Product deep dive

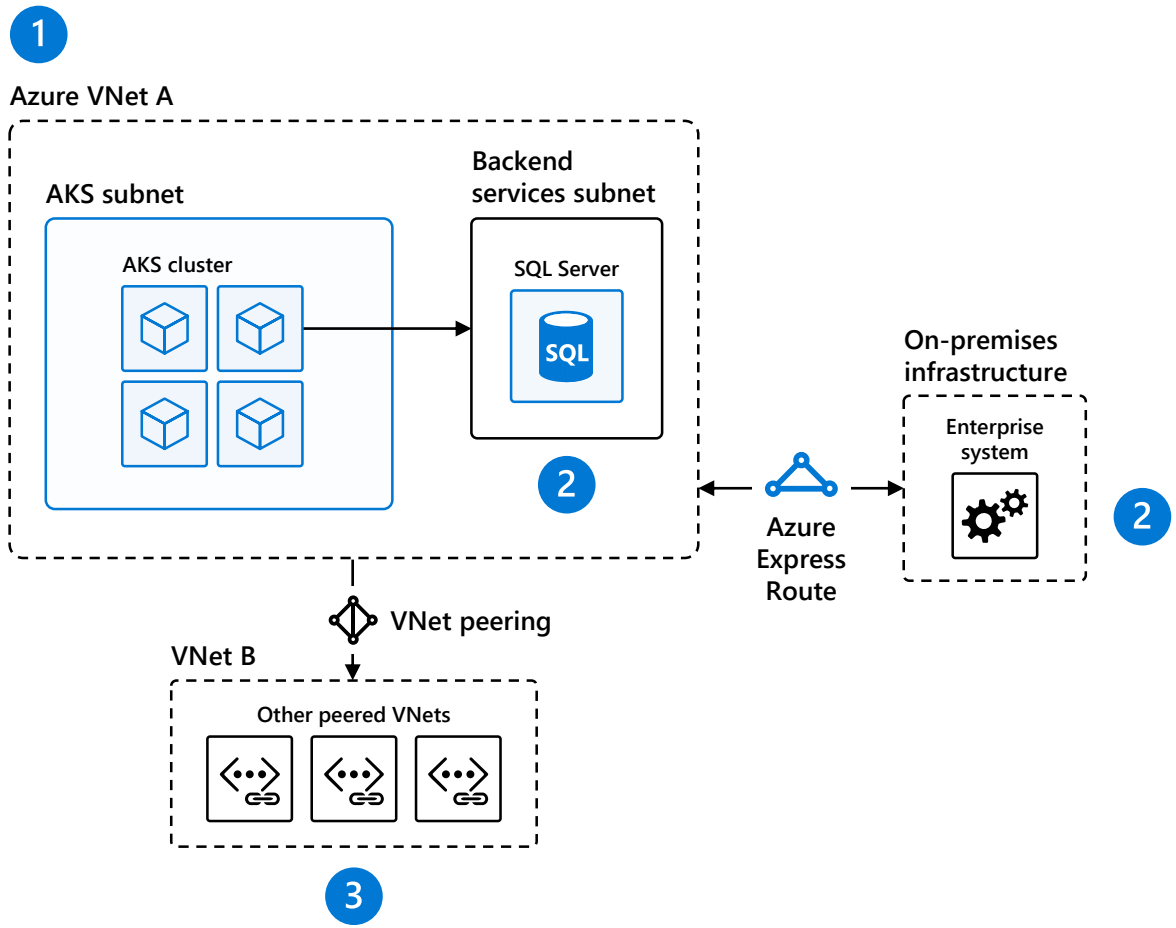
Secure network communications with VNET and CNI

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 VNET peering to other VNET

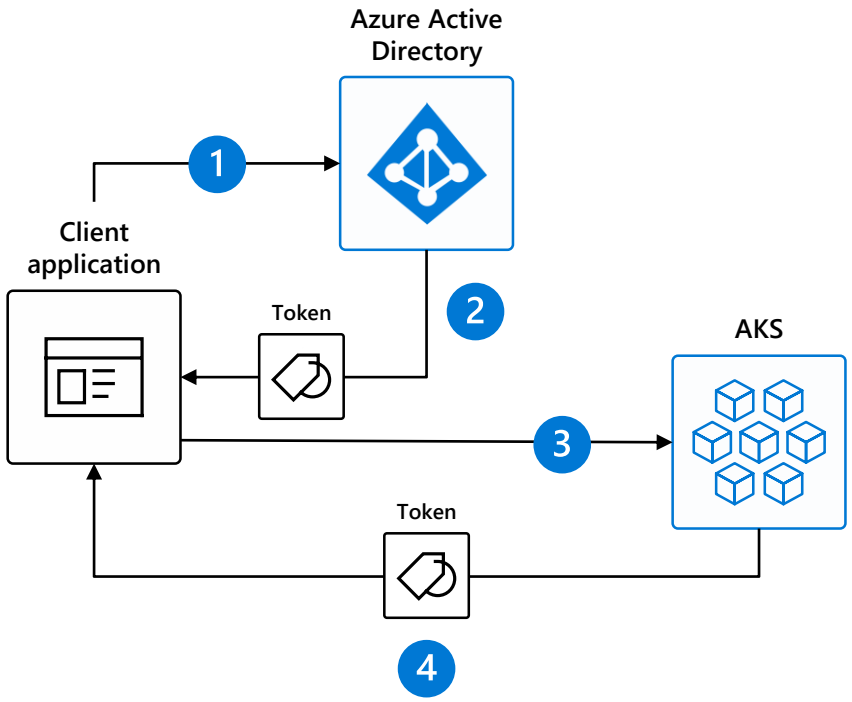
AKS VNet integration works seamlessly with your existing network infrastructure



Identity and access management through AAD and RBAC

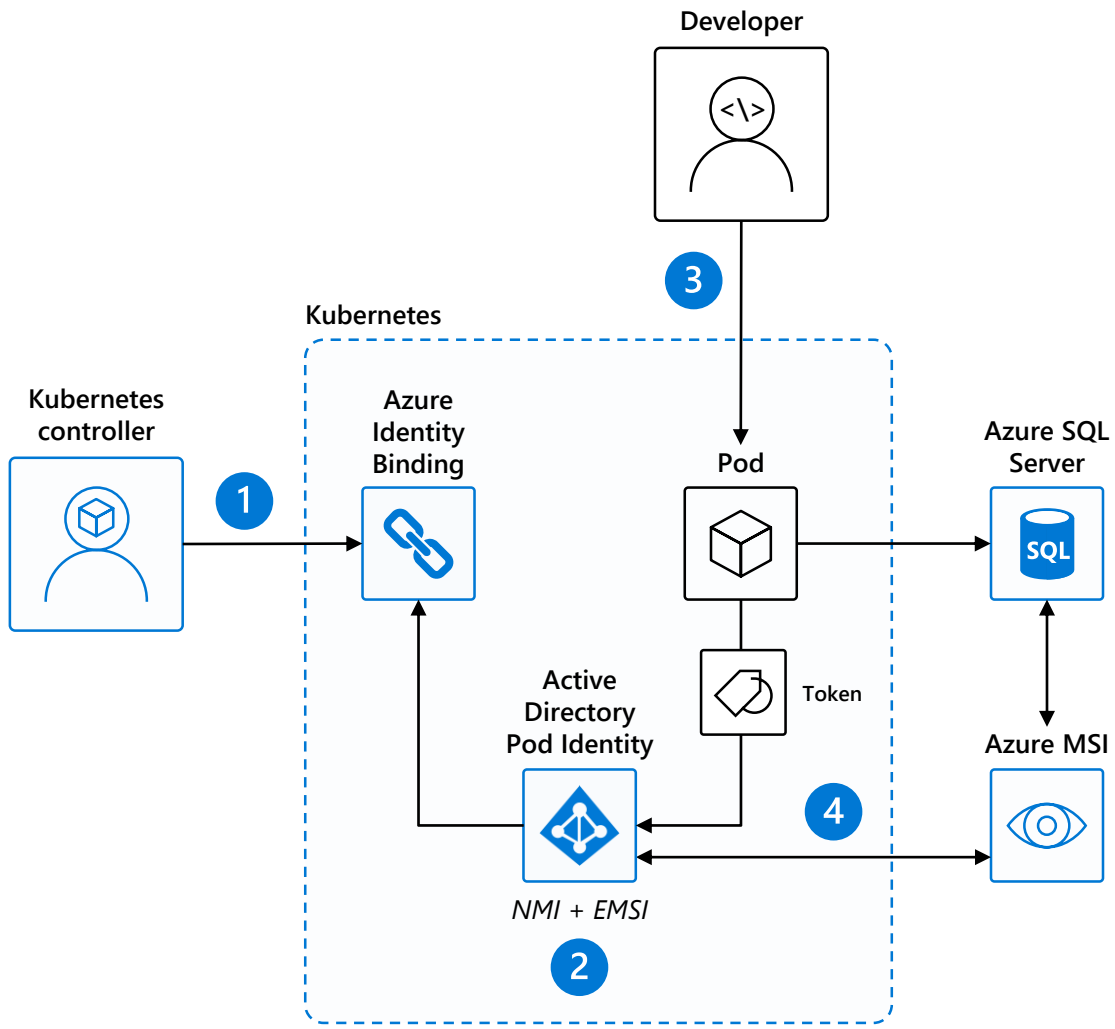
1. The client application authenticates to the AAD token issuance endpoint and requests an access token
2. The AAD token issuance endpoint issues the access token
3. The access token is used to authenticate to the secured resource
4. Data from the secured resource is returned to the web application

Azure delivers a streamlined identity and access management solution with Azure Active Directory (AAD) and Azure Kubernetes Services (AKS)



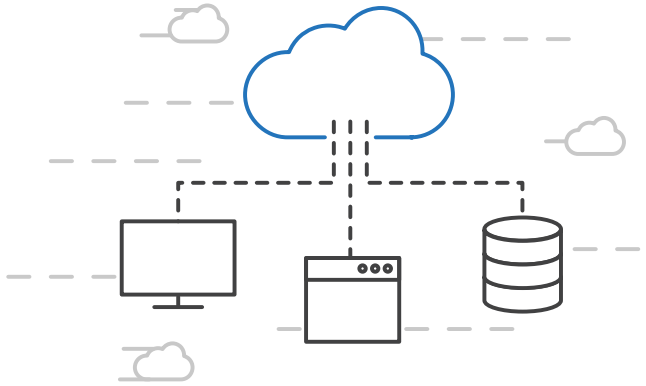
Pod identity

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

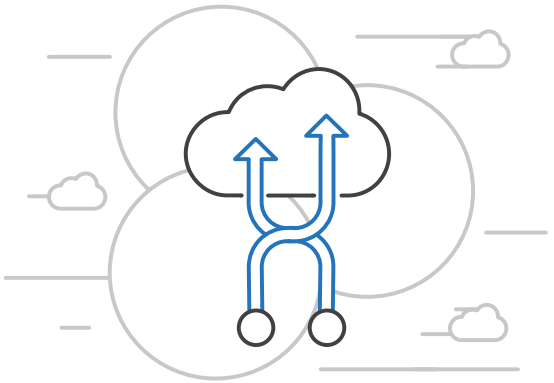


Open Service Broker for Azure (OSBA)

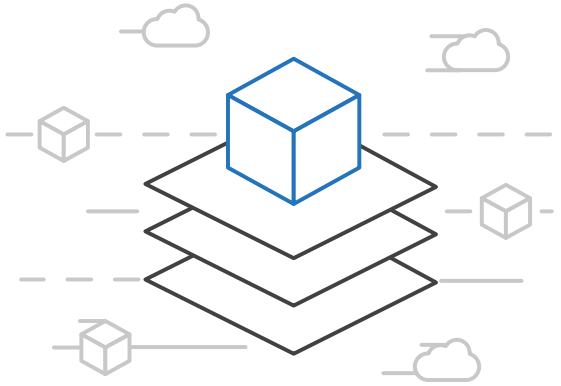
Connecting containers to Azure services and platforms



A standardized way to connect with Azure services



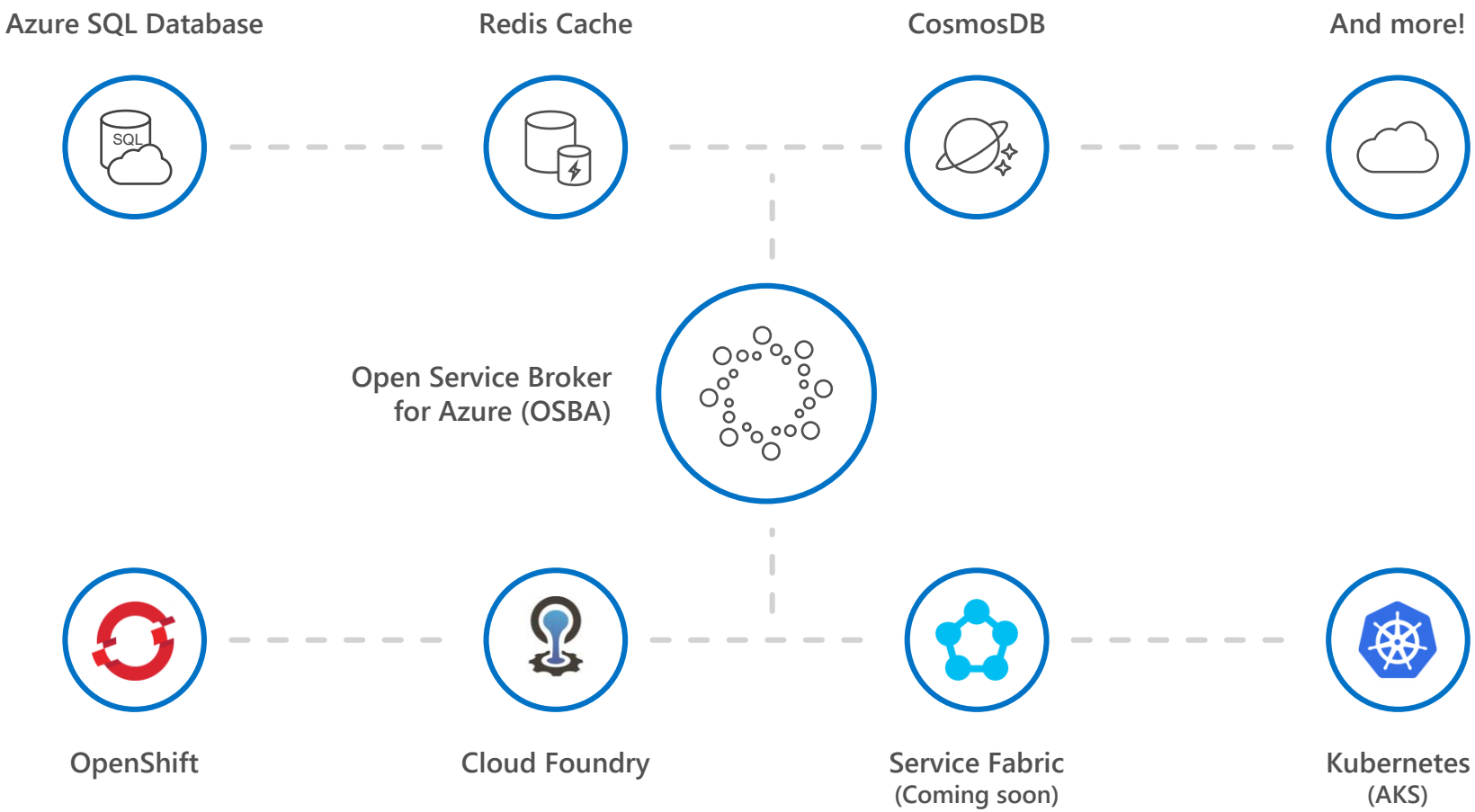
Simple and flexible service integration



Compatible across numerous platforms

Open Service Broker for Azure (OSBA)

An implementation of the Open Service Broker API



Open Service Broker for Azure (OSBA)

mysql-instance.yaml

```yaml
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ServiceInstance
metadata:
 name: mysql-instance
 namespace: default
spec:
 clusterServiceClassExternalName: azure-mysql-5-7
 clusterServicePlanExternalName: basic
 parameters:
 location: westeurope
 resourceGroup: testcontainer
```

mysql-binding.yaml

```yaml
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ServiceBinding
metadata:
 name: mysql-binding
 namespace: default
spec:
 instanceRef:
  name: mysql-instance
 secretName: mysql-secret
```

```
kubectl apply -f mysql-instance.yaml

kubectl apply -f mysql-binding.yaml
```