# AKS Security

Contents

[AKS Security 1](#_Toc120450396)

[Overview 1](#_Toc120450397)

[Best Practices for Authentication and Authorization in AKS 1](#_Toc120450398)

[Build Security 1](#_Toc120450399)

[Registry Security 1](#_Toc120450400)

[Best Practices for Container Image Management and Security in AKS 1](#_Toc120450401)

[Cluster Security 1](#_Toc120450402)

[Pod Management Security 1](#_Toc120450403)

[Key Vault with Secret Store 1](#_Toc120450404)

[Node Security 1](#_Toc120450405)

[Node Security Patches (Windows Server nodes) 1](#_Toc120450406)

[Node Storage 1](#_Toc120450407)

[Cluster Upgrades 1](#_Toc120450408)

[Upgrade process 1](#_Toc120450409)

[Network Security 1](#_Toc120450410)

[Web Application Firewall 1](#_Toc120450411)

[Kubernetes Network Policy 1](#_Toc120450412)

[Application Security 1](#_Toc120450413)

[Kubernetes Secrets 1](#_Toc120450414)

[Security Hardening for AKS Linux agent node host OS 1](#_Toc120450415)

[Azure Security Benchmark 1](#_Toc120450416)

[Environment Security 1](#_Toc120450417)

[ Registry Security 1](#_Toc120450418)

## Overview

**Container security** acts as a protector for the **entire end-to-end pipeline from build to the running applications** in AKS.

Multiple Azure components (AD, MS Defender, AZ Policy, AZ Key Vault) are combined throughout the process to provide following security:

* complete **authentication** and **authorization**
* AKS **Built-in Azure Policy** that secures applications
* E2E insight from build through application with **MS Defender for Containers**
* AKS cluster has the **latest OS security updates** and **Kubernetes releases**
* Secure **pod traffic** and **secured access to sensitive credentials**

## Best Practices for Authentication and Authorization in AKS

1. Azure Active Directory (AD) - <https://docs.microsoft.com/en-us/azure/aks/managed-aad>
2. Kubernetes RBAC - <https://docs.microsoft.com/en-us/azure/aks/azure-ad-rbac>
3. Pod-managed Identities

## Build Security

It is important to conduct **static analysis of image builds** before promoting them down the pipeline.

Static analysis includes **vulnerability** and **compliance assessment.**

Try to **use as fewer base images as possible.**

**Don’t add unnecessary components like Curl** who can pose security threat.

Reference: <https://thechief.io/c/editorial/7-static-analysis-tools-to-secure-and-build-stable-kubernetes-clusters/>

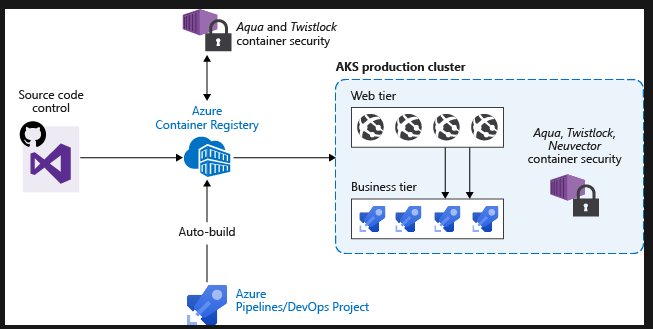
## Registry Security

**Assess vulnerability state of the image in the Registry** – use Sigstore **Cosign or** **Notary V2** to attach signature to your images to ensure that those are images from a trusted location.

Reference: <https://github.com/notaryproject/notation>

## Best Practices for Container Image Management and Security in AKS

#### How to secure images

1. Scan container images for vulnerabilities
2. Only deploy validated images
3. Regularly update base images and application runtime
4. Redeploy workloads in AKS cluster  
     
   

Above is an example of **CI/CD pipeline to automate image scans, verification and deployments**. ACR includes these vulnerabilities scanning capabilities.

#### Build new images on base image update

Use **automation to build new images** when the base image is updated.

Usually updated base images contain security fixes, so it’s needed to update downstream application container images.

Tutorial for automating container image builds when base image is updated in ACR: <https://docs.microsoft.com/en-us/azure/container-registry/container-registry-tutorial-base-image-update>

## Cluster Security

Each AKS cluster has its own dedicated Kubernetes master to provide the API Server, Scheduler etc.

By default, **API Server uses a public IP address**, but access to the API Server can be limited with the **Kubernetes role-based access control (RBAC)** or **Azure RBAC**.

Reference: <https://docs.microsoft.com/en-us/azure/aks/managed-aad>

To limit container actions, **AppArmor** is used enabled by default for Linux applications.

Reference: <https://kubernetes.io/docs/tutorials/clusters/apparmor/>

## Pod Management Security

A **managed identity for Azure resources** lets a pod authenticate itself against Azure services like Storage or SQL.

The pod is assigned an Azure Identitiy that lets them authenticate to AZ AD and receive a digital token that is presented to other Azure services.

No secrets are required for connections with this approach.

Reference: <https://docs.microsoft.com/en-us/azure/aks/developer-best-practices-pod-security>

## Key Vault with Secret Store

For services without managed identities for Azure resources, it is possible to authenticate with credentials stored in the digital vault.

Azure Key Vault can be integrated with AKS cluster with the **Azure Key Vault provider for the Secrets Store CSI Driver**.

Reference: <https://github.com/Azure/secrets-store-csi-driver-provider-azure#usage>

The SS CSI driver enables AKS cluster to retrieve secret contents securely and provide them only to the requesting pod.

## Node Security

When an AKS cluster is created or scaled up, its nodes are **automatically deployed with the latest OS security updates and configurations.**

### Node Security Patches (Windows Server nodes)

Windows Update **doesn’t automatically run and apply the latest update**: you need to schedule upgrades **manually**.

Schedule Windows Server node pool upgrades in AKS cluster around regular Windows Update release cycle.

This will **create nodes that run the latest Windows Server image and patches** and will **remove the older nodes**.

Reference: <https://docs.microsoft.com/en-us/azure/aks/use-multiple-node-pools#upgrade-a-node-pool>

### Node Storage

Azure Managed Disks, which are Premium disks backed by SSD, are used for storage purposes.

Data stored is **automatically encrypted** and to improve redundancy, Azure Managed Disks are **securely replicated within the Azure datacenter**.

## Cluster Upgrades

Azure provides orchestration tools to upgrade AKS cluster and components, maintain security and compliance and access the latest features.

To start the upgrade process, a **Kubernetes version must be provided** and Azure then safely does the **upgrade for each AKS node**.

Reference: <https://docs.microsoft.com/en-us/azure/aks/supported-kubernetes-versions?tabs=azure-cli>

### Upgrade process

1. New node with latest OS image and patches is deployed into the node pool
2. One of existing nodes is identified for upgrade
3. Pods on identified node are terminated and scheduled on the other nodes
4. Empty node is deleted from AKS cluster
5. Steps 1-4 are repeated until all nodes are replaced

## Network Security

To filter virtual network traffic flow, Azure uses **network security group rules.**

When services with load balancers, port mappings or ingress routes are created, AKS automatically modifies the network security group for traffic flow.

**If own subnet for AKS cluster is provided, do not modify the NIC-level network security group managed by AKS.**

Instead, create **more subnet-level network security groups** to modify the flow of traffic, but they **must not interfere with necessary traffic managing the cluster, communication with the control plane and egress**.

## Web Application Firewall

Azure Web Application Firewall (WAF) on Azure Application Gateway provides centralized protection of your web applications from common exploits and vulnerabilities.

<https://learn.microsoft.com/en-us/azure/web-application-firewall/ag/ag-overview>

<https://learn.microsoft.com/en-us/security/benchmark/azure/baselines/application-gateway-security-baseline>

## Kubernetes Network Policy

To limit network traffic between pods in cluster, **Kubernetes network policies** can be used.

Reference: <https://docs.microsoft.com/en-us/azure/aks/use-network-policies>

## Application Security

To protect pods running on AKS, use **Microsoft Defender for Kubernetes** to detect and restrict cyber attacks against your application.

**Continual scanning** needs to be run to **detect drift in the vulnerability state** and implement a process to patch and replace vulnerable images.

Reference: <https://docs.microsoft.com/en-us/azure/defender-for-cloud/container-security>

## Kubernetes Secrets

With **Kubernetes Secrets**, sensitive data (credentials, keys…) is injected into pods with the following procedure:

1. Create a Secret with Kubernetes API
2. Define pod or deployment and request a specific Secret that will be stored in tmpfs, not on disk
3. When last pod on a node is deleted requiring a Secret, the Secret is deleted from tmpfs.

This approach **reduces the sensitive information defined in the pod or service YAML** manifest.

## Security Hardening for AKS Linux agent node host OS

AKS provides a security-optimized host OS by default, but **no option to select an alternate operating system**.

**Daily patches are applied** (some require reboot, some not).

Scheduling of AKS VM host reboots are **manual** (we are responsible for it).

To reduce the attack surface area, some unnecessary kernel module drives are disabled.

Reference: <https://docs.microsoft.com/en-us/azure/aks/security-hardened-vm-host-image>

### Azure Security Benchmark

It provides recommendations on how to secure cloud solutions on Azure. List below contains policies that needed to be adopted to be compliant with certain standards.

#### **Azure Policies**

1. Authorized IP ranges should be defined on Kubernetes Services
2. RBAC should be used on Kubernetes Services
3. Kubernetes clusters should be accessible only over HTTPS
4. AKS clusters should have Azure Defender profile enabled
5. Azure Policy Add-on for AKS should be installed and enabled on clusters
6. K8S cluster containers CPU and memory limits shouldn’t exceed specified limits
7. K8S cluster containers should not share host process ID or host IPC namespace
8. K8S cluster container should only listen on allowed ports
9. K8S cluster containers should only use allowed AppArmor profiles
10. K8S cluster containers should only use allowed capabilities
11. K8S cluster containers should only use allowed images
12. K8S cluster containers should run with a read only root file system
13. K8S cluster pod hostPath volumes should only use allowed host paths
14. K8S cluster pods and containers should only run with approved user and group IDs
15. K8S cluster pods should only use approved host network and port range
16. K8S cluster should not allow privileged containers
17. K8S cluster should disable automounting API credentials
18. K8S clusters should not grant CAP\_SYS\_ADMIN security capabiliteis
19. K8S clusters should not use the default namespace
20. K8S services should be upgraded to a non-vulnerable K8S version
21. Both OS and data disks in AKS clusters should be encrypted by customer-managed keys
22. Temp disks and cache for agent node pools in AKS clusters should be encrypted at host

Reference: <https://docs.microsoft.com/en-us/azure/aks/security-controls-policy>