Open in app

Kumar Allamraju

Follow 17 Followers About

Accessing Secrets from an external Key Vault in AKS

Kumar Allamraju Jan 21 · 3 min read

The Container Storage Interface (CSI) is a standard for exposing arbitrary block and file storage systems to containerized workloads on Container Orchestration Systems (COs) like Kubernetes. Using CSI third-party storage providers can write and deploy plugins exposing new storage systems in Kubernetes without ever having to touch the core Kubernetes code.

Azure Key Vault provider for Secrets Store CSI driver allows you to store and retrieve secrets stored in an Azure Key Vault instance and use the Secrets Store CSI driver interface to mount them into Kubernetes pods.

The Secrets Store CSI driver `secrets-store.csi.k8s.io` allows Kubernetes to mount multiple secrets, keys, and certs stored in enterprise-grade external secrets stores into their pods as a volume. Once the Volume is attached, the data in it is mounted into the container's file system.

Prerequisites

- A valid Azure subscription
- Create an AKS cluster with managed identity
- Helm3

- Connect to your AKS Cluster

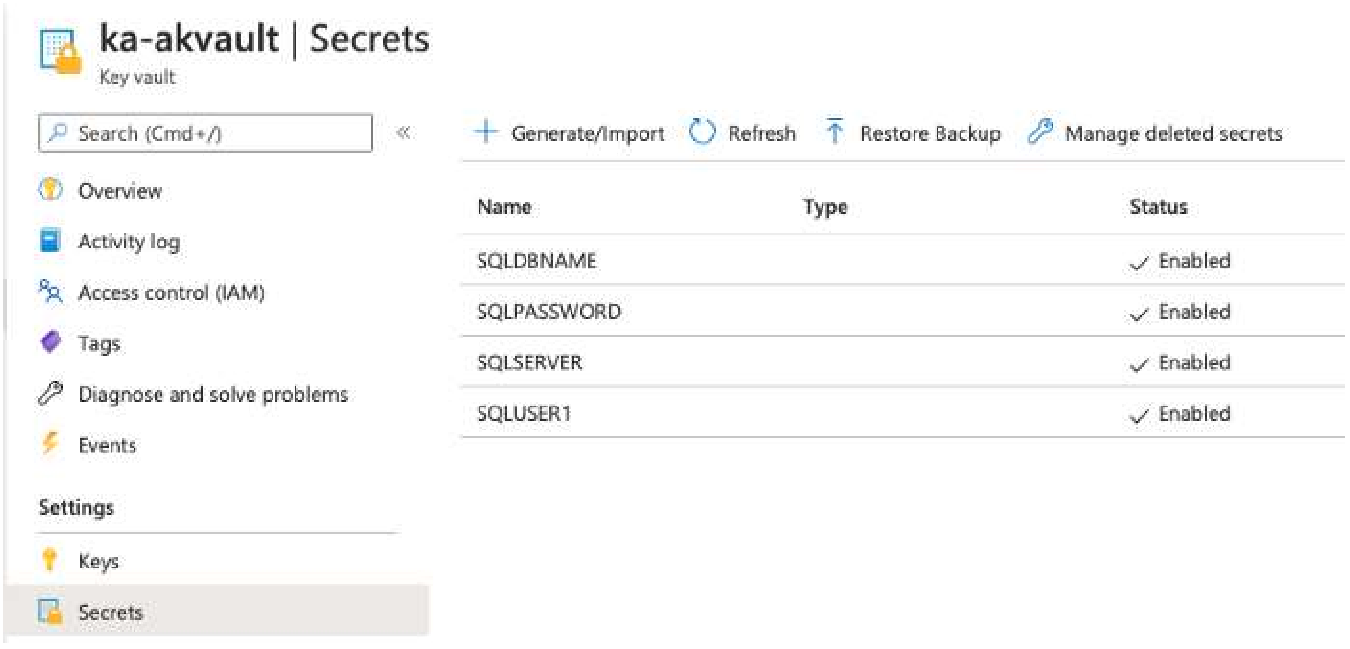
```
az aks get-credentials -g myResourceGroup -n myCSICluster2
```

- Install the CSI driver using helm3

```
$ helm repo add csi-secrets-store-provider-azure
https://raw.githubusercontent.com/Azure/secrets-store-csi-driver-
provider-azure/master/charts

$ helm install csi-secrets-store-provider-azure/csi-secrets-store-
provider-azure --generate-name
```

- Create an Azure Key Vault or use an existing one. In my example I'm storing database credentials

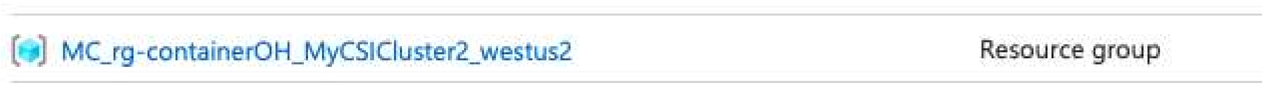


Name	Type	Status
SQLDBNAME		✓ Enabled
SQLPASSWORD		✓ Enabled
SQLSERVER		✓ Enabled
SQLUSER1		✓ Enabled

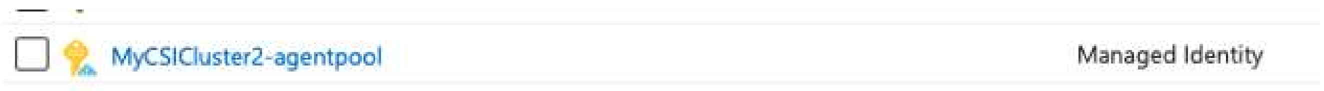
- The Azure Key Vault Provider offers 4 modes for accessing a Key Vault instance:
- 1. Service Principal
- 2. Pod Identity

I'll be focusing on #3 in this post.

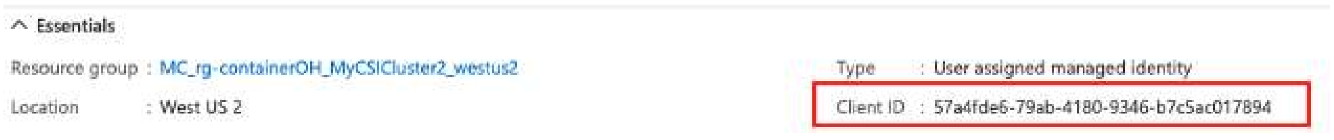
In order to get userAssignedIdentityID, go to your AKS Cluster Resource Group. Look at the resource group that was created by Azure e.g. If your AKS Cluster's name is MyCSICluster2, you'll see a resource group named as below


MC_rg-containerOH_MyCSICluster2_westus2 Resource group

- Go to this resource group and look for this Cluster's agentpool


MyCSICluster2-agentpool Managed Identity

- Click on the agentpool and copy the Client ID


Essentials

Resource group : MC_rg-containerOH_MyCSICluster2_westus2

Location : West US 2

Type : User assigned managed identity

Client ID : 57a4fde6-79ab-4180-9346-b7c5ac017894

You'll need to create a SecretProviderClass and reference the same in your Deployment. Here are my examples

kumarallamraju/medium

medium posts. Contribute to kumarallamraju/medium development by creating an account on GitHub.

github.com

- Now deploy these files

```
kubectl apply -f MySecretProvider.yaml
kubectl apply -f poi.yaml
```

```
NAME                           READY     STATUS      RESTARTS     AGE

poi-deployment-55bfb96bcf-nwtq8   1/1     Running     0            2m29s
poi-deployment-55bfb96bcf-s8hh7   1/1     Running     0            2m29s
```

That's it. You have successfully integrated Azure Key Vault provider with Secrets CSI Driver. In a future blog I'll talk about doing the same functionality with a service principal.

References:

Azure Key Vault Provider for Secrets Store CSI Driver

Azure Azure Kubernetes Service