How to reduce the TCO of your AKS cluster

Maheshkumar R
Cloud Solution Architect
12/4/2023
maheshk@microsoft.com

Ways to minimize the no. of unused cores; ways to densify your workloads; ways to reduce the no. of VMs to the bare minimum

(1) Use cluster autoscaler, Kubernetes Event-Driver Autoscaler (KEDA), and Horizontal Pod Autoscaler to scale in and scale out the no. of pods and the no. of nodes based on the traffic conditions.

(2) Set requests and limits for pods to avoid assigning too many resources in terms of CPU and memory to the user-defined workloads and improve application density. You can observe the average and maximum consumption of CPU and memory using Prometheus or Container Insights and properly configure limits and quotas for your pods in the YAML manifests, Helm charts, Kustomize manifests for your deployments. For more information, see Best practices for application developers to manage resources in AKS. There are 3rd party tools like Densify that, by gathering granular container data from frameworks like Prometheus, learning the patterns of activity, and applying policies, can suggest requests and limits for each pod container, optimizing the overall density.

(3) Use ResourceQuota objects to set quotas for the total amount of memory and CPU that can be used by all Pods running in a given namespace - improve the application density,and reduce the number of agent nodes. Use LimitRange objects to configure the default requests in terms of CPU and memory for pods running in a namespace.

Azure Policy integrates with AKS through built-in policies to apply at-scale enforcements and safeguards on your cluster in a centralized, consistent manner. Follow Azure Policy add-on on your cluster and apply the Ensure CPU and memory resource limits policy, ensuring CPU and memory resource limits are defined on containers in an AKS cluster.

(4) Use Vertical Pod Autoscaler (VPA), based on the open-source Kubernetes version, to analyze and set CPU and memory resources required by your pods.

Instead of running tests to calculate the optimal CPU and memory requests and limits for the containers in your pods, you can configure vertical Pod autoscaling to provide recommended values for CPU and memory requests and limits that you can use to update your pods manually, or you can configure vertical Pod autoscaling to update the values automatically. When configured, the (VPA) automatically sets resource requests and limits on containers per workload based on past usage.

K8s Monitor Pod CPU and memory usage with Prometheus
K8s Vertical Pod Autoscaling
Practical Guide to Kubernetes Scaling #1 Pods
Practical Guide to Kubernetes Scaling #2 Nodes

(5) Choose the right VM size for the node pools of your AKS cluster based on the needs in terms of CPU and memory of your workloads.

(6) Create multiple node pools with diff VM sizes for particular purposes and workloads and use Kubernetes node labels, node selector, pod and node affinity and anti-affinity, taints and tolerations, and pod topology spread constraints to place applications on specific node pools to avoid noisy neighbor issues and improve the pod density. Keep node resources available for workloads that require them, and don't allow other workloads to be scheduled on these nodes. Using different VM sizes for different node pools can also be used to optimize costs. For more information, see Use multiple node pools in AKS - Microsoft Doc...

(7) The higher the VM SKU, the higher the number of vCores, and the higher the chance of unused cores. Assuming that the Kubernetes Scheduler and cluster autoscaler do a good job consolidating pods in a set of nodes, there will always be a fraction of unused vCores.

(8) The higher the number of node pools, the higher the chance of unused vCores, as each node pool scales separately.

Recommendations to reduce the TCO of an AKS cluster

(1) Review the Cost optimization section of the Azure Well-Architected Framework for AKS.

(2) Use Azure Advisor to monitor and release unused resources. Find and release any resource not used by your AKS cluster, such as public IPs, managed disks, etc. For more information, see Find and delete unattached Azure managed and unmanaged disks.

(3) Use Microsoft Cost Management budgets and reviews to keep track of expenditures.

(4) Use Azure Reservations to reduce the cost of the agent nodes. Azure Reservations help you save money by committing to one-year or three-year plans for multiple products. Committing allows you to get a discount on the resources you use.

Reservations can significantly reduce your resource costs by up to 72% from pay-as-you-go prices. Reservations provide a billing discount and don't affect the runtime state of your resources. After you purchase a reservation, the discount automatically applies to matching resources.

> You can purchase reservations from the Azure portal, APIs, PowerShell, and Azure CLI.

(5) As an alternative to Azure Reservations, we can use Azure Savings Plans to save money by committing to a fixed hourly spend on the VMs used by your AKS clusters for one-year or three-year terms. A savings plan can significantly reduce your resource costs by up to 65% from pay-as-you-go prices. Discount rates per meter vary by commitment term (1-year or 3-year), not commitment amount. With Azure Reservations, you commit to a specific virtual machine type in a particular Azure region.

For example, a D2v4 VM series in Central India for one year. With an Azure savings plan, you commit to spending a fixed hourly amount collectively on all the compute resources. For example, $5.00/hour on compute services for one year. Reservations only apply to the identified compute service and region combination.

Savings plan benefits apply to all usage from participating compute services across the globe, up to the hourly commitment. It would be best if you opted for a reservation for highly stable workloads that run continuously and where you have no expected changes to the VM series or region because Azure Reservations provide the greatest savings. Consider a compute savings plan for dynamic workloads where you need to run different-sized virtual machines or frequently change data center regions. Savings plans provide more flexibility and automatic optimization than reservations.

(6) Add spot node pools to your AKS cluster. Spot node pool is backed by a spot (VMSS). Using spot VMs for nodes with your AKS cluster allows you to take advantage of unutilized capacity in Azure at significant cost savings -Billed only 10% of the actual VM Cost

When deploying a spot node pool, Azure will allocate the spot nodes if there's capacity available. But there's no SLA for the spot nodes. A spot scale set that backs the spot node pool is deployed in a single fault domain and offers no high availability guarantees. For spot node pools, the cluster autoscaler will scale out the number of nodes after an eviction if additional nodes are still needed. Add a spot node pool to an AKS cluster.

(7) System pools must contain at least one node, while user node pools may contain zero or more nodes. Hence, you could set up a user node pool to automatically scale from 0 to N node. Using HPA based on CPU and memory or the metrics of an external system like Apache Kafka, RabbitMQ, Azure Service Bus, etc., you could configure your workloads to scale out and scale in using KEDA.

(8) Your AKS workloads may not need to run continuously, for example, a dev cluster with node pools running specific workloads. To optimize your costs, you can completely turn off an AKS cluster or stop one or more node pools in your AKS cluster, allowing you to save on compute costs.

- Stop and Start an Azure Kubernetes Service (AKS) cluster
- Start and stop a node pool on Azure Kubernetes Service (AKS)

(9) Deploy and manage containerized apps with AKS running on Ampere Altra Arm-based processors. For more information, see Azure Virtual Machines with Ampere Altra Arm-based processors.

(10) Migrate app workloads written in full .NET Framework, which requires running in Windows containers to .NET Standard. Migrated workloads will run in Linux containers with a smaller footprint in terms of a container image and hence provide better density. This decreases the number of agent nodes required to host and run applications.

(11) For multitenant solutions, physical isolation is more costly and adds management overhead. Logical isolation requires more Kubernetes experience and increases the surface area for changes and security threats, but it shares the costs.

(12) Use OSS tools like kubecost to monitor and govern an AKS cluster cost. Cost allocation can be scoped to a deployment, service, label, pod, and namespace, giving flexibility in how you charge back or show back costs to cluster users.

For more information, see Cost governance with Kubecost. In a multi-tenant scenario, you can track and associate Azure costs to individual tenants, based on their actual usage. Third-party solutions, such as kubecost, can help you calculate and break down costs across different teams and tenants.

(13) If you use an Azure Log Analytics workspace and Azure Monitor Container Insights to monitor the health and performance of your AKS workloads, you can use the ContainerLogV2 schema for container logs and configure this table to use the Basic Log data plan, which provides a low-cost way to ingest and retain logs for troubleshooting, debugging, auditing, and compliance.

(14) Use Azure tags in AKS, Azure tags on an AKS cluster to associate its related resources to a given workload or tenant. Azure tags are a helpful mechanism to track resource usage and chargeback their costs to separate tenants or business units within an organization.

Next Steps

In addition to this guide, also check the following resources:

- Cost Optimization section under WAF for AKS
- Cost Management section under Baseline architecture for an AKS cluster
- AKS Container Insights logging level and associated costs
- AKS – Cost Optimization Techniques
- The Azure FinOps Guide
- Well Architected for AKS cost savings