SQL Server Verschlüsselung – Always Encrypted

Die Verschlüsselungsmethode **Always Encrypted** ist ein Feature, das zum Sichern von sensitiven Daten in Azure SQL und SQL Server verwendet werden kann. Es ermöglicht die Verschlüsselung innerhalb einer Client-Anwendung und erzeugt so seine klare Trennung zwischen **Besitzer** und **Verwalter** von Informationen. Der entscheidende Vorteil ist, dass Benutzer sensitive Daten außerhalb ihrer Zugriffsrechte speichern können.

Die Realisierung der Verschlüsselung zwischen Datenbank und Anwendung basiert dabei auf einem externen Treiber, der auf dem Client-Computer installiert sein muss. Dieser Treiber ver- und entschlüsselt die sensitiven Daten der Client-Anwendung automatisch. Innerhalb dieses automatischen Prozesses werden Daten, die sich in sensitiv gekennzeichneten Spalten befinden, erkannt und Abfragen gegebenenfalls angepasst (die Semantik der Anwendung bleibt erhalten).

Typische Szenarien

Typische Szenarien für die Verwendung von **Always Encrypted**:

1. Der Kunde/Benutzer besitzt eine lokale (on-premise) Client-Anwendung und die sensitiven Daten werden innerhalb einer Datenbank in Azure verwaltet. Zugehörige Schlüssel werden ebenfalls lokal gespeichert.
2. Der Kunde/Benutzer besitzt eine Client-Anwendung, die in Azure verwaltet wird. Die sensitiven Daten werden ebenfalls innerhalb einer Datenbank in Azure gespeichert.

Funktionsweise

Die **Always Encrypted** Verschlüsselung wird für individuelle Spalten von Tabellen innerhalb einer Datenbank konfiguriert. Dafür muss sowohl der für die Verschlüsselung verwendete Algorithmus, als auch kryptographische Schlüssel angegeben werden. Bei diesem Schlüssel wird zwischen zwei verschieden Arten unterschieden:

→

Kathegorie
SQL Server

Aaron
Priesterroth

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz

info@madafa.de
madafa.de

+49 6131 3331612

1. Der sog. **column encryption key**, der für die Verschlüsselung der Daten einer Spalte dient.
2. Der sog. **column master key**, der für die Sicherung der **column encryption keys** verwendet wird. Mit seiner Hilfe werden ein oder mehrere **column encryption keys** verschlüsselt.

Um auf Daten, die sich innerhalb einer verschlüsselten Spalte befinden, zugreifen zu können, muss der zuvor erwähnte Treiber zur Verfügung stehen. Wird beispielsweise eine Abfrage ausgeführt, entscheidet der Treiber zusammen mit der Datenbank-Engine, welche der abgefragten Spalten verschlüsselt ist. Dafür müssen also folgende Informationen für jeden Parameter der Abfrage in Erfahrung gebracht werden:

- Verschlüsselungsalgorithmus
- Verschlüsselter Wert des **column encryption key**
- Der Zugehörige **column master key**
- Die Parameter-Zielwerte

Nachdem der Treiber den **column master key** lokalisiert hat, kann dieser verwendet werden, um den **column encryption key** zu entschlüsseln. Der entschlüsselte **column encryption key** kann im Umkehrschluss nun zur Entschlüsselung des Abfrage-Parameters verwendet werden. Der entschlüsselte **column encryption key** wird dabei zwischengespeichert, um den Overhead in evtl. folgenden Abfragen zu verringern.

Basierend auf dem zuvor beschriebenen Ablauf kann der Treiber nun die verschlüsselten Abfrage-Parameter durch Klartext-Werte ersetzen und die resultierende Abfrage an den Server weiterleiten.

Der Server verarbeitet die Abfrage und fügt der Ausgabe zusätzliche Meta-Informationen (verwendeter Algorithmus, Schlüssel, etc.) an. Der Treiber kann basierend auf dieser Rückgabe die verschlüsselten Werte entschlüsseln und diese der Client-Anwendung bereitstellen.

Bemerkungen

Always Encrypted benötigt die Unterstützung des Treibers und ist somit Client-basiert. Das bedeutet, dass Funktionen, die Server-seitig umgesetzt werden, evtl. nicht funktionieren bzw. verwendet werden können, wenn **Always Encrypted** im Einsatz ist.

Deterministische-/Randomisierte-Verschlüsselung

- **Deterministische Verschlüsselung** basiert auf bijektiven Funktionen (für die selbe Eingabe wird immer die gleiche Ausgabe erzeugt). Sie ermöglicht Punkt-Abfragen, Gleichtheits-Joins, Gruppierung und Indizierung auf verschlüsselten Spalten. Gleichzeitig besteht jedoch die Gefahr, dass unautorisierte Benutzer Muster in den verschlüsselten Spalten erkennen und so an Informationen gelangen können.
  - → Für Spalten verwenden, die in einer **Group-By-** oder **Where-**Klausel auftauchen.
- **Randomisierte Verschlüsselung** basiert auf unvorhersehbaren Ergebnissen von randomisierten Funktionen. Sie bietet erhöhte Sicherheit bzgl. Brute-Force-Attacken, verhindert jedoch die Verwendung von Punkt-Abfragen, Gleichtheits-Joins, Gruppierung oder Indizierung.
  - → Für Spalten verwenden, die sensitive Daten enthalten und nicht in einer **Group-By-** oder **Where-**Klausel auftauchen.

Konfiguration von Always Encrypted

Die Konfiguration von **Always Encrypted** läuft nach dem folgenden, simplen Schema ab:

1. Erzeugen der **Always Encrypted** Schlüssel
2. Erzeugen der Schlüssel Meta-Informationen
3. Konfigurieren der Verschlüsselungs-Eigenschaften der ausgewählten Tabellen-Spalten
4. (Optional) Verschlüsseln der Daten einer Tabellen-Spalte, die bereits existieren

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

Erste Schritte

Um **Always Encrypted** zu benutzen, kann der sogenannte **Always Encrypted Wizard** verwendet werden. Schlüssel und Konfiguration werden darüber bereitgestellt.

Anhand des folgenden Beispiels soll der Umgang mit **Always Encrypted** und dem **Always Encrypted Wizard** genauer verdeutlicht werden:

1. Stellen Sie eine Verbindung zu einer existierenden Datenbank mit Hilfe des SQL Server Management Studio (SSMS) her.
2. Mit einem Rechts-Klick auf die gewünschte Datenbank öffnen Sie den Always Encrypted Wizard über **Tasks → Encrypt**.
3. Wählen Sie die zu verschlüsselnden Spalten innerhalb der **Column Selection** aus.
4. Für jede ausgewählte Spalte muss der gewünschte Verschlüsselungs-Typ (deterministisch oder randomisiert) ausgewählt werden.
5. Für jede ausgewählte Spalte muss der **Encryption key** ausgewählt werden. Alternativ besteht die Möglichkeit mit Auswahl von **auto-generation** einen Schlüssel automatisch generieren zu lassen.
6. Auf der Seite bezüglich der **Master key configuration** muss ein Ort zum Ablegen der neuen Schlüssel angegeben werden. Zusätzlich muss ein Master-Schlüssel angegeben werden.
7. Auf der Seite bezüglich der **Validation** kann ausgewählt werden, ob das resultierende Skript sofort ausgeführt, oder ein PowerShell Skript erzeugt werden soll.
8. Auf der Seite **Summary** kann die getätigte Auswahl überprüft werden. Mit einem Klick auf Finish wird die Auswahl bestätigt.

always, always encrypted, column, column encryption, decryption, encrypted, enryption, entschlüsseln, key, schlüssel, sql server, ssms, verschlüsseln, wizard

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612