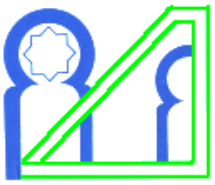

UNIVERSITÉ MOHAMMED V – AGDAL
FACULTÉ DES SCIENCES
Rabat

Thèse

*En vue l'obtention du grade de **DOCTEUR** en Sciences*

Karim EL BOUCHTI

Structure de recherche : Intelligent Processing Systems & Security

Discipline : Informatique

Spécialité : Sécurité Informatique

Sujet de thèse

Sécurité des Bases de Données : modèles de chiffrement des données et de protection des clés

*Soutenue publiquement le **01/12/2020**, devant le jury composé de :*

Fouzia BENABBOU	PES, Université Hassan II, Casablanca. Faculté des sciences Ben M'Sik.	Présidente
Fouzia OMARY	PES, Université Mohammed V, Rabat. Faculté des sciences.	Directrice de thèse
Soumia ZITI	PH, Université Mohammed V, Rabat. Faculté des sciences.	Co-directrice de thèse
Mohamed DAKKI	PES, Université Mohammed V, Rabat Institut Scientifique	Rapporteur
Abderrahim TRAGHA	PES, Université Hassan II, Casablanca. Faculté des sciences Ben M'Sik.	Rapporteur
Karim DOUMI	PH, Faculté des Sciences Juridiques et Sociales Université Mohammed V de Rabat	Rapporteur
Abderrahmane EZ ZAHOUT	PA, Université Mohammed V, Rabat. Faculté des sciences.	Invité

DEDICACE

A Dieu le tout puissant

A mon prophète Mohammed (bénédiction et paix sur Lui)

A mes très chers parents Taibi et Amina,

A ma chère femme Amina et mes chers enfants Aroua et Nouh,

Aucun mot ne pourra exprimer mes sentiments envers vous.

A mes chères sœurs,

A mon cher frère,

Je ne sais comment vous remercier pour tout ce que vous avez fait pour moi.

A toute ma famille.

A tous mes chers amis de Casablanca et de Kenitra

A mes chers amis de la Faculté des Sciences de Rabat

Pour tout le soutien que vous m'avez offert, je vous dis MERCI.

A tous ceux qui m'aiment.

A tous les musulmans à travers le monde, je dédie ce travail…

Karim

AVANT PROPOS

Ces travaux de recherche ont été réalisés au sein de l'équipe Intelligent Processing Systems & Security (IPSS) à la Faculté des Sciences de l'Université Mohammed V de Rabat sous la direction des Professeurs Fouzia OMARY et Soumiya ZITI.

A la tête de ces personnes, je tiens à remercier ma directrice de thèse, Mme Fouzia OMARY, Professeur de l'Enseignement Supérieur à la Faculté des Sciences de Rabat, de m'avoir fait confiance et m'accepté au sein de son équipe de recherche en me proposant ce sujet de thèse, ainsi que de me guider tout au long de mes travaux de recherches en doctorat. J'ai beaucoup appris de sa méthodologie de recherche, de son raisonnement scientifique et de sa précision. Merci Professeur pour vos précieux conseils, votre temps et votre soutien moral.

J'exprime mes profondes gratitudes et je tiens à remercier Mme Soumia ZITI, Professeur habilitée à la Faculté des Sciences de Rabat, pour son encadrement et son soutien, sa motivation, sa disponibilité et son encouragement, ses conseils et ses idées pertinentes dans la conduite de ce travail qui m'ont vraiment apporté de la valeur ajoutée dans mes travaux.

Je tiens à remercier Mme Fouzia BENABBOU, Professeur de l'Enseignement Supérieur à l'Université Hassan II-Mohammedia, Faculté des Sciences Ben M'Sik de Casablanca, d'avoir accepté de présider le jury. Je tiens à vous exprimer mes respectueuses considérations.

Mes remerciements les plus sincères vont également à Mr. Abderrahim TRAGHA, Professeur de l'Enseignement Supérieur à l'Université Hassan II-Mohammedia, Faculté des Sciences Ben M'Sik de Casablanca, d'avoir accepté de rapporter et de juger ce travail.

Mes vifs remerciements et mes respects s'adressent également à Mr. Mohamed DAKKI, Professeur de l'Enseignement Supérieur à l'Institut Scientifique, Faculté des Sciences de Rabat, d'avoir accepté d'être parmi le jury de cette thèse ainsi que d'examiner et de rapporter ce travail.

Mes remerciements les plus sincères vont également à Mr. Karim DOUMI, Professeur habilité à l'Université Mohammed V de Rabat, Faculté des Sciences Juridiques et Sociales, d'avoir accepté de rapporter et de juger ce travail.

Je tiens à remercier chaleureusement Mr. Abderrahmane EZ ZAHOUT, Professeur assistant à la Faculté des Sciences de Rabat, d'avoir accepté d'être parmi le jury de cette thèse ainsi que pour son soutien et sa disponibilité.

Je remercie très particulièrement Mr. El Mehdi HAMZAOUI et Mr Abdelfettah BENCHRIF, Docteurs de la Faculté des Sciences de Rabat et chercheurs au Centre National de l'Energie, des Sciences et des Techniques Nucléaires (CNESTEN), pour leur aide à la rédaction du rapport, leurs conseils lucides et leurs idées pertinentes.

Mes remerciements vont également à tous les membres de l'équipe IPSS, plus particulièrement à mon cher ami Nassim KHARMOUM pour ses conseils, encouragements et pour son aide cruciale dans les différentes étapes de ce travail.

Mes remerciements vont à tous ceux et celles qui ont contribué de près ou de loin à l'accomplissement de ce travail.

RÉSUMÉ

Ce travail de thèse s'inscrit dans le contexte d'améliorer la sécurité des bases de données via le chiffrement des données en proposant le développement de deux principaux axes : les modèles de chiffrement des bases de données et les modèles de protection des clés de chiffrement.

D'une part, nous avons conçu et implémenté un nouveau concept nommé « Modèle Global de Chiffrement » qui assure, en plus de la protection des données, un autre type de protection qui est celui de la structure de la base de données. D'autre part, nous avons présenté deux travaux relatifs à la protection des clés. Le premier consiste à proposer et implémenter quatre approches de protection des clés de chiffrement au sein des Systèmes de Gestion de Base de données selon la granularité du chiffrement choisie, et le deuxième se base sur une proposition d'une approche de protection des clés de chiffrement lorsque le chiffrement des données est implémenté au niveau des applications.

Mots-clés : Sécurité des bases de données, Modèle de chiffrement des bases de données, Protection de la confidentialité des bases de données, Modèle de protection des clés de chiffrement, Modèle de protection des clés de chiffrement dans le Cloud.

ABSTRACT

In this thesis, we aim to improve the database's security using data encryption by proposing the development of two main axes: database encryption models and encryption keys protection models. First, we have designed and implemented an original concept called « The Global Encryption Model » which provides, in addition to data protection, another type of protection. This last consist of the protection of database structure. On the other hand, we have presented two works related to the encryption keys protection. The first one is the proposal and the implementation of four approaches to protect encryption keys within the Database Management Systems according to the encryption granularity level. The second work focuses on a proposal of a new approach aiming to protect the encryption keys when data encryption is performed at the application level.

Keywords: Database security, Database encryption model, Database confidentiality protection, Encryption keys protection model, Encryption keys protection in the Cloud.

RÉSUMÉ DÉTAILLÉ

Les bases de données contiennent une quantité énorme de données qui existent dans le monde, et elles constituent la cible privilégiée des pirates qui souhaitent exploiter leurs contenues de valeur de manière illégitime. La sécurité des bases de données vise à assurer la protection des données sensibles contre la fuite, la divulgation et la modification de données dans le but de se défendre contre les menaces internes et externes des attaquants. Les solutions conventionnelles de sécurité d'une base de données portent principalement sur la mise en œuvre de trois niveaux de sécurité : sécurité physique, sécurité au niveau du système d'exploitation et la sécurité au niveau du Système de Gestion de Base de Données Relationnel.

La sécurité au niveau du Système de Gestion de Base de Données Relationnel implémente le mécanisme de chiffrement des données, c'est l'un des éléments puissants qui renforce extrêmement la sécurité des données aux repos. Actuellement, la plupart des Systèmes de Gestion de Bases de Données Relationnel proposent des solutions de chiffrement qui se différent l'une de l'autre par le modèle de chiffrement qu'elles adoptent. En effet, un modèle de chiffrement est le concept qui détermine la manière dont les données sont chiffrées dans la base de données, il doit forcément respecter le maximum des critères d'un modèle pertinent. Les modèles de chiffrement requièrent encore de l'amélioration dans leurs concepts afin d'apporter plus de sécurité. En effet, ils souffrent de deux problèmes fondamentaux. Premièrement, Ils n'arrivent pas à combiner le maximum des critères d'un modèle pertinent dans un seul modèle. D'autres part, ils nécessitent de l'innovation dans leurs conceptions, d'intégration de nouvelles caractéristiques pour faire face aux nouveaux défis imposés par le développement rapide de la technologie et de l'évolution exponentielle des moyens et des types d'attaques.

Le travail de cette thèse s'inscrit dans le but d'améliorer la sécurité des bases de données via la contribution dans deux volets principaux : les modèles de chiffrement des bases de données respectant le maximum des critères d'un modèle pertinent de chiffrement, et les modèles de protection des clés de chiffrement des données. Ainsi, nos contributions sont résumées dans les travaux suivants :

La première contribution porte sur l'étude des solutions de chiffrement des bases de données qui existent sur le marché. Cette étude nous a donné une vision claire sur le fonctionnement des modèles de chiffrement intégrés au sein de ces solutions, leurs points forts et faibles ainsi

que les méthodes que peuvent déployer les attaquants pour mener des attaques sur les données stockées.

La deuxième contribution est relative à la proposition d'un nouveau modèle qui assure la protection de la confidentialité d'une base de données. C'est un travail original qui se base sur la notion des « Classes de chiffrement » permettant de protéger la confidentialité des données des tables sensibles. Ce modèle propose quatre sous modèles qui assurent respectivement le chiffrement des données, la génération des clés de chiffrement, leur protection ainsi que la protection de la structure de la base de données.

La troisième contribution porte sur une nouvelle approche de protection des clés de chiffrement au sein du Système de Gestion de Bases de Données Relationnel. Cette méthode permet de générer des clés principales qui vont protéger les clés de chiffrement de la base de données en fonction du niveau de la granularité de chiffrement adopté par le modèle de chiffrement.

Quant à la quatrième contribution, nous nous sommes intéressés à la problématique de la protection des clés de chiffrement lorsqu'on chiffre la base de données coté client, autrement dit au niveau des applications. Le principe de la solution qu'on propose consiste à transformer les requêtes des clients contenant des clés de chiffrement vers des nouvelles requêtes qui appellent à distance un service de sécurité stocké sur le serveur de base de données. Ce dernier convertit les clés de chiffrement définies sur les clients vers les clés réelles de chiffrement et de déchiffrement des données.

Mots-clés : Sécurité des bases de données, Modèle de chiffrement des bases de données, Protection de la confidentialité des bases de données, Modèle de protection des clés de chiffrement, Modèle de Protection des clés de chiffrement dans le Cloud.

TABLE DES MATIETRES

LISTE DES NOTATIONS ET ABREVIATIONS

DDOS: Distributed Denial of Service.

DOS: Denial of Service.

NIST: National Institute of Standards and Technology.

SPN: Substitution Permutation Network.

AES: Advanced Encryption Standard.

DES : Data Encryption Standard.

3 DES : Triple Data Encryption Standard.

SHA0: Secure Hash Algorithm 0.

SHA1: Secure Hash Algorithm 1.

SHA2: Secure Hash Algorithm 2.

SHA3: Secure Hash Algorithm 3.

MD4: Message Digest 4.

CBC: Cipher Block Chaining.

ECB: Electronic Codebook.

EFF: Electronic Frontier Foundation.

CTR: CounTeR.

TPM: Trusted Platform Module.

HSM: Hardware Security Module.

MLR: Multilevel Relational Model.

DBAAS: Database As a Service.

TDE : Transparent Data Encryption.

IA : Intelligence Artificial.

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ALGORITHMES

INTRODUCTION GENERALE

Au cours de cette dernière décennie, le monde connait une phase de transformation numérique de grande envergure. Les investissements colossaux lancés par les pays industriels pour développer la technologie du numérique sont l'un des principaux facteurs ayant contribué à cette transformation. L'intelligence artificielle, la technologie du Blockchain et les objets connectés ne sont que de simples exemples de la révolution qui va bouleverser notre monde devenant de plus en plus interconnecté dans le futur proche. Cette transformation de grandes ampleurs s'est accompagnée de l'émergence d'une nouvelle forme de criminalité qui est la cybercriminalité [1].

Les systèmes informatiques sont devenus de plus en plus vulnérables aux cyber-attaquants qui exploitent les failles de conception, les faiblesses de la technologie et l'erreur humaine. Le nombre d'attaques a augmenté rapidement suite à la numérisation de l'économie, la transformation vers la mobilité et l'évolution remarquable de l'informatique en nuage. Les cyberattaques visent aussi bien les individus que les entreprises de grandes renommées. Aujourd'hui aucun secteur n'est à l'abri. Le secteur financier et de santé sont à la tête des cibles préférées en raison de l'appât du gain et des données sensibles des clients.

Un autre visage de la cyberattaque se manifeste dans la nouvelle guerre menée par les grandes puissances numériques. Une guerre immatérielle qui se joue dans un champ de bataille nommé le cyberspace. Une guerre dont l'objectif est d'espionner l'ennemi, détruire ses projets, ses secrets et ses infrastructures vitaux [27]. L'énorme campagne cyberattaque lancée récemment par les Etats-Unis d'Amérique contre des systèmes de lancement de missile Iranienne suite à la destruction d'un drone américain est un véritable témoin de cette guerre [13].

Les cyberattaques peuvent être de différentes formes et sources, à l'échelle d'un individu ou d'une entreprise, d'un pays ou universelle. Les attaques de mai et juin 2017 par les rançongiciels Wannacry et NotPetya est un exemple concret [68]. Ces attaques avaient touché une diversité de victimes et dont l'étendue de la propagation et les dommages causés ont été considérés historiques. A titre d'exemple, plusieurs services d'entreprises dans beaucoup de pays ont été paralysés outre que la perte de leurs données sensibles.

Dans un contexte national, le Maroc est considéré parmi les pays les plus touchés par le risque cyberattaque. Selon une étude réalisée en 2018 par Kaspersky Lab et Averty, les logiciels

malveillants, les virus, et la perte de données se positionnent à la tête des trois menaces encourues par l'entreprise marocaine [19].

Pour rappel, selon un rapport publié en janvier 2018 par le Forum économique mondial de Davos, le risque de la cyberattaque n'est pas un fléau d'hier ou d'aujourd'hui, il a été classé en troisième place de l'ensemble des risques qui vont potentiellement apparaitre dans les années à venir [68].

Les cyberattaques contre les Systèmes d'Information (SI) est une forme d'attaque qui est devenue une inquiétude majeure dans ces dernières années pour les entreprises. Elles nuisent à leurs images de marque et touchent à la confiance des clients. L'arrêt partiel ou complet d'un ensemble de services d'une entreprise peut lui faire perdre des millions de dollars.

Les conséquences d'une attaque de fuite de données sont pires encore. Elles peuvent faire chuter fortement les actions de l'entreprise en bourse, chose qui peut changer son destin. En 2018, l'attaque de fuite de données a été classée en premier rang dans l'ensemble des cyberattaques affectant les entreprises à l'échelle mondiale. Par exemple, en mars 2018, Facebook a été confronté à une fuite de données de 87 millions de ses utilisateurs au profit d'une société de profilage politique, ce qui a engendré une chute de 7% (environ 48 milliards de dollars de perte) de ses actions en bourse en une semaine [68]. En même année, la compagnie aérienne British Airways a été touchée également par une fuite de donnée massive suite à un piratage. Des données personnelles et financières ont été dérobées et plus de 380000 cartes bancaires ont été compromises. La compagnie a risqué une amende de 183 millions de livres sterling [61].

Le rapport publié par le ministère de l'intérieur français portant sur l'état de la menace liée au numérique en 2019, estime que le vol de données à partir des Bases de Données (BD) d'entreprises se positionne à la tête des premiers objectifs d'intrusions dans les systèmes d'informations [69]. Souvent gérées par un Système de Gestion de Bases de Données Relationnel (SGBDR), les données récupérées sont réutilisées pour mener des attaques de (Phishing) ou des opérations d'escroquerie à la vente. Le problème est plus grand lorsqu'il s'agit de données à hauts degrés de confidentialité tels que les secrets d'états ou des projets militaires. Le vol de données d'une BD se pratique actuellement de manière très sophistiquée. A l'aide d'un « Cheval de Troie », l'attaquant peut créer des accès illicites via des Backdoors (un compte avec privilèges d'administrateur) et ainsi installer un Botnet qui permet de voler les données de la BD [1]. L'attaquant peut contrôler totalement la machine à distance.

Le développement rapide de la technologie et l'exploitation des vulnérabilités sont les grands défis qui touchent à la sécurité des BD représentant aujourd'hui une préoccupation majeure pour les entreprises. Récemment, ce domaine a fait l'objet de plusieurs études et recherches par la communauté des chercheurs en sécurité informatique dont les objectifs sont d'assurer une protection au profit des données stockées dans la BD centralisée ou distribuée, contre la fuite, la divulgation et la modification face aux menaces internes et externes des attaquants [28], [39].

Les solutions mises en œuvre reposent principalement sur l'établissement de trois niveaux de sécurité : la sécurité physique, la sécurité au niveau du système d'exploitation, et la sécurité au niveau du SGBDR [79]. L'implémentation des deux premiers niveaux est incontournable. Toutefois, la sécurité au niveau du SGBDR via le mécanisme de contrôle d'accès, autrement dit l'authentification et l'autorisation, reste également un moyen utile pour accéder aux données à travers les interfaces systèmes. Cet aspect n'est pas suffisant, car il ne protège que face aux attaques en dehors du système d'information. D'autant plus, il est inefficace contre les bugs, le vol et la destruction des données. Par conséquent, il faut qu'il soit accompagné d'une défense en profondeur que l'établissement met en place à l'aide du chiffrement de la BD [40].

Le chiffrement de la BD est l'une des mesures nécessaires à mettre en place qui renforce davantage la sécurité des données. L'implémentation du chiffrement peut s'appliquer sur les trois niveaux suivants : l'application, le SGBDR, ou le disque dur [79]. Dans le Cloud par exemple, le chiffrement est une opération cruciale qui garantit la confidentialité et l'intégrité des données exportées à des prestataires externes. Plusieurs architectures ont été développées par les chercheurs notamment les travaux des auteurs [14] et [64]. Le chiffrement des BD repose sur le développement de modèle de chiffrement pertinent qui est considéré un des axes de recherche d'actualité. Un modèle de chiffrement pertinent doit répondre à des critères bien définis à savoir, le niveau de la granularité du chiffrement, la gestion des clés de cryptage et la performance [78]. L'apport d'une protection à niveau élevé est fortement lié à la combinaison des trois critères cités précédemment. Plusieurs modèles de chiffrement ont été proposés dans la littérature, parmi eux nous citons le travail des auteurs de [85]. Ce travail a été amélioré ultérieurement par la contribution des auteurs de [79] et [28]. Un autre modèle innovant a été proposé par les auteurs de [77]. Ce modèle est considéré comme une référence dans le domaine. Il englobe les trois critères d'un modèle pertinent de chiffrement de BD.

D'autre part, tout modèle de chiffrement de BD est limité par la politique de la génération et

la protection de ses clés dont la connaissance est le but ultime de tout attaquant. Des approches de protection des clés ont été présentées également par des chercheurs notamment dans les travaux de [39], [45].

Malgré les efforts déployés par les chercheurs pour améliorer les modèles de chiffrement des BD, ces derniers nécessitent d'être encore développés pour qu'ils apportent plus de sécurité. Cette nécessité est fortement corrélée à l'évolution des types d'attaques et des vulnérabilités et aux nouvelles exigences de sécurité. Le développement d'un modèle qui répond parfaitement à la combinaison des trois critères soulevés précédemment est une tâche qui n'est pas simple. L'intégration de cette combinaison dans un seul modèle de chiffrement n'a pas été prise en compte dans beaucoup de travaux proposés dans la littérature. En outre, les modèles de chiffrement proposés sont limités en terme de mécanismes et de fonctionnalités qui doivent renforcer davantage la sécurité des données à savoir, l'utilisation de plusieurs algorithmes et clés pour chiffrer les données, soit en fonction du niveau de granularité du chiffrement ou en fonction du niveau de la sensibilité des données, y compris une approche de protection de clés au sein du serveur de BD pour éviter les coûts et les risques liés à la gestion de cette protection à l'extérieur du serveur.

C'est dans ce contexte que s'inscrit ce travail de thèse, qui vise à améliorer la sécurité des BD via le développement de deux volets principaux : les modèles de chiffrement des BD respectant à la fois les trois critères d'un modèle pertinent de chiffrement, et les modèles de protection des clés de chiffrement des données. Ainsi, la présente thèse sera répartie en cinq chapitres :

Le premier chapitre est divisé en trois sections. Dans les deux premières sections nous allons passer en revue les différents types d'attaques qui compromettent les BD. Nous allons catégoriser les différents types acteurs d'attaques, leurs principales motivations et les moyens qu'ils possèdent. Nous verrons par la suite les moyens techniques intégrés dans les SGBDR qui protègent les BD. La troisième section du chapitre sera consacrée à un rappel sur les notions en cryptographie dont nous aurons besoin au cours de la réalisation des travaux de cette thèse. Nous y profiterons pour présenter le principe de la cryptographie symétrique et ses principales constructions à savoir, le chiffrement par bloc et à flot ainsi que les fonctions de hachage.

Dans le second chapitre, nous allons aborder les différents niveaux de chiffrement des BD et discuter également les points positifs et négatifs de chaque niveau. Par la suite, nous allons

expliquer le principe de base du chiffrement de la BD au niveau du SGBDR. A cet effet, une description des couches qui en sont responsables sera évoquée. En outre, les résultats des premiers travaux réalisés dans le cadre de cette thèse seront exposés dans ce chapitre. Il s'agit d'une étude approfondie sur des solutions de chiffrement des BD qui existent sur le marché, d'une part. D'autre part, une analyse critique et des recommandations à prendre en compte seront discutées. La dernière partie de ce chapitre est dédiée à présenter l'état de l'art des modèles de chiffrement des BD.

Le troisième chapitre se focalisera sur la proposition d'un nouveau modèle global de chiffrement de BD. C'est un modèle développé sur la base d'un concept original que nous avons appelé les 'Classes de chiffrement'. La puissance de ce modèle réside dans sa capacité de répondre parfaitement aux critères exigés par les modèles pertinents de chiffrement de BD, en plus de sa particularité à prendre en charge le chiffrement de la structure de la BD. En effet, il englobe des sous--modèles que nous suggérons d'intégrer au niveau du SGBDR.~~,~~ ~~I~~Leurs fonctions couvrent les aspects de définition du chiffrement sur les colonnes sensibles, le chiffrement des données, la génération et la protection des clés.

L'avant dernier chapitre traite la proposition d'une nouvelle approche de protection des clés de chiffrement dans une BD. En effet, il s'agit de générer des modèles de création de clés principales (Km) à partir de la définition de la structure de la BD. La création de Km se fait de manière automatique lors de la création et la définition d'un chiffrement sur un objet sensible. Chaque modèle généré de Km protège les clés en fonction de la granularité du chiffrement choisit par le SGBDR. Nous suggérons d'implémenter cette approche à l'intérieur du SGBDR.

Finalement, le dernier chapitre portera sur une nouvelle solution de protection des clés de chiffrement au sein des applications contre les attaques internes. Le principe de base de la solution consiste à transformer les requêtes des utilisateurs contenant la clé de chiffrement des données à des nouvelles requêtes. Le processus se fait par appel à distance d'une fonction stockée sur le serveur de BD qui permet de convertir la clé située sur les applications à une nouvelle clé considérée comme la clé réelle de chiffrement et de déchiffrement des données.

In fini, la présentation des travaux de notre thèse s'achève par une conclusion qui rappelle l'ensemble de nos contributions et décrit les perspectives futures de recherche dans le domaine de chiffrement des BD.

CHAPITRE I : SECURITE DES BASES DE DONNEES : ETUDE DES MENACES PRINCIPALES ET DES MOYENS DE PROTECTION

Introduction :

Les bases de données constituent actuellement un actif critique pour l'entreprise suite au nombre d'attaques élevées et les méthodes des menaces qui se développent de jours en jours pour compromettre leur sécurité. L'ouverture sur internet a accru ces risques et rend l'accès non autorisé à une BD une opération quasiment simple. Actuellement une pénétration à une BD dans le web via une injection SQL devient assez simple à l'aide d'outils spécialisés [6], [67]. Différentes menaces peuvent survenir de différentes sources ; il y en a ceux qui proviennent des utilisateurs de confiance de la BD, d'autres de l'extérieur du groupe des utilisateurs, il y en même qui se construisent par conspiration entre les deux. Cependant, les menaces des administrateurs de la BD restent les plus dangereuses [7], [79].

Le vol de données est une attaque dangereuse, notamment si elle vise des données à haut degré de sensibilité. Cependant, les attaques qui touchent à l'intégrité des données peuvent générer des conséquences plus lourdes, malheureusement elles sont difficilement détectables que le vol de données. Une autre forme d'attaque plus méchante est celle qui empêche les utilisateurs légitimes d'accéder aux services d'une BD suite à l'arrêt du serveur. Dans les infrastructures critiques industrielles, les attaques informatiques prennent une autre forme d'ampleur, une attaque de déni de service ou de corruption de données peut causer des dommages physiques inimaginables [38]. Face à toutes ces menaces, il est toujours possible d'éliminer ou de réduire considérablement les risques en déployant les moyens de sécurité nécessaires, en particulier ceux implémentés dans les SGBDR à savoir, l'authentification, le contrôle d'accès, les systèmes d'audit et les outils de chiffrement.

Dans le présent chapitre, nous allons passer en revue les principales menaces de sécurité compromettant les BD. Nous allons exposer le rôle que jouent les mécanismes de sécurité implémentés au sein des SGBDR, en particulier le contrôle d'accès et le chiffrement des données. La deuxième section de ce chapitre sera consacrée à des notions cryptographiques de base qui vont nous servir de lien avec tous ce que nous aborderons dans les prochains chapitres.

I. Les différentes menaces compromettant les Bases de Données

Les BD sont vulnérables à un grand nombre d'attaques plus au moins important. Dans cette section, nous allons identifier et décrire les neuves menaces les plus dangereuses. Il est a souligné que quelques menaces feront l'objet d'une analyse détaillée dans les prochains chapitres.

1. Abus de privilège excessif

L'abus de privilège survient quand un utilisateur reçoit un grand nombre de privilèges qui dépasse ses propres fonctions de travail. Un utilisateur malveillant peut les utiliser abusivement à des fins illégales. Par exemple un utilisateur dans une administration publique qui possède le droit de modifier les informations du personnel peut tirer parti du privilège de mise à jour excessif sur la BD pour modifier des informations salariales [7], [51].

2. Abus de privilège légitime

Un abus de privilège légitime survient quand un utilisateur légitime de la BD abuse avec ses propres droits qui lui ont été accordés de façon légitime à des fins illégales. L'abus de privilège légitime est considéré parmi les menaces les plus dangereuses pour les BD. Les comptes à privilèges élevés accordés à certains utilisateurs peuvent être exploités de façon illégale pour compromettre des BD dans une entreprise. Dans beaucoup de cas, un incident de perte ou de corruption de données suite à un abus de privilèges légitimes est difficilement détectable. L'entreprise est obligée à organiser des audits de sécurité en continu et dépenser des investissements colossaux pour limiter les dégâts et sauver sa réputation et son image de marque [7], [51].

3. Elévation de privilège

Parfois il arrive des situations ou des erreurs et des vulnérabilités logicielles surviennent, l'auteur d'attaque peut en profiter pour modifier ses droits d'accès d'un simple utilisateur à des droits d'un super utilisateur, ceci lui permet de posséder un large spectre de contrôle sur la BD. Les conséquences dans ce cas peuvent être désastreuses, l'attaquant peut désactiver le mécanisme d'audit, transférer des fonds ou détruire des données sensibles sans laisser ses traces [35], [45].

4. Exploitation de failles des Bases de Données vulnérables ou mal configurées

Les BD posent souvent le problème de vulnérabilités, l'évolution de la technologie et des moyens des attaquants oblige les entreprises à apporter régulièrement les correctifs nécessaires au niveau du SGBDR et du système d'exploitation qui le supporte [80]. Avant d'attaquer une BD, la première action mise en œuvre par un attaquant est de tester l'existence de vulnérabilités non corrigées. D'autres vulnérabilités sont reliées à la configuration de la

BD à savoir, les configurations définies par défaut lors de l'installation du serveur de BD. A titre d'exemple, les comptes par défaut Tiger/Scoot et System/Manager.

Le virus SQL Slammer est l'exemple célèbre d'attaque qui a exploité une vulnérabilité sur des BD. En fait, il s'agit d'une faille de dépassement de tampon sur les serveurs SQL Server. Ce virus s'est propagé à la vitesse de la lumière, il a attaqué les serveurs en envoyant une quantité énorme d'adresse IP pour les surchargés en créant des conditions de déni de service. Au-delà de 10000 guichets bancaires sont tombés en panne à travers le monde. Certains pays comme la Corée de Sud, la connexion internet et le réseau téléphonique ont été suspendus sur tout le pays pendant plusieurs heures. La Corée de Sud a subi des pertes financières importantes suite à cet incident [57].

5. Injection SQL

L'attaque par injection SQL est l'une des attaques les plus dangereuses qui menace les BD. Aujourd'hui, il n'est plus demandé d'être un crack en informatique pour la mettre en place, des outils accessibles sur le web et facile à utiliser permettent à n'importe qui de mener cette attaque. Raut et al., ont démontré dans [70] que les conséquences d'une telle attaque sont sérieuses, l'attaquant peut prendre le contrôle total de la BD à distance. Ainsi, il peut falsifier des données à haut degrés de sensibilité, comme il peut les détruire définitivement dans des scenarios catastrophiques. Un attaquant habile et largement expérimenté peut exécuter des scripts à distance en produisant un arrêt complet de la BD et de toutes les applications qui y sont connectées.

L'attaque par injection SQL consiste à injecter une chaine de caractère dans une instruction SQL ou de modifier sa structure afin qu'elle devienne vulnérable. Elle est exécutée par la suite dans un formulaire web d'une application ou dans un éditeur de ligne de commande [67]. L'attaque qui a eu lieu envers le site internet officiel de Volkswagen Maroc en 2013 est un meilleur exemple qui illustre l'ampleur de cette attaque. Un pirate présenté sous le nom de « XIIV » a exploité une faille critique du site web de Volkswagen en accédant à la BD de la société. Le pirate a extrait des informations à haute degrés de confidentialité à savoir, les comptes des administrateurs du site, les informations personnelles des clients et les identifiants de tous les serveurs de l'entreprise. Le pirate a diffusé toutes ces informations sur le site PasteBin [46].

6. Faiblesse de l'audit natif

Une politique d'audit de BD permet d'assurer un suivi automatisé et ponctuel des transactions sensibles ou inhabituelles dans une BD, elle doit figurer en premier lieu dans la politique globale de sécurité d'une entreprise. Le mécanisme d'audit est considéré le dernier mur de défense dans le cas où l'attaquant arrive à contourner tous les systèmes de défense mis en place. Ce mécanisme a une importance majeure et décisive dans l'identification d'une violation d'une BD. L'absence ou la faiblesse de ce mécanisme peut causer de sérieux problèmes à des niveaux distincts [34], [51], [80].

7. Déni de service

En général, le déni de service (DOS : Denial Of Service) est une typologie d'attaque qui empêche les utilisateurs légitimes d'accéder partiellement ou complètement aux services d'une BD. En effet, il existe plusieurs techniques pour créer un DOS, la surcharge en ressource du serveur de BD, l'injection SQL et l'attaque de corruption de données sont des techniques très utilisées par les pirates pour créer un DOS. D'autres techniques sophistiquées déploient des vers informatiques telle que la fameuse attaque « Stuxnet » qui a ciblée la centrale nucléaire Iranienne de Bushehr [17], [38].

La surcharge en ressources d'un serveur de BD est actuellement la technique la plus simple pour créer un DOS. Son principal but consiste à consommer les ressources du serveur pour pouvoir l'empêcher de rendre ses services de façon normale. En effet, l'attaquant submerge la machine par une série de requête afin de la planter. Dernièrement, les attaques DOS ont été évoluées pour prendre une autre forme très puissante, celle du déni de service distribué (DDOS : Distributed Denial Of Service). Ce type d'attaque est assuré par une multitude de machines Zombies [1].

8. Vulnérabilités des protocoles de communication des Bases de Données

Un grand nombre de vulnérabilités de sécurité a été identifié dans les protocoles de communication des BD, ces vulnérabilités se multiplient en fonction des mises à jour et des nouvelles versions mises sur le marché par les fournisseurs des SGBDR. Les correctifs lancés par ces derniers couvrent dans plusieurs cas des failles de protocoles de BD. Les auteurs d'attaques peuvent exploiter ces vulnérabilités pour cibler des machines et mener des attaques de corruption de données, d'accès non autorisé et des dénis de service [21].

9. Exposition de données de sauvegarde

L'exposition des dispositifs de sauvegarde des BD est une menace très importante qu'il faut prendre en compte. Les données de sauvegarde sont souvent non sécurisées contre les attaques informatiques à savoir la destruction et le vol, ou exposées à des catastrophes naturelles tels que les inondations et les séismes [52].

II. Protection technique des Bases de Données

Nous avons passé en revue les principales menaces liées à la sécurité des BD. Bien évidemment une description plus détaillée de ces menaces peut prendre tout le volume de cette thèse, nous nous sommes limités à une description générale du principe de base de chaque menace. Avant d'entamer le sujet sur les moyens de protection, il faut tout d'abord comprendre le risque de manière précise, identifier les acteurs d'attaques, connaitre les motivations des attaquants et les moyens qu'ils possèdent.

1. Les acteurs d'attaques

Afin d'identifier les acteurs d'attaques, il vaut mieux avant tout les cerner de point de vue système d'informations, cela va nous permettre de bien saisir le lien solide qui existe entre les deux, autrement dit entre les acteurs d'attaques et les acteurs systèmes d'informations.

La notion d'un acteur système d'informations identifie tout simplement les utilisateurs directs et indirects d'un système d'informations, ceux qui interagissent avec. Il regroupe les éléments suivants [40] :

- Les propriétaires de données : ce sont les personnes qui produisent ou possèdent des données dans le serveur de la BD, elles ont un accès à distance au serveur avec des droits limités qui leurs sont attribuées en fonction de leur degré de confiance et du contenu qu'elles gèrent dans la BD. Ces personnes ne possèdent pas d'accès physique au serveur qui héberge les données.
- Les clients : ce sont les personnes qui peuvent avoir des données ou non sur le serveur de la BD. Les clients possèdent des droits mois importants sur la BD et la façon d'y accéder se fait également à distance après un processus d'authentification.
- Les administrateurs de la BD : ce sont les personnes qui assurent la gestion de la BD. Les rôles d'un administrateur de BD sont multiples, il installe et configure les BD, gère les droits d'accès, contrôle les activités des utilisateurs et assure les sauvegardes. L'administrateur de la BD possède tous les droits relatifs à la gestion de la BD sauf les

droits de la machine, c'est à dire du serveur lui-même.

- L'administrateur de la sécurité de la BD : dernièrement, cette fonction est nouvellement créée par les entreprises lorsque la valeur des données stockées dans les BD est très sensible, ou lorsque les personnes qui s'occupent de l'administration des BD sont non fiables. La fonction d'un administrateur de sécurité est dérivée de l'administrateur de la BD, elle consiste à gérer uniquement les taches de la sécurité. Bouganim et al., a mis en évidence cette fonction dans le cas si on souhaite sécuriser les clés de chiffrement à l'extérieur du serveur de BD [45].
- L'administrateur du serveur : c'est la personne qui possède tous les accès et droits sur le serveur de la BD.
- L'hébergeur de la BD : parfois l'entreprise souhaite externaliser la gestion de ses BD à un hébergeur externe dans le but de minimiser les couts de gestion et de maintenance de ses serveurs. Les employés de l'hébergeur peuvent disposer d'un accès direct aux données hébergées. Un technicien de maintenance par exemple peut ne pas disposer d'un accès à distance sur les BD hébergées, par contre il dispose forcément d'un accès physique en vue d'effectuer des réparations sur la machine.

Il parait évident que les différents acteurs système d'informations sont le facteur numéro 1 qui menace la sécurité d'une BD. Les types d'accès qu'ils possèdent sur les données reflètent les vulnérabilités qui y sont liées. La menace peut survenir non seulement de ces acteurs, les attaques en dehors du système d'informations (les hackers, les auteurs de malware, les organisations malveillantes, …etc.) sont aussi dangereuses, elles représentent environ les deux tiers des violations des BD.

En effet, les acteurs d'attaques peuvent être repartis selon trois classes : les acteurs d'attaques internes, externes et administrateurs [28], [79].

- Acteur d'attaque interne : c'est une attaque qui s'effectue par une personne (ou groupe de personnes) qui appartient au groupe des utilisateurs de confiance de la BD, il tente d'obtenir des informations au-delà de ses propres droits d'accès [10].
- Acteur d'attaque externe : c'est une attaque qui s'effectue par une personne (ou groupe de personnes) qui se situe à l'extérieur du groupe des utilisateurs de la BD, il possède un accès sur un système informatique et tente d'extraire des informations sensibles [10], [86].

- Acteur d'attaque administrateur : c'est la personne qui s'occupe de l'administration de la BD, il possède des droits et des privilèges spécifiques sur elle, bien qu'il les exploite pour extraire des informations sensibles et précieuses [64].

Un acteur d'attaque peut appartenir à une classe ou plus. Un pirate externe peut par exemple s'octroyer les droits d'un utilisateur de confiance de la BD, il peut ainsi manipuler et contrôler la BD à distance identiquement à un utilisateur légitime. Un pirate habile arrive même à obtenir les droits de l'administrateur.

2. Les principales motivations des attaquants

Il est intéressant de comprendre ce que motive les attaquants quand ils pratiquent un acte malveillant. Les sources de motivation en effet sont multiples, changent d'une personne à une autre. Un attaquant trouve son désir à exploiter un système en recherchant ces failles et ces limites, d'autres trouvent leurs sources de motivation dans la vengeance ou l'appât du gain.

Dans son livre référence qui s'intitule « La cybercriminalité au Maroc », le célèbre expert marocain en sécurité informatique Dr Ali El Azzouzi identifie quatre grandes classes de motivations : l'argent, La curiosité intellectuelle, L'ego et l'idiologie [1].

- L'argent : c'est la principale motivation qui attire les attaquants. Dernièrement, ce phénomène commence à prendre de l'ampleur grâce au développement de la technologie du Blockchain, particulièrement les crypto monnaies.
- La curiosité intellectuelle : beaucoup d'attaquants portent un grand désir dans l'exploration des limites d'un système informatique, leurs motivations ultimes est de rechercher des systèmes parfaits et sans failles.
- L'égo : plusieurs attaquants exercent des activités de piratage pour satisfaire leurs égos. Lorsqu'un attaquant divulgue des données confidentielles d'une entreprise dans le web, il attend un retour de la part des autres pirates pour lui étiqueter le label de la force et de l'intelligence.
- L'idiologie : plusieurs actes de piratage motivés par des considérations religieuses ou politiques ont été recensés ces dernières années. En 2009, un pirate qui porte le nom de Ibn Al Walid a détruit complètement la BD et le site web de l'association marocaine Kif Kif qui rassemble des homosexuels, il a publié sur la page d'accueil du site des versets coraniques encourageant l'assassinat des homosexuels [1], [81].

Une fois que nous avons identifié les motivations des attaquants, il est essentiel de les appréhender par rapport aux objectifs de la sécurité à savoir, la confidentialité, l'intégrité et la disponibilité. En effet, les objectifs d'attaques sur les BD reposent sur trois axes principaux : l'atteinte à la confidentialité, l'intégrité et à la disponibilité de la BD [24], [45].

a. L'atteinte à la confidentialité de la Base de Données

Le vol ou la divulgation des données sont deux expressions qui portent le même visage dans l'atteinte à la confidentialité d'une BD. Les attaques qui visent la confidentialité peuvent avoir des motifs variés, un simple exemple est celui des entreprises de marketing qui cherche à acheter illégalement des grandes BD piratées contenant les coordonnées personnelles du public (nom et prénom, n° de téléphone, email etc.). Ces entreprises utilisent ces données pour contacter un public ciblé afin de leur proposer des offres commerciales.

b. L'atteinte à l'intégrité de la Base de Données

L'atteinte à la confidentialité n'est pas la seule option envisageable pour attaquer une BD, un effacement ou une falsification de données peut représenter un grand intérêt pour l'attaquant. Un acte malveillant qui touche à l'intégrité d'une BD peut donner lieu à des conséquences désastreuses. En effet, l'effacement ou la modification de données peut servir à cacher des erreurs ou à nuire complètement des systèmes.

c. L'atteinte à la disponibilité de la Base de Données

L'atteinte à l'accessibilité des données est une typologie d'attaque qui devient de plus en plus à la mode. Il s'agit des attaques DOS et DDOS qui peuvent aller de l'altération du fonctionnement du système jusqu'à son arrêt complet.

3. Les moyens des attaquants

En fonction de la cible attaquée, l'attaquant choisit les moyens et les outils qu'il le convient. Les moyens des attaquants sont variés, ils se diffèrent par rapport au niveau où l'acteur se positionne. Leurs efficacités dépendent de leurs capacités à cacher les traces et d'éviter que nous remontions à l'attaquant le plus facilement possible. Généralement, nous pouvons classer les moyens des attaquants selon deux catégories : les moyens d'accès physique et les moyens d'accès à distance.

Dans le cas où l'attaquant possède un accès physique sur la machine, l'opération est quasiment simple, le système de contrôle d'accès n'est plus un élément bloquant. Dans les attaques à accès distant, le degré de complication diffère d'un accès à l'autre. L'attaquant peut

être obligé de franchir plusieurs barrières de sécurité pour atteindre la cible, il peut s'appuyer sur des outils spéciaux et des programmes malveillants pour détourner des réseaux, casser des mots de passes, ou de profiter des vulnérabilités qui existent. La liste des moyens des attaquants est en fait très large, nous allons se limiter de citer trois outils que nous estimons les plus intéressants : les Botnets, les Rootkits et les Keyloggers.

3.1. Les Keyloggers

Un Keylogger est un logiciel qui enregistre les frappes du clavier d'un utilisateur. En effet, les fonctionnalités des Keyloggers sont évoluées pour réaliser des opérations très compliquées à savoir, surveiller les clics de la souris et faire des captures d'écran. Les Keyloggers sont des outils très dangereux lorsqu'ils sont utilisés pour attaquer les BD. Les attaquants peuvent les déployer pour récupérer des mots de passes de connexion à une BD et accéder à des données non autorisées [2]. Le Keylogger est un outil de malware qui s'infiltre dans les machines pour se cacher, le problème est que l'antivirus n'arrive pas souvent à le détecté. Contrairement au malware, il ne présente aucun risque sur le système. Lors de démarrage de la machine, le Keylogger s'active et commence à enregistrer toutes les frappes du clavier dans un fichier crypté. Ce fichier sera envoyé par la suite vers une adresse mail ou un serveur Web. Plusieurs modes opératoires sont utilisés pour installer un Keylogger sur une machine à distance à savoir, les chevaux de Troie [2].

3.2. Les Rootkits

Les Rootkits font partie de la catégorie des malwares, ils sont conçus pour infecter une machine en permettant à l'attaquant d'installer un ensemble d'outils qui lui favorise d'accéder à distance à cette machine. Similairement que les Keyloggers, les Rootkits sont difficilement détectables par les logiciels anti-virus et autres outils de sécurité. Le danger extrême d'un Rootkit réside dans son action sur un outil de sécurité, il peut l'empêcher de s'exécuter de manière directe par le fait de lui cacher des informations. Un Rootkit peut se présenter sous forme d'un pack qui contient plusieurs outils à savoir, un programme de vol de données, un enregistreur de frappe de clavier, un programme de vol des mots de passes, un programme automatisé d'effacement des traces sur une machine. Pire encore, un Rootkit peut être conçu en tant que robot spécialiste dans les attaques DDOS [1].

Les Rootkits sont des éléments hyper puissants lorsqu''ils sont utilisés pour attaquer les BD. Un simple exemple est celui de leurs utilisations en tant que portes dérobées, ils peuvent agir typiquement à ces derniers en permettant à un attaquant de se connecter facilement à une machine. L'attaquant peut ainsi exploiter la fonctionnalité d'enregistreur de frappe pour

récupérer les mots de passes d'authentification des utilisateurs légitimes et ainsi manipuler la BD tout en effaçant ses traces à la fin de l'opération.

3.3. Les Botnets

Un Botnet est un ensemble d'ordinateurs zombies connectés entre eux et contrôlés par un pirate à distance pour commettre des actions malveillantes comme l'envoi de Spam, le vol de données, la propagation de virus et les attaques DOS. Les réseaux zombies peuvent rassembler plusieurs ordinateurs voir des dizaines de millions d'ordinateurs à l'échelle d'un pays ou même du monde entier. La redoutable utilisation de ces réseaux d'ordinateurs consiste à profiter de leur temps de calcul pour casser des clés de cryptage utilisées dans la protection des BD, ou des clés de sécurité de certains protocoles de communication. Un Botnet est établit en faisant infecter au départ quelques dizaines d'ordinateurs (appelés maitres), qui réagiront à leur tour pour infecter d'autres ordinateurs (les zombies). Le nombre des zombies peuvent atteindre des milliers voir des millions d'ordinateurs. Dans ce type d'attaque, le créateur de Botnet se retrouve dans l'anonymat absolu, il est impossible de retrouver ses traces parmi un nombre énorme d'ordinateurs [1].

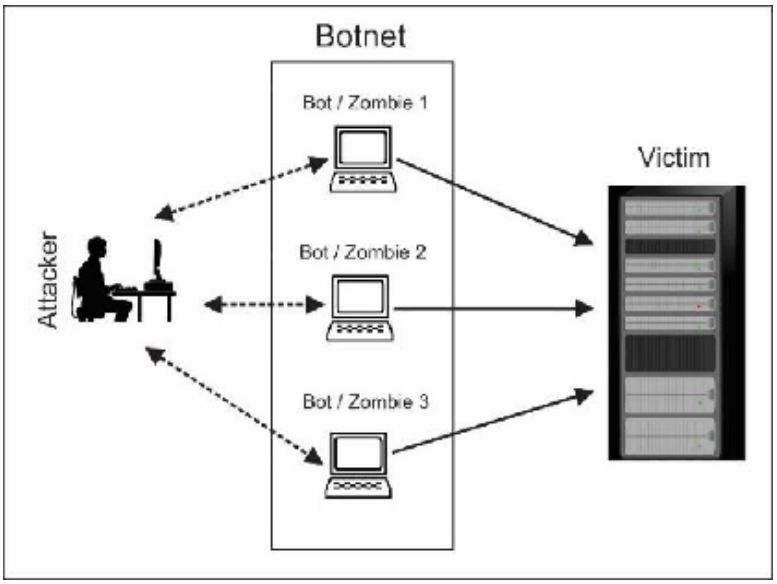

Figure 1.1: La structure d'un Botnet [1].

4. Les moyens de protection des Bases de Données

Afin de faire face à toutes ces menaces de sécurité, les entreprises sont obligées de mettre en place les moyens organisationnels et techniques nécessaires pour sécuriser au maximum leurs données. La mise en place de ces moyens doit être bien étudiée par une équipe de spécialiste dans le domaine.

La sécurisation des BD repose sur l'établissement des trois niveaux de sécurité suivants : sécurité au niveau physique, sécurité au niveau du système d'exploitation et sécurité au niveau du SGBDR. Les deux premiers niveaux ne suffisent pas pour garantir une sécurité extrême, ils peuvent être contournés facilement comme on l'a démontré précédemment. L'implémentation du troisième niveau (sécurité au niveau du SGBDR) est incontournable, il englobe le mécanisme du système de contrôle d'accès, l'identification et l'authentification des utilisateurs, le mécanisme d'audit et le cryptage des données.

4.1.L'identification et l'authentification des utilisateurs

Avant toute tentative d'accès à une BD, une phase d'identification est nécessaire, l'utilisateur doit posséder un identifiant unique ou « Compte d'accès » personnel pour y accéder. L'authentification est la seconde phase qui permet à l'utilisateur de donner la preuve de son identité en s'authentifiant par un mot de passe [7], [22].

4.2. Le mécanisme d'audit

Parallèlement au processus d'identification et d'authentification, le mécanisme d'audit peut jouer un rôle crucial dans la sécurité de la BD, il sert à tracer les accès et les actions des utilisateurs sur les objets de la BD dans des journaux « Log ». Ce mécanisme peut servir également dans l'identification et la confirmation de l'existence d'une attaque ainsi que la détection du lien entre une violation et un utilisateur. Dans une attaque visant l'intégrité des données, le recours au mécanisme d'audit est incontournable pour réparer rapidement le système après l'attaque [7], [22], [37].

4.3. Le système de contrôle d'accès

Le système de contrôle d'accès est le mécanisme permettant de contrôler l'accès des sujets (utilisateurs) sur les objets de la BD. C'est la première muraille de défense contre les accès non autorisés. En effet, il existe trois principaux modèles de contrôle d'accès : le modèle de contrôle d'accès obligatoire, le modèle de contrôle d'accès discrétionnaire et le modèle de contrôle d'accès basé sur les rôles [37], [40], [41].

a. *Contrôle d'accès obligatoire (MAC)*

C'est une politique définit au niveau du système, elle est multi niveaux, elle accorde aux utilisateurs et aux objets respectivement des niveaux d'accréditations et de classifications. L'objectif du MAC est de lutter contre les « chevaux de Troie »

b. Contrôle d'accès discrétionnaire (DAC)

Dans le modèle (DAC), celui qui crée un objet dans la BD définit la politique du contrôle d'accès sur cet objet. Le créateur de l'objet peut aussi déléguer quelques permissions à d'autres utilisateurs.

c. Contrôle d'accès basé sur les rôles (RBAC)

Un ensemble de privilèges associés à une fonction bien déterminée appelée rôle sont attribués par l'administrateur de la BD aux utilisateurs. L'accès à la BD est contrôlé donc en fonction de ces rôles.

Malgré l'utilité évidente du système de contrôle d'accès dans le renforcement de la sécurité d'une BD, ce mécanisme n'est pas suffisant, il ne protège que face à des attaques situées en dehors du système d'informations, comme il peut engendrer les limites suivantes [40] :

- Protection inefficace contre les bugs : c'est la faille potentielle d'un système de contrôle d'accès qui touche le système de gestion des accès.
- Protection inefficace contre le vol : le mécanisme de contrôle d'accès permet de limiter les accès numériquement, cela n'interdit pas le vol d'un disque dur par exemple en moyennant un support de stockage auxiliaire. L'accès physique d'un attaquant sur une machine est typiquement similaire à un accès administrateur sur cette machine.
- Protection inefficace contre une destruction : le mécanisme de contrôle d'accès est inefficace contre un accès physique qui a pour vocation la destruction du disque, cela le prouve certaines attaques de type déni de service où l'attaquant écrase tous les fichiers de données stockés sur le serveur de BD.

Il parait évident que le système de contrôle d'accès même on le combinant avec les autres moyens de protection ne peut satisfaire une sécurité extrême à la BD. Les données sont conservées sous une forme lisible. Toute personne malveillante (attaquant interne, externe y compris l'administrateur) arrivant à accéder à la BD est capable de lire les données. D'autant plus, les fichiers de données de la BD sont sauvegardés fréquemment, l'accès à ces derniers doit également être contrôlé. Il est nécessaire alors d'établir une autre couche de sécurité pour consolider davantage la protection des données. Une solution consiste à adopter une défense en profondeur à la BD en déployant les solutions de chiffrement des données.

4.4. Le chiffrement de la Bases de Données

Le chiffrement des données est le processus qui transforme des données brutes dites (données claires) vers des données sous formes incompréhensibles dites (données chiffrées) en utilisant un algorithme de chiffrement et une clé secrète [8], [23], [33]. Le chiffrement d'une BD est le processus qui transforme les données claires qui proviennent des utilisateurs vers des données chiffrées en vue de les stocker dans la BD en moyennant un algorithme de chiffrement approprié et une clé secrète. Lorsque l'utilisateur souhaite extraire les données à partir de la BD, elles sont déchiffrées via un algorithme de déchiffrement et la même clé secrète [23], [43].

La Figure 1.2 ci-dessous schématise le fonctionnement de base d'un processus de chiffrement/déchiffrement dans une BD.

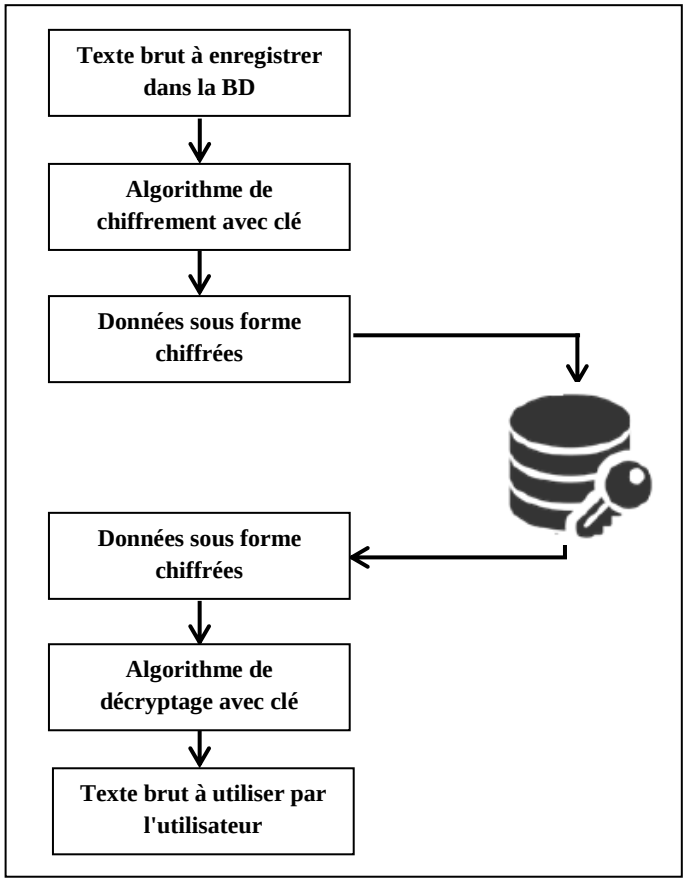

Figure 1.2: Processus du chiffrement/déchiffrement dans une BD.

Avant d'aborder le sujet du chiffrement des BD qui représente le noyau de cette thèse, il nous a paru essentiel de rappeler quelques notions en cryptographie, cela va nous aider à comprendre profondément tous les aspects que nous traitons dans les prochains chapitres. Certes, nous n'allons pas approfondir nos connaissances en la matière mais nous allons introduire les notions de base que nous allons utiliser tout au long de cette thèse.

III. Notions en cryptographie

La science du secret, c'est le surnom donné à la cryptographie, cette science très ancienne a pour objectif la protection de l'information. La cryptographie regroupe deux branches principales, la cryptographie et la cryptanalyse.

La cryptographie a pour but de concevoir des mécanismes de protection de l'information qui transite à travers un canal non fiable face aux différents types d'attaques et d'attaquants. La cryptanalyse est par contre l'ensemble des techniques qui cherchent à trouver les vulnérabilités dans ces mécanismes afin de les affaiblir [12]. Les systèmes cryptographiques datent depuis l'antiquité, le système cryptographique « Scytale » par exemple a été utilisé en Grèce en 487 avant J.C, d'autres systèmes anciens ont marqués l'histoire de l'humanité tels que le « carré de Polybe », le crypto-système de « César » (Rome) et le crypto système de « Vigenère » [12].

La cryptographie a joué un rôle déterminant sur le plan militaire au 20 ème siècle, notamment dans les deux guerres mondiales, l'utilisation de la machine ENIGMA par les allemands et le rôle qu'elle a apporté durant cette guerre est un véritable témoin. Jusqu'au nos jours, la cryptographie reste une science qui contribue énormément au développement humain dans plusieurs secteurs, notamment en sciences de l'information. Elle est largement utilisée dans la protection de la vie privée sur internet et la sécurisation des transactions bancaires. À vrai dire, dans toutes les communications qui nécessitent un niveau élevé de sécurité.

En effet, la cryptographie moderne a pour but d'assurer les exigences suivantes [11], [12] :

- La confidentialité : la protection de la confidentialité d'un message est l'opération de rendre incompréhensible un message aux entités non autorisées.
- L'intégrité : c'est la propriété qui assure que les messages échangés entre l'expéditeur et le destinataire n'ont pas été modifiés par des entités non autorisées.
- L'authentification : cette propriété assure au destinataire que le message reçu provient du vrai expéditeur.
- La non répudiation : il permet de garantir qu'une transaction (émission ou réception) d'un message ne peut être niée.

Généralement, il existe trois types de cryptographie : symétrique, asymétrique et hybride [11], [12], [16].

La cryptographie asymétrique s'appuie sur l'utilisation de deux types de clés, la première clé

est privée l'autre est public. La clé privée sert à déchiffrer les données et la clé publique chiffre les données. La cryptographie hybride est la combinaison des deux, elle comporte la cryptographie symétrique et asymétrique. La cryptographie symétrique repose sur l'utilisation d'une seule clé pour chiffrer et déchiffrer les données. D'ailleurs, c'est le type de cryptographie qui nous intéresse dans le chiffrement des BD.

1. La cryptographie symétrique

La cryptographie symétrique, appelée aussi cryptographie à clé secrète, est une forme de cryptographie qui repose sur l'échange d'une clé secrète entre un émetteur et un destinataire. La clé est utilisée en même temps pour chiffrer et déchiffrer une information secrète. Les systèmes de chiffrement qui adoptent ce mode de fonctionnement sont appelés des crypto-systèmes symétriques, ils sont formés d'une paire d'algorithmes : un algorithme de chiffrement E et un algorithme de déchiffrement D [11], [16], [42]. Le fonctionnement d'un crypto-système symétrique est censé être public sans que la sécurité soit impactée, la sécurité repose plutôt sur la clé de chiffrement [12], [42], [89]. Avant de procéder à la description des différentes constructions de la cryptographie symétrique, il est primordial de définir avant tout ces types d'algorithmes.

Soit M un message clair et K une clé de chiffrement, le chiffrement de M se déduit comme suit :

$$E\ (K, M) = E_K(M) = C$$

C est le message chiffré obtenu par le chiffrement de M. Pour déduire le message d'origine M à partir de C, nous procédons comme suit :

$$M = D\ (K, C) = D_K(C)$$

La Figure 1.3 ci-dessous montre le fonctionnement d'un système de chiffrement symétrique.

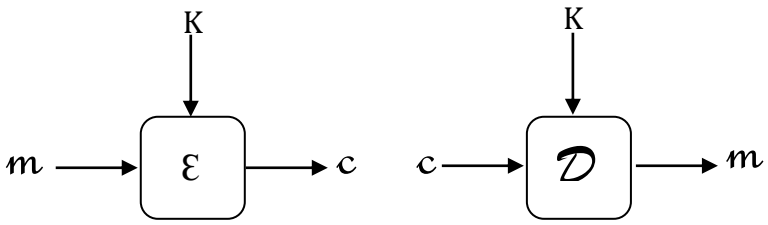

Figure 1.3: Système de chiffrement symétrique [12].

L'algorithme D ici est obligatoirement déterministe, chaque couple (K, C) est associé à un seul message M. En revanche, il est possible que plusieurs messages chiffrés soient associés à un couple (M, K) unique dans le cas où le message est randomisé ou muni d'un état interne

[40]. La clé secrète échangée entre l'émetteur et le destinataire est le problème fondamental de l'implémentation des systèmes de chiffrement symétriques, elle requière une protection efficace contre les attaques d'adversaires. En fait, il existe deux grandes classes de la cryptographie symétrique : les systèmes de chiffrement par bloc et les systèmes de chiffrement à flot.

2. Le chiffrement par bloc

Un système de chiffrement par bloc utilise une transformation inversible qui prend en entrée des blocs de texte clair de taille fixe (qui peut valoir 64, 128, 160 ou 256 bits), une clé de chiffrement K et il renvoie des blocs de texte chiffré de taille fixe. Généralement la taille du bloc d'entrée est égale à la taille du bloc de sortie chiffrée [3].

Le chiffrement du message clair M d'origine se fait en le divisant en bloc fixe de n bits, chaque bloc est chiffré à l'aide d'une fonction définie avec une clé K de taille k bits. Le déchiffrement s'effectue en appliquant l'inverse de la fonction de chiffrement qui prend en entrée le texte chiffré et la clé K [12] [42].

On peut décrire mathématiquement un chiffrement par bloc par la transformation :

$$E : \{0,1\}^n * \{0,1\}^k \longrightarrow \{0,1\}^n$$

Avec chaque clé K appartenant à $\{0,1\}^k$ et M un texte clair, nous avons E (M, K) est une bijection inversible de $\{0,1\}^n$ à $\{0,1\}^n$ que nous notons $E_K(M)$.

Un algorithme de chiffrement par bloc fonctionne en faisant itérer une fonction de tour f. Cette fonction se change d'un tour à un autre en utilisant des sous clés formulées à partir de la clé K. La génération des sous clés se fait par un algorithme appelé « Cadencement de clés ». Identiquement, plusieurs paramètres de la fonction f (constantes, …) peuvent changer également d'un tour à un autre [3], [12]. Il existe deux méthodes principales pour construire un chiffrement par bloc : les réseaux de substitution-permutation (SP networks) et les schémas de Feistel.

2.1.Les réseaux de Feistel

Cette une construction inventée par le célèbre cryptologue allemand Horest Feistel, elle repose sur le découpage du bloc m en deux parties de taille égale L0 et R0 (Figure 1.4). Au moment de chaque tour, la fonction f transforme {Li, Ri} en {Li+1, Ri+1} de la manière suivante [3], [11] :

$$Li+1 = Ri$$

$$Ri+1 = Li \oplus f(K, Ri)$$

Avec un réseau de Fiestel nous utilisons le même chemin pour chiffrer et déchiffrer, il est donc inversible. Le déchiffrement se déduit comme suit :

$$Li = Ri+1 \oplus f(K, Li+1)$$

$$Ri = Li+1$$

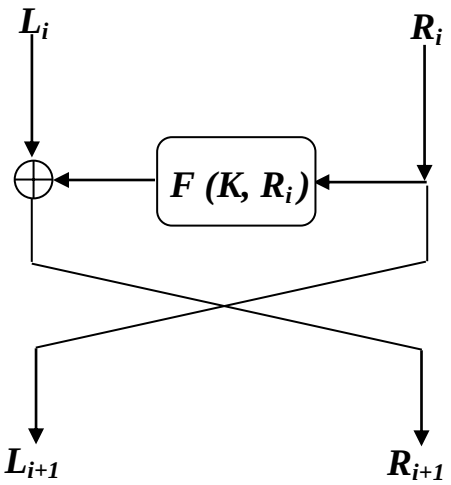

Figure 1.4: Un tour de chiffrement Feistel [11].

La sécurité dans un chiffrement par bloc basé sur un réseau de Fiestel est garantie par un grand nombre de tours.

2.2.Les réseaux de substitution-permutation

Dans une construction par réseau de substitution-permutation, la fonction f est constituée de substitutions et de permutations des bits du bloc, d'un nombre successif de tours d'ajout de clé et d'application d'une fonction non linéaire (Boîtes-S) et linéaire comme indiqué dans la Figure 1.5. L'algorithme AES est un exemple d'algorithme qui adopte cette construction. A vrai dire, il est considéré le standard actuel des algorithmes de chiffrement par blocs.

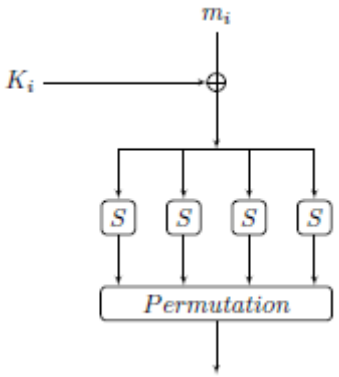

Figure 1.5: Exemple de tour du réseau substitution-permutation [11].

3. Algorithmes de chiffrement par bloc

Afin de bien illustrer les deux types de chiffrement par blocs basés sur les réseaux de substitution-permutation et les réseaux de Fiestel, nous allons décrire ci-dessous les deux algorithmes par blocs les plus connus en cryptographie : AES et DES.

DES (Data Encryption Standard) : Il a été publié en janvier 1977, c'est un système de chiffrement par bloc de 64 bits qui utilise un réseau de Feistel à 16 tours et une clé de 56 bits stockée avec 8 bits de parité sur 64 bits [32]. Sans tester toutes les clés possibles, le DES a été cassé plusieurs fois avec plusieurs types d'attaques. La première tentative qui date de 1991 a été menée par Eli Biham et Adi Shamir, ils ont trouvé la clé à partir de 2^{47} couples clairs-chiffrés choisis en utilisant leur propre méthode appelée cryptanalyse différentielle [25]. En 1993 Mitsuru a publié une méthode appelée cryptanalyse linéaire qui permet de retrouver la clé à partir de 2^{43} couples clairs-chiffrés [49]. En 1998 l'Electronic Frontier Fondation (EFF) a spécialement conçu une machine qui permet de retrouver la clé en 3 jours, ce temps a été réduit par la suite en moins d'une journée en utilisant une autre machine plus puissante. Suite à cela, le standard du DES a été retiré par le National Institute of Standards and Technology (NIST) en 2005. L'amélioration du DES a été considérée une nécessité à l'époque, le premier résultat de réflexion a donné naissance d'un double DES, il n'apportait pas vraiment une grande sécurité, le cout d'une attaque sur un double DES était 2^{57}, il a été enchainé après par un autre DES en donnant un 3DES dans l'ordre de DES / DES-1 / DES avec 112 bits de clé, cela a amélioré le cout d'attaque à 2^{113}.

AES (Advanced Encryption Standard) : c'est un système de chiffrement par bloc qui a été normalisé officiellement en novembre 2001 [31]. Il est développé au début par Joan Daemen et Vincent Rijmen sous le nom de Rijndael. Le but derrière le développement de l'AES était de remplacer le DES qui est devenu faible à l'époque. L'algorithme AES utilise un réseau de substitution-permutation, il opère sur des blocs de texte clair de 128 bits qu'il les transforme en blocs chiffrés de 128 bits avec une succession de Nr opérations ou « rounds » et des clés de 128, 192 ou 256 bits. En fonction de la taille des clés, le nombre de « rounds » change, il vaut respectivement 10, 12 et 14 rounds. La Figure 1.8 illustre le déroulement du chiffrement AES.

Pour simplifier le fonctionnement d'un AES128, nous représentons un bloc de 128 bits sous forme de matrice carré comme indiqué à la Figure 1.6. La fonction linéaire ByteSub permet de transformer chaque $a_{i,j}$ en $S(a_{i,j})$. La fonction ShiftRows permet d'appliquer à son tour des

décalages aux lignes de l'état interne comme illustré à la Figure 1.7. De même la fonction MixColumns mélange les octets qui se situent sur une même colonne. A la fin, l'algorithme ajoute la clé de « rounds » K_i (sous clé calculée à partir de la clé principale) via la fonction AddRoundKey, elle change chaque $a_{i,j}$ en $a_{i,j}$ XOR $K_{i,j}$, avec $K_{i,j}$ représente l'octet dans la sous-clé du tour concerné. La Figure 1.8 résume globalement le déroulement du chiffrement AES [31].

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

Figure 1.6: Etat interne d'AES [31].

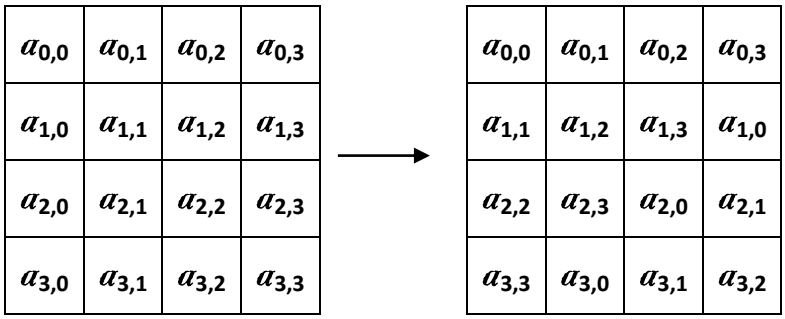

Figure 1.7: Fonction ShiftRows d'AES [31]

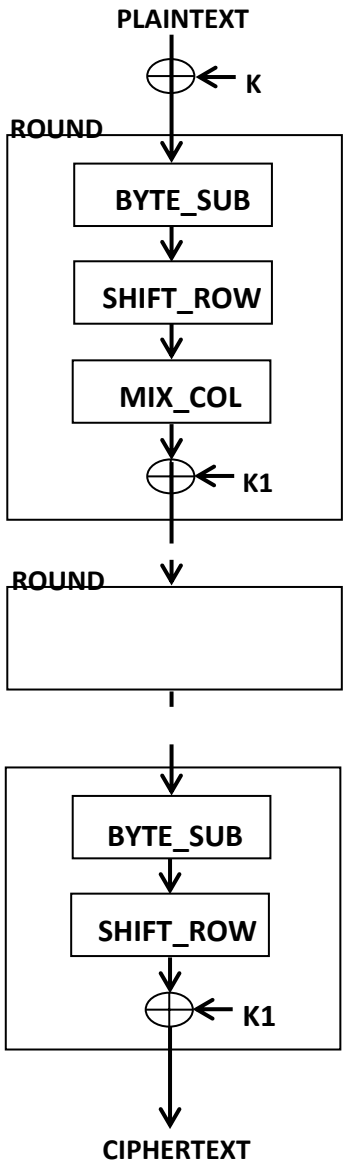

Figure 1.8: Le déroulement du chiffrement AES [11].

Dans l'ensemble des algorithmes de chiffrement symétriques, AES est considéré actuellement l'algorithme le plus sûr et le plus robuste même contre le 3DES. Il offre une grande résistance face aux attaques par cryptanalyse différentielle et par cryptanalyse linéaire. En outre, AES fourni une grande rapidité de traitement et un besoin en ressources mémoires très faible, ce qui rend son implémentation possible sous forme logicielle que matérielle [40].

4. Notion de sécurité des algorithmes de chiffrement par bloc et attaques cryptographiques

L'évaluation de la sécurité d'un algorithme de chiffrement par bloc ne doit pas se reposer sur le fonctionnement interne de l'algorithme, il suppose que le fonctionnement de ce dernier est

connu et sa sécurité est plutôt liée à la clé de chiffrement. La connaissance de la clé est l'objectif de l'attaquant, il peut déchiffrer tous messages chiffrés avec celle-ci. Cependant, l'attaque de détention de la clé n'est pas le seul scenario envisageable par un attaquant, il existe en effet plusieurs niveaux de sécurité qui s'appliquent aux algorithmes de chiffrement par blocs. Nous allons donner ci-dessous les définitions suivantes [40], [42] :

Définition 1.1. Non-inversibilité (OW: one-wayness). Soient un algorithme de chiffrement non-inversible E et un message clair m. Il est impossible de trouver m à partir de E (K, m) sans connaître la clé K.

Définition 1.2. Sécurité sémantique (IND: indistinguishability). Soit un adversaire ayant une puissance de calcul polynomiale, c'est-à-dire qu'il ne peut faire qu'un nombre polynomial de calculs et un chiffré. Un chiffrement sûr sémantiquement assure que l'attaquant ne peut déduire aucune information sur le clair à partir du chiffré.

Définition 1.3. Non-malléabilité (NM: non-malleability). Soient E un algorithme de chiffrement non-malléable, M un message clair et C = E (K, m) le chiffré de M. Aucun attaquant polynomial ne doit être en mesure de dériver un deuxième chiffré C′= E (K, m) tel que M et M′soient reliés.

L'étude de ces propriétés nécessite qu'un attaquant possède la possibilité d'accéder à certains échantillons de données, messages chiffrés ou clairs par exemple. En fonction de ces échantillons, différents types d'attaques peuvent être appliquées. Nous allons citer ci-dessous les modèles d'attaques suivants [11] :

- Attaque à chiffrer seul « Ciphertext only attack » : dans ce modèle d'attaque, l'adversaire essaye d'obtenir le texte clair à partir d'un ensemble de textes chiffrés.
- Attaque à clair connu « Known plaintext attack » : l'attaquant dans ce modèle arrive à trouver des combinaisons de messages clairs et leurs correspondants chiffrés.
- Attaque à clair choisi « Chosen plaintext attack » : dans ce modèle d'attaque, l'adversaire peut obtenir le message chiffré correspondant pour tout choix de message clair.
- Attaque à clair adaptatif choisi « Adaptively chosen plaintext attack » : même principe que l'attaque à clair choisi sauf que l'attaquant ici peut choisir le dernier message clair à chiffrer en fonction des chiffrés reçus précédemment.

- Attaque à chiffré choisi « Chosen ciphertext attack » : il est identique à l'attaque à clair choisi. Dans ce modèle l'attaquant choisi les messages chiffrés pour les déchiffrer en vue d'obtenir les messages clairs.
- Attaque à chiffré adaptatif choisi « Adaptively chosen ciphertext attack » : Même principe que l'attaque à chiffré choisi sauf qu'ici l'attaquant peut choisir le message chiffré à déchiffrer en fonction des résultats reçus précédemment.

A l'égard des attaques mentionnées ci-dessus, d'autres versions d'attaques cryptographiques sont pratiquées, la plus naïve et utilisée est l'attaque par recherche exhaustive de la clé qui est appelée aussi attaque par force brute [44], [56]. Le principe de l'attaque consiste à générer et tester toutes les combinaisons possibles d'une clé jusqu'à trouver la clé réelle de chiffrement. C'est une méthode de redoutable efficacité notamment si elle utilise un matériel de haute performance. L'utilisateur a fortement de chance pour se protéger contre ce type d'attaque en utilisant des algorithmes de chiffrement solides avec une clé de taille assez grande.

La réussite de toute attaque sur un crypto-système est mesurée en fonction des ressources déployées. En effet, l'évaluation de la performance d'une attaque se fait par l'évaluation de la complexité des éléments suivants : le temps, la mémoire et les données [12].

- La complexité en temps : c'est la durée nécessaire pour que l'attaque réussisse.
- La complexité en mémoire : elle correspond à la mémoire allouée pour le stockage des données utilisées. Une attaque est considérée non pratique si elle demande un très grand espace mémoire.
- La complexité en données : elle est relative à la quantité de messages chiffrés nécessaires pour réaliser l'attaque. Dans plusieurs cas, une attaque exige un grand nombre de messages plutôt que de couter cher en temps.

5. Les modes opératoires du chiffrement par bloc

Dans un algorithme de chiffrement par bloc, nous ne chiffrons qu'un nombre exact d'octets qui constitue un bloc (AES chiffre des blocs de 16 octets). Si nous chiffrons un message plus petit que la taille du bloc, nous ajoutons à ce dernier une chaine appelée « Padding » pour atteindre la taille du bloc. Si le message est plus long à la taille du bloc utilisé, nous découpons le message en blocs égaux et nous réitérons le chiffrement sur chaque bloc entrant de données, la sortie est un texte chiffré de même longueur que celui en clair. Dans cette transition de texte clair vers un texte chiffré, les blocs ont été manipulés les uns avec les

autres selon un mode opératoire [42]. En effet, il existe plusieurs modes opératoires de chiffrement, nous allons choisir de décrire les trois modes opératoires ci-dessous : ECB, CBC et CTR [40], [71].

5.1. Le mode ECB (ELECTRONIC CODEBOOK)

C'est le mode le plus simple à mettre en œuvre, les blocs sont chiffrés les uns après les autres. Le texte chiffré est alors formé par la concaténation du premier bloc chiffré jusqu'à le dernier [12], [42].

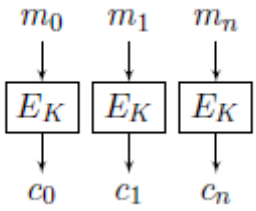

Figure 1.9: Mode ECB [40].

5.2. Le mode CBC (CIPHER BLOCK CHAINING)

Dans le mode CBC, nous appliquons une opération d'où-exclusif sur chaque bloc de texte clair avec le bloc chiffré précédent avant d'être chiffré [12], [42].

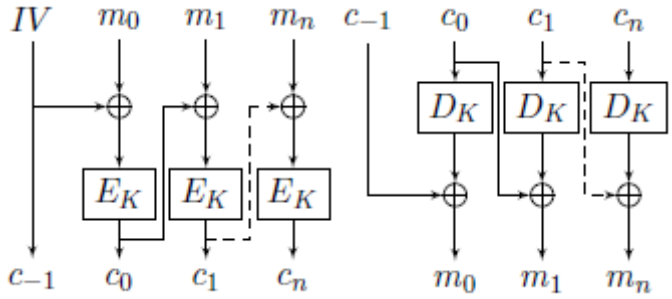

Figure 1.10: Mode CBC [40].

5.3. Le mode CTR (COUNTER)

Dans ce mode, le flux de clé est obtenu en chiffrant les valeurs successives d'un compteur [12], [42].

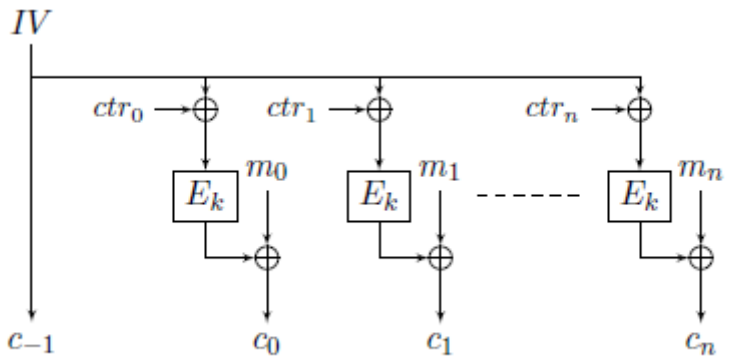

Figure 1.11: Mode CTR [40].

Pour utiliser les modes opératoires CBC et CTR, il est nécessaire de paramétrer l'algorithme de chiffrement par une valeur d'initialisation (Initial Value (IV)). Cette valeur doit être utilisée de façon unique, autrement dit qu'un message clair ne doit être chiffré qu'une seule fois avec la même valeur d'initialisation (IV). Cela assure que deux messages similaires auront des chiffrées différents.

6. Le chiffrement à flot

Les algorithmes de chiffrement à flot sont repartis généralement en deux catégories : synchrone et asynchrone. Le chiffrement asynchrone génère une suite chiffrante seulement à partir de la clé de chiffrement et un nombre de bits chiffrés précédents, ces systèmes sont presque tombés en désuétude. Dans le chiffrement synchrone nous générons la suite chiffrante uniquement à partir de la clé de chiffrement [11], [40].

A l'égard du chiffrement par bloc qui consiste à chiffrer de gros blocs de texte clair de taille fixe en utilisant une transformation. Un algorithme de chiffrement à flot synchrone permet de chiffrer un message clair en le combinant avec une suite binaire appelée suite chiffrante qui a la même taille que le message à chiffrer. La loi de combinaison peut être un XOR par exemple appliqué bit à bit sur le message clair et la suite chiffrante. La génération de la suite chiffrante se fait de façon indépendante à la fois du clair et du chiffré par un générateur de nombre pseudo aléatoire (Figure 1.12).

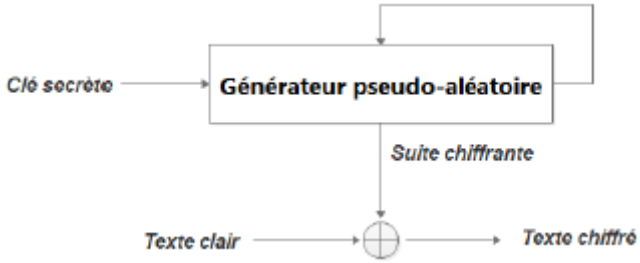

Figure 1.12: Principe général d'un chiffrement à flot [11].

Le chiffrement à flot possède de nombreux avantages, de point de vue performance il est rapide. D'ailleurs les données sont traitées en flux continu et le traitement de chaque bit (en chiffrement et déchiffrement) se fait indépendamment des autres bits du message, donc il n'y a pas besoin d'attendre tout le message complet ni d'avoir sa longueur pour commencer le chiffrement. Cela permet en même temps d'optimiser le délai et la taille de la mémoire tampon pour stocker le message avant l'obtention d'un bloc complet. Ces caractéristiques sont fortement demandées lorsque les ressources sont limitées et le critère de la rapidité du chiffrement et de déchiffrement des messages s'oppose. Généralement, les algorithmes de chiffrement à flot sont très pratiques dans les systèmes embarqués, parmi les plus utilisés, nous citons le A5/1 et RC4. L'algorithme A5/1 a été publié en 1994, il est utilisé dans les communications mobiles de type GSM pour chiffrer les communications entre le mobile et l'antenne-relais. Le RC4 a été conçu en 1987 par Ronald Rivest, il est utilisé dans le protocole WEP du Wifi [65].

La particularité que possède les algorithmes de chiffrement à flot rendent leurs utilisations bien adaptées au chiffrement des BD, notamment dans des contextes ou les champs des tables possèdent une petite taille. Cela va optimiser la taille de la BD après le chiffrement et fournir une meilleure performance des requêtes. Cependant, ce type d'algorithmes présente une faiblesse en sécurité par rapport aux algorithmes de chiffrement par blocs, ce qui limite leurs utilisations dans la protection des BD contenant des données à haute degré de sensibilité [40]. Certes, les SGBDR mis sur le marché implémentent des algorithmes de chiffrement à flot (RC4 sous Oracle par exemple) pour chiffrer les données, bien qu'il faut les utiliser avec précaution et dans des contextes bien définis. Les algorithmes de chiffrement à flot nécessitent encore d'être développés pour qu'ils s'adaptent au chiffrement des BD.

7. Les fonctions de hachage

Les algorithmes de cryptographie symétrique que nous avons étudiés précédemment servent à protéger la confidentialité des données. Les outils de base avec lesquels nous garantissons

l'intégrité des données sont les fonctions de hachage cryptographiques. Une fonction de hachage H est une fonction à sens unique qui prend une chaine de bits en entrée de taille quelconque et fournie une chaine en sortie unique de taille fixe. Nous appelons la chaine de sortie une « empreinte » ou un « haché » [15], [61], [87], [88]. Généralement H est défini, pour n fixé, de la manière suivante :

$$f : \{0,1\}^{*} \longrightarrow \{0,1\}^{n}$$

$$m \longrightarrow h(m)$$

L'intérêt principal d'une fonction de hachage est son utilisation pour garantir l'intégrité des données, chaque texte clair possède une seule empreinte, si le texte clair change, l'empreinte doit également changer. Une empreinte d'un message doit fortement dépendante des bits du message, cela veut dire qu'une fonction de hachage doit être un procédé complexe de telle façon que si nous changions un bit du message clair, l'empreinte ne doit pas avoir de liaison avec l'empreinte du message précédent. Une fonction de hachage est dite sécurisée si elle vérifie les quatre propriétés ci-dessous [15], [40] :

- Résistance à la collision : il est difficile de trouver deux messages m et m' tels que $m \neq m'$ et $H(m) = H(m')$.
- Résistance aux préimages : Etant donné un haché Ho, il est difficile d'inverser H pour trouver un message m vérifiant $H(m) = Ho$.
- Résistance aux secondes préimages : Etant donné un message mo et son haché Ho, il est difficile de trouver un message m différent de mo tel que $H(m) = H(m') = Ho$.

Parmi les premières fonctions de hachages utilisées en informatique nous citons MD4 et MD5, elles ont été conçues par Ron Rivest pour fournir des empreintes de 128 bits [65], [66]. Actuellement, elles sont presque obsolètes car elles présentent des vulnérabilités aux attaques par collisions et par préimages. Les problèmes de MD4 et MD5 ont été corrigés par le NIST en proposant la fonction SHA0, cette dernière a également été améliorée pour donner la fonction SHA1, la taille des empreintes de SHA0 et SHA1 sont sur 160 bits [30]. Des corrections apportées aux faiblesses de SHA1 ont donné naissance à SHA2. En effet, SHA2 est un ensemble de fonctions avec des tailles de hachées élevées, elle est composée des 4 fonctions SHA-224, SHA-256, SHA-384 et SHA-512 possédant respectivement des empreintes sur 224, 256, 384 et 512 bits [30]. Récemment, la fonction SHA3 est recommandée, l'objectif étant de ne pas remplacer SHA2 qui n'a pas été réellement

compromise par une attaque significative, mais pour donner une suite d'amélioration aux faiblesses des MD4, MD5, SHA0 et SHA1.

Les fonctions de hachage assurent énormément de besoins en informatique de telle façon que nous pouvons les considérer comme le couteau suisse de la cryptographie. Parmi leurs utilisations les plus courantes nous citons la protection des mots de passe, la signature électronique et les protocoles de sécurité informatique comme dans le cas du MAC. Dans le contexte des BD les fonctions de hachages servent spécialement à contrôler les attaques de modifications de données, par exemple les attaques de modification qui interceptent les données en transit [78]. Avant de stocker les données dans la BD, le SGBDR calcule les empreintes de données à l'aide d'une fonction de hachage. Le SGBDR effectue la même opération en comparant les deux empreintes au moment du retour des données, si elles sont identiques, cela explique que les données n'ont pas été modifiées.

CHAPITRE II : PROTECTION DES BASES DE DONNÉES PAR LE CHIFFREMENT : SOLUTIONS ET MODÈLES DE CHIFFREMENT

Introduction :

Suite à l'augmentation des cas signalés de violation de données et des accès non autorisés, le chiffrement des BD est devenu l'une des mesures de sécurité les plus adoptées par les entreprises. Il permet d'offrir un niveau élevé de protection des données en garantissant que seuls les utilisateurs autorisés peuvent accéder aux données, ainsi que de protéger les sauvegardes de la BD en cas de perte, vol ou autre compromission de la sauvegarde [63], [77]. Grâce au chiffrement, les BD sont devenues facilement accessibles à distance à partir de n'importe quelle machine comme dans le Cloud par exemple, les entreprises ont pu se décharger de la gestion de leurs serveurs de BD et de profiter du savoir-faire et de la compétence des prestataires externes sans se préoccuper de la sécurité de leurs données [14], [33].

Actuellement, la majorité des langages de programmations, SGBDR et outils de chiffrement de disques durs proposent des solutions de chiffrement de BD. Le chiffrement peut avoir lieu selon trois niveaux : SGBDR, disque dur et application [79]. Cependant, ces niveaux engendrent certaines limites, notamment dans la protection des clés de chiffrement et la baisse de la performance du système due au temps de chiffrement /déchiffrement [28]. Toute solution de chiffrement de BD est basée sur un concept ou modèle de chiffrement, il définit la manière dont les données sont chiffrées, le niveau où elles sont chiffrées, la manière dont les clés sont gérées et protégées. En effet, un modèle de chiffrement n'est pas un concept délimité, il doit plutôt être extensible, censé être amélioré en continu pour implémenter des nouveaux concepts consolidant davantage la sécurité des données. Ce n'est pas un choix, c'est une nécessité imposée par les nouveaux défis de sécurité, ceux de l'évolution rapide des types d'attaques et des moyens déployés par les attaquants, de l'évolution de la technologie informatique de façon générale.

Dans ce chapitre, nous allons présenter les différents modèles d'attaques sur les BD. Nous allons aborder les niveaux où le chiffrement doit avoir lieu, chaque niveau va nous amener à discuter les points positifs et négatifs qu'il engendre. Dans ce qui suit, nous allons exposer les résultats d'une étude que nous avons réalisés sur les outils de chiffrement intégrés dans quelques SGBDR du marché. La dernière partie de ce chapitre va être consacrée à la présentation de l'état d'art des modèles de chiffrement des BD que nous avons étudiés dans la littérature et les exigences auxquelles ils doivent répondre.

I. Les modèles d'attaques

Les moyens et les stratégies des attaquants sont multiples. Ils varient selon le niveau où se situe l'attaquant, c'est à dire aux trois classes citées dans le chapitre précédent. Dans plusieurs cas, les stratégies des attaquants peuvent excéder les prévisions de ceux qui gèrent la sécurité de la BD. En effet, les BD peuvent être victime d'attaques passives et actives [79].

1. Les attaques passives

Les attaques passives sont des attaques de fuite d'informations, un utilisateur non autorisé ne doit révéler aucune information sur les données claires d'une BD bien sécurisée. Les attaques passives sont classées en trois catégories [78], [79] :

- Les fuites statiques : un attaquant peut révéler des informations sur les données claires en analysant les données de la BD à un moment donné. Par exemple, dans une BD dont les données égales sont chiffrées de la même façon, un attaquant peut attaquer la BD en s'appuyant sur une analyse des données identiquement chiffrées (fréquences d'apparition, emplacement, …) pour collecter des statistiques sur les données claires.
- Les fuites de liaisons : un attaquant peut obtenir des informations sur les valeurs de texte clair de la BD en liant une valeur de la BD à sa position dans l'index. Par exemple, si la valeur de la BD et la valeur de l'index sont chiffrées de la même façon (les chiffrées sont égales), un observateur peut rechercher la valeur du texte chiffré de la BD dans l'index, déterminer sa position et estimer sa valeur en texte clair.
- Les fuites dynamiques : un attaquant peut obtenir des informations sur les valeurs de texte clair de la BD en observant et en analysant les modèles d'accès et les modifications apportées à la BD sur une période donnée. Par exemple, si un utilisateur surveille l'index pendant une période donnée et si, pendant cette période, une seule valeur est insérée (aucune valeur n'est mise à jour ou supprimée), l'observateur peut estimer sa valeur en texte brut en fonction de sa nouvelle position dans la l'index.

2. Les attaques actives

Outre que les attaques de fuite d'informations, les attaques actives sont des attaques de modification des données dans la BD, elles sont classées en trois catégories [21], [78] :

- Corruption des données chiffrées : l'attaquant remplace une donnée chiffrée de la BD par une autre générée. Cette attaque est assez compliquée à réaliser car elle demande la détention des clés de chiffrement avant de l'appliquer.
- Substitution des données chiffrées : l'attaquant remplace une donnée chiffrée de la BD par une autre donnée chiffrée située dans un emplacement différent de la BD.
- Rejeu des données : l'attaquant remplace une donnée chiffrée de la BD par une ancienne donnée précédemment supprimée ou mise à jour [85].

II. Les niveaux de chiffrement dans les Bases de Données

Dans une BD, la protection de la confidentialité des données se fait par le chiffrement. Le choix de l'endroit où le chiffrement/déchiffrement doit se faire est un point fondamental. Nous pouvons chiffrer les BD selon trois niveaux : application, SGBDR et fichiers ou disques durs [45], [52], [79].

1. Le chiffrement au niveau d'application

Le chiffrement au niveau applicatif signifie que le processus de chiffrement/déchiffrement s'effectue au niveau des applications qui génèrent les données. Les applications des clients chiffrent et envoient les données pour le stockage dans la BD, de la même manière elles récupèrent les données chiffrées de la BD afin d'être déchiffrées. Cette stratégie possède l'avantage de résister aux attaques internes, c'est à dire aux attaques menées par les administrateurs de la BD, puisque toutes les clés sont détenues par les applications, il n'y a pas de transmission de données claires ni de clés au serveur. Bien que cette solution présente de grands avantages dans beaucoup de cas d'implémentation, elle engendre des problèmes à savoir, il faut déporter une grande partie du traitement au client puisque l'application prend à sa charge une grande partie de l'évaluation de la requête. La gestion des droits d'accès est confiée au client, ainsi que les données et les clés seront forcément stockées en claires sur ce dernier. Cela pose donc un problème de confiance vis-à-vis du client puisqu'il n'est pas forcément un site de confiance. Il peut attaquer les droits d'accès ou les clés de chiffrement.

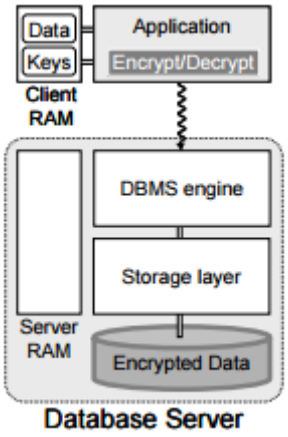

Figure 2.1: Chiffrement au niveau applicatif [45].

2. Le chiffrement au niveau de la Base de Données

Le chiffrement au niveau de la BD signifie que les données sont chiffrées et déchiffrées au sein du SGBDR avec des clés détenues par le serveur de BD. Le chiffrement dans ce cas est transparent pour les applications. Les données sont reçues en claires en entrée par le SGBDR et elles ressortent identiquement. Ce type de chiffrement offre des avantages particuliers. Il n'impacte pas les applications autour desquelles sont conçues les BD. D'autant plus, il est sélectif, comme il peut être relié à la sensibilité des données ou au niveau des droits d'accès des utilisateurs. La granularité de chiffrement et aussi un point fort de ce type de chiffrement, elle peut couvrir les colonnes, les Tablespaces et les cellules.

Cependant, le chiffrement au niveau de la BD possède des limites. Il peut produire des dégradations de la performance du SGBDR, car globalement l'utilisation des index est à éviter, voire impossible sur les données chiffrées. Au moment de l'exécution des requêtes, le chiffrement/déchiffrement des données nécessite des clés stockées au niveau du serveur de la BD, ou transmises à partir d'une entité externe à savoir, l'utilisation de Wallet, d'un Hardware Security Module (HSM) ou d'un serveur de sécurité. Ces solutions offrent une meilleure protection des clés contre les attaques externes. Par contre la probabilité d'attaque par l'administrateur n'est pas négligeable. Ce dernier possède tous les droits sur la BD, il peut l'attaquer directement ou par conspiration via une collaboration avec une personne externe (divulgation du compte administrateur par exemple), tout en effaçant ses traces.

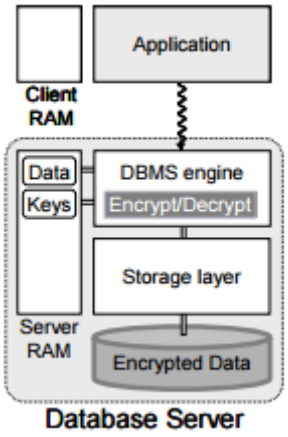

Figure 2.2: Chiffrement au niveau de la base de données [45].

3. Le chiffrement au niveau du stockage : fichiers, partitions et disques durs

Le chiffrement au niveau du stockage, autrement dit les fichiers, partitions ou disques durs, consiste à chiffrer les données dans le système de stockage en les protégeant statiquement. Cela est indispensable quand on a la vocation de protéger les données de la BD contre des attaques moyennant un accès physique sur le disque. De point de vue BD, le chiffrement au niveau du stockage offre l'avantage d'être transparent pour les applications, il évite tout changement ou modification dans les applications. Le chiffrement à ce niveau ne prend pas en considération les objets de la BD, comme il ne peut pas être lié aux privilèges des utilisateurs. Il est sélectif, c'est à dire qu'on ne peut chiffrer que quelques parties de la BD sensibles.

Il y a principalement deux façons pour chiffrer la BD au niveau du stockage, la première se fait en dessous du système de gestion des fichiers. La gestion du chiffrement ici est assurée lors de l'écriture des données sur le disque, c'est le cas du « Full Disk Encryption » [40]. L'autre solution est d'utiliser un système de gestion de fichiers chiffrés, cela offre une souplesse à la gestion des accès aux données et des clés de chiffrement.

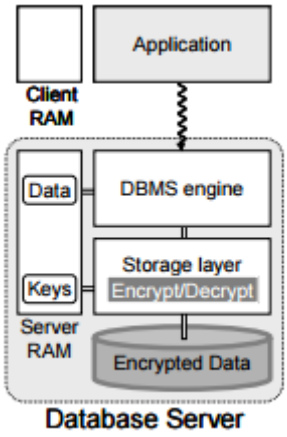

Figure 2.3: Chiffrement au niveau du stockage [45].

4. La surcharge du chiffrement

Tout renforcement de la sécurité d'une BD par le chiffrement entraine une surcharge supplémentaire qui influence la durée d'exécution des requêtes des utilisateurs. L'impact de la surcharge est fonction de plusieurs facteurs : la quantité de données à chiffrer/déchiffrer, le choix de l'algorithme et la méthode chiffrement (un seul algorithme ou imbrication de plusieurs algorithmes) et le niveau souhaité de chiffrement [48], [79].

L'impact du chiffrement sur la performance des requêtes est considéré actuellement un sujet d'actualité en recherche, plusieurs travaux ont été proposés dans ce sens notamment les contributions des auteurs de [4], [23]. La performance d'une BD chiffrée devient plus critique si le souci du transfert des données est à prendre en compte. La protection du transfert par un SSL par exemple alourdit énormément les applications qui traitent des grands volumes de données, notamment dans le cas d'un client et serveur qui sont distants l'un par rapport à l'autre. De ce fait, le choix du niveau de chiffrement reste un point crucial, il doit assurer le chiffrement uniquement des données sensibles et laisser les données insensibles en claires lors de l'exécution des requêtes.

5. La gestion des clés de cryptage

La gestion des clés de cryptage fait référence à la manière dont les clés sont gérées de leur création jusqu'à leur destruction. Chaque opération de chiffrement/déchiffrement repose sur une clé de cryptage. La connaissance de cette clé est l'objectif de tout attaquant. Tout système cryptographique est limité par la protection de ses clés, leurs emplacements et les restrictions d'y accéder sont donc un problème fondamental dans le chiffrement des BD. Bouganim et al., ont présenté dans [45] les approches que nous pouvons utiliser pour protéger les clés de chiffrement. Le principe général consiste à stocker les clés dans un endroit restreint et les

chiffrer par une clé principale appelée « Master Key », qui est elle-même stockée quelque part sur le serveur de BD ou à l'extérieur de ce dernier. Nous allons exposer en détail dans le chapitre 3 toutes ces solutions. Nous allons proposer dans le même chapitre une nouvelle approche de protection de ces clés en fonction de la granularité du chiffrement choisie.

6. Les couches de chiffrement des données dans un Système de Gestion de Base de Données

Au sein d'un SGBDR, l'opération de chiffrement/déchiffrement des données s'effectue dans deux couches : la couche « SQL Interface » et la couche « Storage Engine » comme illustré dans la Figure 2.4.

6.1. La couche « SQL Interface »

Cette couche chiffre les données en utilisant les fonctions, procédures et déclencheurs internes du SGBDR. Le chiffrement à ce niveau n'apporte pas généralement des changements par rapport aux applications, bien qu'il engendre quelques limites à savoir, certains mécanismes peuvent ne pas fonctionner comme les index puisque le chiffrement s'effectue au-dessus du moteur d'exécution des requêtes. L'utilisation d'une procédure stockée ou d'une fonction de chiffrement signifie que le chiffrement se fait au niveau du langage des procédures stockées et non pas à la décomposition de l'arbre algébrique par le moteur de l'exécution des requêtes. Cela implique automatiquement un impact sur la performance. Ces mécanismes ne sont pas à l'abri des menaces des attaquants, un administrateur malveillant peut facilement les désactiver [29], [79].

6.2. La couche « Storage Engine »

Le chiffrement au niveau de cette couche est identique au chiffrement au niveau des fichiers et disques durs que nous avons présentés précédemment. L'opération de chiffrement/déchiffrement est effectuée par le SGBDR. Les pages disques sont chiffrées et déchiffrées au moment de la lecture et de l'écriture à partir du disque. Lorsque la page est chargée du disque, ses valeurs sont déchiffrées et lorsqu'elle est stockée sur le disque ses valeurs sont chiffrées. La limite du chiffrement à ce niveau parait lorsqu'une page est chargée dans la mémoire pour une lecture. Toutes les valeurs de cette page sont déchiffrées même si la requête de l'utilisateur ne possède pas le droit d'accéder à certaines valeurs de données de la page. Cela implique aussi une dégradation de la performance [28], [79].

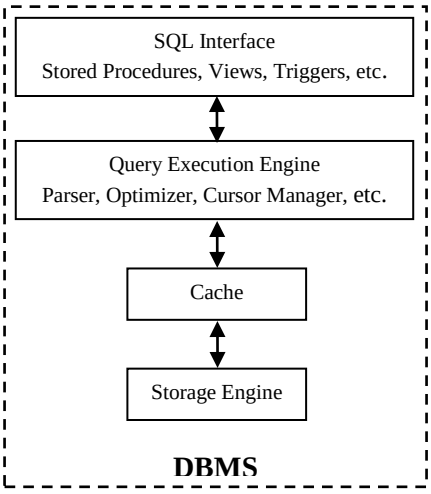

Figure 2.4: Architecture interne d'un SGBDR [79].

III. Les solutions de chiffrement des Bases de Données existantes au marché

Dans cette section, nous allons étudier et analyser les solutions de chiffrement implémentées dans quelques SGBDR du marché. Cela nous permettra d'avoir une idée générale sur ces solutions en termes de fonctionnalités de chiffrement qu'elles intègrent. Nous allons prendre les SGBDR les plus utilisés dans la gestion des systèmes d'informations à savoir, ORACLE, Microsoft SQL Server, Microsoft Access et MySQL.

1. Le chiffrement avec ORACLE

ORACLE est un SGBDR qui offre plusieurs solutions de sécurité telles que l'authentification, le contrôle d'accès, l'audit et le chiffrement des données. Oracle propose de chiffrer les données via deux outils principaux : le chiffrement avec Oracle TDE et le chiffrement avec le package DBMS_CRYPTO [22], [62].

1.1. Le chiffrement avec ORACLE TDE

Oracle Transparent Data Encryption ou Oracle TDE est un outil de sécurité d'Oracle qui permet la protection des BD contre les accès non autorisés par le chiffrement au repos des données. C'est un outil qui chiffre/déchiffre les données de manière transparente vis-à-vis des utilisateurs et des applications. Le chiffrement s'effectue au niveau du SGBDR dans la couche « Storage Engine ». Oracle TDE propose deux solutions pour chiffrer les données : TDE Column Encryption et TDE Tablespace Encryption [62].

1.1.1. TDE Column Encryption

La solution TDE Column Encryption est utilisée pour protéger les données sensibles stockées dans des colonnes précises d'une BD. Au lieu de chiffrer toutes les colonnes de la table, nous ne chiffrons que les colonnes sensibles. C'est une manière d'optimiser la performance du système en réduisant le temps de chiffrement/déchiffrement afin d'améliorer la rapidité d'exécution des requêtes [62].

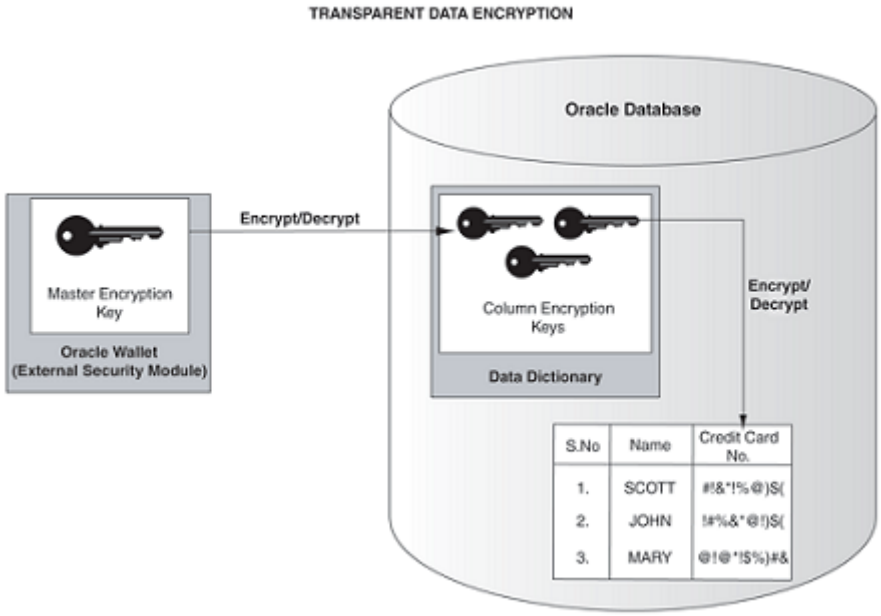

Figure 2.5: Modèle de chiffrement des colonnes sous Oracle TDE Column Encryption [62].

1.1.2. TDE Tablespace Encryption

La solution TDE Tablespace Encryption permet de chiffrer une Tablespace entière. Tous les objets créés dans la Tablespace sont automatiquement chiffrés. Ce mécanisme est indispensable si nous souhaitons protéger les données sensibles au niveau des tables. Autrement dit, qu'il n'est pas nécessaire de faire une analyse granulaire pour chaque colonne de table en vue de préciser les colonnes qui nécessitent un chiffrement. C'est une bonne alternative au chiffrement de la colonne dans le cas où les tables contiennent plusieurs colonnes sensibles.

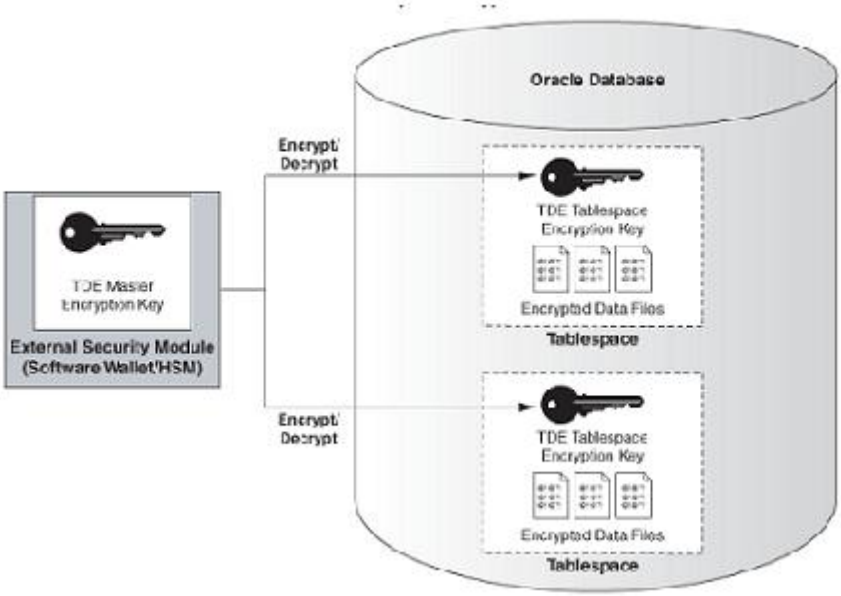

Figure 2.6: Modèle de chiffrement des Tablespaces sous Oracle TDE Tablespace Encryption [62].

1.2. Le chiffrement avec le Package DBMS_CRYPTO

Oracle propose un package PL/SQL nommé « DBMS_CRYPTO » pour chiffrer les BD. Ce package supporte une multitude d'algorithmes de chiffrement et de fonctions de hachage dont les principaux sont les suivants [29] :

- Algorithme de chiffrement : DES, Triple DES (3DES, 2-Key et 3-Key) et AES (128, 192, 256).
- Les fonctions de hachages : MD5, MD4, SHA1 et SHA2.

L'utilisation du package DBMS_CRYPTO doit être pilotée par les applications. Elles appellent le package à distance ou localement sur le serveur de BD. Le chiffrement s'effectue au niveau du SGBDR dans la couche « SQL Interface ».

1.3. La protection des clés de chiffrement

Oracle TDE chiffre/déchiffre les données en utilisant deux clés, une clé principale (Km) et une clé de chiffrement de la table ou de la Tablespace. La clé de chiffrement de la table sert à chiffrer/déchiffrer les colonnes de cette table. La clé de chiffrement de la Tablespace sert à chiffrer/déchiffrer toutes les colonnes des tables stockées dans la Tablespace. Chaque table et Tablespace possède sa propre clé de chiffrement/déchiffrement. Le rôle de la clé principale (Km) consiste à protéger les clés de chiffrement des tables et Tablespaces en chiffrant ces

derniers. Cette clé est stockée, chiffrée elle-même par un mot de passe, dans un portefeuille Oracle qui se situe soit à l'intérieur ou à l'extérieur du serveur de BD, ou bien dans une autre solution de protection des clés comme le Hardware Security Module (HSM). L'endroit de stockage et le mot de passe de protection de la clé principale ne devraient être connus que par l'administrateur de la BD ou de sécurité s'il existe bien évidemment. Par ailleurs, si nous utilisons le Package DBMS_CRYPTO, souvent les clés de chiffrement restent en claires sur les applications, ou logées dans un fichier protégé par un mot de passe. Normalement l'accès à ce fichier doit être restreint et l'accessibilité est unique à l'administrateur d'applications.

2. Le chiffrement avec Microsoft SQL Server

Microsoft SQL Server vient en deuxième position après Oracle en termes de la puissance des solutions de chiffrement des BD. Il propose de chiffrer les données via les outils suivants : SQL Server TDE, Always Encrypted et Column/Cell-Level Encryption.

2.1. SQL Server TDE

SQL Server TDE est une solution de chiffrement de BD qui est disponible à partir de la version 2008 et ultérieure de Microsoft 8SQL Server [83]. En fait, cette solution est similaire à Oracle TDE, elle permet de protéger la BD par le chiffrement des données au repos, de manière transparente vis-à-vis des utilisateurs et des applications. Toutes les données écrites dans la BD sont automatiquement chiffrées. Le déchiffrement s'effectue au fur et à mesure que les données sont lues par un utilisateur ou une application, par contre les données sont en claires au moment du transfert. Le chiffrement s'effectue au niveau du SGBDR dans la couche « Storage Engine » [55].

2.2. Always Encrypted

C'est une solution nouvellement mise en service en 2016 dans Microsoft SQL Server. Elle permet de chiffrer les données dans la BD et en mouvement, c'est à dire entre l'application et le serveur de la BD. Le déchiffrement ne s'effectue qu'au niveau des applications. Cette technique favorise une protection efficace contre les attaques des administrateurs malhonnêtes, des voleurs de sauvegarde et des attaques de types « Man in the Middle ». D'autant plus, « Always Encrypted » offre la possibilité de chiffrer que les colonnes sensibles, contrairement à SQL Server TDE qui chiffre la BD toute entière [55].

2.3. Column/Cell-Level Encryption

Cette solution est équivalente au chiffrement avec le Package DBMS_CRYPTO d'Oracle, elle est disponible à partir de la version 2005 de Microsoft SQL Server. Le chiffrement ici est appliqué sur des colonnes spécifiques, nous ne chiffrons ou déchiffrons que les colonnes sensibles d'une table, ce qui permet une meilleure performance dans l'exécution des requêtes. Le déchiffrement des données déploie uniquement une fonction (decryptbykey) dans la requête.

Par rapport à SQL Server TDE, le principal inconvénient de cette solution est la nécessité de modifier les applications pour la mettre en œuvre. D'autant plus, la performance peut également être impactée si des requêtes de recherche ne sont pas optimisées en évitant des données chiffrées. Plusieurs algorithmes de chiffrement sont supportés tels que AES (128, 192,256), 3DES et le DES [55].

2.4. La protection des clés de chiffrement

SQL Server TDE chiffre les données avec une infrastructure hiérarchique de gestion de clés très compliquée. L'utilisateur crée une clé principale et un certificat protégé par cette clé principale. En plus, il crée la clé de cryptage de la BD et il la protège par le certificat [53], [54], [58].

Lorsqu'on chiffre une BD avec « Column/Cell-Level Encryption », les clés restent en claires sur les applications ou sur le serveur d'applications. Elles sont stockées dans des fichiers avec un accès protégé par un mot de passe.

3. Le chiffrement avec Microsoft Access

Microsoft Access est un SGBDR qui offre moins de fonctionnalités de chiffrement des données. Le principe du chiffrement se base sur l'utilisation d'un mot de passe pour chiffrer toute la BD.

4. Le chiffrement avec MySQL

MySQL est un SGBDR qui fait partie des systèmes les plus utilisés au monde par les applications Web. MySQL possède deux solutions de chiffrement de données : MySQL Entreprise Transparent Data Encryption (MySQL Entreprise TDE) et un ensemble de procédures et fonctions pour le chiffrement des données.

4.1. MySQL Entreprise TDE

Oracle ne cesse de développer les fonctions de sécurité de MySQL en implémentant la solution MySQL Entreprise TDE. Cet outil est disponible à partir de la version 5.7.12 de MySQL, il est implémenté dans les tables du moteur de stockage InnoDB qui stocke les données dans les Tablespaces « innodb_file_per_table ». Similairement à Oracle TDE et SQL Server TDE, MySQL Entreprise TDE fournit un chiffrement transparent au repos des fichiers de données vis-à-vis des utilisateurs et des applications. Les données sont chiffrées en écriture et déchiffrées en lecture dans la couche « Storage Engine » du SGBDR. MySQL Entreprise TDE supporte uniquement l'algorithme AES en mode ECB et CBC pour chiffrer respectivement les données et les clés de chiffrement des Tablespaces [59], [60].

4.2. MySQL : Encryption functions

A l'égard des fonctions de chiffrement implémentées dans Oracle et SQL Server, MySQL supporte un ensemble d'algorithmes de chiffrement et fonctions de hachage à savoir, AES, DES, 3DES, SHA, MD5…etc. [59], [61].

4.3. La protection des clés de chiffrement

MySQL Enterprise TDE utilise deux clés pour chiffrer/déchiffrer les données, une clé de chiffrement pour chaque Tablespace « InnoDB » et une clé principale pour la protection de ces clés. Le principe de protection des clés est similaire à Oracle TDE [59], [61].

5. Analyse et discussions

D'après ce que nous avons évoqué ci-dessus, il parait évident que les solutions de chiffrement supportées par les SGBDR ne sont pas identiques en termes de fonctionnalités intégrées. Chaque solution possède sa propre technique de chiffrer les données et de protéger les clés. Il est certain que chaque outil à des points forts et des faiblesses que nous allons démontrer dans ce qui suit. L'exploitation de ces faiblesses par des attaquants expérimentés peut engendrer des conséquences catastrophiques à la sécurité des données. L'expérience que nous avons acquis durant toute cette période de thèse en analysant les travaux et les documents relatifs au chiffrement des BD, nous a permis de développer les contraintes ci-dessous dont nous recommandons vivement de prendre en compte avant toute adoption d'une solution de chiffrement de BD.

5.1. Les contraintes liées à la sécurité des données

Les concepteurs des BD doivent penser à l'avance au niveau de sécurité sur lequel leurs BD seront abritées. Les solutions de chiffrement les moins sécurisées sont à éviter, notamment si la BD contient des données de très haut degré de sensibilité. Microsoft Accès par exemple est une solution à éviter catégoriquement. Elle est considérée la plus faible de toutes les solutions qui existent sur le marché. Dans la dernière partie de ce chapitre nous allons voir que parmi les exigences d'un modèle de chiffrement efficace de BD, sa possibilité de chiffrer la BD avec plusieurs clés et algorithmes. Cela permet de réduire la probabilité d'un déchiffrement complet de la BD si la clé tombe entre les mains d'un attaquant.

Un autre point très important s'impose lorsque nous adoptons des solutions comme ORACLE TDE, SQL SERVER TDE et MySQL Entreprise TDE. Il faut nécessairement chiffrer le transfert des données entre l'application et le serveur avec des protocoles tel que le SSL. Cela élimine toute probabilité d'attaque de type « Man in the middle ». Dans la pratique, la sécurisation du transfert impacte énormément la performance notamment si les deux sites sont distants et la quantité de données échangées est importante. L'outil « Always Encrypted » de Microsoft SQL Server est une excellente alternative qui peut résoudre ce problème. Il chiffre conjointement les données au moment du transfert et sur la BD. Les données sont donc chiffrées à la fois sur la BD et dans les deux sens entre l'application et la BD. Cependant, cet outil n'est pas pris en charge actuellement par la plupart des bibliothèques clientes. ADO.NET 4.6 est le seul fournisseur qui fonctionne avec cet outil, il faut s'assurer qu'il soit installé sur chaque machine qui exécute une application cliente et qui communique avec des données Always Encrypted.

Relativement aux outils de chiffrement Column/Cell-Level, DBMS_CRYPTO et les fonctions de chiffrement de MySQL, les principaux inconvénients résident dans la quantité de modification du code qu'il faut mettre en place pour réadapter les applications au chiffrement. Un changement d'un algorithme ou une clé de chiffrement d'une colonne par exemple doit se faire sur toutes les machines clientes (dans le cas où on n'utilise pas un serveur d'application) ainsi que sur le code de tous les modules de l'application. En outre, le déploiement de ces outils pour chiffrer la BD peut engendrer un impact sur la performance par les requêtes de recherche non optimisées et qui incluent une multitude de colonnes chiffrées.

Parfois, la mise en œuvre d'une solution de chiffrement de BD est fortement corrélée à deux facteurs principaux : la complexité de la gestion de la solution et la compétence de l'équipe informatique qui s'en occupe. Ces éléments sont extrêmement importants et à ne pas sous-estimer. Ils peuvent conduire à des risques élevés qui menacent la sécurité de la BD. ORACLE et SQL SERVER par exemple nécessitent une administration et une configuration préalable très pointue pour activer le processus de chiffrement dans la BD (choix des colonnes sensibles, choix des algorithmes, choix de la solution de protection des clés, etc.). Cette configuration doit être faite de manière minutieuse. Il faut qu'elle soit revue et maintenue dans le temps. A titre d'exemple, chiffrer une BD avec l'outil Oracle TDE Column Encryption nécessite un processus préalable d'identification des colonnes sensibles dans la BD toute entière. Le fait de laisser des données sensibles sans chiffrement pour plusieurs raisons, oublie, l'incompétence des responsables de sécurité, ou la sous-estimation de la sensibilité de certaines données peut impacter gravement la sécurité de la BD. Un attaquant peut exploiter ces erreurs pour déterminer non seulement les valeurs de ces éléments non chiffrés mais d'autres valeurs de données qui y sont corrélées.

L'implémentation d'une politique de chiffrement nécessite la mise en place d'une politique de contrôle d'accès pointue au profit des utilisateurs. Le chiffrement de données sans un mécanisme de contrôle des privilèges d'accès des utilisateurs laisse la BD confrontée à des attaques internes probables, soit par élévation de privilèges ou par abus de privilèges excessifs ou légitimes. Par exemple, un utilisateur d'une BD dont la fonction est de superviser seulement les accès des travailleurs qui accèdent à des zones contrôlées dans une centrale nucléaire peut exploiter le privilège de sélection excessif sur la table des doses reçues par les travailleurs pour voir, voler ou divulguer ces valeurs de doses qui sont considérées des secrets médicaux. L'utilisateur s'est retrouvé avec des privilèges au-delà de ses fonctions pour la simple raison que les administrateurs de la BD n'ont pas défini des mécanismes de contrôle d'accès sur cette table, soit par oubli ou négligence de cette opération. L'utilisateur a pu également voler la clé de chiffrement stockée dans un endroit non sécurisé sur le serveur d'application et ainsi déchiffrer toute la table.

5.2. Les contraintes liées à la sécurité des clés de chiffrement

La réussite de toute politique de sécurisation d'une BD via le chiffrement repose sur une protection efficace des clés de chiffrement. Toutes les solutions de chiffrement intégrées dans les SGBDR prennent en considération cet aspect. Chaque solution propose sa propre méthode

pour protéger les clés. Cependant, chaque méthode à des limites qu'il faut sérieusement prendre en compte avant toute décision de chiffrement de données.

Lorsque nous utilisons des outils comme Column/Cell-Level et DBMS_CRYPTO pour chiffrer une BD, souvent l'administrateur de l'application laisse les clés en claires sur les applications, ou dans les meilleurs cas protégés dans des fichiers à accès restreins avec mot de passe. L'administrateur n'est pas toujours une source de confiance. Il peut attaquer les clés lui-même ou par conspiration avec une personne externe.

Quant aux outils ORACLE TDE, MYSQL Entreprise TDE, SQL SERVER TDE, les clés sont protégées par une clé principale qui est stockée généralement dans un endroit séparé sous forme chiffrée avec un mot de passe. La création et la sécurisation de la clé principale font partie des fonctions de l'administrateur ou de sécurité de la BD, or ces derniers peuvent toujours constituer une source de menace directe ou indirecte à la sécurité de cette clé.

5.3. Les contraintes liées à la performance du chiffrement

Généralement, le chiffrement des BD introduit une surcharge supplémentaire aux requêtes des utilisateurs, cette surcharge est fonction de la taille de la BD à chiffrée, de l'algorithme et la méthode du chiffrement utilisé, ainsi que du niveau de chiffrement choisi. Lorsque nous choisissons de chiffrer une BD qui possède une taille importante avec l'outil Column/Cell-Level ou DBMS_CRYPTO, la performance décroit par rapport à son chiffrement avec TDE Column Encryption d'Oracle. De même, lorsque nous chiffrons une BD possédant moins de colonnes sensibles avec Oracle TDE Tablespace Encryption, la performance décroit par rapport à son chiffrement avec Oracle TDE Column Encryption. Dans le cas inverse, c'est-à-dire que la BD possède un nombre de colonnes importants reparti sur plusieurs tables, le chiffrement avec TDE Tablespace Encryption est bien évidement plus bénéfique.

IV. Les modèles de chiffrement des Bases de Données

Avant d'entamer le sujet principal de cette thèse qui porte sur le développement des modèles de chiffrement des BD relationnelles, il nous parait nécessaire de cerner une définition d'un modèle de chiffrement.

Un modèle de chiffrement de BD détermine l'ensemble des méthodes et moyens utilisés pour assurer la sécurité d'une BD en utilisant le chiffrement. Un modèle de chiffrement de BD doit définir à quelle granularité le chiffrement sécurise les données, la manière avec laquelle le modèle génère et protège les clés, le type d'algorithme utilisé, la couche ou le chiffrement doit

être implémenté. La notion d'un modèle de chiffrement ne s'arrête pas ici, il faut qu'il soit extensible et évolutif selon les besoins de sécurité et les contraintes imposées par l'évolution rapide des types d'attaques et des moyens des attaquants. Certes, les modèles de chiffrement prendront une place déterminante dans l'avenir suite à l'émergence et l'évolution accrue du domaine de l'intelligence artificielle.

1. Les caractéristiques d'un modèle pertinent de chiffrement des Bases de Données

Le grand challenge d'un modèle de chiffrement de BD est sa possibilité de prendre en considération les éléments cités ci-dessous :

La granularité de chiffrement : le niveau de granularité d'un chiffrement est considéré un problème fondamental dans le chiffrement des BD [79]. Les niveaux les plus pertinents (ou fins) où le chiffrement doit être implémenté sont les suivants : cellule, enregistrement, table et page. Un niveau de granularité de chiffrement pertinent doit apporter à la sécurité de la BD les avantages suivants :

- Il ne faut chiffrer que les données sensibles et laisser les données insensibles en claires. Cela implique que le SGBDR chiffre et déchiffre que les données sensibles lors de l'exécution d'une requête d'un l'utilisateur.
- Une granularité de chiffrement doit assurer une sécurité extrême des données de la BD, cela doit se refléter par une impossibilité de casser le chiffrement.
- Chiffrer les données sensibles d'une BD avec une seule clé n'apporte plus une sécurité élevée, au contraire il peut être catastrophique, même si nous lui associons un mécanisme de contrôle d'accès. Dans le cas où l'attaquant arrive à obtenir la clé, il peut déchiffrer tout. Un niveau de granularité de chiffrement pertinent doit fournir la possibilité de chiffrer les données de la BD avec plusieurs valeurs de clés.
- Une granularité de chiffrement fine si elle n'est pas correctement implémentée engendrera des problèmes graves à la sécurité de la BD en termes de fuite de données et de modification non autorisée.
- Une granularité de chiffrement doit assurer une indépendance entre le déchiffrement d'un enregistrement de la BD avec d'autres enregistrements.

La gestion des clés de cryptage : La gestion des clés de cryptage est un point fondamental dans tout modèle de chiffrement de BD. Elle définit la méthode dont les clés sont générées,

stockées et protégées au cours de leurs utilisations jusqu'à leurs destructions [34], [45], [52]. Une gestion efficace des clés de cryptage dans un modèle de chiffrement doit prendre en considération les points suivants :

- Elle doit définir une méthode sécurisée de génération de clé de chiffrement.
- Elle doit donner la possibilité de fournir plusieurs valeurs de clés pour chiffrer les données de la BD.
- Elle doit définir une méthode de protection des clés de chiffrement contre l'exposition.

La performance : Le déploiement des mécanismes de chiffrement engendre une surcharge de calcul qui influence la performance des SGBDR, automatiquement les requêtes des utilisateurs. L'impact est général sur tout le système d'informations. La première mesure à prendre en compte dans une conception d'un modèle de chiffrement est l'adoption d'un chiffrement sélectif, c'est à dire chiffrer que les données sensibles et laisser les données insensibles intactes. Un autre facteur très important est la minimisation de la charge du chiffrement/déchiffrement par le fait d'utiliser une seule opération au moment du chiffrement ou de déchiffrement.

Taille de la BD : La BD chiffrée ne doit pas être trop volumineuse par rapport à la BD d'origine.

Influence dans l'architecture du SGBDR : Le modèle ne doit générer aucune modification de l'architecture interne ni des fonctionnalités du SGBDR. Une nouvelle implémentation doit conserver les fonctionnalités internes du SGBDR (exemple : index, clé primaire et étrangère etc.).

2. Etat de l'art sur les modèles de chiffrement des Bases de Données

Afin de bien assimiler l'importance des modèles de chiffrement des BD qui constituera le cœur de cette thèse, nous allons présenter dans cette partie les meilleurs travaux que nous avons étudiés dans la littérature.

2.1. Le modèle d'Elovici et al.

Ce modèle a fait l'objet d'un brevet d'invention en 2018. Il est présenté en tant que modèle originale de chiffrement de BD implémenté au sein du SGBDR [28]. C'est un modèle qui permet d'assurer le chiffrement des données avec une transparence totale vis-à-vis des applications. La première version de ce modèle a été publiée par Elovici et al dans [85]. Elle a

été améliorée ensuite par Shmueli et al. [79]. Le concept s'appuie sur l'implémentation du module de chiffrement dans la mémoire cache du SGBDR, juste au-dessus de la couche « Storage Engine ». La technique de chiffrement des données consiste à utiliser les cordonnées des données de la table, chaque donnée est chiffrée avec ses propres cordonnées de la table de la BD où elle appartient. Ce modèle permet d'atteindre un haut niveau de sécurité en éliminant les attaques statiques de fuite de données. Les résultats de la mise en œuvre du modèle ont démontré une amélioration considérable dans la performance par rapport à l'implémentation du même modèle dans d'autres couches à savoir, la couche « Storage Engine ». La principale limite de ce modèle est son utilisation d'un seul algorithme (AES192) et une seule clé pour chiffrer les données. Cela présente un inconvénient majeur si l'attaquant arrive à obtenir la clé.

2.2. Le modèle de Sesay et al.

Il est considéré parmi les meilleurs modèles qui ont été proposés dans la littérature. Sa conception innovante offre une meilleure sécurité et une excellente limitation du temps de chiffrement/déchiffrement des données. Le modèle chiffre les données au niveau du SGBDR. Son principe est de chiffrer les données en fonction des classes de degré de sensibilité suivants : « Unclassified », « Classified » et « Private ». Le modèle adopte également une classification des utilisateurs selon deux catégories, les utilisateurs inferieurs (L1) et les utilisateurs supérieurs (L2). Les utilisateurs de classe (L1) possèdent le droit d'accéder uniquement aux données de classe « Unclassified », tandis que les utilisateurs de classe (L2) accèdent aux données des classes « Unclassified » et « Classified ». Par contre (L1) et (L2) accèdent chaque un à ses propres données de classe « Private ». La gestion d'accès des utilisateurs est pilotée par le mécanisme d'accès obligatoire (MAC) du SGBDR. Les auteurs de ce modèle ont défini une méthode assez spéciale pour générer les clés de chiffrement, une clé KJ pour chiffrer/déchiffrer les données de classe « Classified » et une clé KP pour chiffrer/déchiffrer les données de classe « Private ». Les données de classe « Unclassified » restent en claires. Elles sont publiques et tout le monde peut les consultées. Les deux clés KP et KJ sont générées à partir d'une clé principale Km unique qui est générée et stockée dans un contrôleur inviolable [77].

Ce modèle connait des limites malgré son importance au sein des contributions proposées dans la littérature. Le concept de génération des clés de chiffrement se base, comme nous avons souligné, sur la génération de ces clés à partir d'une clé Km unique, générée et stockée dans un contrôleur inviolable dans le SGBDR. Le concept du contrôleur inviolable n'a pas été

clarifié par les auteurs, deux concepts probables peuvent exister, un concept Hardware (une puce électronique) implanté dans la carte mère du serveur de BD, ou bien un concept software (générateur pseudo aléatoire par exemple) implémenté dans le dictionnaire du SGBDR. Dans les deux cas, deux problèmes peuvent apparaitre. Si le concept est Hardware une défaillance de ce dernier peut provoquer soit une rupture du chiffrement ou une altération des nouvelles définitions de chiffrement sur les colonnes sensibles, cela sans compter bien évidement le déni du service temporaire de la BD. De même, l'hébergement de la BD sur un autre serveur de BD présentera également un problème. Le rafraichissement des clés de chiffrement est impossible. Dans le cas d'un concept software, cela pose un problème vis-à-vis des attaques internes ou par administrateur. Le concept peut être désactivé facilement en créant un déni de service au serveur de la BD.

2.3. Le modèle Sallam et al.

Les auteurs de ce travail ont proposé un modèle sécurisé pour les BD multi-niveaux. Il combine entre le modèle MLR et le système de cryptage afin de résoudre les problèmes de sécurité associés au modèle MLR. Lorsqu'un administrateur crée un niveau dans la BD, le moteur de la BD créé et associe automatiquement une clé de chiffrement symétrique pour ce niveau, qui sera stockée dans la BD multi-niveaux. Chaque niveau possède sa propre clé pour chiffrer les données. L'ajout de système de chiffrement au modèle MLR permet de supprimer les attributs de classification de la BD multi-niveaux, ce qui implique une bonne réduction de la taille de la BD ainsi qu'une meilleure administration. En ce qui concerne les clés de chiffrement, elles sont stockées d'une manière masquée vis-à-vis des niveaux de classification. L'administrateur ne peut pas y accéder, il accède uniquement aux niveaux de classification des données. Le modèle possède deux particularités assez puissantes pour sécuriser les données. D'une part, il permet d'éliminer l'exposition des données déchiffrées dans la mémoire en les bloquant afin qu'elles ne soient accessibles que depuis l'instance du moteur de la BD. D'autre part, il prend en charge la sécurité multi-niveaux des données par rapport au niveau de classification des utilisateurs. Chaque utilisateur ne peut voir que les données de son niveau et du niveau inférieur [75].

2.4. Le modèle de Priebe et al

Le modèle de Priebe et al est considéré parmi les travaux les plus intéressants dans le domaine. Ses auteurs ont proposé un moteur de BD nommé « EnclaveDB » qui assure la confidentialité et l'intégrité des données et des requêtes. Le système « EnclaveDB » est

destiné à protéger les BD face aux attaques des administrateurs, ou lorsque la BD est externalisée sur un hôte du Cloud. Le système « EnclaveDB » est similaire à un modèle de BD relationnelle. L'utilisateur peut créer des tables et les consulter via des procédures SQL stockées. Les données sensibles sont stockées dans des enclaves (ou mémoires enclaves) et elles sont protégées par un matériel sécurisé à savoir, le Intel SGX. Par rapport à une BD conventionnelle, le fonctionnement du système « EnclaveDB » est un peu différent. Il compile les requêtes sur les données sensibles en utilisant un compilateur interne sur un client. Chaque requête compilée est signée, chiffrée et envoyée par la suite vers l'enclave du serveur de BD, ce dernier authentifie les demandes, décrypte les requêtes et les exécutent. Le résultat de la requête sera ainsi chiffré avant d'être renvoyé au client. Le système « EnclaveDB » garantit également l'intégrité des données, c'est une propriété très utile pour de nombreux systèmes à savoir, les systèmes bancaires ou les systèmes de votes. La protection de l'intégrité des données des tables est assurée par le matériel de l'enclave. Ce dernier assure également certains contrôles pour détecter les violations de l'intégrité au moment du traitement des requêtes [64].

2.5. Le modèle de Liu and Gai

Afin de remédier aux problèmes du chiffrement des données au niveau applicatif, les auteurs de ce travail ont proposé un concept de chiffrement assez particulier qui améliore davantage le chiffrement à ce niveau. Le concept porte le nom de « Cryptage en tant que service ». Son principe consiste à externaliser le système de cryptage en dehors des applications en tant qu'unité fournisseur de service de cryptage, indépendante vis-à-vis des applications et du serveur de la BD.

C'est un concept qui a permis d'apporter plusieurs avantages. Tout d'abord il va éliminer les d'attaques mémoires, puisque le chiffrement/déchiffrement s'effectue à l'extérieur du serveur et des postes clients de la BD. Les métadonnées sont également sécurisées par un chiffrement. En outre, l'isolement de l'unité fournisseur de service de cryptage lui permet de se trouver à l'abri des menaces. Par conséquent, le stockage des clés de chiffrement se fait en toute sécurité [47].

Le modèle de Liu and Gai adopte une granularité de chiffrement au niveau des colonnes. Le déroulement du chiffrement est divisé en deux phases : la phase d'initialisation et la phase d'exécution. La phase d'initialisation définit les colonnes sensibles à chiffrer, les algorithmes de chiffrement et les index. Toutes les métadonnées sont stockées à l'intérieur d'un fichier

appelé dictionnaire de sécurité (SD). La phase d'exécution permet de répondre à l'émission d'une requête par le fournisseur de service de cryptage, précisément le moteur de cryptage/décryptage. Ce dernier consulte le dictionnaire de sécurité pour récupérer les métadonnées nécessaires. Il transforme la requête à une nouvelle requête adaptée à la structure de la BD et il l'exécute. Le modèle de Liu and Gai s'appuie sur une gestion de clé à trois niveaux, une clé principale pour protéger la clé de l'utilisateur, la clé utilisateur et la clé de travail qui est le résultat de la clé d'utilisateur chiffrée par la clé principale. La clé de travail est la véritable clé qui chiffre les données de la BD.

2.6. Le modèle d'Al-Souly et al.

Les auteurs de ce travail proposent une méthode sécurisée et efficace pour chiffrer une BD sensible. Ils ont amélioré l'algorithme TSFS proposé par Manivannan en étendant son jeu de données à des caractères spéciaux [5], [16]. L'amélioration de cet algorithme consiste à corriger les erreurs pendant les étapes de déchiffrement et des tests de validation. Une comparaison de la version améliorée de TSFS avec DES et AES a été effectuée en démontrant une bonne performance en termes d'exécution des requêtes et de la taille de la BD. En dépit de l'apport considérable de ce travail, il ne couvre pas de manière globale tous les aspects d'un modèle de chiffrement de BD pertinent. L'approche proposée par les auteurs repose principalement sur l'utilisation d'un seul algorithme avec une seule clé. D'autant plus, le choix d'intégrer leur modèle au niveau des applications et les résultats des tests qu'ils ont obtenus ne reflètent pas la performance de leur algorithme par rapport à une implémentation au niveau du SGBDR.

Conclusion :

Nous avons présenté dans ce chapitre les différents niveaux où le chiffrement des BD doit avoir lieu. En effet, ces niveaux engendrent des limites qu'il faut prendre en considération avant toute adoption d'une stratégie de chiffrement des BD. L'étude que nous avons réalisée sur les solutions de chiffrement nous a permis de cerner le cadre général des solutions qui existent sur le marché ainsi que les capacités offertes par ces derniers. Chaque solution est basée sur un modèle de chiffrement bien précis. Finalement, nous avons abordé à la fin de ce chapitre une présentation de l'état d'art des meilleurs modèles de chiffrement que nous avons étudiés dans la littérature. Le prochain chapitre sera consacré à la proposition de notre première idée d'un modèle de chiffrement de BD.

CHAPITRE III : UN NOUVEAU MODÈLE POUR PROTEGER LA CONFIDENTIALITÉ DE LA BASE DE DONNÉES EN UTILISANT LE CONCEPT DES CLASSES DE CHIFFREMENT

Introduction :

Toute solution de chiffrement de BD est basée sur un modèle de chiffrement spécifique. C'est le concept qui détermine la manière dont les données sont chiffrées dans la BD. Un modèle de chiffrement de BD doit respecter les caractéristiques d'un modèle pertinent, il doit définir le niveau de granularité du chiffrement sécurisant au maximum les données, la nature et le nombre des algorithmes utilisés, la méthode de génération des clés, leurs emplacements et la technique de leurs protections [79]. Tout modèle reflète sa pertinence et sa puissance par la force d'apporter une sécurité élevée aux données, aux clés de chiffrement, à la performance du SGBDR et à la taille de la BD [26], [28].

Les modèles proposés par les chercheurs dans la littérature et dont nous avons déjà exposé quelques-uns dans le chapitre précédent nécessitent encore d'être développés et améliorés. À vrai dire, ils souffrent de deux problèmes fondamentaux. Premièrement, ils n'arrivent pas à combiner toutes les caractéristiques d'un modèle pertinent dans un seul modèle, parfois des modèles suggèrent une meilleure implémentation d'un niveau de granularité qui sécurise extrêmement les données ainsi qu'une meilleure performance, bien qu'ils soient limités par le fait d'utiliser un seul algorithme ou une seule clé dans le modèle pour chiffrer toute la BD. Cela rend le modèle faible face aux attaques si la clé tombe dans les mains d'une personne malveillante. Deuxièmement, les modèles de chiffrement nécessitent de l'innovation dans leurs conceptions, d'intégration de nouvelles caractéristiques pour faire face aux nouveaux défis imposés par le développement rapide de la technologie et de l'évolution exponentielle des menaces.

L'objectif de ce chapitre vise l'amélioration des modèles de chiffrement de BD par la proposition d'un nouveau modèle qu'on va nommer le "Modèle Global de Chiffrement". Ce modèle est basé sur l'utilisation d'un nouveau concept appelé les "Classes de Chiffrement". Le "Modèle Global de Chiffrement" est composé de quatre modèles, les trois premiers assurent respectivement le chiffrement des données, la génération des clés et leurs protections. Sa particularité sera démontrée via un quatrième modèle qui permet d'assurer un autre niveau de sécurité sur les tables de la BD, celui du chiffrement de la structure de la BD.

I. Description du "Modèle Global de Chiffrement"

Nous allons décrire dans cette section le fonctionnement du "Modèle Global de Chiffrement". Nous commencerons par définir ses quatre composantes, son niveau d'implémentation et la notion de "Classe de chiffrement". A la fin, une description détaillée du fonctionnement de

chaque modèle sera abordée.

1. Implémentation du "Modèle Global de Chiffrement"

Le " Modèle Global de Chiffrement " est composé des quatre modèles suivants :

1) Le modèle de chiffrement des données,
2) Le modèle de chiffrement de la structure de la BD,
3) Le modèle de génération des clés de chiffrement,
4) Le modèle de génération des clés principales (Master Keys) qui protègent les clés de chiffrement des données,

Nous proposons une implémentation du "Modèle Global de Chiffrement" à l'intérieur du SGBDR, précisément dans la couche « Data Base Management System Layer » comme illustré dans la Figure 3.1. Les quatre modèles qui le composent sont implémentés dans trois blocs distincts. Le fonctionnement de chaque bloc est résumé ci-dessous :

- Le bloc « Data Encryption Module » : ce bloc implémente le modèle (1), il s'occupe du chiffrement/déchiffrement des données de la BD. Il communique de manière directe avec le bloc « Keys Generation » qui lui fournit les clés de chiffrement nécessaires lors de chaque opération de chiffrement/déchiffrement.
- Le bloc « Data Base Structure Encryption » : ce bloc implémente le modèle (2), il s'occupe du chiffrement de la structure de la BD. Il communique avec le bloc « Keys Generation » qui lui fournit les clés nécessaires pour chiffrer la structure de la BD.
- Le bloc « Keys Generation » : ce bloc implémente le modèle (3) et (4), il assure la génération des clés de chiffrement des données, de la structure de la BD et des clés principales.

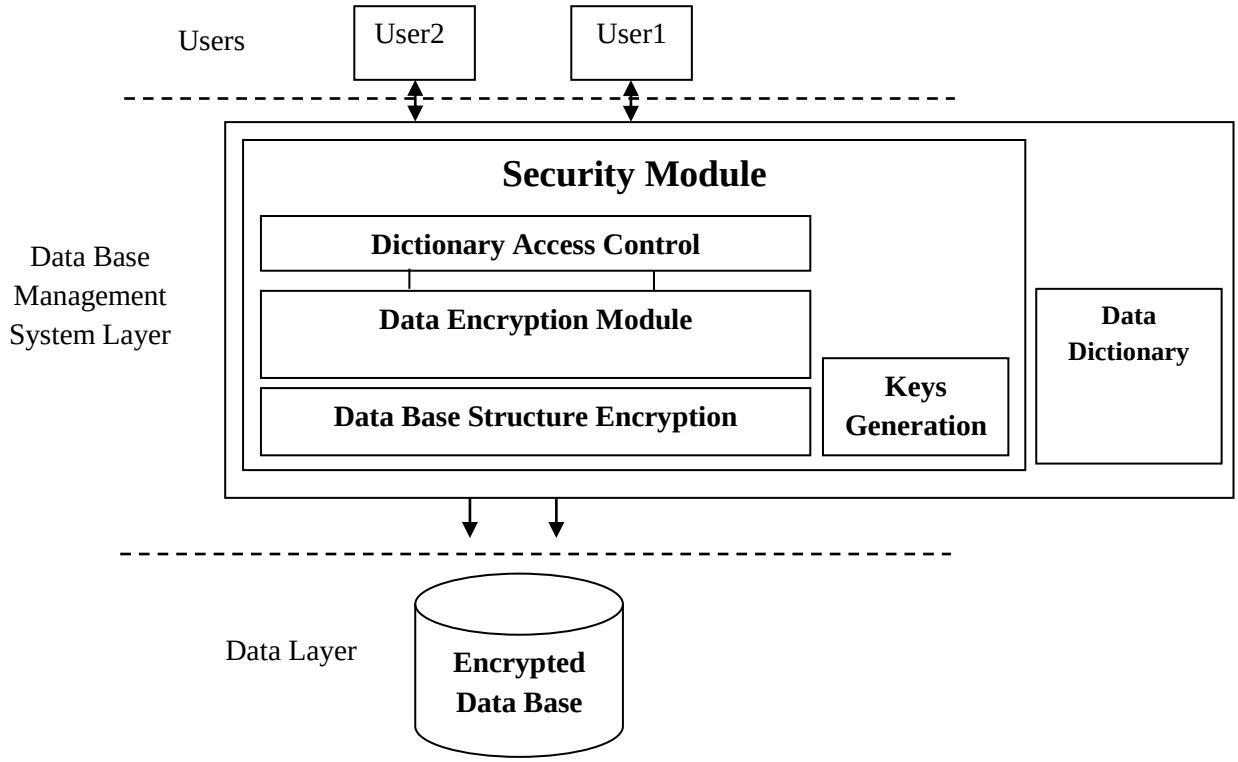

Figure 3.1: Implémentation du "Modèle Global de Chiffrement" au sein du SGBDR.

2. Le concept de "Classe de Chiffrement"

Le "Modèle Global de Chiffrement" fonctionne autour d'un concept original que nous avons développé, il est nommé "Classe de Chiffrement". Une classe de chiffrement représente le degré de sensibilité des données d'une colonne précise d'une BD, ou d'un nom de colonne. Elle a la forme d'un vecteur avec des arguments paramétrables comme représenté ci-dessous :

Class (i) (Id_class(i), Key_class(i), Algorithm(i), Sensitivity(i))

Avec :

- Id_class(i) : L'identifiant de la Class(i), il est codé sur 4 chiffres.
- Key_class(i) : La clé utilisée pour générer la clé de chiffrement de la classe.
- Algorithm(i) : L'algorithme de chiffrement utilisé par la classe.
- Sensitivity(i) : Le degré de sensibilité de la classe.

Par exemple :

Class (1) ('0001','Bigstorm1234567812345678', AES192, 'Confidential').

Class (2) ('0002','Myuniquekey@12345678@mysecretkey', 'AES256', 'Secret').

Class (3) ('0003','ilove@DBsecurity', 'AES128', 'Confidential').

Class (4) ('0004','&@I@encrypt@my@sensitive@data1@&', 'AES256', 'Secret').

2.1. Le modèle de définition d'un chiffrement sur une colonne en utilisant une "Classe de Chiffrement"

Le "Modèle Global de Chiffrement" permet de définir un chiffrement sur les colonnes sensibles au moment de la création des tables selon deux modèles. Le premier modèle est utilisé pour définir un chiffrement sur les données des colonnes, tandis que le deuxième assure une définition de chiffrement sur les noms de ces colonnes. Ces deux modèles fonctionnent en collaboration et en corrélation l'un avec l'autre. Aucune définition de chiffrement sur une colonne ne peut réussir si nous n'appliquons pas les deux modèles respectivement comme nous allons le découvrir par la suite.

Soit une table d'une BD qui possède la structure suivante :

R1 (COL(i), COL(i+1), COL(i+2),……..,COL(n)).

a. Modèle de définition de chiffrement sur les données de la colonne

La définition d'un chiffrement sur les données de la colonne COL(i) suit le modèle ci-dessous :

$$COL(i)\,Type\ encrypt\ with\ Class(x) \quad (1)$$

b. Modèle de définition de chiffrement sur le nom de la colonne

La définition d'un chiffrement sur le nom de la colonne COL(i) suit le modèle ci-dessous :

$$Struct(R1.COL(i))\,encrypt\ with\ Class(y) \quad (2)$$

Les classes Class(x) et Class(y) sont deux classes de chiffrement distinctes. L'implémentation des modèles de définition de chiffrement sur les données et les noms des colonnes n'est pas prise en considération dans ce travail.

Exemple : Soient les définitions de chiffrement (A), (B), (C) et (D) appliquées sur la table R1 (COL1, COL2, COL3) tel que :

(A): COL2 varchar2(100) encrypt with Class (1);

(B): Struct (R1.COL2) encrypt with Class (3);

(C): COL3 varchar2(100) encrypt with Class (2);

(D):Struct (R1.COL3) encrypt with Class (4);

Dans (A), nous avons défini un chiffrement sur les données de la colonne sensible COL2 en utilisant Class (1). Cela veut dire que ces données sont de nature confidentielle. Elles seront ainsi chiffrées avec l'algorithme AES192 en utilisant une clé de chiffrement qui sera générée sur la base de la clé k='Bigstorm1234567812345678'. De même, nous avons défini dans (C) un chiffrement sur les données de la colonne COL3 qui sont de nature secrète. Elles seront chiffrées avec l'algorithme AES256 en utilisant une clé qui sera générée sur la base de la clé k=' Myuniquekey@12345678@'. Dans (B) nous avons défini un chiffrement sur le nom de la colonne COL2 en utilisant Class (3). Cela veut dire que le nom de cette colonne est classé dans le degré de sensibilité « Confidential », il sera chiffré donc avec l'algorithme AES128 et une clé qui va être générée sur la base de la clé k= 'ilove@DBsecurity'. De même dans (D) nous avons défini un chiffrement sur le nom de la colonne COL3 en utilisant Class (4). Le nom de cette colonne est classé dans le degré de sensibilité « Secret ». Il sera chiffré donc avec l'algorithme AES256 avec une clé qui va être générée sur la base de la clé k=' &@I@encrypt@my@sensitive@data1@&'. Les deux tableaux (Table 3.1 et Table 3.2) ci-dessous exposent le résultat de la création et d'insertion de données dans la table (R1) avec et sans définition de chiffrement.

COL1	**COL2**	**COL3**
1000	Dupont	PW@124
2482	James	PW@884

Table 3.1: La table (R1) sans définition de chiffrement.

COL1	**C7E220367559EE77 B5221D27B92AE495**	**3F99A273812785D0 25100177BBDFA307**
1000	490A39A1BBA9888D CB1DD4158D5975F9	4DDF495C731928AF A79EC598CDB31E2
2482	2091BF0941B816B1 49654605D33A4685	9A2B6B09120DCB9 8B56FFF0233C27270

Table 3.2: La table (R1) avec définition de chiffrement.

La gestion des classes de chiffrement est pilotée par l'administrateur de la BD, elle est

obligatoirement soumise aux règles de gestion suivantes :

- Chaque classe Class(i) possède un identifiant Id_class et une Key_class unique.
- Une classe Class(i) peut être utilisée pour chiffrer les données d'une colonne ou de plusieurs colonnes. Identiquement, une classe Class(i) peut être utilisée pour chiffrer le nom d'une colonne ou de plusieurs colonnes.
- Un chiffrement définit sur les données d'une colonne COL(i) en utilisant Class(i) ne peut pas fonctionner si nous n'avons pas défini un chiffrement sur le nom de cette colonne en utilisant soit Class(i) ou une autre classe Class(j) et vice versa.
- L'administrateur de la BD gère la création, la mise à jour, l'attribution ou la révocation des classes aux (données et noms des colonnes) de la BD selon les besoins de la sécurité. Cette gestion est assurée à l'intérieur du SGBDR par le mécanisme de contrôle d'accès DAC (Figure 3.1).

3. Le modèle de chiffrement des données

Le modèle de chiffrement des données assure le chiffrement des données de la BD au niveau des cellules des tables. Chaque valeur de cellule est chiffrée de manière indépendante et différente des autres cellules selon le modèle définit ci-dessous :

$$E\left(Data1\right) = E\left(K_{COL(i),}Data1 \parallel P\left(\left(rownum+1\right) + Column\_id + Table\_id + Id\_class(i)\right)\right) \quad \textbf{(3.1)}$$

Soit X une valeur chiffrée comme définie ci-dessous :

$$E\left(Data1\right) = X$$

Lors de la consultation des colonnes, les valeurs chiffrées sont déchiffrées selon le modèle défini ci-dessous :

$$D\left(X\right) = Rep\left(D\left(K_{COL(i)}, X\right), P\left(\begin{matrix}\left(rownum\right) + Column\_id + \\ Table\_id + Id\_class\left(i\right)\end{matrix}\right)\right) \quad \textbf{(3.2)}$$

Dans les modèles (3.1.) et (3.2) nous avons :

- E : un algorithme de chiffrement symétrique.
- Data1 : donnée claire à chiffrer située dans la colonne COL(i).
- K COL(i) : la clé de chiffrement/déchiffrement de la colonne COL(i), la méthode de sa génération sera décrite dans la partie (4).
- Rownum : c'est un entier qui représente le numéro du dernier enregistrement dans une table, précisément la cellule contenant la valeur (Data1).
- Column_id: c'est un entier unique qui représente l'identifiant de la colonne COL(i).
- Table_id: c'est un entier unique qui représente l'identifiant de la table où se trouve COL(i).
- Id_class(i)): c'est l'entier qui représente l'identifiant de la classe (Class(i)) utilisée pour chiffrer les données de la colonne COL(i).
- Rep (str1, str2): c'est une une fonction qui annule la chaîne (str2) qui existe dans la chaîne (str1).
- P : c'est un polynôme qui est défini de la manière suivante :

$$P : N \rightarrow N$$

$$P(n) = a_0 n^0 + a_1 n^1 + a_2 n^2 + a_3 n^3 + \ldots\ldots\ldots a_n n^n \qquad \textbf{(4)}$$

Les facteurs du polynôme (a0, a1, a2, …, an) sont des entiers que nous fixons dans l'algorithme général de chiffrement/déchiffrement, c'est à dire dans l'algorithme qui exécute les modèles (3.1) et (3.2), cela montre que nous pouvons modifier le chiffrement des données en changeant ces facteurs. La valeur fournie par P ((rownum+1) + Column_id + Table_id + Id_class(i) est unique à chaque fois que nous voulons chiffrer une cellule d'une colonne, puisque (rownum) se change d'une cellule à une autre. Toutes les valeurs des cellules égales sont chiffrées d'une manière différente l'une par rapport à l'autre. Cette technique augmente énormément la sécurité du chiffrement en éliminant la probabilité des attaques statiques.

4. La protection de la structure d'une Base de Données par le chiffrement

La protection de la structure d'une BD (les noms des tables et des colonnes) par le chiffrement est une manière de sécuriser la BD en plus des valeurs de données stockées dans les colonnes. Un attaquant pourrait se servir des informations de la structure pour mener des attaques à savoir, deviner les distributions de données [78]. Le chiffrement de la structure peut

apporter plus de sécurité à une BD et renforce la complexité pour un attaquant de déchiffrer ses données. Ainsi, ce dernier peut être intéressé à attaquer seulement les données d'une table ou d'une colonne précise (table des clients ou colonne des numéros des cartes de payement par exemple). En adoptant une structure de BD en claire et quel que soit sa taille, petite ou grande en nombre de table ou de colonne, et malgré que le chiffrement concerne que les données, un attaquant arrive aisément à se focaliser sur une ou plusieurs colonnes pour les déchiffrer. Par contre, si la structure est également chiffrée (cas d'une BD qui contient 1000 tables par exemple), la complexité de l'attaque est multipliée par 1000. Dans une BD à structure chiffrée, tant que le nombre de tables et de colonnes augmente, tant que la complexité pour l'attaquer devient énormément élevée.

Un autre type de protection de la structure d'une BD dite par le droit d'auteur, conformément aux dispositions de l'article L.112-3 et suivant le code de la propriété intellectuelle [20]. Le droit d'auteur s'applique au contenant de la BD, c'est à dire à sa structure, à la manière dont les données sont disposées et classées, à condition qu'elle soit originale. Les entreprises qui conçoivent et produisent des BD métiers sont confrontées à des difficultés majeures pour générer certaines conceptions de BD au profit de quelques domaines d'activités qui ne sont pas facile à modéliser. Pendant des colossaux travaux de réflexion et de modélisation ainsi que des couts d'investissements importants, le résultat de la structure de la BD générée reste un travail confidentiel à l'entreprise et un secret métier qu'il faut préserver contre toute divulgation à raison du défi de la concurrence. Par conséquent, la protection de la confidentialité d'une telle structure par le chiffrement reste un recours indispensable pour les entreprises afin de garantir la pérennité de leurs projets.

4.1. Le modèle de chiffrement de la structure de la Base de Données

Le "Modèle Global de Chiffrement" adopte le concept du chiffrement de la structure de la BD dans le but de bénéficier de deux avantages majeurs :

- Chiffrer la structure de la BD pour bénéficier de tous les avantages cités précédemment.
- Mettre à côté du chiffrement des données, un autre niveau de sécurité ou de contrôle qu'il faut franchir avant d'accéder aux données, soit par chiffrement ou déchiffrement. La règle suivante est adoptée par notre "Modèle Global de Chiffrement" : un utilisateur, qui possède une classe qui chiffre/déchiffre les données d'une colonne, ne peut pas accéder aux données sans posséder la classe qui chiffre/déchiffre le nom de

cette colonne. Le bloc (DAC) de la Figure 3.1 n'accepte pas d'exécuter l'instruction (A) ci-dessous sans l'exécution de l'instruction (B) et vice versa.

/* Définition d'un chiffrement sur les données de COL(i)*/

COL(i) Type encrypt with Class(x); (A)

/* Définition d'un chiffrement sur le nom de la colonne COL(i)*/

Struct (R1.COL(i)) encrypt with Class(y); (B)

Soit une table d'une BD qui possède la structure suivante :

R(i) (COL(i), COL(i+1), COL(i+2), COL(i+3)).

Soit les deux classes de chiffrement Class (i) et Class (j):

Class (i) (id_class(i), key_class(i), Algorithm(i), Sensitivity(i))

Class (j) (id_class(j), key_class(j), Algorithm(j), Sensitivity(j))

Soit les deux définitions de chiffrement sur les noms des deux colonnes sensibles COL(i+1) et COL(i+3).

Struct R(i).COL(i+1) encrypt with Class(i);

Struct R(i).COL(i+3) encrypt with Class(j);

La structure chiffrée de R(i) suit le modèle suivant :

$$R(i)\begin{pmatrix} COL(i),\ H\left(E\left(COL(i+1)\right)\right), \\ COL(i+2),\ H\left(E\left(COL(i+3)\right)\right) \end{pmatrix} \quad \textbf{(5)}$$

Avec :

- H : Une fonction de hachage dont le résultat ne dépasse pas 32 caractères.
- E : Un algorithme de chiffrement symétrique.

5. Le modèle de génération des clés de chiffrement

Le bloc « Keys Generation » génère deux types de clés, les clés de chiffrement des données des colonnes sensibles et les clés de chiffrement des noms de ces colonnes. La génération des deux clés suit les modèles définis ci-dessous :

$$K_{COL(i)} = H\left(Table\_id \parallel Column\_Id \parallel key\_class(i)\right) \quad \textbf{(6)}$$

$$K_{Struct\ (COL(i))} = H\left(Table\_id \parallel Column\_Id \parallel key\_class(i)\right) \quad \textbf{(7)}$$

Avec :

$K_{COL(i)}$: la clé de chiffrement des données de la colonne sensible COL(i).

$K_{Struct\ (COL(i))}$: la clé de chiffrement du nom de la colonne sensible COL(i).

H : une fonction de hachage.

Table_id: c'est un entier unique qui représente l'identifiant de la table où se trouve COL(i).

Column_id : c'est un entier unique qui représente l'identifiant de la colonne COL(i).

key_class(i)): la clé définie dans la classe qui est attribuée à la colonne COL(i). Nous rappelons que cette clé est unique dans chaque classe.

Les modèles (6) et (7) de génération des clés sont identiques. Dans une table de BD, si COL(i) et Struct(COL(i)) sont chiffrés par des Class(i) distinctes, nous aurons les valeurs de $K_{COL(i)}$ et $K_{Struct\ (COL(i))}$ qui sont différentes. Si le contraire nous aurons $K_{COL(i)} = K_{Struct\ (COL(i))}$. Deux colonnes sensibles COL(i) et COL(ii) qui appartiennent à deux tables différentes ne seront jamais chiffrées par les mêmes clés même si elles utilisent la même classe et c'est le même cas si elles appartiennent à la même table.

6. Le modèle de génération de la «Master Key»

La sécurité des données chiffrées dans une BD dépend de la protection des clés. La gestion de ces clés est un processus fondamental pour la sécurité globale du SGBDR [22], [39]. Nous proposons de protéger les clés de chiffrement de notre "Modèle Global de Chiffrement" via leur chiffrement avec une clé principale (km). La génération de cette clé suit le modèle ci-dessous :

$$Km_{(COL(i))} = H\left(Table\_name \parallel COL(i)\right) \quad \textbf{(8)}$$

H : une fonction de hachage.

COL(i) : le nom de la colonne sensible.

Table _name : le nom de la table ou appartient la colonne COL(i).

La valeur de Km est unique pour chaque colonne sensible COL(i). Afin de ne pas impacter la performance de notre modèle, nous proposons qu'il n'y ait pas de protection pour ces clés via

les clés principales (Km).

II. Implémentation du "Modèle Global de Chiffrement"

Nous allons présenter dans cette section une comparaison analytique entre le "Modèle Global de Chiffrement" et deux modèles de chiffrement de BD. Le premier modèle est proposé par Sesay et al., le second est proposé par Shmueli et al. Dans la littérature ces deux modèles sont les plus proches de notre approche en termes de protection de la confidentialité de la BD par le chiffrement. En outre, ils fournissent une sécurité maximale à la BD par leurs concepts originaux. Le modèle proposé par Shmueli et al., a fait l'objet d'un brevet en 2018, celui proposé par Sesay et al., a introduit un nouveau concept de la hiérarchisation des utilisateurs et la catégorisation des données selon des classes de sensibilité. Les tableaux comparatifs 3.3 et 3.4 exposent le résultat de la comparaison.

	Algorithmes utilisés	**Clés (utilisation / génération)**	**Niveau de sensibilité des données**
Modèle Global de Chiffrement	• Le modèle utilise plusieurs algorithmes pour chiffrer les données. • Le modèle utilise plusieurs algorithmes pour chiffrer la structure de la BD. • Le modèle permet d'implémenter d'autres algorithmes de chiffrement outre que les algorithmes de chiffrement conventionnels tels que (DES, AES, …).	• Le modèle génère et utilise plusieurs valeurs de clés de chiffrement : les clés pour chiffrer les données de colonnes et les clés pour chiffrer les noms de colonnes.	• Le modèle définit plusieurs niveaux de sensibilité des données. • Le modèle permet de créer d'autres niveaux de sensibilité de données. • Le modèle permet de définir des niveaux plus fins à partir d'un seul niveau de sensibilité de données.
(Sesay et al., 2005)	• Le modèle utilise un seul algorithme pour chiffrer la BD.	• Le modèle utilise une clé unique pour chiffrer les données de sensibilité "Classified" et plusieurs clés pour chiffrer les données de sensibilité "Private".	• Le modèle définit trois niveaux de sensibilité des données (Unclassified, Classified et Private). • Le modèle ne permet pas de créer un autre niveau de sensibilité des données.
(Shmueli et al., 2014)	• Le modèle utilise un seul algorithme pour chiffrer la BD.	• Le chiffrement des données utilise une valeur de clé unique.	• Le modèle définit un seul niveau de sensibilité de données. • Le modèle ne permet pas

			de créer un autre niveau de sensibilité des données.

Table 3.3: Comparaison 1.

	Protection des clés de chiffrement	**Niveau de granularité du chiffrement**	**Protection de la structure de la BD**
Modèle Global de Chiffrement	• Le modèle protège les clés de chiffrement conformément au modèle de génération des clés principales.	• Le modèle chiffre les données au niveau de la cellule.	• Le modèle chiffre la structure de la BD conformément au modèle de chiffrement de la structure de la BD.
(Sesay et al., 2005)	• Aucun modèle défini pour protéger les clés de chiffrement.	• Le modèle chiffre les données de type "Private" au niveau de la cellule et les données de type "Classified" au niveau de la colonne.	• Aucun modèle défini pour protéger la structure de BD.
(Shmueli et al., 2014)	• Aucun modèle défini pour protéger les clés de chiffrement. • Proposition d'approches conventionnelles pour protéger les clés: Wallet, HSM.	• Le modèle chiffre les données au niveau de la cellule.	• Aucun modèle défini pour protéger la structure de la BD.

Table 3.4: Comparaison 2.

Dans le but de concrétiser le fonctionnement du "Modèle Global de Chiffrement" et afin de mener une discussion objective sur l'analyse des résultats de la comparaison, une mise en œuvre d'un cas réel sera conjointement présentée ci-dessous.

1. Etude de cas

Considérons la table "employé" dans une BD qui possède la structure suivante :

employé (code, prénom, nom, salaire)

Supposant que les colonnes prénom, nom, salaire de la table "employé" sont sensibles possédant le même niveau de sensibilité de données et le même niveau de sensibilité des noms de colonnes.

Soit la classe de chiffrement Class (1) telle que :

Class (1) ('0001', 'ilove @ DBsecurity', 'AES256', 'Confidential')

L'application des modèles (1) et (2) donne les définitions de chiffrement suivantes :

prénom varchar2(100) encrypt with Class (1);

Struct (employé. prénom) encrypt with Class (1);

nom varchar2(100) encrypt with Class (1);

Struct (employé. nom) encrypt with Class (1);

salaire varchar2(100) encrypt with Class (1);

Struct (employé. salaire) encrypt with Class (1);

La table "employé" possédera une nouvelle structure après le chiffrement en appliquant les modèles (3.1) et (5) comme indiqué dans le tableau 3.6. Le tableau 3.7 illustre la génération via les modèles (6) et (8) des clés de chiffrement des données et leurs clés de protection. De même, le tableau 3.8 illustre la génération via le modèle (7) des clés de chiffrement des noms de colonnes.

code	**prénom**	**nom**	**salaire**
0001	Paul	Williams	2000
0002	Paul	Watson	4000
0003	Paul	Stevens	3000
0004	Paul	Diaz	6000

Table 3.5: la table « employé » avant le chiffrement.

code	**E7762D87BE5F94520 0E1D6D6FB4BAE13**	**6CD6345907C75C60 24A9F30EF5114128**	**12D000F12B139674 60FD1BDDFDC76A39**
0001	9A92AED8F7556155 627AB4A3C5F04E23	47CF2D656BF168CA5 8757A88A78E5AFE	7C83A5BF4DBB903F D707D581E05CA78B
0002	0A9DDD8178B59885 8BF601AA80285671	B29D5A3EBC24E9C8E 60C40927FCE3539	CD1D6512C2AEFB7 D48CF343BECE9E3DC
0003	62DE828D74C326B8 72897703CD29FCAF	F7DAF7E6EC8EFB39 93ACEAC1617F55AD	4D5E1EF3C302F3E1 90EC4A93F6C94DAA
0004	5B30556038840D1CE 6D4030616B77E0A	BE8BBEB4BAFF6C63 297619F9B1293B48	36BD51A6E388781DF AF33B7647F66642

Table 3.6: La table « employé » après le chiffrement.

Nom de la colonne	La clé de chiffrement des données	La clé principale associée
prénom	9E9A833CD2243929 E932A212610AE8C4	3A58508BE387C914 2ED397047EEC5BE1
nom	280633F706590011F E8DE6B8C2807B04	51995FCA5D1A5F62 8BE8029B95E414B5
salaire	C3D81B8FEE00F1880 BD57EC2031521ED	C052C4B7867D8497 D1D3F9D4BC366325

Table 3.7: Génération des clés de chiffrement des données et leurs clés principales de protection.

Nom de colonne	Les clés de chiffrement des noms de colonnes
prénom	9E9A833CD2243929E932A212610AE8C4
nom	280633F706590011FE8DE6B8C2807B04
salaire	C3D81B8FEE00F1880BD57EC2031521ED

Table 3.8: Les clés de chiffrement générées pour les noms de colonnes.

2. Résultats et discussions

Selon la comparaison que nous avons présentée dans les tableaux 3.3 et 3.4, et compte tenu de la mise en œuvre effectuée dans l'étude de cas, nous avons choisi de représenter notre analyse des résultats sur la base de 6 critères.

a. 1er critère : utilisation d'algorithmes

Le concept de la classe de chiffrement que nous proposons permet d'offrir la possibilité d'utiliser plusieurs algorithmes (via l'argument Algorithm(i)) pour chiffrer les données et la structure dans la même BD, ce qui offre une forte sécurité à la BD. Cette fonctionnalité est originale à notre "Modèle Global de Chiffrement" par rapport aux modèles présentés au tableau 3.3, même de tous les modèles proposés dans la littérature. Un autre avantage fourni par ce concept est la possibilité de créer et d'implémenter de nouvelles classes spécifiques prenant en charge d'autres algorithmes de chiffrement outre que les algorithmes de chiffrement conventionnels à savoir, DES, AES, 3DES, ... etc. A titre d'exemple, implémenter l'algorithme de chiffrement de BD proposé par [76].

b. 2ème critère : clés (utilisations / génération)

Le concept de la classe de chiffrement donne une spécificité à notre modèle par rapport aux autres, il offre la possibilité d'utiliser plusieurs clés pour chiffrer les colonnes et la structure de la BD. Cela nous permet d'éviter le chiffrement naïf des données et de respecter les caractéristiques d'un modèle pertinent de chiffrement de BD que nous décrivons dans le chapitre 2. Les résultats de l'implémentation obtenus dans le tableau 3.7 montrent que chaque colonne sensible (prénom, nom, salaire) de la table "employé" possède sa propre clé de chiffrement. Le nombre de clés généré est égal au nombre des colonnes sensibles. La valeur de l'argument Key_class (i) participe à la génération de la clé de chiffrement réel (modèles (6) et (7)) avec une méthode aléatoire, ce qui rend cette valeur inconnue pour l'administrateur. Toute modification de cet argument modifie automatiquement la clé réelle de chiffrement. Un autre avantage du concept de classe de chiffrement est la possibilité de créer des classes dérivées à partir d'une seule classe. A titre d'exemple, nous pouvons utiliser plusieurs classes pour chiffrer une BD ayant les mêmes algorithmes de chiffrement ; c'est-à-dire les mêmes valeurs des arguments (Algorithme (i)), mais qui se différencient par les arguments (Key_class (i)). Bien entendu, ces classes dérivées doivent avoir des Id_class (i) distinctes.

c. 3ème critère : Niveau de sensibilité des données

Dans le "Modèle Global de Chiffrement" les données sensibles peuvent appartenir à un niveau de sensibilité des données ou à plusieurs niveaux de sensibilité des données. La valeur de sensibilité est fixée par l'argument (Sensitivity(i)) dans les classes. Ce concept permet de chiffrer les données en fonction de leur niveau de sensibilité à l'aide de différentes valeurs d'arguments (Key_class(i)) et (Algorithm(i)).

Les classes de chiffrement nous permettent aussi de créer d'autres classes avec une granularité plus fine tout en ajustant l'argument (Sensitivity (i)). Par exemple, si les données des deux colonnes COL1 et COL2 sont de nature secrète et si le niveau de sensibilité des données de COL2 est plus important que COL1, nous pouvons définir un chiffrement sur COL1 en utilisant Classe (i) en fixant l'argument (Sensitivity(i)) à "Secret" et un autre chiffrement sur COL2 en utilisant Classe (j) en fixant son argument (Sensitivity(j)) à "Top secret". Les deux classes doivent avoir des valeurs d'arguments (Key_class (i)) et (Algorithme (i)) distinctes.

Le chiffrement de la BD, via le "Modèle Global de Chiffrement", selon des niveaux de sensibilité des données apporte deux avantages majeurs. D'abord, il n'autorise le chiffrement/déchiffrement que des données sensibles, cela améliore extrêmement la performance car seules les parties sensibles de la BD sont à chiffrer/déchiffrer lors de l'exécution des requêtes. En outre, chiffrer uniquement la partie sensible de la BD offre une

parfaite optimisation de la taille de la BD.

d. 4ème critère : Protection des clés de chiffrement

L'utilisation d'un modèle spécifique dédiée à la protection des clés de chiffrement au sein du serveur de BD est la particularité du "Modèle Global de Chiffrement". Comme le montre le tableau 3.4, ce concept n'existe pas dans les modèles développés par Shmueli et al., et par Sesay et al.. Notre solution élimine la mise en œuvre des approches classiques à savoir, Wallet, HSM, serveur de sécurité, etc. Nous allons démontrer dans le chapitre 4 les limites susceptibles de décroitre la sécurité de la BD lors de l'adoption de ces approches. Le "Modèle Global de Chiffrement ", via le modèle (8), génère les clés principales automatiquement au moment de la définition du chiffrement sur les données, il n'y a pas de stockage défini pour ces clés, les administrateurs de la BD ne peuvent donc pas y accéder. Les résultats illustrés dans le tableau 3.7 montrent que chaque colonne sensible possède sa propre clé de chiffrement (KCOL (i)) qui possède à son tour sa propre clé de protection (Km (COL (i)). Ce concept renforce énormément la sécurité de la BD et rend l'obtention des clés par des attaquants une opération quasiment impossible.

e. 5ème critère : Niveau de granularité du chiffrement

Le niveau de granularité du chiffrement du "Modèle Global de Chiffrement" est fixé au niveau des cellules des colonnes. Le chiffrement à ce niveau offre une meilleure sécurité aux données, chaque cellule est chiffrée indépendamment et différemment des autres. Comme nous le constatons dans les tableaux 3.5 et 3.6, les valeurs égales de la colonne "prénom" sont chiffrées différemment l'un de l'autre. En utilisant cette technique, nous renforçons la sécurité de la BD contre la probabilité d'attaque par analyse de fréquence.

f. 6ème critère : Protection de la structure de la Base de Données

Par rapport aux modèles comparatifs du tableau 3.4, le chiffrement de la structure est une spécifié du "Modèle Global de Chiffrement", c'est une autre ligne de défense pour la BD à côté du chiffrement des données et du mécanisme de contrôle d'accès. Le "Modèle Global de Chiffrement" permet la mise en œuvre de deux niveaux de sécurité sur la BD. Le premier niveau s'applique sur les données des colonnes, le second est sur les noms de ces colonnes (voir tableau 3.6). Un utilisateur de la BD ne peut pas accéder aux données sans disposer des classes appropriées pour les deux niveaux.

Conclusion :

Dans ce chapitre nous avons présenté un nouveau modèle de chiffrement de BD appelé

"Modèle Global de Chiffrement". Il est basé sur un nouveau concept que nous avons développé, celui des "Classes de chiffrement ". En le comparant avec d'autres modèles, le "Modèle Global de Chiffrement" répond parfaitement aux exigences d'un modèle pertinent de chiffrement de BD. Cela a été prouvé par une étude comparative que nous avons réalisée avec d'autres modèles sur la base de plusieurs critères ainsi que les résultats fournis de son implémentation.

CHAPITRE IV : UNE NOUVELLE APPROCHE DE PROTECTION DES CLÉS DE CHIFFREMENT AU SEIN DES SYSTÈMES DE GESTION DE BASE DE DONNÉES

Introduction :

Le chiffrement des BD s'appuie sur l'utilisation des clés de chiffrement, la gestion de ces clés est un point fondamental. Elle définit la méthode dont les clés sont générées, stockées et protégées au cours de leurs utilisations jusqu'à leurs destructions. La connaissance de la valeur des clés, la manière dont les utilisateurs accèdent et leurs lieux de stockage est le but ultime de tout attaquant. Or, il est nécessaire de leur instaurer une politique de protection pour réduire au maximum leur exposition.

A ce titre, certain nombre de solutions ont été développées afin de remédier à ce problème, nous citons ainsi les travaux de [45], [82]. Quelques modèles de chiffrement de BD que nous avons abordés dans le chapitre 2 proposent aussi dans leurs conceptions des solutions de génération et de protection des clés à savoir, les travaux de [28], [47], [79]. Les solutions de protection des clés qu'implémentent les SGBDR tels qu'Oracle, Ms SQL Server et MySQL se basent principalement sur l'utilisation des « Wallet », HSM et dans des cas particuliers l'ajout d'un serveur de sécurité dédié [60], [62]. A vrai dire, chaque une de ces solutions présentent des avantages et des limites que nous allons détailler dans ce chapitre. Une solution fiable, générique et adaptée à plusieurs niveaux de granularité de chiffrement n'a pas encore été proposée dans le domaine.

L'objectif de ce chapitre vise à proposer une amélioration du concept de la protection des clés de chiffrement dans les BD. Notre nouvelle approche est basée sur des modèles que nous suggérons implémenter à l'intérieur des architectures des SGBDR pour assurer une protection forte des clés de chiffrement. Nos modèles protègent les clés en chiffrant ces derniers par des clés principales. Chaque clé principale sera générée selon un modèle spécifique en fonction du niveau choisi de granularité du chiffrement de la BD.

I. Les approches de protection des clés de chiffrement dans les Systèmes de Gestion des Bases de Données

Dans cette section nous allons présenter les différentes approches de protection des clés de chiffrement déployées par les SGBDR. Nous allons analyser ces approches, déterminer les limites de chacune ainsi que de proposer notre nouvelle approche.

1. Approche basée sur les « Wallet »

Cette solution consiste à stocker les clés de chiffrement dans une table du dictionnaire de données, ou dans un fichier du serveur de BD et d'accéder à ces lieux de stockage de manière

restreinte. Les clés seront ainsi chiffrées par une autre clé appelée clé principale (Km) qui est stockée (en la chiffrant elle-même par un mot de passe) dans un « Wallet » qui se situe sur le même serveur de BD [40], [62]. La principale limite de cette solution réside dans le fait que les administrateurs qui possèdent des droits privilégiés sur le serveur de BD peuvent détenir la clé principale et déchiffrer toutes les données sans laisser de traces.

2. Approche par « HSM »

Le principe de la solution porte sur la génération et le stockage de la clé principale Km dans un équipement appelé HSM. Un HSM « Hardware Security Module » est un matériel qui ressemble au crypto processeur TPM, il est généralement connecté directement au port PCI de l'ordinateur [40]. Le serveur de BD protège les clés par le chiffrement de ces derniers avec la clé Km générée par le HSM. Au moment de l'exécution d'une requête (insertion de données, mises à jour, ou consultation de données), le serveur de BD déchiffre les clés de chiffrement dans sa propre mémoire automatiquement en utilisant la clé Km. A la fin de l'exécution de la requête, le serveur supprime de la mémoire les clés et s'apprête à une autre opération de chiffrement/déchiffrement [45]. La Figure 4.1 schématise le fonctionnement de l'approche HSM.

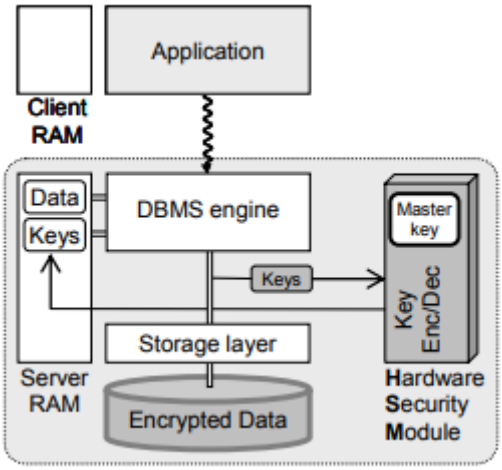

Figure 4.1: Approche par HSM [45].

3. Approche par « Serveur de sécurité »

Cette solution consiste à mettre en place un serveur de sécurité, physiquement distinct à celui qui abrite la BD et lui confier les taches de la gestion des utilisateurs, droits d'accès et les clés de chiffrement. Comme il est schématisé dans la Figure 4.2, le serveur de BD fonctionne en collaboration avec le serveur de sécurité, il chiffre et déchiffre les données en faisant appel

aux clés détenues par le serveur de sécurité. L'avantage principal de cette approche est de séparer complètement les taches de l'administration de la BD à celles de la sécurité de la BD afin de limiter tout accès illégal aux clés de chiffrement, ce qui renforce la sécurité des données. Pour mener une attaque sur la BD, il faut absolument établir une collaboration entre l'administrateur de la BD et l'administrateur de sécurité [40], [45].

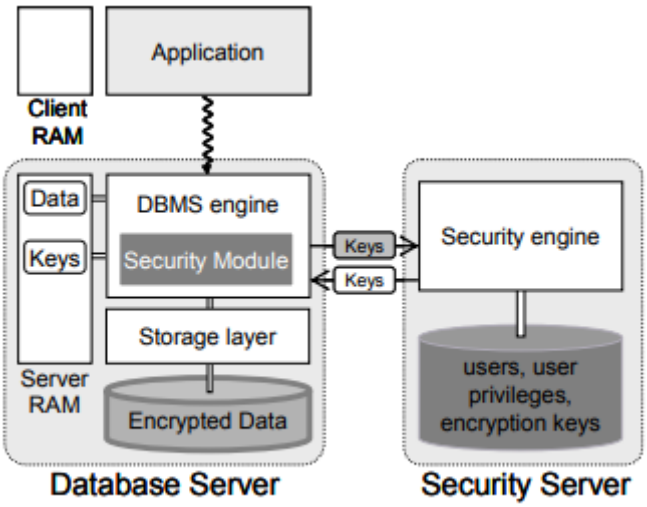

Figure 4.2: Approche par « Serveur de sécurité » [45].

Dans les deux approches décrites ci-dessus, l'ajout d'un « HSM » ou du « Serveur de sécurité » minimise effectivement l'exposition des clés de chiffrement, bien qu'elles ne protègent pas complètement la BD. L'apparition brève des clés et des données est toujours possible dans la mémoire, ce qui peut être toujours la cible des attaquants.

4. L'approche par «HW Security Module»

Bouganim et al., ont proposé une nouvelle approche de protection des clés basée sur la fusion des deux approches que nous avons expliquées précédemment : l'approche par « HSM » et par « Serveur de sécurité ». Il s'agit ici d'une intégration du serveur de sécurité dans un HSM pour former un seul module appelé « HW Security Module ». L'idée de base est de profiter du HSM pour éviter l'exposition des clés de chiffrement dans la mémoire du serveur au cours du processus de chiffrement /déchiffrement.

4.1. L'approche « Serveur-HSM »

La solution consiste à intégrer le module « HW Security Module » à l'intérieur du serveur de la BD. Ce module gère les privilèges des utilisateurs, le chiffrement et les clés. Cette solution

offre les mêmes avantages que ceux par approche « Serveur de sécurité ». Cependant, l'exposition des clés est éliminée ici avec ce concept puisque le chiffrement /déchiffrement s'effectue à l'intérieur du HSM. Dans ce cas, une attaque de type interne, externe ou même par l'administrateur sur la BD n'aboutira plus. Tous les éléments (clés de chiffrement, gestion des utilisateurs et privilèges, etc.) sont intégrés dans le HSM du module « HW Security Module » [45].

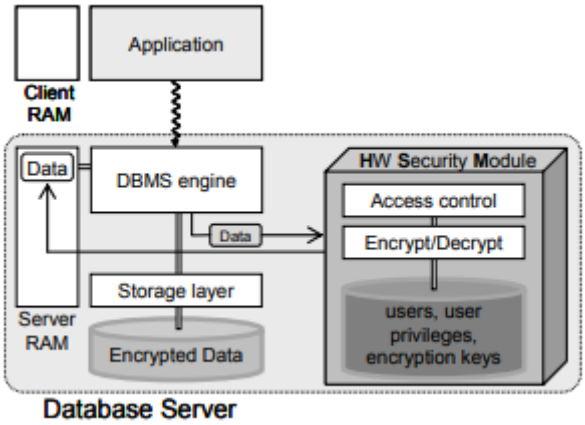

Figure 4.3: Approche par « Serveur HSM » [45].

5. Les limites des approches de protection des clés de chiffrement

Les différentes solutions que nous avons présentées ci-dessus permettent de résoudre le problème de l'exposition des clés de chiffrement face aux susceptibles menaces des attaquants. Cependant, ces solutions engendrent des limites qui peuvent décroitre la sécurité de la BD. Les trois points ci-dessous sont à prendre en considération :

- Dans l'approche par « Serveur de sécurité », il est nécessaire de sécuriser l'envoi entre le serveur de BD et le serveur de sécurité notamment si les machines sont distantes. Les clés et les données transitent en claires à travers la voie de communication. D'autant plus, il est nécessaire d'avoir un administrateur pour la BD et un autre pour le serveur de sécurité, ce qui n'est pas possible lorsque nous souhaitons que toutes les tâches relatives à l'administration de la BD soient effectuées par une seule personne, soit par manque des moyens ou de ressources nécessaires. Dans le cas si la BD est gérée par deux administrateurs, la probabilité d'une attaque par conspiration n'est pas négligeable.
- Dans une approche par « Wallet », d'autres problèmes de sécurité apparaissent dans le

cas où le mot de passe du « Wallet » est compromis par une personne malveillante. Cette solution est limitée principalement par la forte probabilité d'une attaque par les administrateurs qui disposent des accès privilégiés sur la BD, ils peuvent détenir toutes les clés et déchiffrer toutes les données sans laisser aucunes traces. Un autre point relatif à la sauvegarde du « Wallet » s'ajoute aussi, il faut obligatoirement utiliser un système de sauvegarde pour assurer une sécurité extrême. Cela doit se faire soit immédiatement après sa création, au moment du changement de la clé principale, ou lors du changement du mot de passe du « Wallet ». Le « Wallet » est un composant critique qu'il faut sauvegarder dans un endroit sûr [62].

- Concernant l'approche HSM, elle est considérée la plus efficace des solutions, elle est conditionnée par le HSM lui-même, sa présence à côté du serveur de BD est nécessaire. Cela constitue un obstacle dans des contextes où le HSM présente une surcharge et un cout supplémentaire pour mettre en place une politique de sécurité [40], [45]. D'autant plus, une défaillance de ce dernier interrompt totalement la protection des clés, ce qui influence automatiquement le processus du chiffrement /déchiffrement de la BD.
- Dans l'approche « Serveur-HSM », la principale contrainte réside dans sa complexité en termes de réalisation et d'implémentation. Des logiciels embarqués robustes doivent être implémentés dans le « HW Security Module » comportant des fonctionnalités de sécurité.

6. Principe de la solution proposée

Avant de décrire le concept de notre solution, il est nécessaire de définir la notion de la granularité de chiffrement, c'est l'une des piliers de tout modèle pertinent de chiffrement de BD.

Une granularité de chiffrement d'une BD est le plus petit élément choisi d'être chiffré de manière indépendante. Chaque opération d'accès, de suppression ou de modification à cet élément indépendamment chiffré nécessite son chiffrement ou déchiffrement en entier. A titre d'exemple, si la granularité de chiffrement d'une BD est fixée au niveau d'une colonne, cela implique qu'une requête de consultation destinée à cette colonne nécessite tout son déchiffrement. La granularité de chiffrement a un impact sérieux sur la performance du SGBDR ainsi que sur la sécurité des données stockées [26]. Les principales possibilités d'une granularité de chiffrement sont : la table entière, la ligne, la colonne, la cellule et la

Tablespace [40].

La solution que nous proposons consiste à protéger les clés de chiffrement de la BD par leur chiffrement avec une clé principale Km. La génération de cette clé est faite en fonction de la granularité de chiffrement définie par le modèle de chiffrement de la BD. En effet, nous allons générer les clés Km selon les granularités de chiffrement suivantes : colonne, table, ligne et Tablespace.

Au moment de la création d'une table dans la BD, l'utilisateur définit la granularité de chiffrement dans la syntaxe de la requête. Lorsque le SGBDR accepte et valide la création de la table et ses objets, il génère automatiquement la clé Km selon un modèle que nous allons associer à chaque granularité de chiffrement. L'administrateur de la BD n'a aucune chance d'intervenir ni de savoir ce qui se passe derrière cette opération, tout est fait intrinsèquement dans le SGBDR. Avec cette solution, nous allons se prémunir efficacement contre toutes attaques de détention de la clé Km, notamment les attaques des administrateurs.

Comme il a été démontré par Bouganim et Shmueli, le chiffrement des données au niveau du SGBDR s'effectue dans le module « Security Module » du « DBMS Engine » [45], [79]. Dans la Figure 4.4, nous proposons d'intégrer nos modèles de génération de Km dans un bloc que nous nommons « Km Generator » dans le même module.

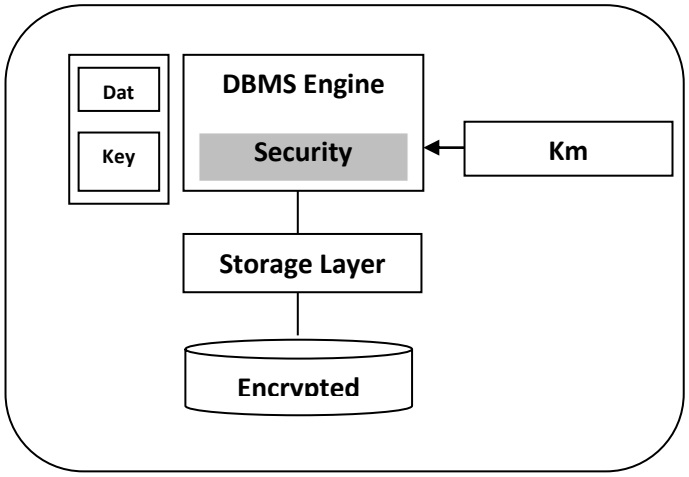

Figure 4.4: Intégration du « Km Generator » dans le SGBDR.

1.1. Description des modèles de génération de Km

La génération des clés Km suit les quatre modèles ci-dessous :

a. La clé Km (C)

La clé Km (C) est la clé principale qui va permettre de protéger les clés de chiffrement des

colonnes de la BD, elle sera générée selon le modèle défini ci-dessous :

Km (C) = H (Table _name || Column_name). **(1)**

b. La clé Km (T)

La clé Km (T) est la clé principale qui va permettre de protéger les clés de chiffrement des tables de la BD, elle sera générée selon le modèle défini ci-dessous :

Km (T) = H (Table _name || Database_name) **(2)**

c. La clé Km (P)

La clé Km (P) est la clé principale qui va permettre de protéger les clés de chiffrement des données privées des utilisateurs de la BD, elle sera générée selon le modèle défini ci-dessous :

Km (P) = H (Table _name || Column_name || USER_name) **(3)**

d. La clé Km (Ts)

La clé Km (TS) est la clé principale qui va permettre de protéger les clés de chiffrement des Tablespaces de la BD, elle sera générée selon le modèle défini ci-dessous :

Km (TS) = H (Tablespace_name|| Database_name) **(4)**

Avec :

H : une fonction de hachage choisie par le SGBDR.

Database_name : le nom de la BD.

Column_name : le nom de la colonne d'une table.

Table_name : le nom d'une table de la BD.

Tablespace_name : le nom de la Tablespace.

USER_name : le nom de l'utilisateur courant de la BD.

II. Implémentation des modèles

Dans cette section, nous allons présenter les résultats de l'implémentation des 4 modèles (1), (2), (3) et (4) de génération des clés Km. Afin d'atteindre cet objectif et dans le but de concrétiser le fonctionnement et la réponse de chaque modèle, nous allons concevoir et implémenter quatre modèles de chiffrement de BD : (A), (B), (C) et (D) possédant respectivement les quatre niveaux de granularité : colonne, table, ligne (donnée privée) et Tablespace. Le fonctionnement de chaque modèle de génération de Km est associé à l'exécution d'un modèle de chiffrement de données comme il est illustré dans la table 4.1. A la fin de cette section nous présenterons une analyse et discussion des résultats obtenus.

Modèle de génération de Km	Modèle de chiffrement de BD associé
Modèle (1): Modèle de génération des clés Km pour protéger les clés de chiffrement des colonnes	Modèle (A) : Modèle de chiffrement de BD selon un niveau de granularité colonne
Modèle (2): Modèle de génération des clés Km pour protéger les clés de chiffrement des tables	Modèle (B) : Modèle de chiffrement de BD selon le niveau de granularité table
Modèle (3): Modèle de génération des clés Km pour protéger les clés de chiffrement des lignes	Modèle (C) : Modèle de chiffrement de BD selon le niveau de granularité ligne
Modèle (4): Modèle de génération des clés Km pour protéger les clés de chiffrement des Tablespaces	Modèle (D) : Modèle de chiffrement de BD selon le niveau de granularité Tablespace

Table 4.1: Modèle de génération de Km et modèle de chiffrement de BD associé.

1. Les éléments communs de chaque implémentation

La création des éléments ci-dessous est commune à l'implémentation des modèles (1), (2), (3) et (4).

MYTABLE_ENCRYPT_OBJET : cette table de la BD contient les enregistrements des noms des objets sur lesquels nous avons défini un chiffrement, les noms d'algorithmes utilisés pour chiffrer ces objets, ainsi que le chiffrement des clés de chiffrement par la clé principale Km. Elle possède la structure suivante :

MYTABLE_ENCRYPT_OBJET (K_OBJECT_NAME, K_ENCRYPT_ALGO, K_KEY)

TEST_MANAGEMENT : cette table de la BD contient les enregistrements des noms des

objets sur lesquels nous avons défini un chiffrement, les noms des algorithmes utilisés pour chiffrer ces objets, les clés de chiffrement des objets et les clés principales (Km) de chaque objet. En fait, la table TEST_MANAGEMENT ne sera pas implémenter dans le cas d'un fonctionnement réel, son rôle consiste uniquement à illustrer les résultats de la création des clés de chiffrement et des clés (Km) afin d'illustrer ce que nos algorithmes génèrent pendant l'exécution. Elle possède la structure suivante :

TEST_MANAGEMENT (C1_OBJECT, C2_ALGO, C3_KEY, C4_MASTERKEY)

Avec :

C1_OBJECT, K_OBJECT_NAME : l'objet sur lequel nous avons défini un chiffrement.

C2_ALGO, K_ENCRYPT_ALGO : l'algorithme utilisé pour le chiffrement.

C3_KEY : la clé de chiffrement

C4_MASTERKEY : la clé principale (Km) générée.

K_KEY : le résultat de la protection des clés de chiffrement par la clé Km.

La fonction de hachage Md5 : c'est la fonction de hachage que nous allons utiliser pour créer les clés de chiffrement et les clés Km dans l'implémentation de tous les modèles.

L'algorithme AES256 : c'est l'algorithme que nous utiliserons pour chiffrer/déchiffrer les données.

Afin d'exposer les résultats de l'implémentation de nos modèles, nous allons choisir l'étude du cas suivant :

Etude de cas :

Soit la base de données BD1 d'une centrale nucléaire destinée à gérer le suivi dosimétrique des agents qui travaillent sous rayonnement ionisants. La table « Agent » stocke le cumul des différents types de doses reçues par chaque agent pendant la période de son travail au sein de la centrale. Cette table possède la structure suivante :

Agent (idf_agent, name_agt, Dose_interne_agt, Dose_superficielle_agt, Dose_profonde_agt, Catégorie_agt)

On va supposer que toutes les données de la table « Agent » sont sensibles, il s'agit en fait des valeurs de doses qui sont considérées dans le domaine nucléaire des secrets médicaux.

2. Implémentation du modèle (1)

Pour créer la table « Agent » ainsi que de définir un chiffrement sur toutes ses colonnes sensibles nous utilisons la syntaxe SQL suivante :

Create table Agent (idf_agent varchar2(100) encrypt using AES256, name_agt varchar2(100) encrypt using AES256, Dose_interne_agt varchar2(100) encrypt using AES256, Dose_superficielle_agt varchar2(100) encrypt using AES256, Dose_profonde_agt varchar2(100) encrypt using AES256, Catégorie_agt varchar2(100) encrypt using AES256) ;

L'algorithme Algo1 prend en charge l'exécution de cette instruction, il crée la table « Agent » et définit un chiffrement sur ses colonnes en utilisant l'algorithme AES256. Il génère et stocke également dans MYTABLE_ENCRYPT_OBJET et TEST_MANAGEMENT les clés de chiffrement des colonnes Kc et les clés principales Km (C) selon les modèles définis ci-dessous :

/*Le modèle de la clé de chiffrement utilisé dans le modèle (A)*/

Kc = H (Column_name)

/*Le modèle de génération de la clé Km d'une colonne*/

Km (C) = H (Table_name || Column_name)

Algo1

```
Input: Sensitive_column _query
Output:Created _sensitive_column_ query

Begin
    Loop
    Decompose (Sensitive_column _query);
    Sensitive_column_name :=  Extract (Sensitive_column _query);
    Kc:= Kc_Generator (Sensitive_column_name);
    Km (C):= Km_Generator (Sensitive_column_name, Table _name) ;
    End loop;
    Insert into TEST_MANAGEMENT values (Sensitive_column_name , Used_algo, Kc,
Km (C));
    Insert into MYTABLE_ENCRYPT_OBJET values (Sensitive_column_name , Used_algo,
Encrypt_AES256 (Km (C), Kc ));
    Execute (Sensitive_column_query);

End
```

Algorithme 4.1: Les processus gérés par Algo1.

L'exécution de l'Algo1génère les enregistrements suivants :

C1_OBJECT	C2_ALGO	C3_KEY	C4_MASTERKEY
idf_agent	AES256	46D18AC7BD06518B 5A33C650CA760D9C	36EE05BA4ACDEC7D 1C07D16FCDCC9CBF
name_agt	AES256	4BF0A8B1EB8CA12C 2912ED25E4D4BDC5	F8422E273F1957702 DE160662923C0EA
Dose_interne_agt	AES256	C8FE8472457B1A943 6EE90E6D022178F	452C55A53935CF705 6CE1C293FE8D4FC
Dose_superficielle _agt	AES256	1FD82F97BE7BA256 5D7FDD4BD6A91179	44BB0C031437DB46 B641257337C7C458
Dose_profonde_ag t	AES256	F67D1D0A822D37E3 AC03D69C94A38994	8D71448C964377D9F 0E539B3FB230133
Catégorie_agt	AES256	822E1CDA2CFA7B1E F36B90523E337682	FE30063A2D96A6D B6059CE708103BD5F

Table 4.2: Résultat des enregistrements créés dans la table TEST_MANAGEMENT dans le modèle (1).

K_OBJECT_NAME	K_ENCRYPT_ALGO	K_KEY
idf_agent	AES256	12BB4076B53A907324D42AFB9B0501006A9 3E1F347EC05536A7115E4554CD8D06662D1 EEEAC1CA8D71BC2A33EFD7810B
name_agt	AES256	90F4D05F7C5A11A5752E8AB1354BE82D797 B FE16051F78DD7018A6085649A709515637D B88472F14509DB73FC76E6B0C
Dose_interne_agt	AES256	6B40D18BF99C712F4F4034E8A11AF73FC8B 422CCC218AF595FD06EB4E4AE4BB3E8D5 D0CCA9BE57434FCD690A577D05D0
Dose_superficielle_agt	AES256	5F03CF21D034BFFBA1FEA4BD5CEA43A15 A32E8C9F4FDE591AF842995BBD30FF53CC 25849363BCF54B8FC2E28FD7FF7A7
Dose_profonde_agt	AES256	3A468DD545A12883543199A45202E4CE3AA A4D006CE4453BFC380377423C24C3FA85A1 BF68F31F6E011C062C5E19D2AE
Catégorie_agt	AES256	34FEFD552E0D63C4A99281FC1B0A8189D5 B527EDC303E9AB760B35C1A228C1F7B0EF

		DAAF26B7E3F507297D351E64FFE9

Table 4.3: Résultat des enregistrements créés dans la table MYTABLE_ENCRYPT_OBJET dans le modèle (1).

Pour tester le modèle (1), l'algorithme Algo2 prend en charge le chiffrement des données insérées par un utilisateur. Les tables 4.4 et 4.5 montrent le résultat d'insertion de quatre lignes avant et après le chiffrement.

Algo2

```
CREATE OR REPLACE TRIGGER Insert_Model_A
BEFORE INSERT ON Agent
FOR EACH ROW
DECLARE
Kc1 varchar2(100);
Kc2 varchar2(100);
Kc3 varchar2(100);
Kc4 varchar2(100);
Kc5 varchar2(100);
Kc6 varchar2(100);

BEGIN

        /*Générer Km et chercher Kc de chaque colonne à partir de la table
                MYTABLE_ENCRYPT_OBJET*/

Kc1:= Search_Encrypt_Key ('idf_agent');
Kc2:= Search_Encrypt_Key ('name_agt');
Kc3:= Search_Encrypt_Key ('Dose_interne_agt');
Kc4:= Search_Encrypt_Key ('Dose_superficielle_agt');
Kc5:= Search_Encrypt_Key ('Dose_profonde_agt');
Kc6:= Search_Encrypt_Key ('Catégorie_agt');

        /* Insertion dans la table « Agent »*/

INSERT INTO Agent VALUES (Encrypt_AES256 (Kc, :new.idf_agent), Encrypt_AES256
(Kc2, :new.name_agt, Encrypt_AES256 (Kc3, :new.Dose_interne_agt), Encrypt_AES256
(Kc4, :new.Dose_superficielle_agt), Encrypt_AES256 (Kc5, :new.Dose_profonde_agt),
Encrypt_AES256 (Kc6, :new.Catégorie_agt));END ;
```

Algorithme 4.2: Le chiffrement des données en utilisant le modèle (1).

idf_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
1000	Azzaoui	10	15	25	A

1001	Rachidi	8	11	12	A
1002	Kharmoum	16	05	14	A
1003	Sajid	14	66	65	B

Table 4.4: La table « Agent » avant chiffrement en utilisant le modèle (1).

idf_agent	**name_agt**	**Dose_interne_agt**	**Dose_superficielle_agt**	**Dose_profonde_agt**	**Catégorie_agt**
5754957F70316A9C0201002A8CBFE412	745A2F1FF2BDAD3226910989994A7287	B5063DD0A036503D02F11B63DF1B12B1	FDBD02C08C389AE7583F8211E428B2A7	D6947290516438962258 9DA79E580A60	D1D88EDED1CF67BF3AB34261052FF335
A6776D65772FBF4992D8AFC2A2B387C9	1C65506D1F8AA015B5B98765133E9782	5C6ABEA848099681319175A05C4B1AB5	BD706EE5E149EBDCE6796457E944FB81	BECBC7C7B6BD0B0E12CF1AD549E22682	D1D88EDED1CF67BF3AB34261052FF335
AFBFEA3BC5FCD28396E974016169DF5D	67BC01E3BF672D7BF568ACC3A582ECB8	B8C892229BBB1E3951290107C524B12A	125180FAB503F868EDC55F42D0BB34CB	A8A68F1B4D1D3DD0F398AFED4A54A0A5	D1D88EDED1CF67BF3AB34261052FF335
14DB0A7462832F494EAC7C8D12A09FC6	7A1FBFCD0E02BC1910055CF76279F21A	23BF97F16393EC6A84B93FF4F9EB74AA	293A72927BD5C9760450FFFF14CEC693	A2B800E15CAFD3AFA9A22FB0A4C257B8	5315BF4B7B530AA729AC1CDCB53F4C8B

Table 4.5: La table « Agent » après chiffrement en utilisant le modèle (1).

3. Implémentation du modèle (2)

Pour implémenter le modèle (2), nous allons définir un niveau de granularité de chiffrement sur toute la table « Agent » en utilisant la syntaxe SQL suivante :

Create table Agent encrypt using AES256 (idf_agent varchar2(100), name_agt varchar2(100), Dose_interne_agt varchar2(100), Dose_superficielle_agt varchar2(100), Dose_profonde_agt varchar2(100), Catégorie_agt varchar2(100));

L'algorithme Algo3 prend en charge l'exécution de cette instruction, il crée la table « Agent » et définit un chiffrement sur toutes ses données en utilisant l'algorithme AES256. Dans ce cas, les données seront chiffrées/déchiffrées par une seule clé K_T, qui sera protégée à son tour par une seule clé principale Km(T). L'Algo3 génère et stocke également dans

MYTABLE_ENCRYPT_OBJET et TEST_MANAGEMENT ces clés selon les modèles définis ci-dessous :

/*Le modèle de la clé de chiffrement utilisé dans le modèle (B)*/

K_T= H (Table_name)

/*Le modèle de génération de la clé Km d'une table*/

Km (T) = H (Table_name || Database_name)

Algo3

```
Input: Sensitive_table _query
Output: Created _sensitive_table _query

Begin
    Decompose (Sensitive_table _query) ;
    Sensitive_table_name:=  Extract (Sensitive_table _query);
    K_T:= K_T_Generator (Sensitive_table_name);
    Km(T):= Km_Generator (Sensitive_table_name, Database _name);
    Insert into TEST_MANAGEMENT values (Sensitive_table_name, Used_algo, K_T,
Km(T));
    Insert into MYTABLE_ENCRYPT_OBJET values (Sensitive_table_name ,  Used_algo,
Encrypt_AES256 (Km(T), K_T));
    Execute (Sensitive_table _query);

End;
```

Algorithme 4.3: Les processus gérés par l'Algo3.

Le résultat de l'exécution de l'Algo3 génère les enregistrements suivants :

C1_OBJECT	C2_ALGO	C3_KEY	C4_MASTERKEY
Agent	AES256	B33AED8F31349967 03DC39F9A7C95783	697C371A913425CF 202D15F143D2DAF0

Table 4.6: Résultat des enregistrements créés dans la table TEST_MANAGEMENT dans le modèle (2).

K_OBJECT_NAME	K_ENCRYPT_ALGO	K_KEY
Agent	AES256	736079082547466884631FC41910AB5770A69 62367465EC1CB9526A864367916929D05016 D02B96D9B55511854D3AB13

Table 4.7: Résultat des enregistrements créés dans la table MYTABLE_ENCRYPT_OBJET dans le modèle (2).

L'algorithme Algo4 prend en charge le chiffrement des données insérées par un utilisateur. Les tables 4.8 et 4.9 exposent le résultat d'insertion de quatre lignes dans la table « Agent » avant et après le chiffrement.

Algo4

```
CREATE OR REPLACE TRIGGER Insert_Model_2
BEFORE  INSERT ON Agent
FOR EACH ROW
DECLARE
K_T  varchar2(100);
BEGIN

        /*Générer Km et chercher K_T de la table à partir de la table
                MYTABLE_ENCRYPT_OBJET*/

K_T :=Search_Ecrypt_Key( 'Agent');

        /* Insertion dans la table «Agent»*/

INSERT INTO Agent VALUES (Encrypt_AES256 (K_T,:new.idf_agent), Encrypt_AES256
(K_T,:new.name_agt), Encrypt_AES256 (K_T,:new.Dose_interne_agt), Encrypt_AES256
(K_T,:new.Dose_superficielle_agt), Encrypt_AES256 (K_T,:new.Dose_profonde_agt),
Encrypt_AES256 (K_T,:new.Catégorie_agt));
END ;
```

Algorithme 4.4: Le chiffrement des données dans le modèle (2).

idf_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
1000	Azzaoui	25	25	25	A
1001	Rachidi	8	11	12	A
1002	Kharmoum	16	05	14	A
1003	Sajid	14	66	65	B

Table 4.8: Table « Agent » avant chiffrement dans le modèle (2).

idf_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
3F843D72B4C F6AF3BFEDD	03377847D26 334CDA665F	7012CB55124 F226FA2E530	7012CB55124 F226FA2E530	7012CB55124 F226FA2E530	91B5BD0999 38E4CE4DF7

0F8A73A46DB	815335C87F9	E0D8133F14	E0D8133F14	E0D8133F14	6529F6740B8A
2D80DCE15D394B76594AD5E18F3405BE	0B71021170A841DAE0CB4641C08076CD	0C04B7B8C2A594ED2D6CE2FCD9EE91FD	52516FBF14B5DB600CF294F47153C168	CF92FB85EF0E373795F9C4D57D66ECF1	91B5BD099938E4CE4DF76529F6740B8A
4CDDFEF4428E61343AD6C823FF1860A4	DA45B069E89AC7BE4C2B69EE232EB1A4	0F8913B9B06C772A688E9489B13A1124	5C9D1E8240D88EAD7072C8B7673AB4C1	CC2CCB3C2994CE12959B9C9B66F478A5	91B5BD099938E4CE4DF76529F6740B8A
CA80281E0ADBD105CE371803DD3F575E	EE6099C500334279227AFA5C27E199FB	CC2CCB3C2994CE12959B9C9B66F478A5	9B07E13AC87C529562187B81CFCD6B2B	EC2F5721D81D6CACE4C6EA7B9B49ED52	72621DFBF38C674D9BA24509BDA41160

Table 4.9: Table « Agent » après chiffrement dans le modèle (2).

4. Implémentation du modèle (3)

Afin d'implémenter le modèle (3), nous avons mis en place les éléments suivants :

- Créer trois nouveaux utilisateurs par l'administrateur (User1='karim') de la BD qui portent les noms suivants : User2='ahmed', User3='salah' et User3='mustapha'.
- Ajouter la colonne USER_NAME dans la table MYTABLE_ENCRYPT_OBJET.
- Définir un niveau de granularité de chiffrement sur les lignes de la table « agent » par l'administrateur de la BD en utilisant la syntaxe SQL suivante :

  Create table Agent (idf_agent varchar2(100) Primary key encrypt using AES256, name_agt varchar2(100),Dose_interne_agt varchar2(100),Dose_superficielle_agt varchar2(100), Dose_profonde_agt varchar2(100) , Catégorie_agt varchar2(100))'.
- Créer le rôle « Role_imple_3 » qui va contenir les privilèges de sélection, mise à jour et suppression sur la table « Agent ».
- Créer la procédure « Proc_imple_model_3 » dont la syntaxe représentée par l'algorithme Algo_Proc.

L'algorithme Algo5 est exécuté par l'administrateur de la BD, il prend en charge la création de la table « Agent » et la définition d'un chiffrement sur ses lignes. Cet algorithme permet de générer la clé privée Kp de l'administrateur et sa clé principale Km(P) selon les modèles

définis ci-dessous tout en les stockant dans MYTABLE_ENCRYPT_OBJET et TEST_MANAGEMENT.

/*Le modèle de la clé de chiffrement définit dans le modèle (C)*/

Kp= H (User_name)

/*Le modèle de génération de la clé Km d'une ligne*/

Km (P) = H (Table _name || Column_name || User_name)

De même, lorsque l'administrateur crée un nouvel utilisateur, il lui accorde le rôle «Role_imple_3» et les rôles contenant les privilèges objets et systèmes nécessaires pour se connecter à la BD à savoir, les rôles Connect et Resource, etc... Il exécute ensuite l'algorithme AlgoY qui permet de créer la clé privée Kp du nouvel utilisateur et sa clé principale Km(P) dans MYTABLE_ENCRYPT_OBJET et TEST_MANAGEMENT selon toujours les mêmes modèles définis ci-dessus.

Chaque ligne de la table « Agent » est considérée sensible et privée à son propriétaire, c'est à dire à l'utilisateur qui est identifié par sa valeur unique de la clé primaire «idf_agent».

Algo_Proc

```
Input: User_name
Output: affect «Role_imple_3» to User_name

Create Procedure  Proc_imple_model_3 (User_name)
Begin

Create  Role_imple_3;
Grant select, update, delete to Role_imple_3;
Grant  Role_imple_3  to  User_name;

End;
```

Algorithme 4.5: Affectation des rôles à chaque utilisateur.

AlgoY

```
Input: Sensitive_ligne_query
Output: Created _sensitive_ligne_table
Begin

    Proc_imple_model_3 ('User_name');
    Decompose (Sensitive_ligne_query) ;
    Sensitive_column_name :=  Extract_1 (Sensitive_ligne_query) ;
    Sensitive_table_name :=  Extract_2 (Sensitive_ligne_query) ;
    Kp := Kp_Generator (User_name);
```

```
    Km(P) := Km_Generator (Sensitive_table_name, Sensitive_column_name,Current_user);
    Insert into TEST_MANAGEMENT values (Sensitive_column_name, Used_algo, Kp,
Km(P));
    Insert into MYTABLE_ENCRYPT_OBJET values (Sensitive_column_name ,
User_name, Used_algo,
    Encrypt_AES (Km(P), Kp));

End;
```

Algorithme 4.6: Création des clés Kp et Km(P) du nouvel utilisateur.

Algo5

```
Input: Sensitive_ligne_query
Output: Created _sensitive_ligne_table
Begin

    Decompose (Sensitive_ligne_query);
    Sensitive_column_name :=  Extract_1 (Sensitive_ligne_query) ;
    Sensitive_table_name:=  Extract_2 (Sensitive_ligne_query) ;
    Kp := Kp_Generator (User_name);
    Km(P) := Km_Generator (Sensitive_table_name, Sensitive_column_name,Current_user);
    Insert into TEST_MANAGEMENT values (Sensitive_column_name, Used_algo, Kp,
Km(P));
    Insert into MYTABLE_ENCRYPT_OBJET values (Sensitive_column_name ,
User_name, Used_algo,
    Encrypt_AES (Km(P), Kp));
    Execute (Sensitive_ligne_query) ;
End;
```

Algorithme 4.7: Les processus gérés par l'Algo5.

Après l'exécution de l'Algo5 par chaque utilisateur, les résultats des enregistrements générés sont illustrés dans les tables ci-dessous :

C1_OBJECT	C2_ALGO	C3_KEY	C4_MASTERKEY
idf_agent	AES256	2167A6AC80340B69 F3B05B800417D6C7	4A819599C157B03A 055FE96728072355
idf_agent	AES256	9193CE3B31332B03F 7D8AF056C692B84	DBEE48FF22067C8A 53A44E5DF04AFEF1
idf_agent	AES256	2A07E3FF3DF21B22 6D0CD044D4A7CC83	FD18434A7DE7224FE 96497A6FACAC77B
idf_agent	AES256	8FC920395E4DF502 94B7347EF99CCA4C	87D5ACC07125F1DC C457C20636DE8E9B

Table 4.10: Résultat des enregistrements créés dans la table TEST_MANAGEMENT dans le modèle (3).

K_OBJECT_NAME	USER_NAME	K_ENCRYPT_ALGO	K_KEY
idf_agent	karim	AES256	8ACA11E9C18F228EEFFB44BE379690DA7B23EB96C9C300350C67A28AFF90331E9D2664183021D608C51783DDC490FD22
idf_agent	ahmed	AES256	587735C1B3D72462CA8AAE29CF1FD6767DB9AA6BF9EA97817BB99247D838C6892B9842E64446405B8AAA5391B097421A
idf_agent	salah	AES256	F3103D53749A1D4B8CDC6A8E17301E8C7DAA6AD3ADA63AE59EEB2A1A96F972388F8E0E0A972BDAD5479E95634187D4FF
idf_agent	mustapha	AES256	8A43E92ED4D1043A83070DE0581EB7957F157B2D248C7CFE92C2544488513FB61A125E543A72A1331131266EC43E59B6

Table 4.11: Résultat des enregistrements créés dans la table MYTABLE_ENCRYPT_OBJET dans le modèle (3).

L'algorithme Algo6 prend en charge le chiffrement des données insérées par l'utilisateur qui exécute cet algorithme (dans Algo6 c'est l'utilisateur User2='ahmed'). La fonction Search_Ecrypt_Key retourne le Kp de chaque utilisateur qui l'a appelée. Les tables 4.12 et 4.13 montrent le résultat avant et après le chiffrement de l'insertion de quatre enregistrements dans la table « Agent » par l'administrateur de la BD et les trois utilisateurs que nous avons créés.

Algo6

```
CREATE OR REPLACE TRIGGER Insert_Model_3
AFTER insert ON karim.agent
FOR EACH ROW
DECLARE
Kp VARCHAR2(100);
BEGIN
```

```
Kp:= ahmed.Search_Ecrypt_Key ('idf_agent');


INSERT INTO karim.agent VALUES (Encrypt_AES256 (Kp,:new.idf_agent)
,Encrypt_AES256 (Kp,:new.name_agt) ,Encrypt_AES256
(Kp,:new.Dose_interne_agt,Encrypt_AES256 (Kp,:new.Dose_superficielle_agt)
,Encrypt_AES256 (Kp,:new.Dose_profonde_agt) ,Encrypt_AES256
(Kp,:new.Catégorie_agt);

END
```

Algorithme 4.8: Le chiffrement des données dans le modèle (3).

Id_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
1000	Karim	10	25	25	A
1001	ahmed	8	11	12	A
1002	salah	16	14	14	A
1003	mustapha	14	14	6	B

Table 4.12: Table « Agent » avant chiffrement dans le modèle (3).

idf_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
6D3DD60FFEE278978EC5D0013F9199D4	40B955DDD4DE1E0635F6397003DB481E	FC6DCE533F00C887D35F4F7F08C97A4E	04FF80302CBAB3EDA0052DC6248613B0	04FF80302CBAB3EDA0052DC6248613B0	30BBF2CDBD69F6880E6698455B2C83F2
67D5F8AF5CBF2B6773A7D57D9597D9AB	D85B62EC214A03E24C1CAC3365E9EBAF	78D2A9833EC11CC98F65B2E4C82549AA	47206093A269ECEF87A63F0C871F393D	530139F1A01307CBDAD0F92ABD43D582	6CB16C60D592096B9F1CF2C966338650
673B3408743F004B84678B4EEE624204	557EC47D2B1056172F6CC40E7131DBC	1F62B47C4C2703D5F1E95C016B2ED7E3	45A1241FD46181F387E04FCE78D9A9A1	45A1241FD46181F387E04FCE78D9A9A1	47393F24E1DA78553E62FDD0865ED1B0
45C68891CDBA57B6D46B8FF582466BF9	F602DF54DCD5E92E0A2B5F3C9079F731	E7F728C33C99DB1E8D751E4B3D857B29	E7F728C33C99DB1E8D751E4B3D857B29	E94F5110E48DB5BDEB4F31EF499A9E8C	541A5608944732791 70DB2D65A171527

Table 4.13: Table « Agent » après chiffrement dans le modèle (3).

5. Implémentation du modèle (4)

L'implémentation du modèle (4) nécessite une définition d'un chiffrement au niveau du Tablespace. Nous allons créer la table « Agent » au sein d'une Tablespace chiffrée que nous nommons « Tab_space_mod4 » en utilisant la syntaxe suivante :

Create table Agent (idf_agent varchar2(100),name_agt varchar2(100),Dose_interne_agt varchar2(100), Dose_superficielle_agt varchar2(100), Dose_profonde_agt varchar2(100), Catégorie_agt varchar2(100)) Tablespace Tab_space_mod4;

L'algorithme Algo7 assure l'exécution de cette instruction, il crée et définit un chiffrement sur la table « Agent ». Dans ce cas les données de cette table et de toutes les tables qui appartiennent à la Tablespace « Tab_space_mod4 » seront chiffrées/déchiffrées par une seule clé K_{TS} qui sera protégée par une seule clé principale Km (TS). L'algorithme Algo7 génère et stocke dans MYTABLE_ENCRYPT_OBJET et TEST_MANAGEMENT les clés selon les modèles définis ci-dessous :

/*Le modèle de la clé de chiffrement utilisé dans le modèle (D)*/

K_{TS}= H (Tablespace_name)

/*Le modèle de génération de la clé Km d'une Tablespace*/

Km (TS) = H (Tablespace_name || Database_name)

Algo7

Input: Sensitive_tablespace_query
Output: Created_sensitive_tablespace_query

Begin

Decompose (Sensitive_tablespace_query);
Sensitive_tablespace_name := Extract (Sensitive_tablespace_query);
K_{TS} := K_{TS}_Generator (Sensitive_tablespace_name);
Km(TS) := Km_Generator (Sensitive_tablespace_name, Database_name);
Insert into TEST_MANAGEMENT values (Sensitive_tablespace_name, Used_algo, K_{TS}, Km(TS));
Insert into MYTABLE_ENCRYPT_OBJET values (Sensitive_tablespace_name, Used_algo, Encrypt_AES256 (Km(TS), K_{TS}));
Execute (Sensitive_tablespace_query);

```
End;
```

Algorithme 4.9: Les processus gérés par l'Algo7.

Le résultat de l'exécution de l'Algo7 génère les enregistrements suivants :

C1_OBJECT	C2_ALGO	C3_KEY	C4_MASTERKEY
Tab_space_mod4	AES256	4E37411448206D4B4 E2EFB8AD923A7D3	ADB33E49D789A789F EF5AC0DC1CE2F2C

Table 4.14: Résultat des enregistrements créés dans la table TEST_MANAGEMENT dans le modèle (4).

K_OBJECT_NAME	K_ENCRYPT_ALGO	K_KEY
Tab_space_mod4	AES256	28E697047F3A6EA6D204A4DF0F00B9C7977 44C1B3D2019F37A8286DF265BC49516B56F 7B42D8F5BE4C754B06CBD086E7

Table 4.15: Résultat des enregistrements créés dans la table MYTABLE_ENCRYPT_OBJET dans le modèle (4).

L'algorithme Algo8 prend en charge le chiffrement des données insérées par un utilisateur. Les tables 4.16 et 4.17 représentent le résultat d'insertion de quatre lignes dans la table « Agent » avant et après le chiffrement.

Algo8

```
CREATE OR REPLACE TRIGGER Insert_Model_4
BEFORE INSERT ON Agent
FOR EACH ROW
DECLARE
K_TS varchar2(100) ;

BEGIN

        /*Générer Km(TS) et chercher K_TS de la Tablespace à partir de la table
                MYTABLE_ENCRYPT_OBJET*/

K_TS :=Search_Ecrypt_Key( 'Agent');

        /* Insertion dans la table « Agent » */

INSERT INTO Agent VALUES (Encrypt_AES256 (K_TS : new.idf_agent), Encrypt_AES256
(K_TS,:new.name_agt), Encrypt_AES256(K_TS : new.Dose_interne_agt, Encrypt_AES256
```

(K_{TS},:new.Dose_superficielle_agt), Encrypt_AES256(K_{TS},:new.Dose_profonde_agt), Encrypt_AES256 (K_{TS},:new.Catégorie_agt));
END

Algorithme 4.10: Le chiffrement des données dans le modèle (4).

idf_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
1000	Karim	10	25	25	A
1001	ahmed	8	11	12	A
1002	salah	16	14	14	A
1003	mustapha	14	14	6	B

Table 4.16: Table « Agent » avant chiffrement dans le modèle (4).

idf_agent	name_agt	Dose_interne_agt	Dose_superficielle_agt	Dose_profonde_agt	Catégorie_agt
DEE4E93A0A13DF2CCB87BBCD247B852A	4A6EA102229CADA5A673A0F1471B6B16	2CF6FD73DED92994478DA5D2031AFCAD	3AE4D01BD43A9DA51FCCB0C5CE36F764	3AE4D01BD43A9DA51FCCB0C5CE36F764	5DB81329E08C0FF0027FEBD51156B462
B163E815CC57AB0A8BE39549FBEDAFEF	0F58AA6FCA1D8D2A6F58614BF8608293	1BCAD72B8D908AA9DDBD09A906863D95	AD7A0C2DCFEAA86904361021ABB03BC2	42FCEA2BC5C062BDE025C549E63F0407	5DB81329E08C0FF0027FEBD51156B462
B45D067698BEFB186B3A950194BD2C6A	F47710B2787E9AAC08C1FC7B065B3C68	717DE91E7380DFA01A0ADCFF20CF46A2	0385BF7284A8EDE6114274F1CBF5B6BE	0385BF7284A8EDE6114274F1CBF5B6BE	5DB81329E08C0FF0027FEBD51156B462
41CBD06CD8535F70E1E0B3739441E1EB	EBDD71A1079E52E129EC2E98417951AF	0385BF7284A8EDE6114274F1CBF5B6BE	0385BF7284A8EDE6114274F1CBF5B6BE	54558E25878452F95306E8AA73A39B04	D92451840AD0670F137E14A0FE376F61

Table 4.17: Table « Agent » après chiffrement dans le modèle (4).

6. Résultats et discussions

D'après ce que nous avons obtenu dans les résultats de l'implémentation des quatre modèles, nous allons présenter notre analyse de résultats sur la base des éléments ci-dessous :

- L'approche que nous proposons élimine l'utilisation des modules de sécurisation externes tels que le Wallet, le HSM et le serveur de sécurité. Cela permettra d'optimiser des couts supplémentaires pour protéger les clés soit en termes d'acquisition de matériel (cas du HSM), ou en ressource humaine (cas du « Serveur de sécurité »).
- Par rapport à l'approche « Wallet » qui est considérée la plus proche à la solution que nous proposons en termes de génération de Km à l'intérieur du SGBDR, la création de la clé Km ne nécessite plus de la sécuriser dans un endroit sûr tels que les systèmes de sauvegarde, à chaque fois que la clé Km est nouvellement créée.
- L'approche proposée ne nécessite aucun contrôle ou gestion de la protection des clés de chiffrement par un administrateur de la BD ou de sécurité.
- Il n'y a pas de lieu de stockage défini pour Km, sa génération se fait de manière automatique à chaque définition du chiffrement sur une table sensible, précisément lors de la création. La détention de la clé Km est une opération quasiment impossible, les attaquants qui arrivent à posséder des privilèges sur la BD à savoir, les attaquants internes, externes et les administrateurs n'ont aucune chance de réussir.
- Notre concept de protection des clés renforce davantage la sécurité des clés et des données. Par rapport à la solution Oracle TDE Column Encryption qui utilise une seule clé principale pour protéger toutes les clés des colonnes, le modèle (1) génère plusieurs Km pour protéger les clés de colonnes, le nombre de Km généré est égal au nombre de colonnes sensibles. Par exemple si une BD contient 20 colonnes sensibles à chiffrer, nous aurons 20 clés pour chiffrer les données et 20 Km pour protéger ces clés. Le principe est le même si nous comparons Oracle TDE Tablespace Encryption, MySQL TDE et SQL Server TDE avec le modèle (4), ces solutions utilisent une seule clé principale pour protéger les clés de toutes les Tablespaces, par contre le modèle (4) génère une Km pour chaque Tablespace. Le même raisonnement est valable si nous voulons comparer notre solution avec les concepts de protection des clés proposés dans les modèles de chiffrement de BD que nous avons étudié dans la littérature et dont on a analysé les plus pertinents parmi eux dans le chapitre 2.
- L'approche proposée est globale en termes de solution de protection de clés de chiffrement, elle peut fonctionner avec n'importe quels modèles de chiffrement de BD, bien évidemment ceux qui implémentent une granularité de chiffrement au niveau

des colonnes, Tablespaces, lignes et tables. Elle est très bien adaptée aux SGBDR à licence libre.

- Les résultats présentés dans les tables 4.4, 4.5, 4.8, 4.9, 4.12, 4.13, 4.16, 4.17 montrent que notre solution fonctionne parfaitement soit lors du chiffrement ou de déchiffrement des données. Pour tous les modèles, un chiffrement nécessite la génération de Km, le déchiffrement d'une valeur de la colonne K_KEY pour obtenir la clé réelle et finalement le chiffrement des données. Le déchiffrement des données s'effectue en suivant les mêmes opérations.

Conclusion :

A l'égard des mécanismes conventionnels mis en place pour sécuriser une BD (protection réseau, authentification et contrôle d'accès), le chiffrement des données côté SGBDR est un moyen fort qui renforce la défense en profondeur des données. Il est fortement lié à la protection des clés de chiffrement dont elles dépendent de 2 facteurs principaux : le lieu de stockage de la clé et les personnes qui ont le droit d'y accéder. Notre contribution dans ce chapitre porte sur la proposition d'une approche qui sécurise les clés de chiffrement au sein des SGBDR. Nous proposons également son implémentation selon des modèles spécifiques corrélés au niveau de granularité du chiffrement de la BD.

CHAPITRE V : UNE NOUVELLE APPROCHE POUR PROTEGER LES CLÉS DE CHIFFREMENT CÔTÉ CLIENT CONTRE LES ATTAQUES INTERNES

Introduction :

Le principe du chiffrement de la BD côté client consiste à déléguer le chiffrement et le déchiffrement aux applications qui y sont connectées. Le serveur de BD traite et manipule que des données chiffrées et les clés de chiffrement n'existent plus sur le serveur, ce qui signifie qu'il est à l'abri de toutes menaces d'attaques susceptibles d'être menées par l'administrateur. Cependant, les clés doivent forcément être stockées soit sur les applications ou dans un endroit où les applications sont gérées (serveur d'applications par exemple). Cette approche présente une limite puisque le client (ou l'administrateur du serveur d'applications) ne sont pas toujours des sites de confiance, ils peuvent attaquer à tout moment les clés de chiffrement [40].

Nous avons souligné dans le premier chapitre que les BD peuvent confronter plusieurs types de menaces internes, celles qui proviennent des utilisateurs légitimes du système. Ces menaces peuvent être menées en collaboration avec un administrateur malveillant du système, un autre utilisateur légitime, ou avec une personne malveillante qui se situe à l'extérieur du système d'informations [21], [36].

Les attaques par abus abusif ou légitime de privilèges, par élévation de privilèges sont des exemples de modèles d'attaques internes qui peuvent compromettre les clés de chiffrement. Un utilisateur légitime d'une BD peut exploiter ses propres privilèges qui dépassent ses fonctions métiers pour attaquer les clés directement depuis sa machine. Il peut également bénéficier des erreurs de configuration, des vulnérabilités non corrigées pour accéder à l'emplacement du stockage des clés. Un administrateur d'application non fiable peut révéler les clés de chiffrement à un attaquant externe, qui à son tour cherchera le chemin réseau convenable pour attaquer la BD en dehors du périmètre du système d'informations [42]. Les utilisateurs privilégiés qui bénéficient de la confiance constituent finalement la menace la plus haute pour la sécurité des clés de chiffrement [80].

Dans la littérature, plusieurs solutions ont été proposées par les chercheurs pour protéger les clés de chiffrement au niveau applicatif, parmi ces travaux nous citons la contribution de Ding et Klein [84]. Les auteurs de ce travail ont proposé un nouveau modèle de chiffrement de BD qui assure la confidentialité des données sensibles. Leur modèle adopte une méthodologie qui utilise une chaîne de clés pour protéger les clés de chiffrement. Elovici et al., ont présenté leur solution de chiffrement de BD en suggérant une protection des clés soit avec un HSM ou à l'aide d'un serveur de sécurité dédié [79]. Bouganim et al., ont suggéré la solution "Client HSM", son principe consiste à intégrer le module "HW Security Module" au niveau de

chaque utilisateur de BD. Ce module stocke et protège les clés de chiffrement contre l'exposition et les attaques internes [45].

Dans ce chapitre, nous allons proposer une nouvelle solution de protection des clés de chiffrement au niveau des applications. Le principe général de la solution consiste à convertir les requêtes des utilisateurs vers des nouvelles requêtes appelant à distance des fonctions stockées dans le serveur de BD. Le rôle des fonctions consiste à transformer les clés détenues dans les requêtes des clients vers les clés réelles de chiffrement/déchiffrement de la BD.

I. Chiffrement côté client

Dans cette section nous allons mettre l'accent sur la place que prennent les attaques internes dans le contexte des systèmes d'informations sensibles, plus particulièrement celles qui compromettent la sécurité des BD. Afin d'exposer la problématique que nous traitons dans ce chapitre, nous allons introduire l'approche du chiffrement côté client et les limites qu'elle engendre vis-à-vis de la sécurité des clés de chiffrement.

1. Les attaques internes

Dans la vie réelle, de véritables enquêtes criminelles et policières ont conduit à des crimes commis par des personnes connues de la victime, des personnes de très proche. Similairement, le même principe existe dans le monde de la cybercriminalité. Lorsque nous parlons d'attaque informatique, nous avons toujours à l'esprit qu'il s'agit d'acte malveillant piloté par des gens situés à l'extérieur de l'entreprise, or il est désormais de plus en plus fréquent que les attaques proviennent de l'intérieur de l'entreprise. Ces actes peuvent être pilotés par le personnel de l'entreprise, les sous-traitants, les fournisseurs, les collaborateurs externes, des gens à qui l'entreprise fait confiance [36].

Les attaques internes sont en forte progression ces dernières années, elles sont difficilement détectables par rapport à celles qui proviennent de l'extérieur du système d'informations. Malheureusement, l'impact et la gravité de ces attaques sont encore sous-estimés par les responsables de la sécurité [80]. Une étude récente qui s'intitule «Insider Threat Report 2019» de BITGLASS a mis en évidence des résultats captivants sur les attaques d'origines internes qui ont frappées les entreprises en 2019. Plus de 59% d'entreprises ont subi au moins une attaque de ce genre [72]. La même étude a souligné que le principal obstacle qui a rendu ces attaques difficilement détectables est que les attaquants possèdent des comptes et des identifiants légitimes, il est donc difficile de détecter une activité légale par un système

d'intrusion qui se comporte face à eux de manière très normale et naïve.

2. Les attaques internes sur les Bases de Données

Les BD sont souvent compromises par des attaques internes, la première source de menace provient des employés détenteurs de privilèges spéciaux sur la BD. Pour des raisons de mécontentement ou de vengeance, ces gens déploient leurs accès privilégiés intentionnellement pour commettre des violations de données. Une autre catégorie d'employés attaque les BD en pratiquant de l'espionnage pour leur compte en vendant des informations sensibles dans le marché noir. Il y en a aussi ceux qui sont infiltrés par des tierces parties pour exercer de l'espionnage industriel, économique ou politique. Nous avons déjà signalé dans le premier chapitre la dangerosité des attaques de type abus de privilèges légitimes et excessifs qui font partis des menaces internes les plus redoutables [80].

L'attaque interne peut venir de manière involontaire, parfois l'erreur humaine conduit à des situations de violation de données. A titre d'exemple, une infermière d'un hôpital peut copier des informations d'une table de BD qui contient des secrets médicaux des patients dans un courrier électronique à des fins de dépannage et inclure accidentellement des adresses électroniques externes dans la liste des destinataires.

3. Le principe du chiffrement des Bases de Données côté client

Le chiffrement d'une BD côté client signifie que le processus de chiffrement /déchiffrement des données se fait localement sur le client avant que les données ne soient transmises au serveur. Ce principe est similaire à une externalisation de la BD dans un service de stockage en Cloud [73], [74]. La gestion des clés dans ce concept est assurée par le client qui les détiennent toutes. Donc, il n'y a pas de transmission de données claires ni de clés au serveur de BD.

Le chiffrement au niveau du client est une stratégie de sécurité qui présente l'avantage que le serveur résiste aux attaques internes, celles des administrateurs. Cependant, elle comporte plusieurs limites à savoir, il faut déporter une grande partie du traitement au client, de même pour la gestion des droits d'accès. Les données et les clés sont forcément stockées en claires sur le client, cela pose donc un problème de confiance vis-à-vis des utilisateurs de la BD, voir même de l'administrateur des applications, puisqu'ils ne peuvent pas être toujours des sites de confiance.

4. Attaque interne sur un chiffrement côté client

Comme nous l'avons abordé dans le paragraphe précédent, lors d'une approche de chiffrement côté client, les exploitants légitimes de la BD et l'administrateur d'applications constituent une menace sur les clés de chiffrement. Les valeurs des clés et leurs emplacements sont la principale vulnérabilité de ce concept. Dans une architecture à 2 niveaux par exemple, les clés sont en claires sur les clients, soit dans les codes ou dans les fichiers de paramétrage de l'application, cela signifie qu'elles sont totalement soumises à la disposition des utilisateurs. De même dans une architecture à 3 niveaux, souvent les clés sont stockées en claires sur le serveur d'applications, elles sont aussi soumises à la disposition totale de l'administrateur d'application. Un simple scenario d'attaque dans l'architecture à 3 niveaux est une attaque par conspiration menée par un utilisateur est l'administrateur de l'application. Un administrateur malveillant peut révéler toutes les clés de chiffrement à un utilisateur quelconque dans le simple but d'attaquer la BD et la déchiffrer complètement (Figure 5.1).

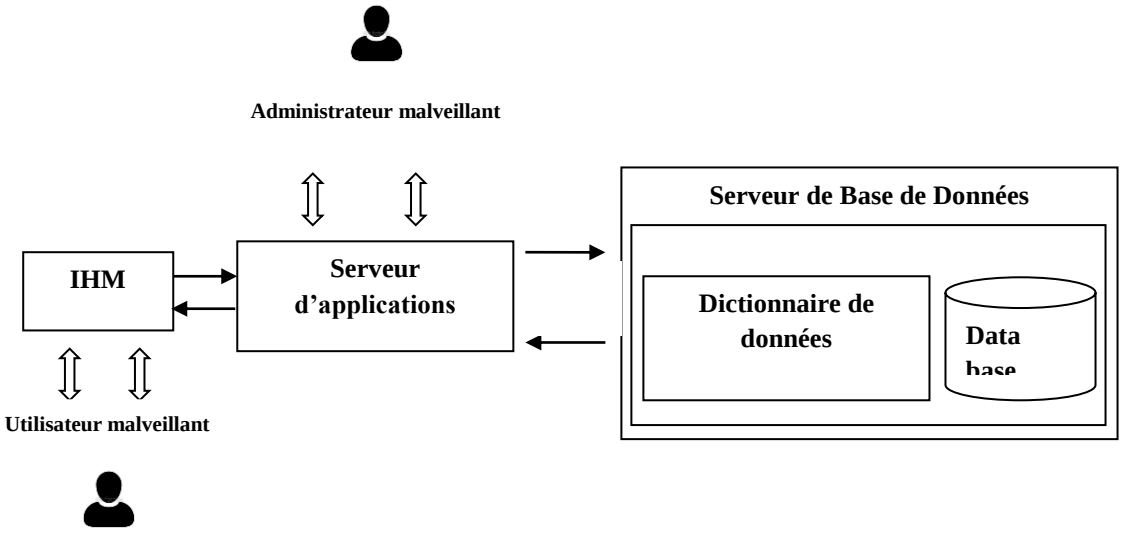

Figure 5.1: Scenario d'attaque interne sur la BD.

II. Description et mise en œuvre de la solution proposée

Dans cette section, nous allons décrire la solution que nous proposons, nous procéderons également à son implémentation et nous discuterons les résultats obtenus.

1. Description générale de la solution

Le principe général de notre solution repose tout simplement sur l'externalisation de la protection des clés de chiffrement du client vers le serveur de BD. En fait, nous transformons

la requête qui s'exécute au niveau de chaque utilisateur et qui utilise une clé de chiffrement en une nouvelle requête qui convertit cette clé à la clé réelle de chiffrement/déchiffrement de la BD. A ce titre, nous allons choisir de proposer et d'implémenter deux types de solutions, la première concerne la protection des clés de chiffrement lors d'une insertion de données dans une seule colonne sensible, tandis que la deuxième concerne la protection des clés lors d'une insertion de données dans plusieurs colonnes sensibles.

Considérons une table Tab1 ayant la structure (A) et supposons que l'attribut "salaire" est une colonne sensible à chiffrer lors d'une insertion et à déchiffrer lors de la consultation de la table.

Tab1 (code, prénom, nom, salaire) (A)

1.1. Solution1 : insertion dans une seule colonne sensible

Soit un utilisateur qui exécute dans un environnement client-serveur une requête ayant la forme suivante :

Insert into Tab1 values ('0001', 'elbouchti', 'karim', AES192 ('Mycolor012345678has @ ml @ p', 2000); (B)

L'utilisateur ici insère dans la table Tab1 une ligne de données. La colonne "salaire" sera chiffrée avec l'algorithme AES192 en utilisant la clé K = "Mycolor012345678has @ ml @ p". Avant l'envoi de cette requête elle sera transformée à la requête (C) ayant la syntaxe ci-dessous :

Insert into Tab1 values ('0001', 'elbouchti', 'karim', AES192 (Func('Mycolor012345678has @ ml @ p'), 2000)); (C)

Func() est une fonction appelée à chaque fois que l'utilisateur exécute une requête d'insertion dans une ligne de table possédant une seule colonne sensible. La description de la fonction Func() n'est pas définie dans les fichiers de l'application de l'utilisateur ou dans le serveur d'applications. Son principal rôle est de transformer la valeur de la clé définie dans la requête qui ressort de l'application en une nouvelle clé dont la valeur est la clé réelle de chiffrement /déchiffrement de la colonne (Figure 5.2).

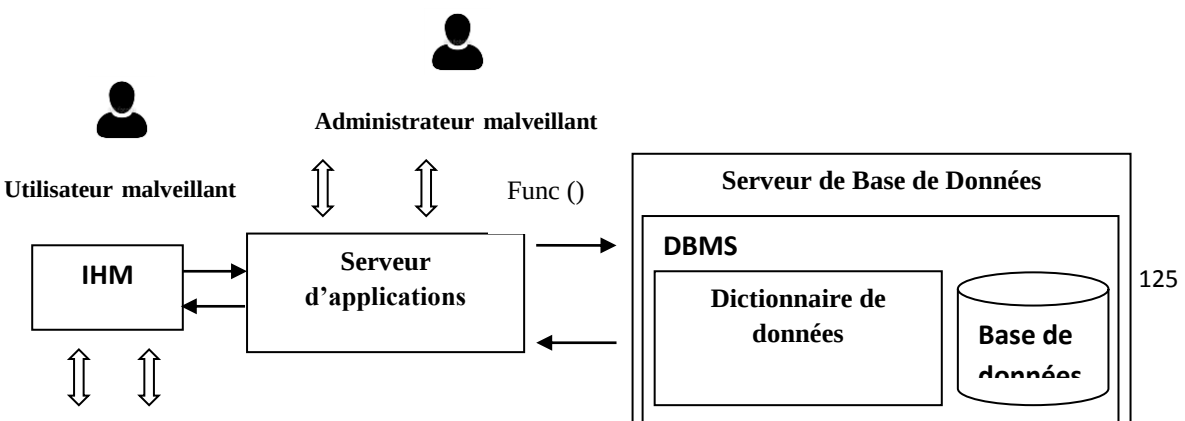

Figure 5.2: Exemple d'appel de la fonction Func() dans une architecture à 3 niveaux.

Nous proposons d'implémenter la fonction Func() dans le serveur de la BD. Elle fera partie parmi les objets crées dans la BD. Son modèle suit l'algorithme suivant :

Func (X)

{ K = E (X, H (Nom_table));

Retour (K); }

Avec :

- E : un algorithme de chiffrement symétrique.
- H : une fonction de hachage.
- X : la clé définie dans les fichiers d'application du poste client ou dans les fichiers d'application du serveur d'applications.
- K : la clé réelle pour chiffrer/déchiffrer les données de la colonne.
- Nom_table : ce nom correspond au nom de la table définie dans l'instruction d'insertion.

La conversion de la requête (B) vers la requête (C) s'effectue par un algorithme de transformation (Algorithme 5.1 et Algorithme 5.2 dans les deux implémentations de notre solution). Cet algorithme doit figurer obligatoirement en mode exécutable parmi les fichiers de l'application comme le montre la Figure 5.3.

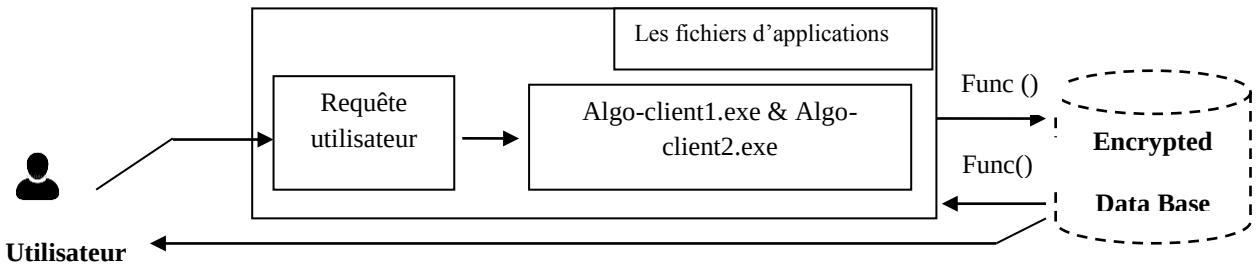

Figure 5.3: Processus de conversion de la requête.

1.2. Implémentation de la solution 1

L'implémentation de notre solution a été réalisée sur une application que nous avons développée sous le logiciel SQLNAVIGATOR 6.2.0.1500. Elle est connectée à distance à la BD nommée "DB_karim" sous ORACLE10G. nous avons choisi dans la fonction Func() l'AES256 comme algorithme de chiffrement et MD5 une fonction de hachage.

Lorsque l'utilisateur exécute une requête d'insertion, cette requête est transformée par l'algorithme « Algo-client1 » en une nouvelle requête qui appelle la fonction Func() du serveur de BD. Le corps de " Algo-client1" est défini comme suit :

Algo-client1

```
Input: executed_query
Output: new_query
Begin
    Decompose (executed_query)
    Func(key)
    Generate (new_query)
    Execute (new_query)
End
```

Algorithme 5.1: Transformation de la requête exécutée par l'utilisateur dans le cas d'une insertion de données dans une seule colonne.

Le test de « Algo-client1 » a été réalisé sur la table « Agent » ayant la structure suivante :

Agent (code, first-name, last_name, dose) (D)

Nous avons défini la colonne dose comme étant une colonne sensible, elle sera protégée par un chiffrement en utilisant l'algorithme AES256 et la clé k = '@mysonmyson@123'. Les deux tableaux (Table 5.1 et Table 5.2) exposent les résultats obtenus avant et après le chiffrement.

code	first-name	last_name	dose
1000	Karim	El bouchti	10
1001	hamid	Afane	10
1002	adil	Faris	15
1003	amina	Lemnawar	04
1004	tayebi	El bouchti	06
1005	amina	Aghbal	09

Table 5.1: La table « Agent » avant chiffrement.

code	first-name	last_name	dose
1000	Karim	El bouchti	3C0354B2295D6898C 5BB2731CB6ADCB9

1001	hamid	Afane	3C0354B2295D6898C 5BB2731CB6ADCB9
1002	adil	Faris	ADE96B60B117A764 F5870665DF4F3D7D
1003	amina	Lemnawar	88659CA5130C2813 6F6AD7B37F74E531
1004	tayebi	Elbouchti	29152E11B0AEA7C D69F4327E6AD4730D
1005	amina	Aghbal	C1EDB798F41A5C19F 18B2A3E35D6ED92

Table 5.2: La table « Agent » après chiffrement.

1.3. Solution 2 : Insertion dans plusieurs colonnes sensibles

Le principe de la protection des clés de chiffrement dans une requête d'insertion de données dans plusieurs colonnes sensibles est similaire à celle d'une insertion dans une seule colonne. L'objectif étant de protéger la clé de chaque colonne d'une manière distincte des autres clés de colonnes.

Supposons qu'un utilisateur exécute dans un environnement client-serveur une requête ayant la forme suivante :

Insert into Tab1 values ('0001',AES192('Mycolor442266775hrt@HH@T','elbouchti'), AES192('Mycolormybeautyy@nn @x','karim'), AES192('Mycolor012345678has@ml@p', 2000); (E)

Avant l'envoi de cette requête, elle sera transformée à la requête (F) qui possède la syntaxe ci-dessous :

Insert into Tab1 values ('0001', AES192(Func_Mc('Mycolor442266775hrt@ HH @ T','elbouchti')), AES192(Func_Mc('Mycolormybeautyy@ nn @ x','karim')), AES192(Func_Mc('Mycolor012345678has @ ml @ p')), 2000); (F)

Func_Mc() est une fonction qui est définie et stockée dans le serveur de BD, elle substitue la clé définie dans la requête de l'utilisateur qui ressorte de l'application par la clé réelle de chiffrement/déchiffrement des données.

Nous suggérons implémenter Func_Mc() dans le serveur de BD parmi les objets de la BD, son modèle suit l'algorithme suivant :

Func_Mc (X)

{ K = E (X, H ((Nom_table) || (Nom_col)) || (Nom_DB) || Sum (id_col, Id_tab)));

Retour (K) ; }

Avec :

- E : un algorithme de chiffrement symétrique.
- H : une fonction de hachage.
- X : la clé définie dans l'application
- K : la clé réelle pour chiffrer/déchiffrer les données.
- Nom_col : le nom de la colonne qui est définie dans l'instruction d'insertion.
- Nom_DB : le nom de la base de données.
- Nom_table : ce nom correspond au nom de la table définie dans l'instruction d'insertion.
- Sum (id_col, Id_tab) : la sommation des identifiants de la colonne et de la table.

1.4. Implémentation de la solution 2

Les mêmes outils que nous avons utilisés pour implémenter la solution 1 ont été déployés pour implémenter la solution 2. Nous avons choisi dans la fonction Func_Mc() l'AES256 comme algorithme de chiffrement et MD5 une fonction de hachage.

L'algorithme « Algo-client2 » présenté ci-dessous permet de transformer la requête de l'utilisateur à la nouvelle requête qui va s'exécuter sur le serveur.

Algo-client2

```
Input: executed_query
Output: new_query
Begin
    Decompose (executed_query)
    Func_Mc(key)
    Generate (new_query)
    Execute (new_query)
End
```

Algorithme 5.2: Transformation de la requête exécutée par l'utilisateur dans le cas d'une insertion dans plusieurs colonnes.

Nous avons testé l'algorithme « Algo-client2 » sur la table « Travailleur» qui possède la structure (G) ci-dessous:

Travailleur (matricule, dose_prof, dose_sup, dose_neut, dose_int) (G)

Nous allons considérer que les colonnes de la table « Travailleur » sont toutes sensibles, elles seront ainsi protégées par un chiffrement selon les éléments de la table 5.3.

Colonne sensible	**Algorithme**	**Clé de chiffrement définie dans la requête**
matricule	AES256	&@I@encrypt@my@
dose_prof	AES256	&@I@encrypt@he@
dose_sup	AES256	&@I@encrypt@sh@
dose_neut	AES256	&@I@encrypt@it@
dose_int	AES256	&@I@encrypt@we@

Table 5.3: Les clés de chiffrement associées aux colonnes de la table « Travailleur ».

Colonne sensible	**Clé de chiffrement définie dans la requête**	**Clé générée par la fonction Func_Mc du serveur**
matricule	&@I@encrypt@my@	11D3F68E95C4FD54A 29D7B3AC57A0ED7
dose_prof	&@I@encrypt@he@	110DD62FCD6649F8 4514408C6FA25A64
dose_sup	&@I@encrypt@sh@	0130A5BFC8A94A13 84DA58A64FFAE4C2
dose_neut	&@I@encrypt@it@	C76A485734BE907A BD6961C9D08E4F09
dose_int	&@I@encrypt@we@	89131623649631ECD 38CA49F418937D7

Table 5.4: Les clés de chiffrement générées par la fonction Func_Mc.

Les deux tableaux (Table 5.5) et (Table 5.6) exposent les résultats obtenus d'insertion de données dans la table « Travailleur » avant et après le chiffrement.

matricule	dose_prof	dose_sup	dose_neut	dose_int
A00001	1,2	10,2	1,2	0
A00002	2,4	5,47	0,2	0
A00003	6,5	7,21	0,8	0
A00004	2,8	3,55	0	0
A00005	1,1	4,7	0,22	0
A00006	6,5	8,6	0,14	0

Table 5.5: La table « Travailleur » avant chiffrement

matricule	dose_prof	dose_sup	dose_neut	dose_int
6F21BF18357B763FEFAB6AAF88A6C56F	B3F0C6211E9CAE6596ED644BC2EA1856	512C11B29D04A49BAF385DADAE8D0AAF	78B5FF9CD2B4E2827C85DCA18CC60B5E	A0565A1A07E68E4BA8F5EEB463C112F4
7932847C2315057248AC7B6CD8AE43C5	F6F7F83191B5DA9DCD87F4D0D65B60A4	7F82F1DB035B32300E10942AF6684B29	B4225DC665046E387DE8F8A9DBAF8A9B	A0565A1A07E68E4BA8F5EEB463C112F4
8DA50B611FD6BAFAA8713E1B11CEDA4B	634404F58283203C69F02205E0598A1A	0EF0396C9FC1A89CB59AF052A4C63500	7CE33F318F7F129860BA295773DDFD98	A0565A1A07E68E4BA8F5EEB46C112F4
9EA525208968EEB35888883B95CB0C63	BC396AB488C210E04FB56270CC988AE6	C8CAB2F2DC06806A5D27C71138654DB9	4C2126BFD4DF526BBD959D55C8F2D63D	A0565A1A07E68E4BA8F5EEB46C112F4
4197B8E915413B5F6BD3B1D93950731C	1E4BA78439A250946FFAF9E36DBD7391	9FE61AA83589BE0F09687FABEC3DE952	83E81FDA145C67925BB45F0BE1F25D8A	A0565A1A07E68E4BA8F5EEB463C112F4
B1265EE20427E09A381297FA2543DE48	634404F58283203C69F02205E0598A1A	F205864977DE23212853B78F6CA4D17C	7DBFE90E0782FDCF32768E287AC4CBB8	A0565A1A07E68E4BA8F5EEB463C112F4

Table 5.6: La table « Travailleur » après chiffrement.

2. Résultats et discussions

Les résultats obtenus de l'implémentation des deux solutions nous permettent de déduire les points ci-dessous :

Les deux solutions permettent une meilleure protection des clés de chiffrement. Aucun acte malveillant de la part du client en vue de détenir les clés ne peut réussir, leurs valeurs réelles ne sont pas définies sur le client. En effet, l'externalisation de la protection des clés du client vers le serveur est une manière efficace de sécuriser les clés notamment si le serveur est un site de confiance. Nos solutions sont bien adaptées à ce concept, autrement dit qu'elles sont conditionnées par un niveau élevé vis-à-vis du serveur de BD et ses exploitants. Shmueli et al. ont souligné dans [79] que le niveau de confiance dans le serveur de BD est un critère fondamental d'une solution de chiffrement de BD. Le niveau de confiance partiel est l'un des trois niveaux de confiance mentionnés dans leur manuscrit, il signifie que le serveur de BD, sa mémoire et le SGBDR sont approuvés, bien que le stockage secondaire utilisé ne le soit pas. Nous avons choisi à ce que notre solution fait partie de ce concept, c'est à dire aux solutions qui portent un niveau de confiance partiel.

Nos solutions conviennent parfaitement aux architectures physiques à 2 et 3 niveaux, pour chaque architecture les fonctions de protection des clés Func() et Func_Mc() sont implémentées à l'intérieur du serveur de BD. Dans le cas où le choix du propriétaire de la BD est tombé sur une gestion complète de la BD dans le Cloud (DBAAS), les fonctions feront parties de l'ensemble des objets de la BD à externalisés.

D'autres types d'attaques peuvent compromettre nos solutions, il s'agit de la retro ingénierie logiciel. En effet, nous suggérons de mettre nos solutions au sein des applications des clients en mode exécutable, nous parlons ici des algorithmes (Algo-client1 et Algo-client2). Un attaquant peut révéler le code source à partir des exécutables de nos fichiers de solution en s'appuyant sur des outils spéciaux. L'attaquant peut ainsi voir le retour des fonctions et déterminer la manière dont les fonctions fournissent les nouvelles clés. Nous prévoyant dans ce cas une autre solution qui consiste à mettre tout le code des requêtes des clients dans des procédures stockées qui vont être localisées également sur le serveur de BD.

Conclusion :

La dangerosité des d'attaques internes sur les BD devient de plus en plus élevée. Lors de la mise en œuvre d'une stratégie de chiffrement côté client, la probabilité d'attaquer les clés est un facteur important à ne pas sous-estimer, l'attaquant peut les détenir et déchiffrer toutes les

données sans lui est remonté. Dans ce chapitre, nous avons proposé et implémenté deux solutions qui protègent les clés de chiffrement côté client. Le principe de base des solutions consiste à transformer les clés de chiffrement définies dans les fichiers de l'application en d'autres clés considérées comme les clés réelles de chiffrement/déchiffrement de la BD. Cette transformation est assurée par des appels à distance à des fonctions stockées sur le serveur de BD. Nos solutions sont faciles à mettre en place, elles ne requièrent aucun matériel ou investissement supplémentaire. En outre, elles sont bien adaptées à une configuration physique à 2 et 3 niveaux, plus particulièrement si le serveur de BD est un site de confiance.

CONCLUSION GENERALE

Ce travail présente le fruit de quatre ans de recherche dans le domaine de sécurité des BD par le chiffrement des données. Certes, les moyens de protection sont nombreux. Toutefois, le chiffrement reste le moyen le plus puissant puisque c'est la dernière ligne de défense contre les menaces des attaquants. Il convient de souligner que cette conviction vient du fait du développement et de l'approfondissement de nos connaissances dans ce domaine. D'ailleurs, un attaquant qui a pu compromettre toutes les barrières de protection mises en place pour protéger une BD n'a quasiment aucune chance de réussir l'attaque si la BD est bien protégée avec un modèle de chiffrement pertinent. Malgré les rares contributions présentées dans la littérature dans ce domaine, nous nous sommes intéressés à l'étude approfondie du déploiement des modèles de chiffrement dans la protection de la confidentialité des BD, et nous considérons que cet aspect à une importance cruciale dans toute conception d'un système de chiffrement dans un SGBDR.

A ce titre, nous avons passé en revue dans une première phase les solutions de chiffrement de BD qui existent sur le marché, ce qui nous a permis d'avoir une vision globale des fonctionnalités disponibles et celles qui manquent dans l'ensemble de ses solutions, ainsi que les problèmes de sécurité et de la performance qui y peuvent être associés.

Une étude particulière a été accordée également dans une deuxième phase à l'ensemble des travaux de chercheurs dans la thématique des modèles de chiffrement des BD. La combinaison des deux phases nous a conduit à contribuer par des propositions portant sur la protection de la confidentialité des BD sensibles et d'autres relatives à la problématique de la protection des clés de chiffrement au niveau du serveur de la BD et des applications.

Ainsi, La première contribution porte sur la proposition d'un nouveau modèle global qui protège la confidentialité d'une BD. C'est un concept innovant et original permettant de changer la définition classique du chiffrement sur les tables de la BD par l'utilisation d'un nouveau concept appelé « Classes de chiffrement ». Les résultats de l'implémentation du modèle démontrent les principaux avantages qu'il apporte par rapport à ceux proposés dans la littérature en termes de mécanisme de chiffrement des données, de protection des clés et de protection de la structure de la BD.

La deuxième contribution concerne une nouvelle approche de protection des clés de chiffrement au sein du SGBDR. Cette méthode permet de générer les clés principales en

fonction du niveau de granularité de chiffrement adopté par le modèle de chiffrement du SGBDR. Notre approche permet de fournir une gamme élargie de clés principales (colonne, Tablespace, table et ligne). Les résultats obtenus de l'implémentation démontrent l'efficacité de notre concept par rapport aux approches existantes. Notre solution convient parfaitement à une implémentation dans les moteurs des SGBDR libres comme un package de protection des clés de chiffrement.

Quant à la troisième contribution, nous nous sommes intéressés aux problèmes d'attaques internes sur les clés de chiffrement lorsque nous adoptons un chiffrement côté client. Le principe de la solution repose sur la transformation de la requête exécutée par un utilisateur en une nouvelle forme de requête qui fait appel à une fonction distante implémentée dans le serveur de la BD. Le rôle de la fonction distante est la conversion de la clé de chiffrement définie sur les applications en une clé réelle de chiffrement et de déchiffrement. Les résultats de l'implémentation de notre solution montrent une meilleure sécurité des clés vis-à-vis des clients. Cette solution convient parfaitement aux architectures physiques à 2 et 3 niveaux. Dans chaque architecture, la fonction est implémentée dans le serveur de BD.

Les solutions que nous avons proposées peuvent être toujours améliorées davantage afin d'obtenir une sécurité maximale. Nous comptons améliorer la notion des « Classes de chiffrement » afin qu'elle intègre plus de fonctionnalités de sécurité, nous pensons à corréler cette notion avec l'authentification des utilisateurs, et ainsi se prémunir des attaques internes. Il serait intéressant également d'implémenter les modèles associés aux classes de chiffrement dans des moteurs de BD « open-source » afin de mieux évaluer la performance des requêtes. En ce qui concerne les approches de protection des clés que nous avons proposées, nous projetons les améliorer dans le cas où nous souhaitons modifier la valeur générée de la clé principale à partir d'une structure de table de BD fixe. Il reste encore d'autres pistes à explorer notamment, l'étude des modèles de chiffrement des BD basés sur l'utilisation des algorithmes de chiffrement à flots.

Enfin, il reste à mettre en lumière deux points importants :

Premièrement, la cybersécurité et la sécurité de l'information sont deux notions qui vont prendre la tête des politiques globales des grandes industries, voire des Etats dans le futur proche. La notion de la pauvreté d'un pays sera fortement corrélée à sa pauvreté numérique, à la manière dont le pays génère, gère et protège ses données stratégiques et sensibles. Toutes les politiques de développement élaborées dans l'avenir par les pays seront bâties sur une

économie du numérique qui doit forcément adopter en parallèle une forte stratégie de sécurité de l'information, que ce soit au niveau des BD, des infrastructures du Cloud et de BIGDATA, et sans oublier bien évidemment la technologie révolutionnaire du Blockchain. D'ailleurs, toutes les institutions financières mondiales telle que la banque mondiale orientent les états membres pour se doter de politiques de sécurité de l'information afin qu'ils puissent conserver leurs patrimoines matériels et immatériels pour réaliser un développement durable.

D'autres parts, nous aimerions souligner ici que tous les concepts de sécurité bâtis autour des BD doivent être forcément revus dans la prochaine décennie. Des menaces très avancées exploitant l'intelligence artificielle (IA) se préparent d'ores et déjà par les cyber-attaquants, et un nouveau terme apparaitra dans le dictionnaire de la sécurité informatique, celui de la cyber-IA offensive. Un rapport très intéressant publié par Darktrace en 2019 expose des cas réels sur des attaques basées sur l'IA [19]. Avec l'IA, l'attaque de vol de données par malware deviendra hyper sophistiquée. Le malware développera de l'intelligence et s'adaptera en fonction des conditions de l'attaque. Les vulnérabilités des BD seront facilement détectables par les robots hacker, et les logiciels malveillants se propageront de manière automatique à travers des décisions autonomes. Avec les robots hacker nous gagnerons en temps et en ressources humaines. Par conséquent, à une cyber-IA offensive devra répondre une cyber-IA défensive.

LISTES DES PUBLICATIONS ET COMMUNICATIONS

1) **Karim El Bouchti**, Soumia Ziti, Fouzia Omary, Nassim Kharmoum, "A New Database Encryption Model Based on Encryption Classes" Journal of Computer Science, vol.15,no.6, pp.844-854, June 2019.
2) **Karim El Bouchti**, Soumia Ziti, Fouzia. Omary, Nassim Kharmoum, "A New Solution to Protect Encryption Keys when Encrypting Database at the Application Level" International Journal of Advanced Computer Science and Applications, Vol. 11, No. 1, 2020.
3) **Karim El Bouchti**, Soumia Ziti, Fouzia Omary, Nassim Kharmoum, "New solution implementation to protect encryption keys inside Database Management System" Advances in Science, Technology and Engineering Systems Journal, 2020, unpublished.
4) **Karim El Bouchti**, Soumia Ziti, Fouzia Omary, "A new approach to protect encryption keys in Database Management System", Advances in Intelligent Systems and Computing, 2019, In press.
5) **Karim El Bouchti**, Soumia Ziti, Fouzia Omary, "A new approach to protect encryption keys in Database Management System", Proceedings of the International Conference Modern Intelligent Systems Concepts, Rabat, Morocco, pp.12-13, December 2018
6) **Karim El Bouchti**, Soumia Ziti, Yassin Ghazali, Nassim Kharmoum, "Sécurité des Bases de Données : Menaces principales et solution de chiffrement existantes", in Proceedings of the JDSIRT Conference Information Systems, Networks Telecommunications, Meckness, Morocco, 2018.
7) **Karim El Bouchti**, Nassim Kharmoum, Soumia Ziti, Fouzia Omary, "A new approach to prevent internal attacks on Database encryption keys" Proceedings of the International Conference Scientific Days Applied Sciences, Larache, Morocco,pp.15-16, February 2019
8) Nassim Kharmoum, **Karim El Bouchti**, Soumia Ziti, & Fouzia Omary. DESCRIPTIVE ANALYSIS OF BUSINESS VALUE MODELS'TRANSFORMATION IN MDA APPROACH. Proceeding of" 3rd Scientific Days of Applied Sciences, 7(3), pp.71-78, 2019.

9) Kharmoum Nassim, Ziti Soumia, Rhazali Yassine, **Elbouchti Karim**, Rhalem Wajih. Analytical Study of Requirements Models Construction and their Transformations in MDA Approach. In 5th Edition of the JDSIRT Conference on Information Systems, Networks and Telecommunications, Meknes, Morocco, page 12. JDSIR, 2018.
10) Nassim Kharmoum, Soumia Ziti, **Karim El Bouchti**, & Fouzia Omary, "Transformations' Study Between Requirement Models and Business Process Models in MDA Approach", in International Workshop on Advancement in Model Driven Engeneering (MADE 2020), Warsaw, Poland, 2020.

REFERENCES

[1] A. El azzouzi, "La Cyber-criminalité au Maroc", 2010.

[2] Ahsan wajahat, A novel approach of unprivileged keylogger detection

[3] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. Handbook of applied cryptography. CRC press, 1996.

[4] Alhanjouri, Mohammed A., A. L. Derawi, and M. Ayman. "A New Method of Query over Encrypted Data in Database using Hash Map." A New Method of Query over Encrypted Data in Database using Hash Map 41.4 (2012).

[5] Al-Souly, Hanan A., Abeer S. Al-Sheddi, and Heba A. Kurdi. "Lightweight symmetric encryption algorithm for secure database." International Journal of Advanced Computer Science and Applications 3.4 (2013): 53-62.

[6] Alwan, Zainab S. et Younis, Manal F. Detection and Prevention of SQL Injection Attack: A Survey. International Journal of Computer Science and Mobile Computing, 2017, vol. 6, no 8, p. 5-17.

[7] Basharat, Iqra, Farooque Azam, and Abdul Wahab Muzaffar. "Database security and encryption: A survey study."International Journal of Computer Applications 47.12 (2012).

[8] Boicea, Alexandru et al. "Database encryption using asymmetric keys: a case study." 2017 21st International Conference on Control Systems and Computer Science (CSCS). IEEE, 2017.

[9] Bouganim, L., & Guo, Y. 2011. Database encryption. Encyclopedia of Cryptography and Security, 307-312.

[10] Bouganim, Luc. Sécurisation du Contrôle d'Accès dans les Bases de Données. Diss. 2006.

[11] C. Hanin, " Nouvelle approche de sécurité informatique basée sur les automates cellulaires", Université Mohammed V Rabat, 2017.

[12] C. Mavromati, "Cryptanalyse des algorithmes de type Even-Mansour", Univeristé Paris-Saclay, 2017.

[13] Catalin Cimpanu, "Washington lance une cyber-attaque contre les systèmes de missiles iraniens". Online : https://www.zdnet.fr/actualites/washington-lance-une-cyber-attaque-contre-les-systemes-de-missiles-iraniens-39886467.htm (consulté le: 24/06/2019).

[14] Chen, B. H., Cheung, P. Y., Cheung, P. Y., & Kwok, Y. K. 2018. Cypherdb: A novel architecture for outsourcing secure database processing. IEEE Transactions on Cloud Computing, 6(2), pp.372-386.

[15] Christina Boura. Analyse de fonctions de hachage cryptographiques. Cryptographie et sécurité [cs.CR]. Université Pierre et Marie Curie - Paris VI, 2012. Français. tel-00767028

[16] D. Manivannan, R .Sujarani, Light weight and secure database encryption using TSFS algorithm, Proceedings of the International Conference on Computing Communication and Networking Technologies, 2010, pp. 1-7.

[17] Dakheel, A. H., Ucan, O. N., Bayat, O., & Jasim, H. H. (2019). CYBER ATTACK DETECTION IN REMOTE TERMINAL UNIT OF SCADA SYSTEMS. International Journal of Computer Science and Mobile Computing, Vol.8 Issue.3, March- 2019, pg. 193-203

[18] Darktrace, "The next Paradigm Shift AI- Driven Cyber attacks", Research White Paper, 2019.

[19] DATAPROTECT, AUSIM. "Les enjeux de la scurité au Maroc", juin 2018.

[20] Decision of the European Court of Justice (Fifth Chamber) 19 December 2013 – Case No. C-202/12 IIC (2014). Directive No. 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the Legal Protection of Databases, Art. 7(1) and (5) – Innoweb BV v. Wegener ICT Media BV and Wegener Mediaventions BV. IIC - International Review of Intellectual Property and Competition Law. Vol.45, Issue 4, pp 465-465.

[21] Deepicata. Soni,N. (2015,May). Database Security: Threats and Security Techniques.International Journal of Advanced Research in Computer Science and Software Engineering, 5, 621-624.

[22] Deshmukh, Dr, Anwar Pasha, and Dr Qureshi. "Transparent Data Encryption--Solution for Security of Database Contents." arXiv preprint arXiv:1303.0418 (2013).

[23] Deshpande, Varad, and Debasis Das. "Efficient Searching Over Encrypted Database: Methodology and Algorithms." International Conference on Distributed Computing and Internet Technology. Springer, Cham, 2019.

[24] Dixit, P., Gupta, A. K., Trivedi, M. C., & Yadav, V. K. 2018. Traditional and Hybrid Encryption Techniques: A Survey. In Networking Communication and Data Knowledge Engineering(pp. 239-248). Springer, Singapore

[25] E. Biham et A. Shamir : Differential cryptanalysis of DES-like cryptosystems. Journal of Cryptology, 4(1):3–72, 1991.

[26] E. Shmueli, R. Waisenberg, Y. Elovici, E. Gudes, Designing secure indexes for encrypted databases, Data and Applications Security XIX (2005) 925-925

[27] E.Meneut, "La cyberguerre et la structuration des relations internationales : le cas Nord-Coreen,", Institut de relations internationales et strategiques, 2017.

[28] Elovici, Y., Vaisenberg, R., & Shmueli, E. (2018). U.S. Patent No. 9,934,388. Washington, DC: U.S. Patent and Trademark Office.

[29] Encrypter vos données avec DBMS CRYPTO. Online : http://droe-dba.over-blog.fr/article-encrypter-vos-donnees-avec-dbms_crypto-42771236.html (consulté le 08/2018).

[30] FIPS PUB 180-2 : Secure hash standard. FIPS Publications,août 2002. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf.

[31] FIPS PUB 197: Advanced encryption standard (AES). FIPS Publications, novembre 2001. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[32] FIPS PUB 46-3:Data Encryption Standard (DES), October 1999. https://csrc.nist.gov/csrc/media/publications/fips/46/3/archive/1999-10-25/documents/fips46-3.pdf

[33] Gahi, Youssef, Mouhcine Guennoun, and Khalil El-Khatib. "A secure database system using homomorphic encryption schemes." arXiv preprint arXiv:1512.03498 (2015).

[34] Galushka, V. V., Aydinyan, A. R., Tsvetkova, O. L., Fathi, V. A., & Fathi, D. V. 2018, System of end-to-end symmetric database encryption. In Journal of Physics: Conference Series(Vol. 1015, No. 4, p. 042003). IOP Publishing

[35] Hashim, Hassan B. "Challenges and Security Vulnerabilities to Impact on Database Systems." Al-Mustansiriyah Journal of Science 29.2 (2018): 117-125.

[36] Homoliak, Ivan et al. "Insight into insiders and it: A survey of insider threat taxonomies, analysis, modeling, and countermeasures." ACM Computing Surveys (CSUR) 52.2 (2019): 30.

[37] Iqra Basharat, Farooque Azam, Abdul Wahab Muzaffar," Database Security and Encryption: A Survey Study", International Journal of Computer Applications (0975 – 888) Volume 47– No.12, June 2012.

[38] Irmak, Erdal, and İsmail Erkek. "An overview of cyber-attack vectors on SCADA systems." 2018 6th International Symposium on Digital Forensic and Security (ISDFS). IEEE, 2018.

[39] Itamar, Einav, and Achi Rotem. "Encryption directed database management system and method." U.S. Patent Application No. 15/570,775, 2018.

[40] Jacob, S., 2012. Protection cryptographique des bases de données: conception et cryptanalyse. Unpublished dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Pierre et Marie Curie-Paris VI University, Paris, France.

[41] Jayant D, Bokefode et al. "Analysis of dac mac rbac access control based models for security." International Journal of Computer Applications 104.5 (2014): 6-13.

[42] Kazuo Sakiyama, Yu Sasaki, and Yang Li. Security of Block Ciphers : From Algorithm Design to Hardware Implementation. John Wiley & Sons, 2016.

[43] Kulkarni, S., and Siddhaling Urolagin. "Review of attacks on databases and database security techniques." International Journal of Emerging Technology and Advanced Engineering, SSN (2012): 2250-2459.

[44] L. Bouganim, P. Pucheral, Chip-secured data access: confidential data on untrusted servers, Proceedings of the 28th international conference on Very Large Data Bases-Volume 28 (2002) 131-142

[45] L. Bouganim, Y. Guo et al., Database encryption, Encyclopedia of Cryptography and Security (2009) 1-9

[46] Le site Internet officiel Volkswagen Maroc piraté. Online : https://www.undernews.fr/hacking-hacktivisme/volkswagen-maroc-pirate-hacked-database-leak-pastebin.html. (Consulté le 02/2017).

[47] Liu, Lianzhong, and Jingfen Gai. "A new lightweight database encryption scheme transparent to applications." 2008 6th IEEE International Conference on Industrial Informatics. IEEE, 2008.

[48] M. Canim, M. Kantarcioglu, Design and analysis of querying encrypted data in relational databases, Data and Applications Security XXI (2007) 177-194.

[49] M. Matsui : Linear cryptanalysis method for DES cipher. In Advances in Cryptology - EUROCRYPT 1993, volume 765 de Lecture Notes in Computer Science, pages 386–397. Springer-

[50] Ma, S., Mu, Y., & Susilo, W. 2018. A Generic Scheme of plaintext-checkable database encryption. Information Sciences, 429, pp.88-101. Doi.org/10.1016/j.ins.2017.11.010

[51] Malik, Mubina, and Trisha Patel. "Database securityattacks and control methods." International Journal of Information6.1/2 (2016) : 175-183.

[52] Mattsson, U. T. 2005. A practical implementation of transparent encryption and separation of duties in enterprise databases: protection against external and internal attacks on databases. In Seventh IEEE International Conference on E-Commerce Technology (CEC'05) (pp. 559-565). IEEE.

[53] Maurya, Arvind Kumar et al. "Protection of Data Stored in Transparent Database System using Encryption." Journal of Computer and Mathematical Sciences 10.1, pp.190-196, 2019.

[54] Microsoft Docs. Transparent Data Encryption TDE. Online: https://docs.microsoft.com/en-us/sql/relational databases/security/encryption/transparent-data-encryption?view=sql-server-2017.

[55] Microsoft SQL Server docs: Encrypt a Column of data.Online: https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/encrypt-a-column-of-data?view=sql-server-2017.

[56] Mirante, Dennis, and Justin Cappos. "Understanding password database compromises." Dept. of Computer Science and Engineering Polytechnic Inst. of NYU, Tech. Rep. TR-CSE-2013-02 (2013).

[57] Moore, David et al. "Inside the slammer worm." IEEE Security & Privacy 4 (2003): 33-39.

[58] Mukherjee, Sourav. "Popular SQL Server Database Encryption Choices." arXiv preprint arXiv:1901.03179 (2019).

[59] MySQL Server Documentation. MySQL 5.7 Reference Manual Online: https://dev.mysql.com/doc/refman/5.7/en/innodb-tablespace-encryption.html.

[60] MySQL Server Documentation. MySQL 5.7 Reference Manual Online: https://dev.mysql.com/doc/refman/8.0/en/encryption-functions.html.

[61] Charlie Osborne ,"British Airways: Cyberattack, data theft bigger than we first thought". Online: https://www.zdnet.com/article/british-airways-cyberattack-data-theft-bigger-than-we-first-thought/. (consulté le :25/10/2018).

[62] Oracle (2016), Oracle® Database Advanced Security Administrator's Guide 11g Release 2 (11.2[online] Technical Document:

[63] Ponemon Institute, "2016 Global Encryption Trends Study", Research Report, Fevrier 2016.

[64] Priebe, C.,Vaswani,K., and Costa,M.(2018). Enclavedb: A secure database using sgx." IEEE Symposium on Security and Privacy (SP). IEEE,

[65] R. Rivest : The MD4 message digest algorithm. In Advances in Cryptology - CRYPT0 1990, volume 537 de Lecture Notes in Computer Science, pages 303–311. Springer, 1991.

[66] R. Rivest : The MD5 message digest algorithm. Internet RFC 1321, avril 1992.

[67] Ramahefy, T.R. Rakotomiraho, S. Rabeherimanana, L. 2015. SQL Parser XML – Xquery : Approche de détection des injections SQL,MADA-ETI, Vol.1, 2015, [online] http://madarevues.recherches.gov.mg/IMG/pdf/Art_no1_2015_vol_1_pp_1-9_SQL_Parser_XML_-_Xquery_Approche_de_detection_des_injections_SQL.pdf.

[68] Rapport n° 2 de la Délégation ministérielle aux industries de sécurité et à la lutte contre les cybermenaces. "Etat de la menace liée au numerique en 2018", mai 2018.

[69] Rapport n° 3 de la Délégation ministérielle aux industries de sécurité et à la lutte contre les cybermenaces, "Etat de la menace liée au nmerique en 2019", mai 2019.

[70] Raut, Sweta et al. "A Review on Methods for Prevention of SQL Injection Attack." (2019).

[71] Recommendation for Block Cipher Modes of Operation: Methods and Technics. NIST Special Publication 800-38A, 2001. http://csrc.nist.gov/publications/ nistpubs/800-38C/SP800-38C_updated-July20_2007.pdf.

[72] Bitglass, "Insider threat report", 2019.

[73] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, Efficient and private access to outsourced data, in: Distributed Computing Systems (ICDCS), 2011 31st International Conference on, IEEE, 2011, pp. 710-719.

[74] S. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, Encryption policies for regulating access to outsourced data, ACM Transactions on Database Systems (TODS) 35 (2) (2010) 12.

[75] Sallam, A. I., El-Rabaie, E. S., & Faragallah, O. S. 2012. Encryption-based multilevel model for DBMS. Computers & security, 31(4), pp.437-446. Doi.org/10.1016/j.cose.2012.02.008.

[76] Sekhar, J. Raja, and G. Sivaranjani. "Database Encryption Using TSFS Algorithm." (2018)

[77] Sesay, S., Yang, Z., Chen, J., & Xu, D. 2005, A secure database encryption scheme. In Second IEEE Consumer Communications and Networking Conference, 2005. CCNC. 2005 (pp. 49-53). IEEE. ISSBN :1546-9239

[78] Shmueli, E., Vaisenberg, R., Elovici, Y., & Glezer, C. 2010. Database encryption: an overview of contemporary challenges and design considerations. ACM SIGMOD Record, 38(3), pp.29-3.

[79] Shmueli, E., Vaisenberg, R., Gudes, E., & Elovici, Y. (2014). Implementing a database encryption solution, design and implementation issues. Computers & security, 44, 33-50.

[80] Shulman, Amichai, and C. T. O. Co-founder. "Top ten database security threats." How to Mitigate the Most Significant Database Vulnerabilities (2006).

[81] Sortie médiatique de l'Association «Kif Kif». Online : http://aujourdhui.ma/focus/sortie-mediatique-de-lassociation-kif-kif-les-homosexuels-marocains-sortent-du-placard-68121

[82] Sun, Xun et al. "Application-level in-place encryption." U.S. Patent Application No. 15/830,748.

[83] Sung Hsueh, Database Encryption in SQL Server 2008 Enterprise Edition, SQL Server Technical Article, 2008. http://msdn.microsoft.com/enus/library/cc278098.aspx.

[84] Y.Ding, K.Klein. Model-driven application-level encryption for the privacy of e-health data. In Availability, Reliability, and Security, 2010. ARES'10 International Conference on (pp. 341-346). IEEE.

[85] Y. Elovici, R.Waisenberg, E. Shmueli, E. Gudes, A Structure Preserving Database Encryption Scheme., Workshop on Secure Data Management,, Canada, August.

[86] Zabihimayvan, M., & Doran, D. (2019) "Fuzzy Rough Set Feature Selection to Enhance Phishing Attack Detection." arXiv preprint arXiv:1903.05675.

[87] Hanin, Charifa & Echandouri, Bouchra & Omary, Fouzia & Bernoussi, Souad. (2017). L-CAHASH: A Novel Lightweight Hash Function Based on Cellular Automata for RFID. 287-298. 10.1007/978-3-319-68179-5_25.

[88] Echandouri, Bouchra & Hanin, Charifa & Omary, Fouzia & Elbernoussi, Souad. (2017). LCAHASH-MAC: A new lightweight message authentication code using cellular automata for RFID. 1-6. 10.1109/WINCOM.2017.8238158.

[89] Bekkaoui, K., Ziti, S., & Omary, F. (2020). A Robust Scheme to Improving Security of Data using Graph Theory. International Journal of Advanced Computer Science and Applications, 11.