# AKS Checklist

## Intent

As a Cloud Solution Architect, I work with many partners who want to migrate their application to Kubernetes and want a checklist of considerations.

The items listed here were inspired by my own findings as well as [The AKS Checklist](https://www.the-aks-checklist.com/)

## Cluster

* Use [AKS Managed AAD Integration](https://docs.microsoft.com/en-us/azure/aks/managed-aad)
* Limit cluster access via [K8S RBAC for users & workloads](https://docs.microsoft.com/en-us/azure/aks/azure-ad-rbac)
* Use [Managed Identity in AKS](https://docs.microsoft.com/en-us/azure/aks/use-managed-identity)
* Use [system node pools](https://docs.microsoft.com/en-us/azure/aks/use-system-pools)
* Use [kured](https://docs.microsoft.com/en-us/azure/aks/node-updates-kured) for security updates that need reboot
* Set [auto-upgrade channel](https://docs.microsoft.com/en-us/azure/aks/upgrade-cluster#set-auto-upgrade-channel)
* Prefer fewer, larger clusters over of many, smaller clusters
* Use [AKS API Server IP Whitelisting](https://docs.microsoft.com/en-us/azure/aks/api-server-authorized-ip-ranges)
* Use [AKS Private Cluster](https://docs.microsoft.com/en-us/azure/aks/private-clusters)
* Use [Managed Identity](https://docs.microsoft.com/en-us/azure/aks/use-managed-identity)
* Use [Proximity Placement Groups](https://docs.microsoft.com/en-us/azure/aks/reduce-latency-ppg)

## Container Image / Registry

* Use [ACR](https://docs.microsoft.com/en-us/azure/aks/cluster-container-registry-integration)
* Implement [Image Scanning](https://docs.microsoft.com/en-us/azure/security-center/defender-for-container-registries-introduction)
* Use only trusted images (e.g. only use images you created via CI/CD)
* Use [distroless images](https://github.com/GoogleContainerTools/distroless)
* Scan all container images against vulnerabilities
* Only use images from trusted container registries
  + Even better, only use a single ACR instance
* Use [least privilege RBAC](https://docs.microsoft.com/en-us/azure/container-registry/container-registry-roles) for your ACR
* Use [Private Endpoint / Private Link](https://docs.microsoft.com/en-us/azure/container-registry/container-registry-private-link)

## Security

Consider the following:

* Are you subject to security audits?
* Do you have any compliance requirements?

Checklist:

* Read [The Definitive Guide to Securing Kubernetes](https://info.aquasec.com/securing_kubernetes) whitepaper
* Use a K8S security Product (e.g. Sysdig, AquaSec, Twistlock, etc.)
* Enable [Azure Defender for Kubernetes](https://docs.microsoft.com/en-us/azure/security-center/defender-for-kubernetes-introduction)
* Do not run containers as root or privileged
  + Example: allowPrivilegeEscalation: false
* Use [App Armor](https://docs.microsoft.com/en-us/azure/aks/operator-best-practices-cluster-security#app-armor) and [seccomp](https://docs.microsoft.com/en-us/azure/aks/operator-best-practices-cluster-security#secure-computing)

## Secrets

NOTE: By default, K8S Secrets are only encrypted at rest

* Understand [Secret Management in K8S](https://www.youtube.com/watch?v=KmhM33j5WYk&list=PLLasX02E8BPCrIhFrc_ZiINhbRkYMKdPT&index=11)
* Use [Azure Key Vault](https://docs.microsoft.com/en-us/azure/key-vault/general/key-vault-integrate-kubernetes) to store Secrets/Certificates
* Don’t inject secrets as environment vars (visible via kubectl describe)
* Don’t store secrets in container images

## Compute

* Set resource quotas on namespaces
* If necessary, [harden AKS agent nodes](https://clouddamcdnprodep.azureedge.net/gdc/gdc8LXmoZ/original)

## Availability

* Use [Uptime SLA](https://docs.microsoft.com/en-us/azure/aks/uptime-sla) for production clusters
  + NOTE: By Default K8S API does not have SLA. Only premium tier
* Configure readiness, liveness and readiness probes
* Use Deployments and run at least 1 replica
* Configure logging, app monitoring an alerting
* Set Pod Disruption Budgets
* Decide if [Availability Zones](https://docs.microsoft.com/en-us/azure/aks/availability-zones) or [multiple regions](https://docs.microsoft.com/en-us/azure/aks/operator-best-practices-multi-region) meet your needs

## Scaling

* Use Pod Requests and Limits
* Use HPA
* Use [Cluster Autoscaling](https://docs.microsoft.com/en-us/azure/aks/cluster-autoscaler)

## Network

* Clearly understand your North-South and East-West network requirements
* Use a [Network Policy](https://docs.microsoft.com/en-us/azure/aks/use-network-policies) (Calico or Azure Network Policy)
  + Example: [Network Policy Recipes](https://github.com/ahmetb/kubernetes-network-policy-recipes)
* Use a [WAF](https://azure.microsoft.com/en-us/services/web-application-firewall/)
  + Examples: [AppGW](https://docs.microsoft.com/en-us/azure/web-application-firewall/ag/ag-overview), [Front Door](https://docs.microsoft.com/en-us/azure/web-application-firewall/afds/afds-overview), [Nginx+](https://docs.nginx.com/nginx-waf/)
* Use [Traffic Manager](https://azure.microsoft.com/en-us/services/traffic-manager/) or [Front Door](https://azure.microsoft.com/en-us/services/frontdoor/) to front traffic
* Only use [Internal Load Balancer](https://docs.microsoft.com/en-us/azure/aks/internal-lb)
* Use [Azure Firewall](https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/containers/aks/secure-baseline-aks#egress-traffic-flow) to inspect egress traffic for data leakage.

## Governance

* Use Azure Policy for Kubernetes
* Example: https://docs.microsoft.com/en-us/azure/aks/policy-reference

## Service Mesh

* Don’t use a service mesh unless you have a specific need

## Business Continuity

## Multi-Region

* NOTE: AKS cluster is region-based.
* Use Paired Regions
* Enable geo-redundancy on your ACR
* If app includes volumes, use a 3rd party solution (Portworx, Kasten)

## Deployment

* Use GitOps
* Don’t use the default namespace

## Multi-Tenancy

* NOTE: Some tenants can be in shared env; Some tenants might be big enough to require separate cluster
* NOTE: Namespace does NOT provide any isolation, but is a common practice for grouping tenants
* Must be coupled with strong Network and Compute isolation (see above)
* Use Helm or CRD to package new tenants

## Tooling

* Helm
* Flux (GitOps)
* Kubectl aliases
* kubectx

References:

* [AKS Cluster Security Best Practices](https://docs.microsoft.com/en-us/azure/aks/operator-best-practices-cluster-security)
* [Cloud Adoption Framework - Kubernetes](https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/innovate/kubernetes/)
* [Azure Defender for Kubernetes](https://docs.microsoft.com/en-us/azure/security-center/defender-for-kubernetes-introduction)
* [We use the term ‘hostile multitenancy’ in Microsoft to describe hosting platforms that must provide strict resource governance and security isolation between different customers. ‘Enterprise multitenancy’ assumes internal customers with good intent and reactive controls – Mark Russinovich](https://twitter.com/markrussinovich/status/1363988454485368832)