# Transparent Data Encryption with Bring Your Own Key support for Azure SQL Database

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

Contents

- Issue
- Investigation/Analysis
- Mitigation
- RCA (optional)
- More Information (optional)
- Public Doc Reference (optional)
- Internal Reference (optional)
- Classification

## Issue

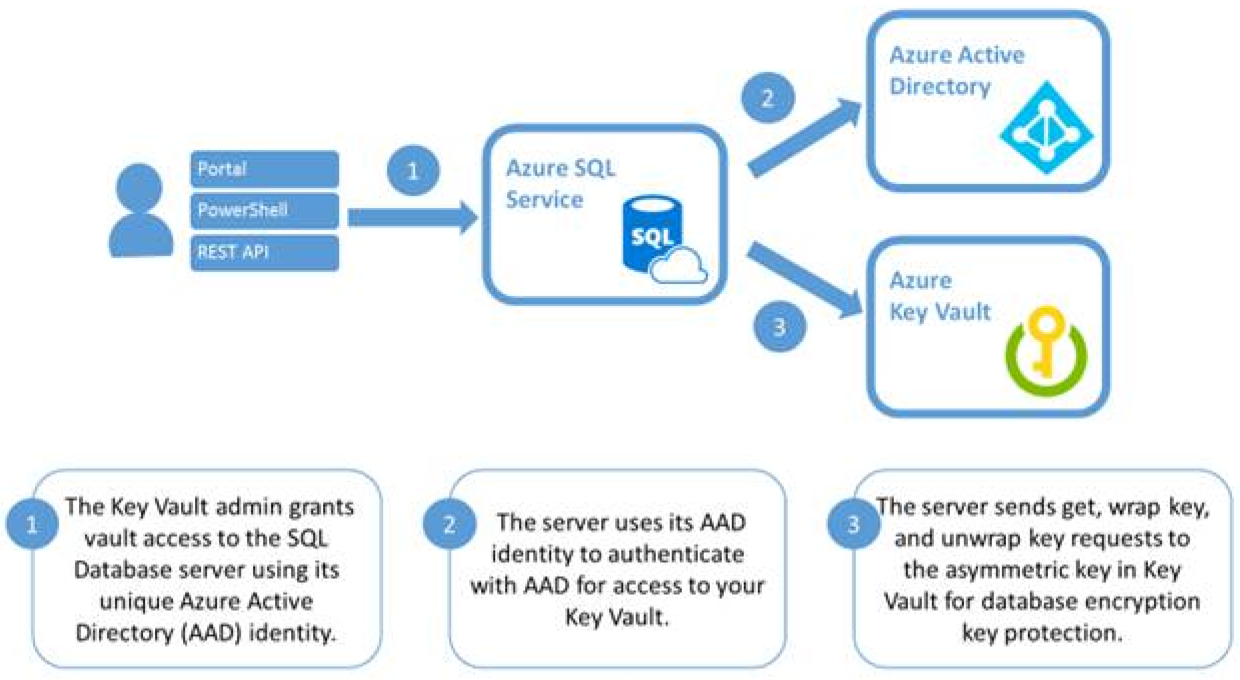
To learn more generally about Transparent Data Encryption, visit: https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption?redirectedfrom=MSDN&view=sql-server-ver15

Bring Your Own Key (BYOK) support for Transparent Data Encryption (TDE) allows the user to take control over their TDE encryption keys and control who can access them and when. Azure Key Vault (AKV), which is Azure’s cloud-based external key management system, is the first key management service that TDE has integrated with for BYOK support. With BYOK, the database encryption key is protected by an asymmetric key stored in AKV. The asymmetric key is set at the server level and inherited by all databases under that server. With BYOK support, users can now control key management tasks including key rotations, key vault permissions, key deletions, and auditing/reporting on all encryption keys. Key Vault provides central key management, leverages tightly monitored hardware security modules (HSMs), and promotes separation of management of keys and data to help meet regulatory compliances. To learn more about Key Vault, visit the https://docs.microsoft.com/en-us/azure/key-vault/key-vault-secure-your-key-vault

Some benefits that TDE with BYOK offer: • Increased transparency and granular control with the ability to self-manage the TDE protector
• Central management of TDE encryption keys (along with other keys and secrets used in other Azure services) by hosting them in Key Vault. • Separation of keys and data management within the organization, to support Separation of Roles. • Greater trust from your own clients, since Key Vault is designed so that Microsoft does not see or extract any encryption keys. • Support for key rotation

Overview of Bring Your Own Key support Compatibility with Service-Managed TDE For those using service-managed TDE who would like to start using Key Vault, TDE will remain on during the process of switching over to a key in Key Vault. There is no downtime nor re-encryption of the database files themselves. Switching from a service-managed key to a Key Vault key requires re-encryption of the database encryption key which is a fast and online operation that usually completes within seconds.

How does TDE with BYOK work?


Portal
PowerShell
REST API
1
Azure SQL Service
SQL
2
Azure Active Directory
3
Azure Key Vault
1 The Key Vault admin grants vault access to the SQL Database server using its unique Azure Active Directory (AAD) identity.
2 The server uses its AAD identity to authenticate with AAD for access to your Key Vault.
3 The server sends get, wrap key, and unwrap key requests to the asymmetric key in Key Vault for database encryption key protection.

Authentication of the Server to the Key Vault

TDE Protector Usage When TDE is configured to use a key from Key Vault, the server sends the database encryption key (DEK) of each TDE-enabled database to Key Vault for a “wrap key” request. The key vault returns the encrypted DEK, which is stored in the user database. It is important to note that once a key is stored in Key Vault, that key never leaves Key Vault. A hardware security module (HSM)-backed key in Key Vault never leaves the HSM security boundary. The server can only send key operation requests to the TDE protector key material within Key Vault. The Key Vault administrator has the right to revoke key vault permissions for the server at any point, in which case all connections to the server are cut off.

Data Encryption & Decryption On database startup, the encrypted DEK will be sent to Key Vault to be decrypted, and then used for decryption and re-encryption of the database files in the SQL Engine process. TDE performs real-time I/O encryption and decryption of the data at the page level. Each page is decrypted when read into memory, and encrypted before being written to disk.

Considerations Key Management Responsibilities Taking over encryption key management of an application’s resources is an important responsibility. By using TDE with BYOK through Key Vault, the following are the key management tasks that are being assumed by the user: • Key backups: the Key Vault administrator is responsible for regularly taking backups of the key material through Key Vault, and (recommended) keeping a local copy of the key material in escrow • Key rotations: the TDE protectors should be rotated according to internal regulations or compliance requirements • Key rotations can be done through the TDE protector’s Key Vault • Key vault permissions: permissions for AKV are on a key vault level (not key level), and the SQL server must be configured to have access to the key vault. • Using the Key Vault access policy, the server’s permissions to the key vault can be revoked at any time. • Key deletions: keys may be dropped from AKV and the SQL server for additional safety or to meet compliance • Auditing/reporting on all encryption keys: Azure Key Vault provides logs which are easy to inject into other security information and event management (SIEM) tools. Operations Management Suite (OMS) https://docs.microsoft.com/en-us/azure/azure-monitor/insights/azure-key-vault is one example service that is already integrated.

Pricing Considerations TDE with BYOK support is a security capability that is built into the Azure SQL service, without any extra fees. However, there are related costs for using Key Vault. Key Vault operations made by the server are charged as normal operations to your vault, and follow Key Vault's [https://azure.microsoft.com/en-us/pricing/details/key-vault/](https://azure.microsoft.com/en-us/pricing/details/key-vault/). The server sends requests to Key Vault for the following events: • SQL instance restarts • Key rollovers • Every 6 hours to check if any changes have been made to the server's permissions to the key vault

Important Warnings Loss of access to keys Keep in mind that once a server no longer has access to the TDE Protector, all connections to the encrypted databases under the server will be blocked, and these databases will go offline and get dropped within 24 hours. The SQL service will check for permission and key changes every 6 hours.

Deleted keys 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. If there is an extreme case where a key is suspected to be compromised (such that a service or user had unauthorized access to the key), it's best to delete the key.

Expired keys Because the TDE Protector's availability directly impacts database availability, a key with an expiration date is not allowed to be added to a SQL server. If a key is already used as a TDE Protector for a server, and is later added an expiration date in Azure Key Vault, once the key becomes expired, the encrypted databases will no longer have access to their TDE Protector and be dropped within 24 hours.

Deleted server identities A server's access to Key Vault is managed through the server's unique Azure Active Directory (AAD) identity. Therefore, if the server's identity is deleted from AAD, the server loses access to its key vaults. As a result, all connections to the encrypted databases under the server will be blocked, and these databases will go offline and get dropped within 24 hours.

Limitations Key vaults being used for TDE on an Azure SQL Database server must be in the same AAD tenant as the SQL Database server. Cross-tenant key vault and server interactions are not supported.

A resource that is being encrypted with a key from Key Vault cannot be moved across Azure subscriptions. Moving the resource across subscriptions breaks the necessary Key Vault access controls which made TDE with BYOK possible. If any resources must be moved to another subscription, please change the TDE mode from BYOK to Service managed.

Unique TDE Protectors for a database or data warehouse are not supported. The TDE Protector is set on the server-level, and inherited by all resources under the server.

Guidelines for Managing Encrypted Databases High Availability & Disaster Recovery There are two ways geo-replication could be set up for two or more Azure SQL servers using Key Vault: • Each server has access to a separate Key Vault (ideally each within their own Azure region) a. Recommended configuration, since each server has its own copy of the TDE protector for the encrypted geo-replicated databases. If one of the server's Azure region goes offline, the other servers can continue to access the geo-replicated databases.
• All servers share the same Key Vault b. Easier to setup, but if the Azure region where the Key Vault is located goes offline, all servers will not be able to read the encrypted geo-replicated databases or their own encrypted databases.

• Use the following PowerShell cmdlet to add each server's Key Vault key to the other servers in the geo-link. a. The combined character length of the key vault name and the key name cannot exceed 94 characters. b.

Example KeyId from Key Vault: https://contosokeyvault.vault.azure.net/keys/Key1/1a1a2b2b3c3c4d4d5e5e6f6f7g7g8h8h

```
<# Include the version guid in the KeyId #>
Add-AzureRmSqlServerKeyVaultKey -KeyId <KeyVaultKeyId> -ServerName
<LogicalServerName> -ResourceGroup <SQLDatabaseResourceGroupName>
```

1. Follow the steps in the https://docs.microsoft.com/en-us/azure/sql-database/sql-database-auto-failover-group?tabs=azure-powershell to configure Active Geo-Replication with these servers, and trigger a failover.

Backup & Restore Once a database is encrypted with TDE using a key from Key Vault, any generated backups will also be encrypted with the same TDE Protector.

To restore a backup encrypted with a TDE Protector from Key Vault, make sure that the key material is still in the original vault under the original key name. When the TDE Protector is changed for a database, old backups of the database are not updated to use the latest TDE Protector. Therefore as a best practice, keep all old versions of the TDE Protector in Key Vault, so database backups can be restored.

To learn more about backup recovery for SQL Database, visit https://docs.microsoft.com/en-us/azure/sql-database/sql-database-recovery-using-backups

## 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)

Best Practices Key Management To ensure quick key recovery and access your data outside of Azure, we recommend the following best practice:

1. Create your encryption key locally on a local HSM device. (Make sure this is an asymmetric, RSA 2048 key so it is storable in Azure Key Vault.)
2. Import the encryption key file (.pfx, .byok, or .backup) to Azure Key Vault.
3. Before using the key in Azure Key Vault for the first time, take an Azure Key Vault key backup. Learn more about the #4884 (https://docs.microsoft.com/en-us/powershell/resourcemanager/azurerm.keyvault/v1.1.11/Backup-AzureKeyVaultKey?redirectedfrom=msdn) command.
4. Whenever any changes are made to the key (e.g. add ACLs, add tags, add key attributes), be sure to take another Azure Key Vault key backup.
5. During a key rollover, keep previous versions of the key in the key vault so older database backups can be restored. Creating the TDE encryption key locally first and importing the asymmetric key is highly

recommended for production scenarios because it allows the administrator to escrow the key in a key escrow system. If the asymmetric key is created in the vault, it cannot be escrowed because the private key can never leave the vault. Keys used to protect critical data should be escrowed. The loss of an asymmetric key will result in permanently unrecoverable data.

Pre-configuration for Replicated Databases If an encrypted database is going to be replicated to another server, make sure that the server has access to a copy of the Key Vault key material used in the other server before moving or replicating the database. We recommend the following best practice - each server should have access to a copy of the Key Vault key material used in the other server, which is stored in a separate Key Vault (ideally each within the same Azure region as the server). With this setup, each server has its own copy of the TDE Protector for the encrypted replicated databases. If one of the servers' Azure region goes offline, the other servers can continue to access the replicated databases.

## Public Doc Reference (optional)

## Internal Reference (optional)

## Classification

Root cause tree: Security/User Request/How-to/advisory

**How good have you found this content?**