

Secteur Tertiaire Informatique
Filière « Etude et développement »

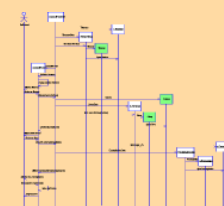
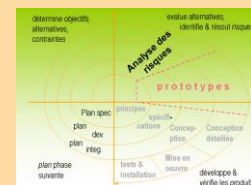
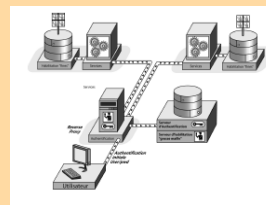
Séquence « Mettre en place une base de données »

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Apprentissage

Mise en situation

Evaluation



Version	Date	Auteur(s)	Action(s)
1.0	18/07/16	Lécu Régis	Création du document

TABLE DES MATIERES

Table des matières	2
1. Introduction	4
2. Faux-semblants et Performances.....	4
2.1 Fonctionnalités fantômes	4
2.2 Performance du SGBD et déni de service	5
2.2.1 L'optimisation du modèle pour le SGBD cible	5
2.2.2 Tests de performances et sécurité.....	5
3. La configuration du serveur.....	6
3.1 Authentification et défense en profondeur	6
3.1.1 Choisir le bon mode d'authentification	6
3.1.2 Ne créer que les utilisateurs indispensables dans votre base de données.....	7
3.1.3 Créer des Schémas	7
3.2 Ne pas laisser les clés sur la porte.....	8
3.2.1 Ne garder aucun compte inutile ou sans mot de passe.....	8
3.2.2 Choisir une bonne stratégie de mots de passe	9
3.3 Contrôler la charge de son serveur	10
3.3.1 Configuration préventive	10
3.3.2 Surveiller la charge pendant l'exploitation (ou les tests d'intégration)	10
3.4 Configurer l'audit du serveur	11
3.4.1 Intérêt d'un audit de serveur pour un développeur	11
3.4.2 Quelques repères avec <i>SQL Server</i>	12
4. Conclusion	14
4.1 Une mise en œuvre délicate sur un serveur pédagogique.....	14
4.2 Mise en pratique optionnelle sur un serveur local.....	14

Objectifs

A l'issue de cette séance, le stagiaire sera capable de :

- Connaître les vulnérabilités et les attaques classiques dues à un mauvais paramétrage du SGBD : mots de passe faibles, comptes système par défaut, choix d'une configuration peu sécurisée (par exemple, la sécurité Windows dans SQL Server).
- Paramétrer le SGBD en évitant les pièges de sécurité courants.

La séance prendra ses exemples dans Microsoft SQL Server mais elle n'est pas spécialisée : les mêmes raisonnements pourraient être faits avec Oracle ou MySQL, en s'appuyant sur la documentation de l'éditeur.

Pré requis

Suit les séances « *Ecrire les scripts SQL de gestion des contraintes sur les données* » et « *Administrer la base de données de test* » qu'elle précise avec une approche sécurité.

Méthodologie

Ce document peut être utilisé en présentiel ou à distance.

Il précise la situation professionnelle visée par la séance, la situe dans la formation, et guide le stagiaire dans son apprentissage et ses recherches complémentaires.

Mode d'emploi

Symboles utilisés :



Renvoie à des supports de cours, des livres ou à la documentation en ligne constructeur.



Propose des exercices ou des mises en situation pratiques.



Point important qui mérite d'être souligné !

Ressources

Documentations techniques en ligne des éditeurs sur l'installation et le paramétrage de leur SGBD. Pour Microsoft SQL Server :

<https://www.microsoft.com/fr-fr/server-cloud/products/sql-server/>

Fiche produit sur SQL Serveur 2016, en français :

SQL_Server_2016FicheProduit.pdf

Livre blanc de Microsoft sur SQL Server 2016 (chap. 2 sur les améliorations en sécurité) :

IntroSQLServer2016.pdf

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Afpa © 2014 – Section Tertiaire Informatique – Filière « Etude et développement »

1. INTRODUCTION

En tant que développeur, vous aurez à paramétrer le SGBD, pour votre base de données de test. Pour mener à bien cette tâche :

- vous devrez prendre des options réalistes et faire des choix de sécurité correspondant au niveau de sécurité attendu de votre application ;
- même si vous ne créez pas vous-même la base de données d'exploitation, il vous faudra expliciter vos choix au DBA en charge de cette base. Cela exigera un dialogue technique, où vous formulerez des demandes précises, afin d'assurer au mieux la sécurité de votre application.

Cette séance va vous préparer à cette situation professionnelle, en complétant la séance précédente sur l'administration de la base de données :

- en terme de savoir-faire, elle n'est pas dissociable, mais elle adopte un point de vue sécurité qui va orienter les choix d'administration ;
- ce point de vue sécurité sera mis en œuvre dans la mise en situation qui suit (en partie).

2. FAUX-SEMBLANTS ET PERFORMANCES

En Merise, nous avons travaillé le MCD pour aboutir à un MLD qui donne une vue logique indépendante du SGBD cible. Cette approche est parfaitement normale dans la phase d'analyse.

Mais pour adopter un point de vue sécurité, il faut la compléter par un modèle physique, au moment du choix du SGBD et de la création de la base. Ce modèle doit prendre en compte très SÉRIEUSEMENT les spécificités du SGBD cible, grâce à la documentation technique de l'éditeur, en vérifiant :

- ce que fait vraiment le SGBD, dans une version et une distribution données ;
- ses performances face à des jeux de données critiques.

Nous avons un devoir de conseil vis-à-vis du client : si le niveau de sécurité attendu de l'application est incompatible avec les possibilités de sécurité d'un serveur, il faut :

- le découvrir au plus tôt dans le cycle de vie du logiciel,
- en avertir le client,
- et si possible choisir un SGBD plus robuste.

2.1 FONCTIONNALITES FANTOMES

Avant toute chose, s'assurer qu'une propriété de sécurité existe dans le SGBD choisi :

- elle est reconnue par l'interpréteur SQL
- elle est VRAIMENT implémentée dans le SGBD, et non pas seulement reconnue pour compatibilité avec la norme SQL.

Les procédures stockées n'existaient pas en *MySQL* avant la version 5, ce qui interdisait de construire une vraie défense en profondeur, avec un deuxième niveau de retranchement basé sur les procédures stockées.

Mais c'était au moins clair : selon le degré de sécurité demandé, on acceptait de travailler sans procédures stockées ou on optait pour un SGBD plus robuste.

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Il y a pire : des fonctionnalités fantômes, décrites dans les diagrammes syntaxiques, qui ne provoquent pas d'erreur à l'interprétation des requêtes mais ne sont pas implémentées.

Exemple *MySQL* :

- les contraintes *CHECK* sont reconnues par l'interpréteur *MySQL* mais elles ne font rien à l'exécution ;
- il faut les émuler par des déclencheurs qui fonctionnent correctement en *MySQL*.

2.2 PERFORMANCE DU SGBD ET DENI DE SERVICE

2.2.1 L'optimisation du modèle pour le SGBD cible

La Méthode Merise intègre les problèmes de performance, en distinguant le MLD brut du MLD optimisé :

- si le MLD comporte trop de tables, les jointures peuvent atteindre des durées inacceptables ;
- il faut alors dénormaliser le modèle, en réduisant artificiellement le nombre de tables, pour revenir à des temps de réponse réalistes.

On peut aussi rajouter des *index*, pour accélérer des requêtes en accès direct : si la requête *select * from auteurs where nom='toto' and prenom='titi'* est trop lente, on peut poser un index composé sur les colonnes *nom* et *prenom*.

A ce stade, il faut se poser concrètement les questions :

- jusqu'où faut-il dénormaliser, pour que la durée des requêtes soit acceptable ?

La dénormalisation a un coût : elle crée de la redondance entre les tables ;

- que valent les index sur le SGBD choisi ?

2.2.2 Tests de performances et sécurité

Après chaque optimisation, il faut relancer un test de performance en vraie grandeur, sur un jeu de données au moins aussi important que celui attendu en exploitation, afin de détecter si certaines requêtes atteignent encore des durées inacceptables.

D'un point de vue fonctionnel, de mauvais temps de réponse vont décourager l'utilisateur et éventuellement entraîner des réserves lors la recette de l'application.

Mais ces lenteurs peuvent aussi être exploitées consciemment par un attaquant pour créer un déni de service :

- c'est une attaque simple : il n'y a pas besoin de casser le mécanisme d'authentification ;
- il suffit de repérer les faiblesses du système et de répéter les requêtes les plus lentes, pour rendre l'application inutilisable.

Tout test de performance est donc aussi un test de sécurité puisqu'il peut mettre en évidence une vulnérabilité de type « déni de service ».

3. LA CONFIGURATION DU SERVEUR

Quel que soit le SGBD choisi, sa configuration aura des conséquences sur la sécurisation des bases de données et des applications.

En prenant comme exemple *SQL Server*, nous allons examiner quelques choix de configuration, en les rattachant aux principes de sécurités déjà énoncés.

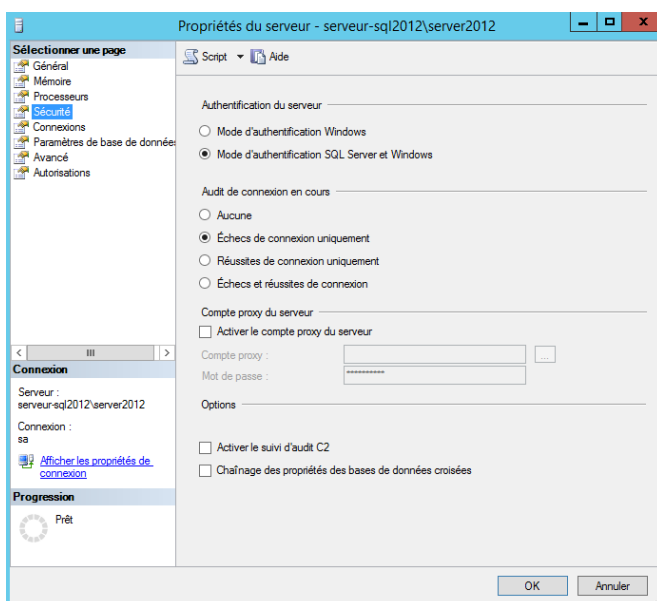
L'objectif n'est pas de devenir DBA *SQL Server*, mais de pouvoir être force de proposition, en demandant au DBA tel ou tel paramétrage, selon le niveau de sécurité attendu de votre base de données et de votre application.

3.1 AUTHENTIFICATION ET DEFENSE EN PROFONDEUR

3.1.1 Choisir le bon mode d'authentification

Une défense en profondeur exige plusieurs authentifications : dans un château-fort, un premier mot de passe pour passer le rempart extérieur, et un deuxième mot de passe ou une reconnaissance visuelle pour entrer dans le donjon.

SQL Server (toutes versions) proposent deux modes d'authentification :



(*SQL Server Management Studio*, propriété de serveur)

Mode d'authentification Windows

- c'est le compte de l'utilisateur connecté à Windows qui est utilisé pour son authentification par le SGBD ;
- avec ce choix, il n'y a pas de défense en profondeur. Si le domaine ou l'annuaire (*Active Directory*) est compromis, il y aura automatiquement compromission du SGBD.

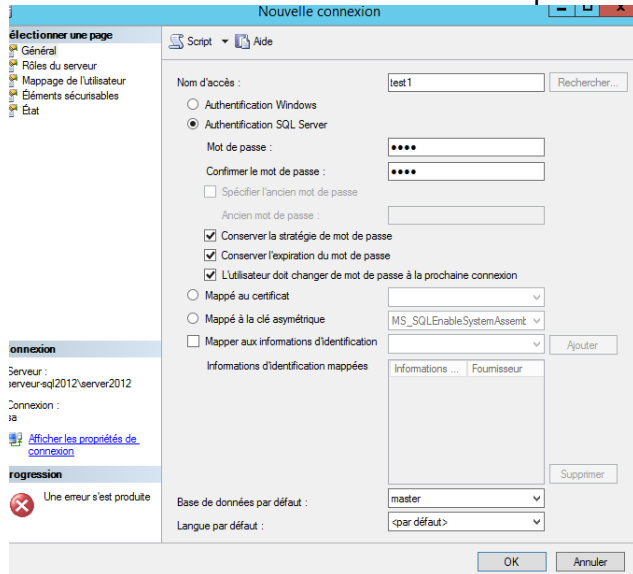
Mode d'authentification SQL Server et Windows

- dans ce mode fortement conseillé, le SGBD dispose de ses propres comptes de connexion, qui sont dans l'onglet Sécurité.

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Afpa © 2014 – Section Tertiaire Informatique – Filière « Etude et développement »

- le serveur est dans un mode mixte et peut reconnaître aussi les comptes Windows :



- mais on se forcera à n'utiliser QUE les comptes de connexion propres au SGBD (*Authentification SQL Server*)
- dans cette configuration, il y aura réellement une défense en profondeur à deux niveaux : si un individu malveillant parvient à se connecter à votre poste client, il n'aura aucun droit sur la base de données.

3.1.2 Ne créer que les utilisateurs indispensables dans votre base de données

La défense en profondeur est bien mise en œuvre dans *SQL Server*, qui distingue la **Connexion** au SGBD et l'**Utilisateur** d'une base de données :

- le compte de connexion au SGBD (par exemple *test1*) permet uniquement de se connecter au SGBD (*le rempart extérieur*), mais ne donne aucun droit sur les différentes bases de données qui y sont contenues (*les tours*) ;
- pour qu'un compte de connexion (*test1*) ait accès à une base de données, il faut créer un Utilisateur (*comptable*) dans cette base de données, et l'associer à la connexion *test1*.

Un utilisateur malveillant qui se connecte indument avec un compte de connexion sans droit sur votre base, ne peut donc pas s'y introduire : *il est comme l'attaquant dans le château-fort, qui a pris le premier rempart sans pouvoir s'introduire dans une tour.*

3.1.3 Créer des Schémas

Toujours dans *SQL Server*, le *Schéma* permet de créer un nouveau niveau de retranchement (*l'étage fortifié dans une tour*) :

- un schéma regroupe différentes tables, vues, procédures, fonctions d'une base de données, auxquelles on veut donner les mêmes autorisations pour certains rôles ou utilisateurs ;
- le schéma est donc une entité de sécurité intermédiaire, entre la base de données et les objets élémentaires comme les tables ou les vues ;

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

- il facilite la mise en œuvre du principe de sécurité « *séparer et minimiser les permissions et les privilèges* », en ne donnant à chaque utilisateur et à chaque rôle que les droits nécessaires.

3.2 NE PAS LAISSER LES CLES SUR LA PORTE

Les remparts et les tours ne servent à rien, si les attaquants ont la clé de la poterne ou connaissent le mot de passe pour passer le pont-levis.

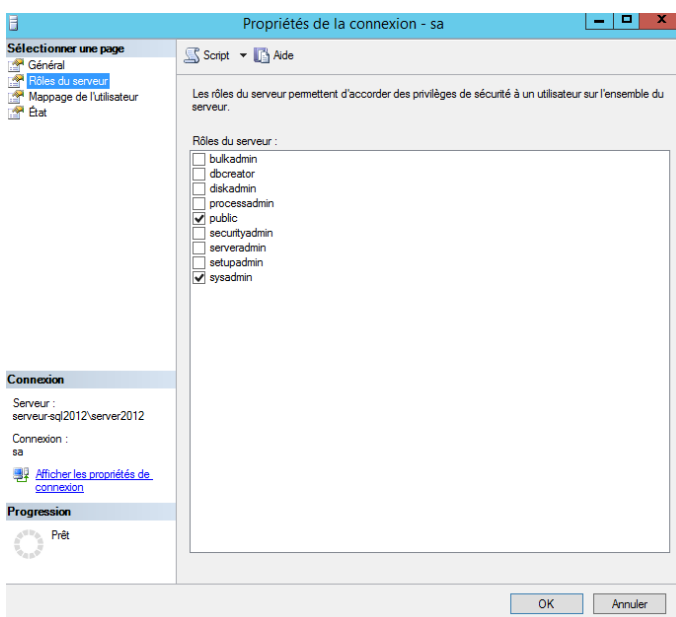
Quel que soit le serveur choisi, il faudra :

- éliminer les valeurs par défaut (décrites dans la documentation), qui permettent des attaques immédiates sur le serveur ;
- choisir une stratégie de mot de passe solide.

3.2.1 Ne garder aucun compte inutile ou sans mot de passe

Lorsque l'on bascule SQL Server dans le mode conseillé « *Authentification SQL Server* », il faut activer le compte d'administration *sa*, avec un mot de passe fort.

Le compte *sa* est membre du rôle de serveur *sysadmin*, qui lui donne tous les droits sur le serveur :



S'il faut créer des administrateurs secondaires pour la gestion de certaines tâches du SGBD :

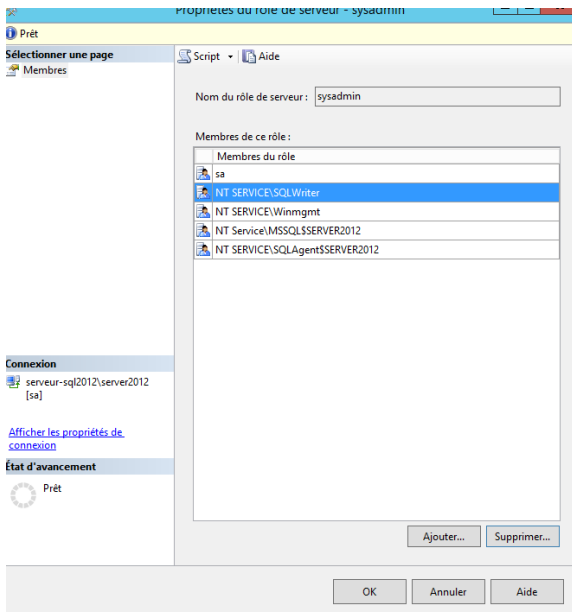
- il faut les insérer dans les rôles qui correspondent à leur tâche, plutôt que dans *sysadmin* (application du principe « *séparer et minimiser les droits* »)
- il faut leur attribuer un mot de passe fort, comme pour le compte *sa*.

Dans une revue de sécurité du SGBD, on vérifiera spécialement les connexions qui sont membre d'un rôle serveur, et en particulier les connexions par défaut liées à des comptes de service Microsoft : *NT Service\MSSQL\$SERVER2012*

Ces connexions que l'on a tendance à oublier, peuvent être un vecteur d'attaque.

Ne conservez que les connexions des services réellement utilisés et renforcez leur mot de passe.

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

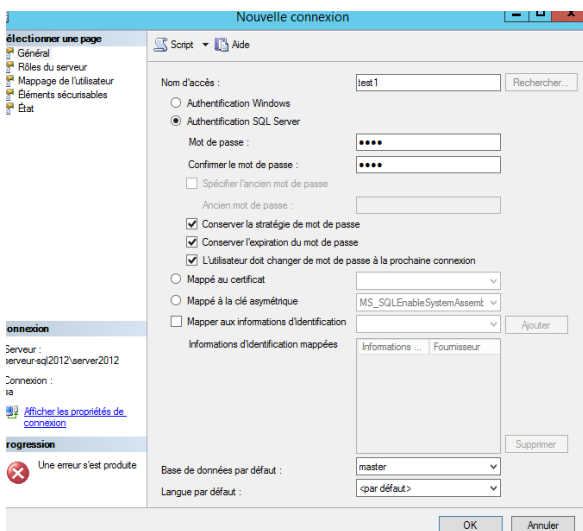


Voir la définition des différents rôles de serveur, chez Microsoft :

<https://msdn.microsoft.com/fr-fr/library/ms188659.aspx>

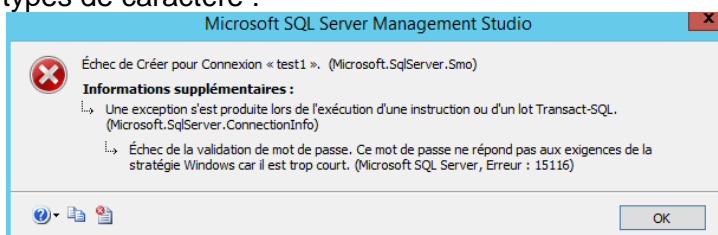
3.2.2 Choisir une bonne stratégie de mots de passe

Revenons sur le formulaire de création de connexion.



Cochez systématiquement « *Conserver la stratégie de mot de passe* ».

Le SGBD n'acceptera que des mots de passe suffisamment longs, et comportant plusieurs types de caractère :



Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Afpa © 2014 – Section Tertiaire Informatique – Filière « Etude et développement »

Cochez l'option « *Conserver l'expiration du mot de passe* » pour tous les comptes non privilégiés, afin d'obliger les utilisateurs à renouveler périodiquement leur mot de passe.

La perte du mot de passe d'un compte privilégié étant désastreuse pour le SGDB, il vaut mieux ne pas cocher cette case pour des comptes d'administration, comme sa.

Le DBA prendra la responsabilité de changer régulièrement son mot de passe, avec une périodicité définie par lui.

De même, l'option « *L'utilisateur doit changer de mot de passe à la prochaine connexion* » ne doit être utilisée que pour des utilisateurs de base. Elle est utile pour un renouvellement exceptionnel des mots de passe (en plus de l'option « *Conserver l'expiration des mots de passe* »).

3.3 CONTROLER LA CHARGE DE SON SERVEUR

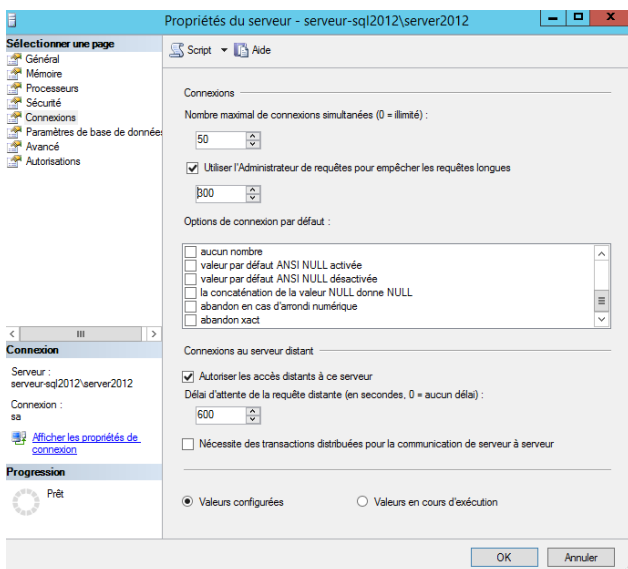
3.3.1 Configuration préventive

Les attaques par déni de service peuvent être évitées ou au moins atténuées par une bonne configuration du serveur. Ces attaques visent à interdire ou à réduire l'activité utile du serveur, en le surchargeant :

- par un grand nombre de connexions simultanées ;
- par l'envoi de requêtes inutiles, construites spécialement pour provoquer des traitements longs.

En prévention, la plupart des SGDB peuvent être configurés pour :

- n'accepter qu'un nombre limité de connexions simultanées. Dans le cadre d'un Intranet avec un nombre d'utilisateurs restreint (50 dans l'exemple suivant), cette mesure de prévention est facile à mettre en œuvre ;
- limiter la durée de traitement d'une requête (en secondes, 300 dans l'exemple suivant).

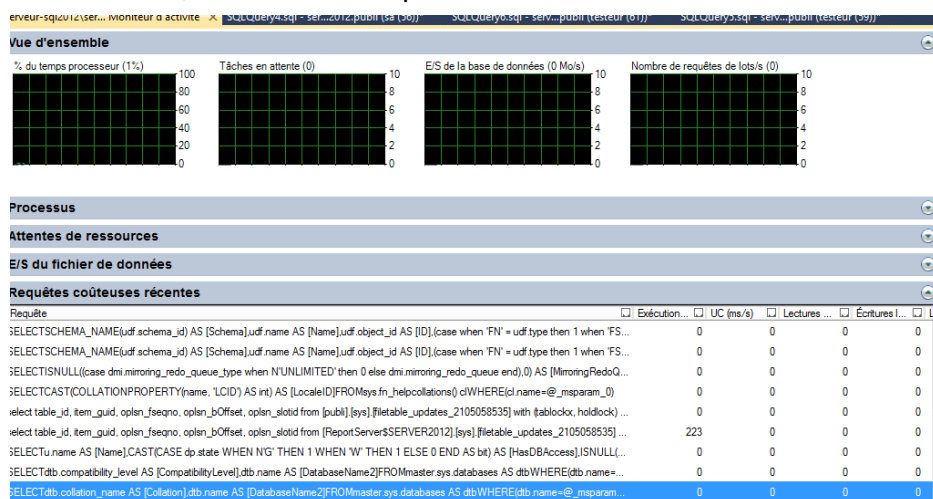


3.3.2 Surveiller la charge pendant l'exploitation (ou les tests d'intégration)

Une configuration préventive suffit rarement.

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

La plupart des SGBD comprennent un « moniteur d'activité » qui permet de surveiller dynamiquement le travail du SGBD, les processus des clients connectés, les ressources consommées, la durée des requêtes :



Cet outil sera utile à l'exploitant pour détecter et analyser toute activité anormale sur le serveur, qui peut être une attaque.

Pendant la phase de test, ce sera davantage un outil de mise au point que de sécurité : on recherchera les requêtes anormalement lentes ou consommatrices de ressources, qui devront être réécrites.

Mais nous avons que les deux aspects sont liés : une lenteur excessive est une vulnérabilité qui pourra être exploitée par un attaquant perspicace.



Utilisation « pas à pas » du moniteur d'activité, dans la documentation Microsoft :

<https://msdn.microsoft.com/fr-fr/library/ms175518.aspx#SSMSProcedure>

3.4 CONFIGURER L'AUDIT DU SERVEUR

Il ne suffit pas de croire que l'on n'est pas attaqué : il faut faire des vérifications, périodiquement et sur des événements clé.

Même avec de bonnes fortifications, il faut poster des sentinelles aux endroits stratégiques et organiser des rondes régulières dans chaque partie du château.

La plupart des SGBD ont une fonctionnalité d'audit qui permet de surveiller tous les événements de sécurité : connexion réussie ou en échec, modification de mot de passe, création d'un compte etc.

Cette fonctionnalité va participer à la **Preuve** de non compromission de notre SGBD : le **P** de **DICP**, qui démontre que le système est toujours capable de garantir la **Disponibilité** des données, leur **Intégrité** et leur **Confidentialité**.

3.4.1 Intérêt d'un audit de serveur pour un développeur

La configuration d'un audit sur un serveur en exploitation relève en général du DBA (qui peut aussi être développeur dans une PME !).

Il faut être prudent et raisonnable dans le choix des objets et des événements à auditer, car les processus d'audit consomment des ressources, et peuvent ralentir le serveur en exploitation.

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Afpa © 2014 – Section Tertiaire Informatique – Filière « Etude et développement »

Mais le développeur peut activer l'audit sur sa base de données de test, pour vérifier certaines propriétés de sécurité de sa base de données et de son application.

Par exemple, on peut :

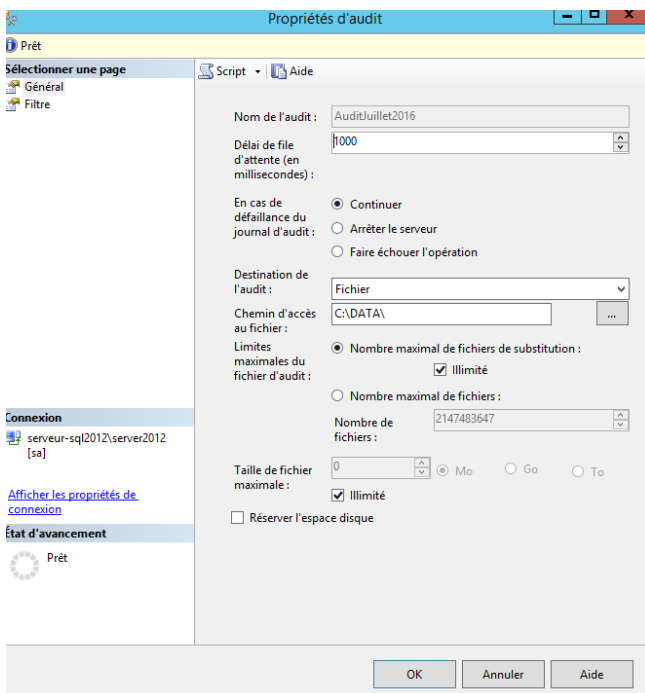
- auditer les tentatives de *login* avec échec, et vérifier que cet échec a provoqué une exception *JDBC* côté client, et a bien été pris en compte par notre application ;
- loguer les opérations réussies / en échec sur une table (*select, insert, update, delete*). Si des opérations sont tentées via *JDBC* par un utilisateur non autorisé, l'analyse du fichier d'audit peut mettre en évidence des vulnérabilités du client, comme l'accès par un utilisateur à un menu qui ne devrait pas être visible.

Si la revue de code client ne révèle pas de vulnérabilité, l'analyse du fichier d'audit peut aussi révéler une attaque par un client *cracké* ou codé spécifiquement dans le but de réaliser l'attaque sur notre base.

Les audits de serveur pratiqués pendant les tests peuvent servir de « reconnaissance », pour conseiller le DBA en charge de l'exploitation : le développeur repère les données critiques pour son application et peut demander qu'un audit réalisé en phase de test soit poursuivi, dans la mesure du possible, pendant l'exploitation.

Le développeur devra encore une fois être force de proposition : il reviendra au DBA de juger si cette demande est raisonnable, en évaluant la consommation de ressources (espace disque des fichiers d'audit etc.) et le ralentissement des réponses résultant de cet audit.

3.4.2 Quelques repères avec *SQL Server*



Il faut :

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

1. Créer un audit : *AuditJuillet2016* et lui associer un fichier et un répertoire de destination.

On ne conservera pas les options par défaut ci-dessus, qui constituent elles-mêmes un risque en cas de suractivité du serveur (attaque par déni de service) : pensez à la combinaison « illimité » avec « arrêtez le serveur » en « cas de défaillance du journal d'audit ».

On fixera au contraire une taille maximale au fichier d'audit, un nombre limité de fichiers, et on réservera la place nécessaire sur le disque.



Voir documentation Microsoft :

<https://msdn.microsoft.com/fr-fr/library/cc280424.aspx>

avec l'utilisation pas à pas de SQL Server Management Studio :

<https://msdn.microsoft.com/fr-fr/library/cc280424.aspx#SSMSProcedure>

2. Définir ensuite la spécification du nouvel audit :

Type d'action de l'audit	Classe d'objets	Schéma d'objet	Nom de l'objet	Nom du
1 FAILED_LOGIN_GROUP				
2				

Quels objets (« *Classe d'objets* », « *Schéma d'objet* », « *Nom de l'objet* ») et quels évènements sur ces objets (« *Type d'action de l'audit* ») doit-on auditer ?

3. Visualiser un fichier d'audit :

Date	Heure de l'événement	Nom de l'instance du serveur	ID d'action	Type de classe	Numéro sé
12/07/2016 15:49:18	15:49:18.7040700	SERVEUR-SQL2012-SERVER2012	LOGIN FAILED	LOGIN	1
12/07/2016 15:49:02	15:49:02.3276283	SERVEUR-SQL2012-SERVER2012	AUDIT SESSION CHANGED	SERVER AUDIT	0

On voit qu'une connexion a échoué le 12/07/2016 à 15:49.

Est-ce un horaire normal d'utilisation de l'application, quelle est l'adresse IP du client qui a tenté cette connexion etc. ?

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Afpa © 2014 – Section Tertiaire Informatique – Filière « Etude et développement »

4. CONCLUSION

Les techniques de paramétrage du SGBD :

- sont à la limite des métiers du développement informatique : elles peuvent être réservées au DBA dans de grosses structures ou confiées au développeur isolé dans une PME ;
- elles sont utiles au développeur pour prendre les bonnes décisions de sécurité et communiquer techniquement avec le DBA.

4.1 UNE MISE EN ŒUVRE DELICATE SUR UN SERVEUR PEDAGOGIQUE

La gestion des comptes utilisateurs, des rôles et des schémas, la stratégie de mots de passe seront mises en pratique dans la séance suivante : mise en situation de la séquence « *Mettre en place une base de données* ».

Mais certaines techniques présentées dans cette séance :

- exigent d'avoir tous les droits sur le serveur ;
- ne pourront donc pas être pratiquées dans la mise en situation qui suit.

Pour ne pas rester dans l'abstrait, nous conseillons une mise en pratique optionnelle, pour les stagiaires motivés par l'approche sécurité.

4.2 MISE EN PRATIQUE OPTIONNELLE SUR UN SERVEUR LOCAL

Le mieux est de reprendre les choses à zéro en :

- installant son propre SGBD en local (*SQL Server Express* ou mieux, *SQL Server 2016* en version d'essai ou dans le cadre du programme « *Academic Alliance* ») ;
- configurant le serveur avec le mode d'authentification le plus sûr (*Authentification SQL Server*) ;
- utilisant la stratégie de mot de passe et la durée maximale du mot de passe, pour les comptes non privilégiés ;
- régulant la charge du serveur (nombre d'utilisateurs simultanés, durée maximum d'une requête) ;
- suivant l'activité du serveur par le moniteur d'activité ;
- créant une base de données de test : à partir d'un script écrit dans les séances précédentes (non fourni) ;
- ne créant que les utilisateurs nécessaires à votre application et votre base. Vérifiez que les autres comptes de connexion ne peuvent pas accéder à la base ;
- définissant un audit (par exemple sur les connexions réussies et en échec) ;
- en visualisant cet audit.

On s'appuiera sur la documentation Microsoft en ligne et en particulier sur les présentations « pas à pas »

CRÉDITS

OEUVRE COLLECTIVE DE L'AFPA

Sous le pilotage de la DIIP
et du centre sectoriel Tertiaire

EQUIPE DE CONCEPTION

Chantal PERRACHON – IF Neuilly-sur-Marne
Régis Lécu – Formateur AFPA Pont de Claix

Reproduction interdite

Article L 122-4 du code de la propriété intellectuelle.

« Toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droits ou ayants cause est illicite. Il en est de même pour la traduction, l'adaptation ou la reproduction par un art ou un procédé quelconque. »

Paramétrer le SGBD en suivant les bonnes pratiques de sécurité

Afpa © 2014 – Section Tertiaire Informatique – Filière « Etude et développement »