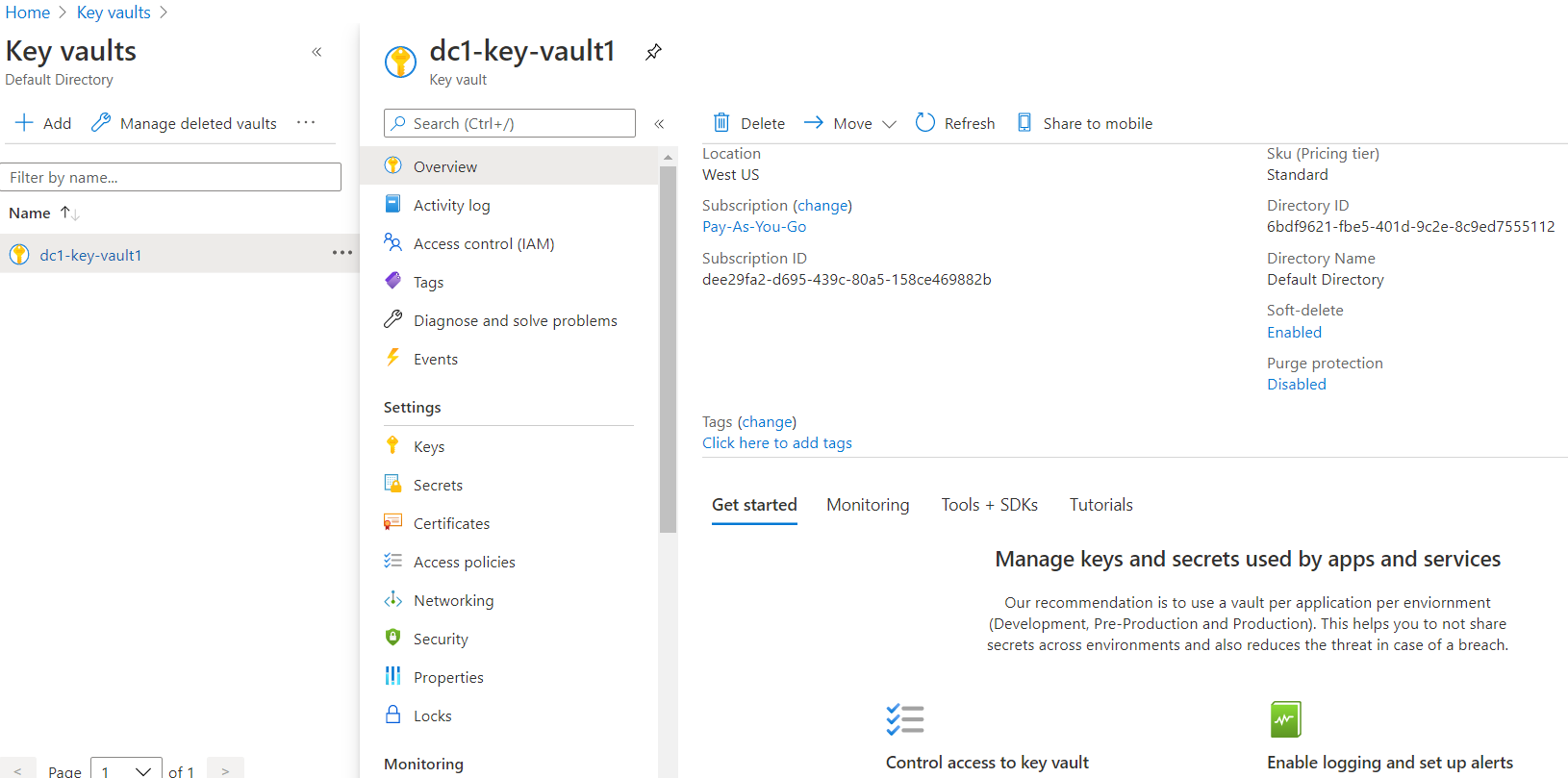
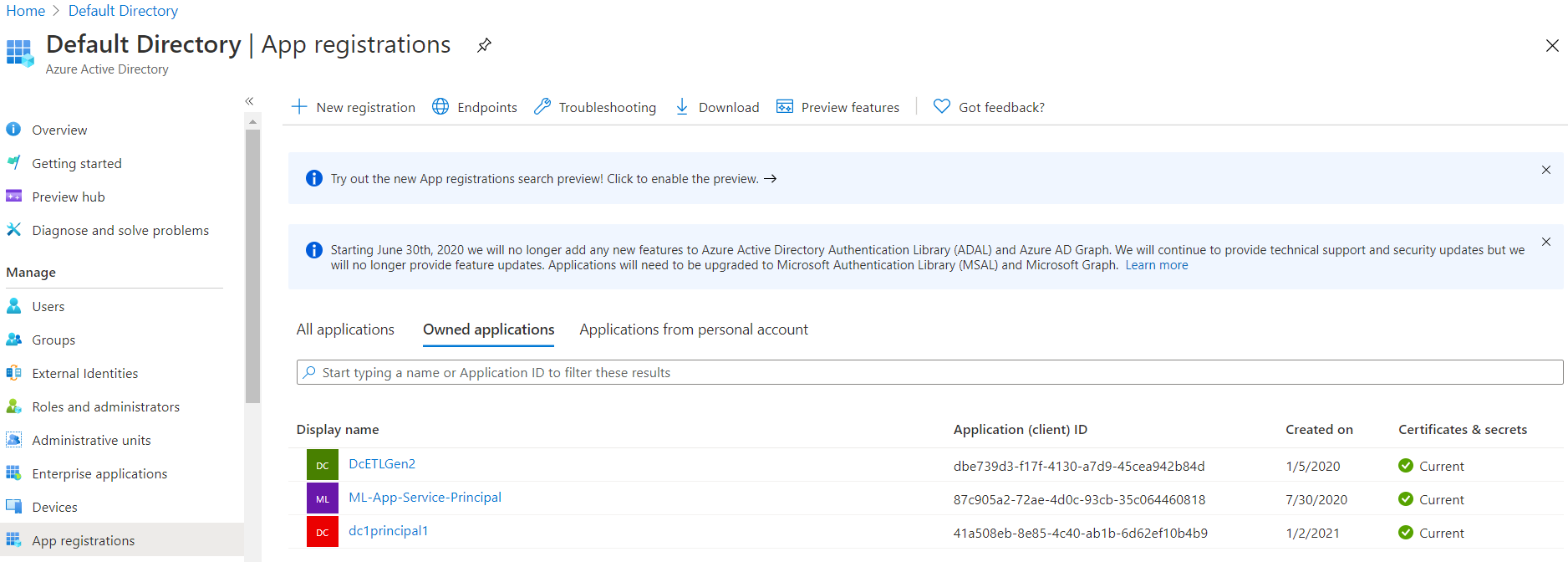
Access Secrets from Azure Key Vault using Service Principal

**Set up environment**

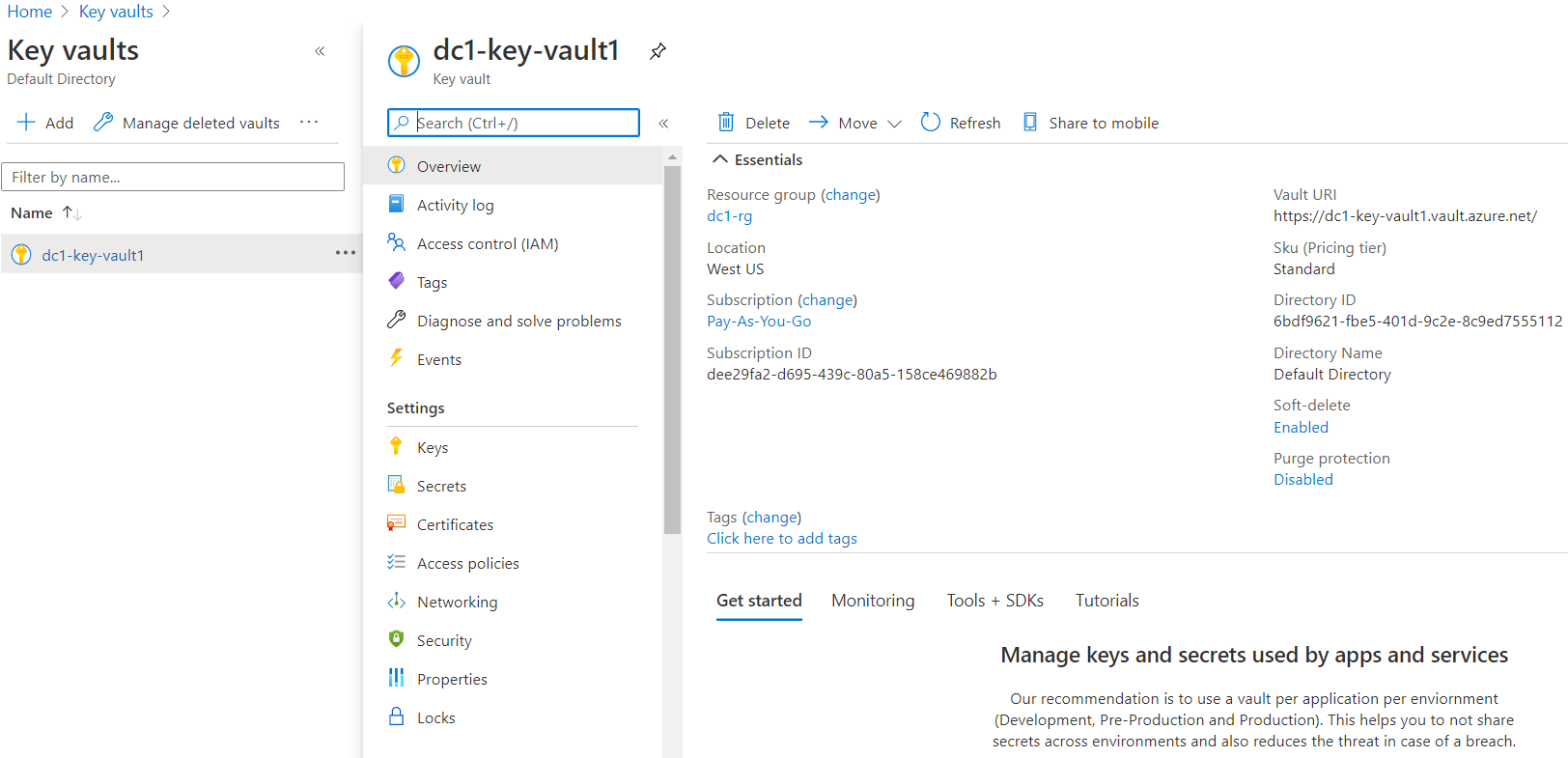
* Ensure Azure Key Vault exists. If not, create it.



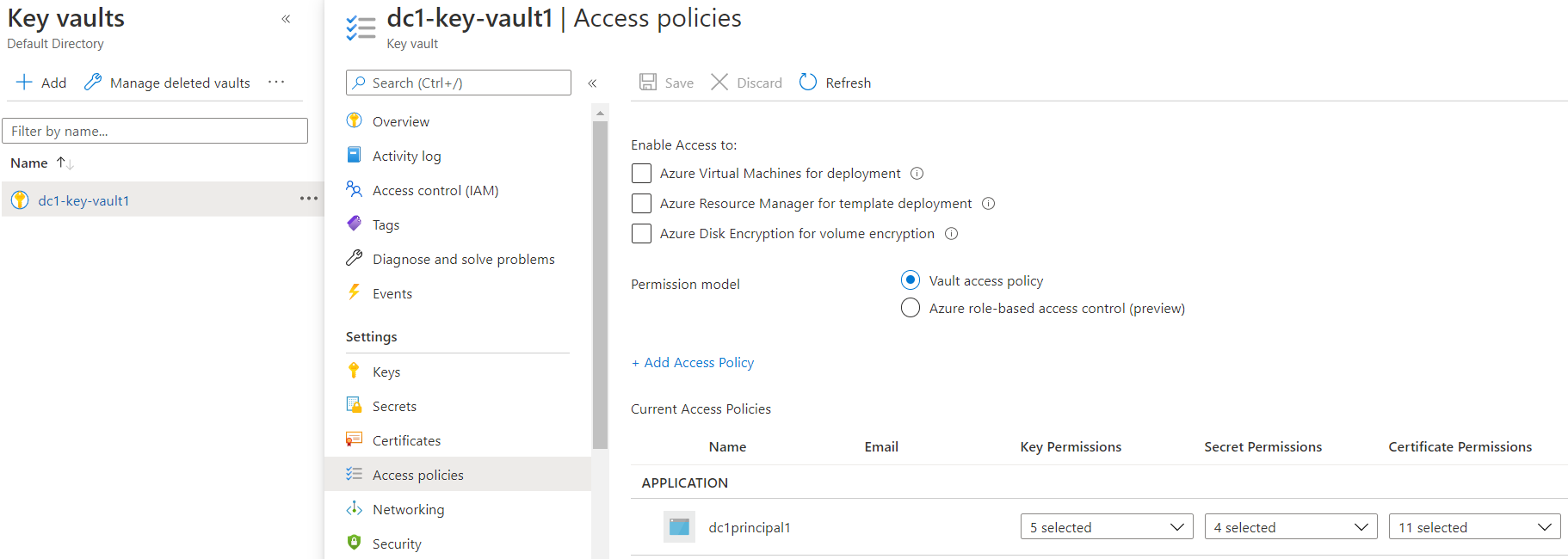
* Go to your Azure Active Directory and create an App Registration in the App registrations (e.g., dc1principal1)**.**



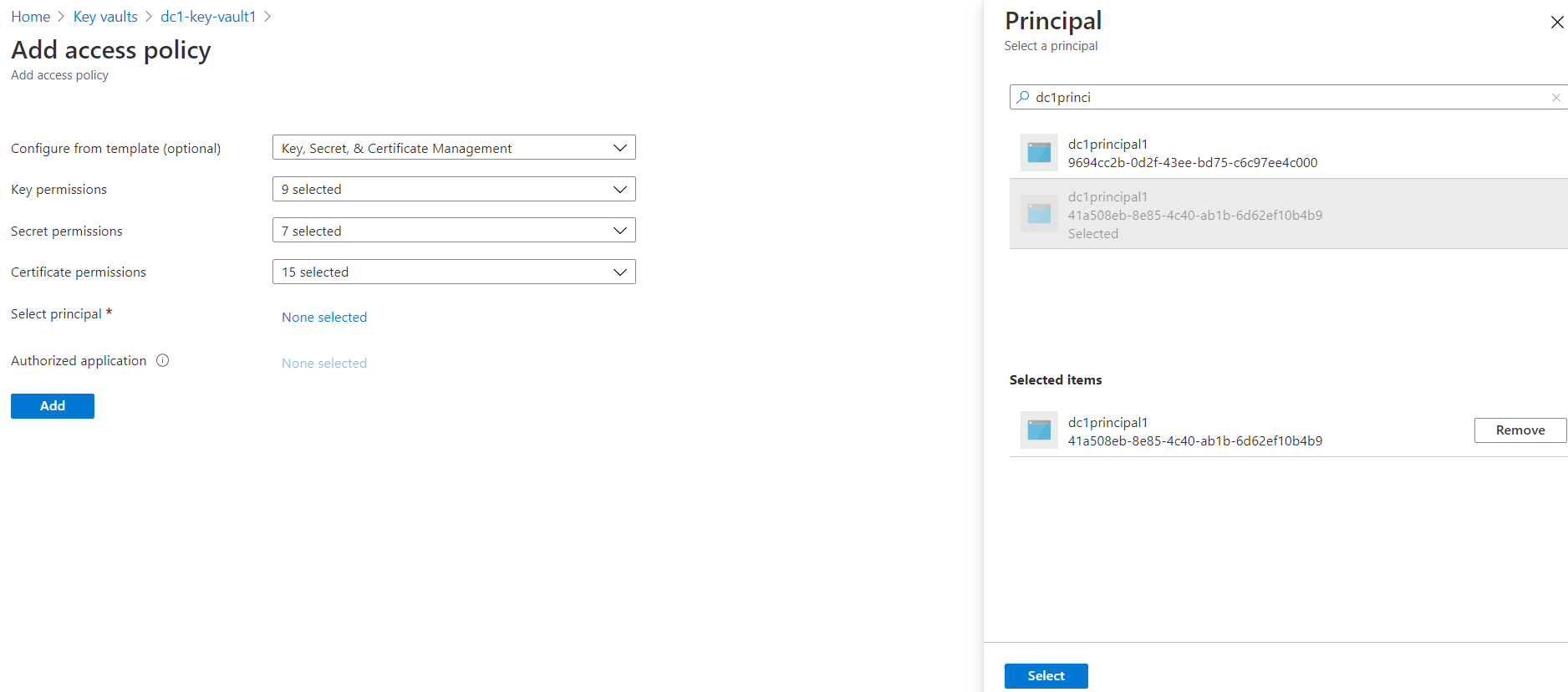
* Go back to Azure Key Vault instance you created earlier to configure Access policies



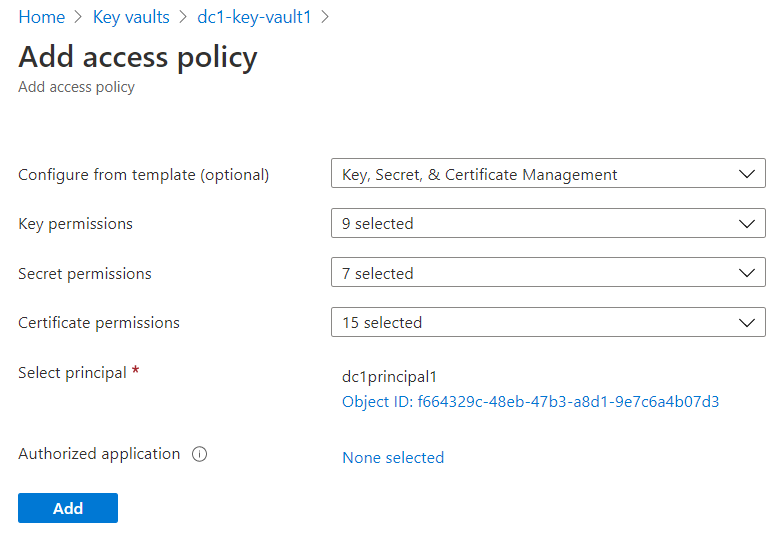
* Click Access policies on the left side menu



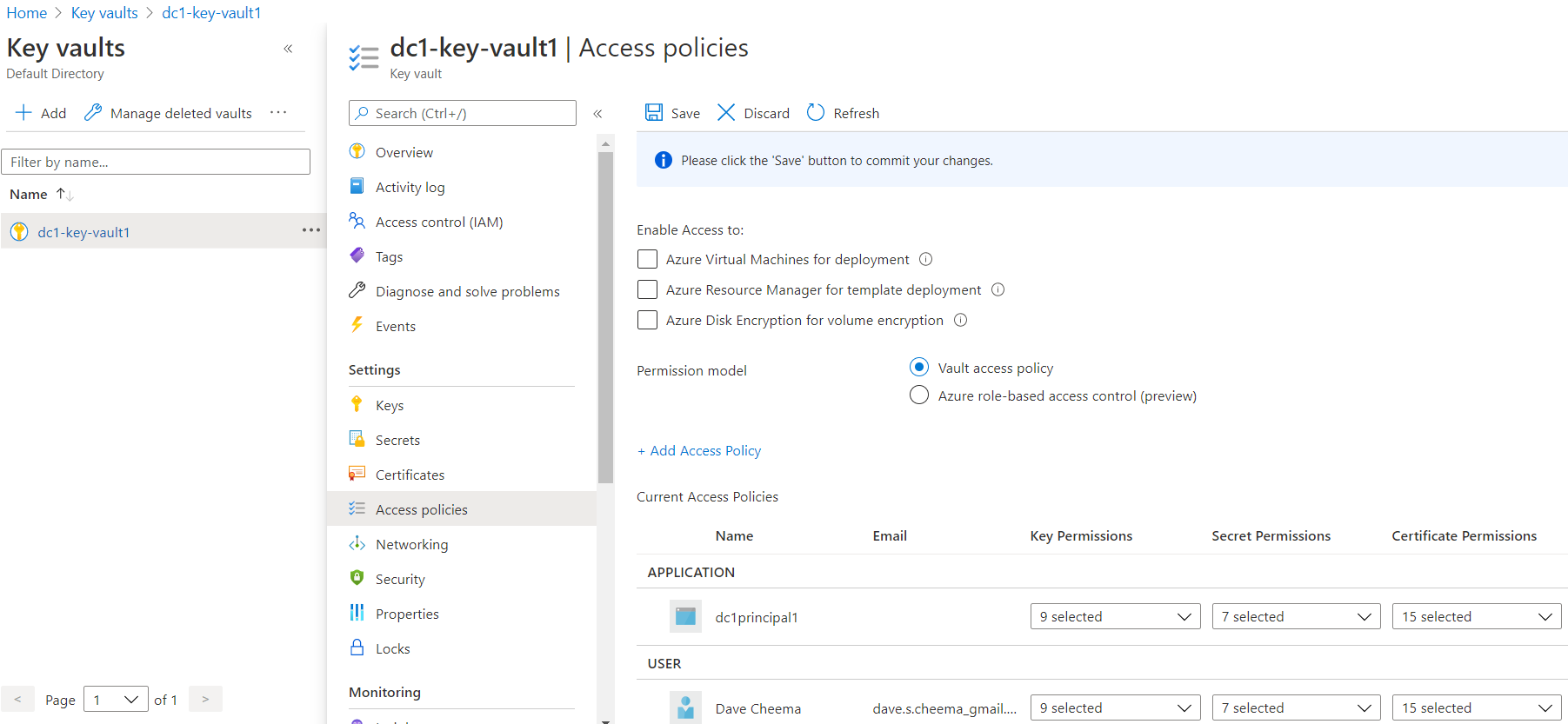
* Click on **+ Add Access Policy**, select appropriate **Key Permissions, Secret Permissions, Certificate Permissions**
* Click on the **Service Principal** link,search for and select the **App registration** (e.g., dc1principal1) in the search a principal blade



* After the principal is selected, click Add at the bottom

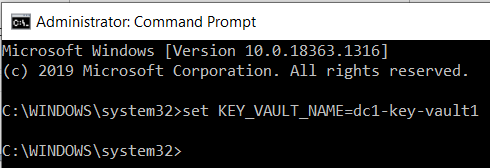


* IMPORTANT: When you get back to the Access policies page, click on the **Save** at the top of the page.



* Add the name of the Azure Key Vault instance name, e.g., dc1-key-vault1 to the environment using **one** of the following commands, depending upon your environment:

Windows: set KEY\_VAULT\_NAME=dc1-key-vault1



or

Windows PowerShell: $Env:KEY\_VAULT\_NAME="dc1-key-vault1"

or

macOS or Linux: export KEY\_VAULT\_NAME=dc1-key-vault1

FOR MORE INFORMATION:[**https://docs.microsoft.com/en-us/azure/key-vault/secrets/quick-create-java**](https://docs.microsoft.com/en-us/azure/key-vault/secrets/quick-create-java)

**// NOTE: The example below is written in JAVA. It demonstrates how to store, retrieve and delete secrets from an Azure Key Vault.**

**// You will not have all three functions in a single service. Most likely, you’ll have secret retrieval in a separate service.**

package dc.keyvaultreader;

import java.io.Console;

import com.azure.core.util.polling.SyncPoller;

import com.azure.identity.DefaultAzureCredentialBuilder;

import com.azure.security.keyvault.secrets.SecretClient;

import com.azure.security.keyvault.secrets.SecretClientBuilder;

import com.azure.security.keyvault.secrets.models.DeletedSecret;

import com.azure.security.keyvault.secrets.models.KeyVaultSecret;

public class AzureKeyVaultReader {

public static void main(String[] args) throws InterruptedException, IllegalArgumentException {

// Get Azure Key Vault instance name from the environment

String keyVaultName = System.getenv("KEY\_VAULT\_NAME");

String keyVaultUri = "https://" + keyVaultName + ".vault.azure.net";

// Create a secret client instance

SecretClient secretClient = new SecretClientBuilder()

.vaultUrl(keyVaultUri)

.credential(new DefaultAzureCredentialBuilder().build())

.buildClient();

// Set the secret name.

String secretName = "MyToasterSecret";

// Set the secret value.

String secretValue = "Toaster123";

try{

// Store the secret

secretClient.setSecret(new KeyVaultSecret(secretName, secretValue));

// Get secret from the Azure Key Vault instance

KeyVaultSecret retrievedSecret = secretClient.getSecret(secretName);

String secretValueReturned = retrievedSecret.getValue();

// For demo purpose only, delete the secret.

// NOTE: This a demo only. In the real-world scenario, creating and accessing secrets will be done by separate services,

// and most likely by separate roles.

SyncPoller<DeletedSecret, Void> deletionPoller = secretClient.beginDeleteSecret(secretName);

deletionPoller.waitForCompletion();

}

catch (Exception e){

System.out.println(e.getStackTrace().toString());

}

finally{

System.out.println("done.");

}

}

}