Azure Data Encryption Decryption using Managed Key stored in Azure Key Vault

### Configuration Steps

Construct the below Azure Resources in your Azure subscription. Make sure your account has valid admin permission (e.g. Global Administrator ) to create resources in subscription

* *Create Azure Key Vault*

In this Sample, we have created a AzureKeyVault named **kvcrysalisprakhour** as shown in the screenshot below.

Graphical user interface, application

Description automatically generated

* Assign Access Policy to your Azure Account used to access the key vault.

In Azure Key vault , Go to Access policies section and Assign required permission to your sign in account including Get , List , Encrypt , Decrypt , Wrap and Unwrap. Just for demo setup , I have selected all permissions as shown in the below screenshots –

Azure Sign-In Account added into access policies.

Graphical user interface, text, application, email

Description automatically generated

Permissions selected.

Graphical user interface, application

Description automatically generated

### Create a Managed Key for uploading to Key Vault

* Create/generate a key (.pem file) to be uploaded to Key Vault as Key

*Generally this key file is a customer managed key which is provided by the customer or the Neserdory owning the project.*

For Demo purpose we will locally generate an encrypted key (.pem) file using Open SSL.

* To generate an Open SSL key file (.pem) , first Install Open SSL plugin on machine either through <https://www.openssl.org/> or using Chocolatey <https://chocolatey.org/install>

Here we are demonstrating the open ssl install using Chocolatey and encrypted key creation through below bulleted steps -

* Using Chocolatey - > first install chocolatey on your windows machine from <https://chocolatey.org/install>
* Then go to Command prompt (in Admin mode) check if Chocolatey is installed using the command

C:\Windows\System32>**chocolatey**

This should display chocolatey version, incase installed successfully

(say Chocolatey v1.2.1 as per screenshot below ).

* Now install Open SSL through below command –

C:\Windows\System32>**choco install openssl**

Text

Description automatically generated

* In same command Prompt , switch directory path to the location where openssl.exe is installed in your machine . (generally path is C:\Program Files\OpenSSL-Win64\bin , else search for the file openssl.exe in your machine and get the path )
* Generate the key (.pem file) using below commands.

C:\Program Files\OpenSSL-Win64\bin>

openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650 -keyout **crysaliskey.pem** -out cert.pem

* On the command prompt Window , follow instructions and fill in the Region , Org and other details for key to be generated ( Eg. Refer response below as sample)

Note – The key file name in this example is crysaliskey.pem

Generating a RSA private key

......+++++

....+++++

writing new private key to 'crysaliskey.pem' -----

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank. -----

Country Name (2 letter code) [AU]:US

State or Province Name (full name) [Some-State]:NY

Locality Name (eg, city) []:NY

Organization Name (eg, company) [Internet Widgits Pty Ltd]:MS

Organizational Unit Name (eg, section) []:.

Common Name (e.g. server FQDN or YOUR name) []:crysalis

Email Address []:crysalis.com

* Once successfully completed details , press Enter to complete the execution.

Go to source path again (say C:\Program Files\OpenSSL-Win64\bin) and get your Key file generated (i.e. chrysaliskey.pem as shown in screenshot below )

Graphical user interface

Description automatically generated

### Import Generated managed Key to Key Vault

* Goto Keys section in Key vault ( kvcrysalisprakhour) in Azure portal. Click Generate/import.

Graphical user interface, text, application, email

Description automatically generated

* Select Option as Import
* Under File upload – select path of generated key file (i.e. crysaliskey.pem)
* Name: Give unique name to key (say crysalisDbEncryptionClientkey)
* KeyType : RSA
* Set Activation date as today and Expiration date (some future date) as per your requirement.
* Enabled: Yes

Click Create

Graphical user interface, application, email

Description automatically generated

### Create Encryption Initialization Vector and Store it in Key Vault Secret

* This is a 16 byte string that act as InitializationVector input seed for AesCryptoServiceProvider. (You can generate the seed online using below bulleted step)
* Goto <https://generate.plus/en/base64> -> section *Bulk base64 string generator*

Select length in bytes as **16** in the dropdown, generate and select one value from the output.

Eg . UWdQnddBFji8GNjM2vlp0w==

* Goto KeyVault 🡪 Secrets and save the generated 16 byte string generated in previous step with a secret name (say **crysalisDbInitializationVectorSecret** )

### Create Data EncryptionKey Secret

* This secret is a base 64 Encrypted Data Key Secret created using managed Key for the AesCryptoServiceProvider. It is generated from a 32byte input seed.

(You can generate the seed online using below steps )

* Goto <https://generate.plus/en/base64> -> section *Bulk base64 string generator*

Select length in bytes as **32** in the dropdown, generate and select one value from the output .

Eg. Secret Value for 32 byte string -> rCZjTh6moRRCUTdWg2yBrCVmh2JKfXNKC1GkvkQv2TI=

Refer step screenshot below -

Graphical user interface, text, application, email

Description automatically generated

* Now generate the base 64 Encrypted Data Key Secret from the 32byte string using the DataEncryptionDecryptionProgram
* Goto Crysalis ADO project 🡪 Repos section and clone the project **AzEncryptDecryptData** into your local machine and open the solution file **AzEncryptDecryptData.sln** in your Visual Studio.
* Open App.config, Goto <appSettings> section and set the 3 keys with following values -

1. Key : KV\_Key\_KeyIdentifier

Value : [Copy and past the KeyIdentifier url value from the AzureKey vault for the Key : crysalisDbEncryptionClientKey ]

Eg.

*https://kvcrysalisprakhour.vault.azure.net/keys/crysalisDbEncryptionClientKey/4ab3e4334edd44ffb7ca71417e26e84b*

Graphical user interface, text, application, email

Description automatically generated

1. Key KV\_Secret\_InitializationVectorName

Value : [KeyVault InitailizationVector Secret name]

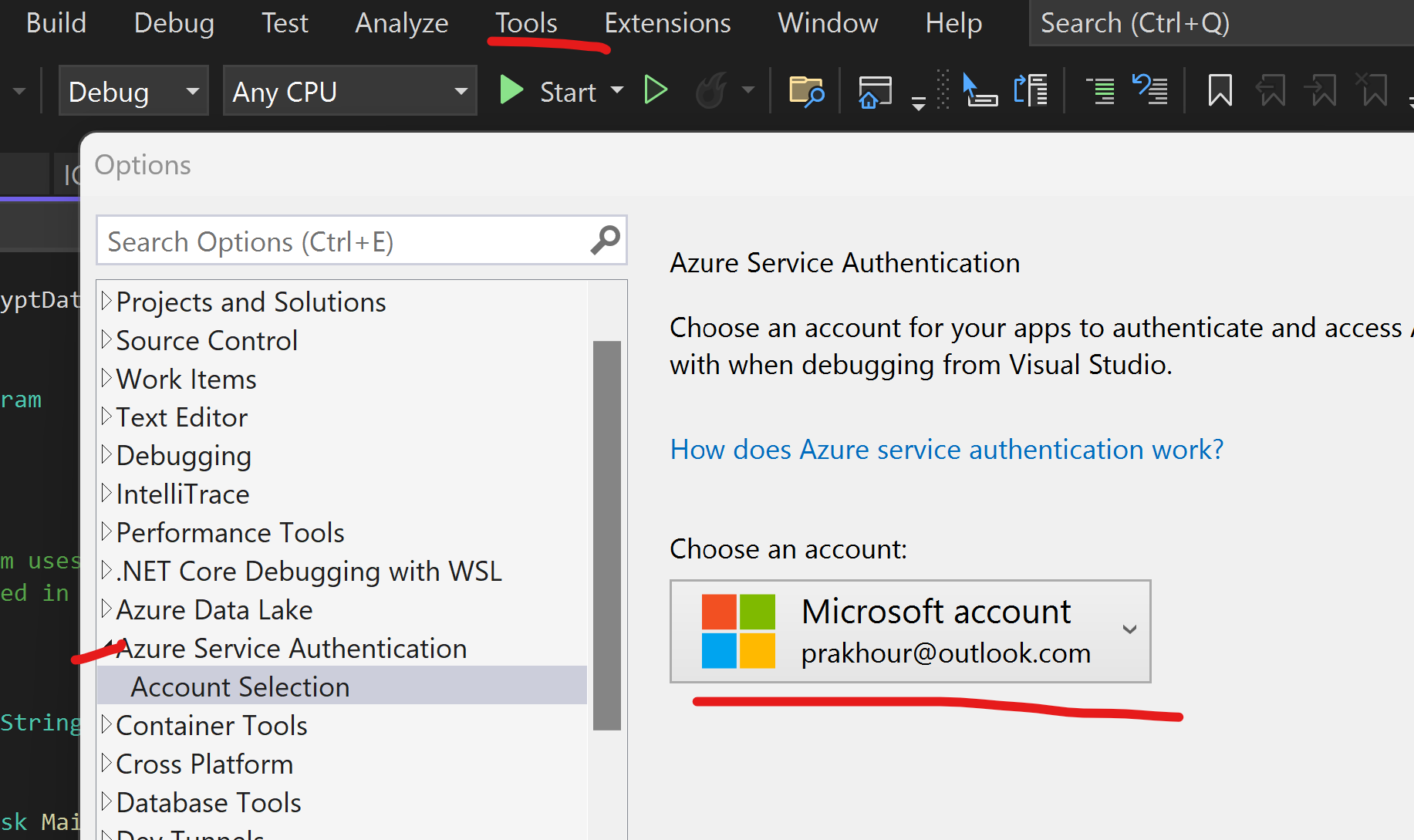
Eg. *crysalisDbInitializationVectorSecret*

1. Key KV\_Secret\_EncryptedKeyName

Value : [KeyVault EncryptedKey Secret name]

Eg. *crysalisDbEncryptedKeySecret*

* Also, Make sure you are logged in to Visual Studio with same Azure Account which was added under the access policies of KeyVault in earlier step.
* Also, Under Tools > Options > Azure Service Authentication, stay Signed in with the same azure account credentials to allow default credentials signup to azure keyvault.



* Run the **AzEncryptDecryptData.sln** in your Visual Studio (preferrably Admin mode)
* In the programs Console , select 1 to start the execution and wait for a while.
* Once encryptor Decryptor is ready , In the program console select Option **0**

(Option 0 will be one time activity )

* Pass the 32 byte string generated earlier as input (i.e rCZjTh6moRRCUTdWg2yBrCVmh2JKfXNKC1GkvkQv2TI= )

Get the Output generated (output string will vary for each run, so copy it for the first time run)

Eg. c60GYyDIkyginX4oU2xiFytwvn+SeoMoJrkW+4mTh2qys6DmnHhaACpLZdG1q/1c4JwziqOhE3FiORqZe3UXfCHDiPOwUMcIbnNDJOIV4HhEQgLnMrJE97W0AwOwXWWShz6SQGf28kOp8IDqhwZ6YFN/5v9rT1d5dT8HXlM6pQ++uxfMJRG+BcxGTZB8rVqQdEfhRQupZ6k9ye+ALqkucj5VuCJpXBHiBd6nkxuN5nTNCpU1zXfYEWijBtMlZ6ta7kWern5yuB2u2aXQjMsfTM/3k2mEeFpyxnvlPUc/oChYVlogw1wuXGWmtZFndZ3paGLw1YD2RXjgycZLdnaD8w==

Text

Description automatically generated

* Goto KeyVault 🡪 Secrets and save the output generated in previous step under a secret name (say **crysalisDbEncryptedKeySecret** )

Graphical user interface, text, email

Description automatically generated

* Once done , Crosscheck Secrets section once again. You should see 2 secrets now –

Eg.

crysalisDbInitializationVectorSecret

crysalisDbEncryptionClientKey

Graphical user interface, application, email

Description automatically generated

### Backup All KeyVault Keys and Secret

As precaution , Please download and keep a backup of the Keyvault keys and secret before proceeding for data encryption decryption .This is to avoid accidently missing the source key & secret values that was used to encrypt and decrypt of all data in your project.

Eg. Backup below key and secret data

crysalisDbEncryptionClientKey

crysalisDbInitializationVectorSecret

crysalisDbEncryptedKeySecret

One sample , Download and backup Kevault keys (.pem file) is shown in screenshot below.

Graphical user interface, text, application, email

Description automatically generated

### Run Data Encryption Decryption Program

* Once both secrets are saved in Keyvault , Run the **AzEncryptDecryptData.sln** in your Visual Studio to launch the program console.
* Stay signed in with same Azure login account (as mentioned in previous steps ).
* In the Program Console , select 1 to initiate the encryptor , decryptor program and follow instructions

#### Test Encryption

* In the console select Option **1** to **Encrypt**
* Input any String to encrypt

(refer next screenshot for sample execution)

#### Test Decryption

* In the console select Option **1** to **Decrypt**
* Pass the previous valid encrypted output string to get the original String

(refer next screenshot for sample execution)

Text

Description automatically generated

Note : You can integrate this utility to Encrypt data before saving to Datalake ( like SQL Server or COsmosDB ) , then Decrypt the same before sending response to source client application or user. This way you can keep the user personal information ( like First name , Lastname , Address , Country , Phone number etc. ) secure and confidential in the project datalake.

-------------------------------------------- Thank you --------------------------------------------------------------------------