Service Operator Sample

This sample demonstrates how to use the Service Operator to create resources using Helm charts.

Prerequisites

Service Operator prerequisites

Creating service principal

The service principal needed to provision resources needs access to an Azure subscription.

There are multiple ways on creating the service prinicipal. The chosen way for this guide is to use the Azure CLI.

1. Run `az login` to login to Azure
2. Run `az account list` to list all subscriptions
3. Run `az account set --subscription <subscriptionId>` to set the subscription to use
4. Run `az account show -o json` and store values for `id` and `tenantId` in environment variables `AZURE_SUBSCRIPTION_ID` and `AZURE_TENANT_ID` respectively (use the .env file)
5. Run `source .env` to load the environment variables
6. Run `az ad sp create-for-rbac --name azure-service-operator --role contributor --scopes subscriptions/<subscriptionId>` to create the service principal

   > info: The name of the service principal can be changed, only the id and password are important.

7. Store the values for `AZURE_CLIENT_ID` and `AZURE_CLIENT_SECRET` in the .env file as well

The service principal is now created and can be used to provision resources.

Configuring the cluster

The cluster needs to be configured to use the Service Operator. This is done by installing the Service Operator Helm chart. Before installing the chart, the cert manager needs to be installed to handle the certificates:

1. Install the cert-manager: `kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.8.2/cert-manager.yaml`
2. Wait until all pods are running when running: `kubectl get pods -n cert-manager -w`
3. Run `helm repo add aso2 https://raw.githubusercontent.com/Azure/azure-service-operator/main/v2/charts` to add the Service Operator Helm repository
4. Run `source .env` to load the environment variables set in the previous step

5. Run:

```
helm upgrade --install --devel aso2 aso2/azure-service-operator \
    --create-namespace \
    --namespace=azureserviceoperator-system \
    --set azureSubscriptionID=$AZURE_SUBSCRIPTION_ID \
    --set azureTenantID=$AZURE_TENANT_ID \
    --set azureClientID=$AZURE_CLIENT_ID \
    --set azureClientSecret=$AZURE_CLIENT_SECRET \
    --set crdPattern='*'
```

Creating a resource group (optional)

To create a resource group, the existing manifest can be used. The manifest is located in the `manifests` folder. The manifest can be applied by running:

```
kubectl apply -f manifests/resourcegroup.yaml
```

To view the created resource group, run:

```
kubectl get resourcegroups
kubectl describe resourcegroups/RESOURCE_GROUP_NAME
```

Creating a Postgres database sample

To create the necessary resources to run the Postgres database sample, the existing manifest from the `manifests` folder can be used.

Before applying the manifest, the following values need to be set in the .env file:

- `AZURE_POSTGRES_SERVER_NAME`
- `AZURE_POSTGRES_SERVER_USERNAME`
- `AZURE_POSTGRES_SERVER_PASSWORD`

Then, the values in the manifest need to be updated with the values from the .env file. This can be done by using envsubst after exporting all the environment variables:

```
export $(cat .env | xargs)
envsubst < manifests/postgres-demo-template.yaml > manifests/postgres-
demo-updated.yaml
```

The manifest can be applied by running:

```
kubectl apply -f manifests/postgres-demo-updated.yaml
```

> info: Ensure that the service principal and cluster are configured before applying the manifest.

After creating the resources, the demo application can be deployed using:

```
kubectl apply -f manifests/postgres-deployment.yaml
```

> info: You can ensure that the pod is running with `kubectl get pods -n asodemo`

Then, the application can be accessed by port-forwarding the pod:

```
kubectl port-forward -n asodemo deployment/azure-votes-postgresql-
deployment 8080:8080
```

The application can be accessed by navigating to http://localhost:8080.

Deleting the demo

To delete the demo, run:

```
kubectl delete namespace asodemo
```

Links

- Service Operator documentation
- Service Operator repository