Les différents modes de chiffrement SQL Server

ETUDE CHIFFREMENT SQL SERVER

Oceanet Technology

Oceanet Technology

Oceanet Technology, qui a financé cet ouvrage, est spécialisé dans l'infogérance et la sécurisation des données depuis 1996. La société exploite à la fois les Clouds publics et les Clouds privés (avec ses 6 datacenters en propre). Oceanet Technology bénéficie de la certification ISO27001.

Avec 25 ans d'expérience, plus de 2000 clients et des milliers de projets, le Groupe Oceanet Technology se donne pour mission d'accompagner les entreprises dans leur transformation digitale, d'améliorer continuellement leur projet, et d'assurer la sécurité informatique de leurs systèmes et réseaux.

Migration partielle ou complète vers le Cloud, architecture, conseil en cybersécurité, hébergement de données sensibles ou de santé… les équipes du Groupe sont là pour vous accompagner sur vos projets, et répondre à vos enjeux.

Les entités du groupe

Oceanet Technology

2 impasse Joséphine Baker
44800 Saint-Herblain - France
contact@oceanet-technology.com
+33 2 28 03 78 78
oceanet-technology.com

NBS System

23-25 rue de Berri 75008 Paris - France
contact@nbs-system.com
+33 1 58 56 60 80
nbs-system.com

Net4All

Chemin de dévent, 7 - 1024 Ecublens VD - Suisse
net4all@net4all.ch
+41 21 625 69 90
net4all.ch

SOMMAIRE

INTRODUCTION

Chaque année, **978 millions de personnes, dans le monde, sont concernées par une cyberattaque,** d'après le Ministère de l'intérieur. Cependant, les cibles privilégiées sont les entreprises. Il est donc important de renforcer la stratégie de sécurité d'entreprises. Le chiffrement des données est une des voies pour l'améliorer.

1 Qu'est ce le chiffrement des données

Cette fonctionnalité permet de convertir vos données d'un format lisible à un format chiffré et donc sécurisé. Elles pourront être lues uniquement avec une clé ou un mot de passe. C'est le moyen le plus simple et le plus efficace de protéger la confidentialité des bases de données quand elles sont stockées dans un système d'information et transmises via internet ou d'autre réseaux.

2 Le chiffrement SQL Server

Dans le cadre d'un projet, nous avons pu expliquer les différents types de chiffrement SQL Server sur certains applicatifs. Durant cette étude, différents enjeux ont été réalisés :

- Chiffrer les données ;
- Etudier l'impact sur les performances ;
- Etudier l'impact sur la partie applicative.

Cependant, nous devons, en aucun cas, oublier certains points de vigilances, à savoir **la taille des bases de données** ainsi que **la sauvegarde et la restauration des données.**

Pour répondre au mieux à ces exigences, tout en considérant que SQL Server propose plusieurs modes de chiffrement pour ses bases de données, nous avons détaillé et résumé les procédures de chacune des méthodes, ainsi que les éventuels impacts sur les ressources du serveur ou l'application source.

3 Différentes méthodes de chiffrement

Il existe 5 méthodes de chiffrement pour SQL Server 2019 (version 15.x) :

- Chiffrement du backup
- Chiffrement SSL au niveau de la connexion SQL Server
- Chiffrement transparent des données (TDE : Transparent Data Encryption)
- Chiffrement au niveau d'une colonne de table
- Chiffrement Global (Always Encrypted)

Il existe également une méthode qui n'est pas du chiffrement mais du masquage : Dynamic Data Masking.

Compatibilité des méthodes selon les éditions SQL Server 2019

Fonctionnalité	SQL Entreprise	SQL Standard	SQL Web
Sauvegarde chiffrée	OUI	OUI	NON en Backup OUI en Restore
Chiffrement TLS de la connexion	OUI	OUI	OUI
Chiffrement transparent de base de données (TDE)	OUI	OUI	OUI
Chiffrement de colonnes de tables	OUI	OUI	OUI
Toujours chiffré (Always Encrypted)	OUI	OUI	OUI
Masquage dynamique des données	OUI	OUI	OUI

SAUVEGARDE CHIFFREE

La sauvegarde chiffrée permet de sécuriser les données sauvegardées. Son utilisation permet de ne restaurer les sauvegardes chiffrées que sur un serveur ou une instance disposant du certificat ou de la clé de chiffrement asymétrique utilisée.

Mise en place du backup :

- Création de la Master Key dans « master ».
- Création d'un certificat protégé par la Master Key.
- Sauvegarde du certificat et de la clé privée (hors site).
- S'assurer que le compte SQL allant effectuer le Backup soit « db_backupoperator »
- Mise en place des Plans de maintenance de sauvegarde et/ou Veeam.

Restauration :

- Si serveur différent : Création d'une nouvelle Master Key dans « master ».
- Si serveur différent : Restauration du certificat et de la clé privée (fichiers hors site).
- S'assurer que le compte SQL allant effectuer le Restore possède l'autorisation de voir le certificat présent dans « master ».

Tips T-SQL :

Création de Master Key :

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Master-KeyPassword';
GO
```

Restauration d'un certificat et de la clé privée :

```
CREATE CERTIFICATE [Backup_Certificate] AUTHORIZATION [dbo]
FROM FILE = 'D:\Backup-Keys\Backup_Certificate.cer'
WITH PRIVATE KEY (DECRYPTION BY PASSWORD = 'myDecrypt-Password', FILE = 'D:\Backup-Keys\Backup_Certificate.key');
GO
```

Création d'un certificat et de la clé privée et sauvegarde :

```
CREATE CERTIFICATE Backup_Certificate WITH Subject = 'myS-SLSubject';
GO
BACKUP CERTIFICATE Backup_Certificate TO FILE = 'D:\
Backup-Keys\Backup_Certificate.cer'
WITH PRIVATE KEY
( FILE = 'D:\Backup-Keys\Backup_Certificate.key',
ENCRYPTION BY PASSWORD = 'myBackupPasword');
GO
```

Lancer un Backup Chiffré :

```
BACKUP DATABASE myDB
TO DISK = 'D:\Backup-Keys\myDB_BackupEncrypted.bak'
WITH COMPRESSION, STATS = 10, ENCRYPTION (ALGORITHM = AES_256,SERVER CERTIFICATE = 'mySSLSubject');
```

CHIFFREMENT TLS DE LA CONNEXION

Afin d'augmenter la sécurité des applicatifs, nous pouvons communiquer avec le serveur SQL en chiffrant les communications à l'aide de TLS. Il faut installer un certificat SSL qui a été attribué à partir d'une autorité de certification publique sur le serveur (à l'aide de la MMC par exemple).

Afin d'utiliser TLS 1.2 sur le serveur SQL, vérifier qu'une mise à jour n'est pas nécessaire : **https://support.microsoft.com/fr-fr/topic/prise-en-charge-de-tls-1-2-pour-microsoft-sql-server-e4472ef8-90a9-13c1-e4d8-44aad198cdbe**

- **Configurer SQL Server :**

A l'aide du « Gestionnaire de configuration SQL Server »

Le chiffrement TDE a été introduit avec SQL Server 2008 Entreprise. Il est pour la première fois disponible avec SQL 2019 en version Standard.

CHIFFREMENT TRANSPARENT TDE

> Son principal but est de protéger les données en chiffrant les fichiers physiques. Les fichiers de données MDF et le journal LDF sont concernés par ce chiffrement.

La technologie a été créée afin que le processus de chiffrement soit complètement transparent pour les applications qui accèdent à la base de données. Le processus est de chiffrer les pages avant de les écrire dans les fichiers. A la lecture les pages sont déchiffrées puis placées en RAM.

Remarques

Si une base de données utilise TDE, alors la base de données TEMPDB sera également chiffrée. Il est à noter que le chiffrement / déchiffrement s'effectuant par le CPU du serveur SQL, les données transitant sur le réseau ne sont donc pas chiffrées. Il est donc possible de mixer cette solution avec la solution de connexion TLS.

Afin de mettre en place le chiffrement TDE, il convient de :

- Créer la Master Key dans « master ».
- Créer un certificat protégé par la Master Key.
- Créer une clé de chiffrement dans la base de données (Database Encryption Key : DEK)
- Activer le chiffrement dans les options de la base de données
- Sauvegarde du Certificat et de la Master Key (hors site).
- S'assurer que le compte SQL allant effectuer le Backup soit « db_backupoperator »

CHIFFREMENT TRASPARENT TDE

Tips T-SQL :

Création de Master Key :

```
USE MASTER;
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Master-KeyPassword';
GO
```

Création du certificat :

```
USE MASTER;
CREATE CERTIFICATE myCertificate WITH SUBJECT = 'My Certificate', START_DATE = '01/01/2021', EXPIRY_DATE = '31/12/2029';
GO
```

Création de la clé de chiffrement dans la base client :

```
USE MADB;
CREATE DATABASE ENCRYPTION KEY WITH ALGORITHM = AES_128 ENCRYPTION BY SERVER CERTIFICATE myCertificate;
GO
```

Activation du chiffrement TDE sur la base client :

```
ALTER DATABASE MADB SET ENCRYPTION ON;
GO
```

Lors de l'activation ou la désactivation de TDE sur la base de données, la mise en action est directe.

> Ce qui veut dire qu'un journal non tronqué contiendra ET des enregistrements non chiffrés ET des enregistrements chiffrés.

Lors de la désactivation de TDE et de la suppression de la clé de chiffrement ainsi que du certificat, il faut penser à tronquer le journal de transactions sous peine de voir la base de données refuser de se monter après un reboot serveur. Concernant le backup, le serveur devant recevoir une restauration de la base de données doit posséder le certificat de chiffrement présent dans la base. Vous n'aurez aucun souci avec Veeam si c'est respecté.

CHIFFREMENT DE COLONNES DE TABLES

Afin de protéger des données, **une base de données n'a pas forcément besoin d'être chiffrée complètement.**

1. On peut choisir de ne vouloir chiffrer que certaines données, comme une adresse email, un numéro de compte, un numéro de carte de crédit, etc.

   > Dans ce cas, SQL Server propose la fonctionnalité de chiffrement de colonnes.

2. Il est également possible avec ce procédé de chiffrer plusieurs colonnes avec des clés différentes afin de proposer par exemple une vue « Documents confidentiels » et une autre « Documents top Secret ».

   > Dans ce cas on pourra utiliser des utilisateurs SQL qui ont accès à 1 seule des clés de chiffrement ou aux 2. Selon l'utilisateur, on n'aura accès à aucune donnée, à une partie ou à la totalité.

Les données sont chiffrées « au repos ». Lors d'opération SELECT avec déchiffrement, les données en RAM ne sont pas chiffrées.

Mise en place :

- Créer une clé de chiffrement Master Key dans la base client
- Créer un certificat autosigné pour le serveur SQL
- Configurer une clé de chiffrement symétrique
- Chiffrer la ou les colonnes
- Tester la solution

CHIFFREMENT DE COLONNES DE TABLES

Tips T-SQL :

Création de Master Key :

```
USE MADB;
GO
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'MasterKeyPassword';
GO
```

Création du certificat :

```
USE MADB
CREATE CERTIFICATE myCertificate WITH SUBJECT = 'My Certificate',
START_DATE = '01/01/2021', EXPIRY_DATE = '31/12/2029';
GO
```

Création d'une clé de chiffrement symétrique :

```
USE MADB;
GO
CREATE SYMMETRIC KEY mySymKey WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE myCertificate;
GO
```

Ajout d'une colonne chiffrée :

```
USE MADB;
GO
ALTER TABLE maTable ADD myCol_Encrypted varbinary(MAX);
GO
```

Tips T-SQL :

Mise à jour d'une table :

```
USE MADB;
GO
OPEN SYMMETRIC KEY mySymKey DECRYPTION BY CERTIFICATE myCertificate;
UPDATE maTable SET myCol_Encrypted = EncryptByKey (Key_GUID('mySymKey'), myCol) FROM maTable;
GO
CLOSE SYMMETRIC KEY mySymKey;
GO
```

Droits nécessaire pour déchiffrer une colonne :

```
USE MADB;
GRANT VIEW DEFINITION ON SYMMETRIC KEY::mySymKey TO myUser;
GO
```

Lecture d'une colonne chiffrée dans une application :

```
USE MADB;
GO
OPEN SYMMETRIC KEY mySymKey DECRYPTION BY CERTIFICATE myCertificate;
SELECT myID, myCol_Encrypted AS 'Encrypted Data', CONVERT(varchar, DecryptByKey(myCol_Encrypted)) AS 'Decrypted Data' FROM maTable;
GO
```

Concernant le backup, le serveur devant recevoir une restauration de la base de données **doit posséder le certificat de chiffrement présent dans la base.** Vous n'aurez aucun souci avec Veeam si c'est respecté.

CHIFFREMENT « TOUJOURS CHIFFRE »

Le chiffrement avec TDE possède plusieurs défauts :

- Seules les données écrites sont chiffrées (cela protège les données au repos)
- Cela nécessite de chiffrer toute la base de données
- Toutes les données sont chiffrées de la même manière

La sortie du chiffrement « Toujours chiffré » corrige ces défauts.

- Les données sont chiffrées au repos, en mouvement et en mémoire
- On peut choisir de chiffrer une seule colonne
- TEMPDB n'est pas chiffré
- 2 choix de chiffrement : Déterministe (pour les indexes) et Random

Par contre il est nécessaire de mettre à jour le pilote de base de données utilisé par le logiciel client afin qu'il supporte cette nouvelle fonctionnalité. Le chiffrement / déchiffrement est automatiquement réalisé par le pilote de base de données.

CHIFFREMENT « TOUJOURS CHIFFRE »

Points de vigilance en cas de l'utilisation de ce chiffrement :

- Les seules opérations que le Moteur de base de données peut effectuer sur les données chiffrées sont les comparaisons d'égalité (disponibles seulement avec le chiffrement déterministe).
- La taille des colonnes en base de données grandit (généralement le double).
- Attention aux types de données à chiffrer (datetime à transformer en datetime2).
- Attention à MAX pour varchar(MAX) et nvarchar(MAX) qui provoquent des erreurs de types lors du déchiffrement.
- Les Tables optimisées mémoire (in-memory OLTP) ne sont pas supportées.
- Les colonnes chiffrées de manière déterministe doivent être en Collation BIN2.
- On ne peut pas chiffrer une colonne IDENTITY.
- Les types de données : XML, IMAGE, NTEXT, TEXT, HIERARCHYID, SQL_VARIANT, GEOGRAPHY, GEOMETRY, ROWVERSION, ainsi que les types définis par les utilisateurs (user-defined types) ne sont pas supportés.
- Nécessite au miniumum dotNet 4.6
- Full Text Search non supporté

L'utilisation de ce chiffrement ne permet plus d'effectuer des tests entre les champs dans les requêtes (>=, <=, LIKE, etc.). Il faut passer par des vues. **Microsoft a mis en place les « Enclaves Sécurisées »** (Secure Enclaves) qui permettent de réaliser de nouveaux certains tests (LIKE, COUNT, etc.) sur les champs.

> Une enclave sécurisée est une région protégée de la mémoire au sein du processus Moteur de base de données. L'enclave sécurisée apparaît comme une zone opaque pour le reste de Moteur de base de données et les autres processus de l'ordinateur d'hébergement.

Il n'existe aucun moyen d'afficher les données ou le code à l'intérieur de l'enclave depuis l'extérieur, même avec un débogueur.

CHIFFREMENT « TOUJOURS CHIFFRE »

Lors de l'analyse d'une instruction Transact-SQL envoyée par une application, le Moteur de base de données détermine si l'instruction contient des opérations sur des données chiffrées qui nécessitent l'utilisation de l'enclave sécurisée.

L'enclave sécurisée à l'intérieur du Moteur de base de données peut accéder à des données sensibles et aux clés de chiffrement de colonnes en clair. Avant d'envoyer une instruction impliquant des calculs d'enclave au Moteur de base de données, le pilote de base de données client à l'intérieur de l'application doit vérifier que l'enclave sécurisée est une véritable enclave et que le code s'exécutant dans l'enclave sécurisée est la véritable bibliothèque Always Encrypted.

Le processus de vérification de l'enclave, **appelé attestation d'enclave**, implique généralement un pilote client au sein de l'application et une communication entre Moteur de base de données et un service d'attestation externe.

> Le processus d'attestation pour les enclaves sécurisées dans SQL Server 2019 (15.x) est l'attestation de runtime Windows Defender System Guard, qui nécessite le service Guardian hôte (SGH) comme service d'attestation.

La mise en place est complexe et on s'aidera de l'interface graphique de SQL management Studio.

- Configurer avec l'assistant :
  Configurer le chiffrement de colonne à l'aide de l'Assistant Always Encrypted - SQL Server | Microsoft Docs
- Développer des applications utilisant Always Encrypted :
  Développer des applications à l'aide d'Always Encrypted - SQL Server | Microsoft Docs

LE MASQUAGE DYNAMIQUE DES DONNEES

Introduites avec la version 2016 de SQL Server, ces fonctionnalités permettent de se conformer à la nouvelle réglementation européenne (RGPD) entrée en vigueur le 25 Mai 2018.

> Le principe général repose non pas sur un chiffrement des données, mais sur le fait de masquer ou de filtrer celles-ci aux utilisateurs non autorisés.

Le "Dynamic Data Masking" (DDM) utilise des fonctions de masquage pré-construites (default, email, random, partial) pour masquer certaines colonnes contenant des données sensibles :

Dynamic Data Masking (DDM) - Functions -SQL Server

Function	Description	Data Type Support	Example – MASKED WITH	Before masking	After masking
Default	Full data masking	All data types	(FUNCTION='default()')	Ellora Caves	XXXX
Partial	Partial data masking	String data type	(FUNCTION='partial(2, « X XXXX », 0)')	Ellora Caves	ElXXXXXXX
Email	Specific to Email Ids	String data type	(FUNCTION='email()')	Ellora.caves@au.co.in	EXXX@XXX.com
Random	Random numbers between range	Numeric data types	(FUNCTION=random(1,9)')	85000	9

DDM permet de Sécuriser les données stockées existantes sans modifier le code d'application et les requêtes. Exemple :

Dynamic Data Masking (DDM) - SQL Server

Type	Data
Name	Ketan Wagh
Date of Birth	28-Oct-195
Email Id	Ketan.wagh@rich.co.in
Identify Number	8ZVLPE1258Q
Phone	98564534213
Salary	85000
Credit Card	9764-3451-0916-1047

Type	Data
Name	XXXXXXXXXXgh
Date of Birth	1900-01-01
Email Id	xxxxxx@xxxxx.com
Identify Number	XXXXXXX
Phone	XXXXXXX
Salary	9
Credit Card	97XXXXXXX

LE MASQUAGE DYNAMIQUE DES DONNEES

Tips T-SQL :

Masquage d'une colonne "salaire" : affichage d'un entier entre 1 et 3

```
USE MADB;
GO
ALTER TABLE maTable ALTER COLUMN salaire int MASKED WITH (FUNC-
TION='random(1,3)');
GO
```

Masquage d'une colonne "email" : affichage XXX@XXX.COM

```
USE MADB;
GO
ALTER TABLE maTable ALTER COLUMN email nvarchar(50) MASKED WITH
(FUNCTION='Email()');
GO
```

Masquage d'une colonne "CardID" : affichage des 2 premiers caractères

```
USE MADB;
GO
ALTER TABLE maTable ALTER COLUMN cardID varchar(20) MASKED WITH
(FUNCTION='partial(2,"XXXXXX",0)');
GO
```

Ajout du droit de visualisation des données à un utilisateur

```
USE MADB;
GRANT UNMASK TO myUser;
GO
```

Retrait du droit de visualisation des données à un utilisateur

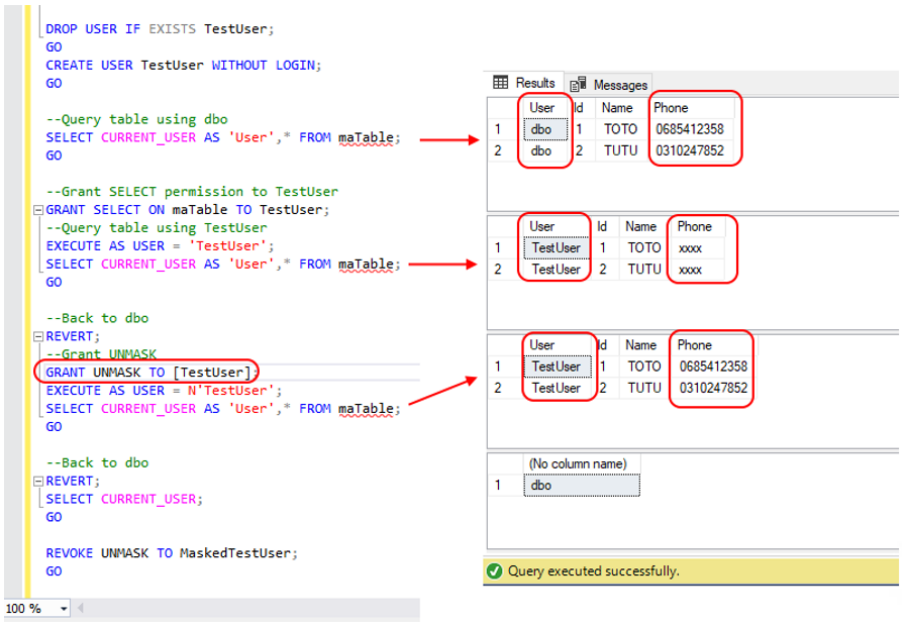
```
USE MADB;
REVOKE UNMASK TO myUser;
GO
```

Tips T-SQL :

Création d'une Table avec 1 champ "Téléphone" masqué

```
USE MADB;
DROP TABLE IF EXISTS maTable;
GO
CREATE TABLE maTable
(
Id          INT             IDENTITY(1,1)
,Name       varchar(255) NOT NULL
,Phone      varchar(10)  MASKED WITH (FUNCTION = 'default()') NOT
NULL
);
GO
```

Exemple



> Concernant le backup, il n'y a rien de particulier. Aucun souci avec Veeam.

LES CONSEILS D'OCEANET TECHNOLOGY

> Tout d'abord, il est des choix qui n'en sont pas. Par exemple, si le client final vous oblige à utiliser une solution donnée. Dans ce cas le choix est vite fait...

Mais pour le reste, nous vous conseillons au final de :

- Utiliser le masquage de données si le chiffrement n'est pas obligatoire. Il n'y a que peu de modifications (voir aucune) à effectuer dans le code de l'applicatif. Le résultat affiché ne dépendra que de l'utilisateur utilisé pour exécuter la requête de sélection.

- Utiliser le chiffrement de colonnes si le chiffrement est obligatoire pour vous et que le code est facilement modifiable. Vous gagnerez en performance à n'avoir que quelques colonnes à déchiffrer.

- Essayer d'utiliser conjointement le masquage de données et le chiffrement de colonnes.

- Utiliser TDE si le chiffrement des données est obligatoire et si l'applicatif n'est pas modifiable facilement.

- Toujours utiliser TLS (si possible 1.2) afin de chiffrer la connexion.

Tableau des plus et des moins :

Chiffrement	Plus	Moins
Backup	- Rien à faire sur le code - Base chiffrée sur le backup	- Chiffré uniquement sur le backup
TDE	- Rien à faire sur le code - Base chiffrée sur le backup - Base chiffrée sur le serveur	- TEMPDB chiffrée - Très consommateur de CPU
Colonne	- Base chiffrée sur le serveur	- Modification de code
Amways Encrypted	- Chiffré de bout en bout	- Pilote dédié - Grosses adaptations de code - Très fortes limitations - Complexe à mettre en place
Masquage	- Rien à faire sur le code à part gestion des users - Rapide	- Pas un vrai chiffrement
Connexion	- Rien à faire sur le code	- Chiffré uniquement sur la connexion

AUTORISATION RELATIVES A LA PROPRIETE REPRIDUCTION, DIFFUSION ET PUBLICATION

Oceanet Technology

AVIS DE NON-RESPONSABILITE

Le Groupe Oceanet Technology décline toute responsabilité découlant de l'utilisation d'informations ou de données fournies dans ce livre blanc.

Le Groupe Oceanet Technology et ses collaborateurs ne sauraient être tenus pour responsables d'une quelconque conséquence - financière ou autre - résultant de l'utilisation d'informations ou de données fournies dans le livre blanc, notamment de l'utilisation inappropriée, incorrecte ou frauduleuse de ces informations ou données.

La lecture du livre blanc implique automatiquement la pleine acceptation de la présente clause de nonresponsabilité. Le Groupe Oceanet Technology décline toute responsabilité quant au contenu d'autres médias qui comporteraient un lien vers ce livre blanc ou une référence à celui-ci.

REMERCIEMENT

Le Groupe Oceanet Technology remercie tous les collaborateurs qui ont participé à ce livre blanc de par leurs idées, recherches, rédactions, relectures, repasses graphiques et organisation du projet en général.

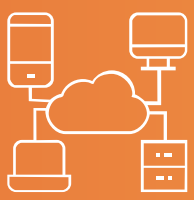
En particulier, un grand merci à Ivan Patillon, d'Oceanet Technology pour la rédaction de ce livre blanc, à Léna Kneusé pour la mise en page et la commnication autour de ce livre blanc et Carine Guillemet pour la supervision et relecture.

Oceanet Technology

ANNEXES

1 PERFORMANCES DU TDE

2 PERFORMANCE DU CHIFFREMENT DE COLONNE



1 PERFORMANCES DU TDE

Les tests suivants ont été réalisés sur une base de données de 3Go. Avant chaque commande le cache a été vidé (DBCC DROPCLEANBUFFERS) car nous savons que le cache n'est pas chiffré. Chaque commande a été lancée 3 fois et c'est la moyenne qui a été placée dans les tableaux résultats.

· Backup

	Avec TDE				Sans TDE			
	CPU (ms)	READS	WRITES	Durée (ms)	CPU (ms)	READS	WRITES	Durée (ms)
Avec compression	230	320	14	57000	200	300	13	51000
Sans compression	200	320	14	70000	180	300	19	55000

On remarque qu'avec TDE la durée de backup est de 10 à 20% plus longue.
On remarque également que le CPU est plus utilisé avec TDE : hausse de 15%.
C'est une hausse acceptable.

1 PERFORMANCES DU TDE

· Select, Insert, Update, Delete

	Avec TDE				Sans TDE			
	CPU (ms)	READS	WRITES	Durée (ms)	CPU (ms)	READS	WRITES	Durée (ms)
SELECT	7500	122000	20	26000000	3000	120000	1	23500000
INSERT	125	800	16	1510000	32	800	16	1250000
UPDATE	500	2200	30	1380000	110	2200	25	1080000
DELETE	500	2400	30	1270000	60	2500	30	710000

On remarque qu'avec TDE le temps d'exécution des requêtes augmente d'environ 10% ce qui est raisonnable. Le plus gros impact se situe au niveau du CPU. On peut noter une hausse de 60% de l'utilisation CPU sur certaines requêtes. En regardant plus en détail, les plus coûteuses sont les INSERT, UPDATE, DELETE.

· Ré-indexation

	CPU (ms)	READS	WRITES	Durée (ms)
Avec TDE	15000	350000	46000	22000
Sans TDE	7400	350000	46200	17000

Concernant la ré-indexation, les résultats sont assez proches, bien que légèrement plus lents avec le TDE. C'est principalement le CPU qui augmente : +50%

2 PERFORMANCE DU CHIFFREMENT DE COLONNE

Nous allons utiliser une base de données avec une table contenant de 1 à 10 millions d'enregistrements. Une colonne contenant un numéro fictif de carte bancaire à 16 chiffres sera chiffrée. Le temps de SELECT TOP x CustomerID sera mesuré avec x de 1 à 10 millions.

Création de la table :

```
USE maDB ;
CREATE TABLE dbo.tblCustomerData
(
  CustomerID  int identity(1,1) NOT NULL,
  CreditCardNumberPlainText varchar(16) NOT NULL,
  CreditCardNumberEncrypted varbinary(MAX) NULL
) ;
GO
```

Remplissage de la table : (exemple ci-dessous pour 10 millions)

```
declare @i int;
set @i=0;
begin transaction
  while @i<10000000
  begin
    INSERT INTO dbo.tblCustomerData with (tablock)
    VALUES (cast((1000000000000000+@i) as varchar(16)),NULL)
    set @i=@i+1
  end
commit
GO
```

2 PERFORMANCE DU CHIFFREMENT DE COLONNE

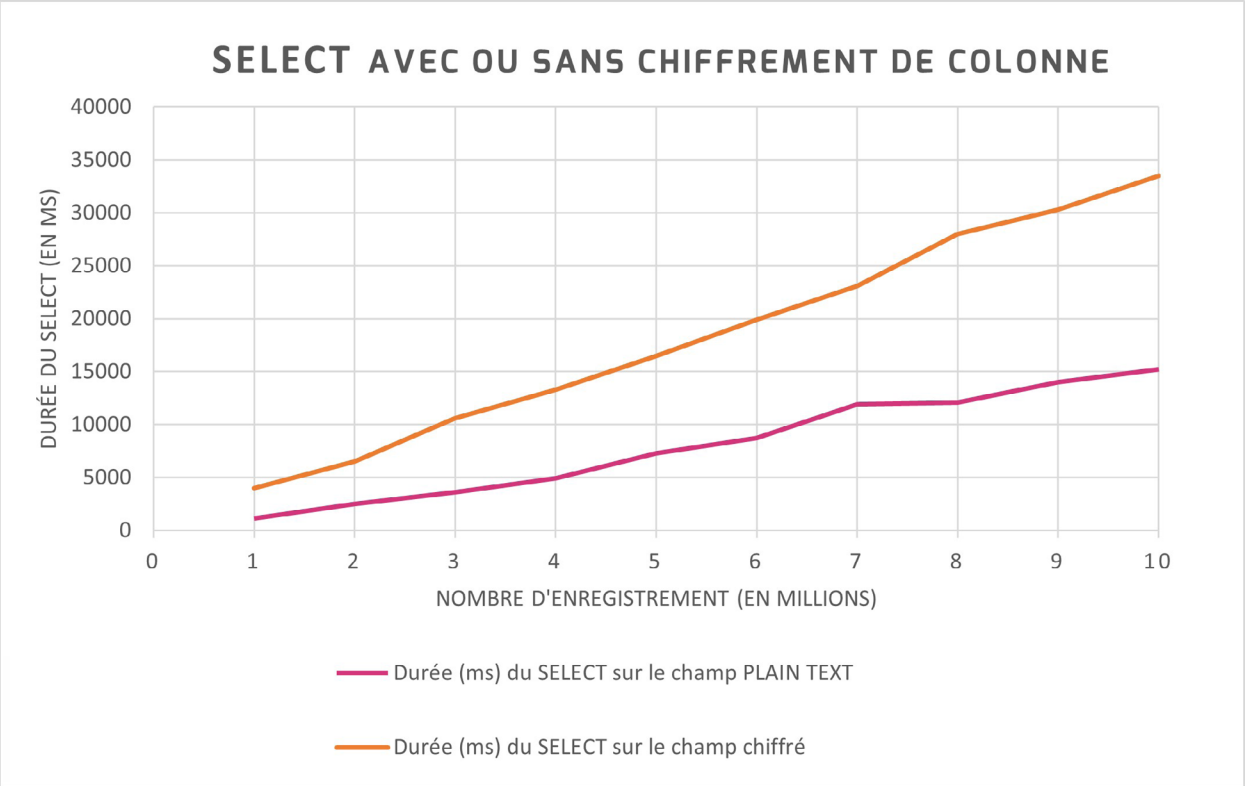
Chiffrement de la colonne CreditCardNumberEncrypted :

```
USE master ;
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'abcdefghijklmnopqrstu-
vwxyz0123456789';
GO
CREATE CERTIFICATE Certificate1
WITH SUBJECT = 'Encrypt Credit Card Number';
GO
USE maDB;
GO
CREATE SYMMETRIC KEY SymmetricKey1
WITH ALGORITHM = AES_128
ENCRYPTION BY CERTIFICATE Certificate1;
GO
OPEN SYMMETRIC KEY SymmetricKey1
DECRYPTION BY CERTIFICATE Certificate1;
GO
UPDATE tblCustomerData
SET CreditCardNumberEncrypted=
   EncryptByKey(Key_GUID('SymmetricKey1'),CreditCardNumberPlainText)
FROM dbo.tblCustomerData;
GO
CLOSE SYMMETRIC KEY SymmetricKey1;
GO
```

Voici les résultats obtenus

Nombre d'enregistrement (En millions)	Durée (ms) du SELECT sur le champ PLAIN TEXT	Durée (ms) du SELECT sur le champ chiffré
1	1100	4000
2	2500	6500
3	3600	10600
4	4900	13300
5	7300	16500
6	8700	19900
7	11900	23100
8	12100	28000
9	14000	30300
10	15200	33500

2 PERFORMANCE DU CHIFFREMENT DE COLONNE



On remarque tout d'abord que c'est plus long de lire un enregistrement chiffré que de lire un enregistrement non chiffré (on s'y attendait).
On remarque ensuite que plus le nombre d'enregistrements augmente, plus le temps nécessaire à la function DecryptbyKey est important.
On remarque également que l'augmentation du temps est linéaire dans les 2 cas.

Éditeur
Groupe Oceanet Technology

Date de publication
Octobre 2021

Contact
contact@oceanet-technology.com
+33 (0)2 28 03 78 78
oceanet-technology.com