Service Principals(Password) Based Authentication for automated applications in Azure with Azure Key Vault.

06 July 2021 09:44

1. Set up Service Principle using the steps listed in the following link which is used to Authenticate all Azure ML Services

   MachineLearningNotebooks/authentication-in-azureml.ipynb at master · Azure/MachineLearningNotebooks · GitHub

2. Set up Azure Key Vault to store all sensitive Information and Secrets(sensitive info) to the Vault Using below link

   https://docs.microsoft.com/en-us/azure/key-vault/secrets/quick-create-portal

   Additionally to Assign policy of access(Giving Key Vault Access to any user, group, resource etc.)

   https://docs.microsoft.com/en-us/azure/key-vault/general/assign-access-policy-portal

   Links To Authenticate keyvault from a python program

   https://docs.microsoft.com/en-us/azure/machine-learning/how-to-use-managed-identities?tabs=python

   Azure Identity client library for Python | Microsoft Docs

3. Crate a Compute Resource on Which our Pipeline will be Executed.
   To Authenticate from a Azure ML Pipeline or any other automated ml service ,Create the service/resource with a managed Ident ity and azure key vault access must be given to that managed identity in order to access any key vaults if necessary.

- Create a compute instance with managed identity.
  Link : https://docs.microsoft.com/en-us/azure/machine-learning/how-to-create-attach-compute-studio#managed-identity
- Give this Managed identity access to converse with Azure key Vault.
  Link : https://docs.microsoft.com/en-us/azure/key-vault/general/assign-access-policy-portal

4. On that Compute instance you can use Azure ml Python SDK Package to authenticate and
   Use any AzureML Service in conjunction with azure vault to hide any sensitive data from the code. Below is a code snippet for the same.

```
 # installing any necessary packages
import os
os.system(f"pip install azure-keyvault")
print("Testing Started !!!")

# ********** Azure Key Vault Authentication Start ********** #
from azure.identity import DefaultAzureCredential
from azure.keyvault.secrets import SecretClient

# Authenticate Azure Key Vault
credential = DefaultAzureCredential()
keyVaultName = os.environ["KEY_VAULT_NAME"]
KVUri = f"https://{keyVaultName}.vault.azure.net"
Secret_client = SecretClient(vault_url= KVUri, credential=credential)

# ********** Azure Key Vault Authentication  End********** #

# ********** Service Principle Based Authentication using Azure Key Fault Fetch Start ********** #
#print(f"Your secret is '{Secret.value}'.")
from azureml.core import Workspace, Dataset
from azureml.core.authentication import ServicePrincipalAuthentication

svc_pr = ServicePrincipalAuthentication(
    tenant_id=Secret_client.get_secret("tenant-id").value,
    service_principal_id=Secret_client.get_secret("service-principal-id").value,
    service_principal_password=Secret_client.get_secret("service-principal-password").value)

subscription_id = Secret_client.get_secret("subscription-id").value
resource_group = Secret_client.get_secret("resource-group").value
workspace_name = Secret_client.get_secret("workspace-name").value


 ws = Workspace(
     subscription_id=subscription_id,
     resource_group=resource_group,
     workspace_name=workspace_name,
     auth=svc_pr
     )
print("Found workspace {} at location {}".format(ws.name, ws.location))
```

```
# ********** Service Principle Based Authentication using Azure Key Fault Fetch End ********** #

# ********** Downloading Sample Code Repository and running services Start********** #
dataset = Dataset.get_by_name(workspace, name='test_code')
dataset.download(target_path='.', overwrite=False)

import main
main.print_hi("Test run Successfully Ended !!!")
# ********** Downloading Sample Code Repository and running services End********** #
```

New Python Libraries Required :

```
azure-keyvault-secrets==4.3.0
```