



Containers as Infrastructure

Containers Advanced Topics

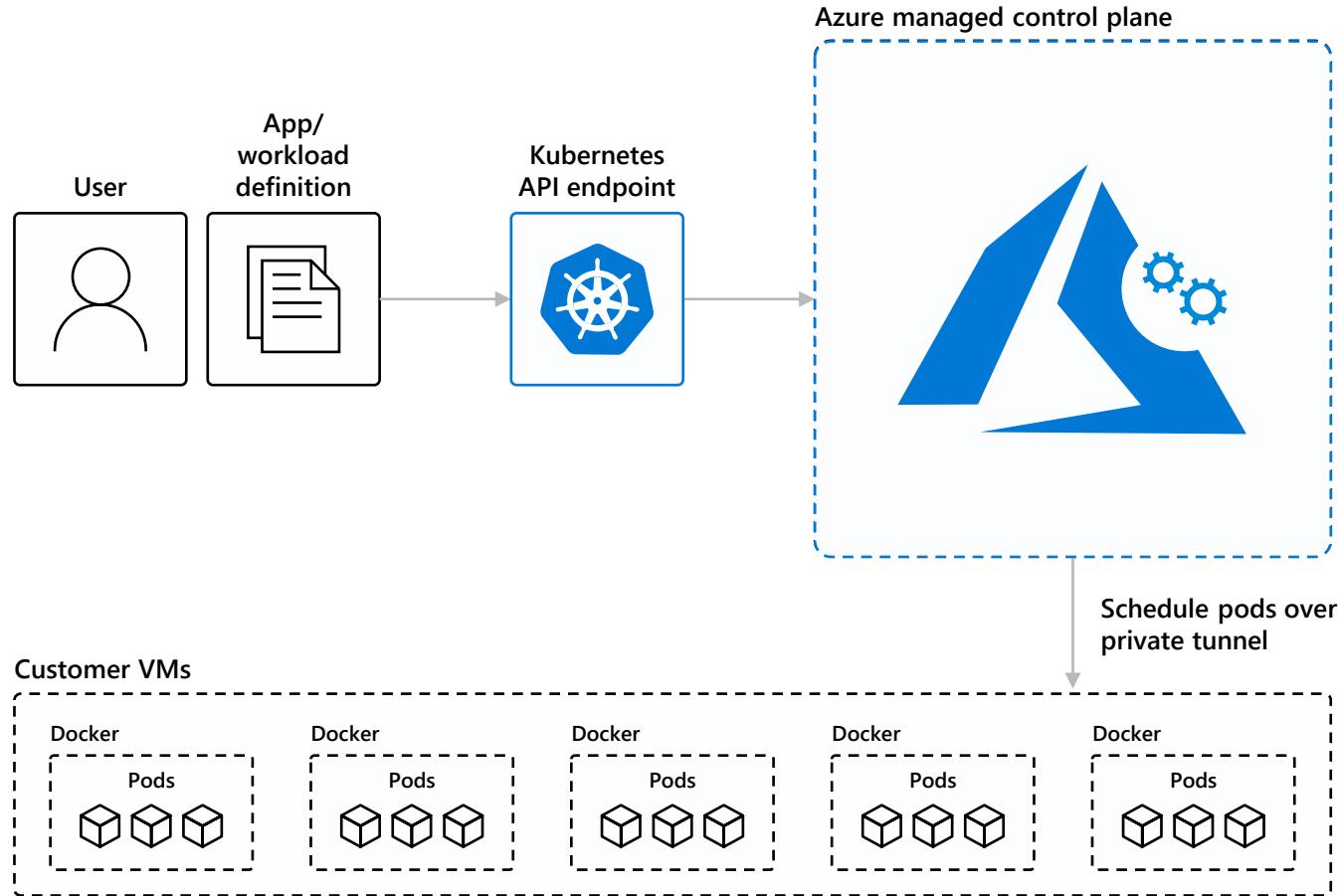
Microsoft Services



Objectives

- Learn how Azure Kubernetes Services (AKS) works
- Learn about AKS advanced concepts and administration
- Learn Azure DevOps for Container Automation

Azure Kubernetes Services (AKS)



Your Kubernetes cluster,
managed by Azure

Why Azure Kubernetes Service?

Easy to use

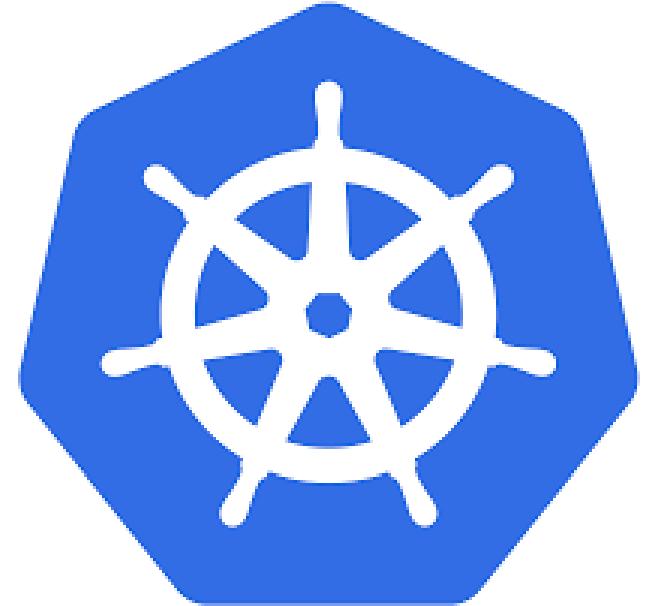
- Fastest path to Kubernetes (K8s) on Azure
- Up and running with 3 simple commands
- AKS is free - you only pay for the agent nodes within your clusters, not for the masters

Easy to manage

- The K8s masters are managed by Azure
- Automated upgrades and patching
- Easily scale the cluster up and down
- Self-healing control plane

Uses Open APIs

- 100% upstream Kubernetes



Why Azure Kubernetes Service?

Focus on your
containers and code,
not the infrastructure

Responsibilities	DIY with Kubernetes	Managed Kubernetes on Azure
Containerization		
Application iteration, debugging		
CI/CD		
Cluster hosting		
Cluster upgrade		
Patching		
Scaling		
Monitoring and logging		

 Customer

 Microsoft

AKS Compliance and Certification

- Azure Kubernetes Service (AKS) has been CNCF certified as Kubernetes conformant.
- Azure Kubernetes Service (AKS) is compliant with SOC, ISO, and PCI DSS.

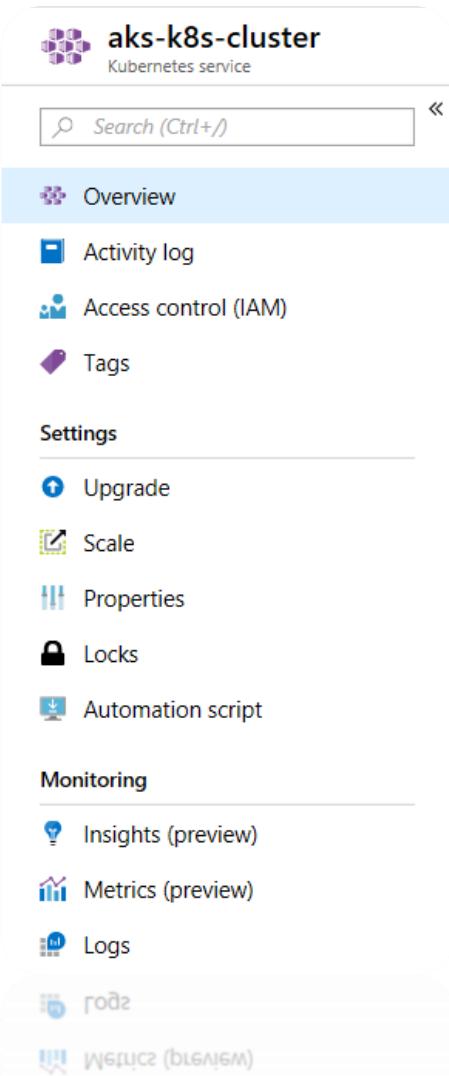


International
Organization for
Standardization



Kubernetes on Azure Infrastructure

- The cluster => Kubernetes Service



- The underlying resources
 - Nodes
 - Networking resources

aks-agentpool-32926962-nsg	Network security group
aks-agentpool-32926962-routetable	Route table
aks-nodepool1-32926962-0	Virtual machine
aks-nodepool1-32926962-0_OsDisk_1_46350a61bd09498b...	Disk
aks-nodepool1-32926962-nic-0	Network interface
aks-vnet-32926962	Virtual network
kubernetes	Load balancer
kubernetes-a01df684ac10c11e8b6880a58ac1f078	Public IP address
nodepool1-availabilitySet-32926962	Availability set
5aee5e-452f-41d1-8b4a-5aee5e-41d1	Load balancer
8701d82e-088a-4b91-901b-48a910e-5aee5e-41d1	Public IP address

Getting Started with AKS

```
$ az aks create -g myResourceGroup -n myCluster --generate-ssh-keys  
\ Running ..
```

```
$ az aks install-cli  
Downloading client to /usr/local/bin/kubectl ..
```

```
$ az aks get-credentials -g myResourceGroup -n myCluster  
Merged "myCluster" as current context ..
```

```
$ kubectl get nodes
```

NAME	STATUS	AGE	VERSION
aks-mycluster-36851231-0	Ready	4m	v1.13.5
aks-mycluster-36851231-1	Ready	4m	v1.13.5
aks-mycluster-36851231-2	Ready	4m	v1.13.5

Managing an AKS Cluster

```
$ az aks list -o table
```

Name	Location	ResourceGroup	KubernetesRelease	ProvisioningState
myCluster	westus2	myResourceGroup	1.13.5	Succeeded

```
$ az aks upgrade -g myResourceGroup -n myCluster --kubernetes-version 1.13.5
\ Running ..
```

```
$ kubectl get nodes
```

NAME	STATUS	AGE	VERSION
aks-mycluster-36851231-0	Ready	12m	v1.13.5
aks-mycluster-36851231-1	Ready	8m	v1.13.5
aks-mycluster-36851231-2	Ready	3m	v1.13.5

```
$ az aks scale -g myResourceGroup -n myCluster --agent-count 10
\ Running ..
```

Demonstration: *Exploring AKS Clusters*

Using kubectl commands





Advanced AKS

AKS Administration

Microsoft Services



Typical AKS Administration

- Planning a cluster
 - On-premises, in the cloud, managed service?
 - Capacity – how many nodes? What configuration? CPU, Memory
 - Cluster resources – compute, storage, networking
- Upgrading Kubernetes
 - New API version
- Node maintenance
 - OS updates, security patches
 - Hardware repair, upgrades
- Security
- Cluster scaling
 - Adding/removing nodes
- Monitoring
- High availability/Disaster recovery

AKS Admin Task

Volume
Persistent volumes
...

Storage

Services
Network Policies
Virtual Network
Ingress
...

Networking

Identity, Access Management and Security

Scale and Cluster Maintenance

RBAC with AAD.
AAD Integration
Secrets with Key Vault.

....

Manually Scale
Horizontal pod autoscaler
Cluster autoscaler
ACI integration with AKS
Kube-Advisor
Updates

....



Advanced AKS

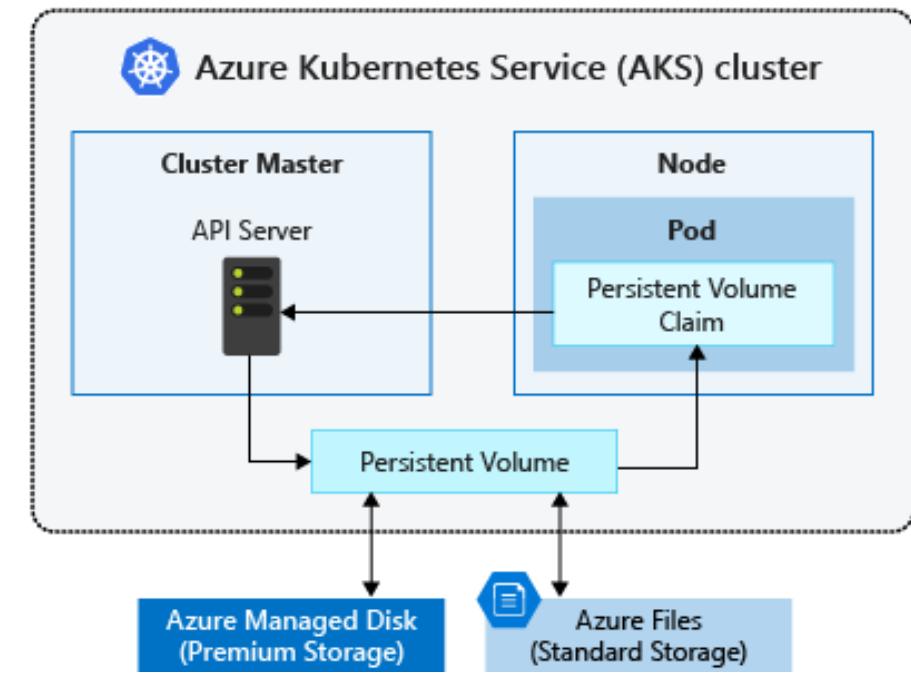
Storage

Microsoft Services



Persistent Volume in AKS

- A persistent volume can be used by one or many pods to read/write persistent data.
- Azure Files and Disk can be mounted statically or dynamically in AKS.
- Azure Disk can only be mounted on one node at a time. (ReadOnlyOnce)
- For persistent volume across multiple nodes, use Azure File.



Setup a Static Persistent Volume in AKS

1- Create the storage resource (Storage Account)

The screenshot shows the Azure Storage Accounts blade. On the left, there's a navigation menu with options like Home, mystorageaccount (selected), Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events, Storage Explorer (preview), Settings, Access keys, CORS, Configuration, Encryption, Shared access signature, Firewalls and virtual networks, Advanced Threat Protection..., and Static website (preview). The main area displays the storage account 'mystorageaccount' with its properties: Resource group (change), Status Primary: Available, Location East US, Subscription (change), Performance/Access tier Standard/Hot, Replication Locally-redundant storage (LRS), Account kind StorageV2 (general purpose v2). Below this, there are sections for Services (Blobs, Files, Tables, Queues) and Tags (change). A green arrow points from the 'Overview' link in the navigation bar to the 'Overview' section of the main content area.

2- Create a Storage Class object

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: azurefile
  provisioner: kubernetes.io/azure-file
  mountOptions:
    - dir_mode=0777
    - file_mode=0777
    - uid=1000
    - gid=1000
parameters:
  skuName: Standard_LRS
  storageAccount: mystorageaccount
```

Setup a Static Persistent Volume in AKS

- 3- Create a cluster role and binding to authorize access
- 4- Create a persistent volume claim to provision the share
- 5- Reference shared volume from pod

```
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRole
metadata:
  name: system:azure-cloud-provider
rules:
- apiGroups: ['']
  resources: ['secrets']
  verbs: ['get', 'create']
---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRoleBinding
metadata:
  name: system:azure-cloud-provider
roleRef:
  kind: ClusterRole
  apiGroup: rbac.authorization.k8s.io
  name: system:azure-cloud-provider
subjects:
- kind: ServiceAccount
  name: persistent-volume-binder
  namespace: kube-system
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: azurefile
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: azurefile
  resources:
    requests:
      storage: 5Gi
```

```
kind: Pod
apiVersion: v1
metadata:
  name: mypod
spec:
  containers:
    - name: myfrontend
      image: nginx
      volumeMounts:
        - mountPath: "/mnt/azure"
          name: volume
  volumes:
    - name: volume
      persistentVolumeClaim:
        claimName: azurefile
```



Advanced AKS

Identity, Access Management and Security

Microsoft Services



What is up with all these Service Principals?

- Cluster Service Principal
 - To allow an AKS cluster to interact with other Azure resources, an Azure Active Directory service principal is used.
- AD Integration Objects
 - When configuring Azure AD for AKS authentication, 2 Azure AD application are configured.
 - Server Application - used to get a users Azure AD group membership
 - Client Application - used when logging in with the Kubernetes CLI (kubectl)
- You can rotate/ update them with:

```
$ az aks update-credentials --resource-group myResourceGroup --name myAKSCluster --reset-service-principal --service-principal $SP_ID --client-secret $SP_SECRET
```

Role Based Access Control (RBAC)

- In Kubernetes, RBAC is defined with 4 top-level types
 - Cluster Role – Describes a role at the cluster level
 - Role – Describes a role at the namespace level
 - ClusterRoleBinding – Grants permissions defined in a Role at the cluster level to a user, group of users or service account
 - RoleBinding – Grants permissions defined in a Cluster Role or Role at the namespace level to a user, group of users or service account

```
kind: Role ←  
apiVersion: rbac.authorization.k8s.io/v1  
metadata:  
  namespace: default  
  name: pod-reader  
rules:  
- apiGroups: [""]  
  resources: ["pods"]  
  verbs: ["get", "watch", "list"]
```

```
# This role binding allows "jane" to read pods in the "default" namespace.  
kind: RoleBinding ←  
apiVersion: rbac.authorization.k8s.io/v1  
metadata:  
  name: read-pods  
  namespace: default  
subjects:  
- kind: User  
  name: jane # Name is case sensitive  
  apiGroup: rbac.authorization.k8s.io  
roleRef:  
  kind: Role #this must be Role or ClusterRole  
  name: pod-reader # this must match the name of the Role or ClusterRole you wish to bind to  
  apiGroup: rbac.authorization.k8s.io
```

Azure Active Directory Integration

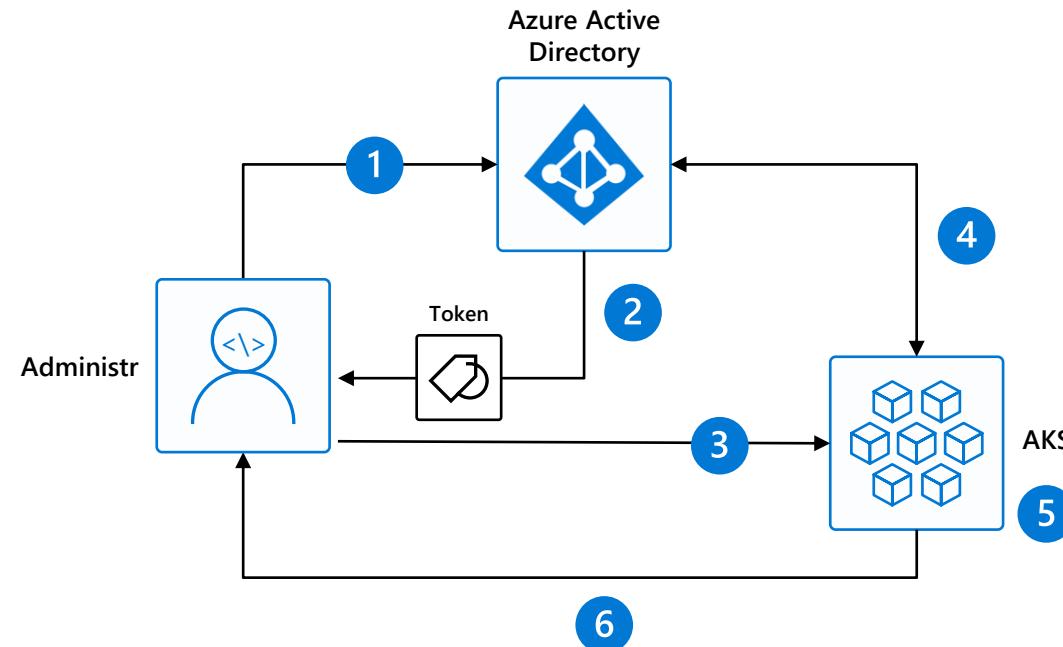
- Cluster RBAC can be backed by Azure Active Directory
- Allows user identity to be managed in Azure
- To log into AKS cluster, user will use their AAD authentication token provided with OpenID Connect
 - AKS must have read access to the AAD to read group membership
 - A second AAD application must be created to be able to log in with kubectl

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: contoso-cluster-admins
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: "894656e1-39f8-4bfe-b16a-510f61af6f41"
```

The group Id will need to be defined in the AAD tenant set at cluster creation

Azure Active Directory Integration

1. Kubernetes Administrator authenticates with AAD
2. The AAD token issuance endpoint issues the access token
3. Developer performs action w/ AAD token.
Eg. `kubectl create pod`
4. Kubernetes validates token with AAD and fetches the Developer's AAD Groups
Eg. Dev Team A, App Group B
5. Kubernetes RBAC and cluster policies are applied
6. Request is successful or not based on the previous validation



Kubernetes Admission Controllers

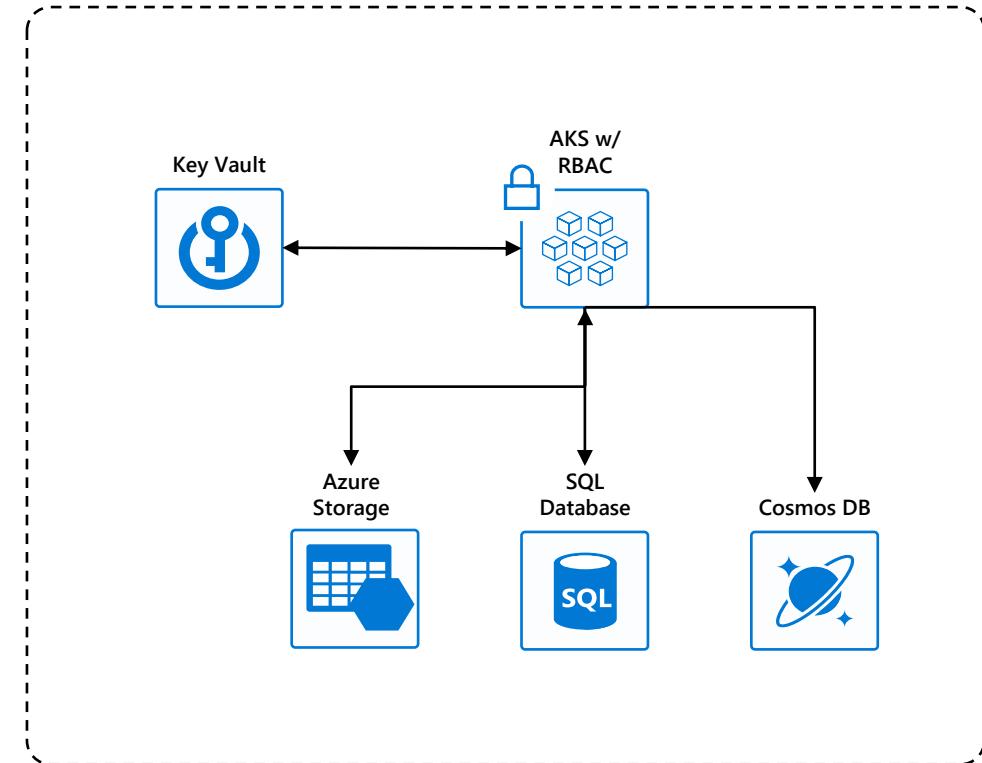
- What is an Admission Controller?
 - Intercepts requests to the Kubernetes API server prior to persistence of the object, but after the request is authenticated and authorized
 - List of features can be enabled to define rules enforcing cluster level policies
- In AKS, we cannot modify the list of Admission controller

- NamespaceLifecycle
- LimitRanger
- ServiceAccount
- DefaultStorageClass
- DefaultTolerationSeconds

- MutatingAdmissionWebhook
- ValidatingAdmissionWebhook
- ResourceQuota
- DenyEscalatingExec
- AlwaysPullImages

Azure Key Vault Integration

- Azure Key Vault is a secret, certificate and key management service.
- FlexVolume is an optional daemonset that enables Kubernetes pods to use Key Vault secrets, certificates, or keys without application integration.
 - Secrets are backed by Azure Key Vault storage
 - To authenticate to the Key Vault, use SP credentials or AAD Pod Identity



[Azure / kubernetes-keyvault-flexvol](#)

[Watch](#) 16

[Unstar](#) 42

[Fork](#) 11

[Code](#)

[Issues 8](#)

[Pull requests 0](#)

[Projects 0](#)

[Wiki](#)

[Insights](#)

Azure keyvault integration with Kubernetes via a Flex Volume



Advanced AKS

Networking

Microsoft Services



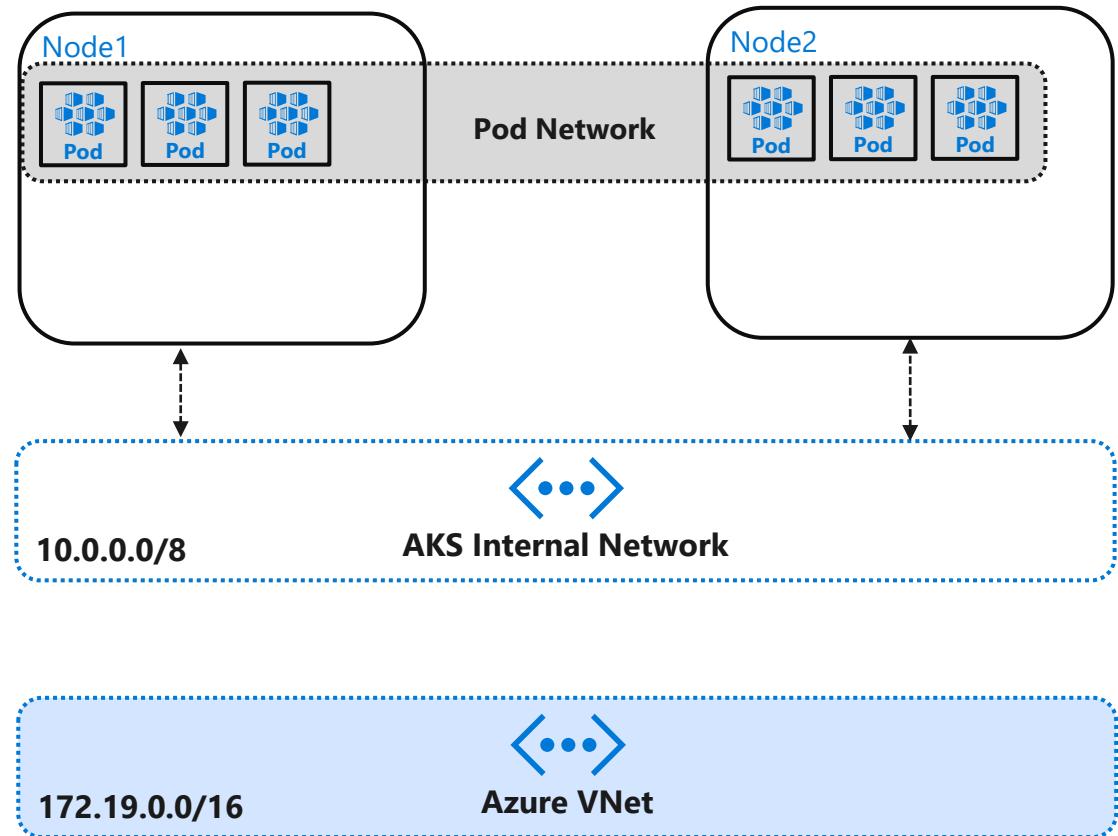
Basic Networking

Uses kubenet network plugin and has the following features

- Nodes and Pods are placed on different IP subnets
- User Defined Routing and IP Forwarding is for connectivity between Pods across Nodes

Drawbacks

- 2 different IP CIDRs to manage
- Performance impact
- Peering or On-premise connectivity is hard to achieve

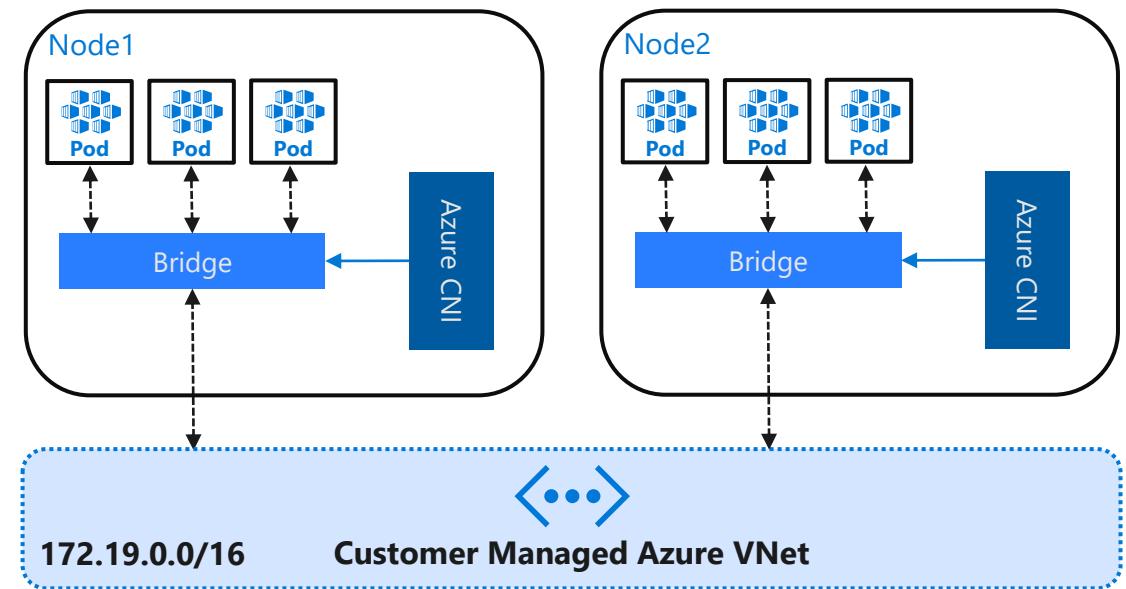


Advanced Networking

- **CNI** is a vendor-neutral protocol, used by container runtimes to make requests to Networking Providers
- **Azure CNI** is an implementation which allows you to integrate Kubernetes with your VNET

Advantages

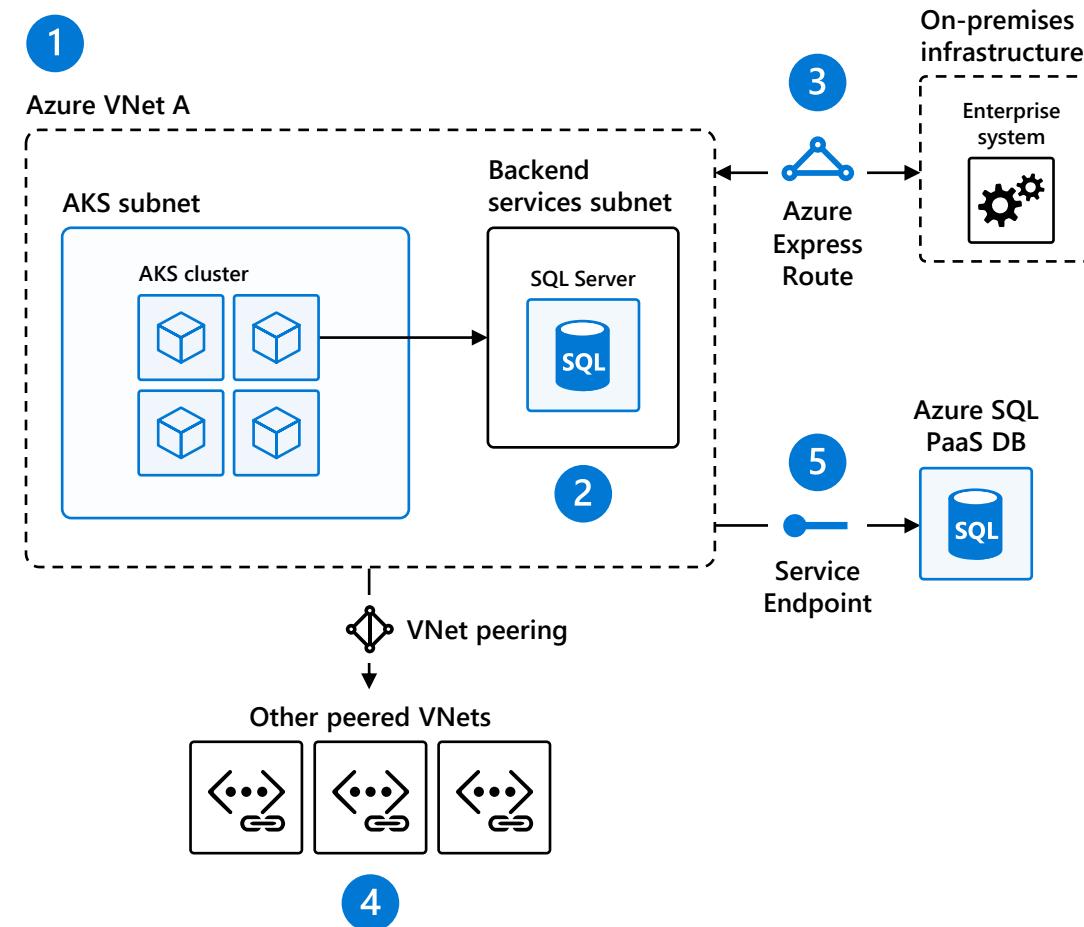
- Single IP CIDR to manage
- Better performance
- Peering and On-Premise connectivity is out of the box



Scenarios enabled by Advanced Networking

1. Uses Azure subnet for both your containers and cluster VMs
2. Allows for connectivity to existing Azure services in the same VNet
3. Use Express Route to connect to on-premises infrastructure
4. Use VNet peering to connect to other VNets
5. Connect AKS cluster securely and privately to other Azure resources using VNet endpoints

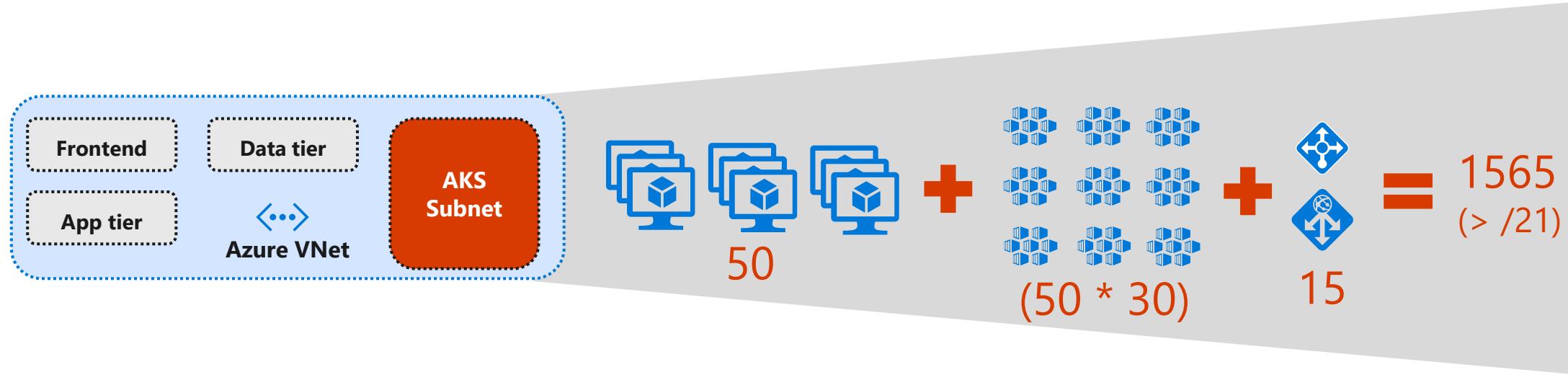
AKS VNet integration works seamlessly with your existing network infrastructure



Advanced Networking: Prerequisites

- Virtual networks for AKS clusters must allow outbound internet connectivity
- No more than one AKS cluster in the same subnet
- AKS clusters may not use 169.254.0.0/16, 172.30.0.0/16, or 172.31.0.0/16 for the Kubernetes service address range
- The service principal used by the AKS cluster must have at least Network Contributor permissions on the subnet within your virtual network. If you wish to define a custom role instead of using the built-in Network Contributor role, the following permissions are required:
 - Microsoft.Network/virtualNetworks/subnets/join/action
 - Microsoft.Network/virtualNetworks/subnets/read

Advanced Networking: Planning your Cluster



Additional subnets

Kubernetes service address range

- Non-overlapping with other subnets in your network (does not need to be routable) and not 169.254.0.0/16, 172.30.0.0/16 and 172.31.0.0/16.
- Smaller than /12

Docker bridge address

- Non-overlapping with AKS Subnet

Service, Ingress and Ingress Controllers

Service

- Logical set of Pods and a policy by which to access them.
- The set of Pods targeted by a Service is (usually) determined by a Label Selector.
- Services provide important features that are standardized across the cluster: load-balancing (L4), service discovery between applications, and features to support zero-downtime application deployments.

Ingress

- A Kubernetes API that manages external access to the services in the cluster
- Supports HTTP and HTTPS
 - Path and Subdomain based routing
 - SSL Termination
 - Serve on Public IPs

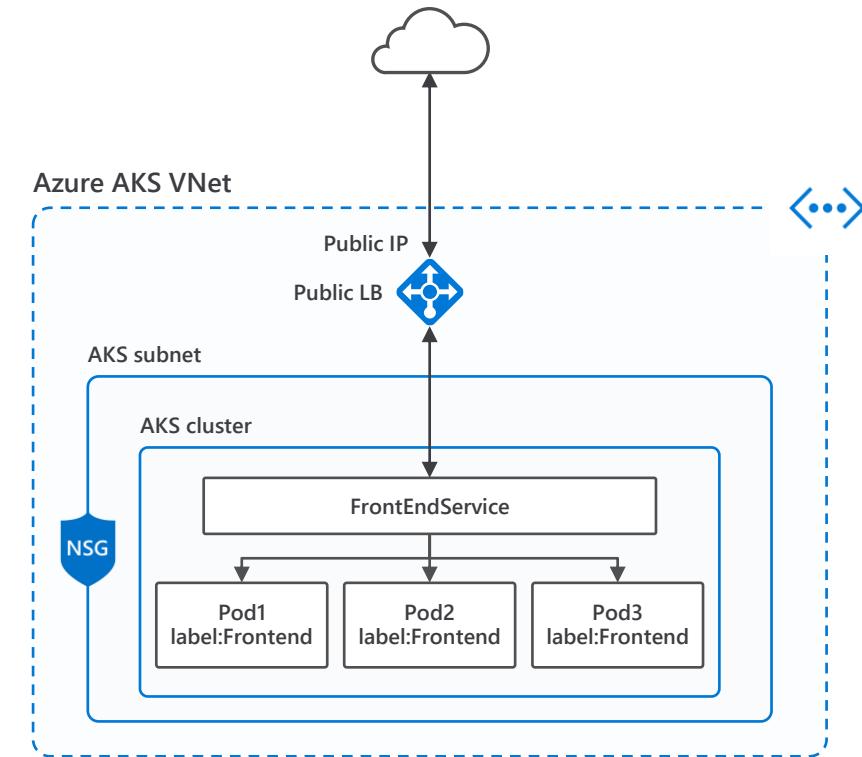
Ingress Controller

A daemon, deployed as a Kubernetes Pod, that watches the Ingress Endpoint for updates. Its job is to satisfy requests for ingresses.
Some popular Ingress Controllers include nginx, Traefik and HAProxy.

Public Service

- Service type LoadBalancer
- Basic layer 4 load balancing (TCP/ UDP)
- Each service is assigned an IP and Public IP on the load balancer

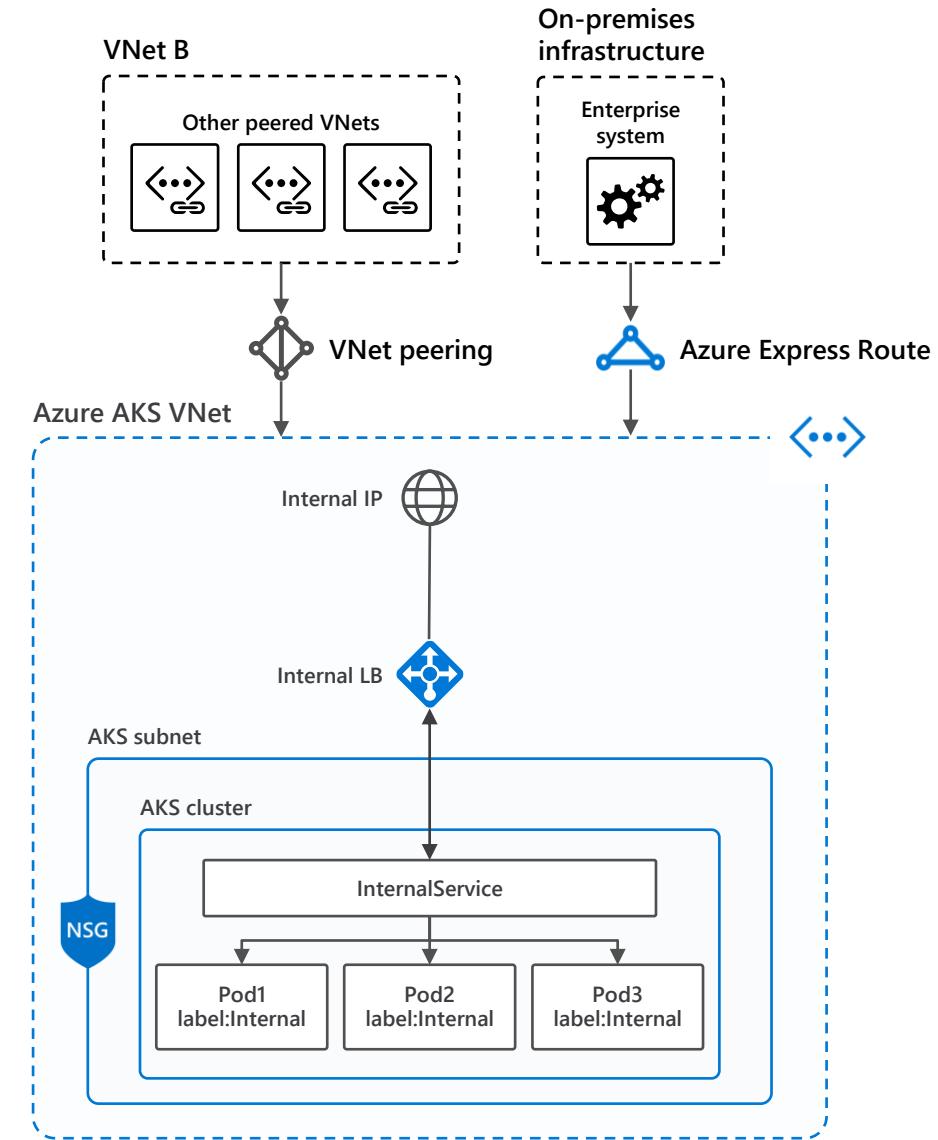
```
apiVersion: v1
kind: Service
metadata:
  name: frontendservice
spec:
  loadBalancerIP: X.X.X.X
  type: LoadBalancer
  ports:
    - port: 80
  selector:
    app: frontend
```



Internal Service

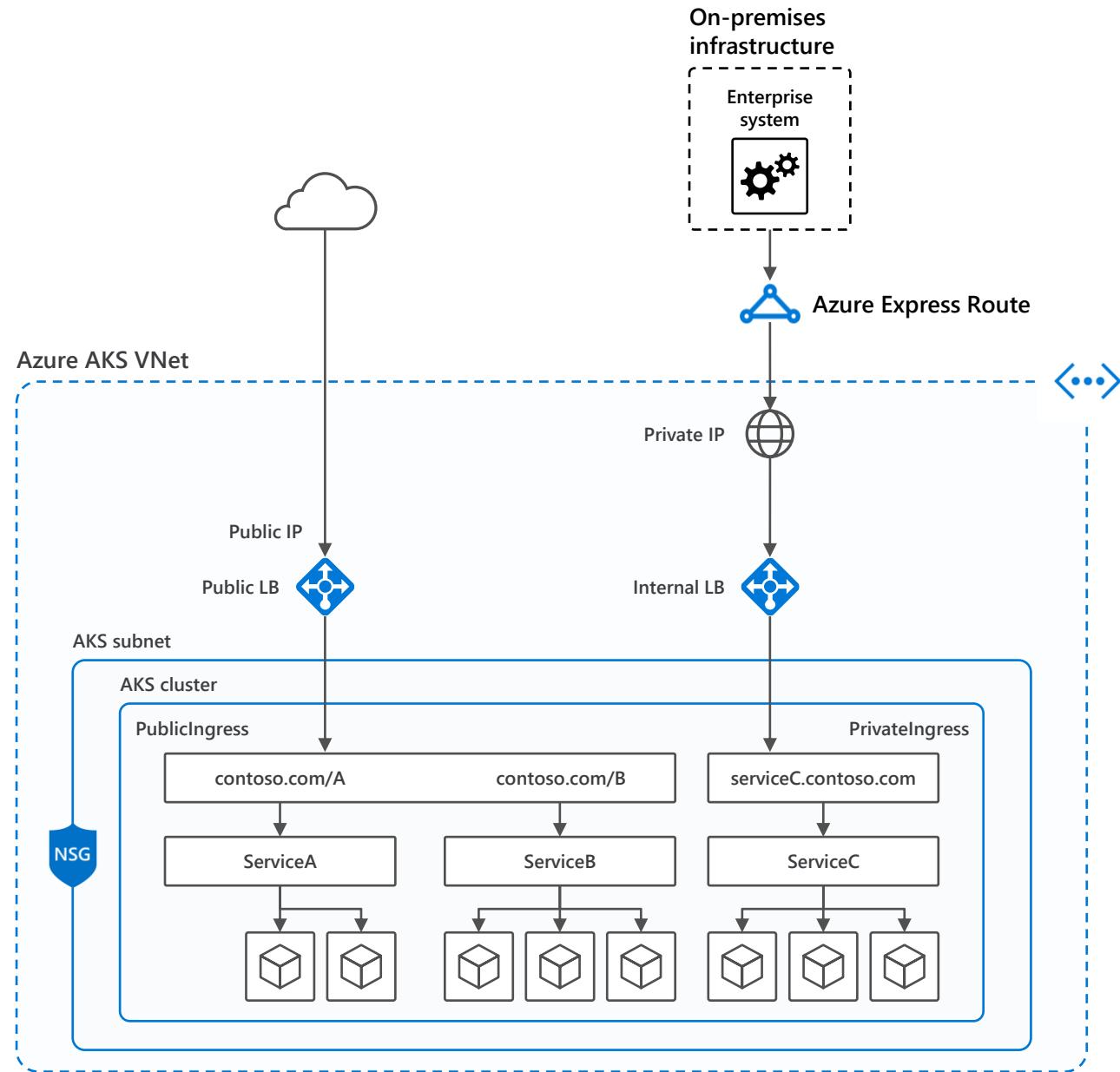
- Service type LoadBalancer
- Basic layer 4 load balancing (TCP/ UDP)
- Each service is assigned a subnet IP on the load balancer

```
apiVersion: v1
kind: Service
metadata:
  name: internalservice
  annotations:
    service.beta.kubernetes.io/azure-load-balancer-internal:
    "true"
spec:
  type: LoadBalancer
  loadBalancerIP: 10.240.0.25
  ports:
  - port: 80
  selector:
    app: internal
```

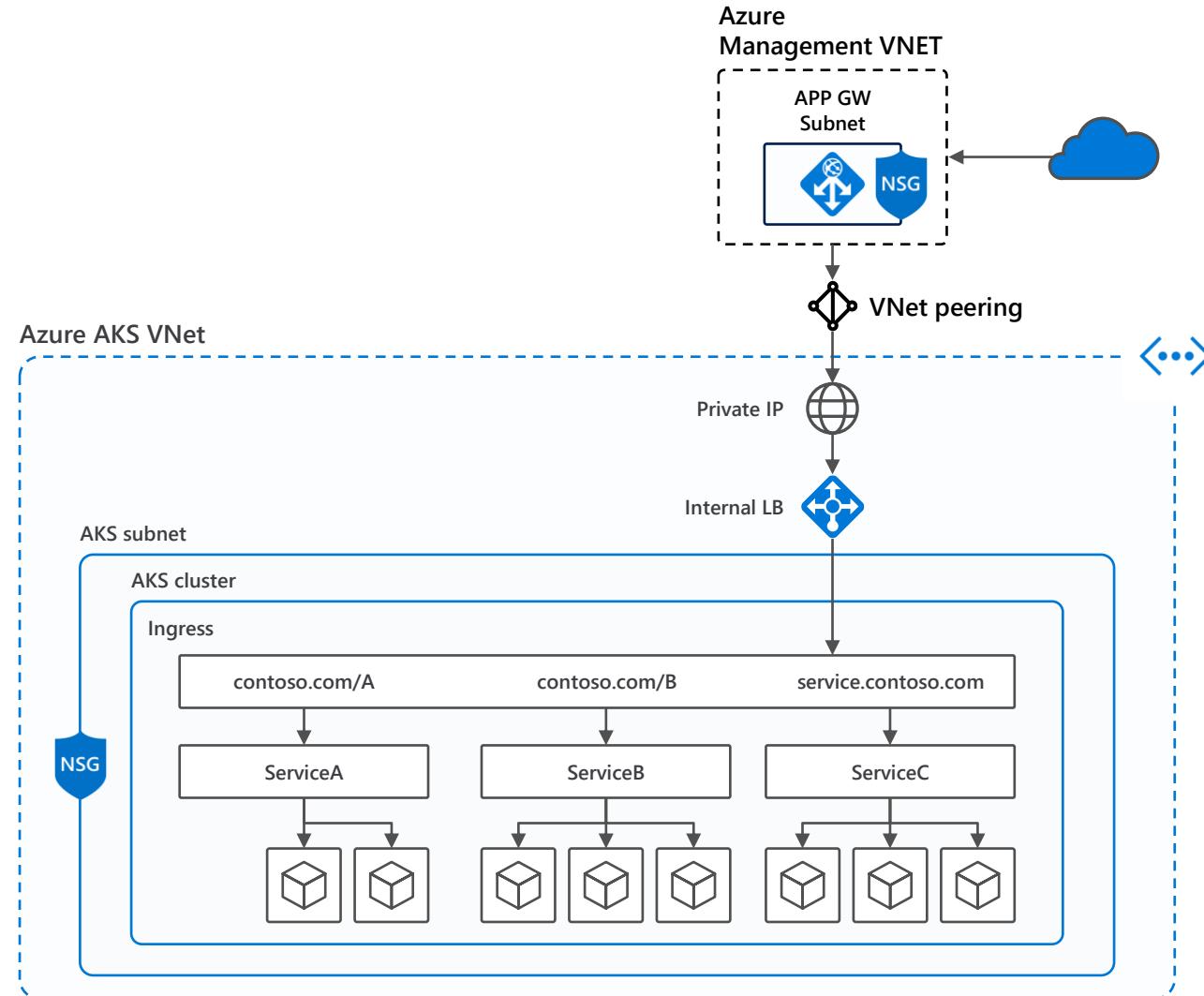


Ingress

```
kind: Ingress
metadata:
  name: contoso-ingress
  annotations: kubernetes.io/ingress.class:
    "PublicIngress"
spec:
  tls:
    - hosts:
        - contoso.com
      secretName: contoso-secret
  rules:
    - host: contoso.com
      http:
        paths:
          - path: /a
            backend:
              serviceName: servicea
              servicePort: 80
          - path: /b
            backend:
              serviceName: serviceb
              servicePort: 80
```



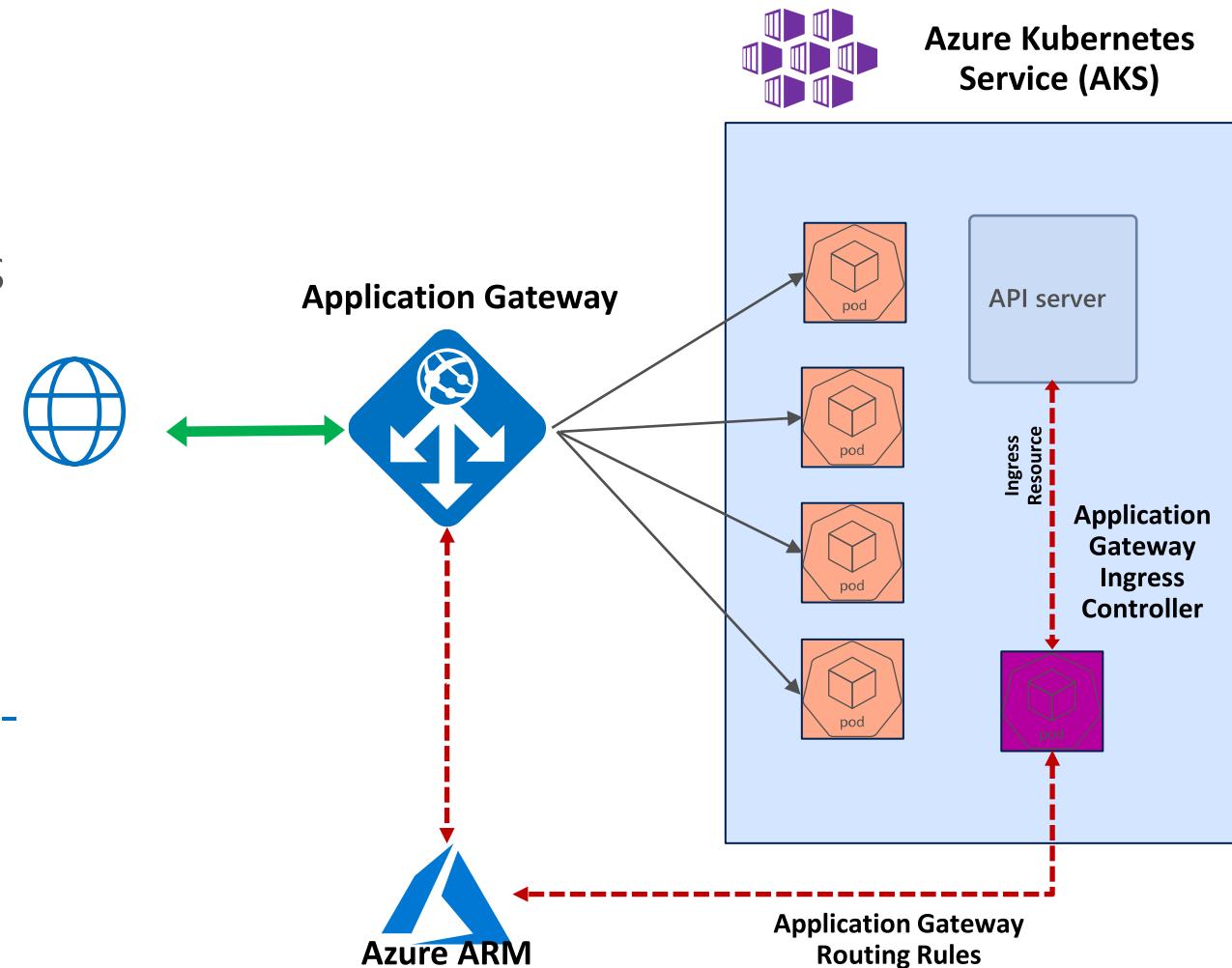
Application Gateway Ingress



Application Gateway Ingress Controller

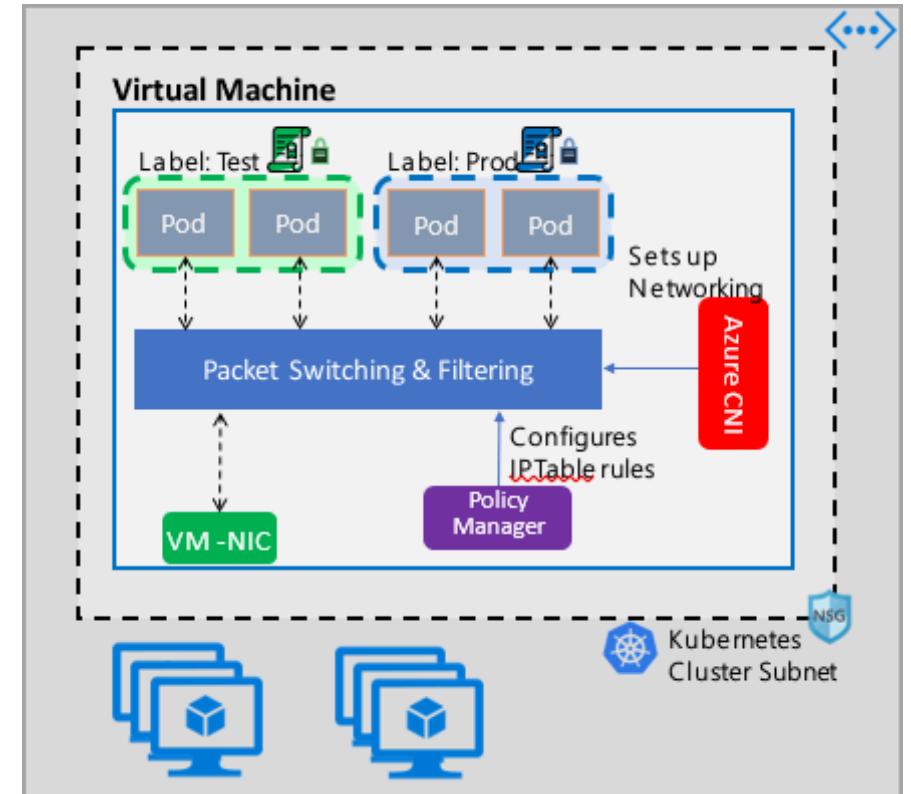
- Attach Application Gateways to AKS Clusters
- Load Balance from the Internet to pods
- Supports features of k8s ingress resource – TLS, multi-site and path-based routing

<https://github.com/Azure/application-gateway-kubernetes-ingress>



Network Policy

- Provides micro-segmentation for containers – think NSGs for VMs
- Label-based selection of Pods
- Policy resource yaml file specifies Ingress and Egress policies
- Works in conjunction with Azure CNI
- Supported in aks-engine
 - Set `networkPolicy` setting to `azure` in cluster definition file



Network Policy

- Pod Selector
- PolicyTypes
- Ingress, egress
 - Ingress
- namespaceSelector
- podSelector
- ipBlock

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
    - Ingress
    - Egress
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
  egress:
    - to:
        - ipBlock:
            cidr: 10.0.0.0/24
```

When to use NSG vs Network Policy

- Azure Network Security Groups
 - Use it to filter North-South traffic, that is, traffic entering and leaving your cluster subnet
- Kubernetes Network Policies
 - Use it to filter East-West traffic, meaning, traffic between pods in your cluster

Demonstration: *Network*

Running your app with services exposed via ELB

Working with Network Policies





Advanced AKS

Scaling

Microsoft Services



Manually Scale

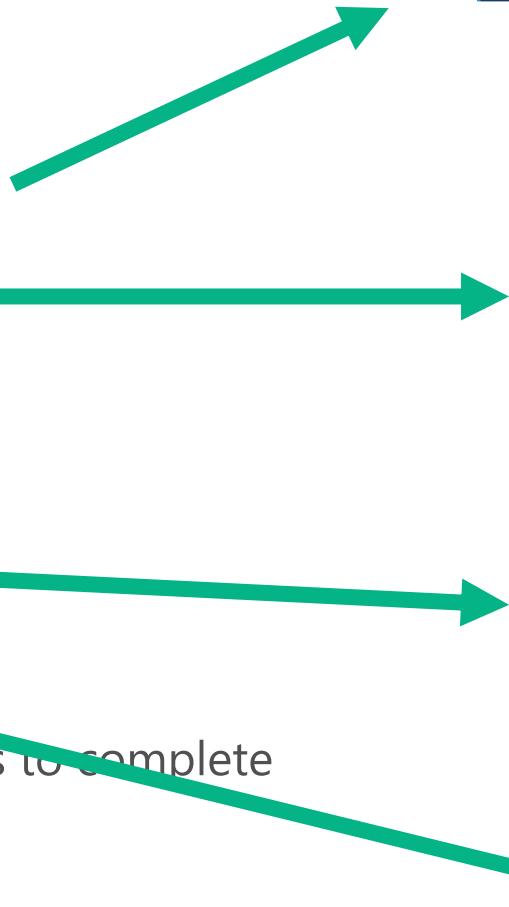
■ Scaling Pods

- Using the kubectl CLI for
- From the Kubernetes dashboard

■ Scaling Nodes

- Using the Azure CLI
- From the Azure Portal
 - Adding a node takes several minutes to complete

Manual scaling is tedious and ineffective



```
PS C:\> kubectl scale deployments/nginx --replicas=4  
deployment "nginx" scaled
```

Screenshot of the Kubernetes dashboard showing the 'Deployments' page for the 'nginx' deployment. The dashboard displays CPU and Memory usage over time, and a table with deployment details. A green arrow points from the 'Scaling Pods' section towards this screenshot.

```
az aks scale --resource-group=MyResourceGroup --name=myAKScluster
```

Screenshot of the Azure Portal's 'aks-k8s-cluster - Scale' blade. It shows cluster capacity settings and a slider for adding nodes. A green arrow points from the 'Scaling Nodes' section towards this screenshot.

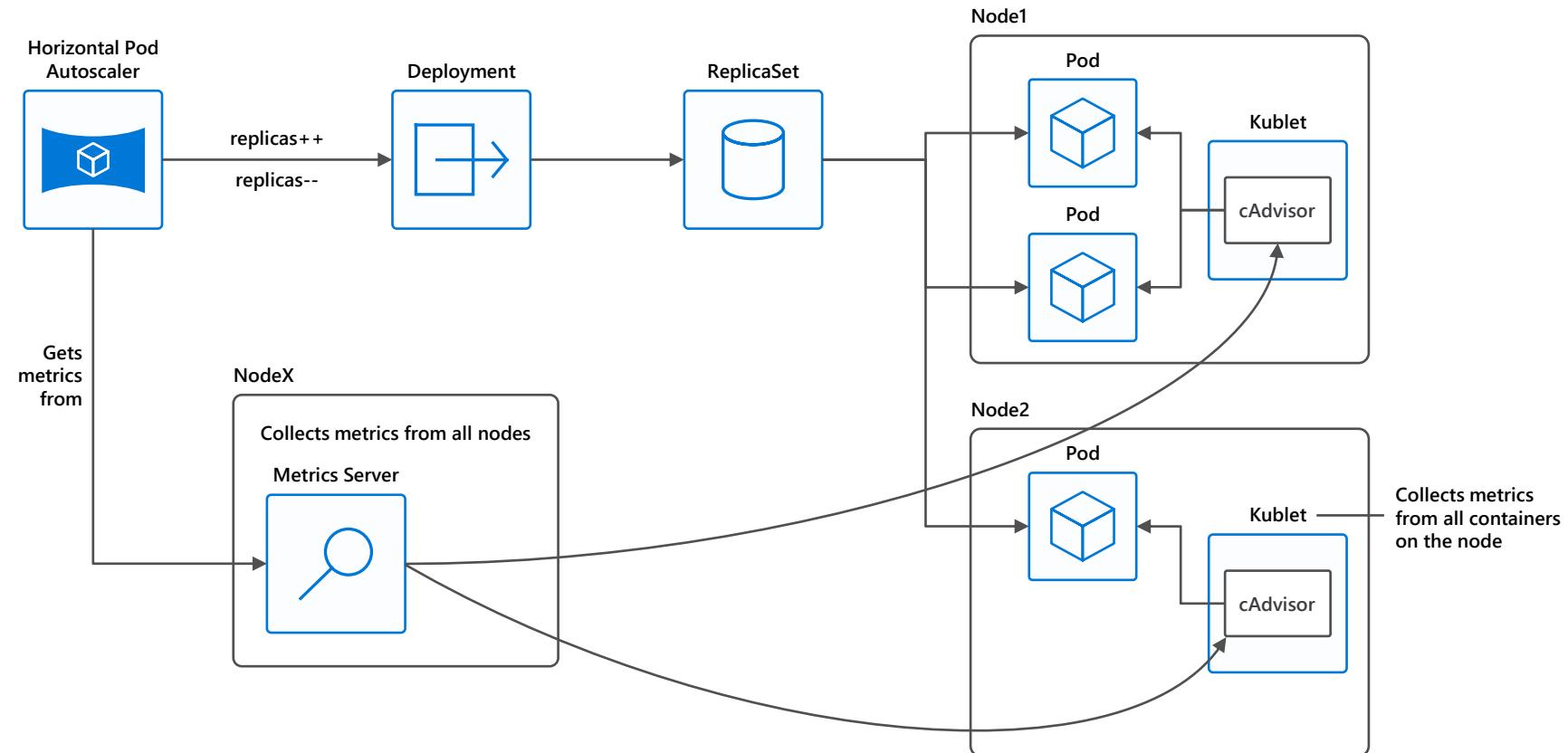
Horizontal Pod Autoscaler

- Works on built-in metrics as well as custom or application-provided metrics
- Can be setup imperatively with `kubectl autoscale` command or declaratively with the `HorizontalPodAutoscaler` K8s object
- Convenient to scale deployments

```
$ kubectl autoscale deployment azure-vote-front --cpu-percent=50 --min=3 --max=10
```

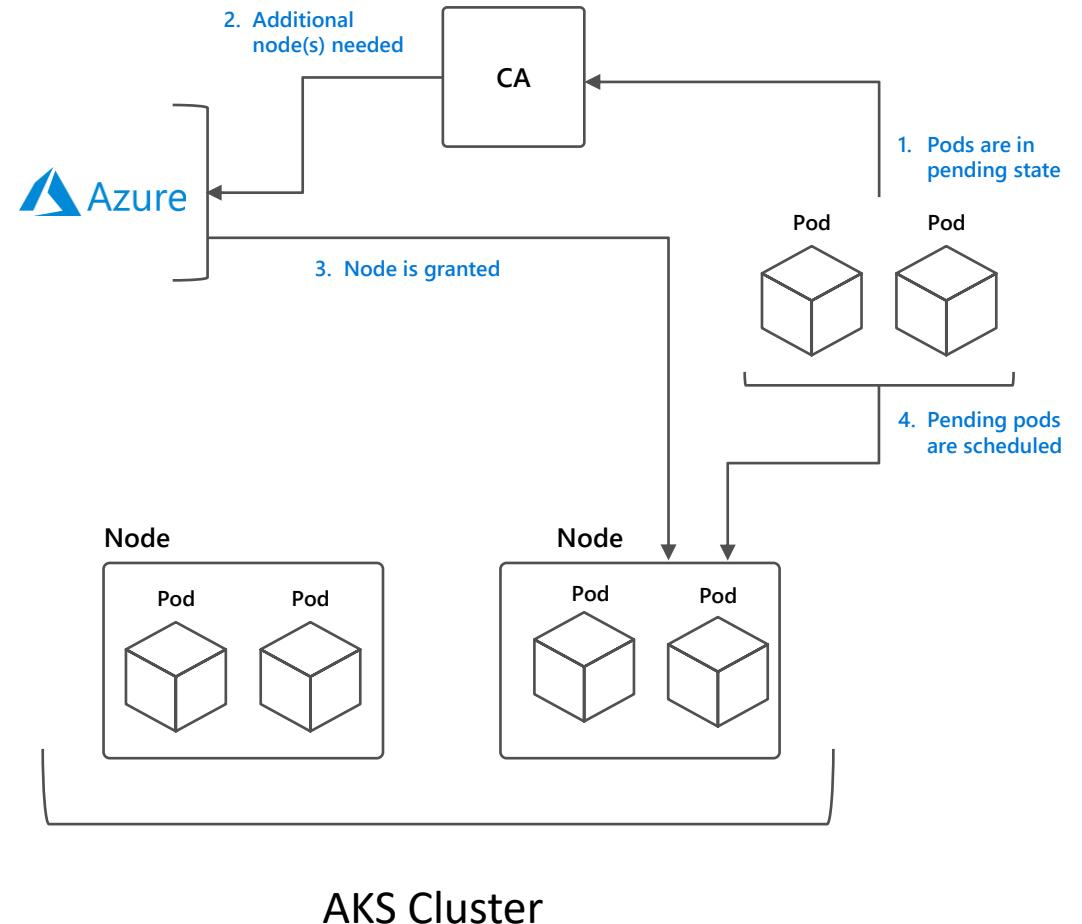
```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: php-apache
  namespace: default
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: php-apache
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
    target:
      type: Utilization
      averageUtilization: 50
```

How HPA works



Cluster Autoscaler

- Scales nodes based on pending pods
- Scale up and scale down
- Reduces dependency on monitoring
- Removes need for users to manage nodes and monitor service usage manually



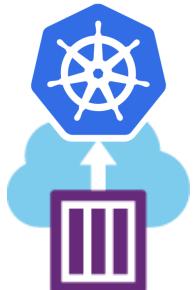
Container Orchestrator with Infinite Scale



Kubernetes provides rich orchestration capabilities

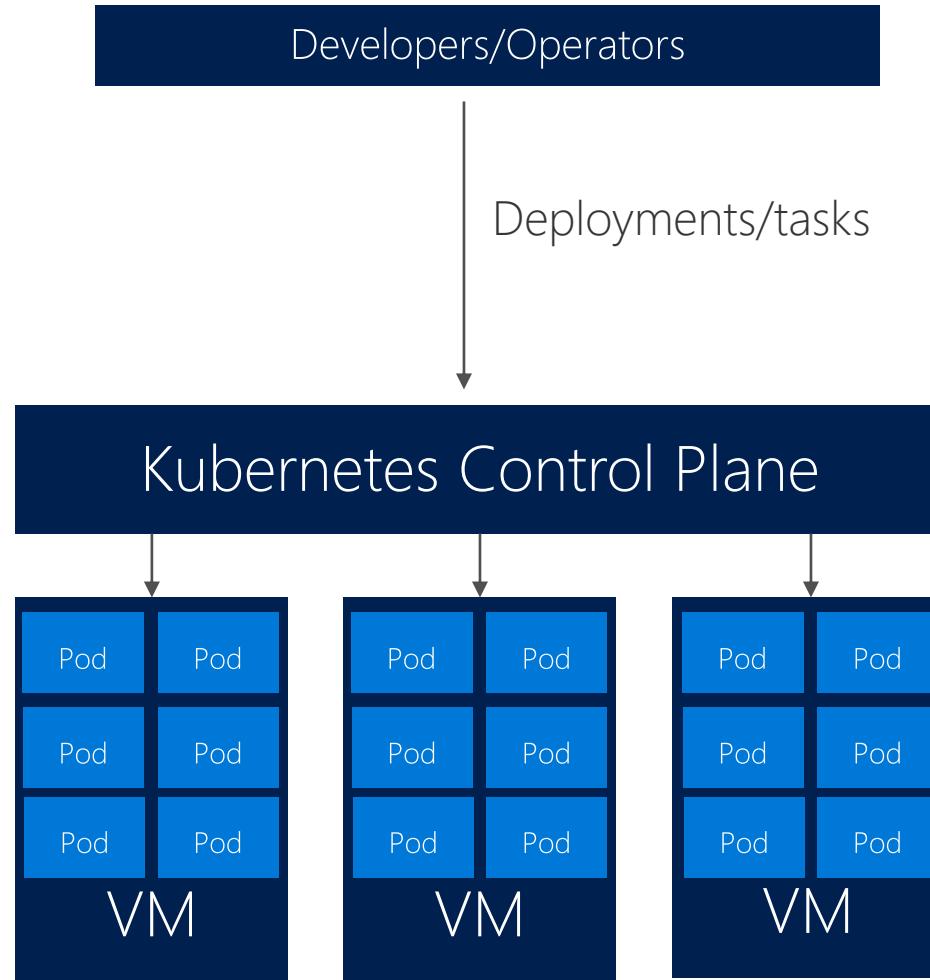


Azure Container Instances provides infinite container-based scale

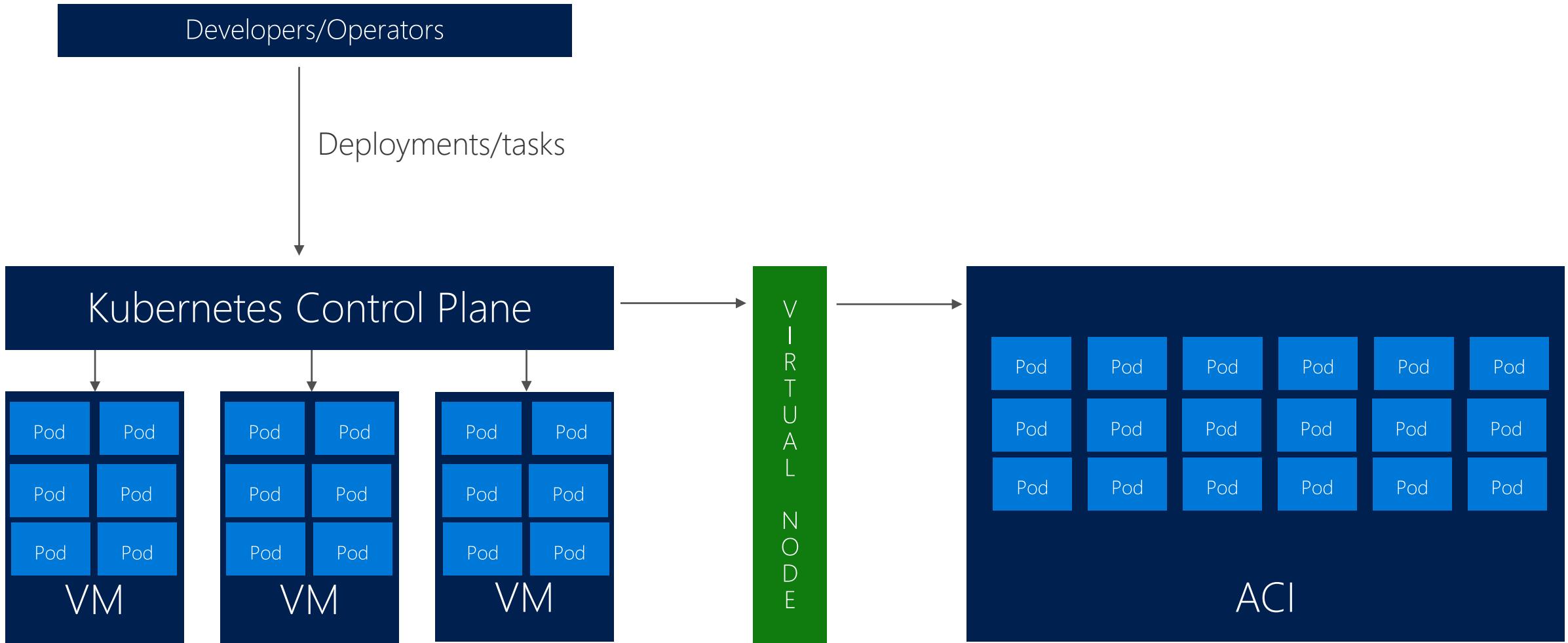


The ACI Connector for Kubernetes brings them together

Bursting with the Virtual Node



Bursting with the Virtual Node





Azure Containers Offerings

AKS Admin Tool and Cluster Maintenance

Microsoft Services



Supporting Kubernetes Version in AKS

- AKS supports four minor versions of Kubernetes:
 - The current minor version that is released upstream (n)
 - Three previous minor versions. Each supported minor version also supports two stable patches.
- For example, if AKS introduces 1.11.x today, support is also provided for 1.10.a + 1.10.b, 1.9.c+ 1.9d, 1.8.e + 1.8f (where the lettered patch releases are two latest stable builds).
- If you upgrade an AKS cluster, Kubernetes minor versions cannot be skipped
- When you deploy an AKS cluster in the portal or with the Azure CLI, the cluster is always set to the n-1 minor version and latest patch

Nodes Update, Upgrade and Patches

- Regular maintenance, security and cleanup tasks

- Maintain, update and upgrade hosts and

Kubernetes - Monthly ideal, 3 months maximum

- Security patches

- AKS automatically applies security patches to the nodes on a nightly schedule
 - You're responsible to reboot as required, leveraging tools like Kured DaemonSet

<https://github.com/weaveworks/kured>

- Upgrade to new versions

```
$ az aks upgrade --name myAKSCluster \
--resource-group myResourceGroup \
--kubernetes-version 1.10.6
```

- Running benchmarks and tests to validate cluster setup

- Kube-bench
 - Kube-hunter
 - Others

Kube-Advisor

- Diagnostic tool for Kubernetes clusters. Today, it returns pods that are missing resource and request limits.
- More info can be found at <https://github.com/Azure/kube-advisor>

		CPU Request Limits Missing	Memory Request Limits Missing
		CPU Resource Limits Missing	Memory Resource Limits Missing
		CPU Request Limits Missing	Memory Request Limits Missing
ISSUE		REMEDIALION	
CPU Request Limits Missing		Consider setting resource and request limits to prevent resource starvation: https://kubernetes.io/docs/concepts/configuration/manage-compute-resources-container/	
Memory Request Limits Missing			
CPU Resource Limits Missing			
Memory Resource Limits Missing			

VS Code extension for warnings

- Kubernetes VS Code extension adding warnings for resource request/limits
- More info can be found at <https://github.com/Azure/vscode-kubernetes-tools>

```
35 ...     containers:
36 ...       - image: itowlson/biscuit2:latest
37 ...       - imagePullPolicy: Always
38 ...         name: biscuit2
39 ...         No CPU limit specified for this container - this could starve other processes
40 ...
41 ...           memory: 12345
```



Azure DevOps for Container Automation

Azure Containers Offerings

Microsoft Services



What is DevOps?

DevOps is the union of people, processes, and tools to enable continuous delivery of value to our customers.



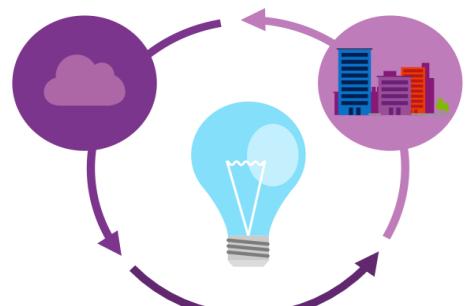
Increase flow of value

Shorten cycle times

Continuously Improve

What are the benefits?

Shorten cycle times
and deliver value faster



Improve quality and availability



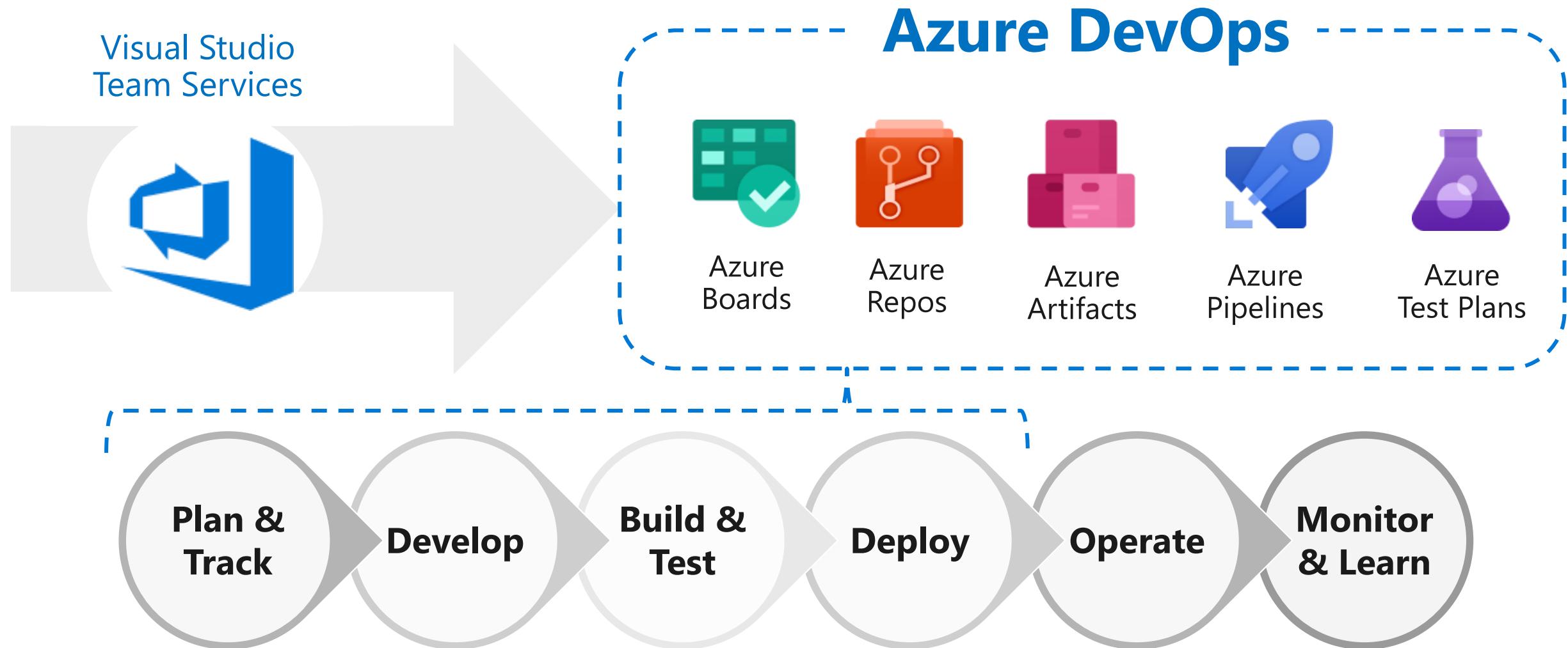
Optimize resources and eliminate waste



Deliver innovation with digital-era velocity



Azure DevOps... the evolution of Visual Studio Team Services



Azure Boards

Track work with Kanban boards, backlogs, team dashboards, and custom reporting



Connected from idea to release

Track all your ideas at every development stage and keep your team aligned with all code changes linked directly to work items.



Scrum ready

Use built-in scrum boards and planning tools to help your teams run sprints, stand-ups, and planning meetings.



Project insights

Gain new insights into the health and status of your project with powerful analytics tools and dashboard widgets.



<https://azure.com/devops>

The screenshot shows the Azure DevOps Boards interface. At the top, there's a navigation bar with the Azure DevOps logo, 'Contoso / AdventureWorks Mobile / Boards / FabrikamFiber'. Below the navigation, there are tabs for 'New' (selected), 'Active', '5/5 Staging', '15/5 Deployed', and 'Mobile (Spike)'. On the left, a sidebar menu includes 'AdventureWorks Mobile', 'Overview', 'Boards' (selected), 'Work Items', 'Backlogs', 'Sprints', 'Queries', 'Plans', 'Repos', 'Pipelines', 'Test Plans', and 'Artifacts'. The main area is titled 'FabrikamFiber Board' and contains a grid of cards representing work items. Each card has a title, a brief description, and a list of assignees. Some cards have status indicators like 'Design', 'Blocked', or 'General'. The cards are color-coded in shades of blue, pink, and light green. A footer at the bottom of the board view says 'Project settings' and has a back arrow icon.

Azure Repos

Unlimited private Git repo hosting and support for TFVC that scales from a hobby project to the world's largest Git repositories



Works with your Git client

Securely connect with and push code into your Git repos from any IDE, editor, or Git client.



Web hooks and API integration

Add validations and extensions from the marketplace or build your own using web hooks and REST APIs.



Semantic code search

Quickly find what you're looking for with code-aware search that understands classes and variables.



<https://azure.com/devops>

The screenshot shows the Azure DevOps interface for the 'AdventureWorks Mobile' project. The left sidebar includes links for Overview, Boards, Repos (selected), Files, Commits, Pushes, Branches, Tags, and Pull requests (also selected). The main content area is titled 'Pull requests' and shows a list of pull requests categorized by assignee: 'Mine', 'Active', 'Completed', and 'Abandoned'. Each pull request card displays the requester's name, the title of the PR, and the target branch. The interface includes standard Azure DevOps navigation elements like 'Contoso / AdventureWorks Mobile / Repos / Pull requests' at the top and a bottom navigation bar with 'Project settings' and a back arrow.

Created by	Title	Assignee	Comments	Actions
Kat Larsson	Initialize client with .client.init	Colin Ballinger	6	6
Kat Larsson	Testing configuration settings	Colin Ballinger	0	0
Colin Ballinger	Check returned identity for null status	Colin Ballinger	1	1
Robin Counts	[WIP] Add tests for deployment mapping	Robin Counts	3	3
Colin Ballinger	Add exception on disconnect	Colin Ballinger	0	0
Robin Counts	Maintain structure when converting isomorphs	Robin Counts	0	0
Robin Counts	Hotfix payload to releases/99	Robin Counts	99+	99+

Azure Artifacts

Create and share Maven, npm, and NuGet package feeds from public and private sources – fully integrated into CI/CD pipelines



Manage all package types

Get universal artifact management for Maven, npm, and NuGet.



Add packages to any pipeline

Share packages, and use built-in CI/CD, versioning, and testing.



Share code efficiently

Easily share code across small teams and large enterprises.



<https://azure.com/devops>

The screenshot shows the Azure DevOps interface for managing artifacts. On the left, there's a sidebar with links for Overview, Boards, Repos, Pipelines, Test Plans, and Artifacts. The Artifacts link is highlighted. The main area is titled "Artifacts" and shows a table of packages. The columns are Package, Views, Source, Last pushed, and Description. The packages listed are:

Package	Views	Source	Last pushed	Description
abbrev		nuget	a year ago	Like ruby's abbrev module, but in js
accepts		npmjs	a year ago	Higher-level content negotiation
acorn		MyFeed	a year ago	ECMAScript parser
acorn-dynamic-import		maven	a year ago	Support dynamic imports in acorn
aclr-jsx		nuget	a year ago	Alternative, faster React.js JSX parser
acorn-object-spread		maven	a year ago	Custom JSON-Schema keywords for ajv validator
ajv		npmjs	a year ago	Alphanumeric sorting algorithm
ajv-keywords		nuget	a year ago	ANSI escape codes for manipulating the terminal
alphanum-sort		npmjs	a year ago	An elegant lib that converts the chalked (ANSI) text to HTML

Azure Test Plans

Get end-to-end traceability. Run tests and log defects from your browser. Track and assess quality throughout your testing lifecycle.



Capture rich data

Capture rich scenario data as you execute tests to make discovered defects actionable. Explore user stories without test cases or test steps. You can create test cases directly from your exploratory test sessions.



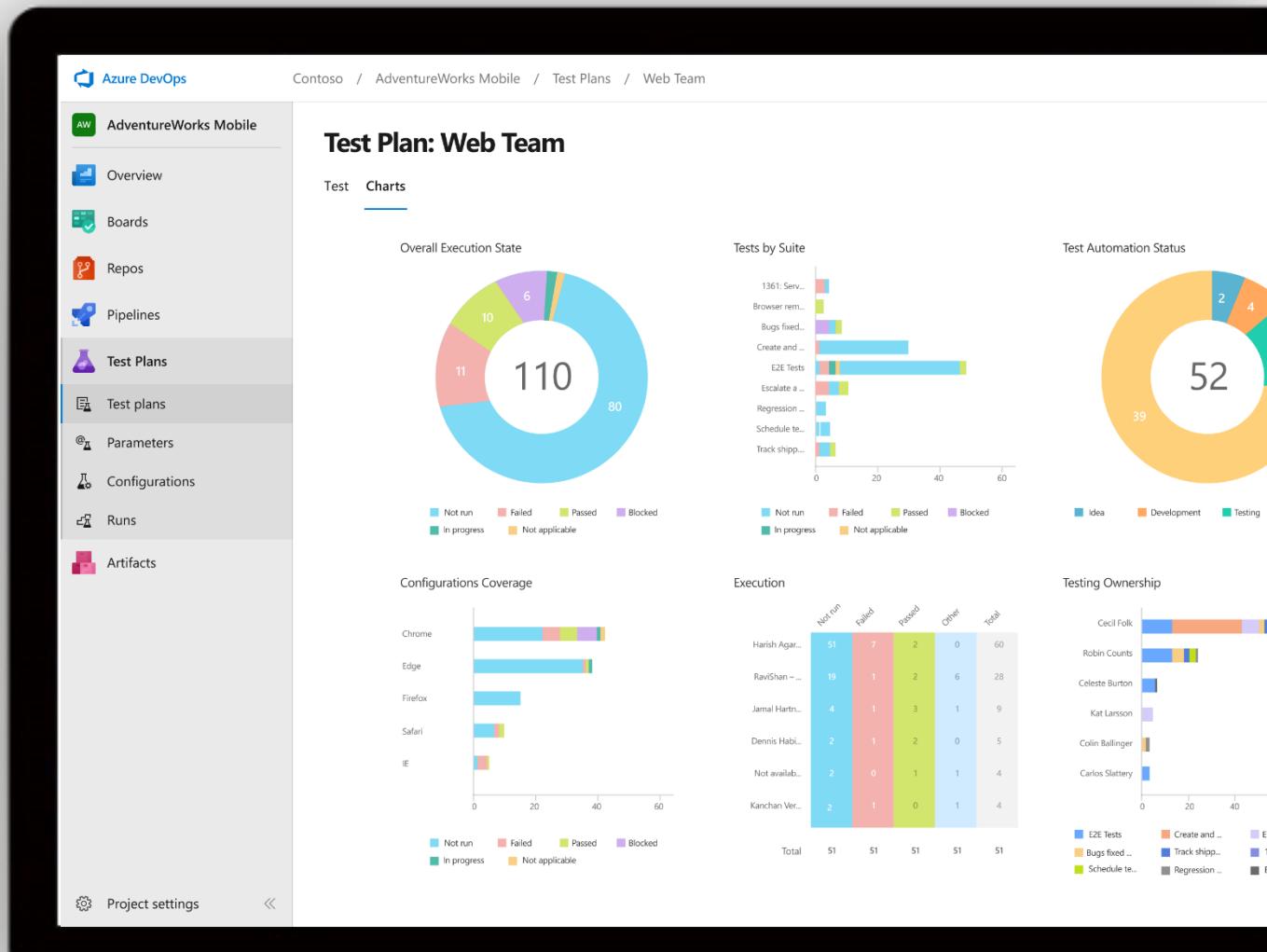
Test across web and desktop

Test your application where it lives. Complete scripted tests across desktop or web scenarios. Test on-premises application from the cloud and vice-versa.



Get end-to-end traceability

Leverage the same test tools across your engineers and user acceptance testing stakeholders. Pay for the tools only when you need them.



<https://azure.com/devops>

Azure Pipelines

Cloud-hosted pipelines for Linux, Windows and macOS, with unlimited minutes for open source



Any language, any platform, any cloud

Build, test, and deploy Node.js, Python, Java, PHP, Ruby, C/C++, .NET, Android, and iOS apps. Run in parallel on Linux, macOS, and Windows. Deploy to Azure, AWS, GCP or on-premises



Extensible

Explore and implement a wide range of community-built build, test, and deployment tasks, along with hundreds of extensions from Slack to SonarCloud. Support for YAML, reporting and more



Containers and Kubernetes

Easily build and push images to container registries like Docker Hub and Azure Container Registry. Deploy containers to individual hosts or Kubernetes.



Best-in-class for open source

Ensure fast continuous integration/continuous delivery (CI/CD) pipelines for every open source project. Get unlimited build minutes for all open source projects with up to 10 free parallel jobs across Linux, macOS and Windows

Enabling feature flags for Preview Attachment and Grid Views
AdventureWorks/PackageFramework master #889

Windows Job
Running 1m 53s

Linux Job
Running 3m 29s

macOS Job
Running 3m 07s

Linux Job
Agent: Hosted Linux

- ✓ Prepare job
- ✓ Initialize job
- ✓ Get sources
- ✓ Cmdline
- ✓ Nodetool
- ⊕ Install dependencies

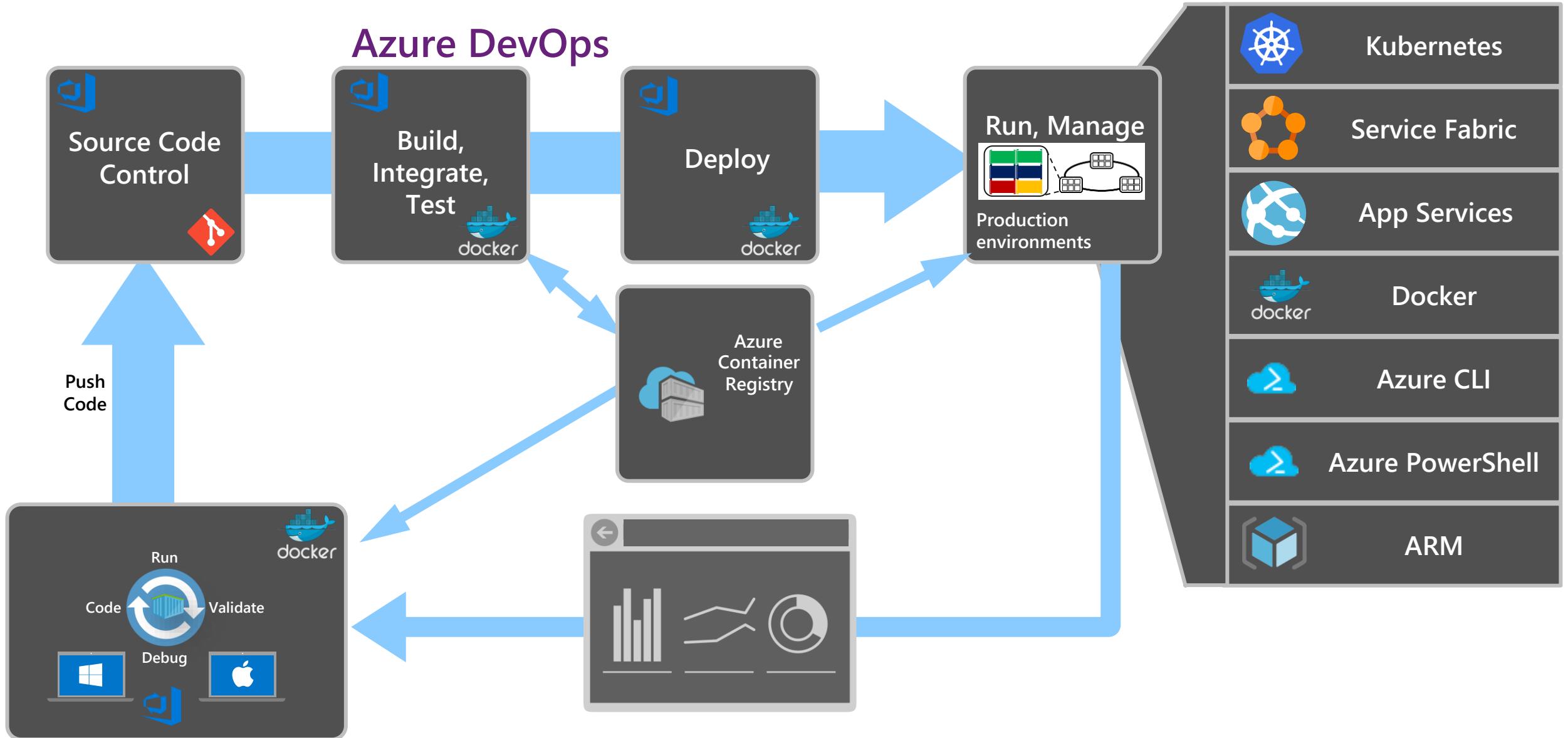
```
yarn install v1.7.0
$ node build/npm/preinstall.js
[1/4] ⚡ Resolving packages...
[2/4] 🛡 Fetching packages...
[3/4] 🔗 Linking dependencies...
[4/4] 🏭 Building fresh packages...
$ npm run compile
#####
> code-oss-dev-build@1.0.0 compile ./adventureworks/build
> tsc -p tsconfig.build.json

⚡ Done in 4.89s.
$ node ./postinstall
[##] 2/2 removed './adventureworks/extensions/node_modules/typescript/lib/tsc.js'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverlibrary.d.ts'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverlibrary.js'
removed './adventureworks/extensions/node_modules/typescript/lib/typescriptServices.d.ts'
removed './adventureworks/extensions/node_modules/typescript/lib/typescriptServices.js'
```



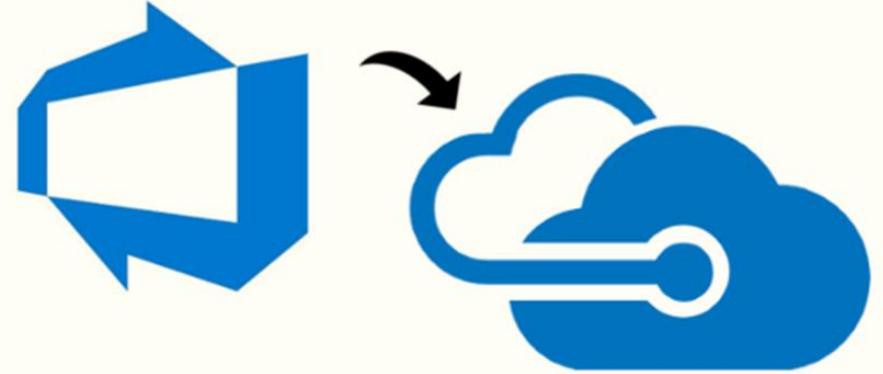
<https://azure.com/pipelines>

Containerized Workflow Pipeline

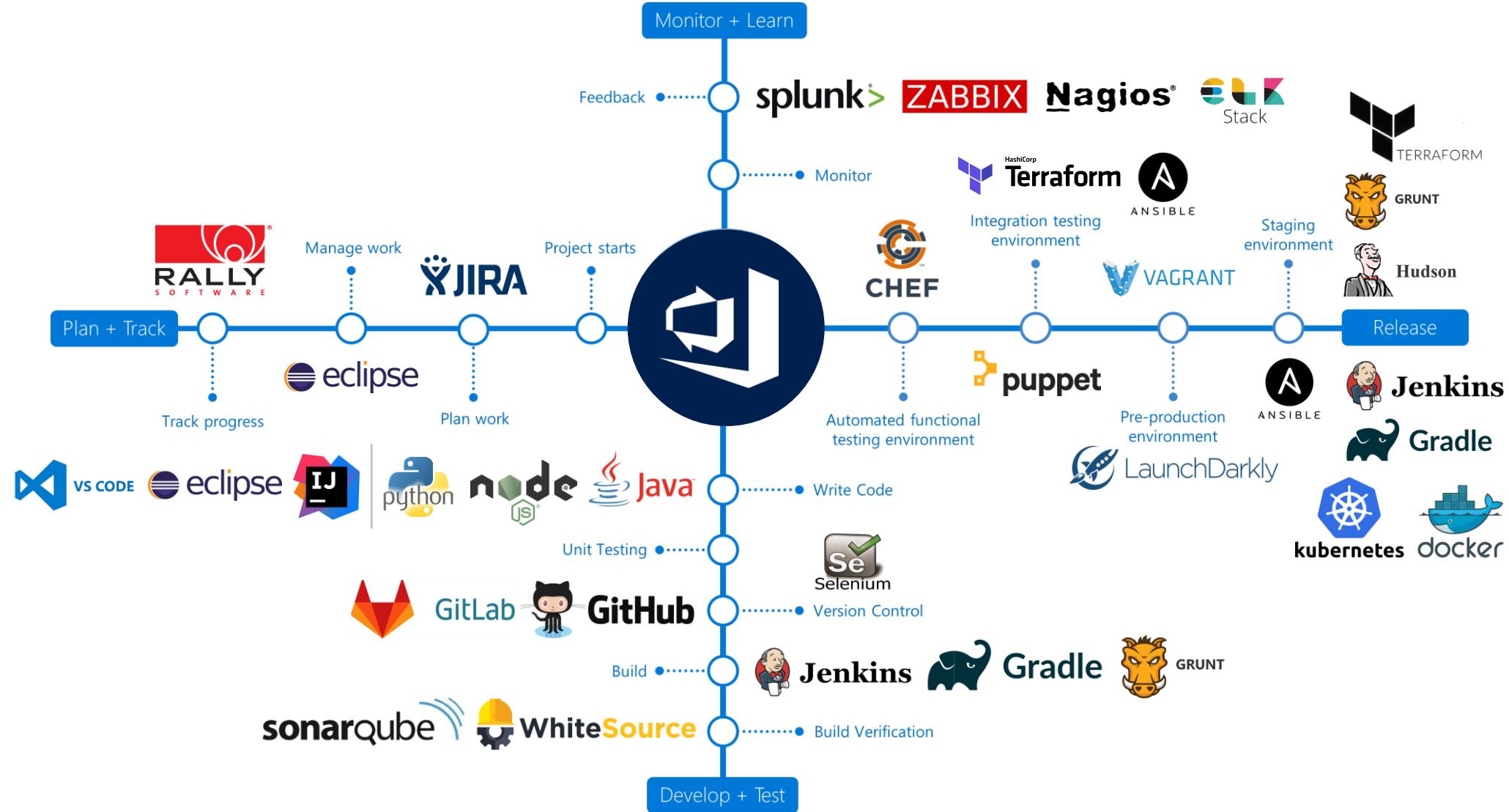


Azure DevOps

- Azure Integration
 - Formerly VSTS – Azure Native
 - Service Endpoints
 - Azure DevOps extensions
- DevOps Projects
 - Present a simplified experience where you can bring your existing code and Git repository or choose from one of the sample applications to create a continuous integration (CI) and continuous delivery (CD) pipeline to Azure.



Azure DevOps + Partners



Lab: AKS Advanced

Upgrading your AKS Cluster

Working with NGINX and Azure Files

Running your app with services
exposes via ELB

Working with Network Policies



Session resources

- DevOps information @ <https://aka.ms/devops>
- Azure DevOps overview @ <https://aka.ms/azuredevops>
- Azure DevOps documentation @ <https://aka.ms/vstsdocs>
- Azure DevOps Projects @ <https://aka.ms/devopsprojects>
- AKS + Jenkins @ [AKS+Jenkins](#)
- AKS + Azure DevOps @ [AKS+DevOps](#)
- AKS + ASP.Net Core + DevOps Projects @ [AKS+DevOpsProjects](#)

