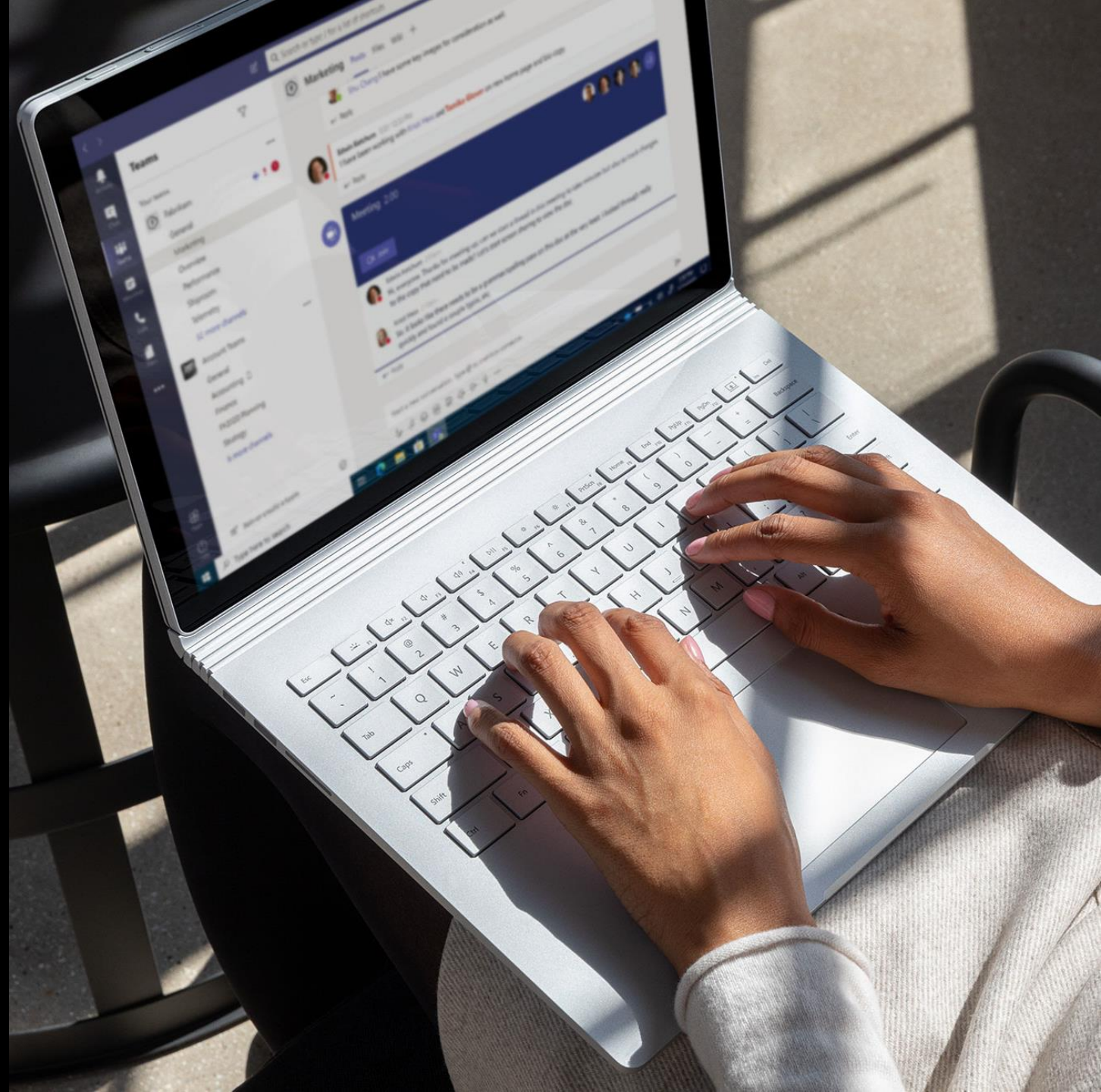
Community Supporter

Tyler Doerksen
Cloud Solution Architect, Winnipeg MB

Microsoft

Agenda

- Azure containers portfolio
- Choosing Azure services
- More Depth:
  - Azure Kubernetes Service
  - Container Apps
  - Azure Spring Cloud
  - App Service
  - Azure Red Hat OpenShift
  - Azure Container Instances



Containers in Azure

Container Apps

Serverless containers for Microservices

Kubernetes Service

Scale and orchestrate Linux containers using Kubernetes

Azure Spring Cloud

Fully Managed Service for Spring Apps

App Service

Deploy web apps or APIs using containers in a PaaS environment

Functions

Run code on-demand without having to explicitly provision or manage infrastructure

Batch

Run large-scale parallel and high-performance computing (HPC) applications efficiently in the cloud

Container Instance

Elastically burst from your Azure Kubernetes Service (AKS) cluster

Ecosystem

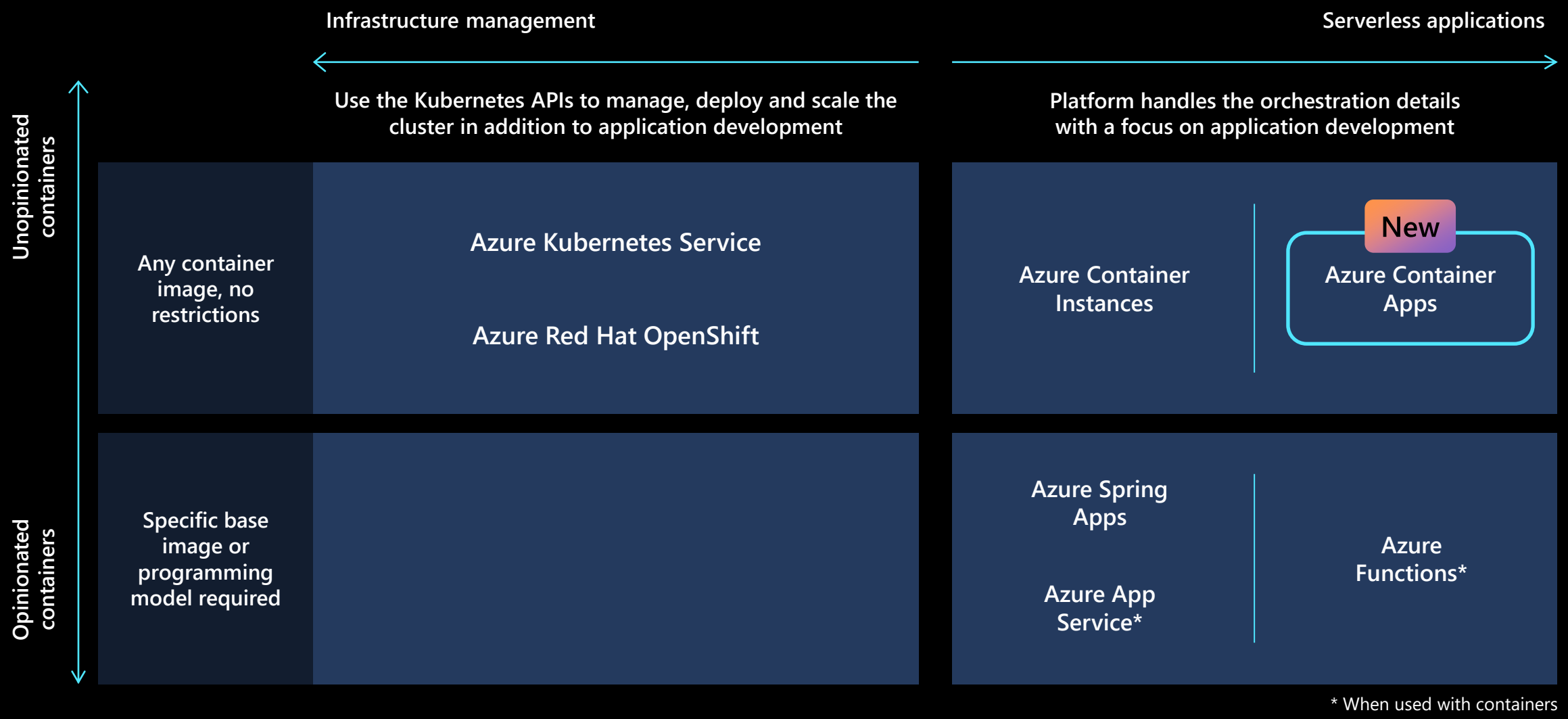
Bring your Partner solutions that run great on Azure

Azure Container Registry

Docker Hub

Choice of developer tools and clients

Azure containers portfolio



* When used with containers

Overview

 **Container Apps**	Managed PaaS container orchestration. Host Kubernetes-style container-based microservices, without complexity of Kubernetes API.	PaaS container orchestrator Stateful and Stateless workloads Autoscale, scale to zero (event-driven)
 **Kubernetes Service (AKS)**	AKS managed K8S service for running containerized applications. Host complex microservices apps, with access to Kubernetes API. Supports orchestration, discovery, security isolation and advanced clustering and workload scheduling capabilities. AKS worker nodes support GPU and could be used for ML.	Container orchestrator Stateful and Stateless workloads Not a PAAS service - many aspects requires mature operational practices
 **Azure Spring Apps**	Managed Spring Apps platform. Host Spring Apps, without a need to learn complexity of Kubernetes.	PaaS applications orchestrator and builder Stateful and Stateless workloads

Overview

App Service

Managed PaaS service. Host Containers and Applications in multiple frameworks and languages. Best fit for web apps including monoliths.

PaaS service with out-of-box support for operational aspects (e.g TLS termination, Custom domains, Health check, Log Aggregation,LB etc)

Azure Container Instance

Great solution for any scenario that can operate in isolated containers, including simple applications, task automation, and build jobs.

PaaS container ideal for batch job or CI/CD agent on demand.

Azure Red Hat OpenShift

Azure Red Hat OpenShift extends Kubernetes. Running containers in production with Kubernetes requires additional tools and resources. Azure Red Hat OpenShift (ARO) combines all this into a single platform.

Container orchestrator
Stateful and Stateless workloads. Ease of operations to IT teams while giving application teams what they need to execute.

	Azure Container Instances	Azure Kubernetes Service	Azure Red Hat OpenShift	Azure Container Apps	Azure Spring Cloud	Azure App Service	Azure Static Web Apps	Azure Functions
Optimized for	Single container	Containerized applications, APIs, and microservices	Containerized applications, APIs, and microservices	Containerized applications, APIs, and microservices	Spring Boot applications	Windows and Linux based 3-tier web apps (code, container)	Web frontend apps	Event-driven code
Language affinity	Any language, any framework	Any language, any framework	Any language, any framework	Any language, any framework	Java	.NET, Java, Python, Node.JS, PHP, Ruby	Angular, React, Svelte, Vue, Blazor	.NET, C#, Node.JS, TypeScript, Java, Python, PowerShell
Ecosystem affinity		CNCF	Red Hat		Spring, VMware	Microsoft		Microsoft
Developer experience	• Minimal developer experience after deploying containers. • No support for service discovery, multiple microservices, etc.	• Deploy containers using Kubernetes YAML files • Use Deployment Center to integrate with GitHub Actions. • Use open-source and CNCF projects like Istio, Knative, and ArgoCD, Tekton. • Use select built-in (managed) projects like KEDA, Dapr, and Azure ML.	• Deploy containerized apps using Red Hat Universal Base Images • Bring application code and build using S2I • Use built-in CI/CD with OpenShift Pipelines • Leverage supported Red Hat components like OpenShift Service Mesh and OpenShift Serverless	• Deploy containerized applications or integrate with GitHub Actions. • Use built-in Dapr for microservices communication, pub/sub, etc. • Rely on built-in application lifecycle management features such as revisions and traffic shifting.	• Build Spring Boot applications using Maven and deploy the JAR artifact. • Integrate with GitHub Actions to automate CI/CD workflow.	• Bring your web app code, integrate with Git, or deploy a container image. • Run Kubernetes anywhere across Azure. • Tight integration of Visual Studio and GitHub.	• Build a website with all popular JavaScript frameworks within clicks • Create backend application with Azure Functions or Blazor • Generate staging versions for quick preview before publishing	• Build event-driven functions using the Azure Functions SDK or the Azure Functions base runtime image. • Trigger functions from different event-sources.
Operator experience	• Use Azure APIs/portal to deploy a single container. • No support for scaling.	• Use Kubernetes native APIs/Azure portal to create and manage workloads. • Integrate with Azure services like Container Registry. • Configure scaling using cluster autoscaler, burst using Virtual Nodes to ACI. • Apply governance policies using Azure Policy.	• Use OpenShift native APIs/OpenShift console to create and manage workloads. • Use the Operator Hub to enable additional functionality such as OpenShift Serverless. • Configure scaling using Machine Set scaler.	• Use Azure APIs/portal to create a container app environment which hosts multiple apps. • Define scaling parameters using KEDA with scale-to-zero support.	• Integrated logging and monitoring with Azure Log Analytics and Application Insights (or 3rd party APMs). • Autoscaling based on schedule and application metrics. • Support for green/blue deployment strategy.	• Azure Web Apps can be deployed on Kubernetes if required using Azure Arc. • Simple operation model and fully managed experience. • Reduce downtime and minimize risk by using deployment slots	• Custom Domain management and free SSL Certificates autorenewal for best security coverage. • Authentication integration with GitHub, AAD and Twitter accounts.	• [..]. Azure Functions can be deployed on Kubernetes if required using Azure Arc.

ACA vs AKS Breakdown

	Advantages	Downsides	Use case
Azure Container Apps	• No cluster management required • Managed KEDA, dapr, Envoy • Lower developer cognitive load • "One throat to choke" • Portal integration for many more things • Cheaper by default with auto scale-to-zero • More secure by default	• All configuration options might not be there • Not yet available in all regions • Many 3rd party solutions need to be ported over (helm charts) • Not all k8s APIs available • Technical limitations (today): o Max 30 replicas per app (pod) o Key Vault secret integration o Only local storage or Azure File mounted volumes o No egress through firewall o Only Azure Monitor support	• Get serverless apps up and running quickly • Desire to use managed features (EasyAuth, etc) • Teams who are not very skilled in K8s
Azure Kubernetes Service	• Endless configuration options! • Multi-tenancy can be better achieved • Can use less IP address space • MTLS is an option • Cost segmentation with Kubecost • Windows container support • GitOps support	• Higher bar of responsibility for everything deployed in cluster • Many more toolsets to learn • Decision making process of what to use for each feature is difficult! • Too many options…	• Need configuration options that are not available in ACA • Desire access to the K8s APIs • Large, multi-team clusters • Teams who are already skilled in K8s

In Depth

App Service, Container Instance, ARO, Container Apps, AKS, Spring Apps

Microsoft Azure

How to Choose the Right Azure Services for Your Applications — It's Not A or B

Get the free e-book

aka.ms/Not-A-or-B

Why IT orgs think A or B

- Resource optimization
- External influence
- Compliance
- Culture
- Knowledge gap
- Generic is better than opinionated
- Politics
- Job preservation, complacency
- We have always done it that way – tyranny of the familiar

Why it is not necessary to think A or B

- Cloud is on-demand and elastic, with no upfront capital expenditures – pay for what you use
- Significant amount of infrastructure and platform responsibility is transferred to cloud provider
- Luxury to use the right tool for the right job

Microsoft Azure

Choose right Azure services for your applications

Lower agility

Longer lead times

Bottle necks

Force fit use case to solution

Fixed mindset

A or B

Speed to market

Innovate faster

Process efficiency

Right tool for the right job

Growth mindset

A + B

Infrastructure

Azure Kubernetes Service

Create, manage and scale infrastructure

- ACR
- Azure Kubernetes Service
- Kubernetes Secrets
- Monitoring

Deploy, manage and scale Spring Cloud middleware

- Spring Cloud Config Server
- Mirror service for Spring Cloud Config Server
- Spring Cloud Service Registry
- Spring Cloud Gateway

Azure Spring Cloud

Create and manage Azure Spring Cloud

Application Lifecycle Management

Azure Kubernetes Service

Do it Yourself

- Build and push container images
- Manage security fixes and updates to container images
- Manage base OS, Java runtime and app code
- Kubernetes equivalent for containers to app lifecycle management
- CI/CD -- Pipelines Tasks, Jenkins Plugins and GitHub Actions, blue-green deployments

Azure Spring Cloud

Built-in

- Simple app lifecycle management
- Automatically wire your app with Spring Cloud infrastructure
- Integrated CI/CD pipeline for deployment
- Easily deploy source code or build artifacts

Monitoring

Azure Kubernetes Service

Do it Yourself

- Instrument containers
- Dashboarding
- APM Integration

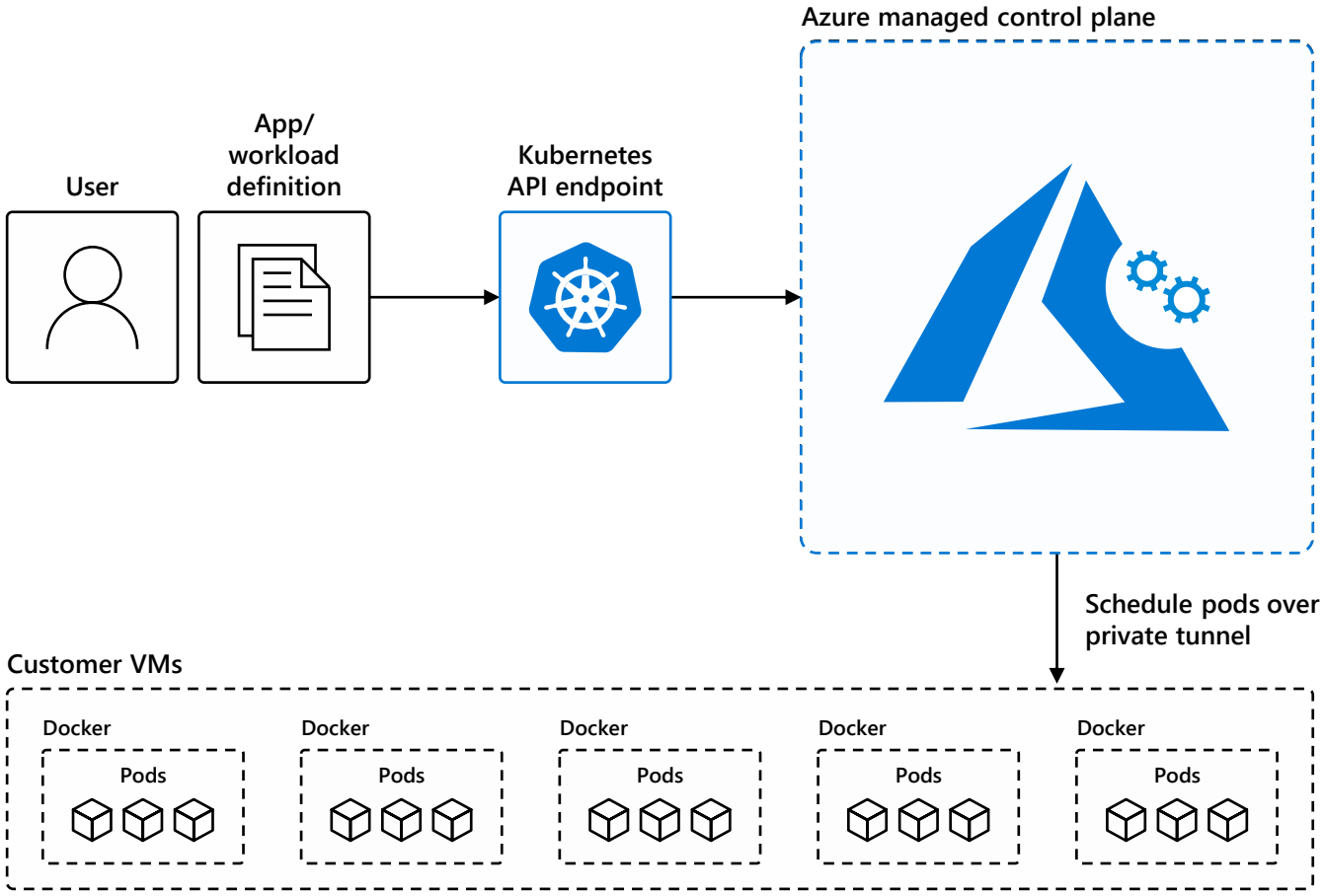
Azure Spring Cloud

Built-in

- Troubleshooting
- Tracing end-user actions
- Planning capacities
- Keeping an eye on production
- APM Integration

Managed Kubernetes on Azure

- Automated upgrades, patches
- High reliability, availability
- Easy, secure cluster scaling
- Self-healing
- API server monitoring
- At no charge



Kubernetes done right is hard

Templating
Validation
OS Setup

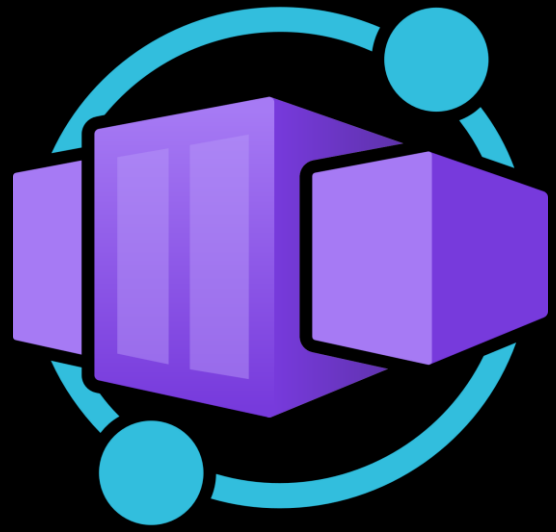
Identity & Security Access
App Monitoring & Alerts
Storage & Persistence
Egress, Ingress & Integration
Host Container Images
Build/Deploy Methodology

Platform Monitoring & Alerts
Metering & Chargeback
Platform Security Hardening
Image Hardening
Security Certifications
Network Policy
Disaster Recovery
Resource Segmentation

OS Upgrade & Patch
Platform Upgrade & Patch
Image Upgrade & Patch
App Upgrade & Patch
Security Patches
Continuous Security Scanning
Multi-environment Rollout
Enterprise Container Registry
Cluster & App Elasticity
Monitor, Alert, Remediate
Log Aggregation

Azure Container Apps

Public preview



Serverless containers for microservices

Build modern apps on open source

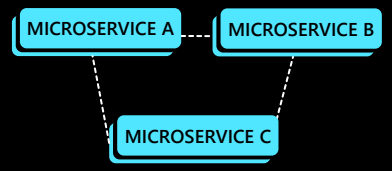
Focus on apps, not infrastructure

Seamlessly port to Kubernetes

Kubernetes KEDA dapr DAPR envoy Envoy

What can you build with Azure Container Apps?

Microservices

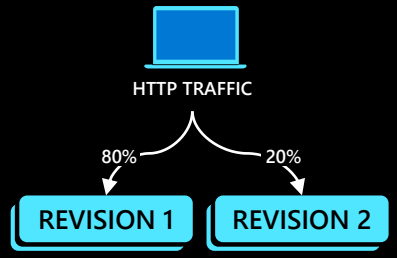


Microservices architecture with the option to integrate with Dapr

AUTO-SCALE CRITERIA

Individual microservices can scale independently using any KEDA scale triggers

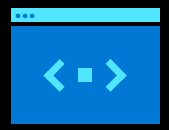
Public API endpoints



E.g., API app with HTTP requests split between two revisions of the app

Scaling is determined by the number of concurrent HTTP requests

Web Apps



E.g., Web app with custom domain, TLS certificates, and integrated authentication

Scaling is determined by the number of concurrent HTTP requests

Event-driven processing

E.g., Queue reader app that processes messages as they arrive in a queue

Scaling is determined by the number of messages in the queue

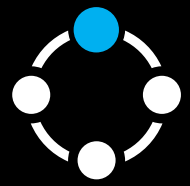
Background processing

E.g., Continuously running background process transforms data in a database

Scaling is determined by the level of CPU or memory load

Common microservices requirements

1. Service to service communication **(Enabled with Dapr)**
    - 1.1 TLS encryption and mutual TLS authentication ✓
    - 1.2 Reliability and retries ✓
    - 1.3 Observability and distributed tracing ✓
2. Independent component lifecycle: versioning and scaling ✓ **(Enabled with revisions and KEDA)**
3. Data encapsulation and governance ✓ **(Enabled with Dapr)**
4. Independent blue-green and canary deployments ✓ **(Enabled with revisions and Envoy)**



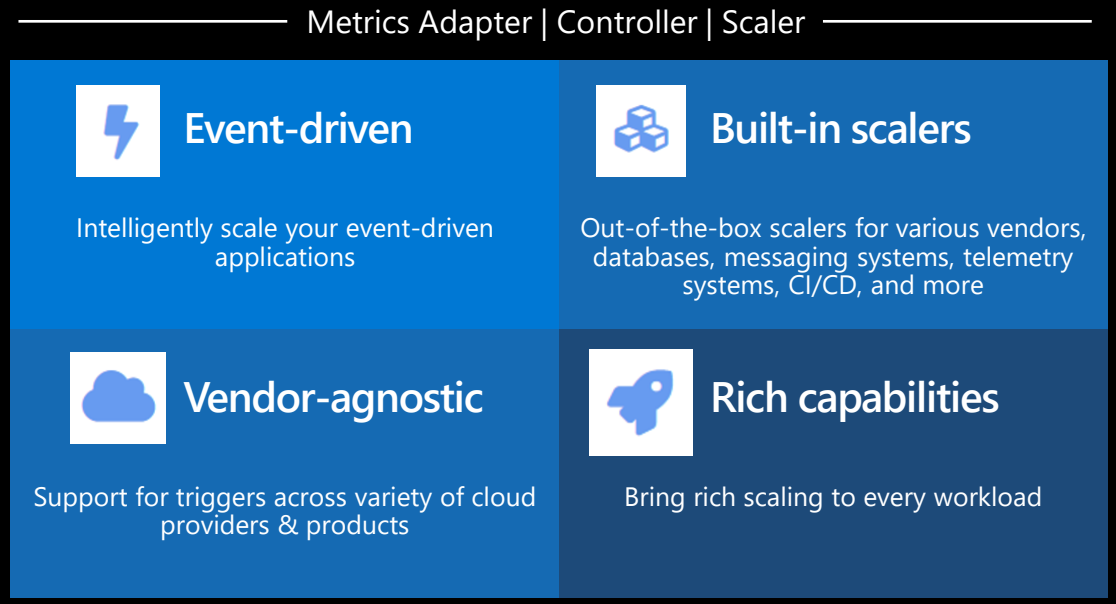
Application autoscaling made simple

Open-source, extensible, and vendor agnostic

KEDA

Kubernetes-based Event Driven Autoscaler

Drive the scaling of any container based on a growing list of event sources, known as: scalers



keda.sh

CLOUD NATIVE COMPUTING FOUNDATION

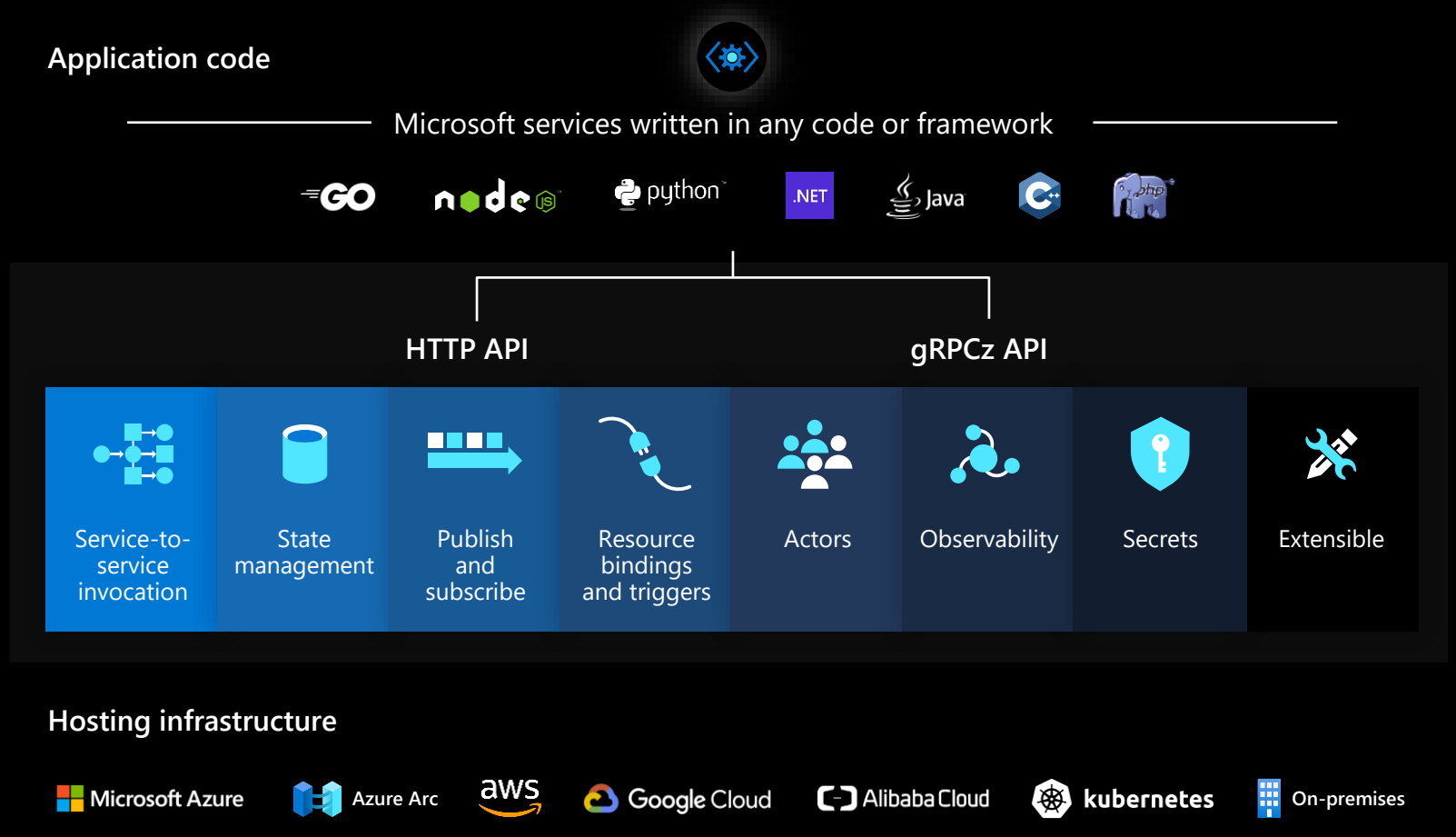
Microservices using any language or framework

Any cloud or edge infrastructure

dapr

Distributed Application Runtime

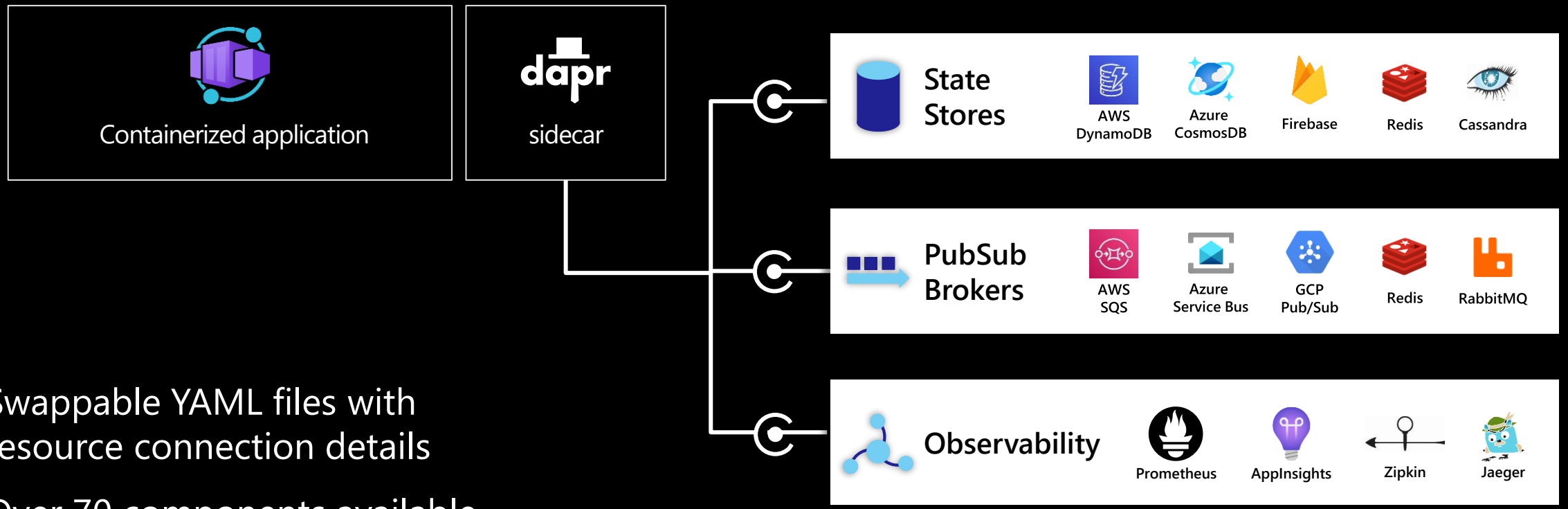
Portable, event-driven, runtime for building distributed applications across cloud and edge



dapr.io

CLOUD NATIVE COMPUTING FOUNDATION

Dapr components



Swappable YAML files with resource connection details

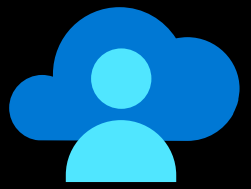
Over 70 components available

Create components for your resource at: github.com/dapr/components-contrib

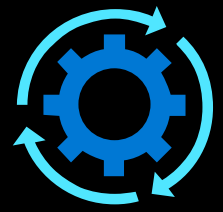
Azure Spring Apps

Fully managed service for Spring Boot apps

Full integration into Azure's ecosystem and services



Fully managed infrastructure



Built-in app lifecycle management

Ease of monitoring

Enterprise ready

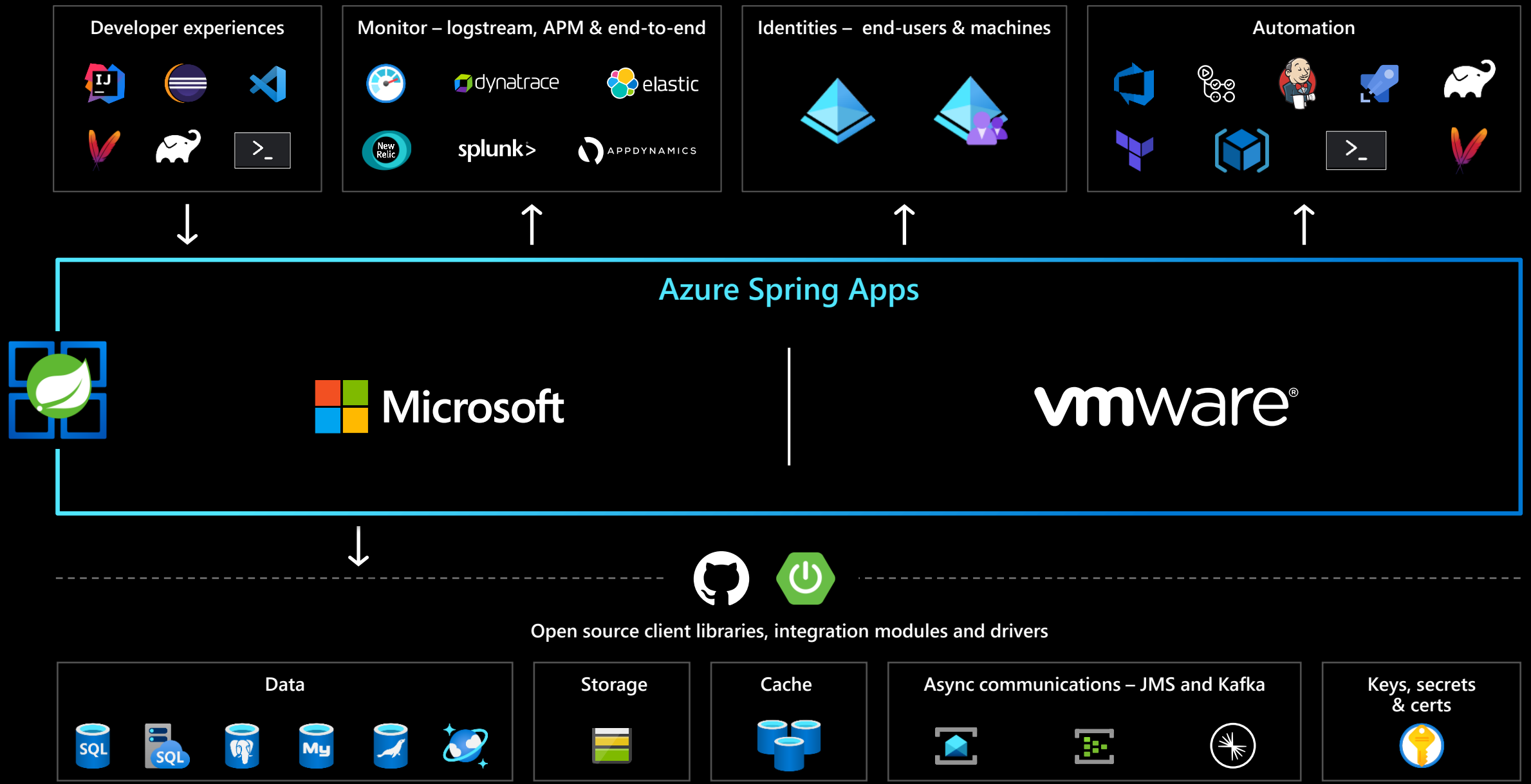
Jointly built, operated and supported by Microsoft and VMware

Microsoft

vmware®

On Kubernetes

You do not have to learn or manage Kubernetes


Developer experiences
Monitor – logstream, APM & end-to-end
dynatrace
elastic
splunk>
APPDYNAMICS
Identities – end-users & machines
Automation
Azure Spring Apps
Microsoft
vmware
Open source client libraries, integration modules and drivers
Data
SQL
SQL
My
Storage
Cache
Async communications – JMS and Kafka
Keys, secrets & certs

Three simple operations

1

Create service

```
az spring create --name ${SPRING_CLOUD_SERVICE} \
--sku standard \
--resource-group ${RESOURCE_GROUP} \
--location ${REGION}
```

2

Create app

```
az spring app create --name ${CUSTOMERS_SERVICE}
```

3

Deploy app

```
az spring app deploy --name ${CUSTOMERS_SERVICE} |
--jar-path ${CUSTOMERS_SERVICE_JAR}
```

Azure Spring Apps in a nutshell

Developers
Easy to build and deploy

- Build and scale distributed workloads at cloud scale
- Externalize config, enable service registry, secure, automate end-to-end and monitoring
- Harness the power of K8S without learning or operating it
- Easy to spawn environments
- Automate testing
- Advance to production across the globe

IT Operators
Easy to operate at scale

- Home for distributed workloads that leverages services on Azure, on-premises and externals
- Eliminate middleware management efforts – say patching, running middleware, etc.
- Unlimited scale without additional hardware procurement or datacenters
- Define roles and responsibilities to match your team structure – Azure RBAC (developer, DevOps, SRE, tester, security)
- Govern using Azure Policy-based management and enforcement
- Monitor end-to-end – detect and react faster
- Automate end-to-end
- Implement charge backs in line with your funding model – through Azure cost management system

Executives
Peace be with you!

- Minimize costs
- High availability through unpredictable volumes and recover faster from failures
- Expand across the globe – 60+ Azure regions with speed and scale to meet your needs
- Strengthen your security posture with Azure
- Supported by Microsoft and VMware

Azure App Service benefits

High-productivity for devs & ops

- .NET, Node, Java, Docker, PHP, Ruby, Python
- Deploy containers on Windows & Linux
- Staging & deployment
- Testing in production
- App gallery marketplace

Fully-managed

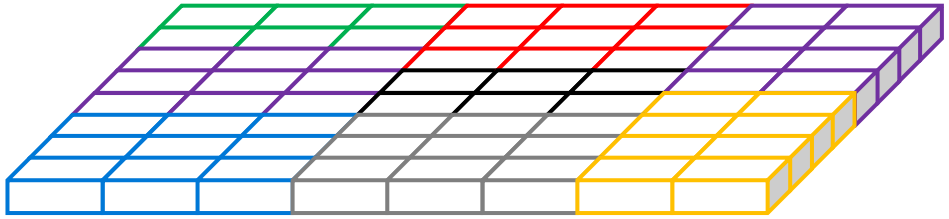
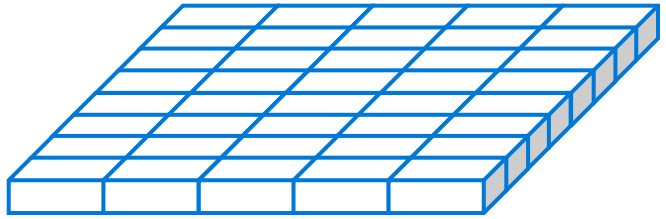
- Auto scale & load balancing
- High availability w/auto patching
- Reduced operations costs
- Backup & recovery

Enterprise-grade

- Global data center footprint
- Hybrid support
- Azure Active Directory integration
- Secure & compliance

Azure App Service in

Managed PaaS service to host code or containers, best fit for web apps and monoliths

	App Service	App Service Environment (ASEv3)
	App Service is a multi-tenant service. Thousands of customers that share the same system safely and securely.	An App Service Environment (ASE) is a single tenant instance of the App Service that deploys into a Virtual Network (VNet)
		
Security	Configure audit policy for private apps and use VNet integration, add private endpoints	Best security and isolation since it deploys to a VNet
Bottom line	Fast scaling	Better security and isolation

Web apps for containers

Easily deploy and run container-based web apps at scale

Accelerated outer loop

Tight integration w/
Docker Hub, Azure Container Registry

Built-in CI/CD w/
Deployment Slots

Intelligent diagnostics &
troubleshooting, remote debugging

Fully managed platform

Automatic scaling
and load balancing

High availability
w/ auto-patching

Backup & recovery

Flexibility & choices

From CLI, portal, or
ARM template

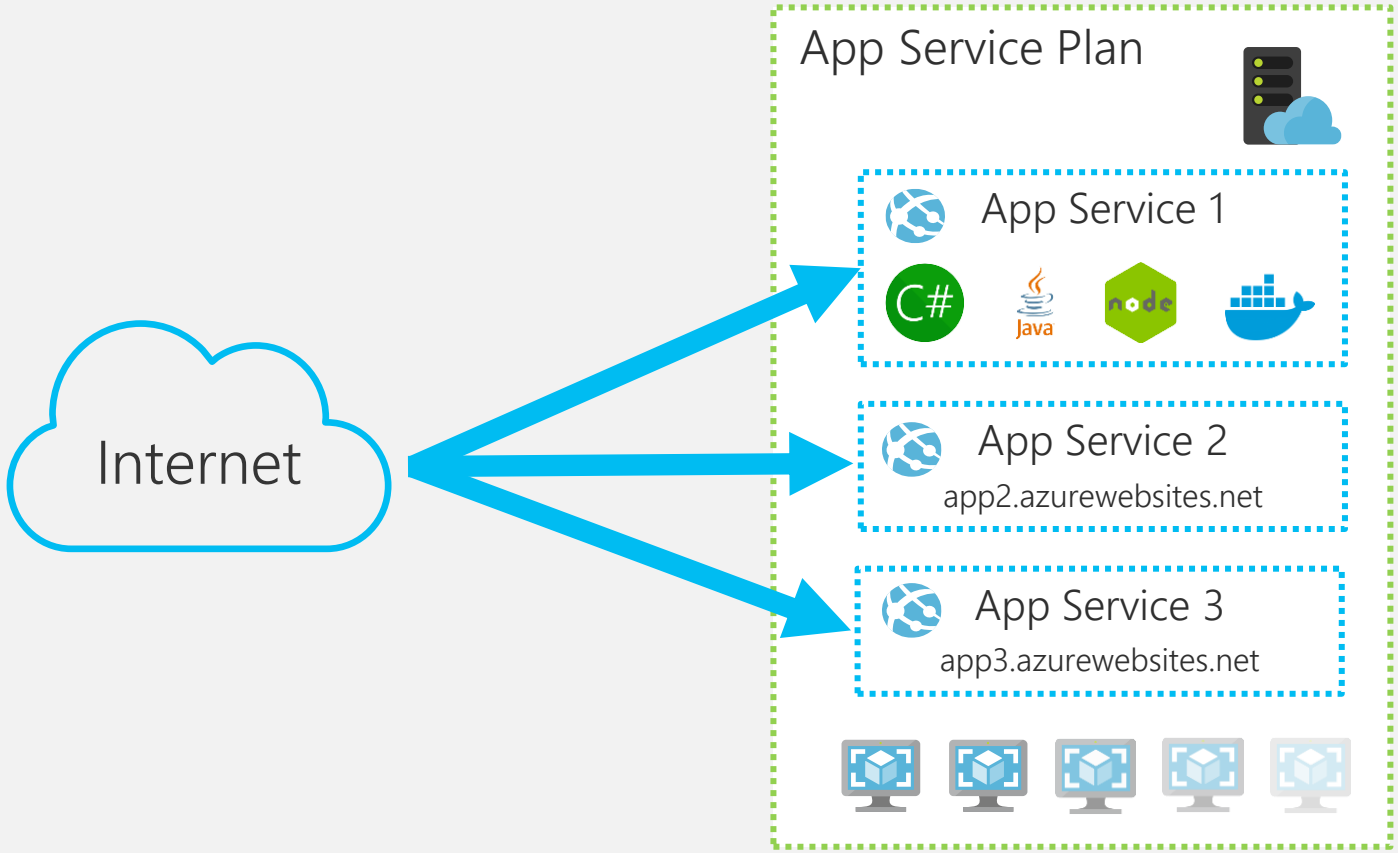
Single Docker image,
multi container w/ Docker compose,
or Kubernetes Pod Definition

IntelliJ, Jenkin, Maven Visual Studio family

App Service

- ✓ Supports Windows and Linux platforms
- ✓ Global scale on demand
- ✓ Built-in autoscale and load balancing
- ✓ High availability with auto-patching
- ✓ Public endpoint
- ✓ Deployment slots
- ✓ Native language support for .NET, PHP, Ruby, Node.js, Java, Python + any containerized web app

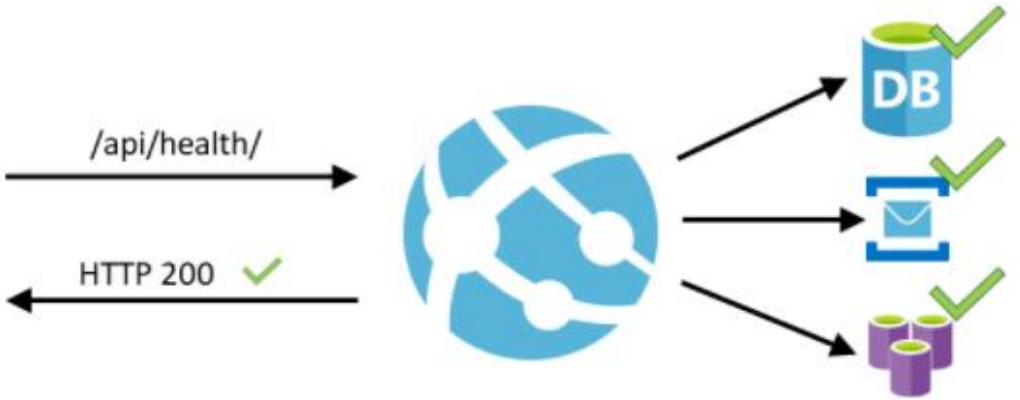
App Service Basics



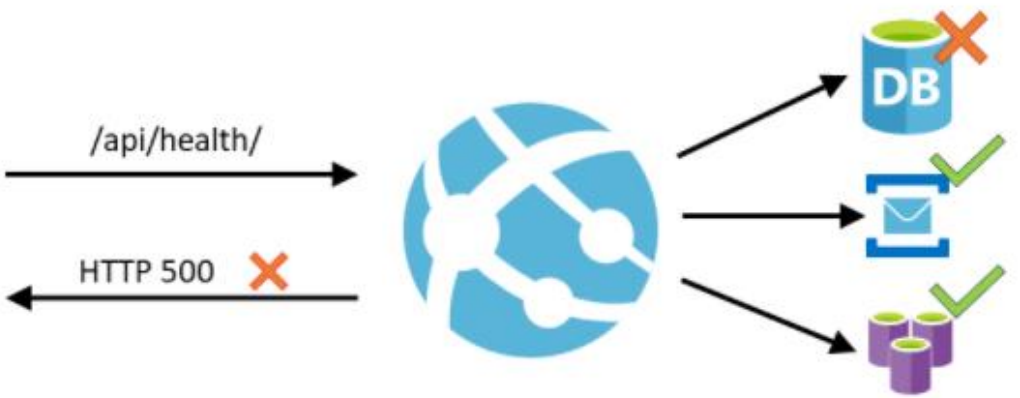
App Service

Health Check

The app instance can connect to necessary services and the app is healthy, so it returns a successful HTTP response code (200 – 299 inclusive).

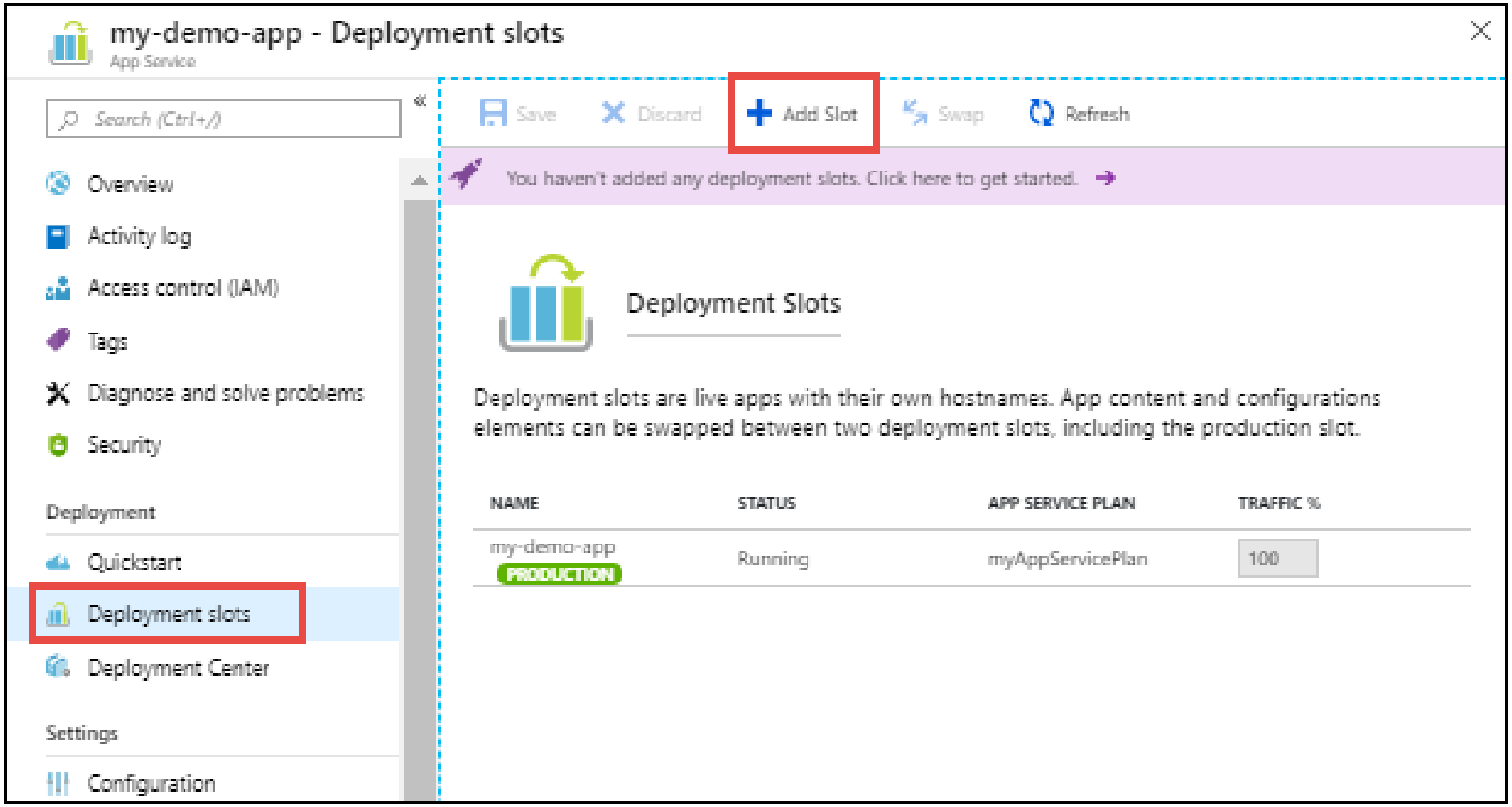


The app instance **cannot** connect to the database, so the app is unhealthy. The app returns an HTTP error code, App Service will re-route requests to the remaining healthy instances.



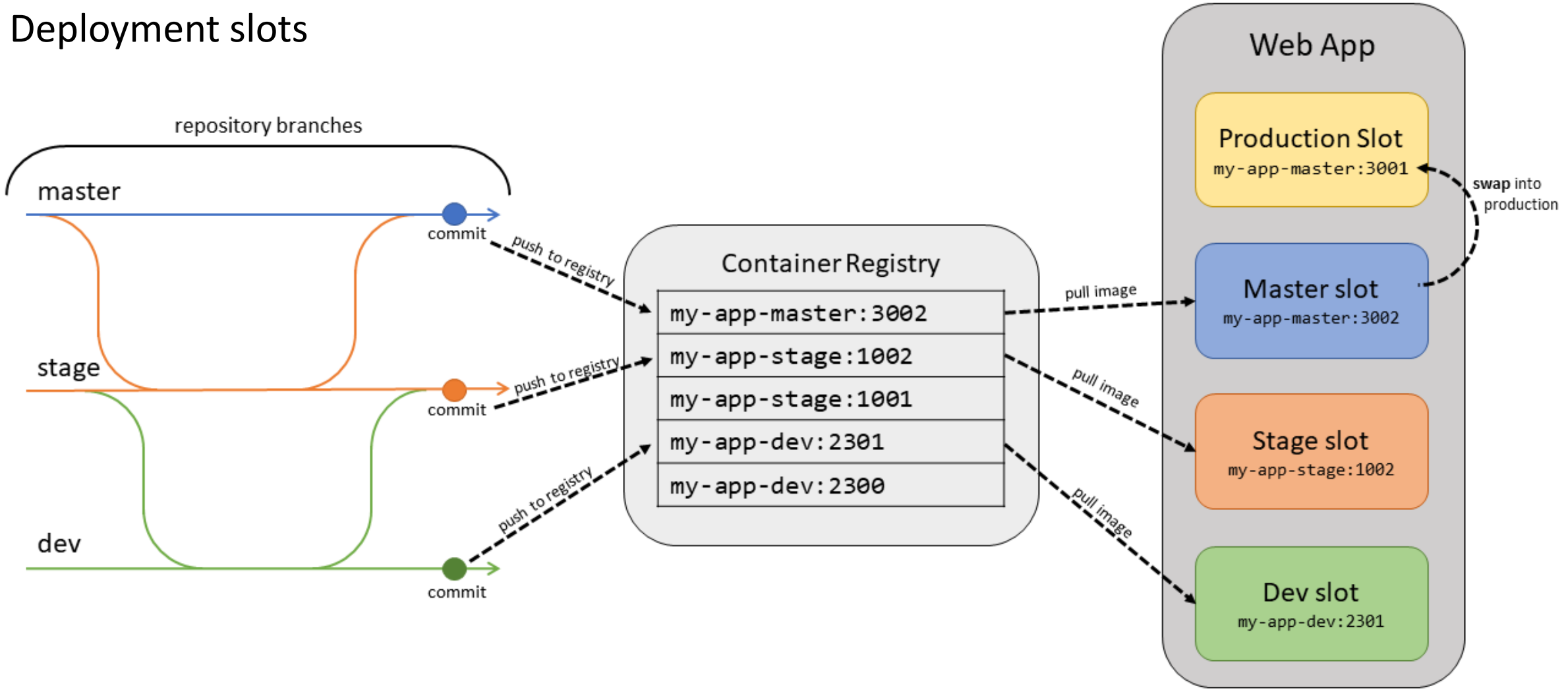
App Service

Deployment slots



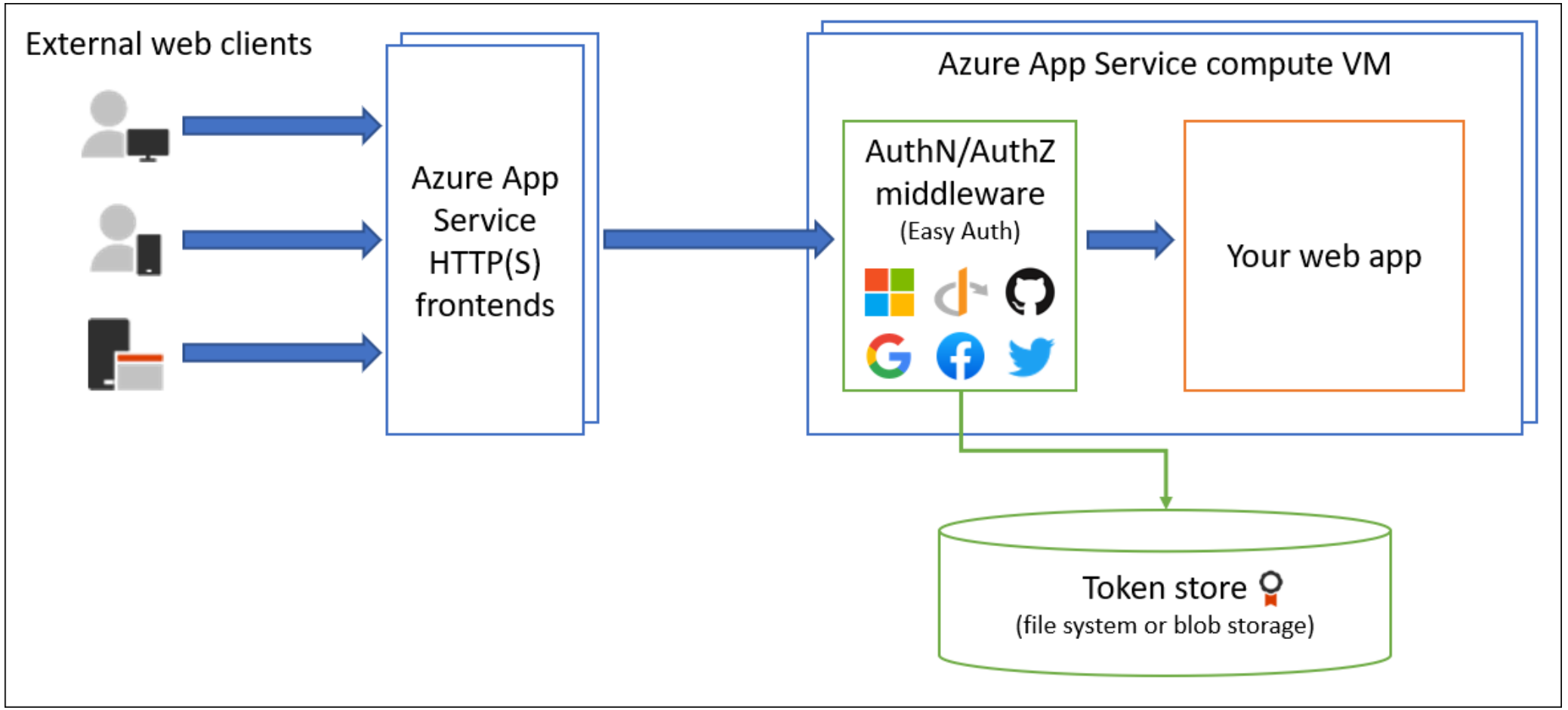
App Service

Deployment slots



App Service

Easy Auth



Containers come with their own challenges

Red Hat OpenShift

Container security
Image scanning, patching, and compliance

Day 2 management
Install, upgrade, and maintain
Integrate existing enterprise technology

Application delivery
Monitoring, metering, and management
Integrate existing developer tools

Trusted enterprise Kubernetes
Continuous security, world-class support and services and deep expertise to confidently run any application

A cloud-like experience, everywhere
Full-stack and automated operations on a consistent foundation for on-premises or hybrid cloud infrastructures

Empowering developers to innovate
Get applications to production sooner with a wide range of technologies and streamlined workflows

Azure Red Hat OpenShift

Jointly engineered, operated, and supported by both Microsoft **and** Red Hat **with an integrated support experience**

Build, deploy and scale apps with confidence

In just minutes, **deploy enterprise-grade Red Hat OpenShift clusters on Azure**

Enterprise-grade operations, security and compliance

Deploy your business-critical apps with confidence with an industry-leading SLA of 99.95% availability, with PCI DSS, ISO 27001, HITRUST, SOC 2 Type II, and FedRAMP certifications.

Empowering developers to innovate

Promote developer productivity with built-in CI/CD pipelines, then easily connect your applications to hundreds of Azure services such as MySQL, PostgreSQL, Redis, Cosmos DB, and more.

Scale on your terms

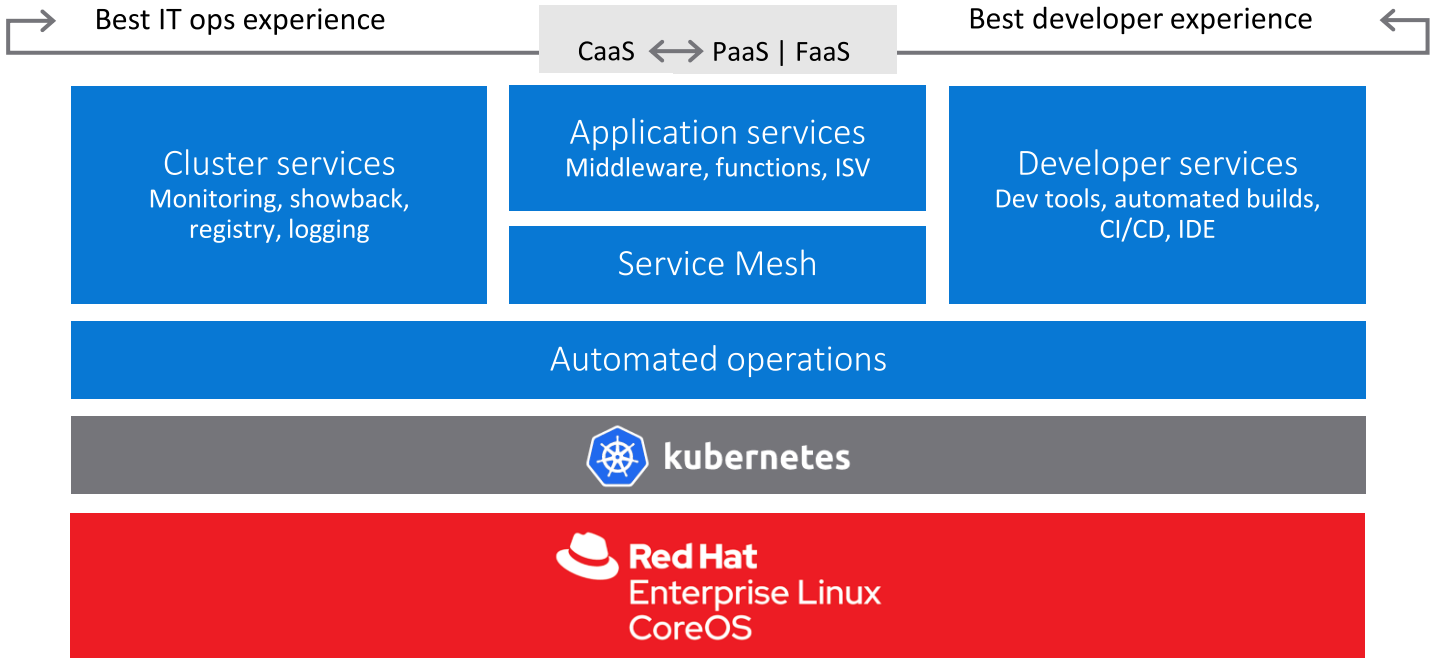
Start a highly available cluster in a few minutes, then scale as your application demand changes; plus, get your choice of standard, high-memory, or high-CPU application nodes. Pay through your Azure subscription.

OpenShift 4 – a smarter Kubernetes platform

Up and running in minutes from the container host to application services.

Scales with your needs from 10 containers to thousands.

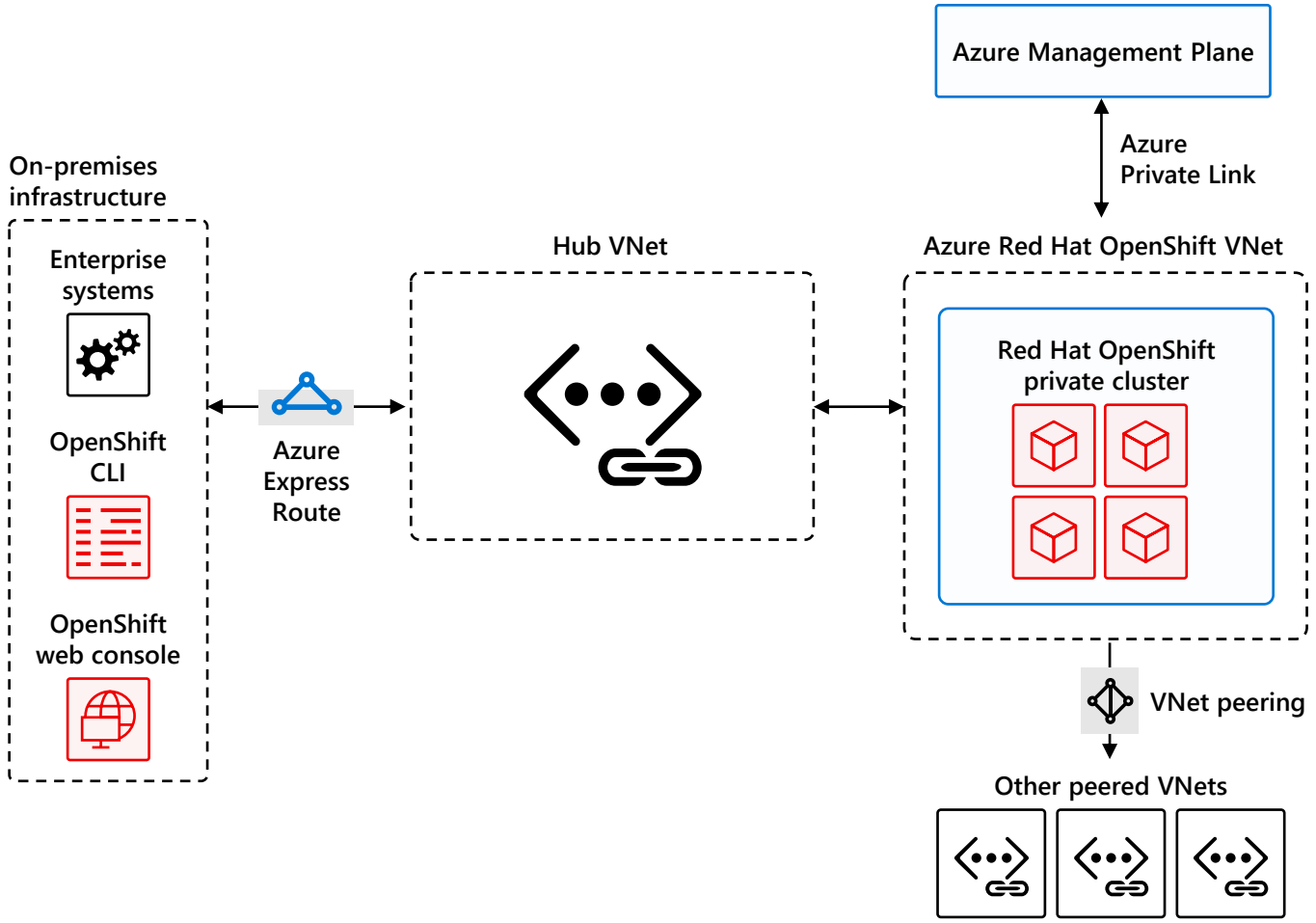
One-click updates for platform, services, and applications



Private cluster management and ingress endpoints

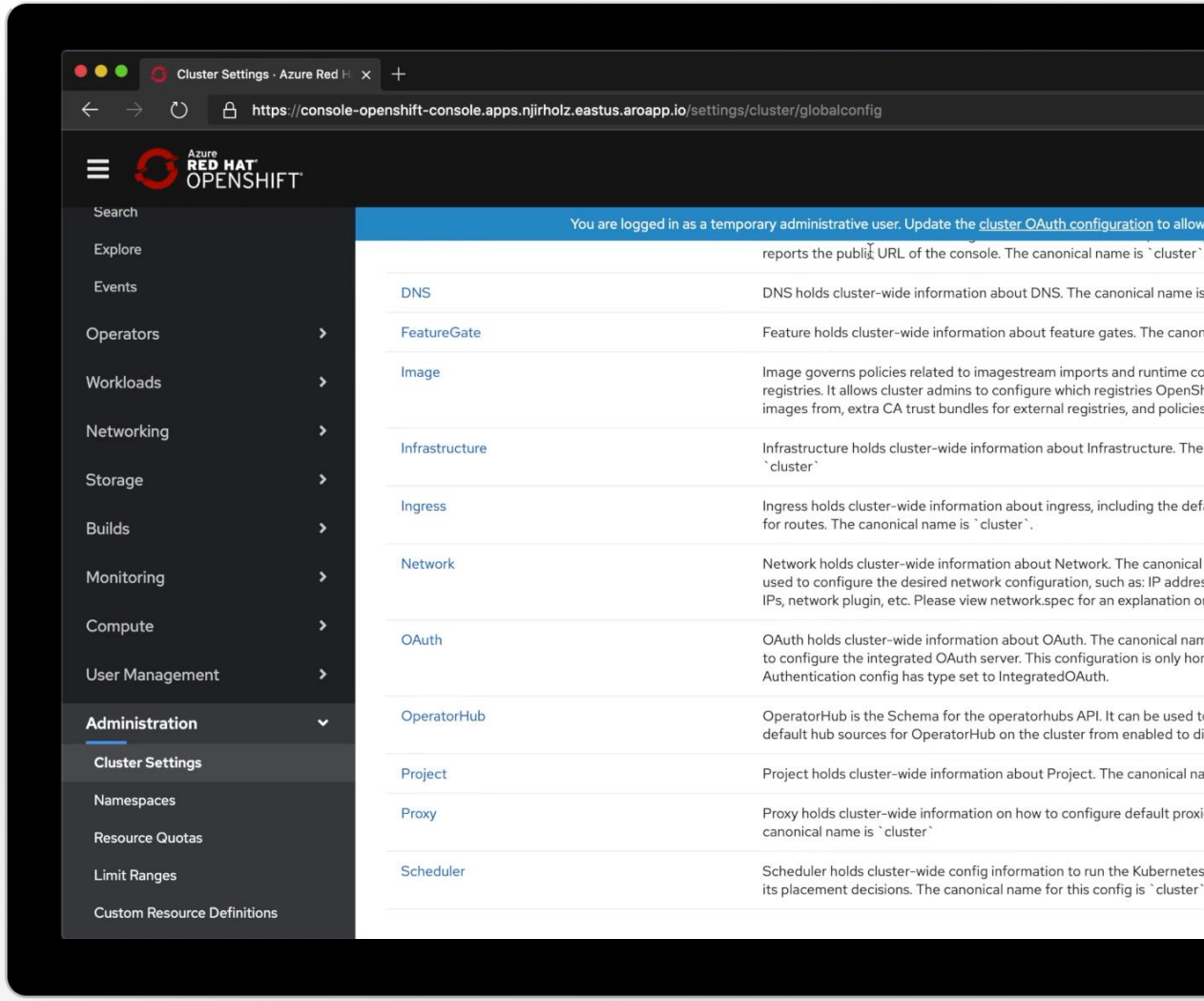
Configure control plane (API) and ingress to use public or private endpoints

The virtual network is configurable, enabling peering with other virtual networks, including Express Route circuits



Single sign-on with your own identity provider

In addition to Azure Active Directory, configure supported OpenShift identity providers, for example using OpenID Connect.

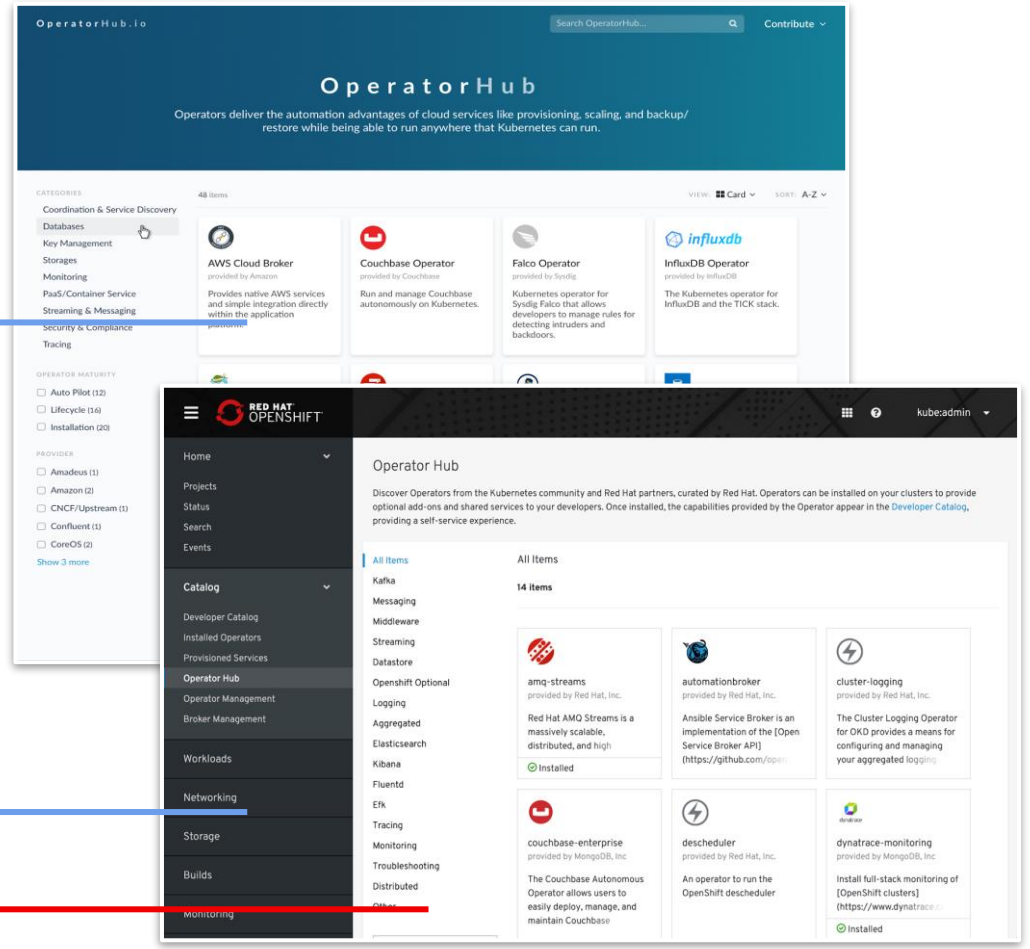


Operator Hub and certified operators

OperatorHub.io launched by Red Hat, Microsoft, AWS and Google

OpenShift Operator Certification

Operator Hub integrated into OpenShift 4



Community Operators

OpenShift Certified Operators

Microsoft | Red Hat

OpenShift developer console

Streamlined console

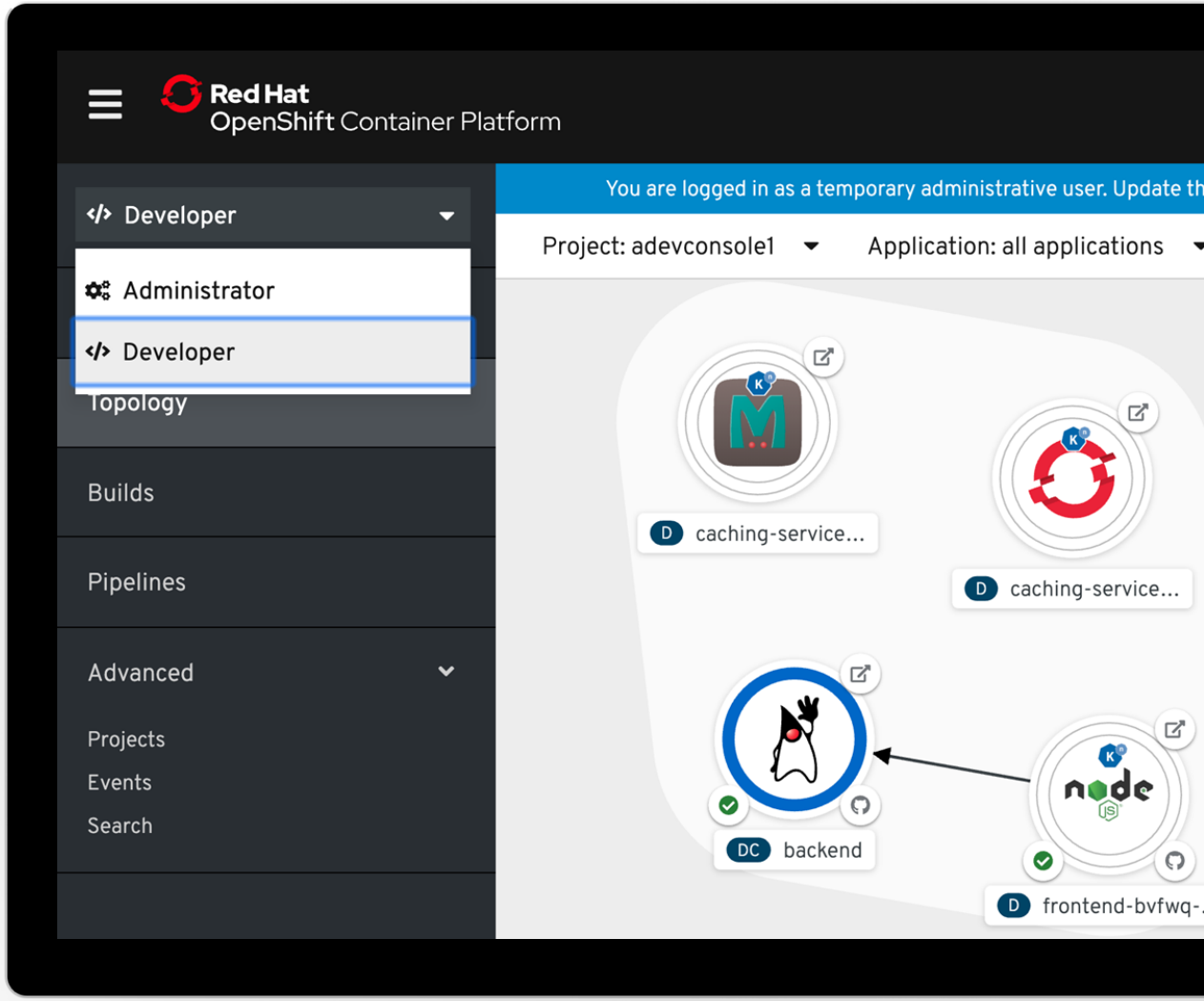
A developer centric perspective that offers a simplified view of Kubernetes concepts so developers can focus on what really matters

Create and manage applications

Import source from Git, view and manage app configuration and workload details, and deploy applications

Application topology

View structure and status of app components, drill into component connections and workload details, quickly navigate to pod logs

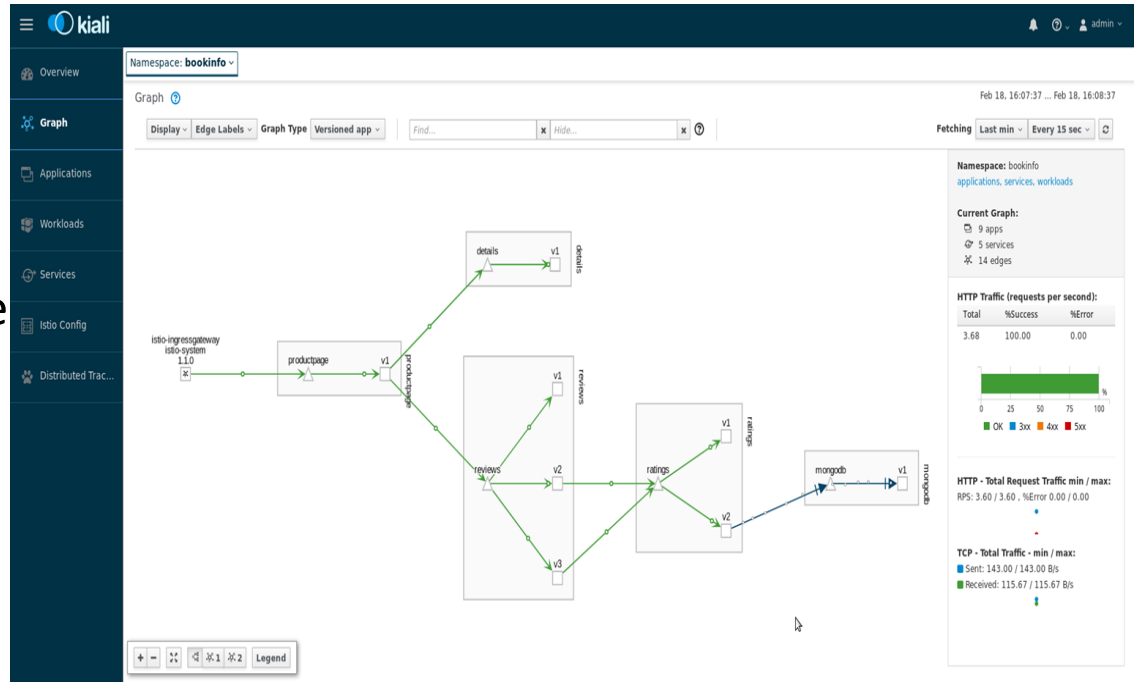


OpenShift Service Mesh

Complete service mesh, including tracing and visualization capabilities, packaged for ease of use

Policy-driven security with a dedicated network for service to service communications

Available via OperatorHub

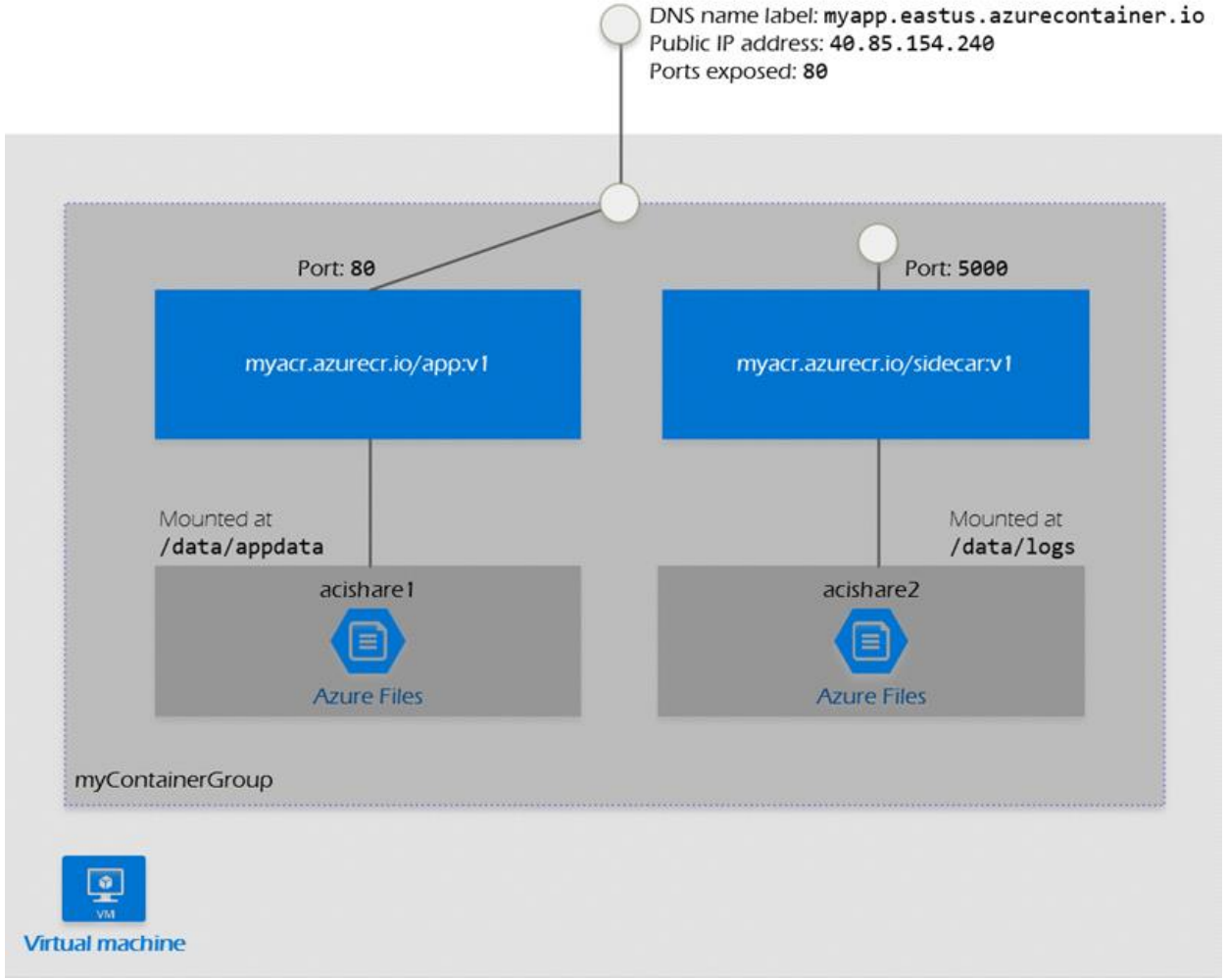


Azure Container Instance

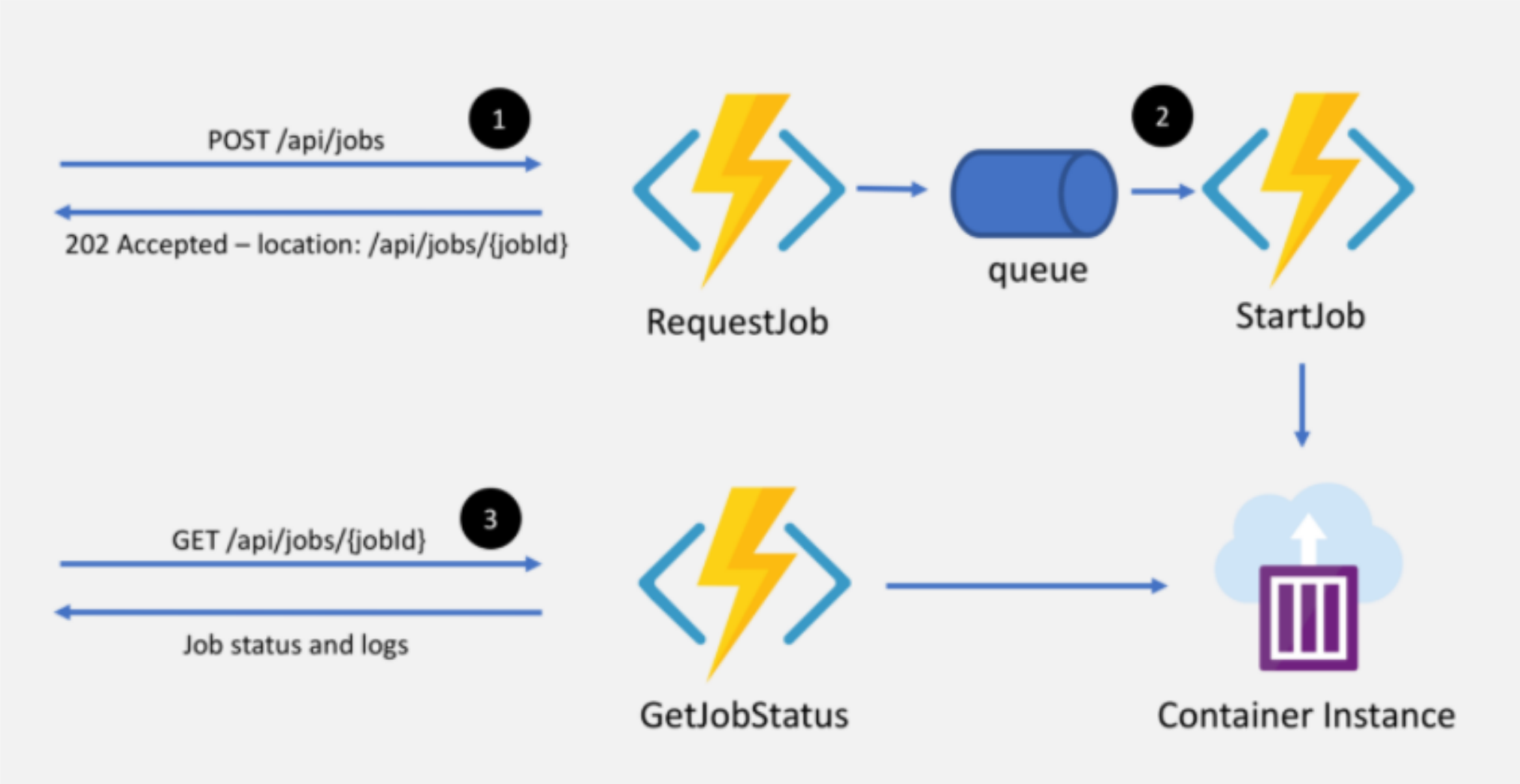
Great solution for solution like task automation and build jobs.

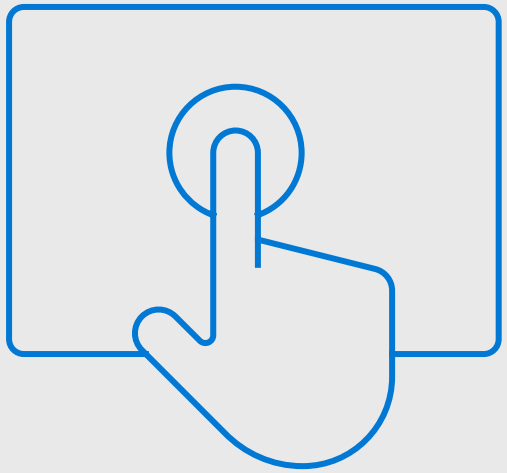
The top-level resource in Azure Container Instances is the *container group*. A container group is a collection of containers that get scheduled on the same host machine.

It's similar in concept to a *pod* in Kubernetes.



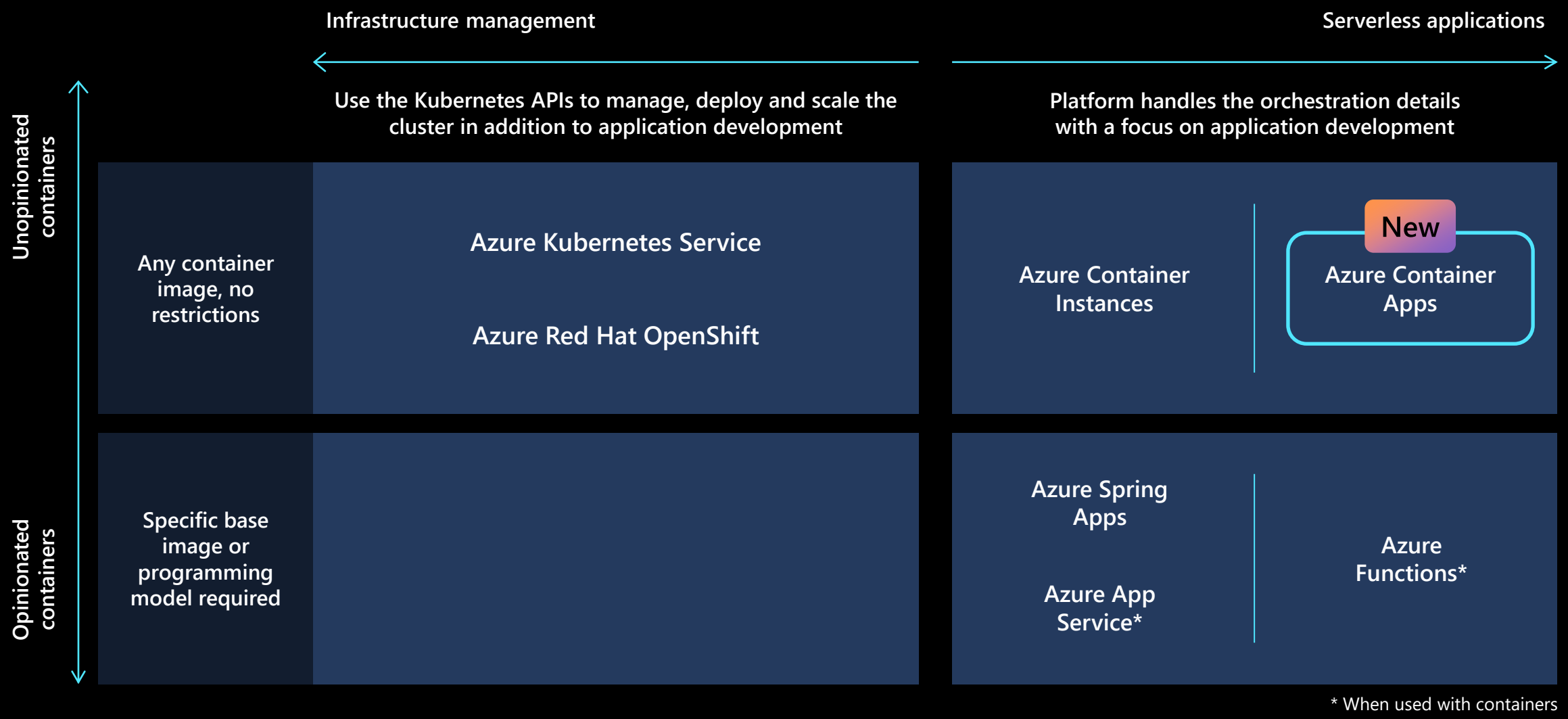
Azure Container Workflow Example





Demo
ACI Virtual Pools

Azure containers portfolio



* When used with containers

Microsoft

Thank you