Key Management

Advanced Conversational Engagement

Exported on 06/06/2025

Table of Contents

1 Azure policies AAB Key Vault 5
2 Key Rotation Policy 6
3 CMK keys with rotation policy 8
4 PostgreSQL Flexible Server 9
5 Azure Managed HSM 10
6 Key Vault Events 11
7 Reference 12
8 Note 13

Encrypting data utilizes an encryption key which can be managed by Microsoft or by yourself (the customer). For several Azure resources within AAB the usage of Customer Managed Keys is mandatory.

To be in more control of the keys we are using Customer Managed keys for data encryption.

The customer managed key is generated in the Key Vault which and can then be used for encryption of data. These keys can be created using Bicep template which than can be referenced in the azure resource which depends on the customer managed key.

Following is the reference bicep template to create the CMK.

```
resource key 'Microsoft.KeyVault/vaults/keys@2022-07-01' = {
  name: keyName
  parent: keyVault
  properties: {
    attributes: {
      enabled: true
      exp: pkeyExpiration
    }
    keyOps: pkeyOps
    kty: keyType
    rotationPolicy: {
      attributes: {
        expiryTime: pExpiryTime
      }
      lifetimeActions: [
        {
          action: {
            type: 'Rotate'
          }
          trigger: {
            timeAfterCreate: rotateTime
          }
        }
        {
          action: {
            type: 'Notify'
          }
          trigger: {
            timeBeforeExpiry: notifyTime
          }
        }
      ]
    }
  }
}
```

Even though bicep supports incremental deployment, we can not use the same template to update/modify the same key after creation. This is because the template defines the key expiration date as 365 days from

the time the key is created. This time will differ with each pipeline run and will result in key not found or key with the same name exist errors.

1 Azure policies AAB Key Vault

There are two AAB azure policies which are added onto the key vault for CMK as below.

1. *Keys should have more than the specified number of days before expiration*: If a key is too close to expiration, an organizational delay to rotate the key may result in an outage. Keys should be rotated at a specified number of days prior to expiration to provide sufficient time to react to a failure. As a best practice, organizations should have sufficient time to rotate a Key before it expires. Hence, Keys should not be created when the number of days before expiration is '7' or less. For that reason, this Azure Policy denies the creation of a Key when the 'Days to expiration' setting is set to '7' or less.
2. *Keys should have the specified maximum validity period*: Manage your organizational compliance requirements by specifying the maximum amount of time in days that a key can be valid within your key vault. As a best practice, Keys within an Azure Key Vault should only be valid for a maximum number of 365 days. Therefore, this Azure Policy denies the creation of a Key when its validity period is above 365 days.

To be compliant on the above two policies we have add the expiry date (key creation date+365 days) as well as key rotation policy at the time of key creation.

2 Key Rotation Policy

The key rotation policy allows users to configure rotation and Event Grid notifications near expiry notification.

Key rotation policy settings:

- *Expiry time*: key expiration interval. It's used to set expiration date on newly rotated key. It doesn't affect a current key.
- *Enabled/disabled*: flag to enable or disable rotation for the key
- *Rotation types*:
  Automatically renew at a given time after creation (default)
  Automatically renew at a given time before expiry. It requires 'Expiry Time' set on rotation policy and 'Expiration Date' set on the key.
- *Rotation time*: key rotation interval, the minimum value is seven days from creation and seven days from expiration time
- *Notification time*: key near expiry event interval for Event Grid notification. It requires 'Expiry Time' set on rotation policy and 'Expiration Date' set on the key.

  Following is the rotation policy added in the properties for the key

```
rotationPolicy: {
      attributes: {
        expiryTime: pExpiryTime
      }
      lifetimeActions: [
        {
          action: {
            type: 'Rotate'
          }
          trigger: {
            timeAfterCreate: rotateTime
          }
        }
        {
          action: {
            type: 'Notify'
          }
          trigger: {
            timeBeforeExpiry: notifyTime
          }
        }
      ]
    }
```

For the Current CMK keys the rotation time is set to 11th month after key creation and the notification time is set to 45 days before key expiry i.e. you would get a notification 15 days before key rotation.

Note: Key vault needs to have set and get rotation policy enabled in key permissions(i.e. RG service connection should have these access policy enabled) to be able to add the rotation policy to the key.

Following are the resources for which are using CMK in DTAP:
1. Azure Data Bricks
2. AI search
3. Cosmos DB
4. PostgreSQL Flexible server
5. Service bus
6. Disk encryption set(for AKS)

When the Key is rotated, it will create a new version of it in the key vault. This new version of key is automatically picked up by the above mentioned resources(apart from PostgreSQL Flexible server). There is no need to manually update the new version of the key in the resource.

3 CMK keys with rotation policy

Below is the list of keys for which key rotation policy has been added

- Azure data bricks: *acep03-{env}-adb-ws-key*
- AI search: *acep03-{env}-aisearch-key*
- Cosmos DB: *acep03-{env}-cosmosdb-key*
- Service Bus: *acep03-d-servicebus-dev-key* in dev and *(acep03-{env}-servicebus-key)* in TAP
- Disk encryption set: *acep03-dev-aks-key* in dev and (*acep03-{env}-aks-key*)-TAP

4 PostgreSQL Flexible Server

Currently PostgreSQL Flexible server does not support automatic pickup of new key versions from the key vault. If the key is expired or disabled, the server goes to inaccessible state which results in downtime. To avoid this we have not added key rotation policy for the CMK key of PostgreSQL Flexible server. To manage the key you have to rotate and update the new version of the key using the following azure CLI command in the maintenance pipeline.

Command: az postgres flexible-server update --resource-group <RESOURCE GROUP NAME> --name <FLEXIBLE SERVER NAME> --key <KEY IDENTIFIER> --identity <MANAGED IDENTITY>

Key rotation pipeline: Pipelines - Runs for Rotate-EncryptionKey (azure.com)[1]
Update the key version on PostgreSQL FLexible server: Pipelines - Run 20240913.1[2]

We had opened a ticket with Microsoft to check when the autorotate feature would be available for PostgreSQL Flexible Server. The Product team mentioned that they have auto rotation feature and are expecting it to be release by the end of Q3 of 2024 or start of Q4 2024. As definitive solution. There are no alert mechanisms to warn you once the feature available but whenever new feature is added, it is include in the release notes: Release notes for Azure DB for PostgreSQL - Flexible Server - Azure Database for PostgreSQL - Flexible Server | Microsoft Learn[3]

As a result, we can check the feature from the above link.

1 https://dev.azure.com/cbsp-abnamro/GRD0001014/_build?definitionId=88401
2 https://dev.azure.com/cbsp-abnamro/GRD0001014/_build/results?buildId=8860399&view=results
3 https://eur03.safelinks.protection.outlook.com/?url=https%3A%2F%2Flearn.microsoft.com%2Fen-us%2Fazure%2Fpostgresql%2Fflexible-server%2Frelease-notes&data=05%7C02%7Ckishita.wahane%40nl.abnamro.com%7C8f120b8a1b5f42c5025b08dca70c0626%7C3a15904d3fd94256a753beb05cdf0c6d%7C0%7C0%7C638568917327186506%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C0%7C%7C%7C&sdata=A2J009lWT8q9AFwENU3xQ8WSHHX%2Bkiaq9e%2FPht0qdec%3D&reserved=0

5 Azure Managed HSM

Apart form the above mentioned resources we have CMK enabled for Storage account which is onboarded onto Azure Key Vault Managed HSM. This centralized service in the Azure Cloud has been setup and owned by the Crypto Services team which is a centralized service in the Azure Cloud. Crypto team is responsible for the rotation of this key.
As a part of HSM onboarding we have created a CMK key in the HSM key vault.
Following is the reference pipeline to Create an HSM key and code snippet.

- *Pipeline*: https://dev.azure.com/cbsp-abnamro/GRD0001014/_build?definitionId=53518
- Repository: https://dev.azure.com/cbsp-abnamro/GRD0001014/_git/acep-platform-infrastructure?path=/pipelines/pre-infra-pipeline.yaml

After the HSM key is created we have to update our encryption in the resource as well. For example, to encrypt our storage account with the same we can add the following code snippet at encryption property for storage account.

```
var pencryption = {
  requireInfrastructureEncryption: prequireInfrastructureEncryption
  keySource: 'Microsoft.Keyvault'
  services: {
    blob: {
      enabled: true
    }
    file: {
      enabled: true
    }
  }
  keyvaultproperties:{
    keyname: pKeyName
    keyvaulturi: pHsmKeyVaultUri
  }
  identity: {
    userAssignedIdentity: pUserAssignedIdentity
  }
}
```

Where *pKeyName* is the HSM key which we created in the HSM key vault.

6 Key Vault Events

We can configure alerts over key vault by integrating the key vault with event grid. These alerts will be triggered whenever there is a change in the status of keys, secrets and certificates stored in our key vault.

Currently the key vault supports 10 events which we can configure:

1. Certificate New Version Created: Triggered when a new certificate or new certificate version is created.
2. Certificate Near Expiry: Triggered when the current version of certificate is about to expire. (The event is triggered 30 days before the expiration date.)
3. Certificate Expired: Triggered when the current version of a certificate is expired.
4. Key New Version Created: Triggered when a new key or new key version is created.
5. Key Near Expiry: Triggered when the current version of a key is about to expire. The event time can be configured using key rotation policy.
6. Key Expired: Triggered when the current version of a key is expired.
7. Secret New Version Created: Triggered when a new secret or new secret version is created.
8. Secret Near Expiry: Triggered when the current version of a secret is about to expire. (The event is triggered 30 days before the expiration date.)
9. Secret Expired: Triggered when the current version of a secret is expired.
10. Vault Access Policy Changed: Triggered when an access policy on Key Vault changed. It includes a scenario when Key Vault permission model is changed to/from Azure role-based access control.

Currently these events have only been configured in Development environment as the bicep api for key vault events is in preview state. Once it becomes GA we can proceed with higher environments.

7 Reference

FSCP Azure Cookbook - Key Vault - Software Development - Confluence (abnamro.com)[4]
Azure Managed HSM Onboarding - Crypto Services - Confluence (abnamro.com)[5]

Azure Key Vault as Event Grid source - Azure Event Grid | Microsoft Learn[6]

Microsoft.EventGrid/systemTopics - Bicep, ARM template & Terraform AzAPI reference | Microsoft Learn[7]

Microsoft.EventGrid/systemTopics/eventSubscriptions - Bicep, ARM template & Terraform AzAPI reference | Microsoft Learn[8]

4 https://confluence.int.abnamro.com/pages/viewpage.action?spaceKey=GRIDAD&title=FSCP+Azure+Cookbook+-+Key+Vault#FSCPAzureCookbookKeyVault-AzurepoliciesAABKeyVault
5 https://confluence.int.abnamro.com/pages/viewpage.action?spaceKey=CS&title=Azure+Managed+HSM+Onboarding
6 https://learn.microsoft.com/en-us/azure/event-grid/event-schema-key-vault?tabs=cloud-event-schema
7 https://learn.microsoft.com/en-us/azure/templates/microsoft.eventgrid/systemtopics?pivots=deployment-language-bicep
8 https://learn.microsoft.com/en-us/azure/templates/microsoft.eventgrid/systemtopics/eventsubscriptions?pivots=deployment-language-bicep

8 Note

If due to some reason we want to change the CMK for a particular Azure resource for example service bus, then first we have to add the new CMK to the Resource. Only after the successful update should you delete the previous key. The reason for this is such that if you delete the pervious key first, then you lose the access to it and the resource goes into in accessible state.