Remove a Transparent Data EncryptionProtector Using PowerShell

Last updated by | Soma Jagadeesh | Jan 10, 2021 at 9:46 PM PST

Contents

Issue

Remove a Transparent Data EncryptionProtector Using PowerShell

Prerequisites

• You must have an Azure subscription and be an administrator on that subscription • You must have Azure PowerShell version 4.2.0 or newer installed and running. • This tutorial assumes that you are already using a key from Azure Key Vault as the TDE Protector for an Azure SQL Database or Azure SQL Data Warehouse. Visit Transparent Data Encryption with BYOK Support to learn more

Overview

This tutorial describes how to respond to a potentially compromised TDE Protector for an Azure SQL Database or Azure SQL Data Warehouse that is using TDE with Bring Your Own Key (BYOK) support. To learn more about BYOK support for TDE, visit the overview page. The procedures described below should only be done in extreme cases or in test environments. Please go over the tutorial carefully, as deleting actively used TDE Protectors from Azure Key Vault can result in data loss. If a key is ever suspected to be compromised, such that a service or user had unauthorized access to the key, it’s best to delete the key. Keep in mind that once the TDE Protector is deleted in Key Vault, all connections to the encrypted databases under the server will be blocked, and these databases will go offline and get dropped within 24 hours. Old backups encrypted with the compromised key will no longer be accessible. This tutorial goes over two approaches depending on the desired result after the incident response: • To keep the Azure SQL Databases / Data Warehouses accessible • To make the Azure SQL Databases / Data Warehouses inaccessible

```
To keep the encrypted resources accessible
```

1. Create a https://docs.microsoft.com/en-us/powershell/module/azurerm.keyvault/add-azurekeyvaultkey?view=azurermps-6.13.0&viewFallbackFrom=azurermps-4.1.0

2. Add the new key to the server, and update it as the server's new TDE Protector.

```
# Add the key from Key Vault to the server
Add-AzureRmSqlServerKeyVaultKey -ResourceGroupName
<SQLDatabaseResourceGroupName> -ServerName <LogicalServerName> -
KeyId <KeyVaultKeyId>


# Set the key as the TDE protector for all resources under the
server
Set-AzureRmSqlServerTransparentDataEncryptionProtector -
ResourceGroupName <SQLDatabaseResourceGroupName> -ServerName
<LogicalServerName> -Type AzureKeyVault -KeyId <KeyVaultKeyId>
```

3. Make sure the server and any replicas have updated to the new TDE Protector.

```
Get-AzureRmSqlServerTransparentDataEncryptionProtector -ServerName
<LogicalServerName> -ResourceGroupName
<SQLDatabaseResourceGroupName>
```

- `Note: It may take up to 24 hours for the new TDE Protector to propagate to all databases and secondary da`

4. Take a https://docs.microsoft.com/en-us/powershell/module/azurerm.keyvault/backup-azurekeyvaultkey?view=azurermps-6.13.0&viewFallbackFrom=azurermps-4.1.0 in Key Vault.

```
<# -OutputFile parameter is optional; if removed, a file name is
automatically generated. #>

Backup-AzureKeyVaultKey -VaultName <KeyVaultName> -Name
<KeyVaultKeyName> -OutputFile <DesiredBackupFilePath>
```

5. Delete the compromised key from Key Vault.

```
Remove-AzureKeyVaultKey -VaultName <KeyVaultName> -Name
<KeyVaultKeyName>
```

6. To restore a key to Key Vault in the future

```
Restore-AzureKeyVaultKey -VaultName <KeyVaultName> -InputFile
<BackupFilePath>
```

```
To make the encrypted resources inaccessible
```

- Drop the databases that are being encrypted by the potentially compromised key.
- The database and log files will be automatically backed up, so a point-in-time restore of the database can be done at any point (as long as you provide the key).
- Backup the key material of the TDE Protector in Key Vault.
- Remove the potentially compromised key from Key Vault
- To restore a key to a Key Vault in the future:

Investigation/Analysis

Describes the steps/queries to use for confirming the issue

Mitigation

What to do to resolve the issue. Maybe file an ICM, might have a canned RCA template to be used, etc.

RCA (optional)

More Information (optional)

Detailed information/background that may be useful but isn’t strictly required for troubleshooting. Often this is the “verbose” details that one usually doesn’t need

Public Doc Reference (optional)

Internal Reference (optional)

Classification

Root cause tree:

How good have you found this content?

🙂 🙁