



Université Cheikh Anta DIOP de Dakar

Faculté des Sciences et Techniques

Département Mathématiques et Informatique

Laboratoire d'Algèbre de Cryptologie de  
Géométrie Algébrique et Applications

LACGAA

## *Master Transmission de Données et Sécurité de l'Information*

Thème :

# Conception et développement d'un scanner de vulnérabilité d'une base de données

Présenté et soutenu par :  
Abdoulaye Boumanga BA

Sous la direction de :  
Dr. Abdoul Aziz CISS

Jury : 2

Président : Pr Cheikh C. T. GUEYE

Enseignant chercheur UCAD

Membres : Pr Djiby SOW

Enseignant chercheur UCAD

M. Ahmedou KHALIFA

Enseignant, Ingénieur SSI

M. Adama MBOGJI

Ingénieur

**Année Académique 2016 – 2017**

## **DEDICACE :**

*Je dédie ce travail à mon père et ma mère pour qui j'ai une grande admiration et avec qui j'ai appris le respect d'autrui et des valeurs morales.*

*À mes frères aînés Gervais MABADI et Jeraud MBOZOKOUE qui sans lesquels je n'aurais jamais fait mon master.*

*À ma famille, mes amis, et à toutes les personnes que j'aime.*

## REMERCIEMENTS :

*Ce travail est l'aboutissement d'un long cheminement au cours duquel j'ai bénéficié de l'encadrement, des encouragements et du soutien de plusieurs personnes, à qui je tiens à dire profondément et sincèrement merci.*

*Je tiens à remercier tout d'abord mon directeur de recherches, Docteur Abdoul Aziz CISS, pour sa patience, et surtout pour sa confiance, ses remarques et ses conseils, sa disponibilité et son esprit d'humilité.*

*Qu'il trouve ici le témoignage de ma profonde gratitude.*

*Je voudrais également remercier les professeurs Mamadou SANGHARE, Oumar DIANKHA, Djiby SOW, Cheikh C. T. GUEYE, ainsi que le corps des enseignants de la formation TDSI pour toutes les initiatives qu'ils ont pu mettre en place afin de produire une telle formation de qualité et de haut niveau.*

*Merci au Docteur Cherif DEME pour toutes les heures dépensées afin de m'orienter dans ce travail.*

*Mes remerciements vont à l'endroit de tous mes collègues et amis de classe, pour leur amour, le respect et la considération qu'ils m'ont accordé tout au long de ma formation en TDSI.*

*Grand merci à toi Joël TANZOUAK pour tout le temps que tu as dû sacrifier en m'assistant dans l'élaboration de ce travail depuis la toute première idée qui pouvait se dégager jusqu'à la dernière correction.*

*Je vous remercie Papa et Maman pour vos multiples bénédictions à mon égard.*

*À ma chère famille, mes petits frères et compagnons de chambre Elie et Josué, à toi mon petit frère Alhassane BA, à ma petite sœur Ramatoulaye BA, et à vous mes chers grands frères et sœurs, belles-sœurs et beaux-frères, je vous dis sincèrement merci pour votre sourire quotidien et vos encouragements.*

*À toutes les personnes dont je n'ai pas cité les noms mais qui pourtant m'ont soutenu d'une façon ou d'une autre je vous dis un grand merci.*

*Merci à mon précieux frère et ami Amen KIOUNGOU, aux pasteurs Mathias GODONOU, Mathieu NZINGOULA, Andronicus G. MABADI, pour toutes vos prières.*

*Et par-dessus tout, merci infiniment à toi Eternel pour ton Amour et ta Grâce qui m'ont toujours porté à l'abri du danger et des échecs.*

## RÉSUMÉ :

Les bases de données sont un élément critique et complexe du système d'information, contenant souvent des données métiers sensibles. De ce fait, un soin tout particulier doit être apporté à leur sécurisation. Il est également fondamental pour un auditeur ou un pentester de connaître les angles d'attaque et vulnérabilités de ces systèmes. Dans ce document nous présentons une approche un peu particulière pour la conception et le développement d'un scanner de vulnérabilités de base de données. Cette approche vise à répondre aux problèmes non pris en compte par les outils existants, à savoir les problèmes de flexibilités, d'extensibilités, et des menaces importantes, telles que le chiffrement des données de la base et des communications, et l'exposition des données de sauvegarde. Au terme du travail effectué, les résultats obtenus à la suite des expérimentations, montrent que l'approche que nous avons adopté est très prometteuse pour les entreprises.

**Mots clés :** Base de données, Sécurité, Menaces, Scanner de vulnérabilités.

## **ABSTRACT:**

Databases are a critical and complex part of the information system, often containing sensitive business data. As a result, particular care must be taken in their security. It is also fundamental for an auditor or a pentester to know the angles of attack and vulnerabilities of these systems, this is where all the importance of vulnerability scanners. In this document we present a somewhat special approach for the design and development of a database vulnerability scanner. This approach aims to address issues not addressed by existing tools, such as issues of flexibilities, extensibility, and significant threats, such as database and communications encryption, and backup data. At the end of this work, the results obtained after the experiments show that the approach we have adopted is very promising for companies.

**Keywords:** Database, Security, Threats, Vulnerability Scanner.

## TABLE DES MATIÈRES :

LISTE DES TABLEAUX : .....	2
LISTE DES FIGURES : .....	3
LISTE DES ABREVIATIONS ET SIGLES : .....	4
INTRODUCTION : .....	5
PARTIE I : .....	6
GENERALITES SUR LES BASES DE DONNEES, SECURITE ET CONFORMITE .....	6
I – GENERALITES SUR LES BASES DE DONNEES .....	7
I.1 – Brève historique des bases de données et des SGBD .....	7
I.2 – Base de données et SGBD : Définitions et nuances .....	8
I.3 – Quelques SGBD (Caractéristiques et évolution au fil des années) .....	9
I.4 – Architectures fonctionnels d'une base de données (SGBD) .....	10
II – SECURITE ET CONFORMITE DANS LES BASES E DONNEES .....	12
II.1 – Conformité des bases de données : La loi Sarbanes Oxley et la norme COBIT .....	12
II.2 – Conformité des bases de données : La norme ISO 27001 .....	13
II.3 – Sécurité et menaces dans les bases de données .....	15
II.4 – Les scanners de vulnérabilités : Fonctionnement de base .....	21
PARTIE II : .....	26
ÉTUDE, CONCEPTION ET DEVELOPPEMENT DE LA SOLUTION .....	26
III – ÉTUDE ET CONCEPTION .....	27
III.1 – Critères recherchés du scanner et état de l'art sur les solutions existantes .....	27
III.2 – Analyse des besoins .....	29
III.3 – Présentation des outils de modélisation .....	31
III.4 – Modélisation .....	31
IV – DEVELOPPEMENT DE LA SOLUTION .....	35
IV.1 – Choix des outils technologiques .....	35
IV.2 – Principe de fonctionnement de base de notre scanner de vulnérabilités .....	36
IV.3 – Architectures, fonctionnement détaillé et sécurité de la solution .....	38
IV.4 – Déploiement et tests de la solution .....	41
PARTIE III : .....	46
INTERPRETATION DES RESULTATS, AMELIORATION ET RECOMMANDATIONS .....	46
V – INTERPRETATIONS DES RESULTATS .....	47
V.1 – Interprétation 1 : La rapidité du programme .....	47
V.2 – Interprétation 2 : La flexibilité du programme .....	48
V.3 – Comparaison .....	49
V.4 – Description des principaux avantages de la solution .....	49
VI – QUELQUES IDEES D'AMELIORATIONS ET LES RECOMMANDATIONS .....	50
VI.1 – Amélioration 1 : Optimisation en intégrant des outils open source existants .....	50
VI.2 – Amélioration 2 : Evolutivité de la solution .....	51
VI.3 – Recommandation 1 : Compte d'authentification à la base de données cible .....	52
VI.4 – Recommandation 2 : La Sécurité du réseau .....	52
CONCLUSION : .....	53
WEBOGRAPHIE : .....	54

## LISTE DES TABLEAUX :

<i>Tableau 1: Caractéristiques et évolution des SGBD</i>	9
<i>Tableau 2: Critères recherchés dans le scanner</i>	28
<i>Tableau 3: Tableau comparatif</i>	49

## LISTE DES FIGURES :

<i>Figure 1: Architecture globale</i>	10
<i>Figure 2: Architecture fonctionnelle interne</i>	11
<i>Figure 3 : La norme Cobit</i>	13
<i>Figure 4: Le modèle PDCA</i>	13
<i>Figure 5: Top 10 des menaces</i>	15
<i>Figure 6: Niveaux de sécurité</i>	20
<i>Figure 7: Restitution des résultats</i>	25
<i>Figure 8: Diagramme des cas d'utilisations</i>	32
<i>Figure 9: Diagramme des séquences</i>	33
<i>Figure 10 : Diagramme de classes</i>	34
<i>Figure 11: Restitution des résultats</i>	37
<i>Figure 12: Centre de traitement</i>	38
<i>Figure 13: SQLite DB</i>	38
<i>Figure 14: Modules</i>	39
<i>Figure 15: Architecture interne complète du système</i>	39
<i>Figure 16: Architecture réseau</i>	40
<i>Figure 17: Image future du scanner</i>	48



## LISTE DES ABREVIATIONS ET SIGLES :

- **COBIT:** -----Control **O**bjectives for **I**nformation and related **T**echnology
- **CVE:** ----- **C**ommon **V**ulnerabilities and **E**xposures  
(Standard d'identification des vulnérabilités en informatique)
- **CVSS:** ----- **C**ommon **V**ulnerability **S**coring **S**ystem
- **DB :** ----- **D**ata **B**ase (Base de Données)
- **GPL :** ----- **G**eneral **P**ublic **L**icense
- **IAS :** ----- **I**nternational **A**ccounting **S**tandards
- **RSSI :** ----- **R**esponsable de la **S**écurité des **S**ystèmes d'**I**nformation
- **IEEE:** ----- **I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
- **IFRS:** ----- **I**nternational **F**inancial **R**eporting **S**tandards
- **IP :** ----- **I**nternet **P**rotocol (Protocole internet)
- **ISO:** -----**I**nternational **O**rganization for **S**tandardization  
(L'Organisation internationale de normalization)
- **Nmap:** ----- **N**etwork **M**apper
- **OAT:** -----**O**racle **A**uditing **T**ools
- **ODAT:** -----**O**racle **D**atabase **A**ttacking **T**ool (Outil offensive de BD Oracle)
- **OS:** -----**O**perating **S**ystem (Système d'Exploitation)
- **PDCA:** -----**P**lan, **D**o, **C**heck, **A**ct
- **RPC:** -----**R**emote **P**rocedure **C**all
- **SGBD :** -----**S**ystème de **G**estion de **B**ase de **D**onnées
- **SIEM:** ----- **S**ecurity **I**nformation and **E**vent **M**anagement
- **SoA:** ----- **S**tatement of **A**pplicability
- **SOX:** -----**S**arbanes **O**xley
- **SSL/TLS:** ----- **S**ecure **S**ockets **L**ayer /**T**ransport **L**ayer **S**ecurity
- **TCP(S):** ----- **T**ransmission **C**ontrol **P**rotocol (**S**ecured)
- **VPN:** ----- **V**irtual **P**rivate **N**etwork
- **UML:** ----- **U**nified **M**odeling **L**anguage

## **INTRODUCTION :**

La protection des données occupe une place primordiale dans de nombreux systèmes sécurisés, un grand nombre d'entreprises comptent sur les outils fournis par les systèmes de gestion de base de données (SGBD) pour gérer la sécurité de leurs données. Les bases de données sont essentielles à beaucoup d'entreprises et d'organisations gouvernementales, car elles détiennent des données sensibles d'une grande importance qui doivent être accessibles plus efficacement tout en demeurant cohérentes. Ces données sensibles doivent être protégées de façon à ne laisser aucune entité non autorisée y accéder.

La sécurisation d'une base de données est une opération assez complexe que toute organisation doit prendre en considération, ceci afin de mener ses activités à l'abri d'une attaque informatique.

Dans un cadre d'architecture de sécurité, une démarche de gestion des risques est un point à identifier absolument. Ainsi la recherche de vulnérabilités dans la base de données est une tâche encore plus délicate et indispensable pour améliorer, prévenir et maintenir la sécurité de sa base de données, tâche qui nécessite très souvent l'utilisation d'outils automatisés appelés scanners de vulnérabilités. Les systèmes de gestion de base de données (SGBD) étant livrés par leurs concepteurs avec une sécurité plutôt minimaliste ou parfois même avec des défauts de conception contiennent bien souvent d'énormes vulnérabilités qu'il faut corriger avant la mise en service de ce dernier.

Cependant, les entreprises sont souvent confrontées à des vulnérabilités liées à ces systèmes de gestion de bases de données (SGBD), souvent pour les raisons suivantes :

- Le manque d'agents experts capable de maîtriser les outils de sécurité propres aux différents SGBD et d'assurer la sécurité d'amont en aval de la base de données ;
- Le manque de temps ou de moyens pour réaliser des audits complets de la base de données ;
- La négligence des responsables de la sécurité des bases de données, car très souvent, la base de données, une fois installée, configurée et mise en production semble désormais être imperturbable.

Dès lors, comment peut-on mettre au point un outil efficace et facilement utilisable, répondant aux standards de sécurité et capable d'effectuer des scans de vulnérabilités complets et ciblés, de manière à fournir un rapport d'audit claire et concis d'une base de données même en production ?

L'objectif de notre travail sera de concevoir et développer un scanner de vulnérabilité d'une base de données quelconque.

Nous présenterons en première partie Vue d'ensemble sur les bases de données, sécurité et conformité. La deuxième partie proposera l'étude, la conception et le développement de la solution. L'analyse, l'interprétation des résultats, et les recommandations seront décrites dans la troisième partie.

**PARTIE I :**  
**Généralités sur les bases de données, sécurité et conformité**

# I – Généralités sur les bases de données

## I.1 – Brève historique des bases de données et des SGBD

Dès 1965 apparaît l'idée de distinguer les données de leurs traitements. Vers la fin des années 60, on commence à voir l'apparition des premiers SGBD (Système de gestion de base de données) : les systèmes réseaux et hiérarchiques. [1] Ce n'est qu'à partir de 1970 qu'apparaît la deuxième génération de SGBD : les systèmes relationnels. Au début des années 80, la troisième génération de SGBD : les systèmes orientés objet, émerge. Ces avancées technologiques ont permis de résoudre les problèmes liés à l'utilisation de fichiers et possèdent les avantages suivants :

- Uniformisation de la saisie et standardisation des traitements (Tous les résultats de consultation sont sous forme de listes et de tableaux) ;
- Contrôle immédiat de la validité des données ;
- Partage de données entre plusieurs traitements, impliquant une baisse de la redondance des données.

Les **SGBD relationnels** créés dans les années 1970 se sont progressivement imposés jusqu'à devenir le paradigme de bases de données très largement dominant au début des années 1990. Plusieurs autres modèles de bases de données ont émergé, tels les SGBD orientés objet, SGBD hiérarchiques, SGBD relationnel-objet mais leur utilisation est restée très limitée. [2]

Au cours des années 2000 avec le développement de grandes entreprises internet (Google, Amazon, eBay...) brassant des quantités énormes de données et le développement de l'informatique en grappes, la domination sans partage du modèle relationnel a été remise en question car souffrant de limites rédhibitoires pour ces nouvelles pratiques.

Afin de répondre à ces limites, ces entreprises ont commencé à développer leurs propres systèmes de gestion de bases de données pouvant fonctionner sur des architectures matérielles distribuées et permettant de traiter des volumes de données importants. Les systèmes propriétaires qui en ont résulté, Google (BigTable), Amazon (Dynamo), LinkedIn (Voldemort), Facebook (Cassandra puis HBase), SourceForge.net (MongoDB), Ubuntu One (CouchDB), Baidu (Hypertable) ont été les premiers précurseurs du modèle **NoSQL**. [3]

Et aujourd'hui le besoin de gérer des données extrêmement volumineuses est flagrant et les technologies d'aujourd'hui ne permettent pas de le faire. Il faut repenser des concepts de base de la gestion de données qui ont été déterminés dans le passé. C'est alors qu'intervient la technologie **BigData** qui Selon V. Tréguier (2014) et selon la « *très courte histoire du big data* » publiée par Gil Press en 2013 [4] pour la période 1944-2012, sa naissance est liée aux progrès des systèmes de stockage, de fouille et d'analyse de l'information numérisée, qui ont permis une sorte de big bang de l'information stockée puis une croissance inflationniste de l'univers de la donnée numérisée. [5]

## I.2 – Base de données et SGBD : Définitions et nuances

La notion de base de données est généralement couplée à celle de réseau ou de système informatique pour désigner toute la structure regroupant les moyens mis en place pour le partage des données.

La base de données est donc la pièce centrale des dispositifs informatiques servant à la collecte, au stockage et à l'utilisation des informations recueillies. Ces dispositifs comportent un système de gestion de base de données (SGBD) qui est une sorte de logiciel moteur pour l'accès et la manipulation de la base de données.

### A. Base de données :

Une **base de données** est un ensemble structuré et organisé permettant le stockage de grandes quantités d'informations afin d'en faciliter l'exploitation (ajout, mise à jour, recherche de données). Ces informations sont en rapport avec une activité donnée et peuvent être utilisées par des programmes ou des utilisateurs communs, d'où la nécessité de leur mise en commun.

Plus simplement, la base de données c'est l'ensemble des fichiers qui contiennent les données, c'est-à-dire des informations à exploiter. [6]

### B. SGBD :

Pour permettre une utilisation optimale d'une base de données, il faut mettre en place un **système de gestion de base de données**, d'où l'intérêt d'un logiciel moteur : c'est le SGBD (en anglais DBMS : Data base management system).

Le SGBD est donc un ensemble de services permettant :

- L'accès aux données de façon simple
- D'autoriser l'accès aux informations à de multiples utilisateurs
- La manipulation des données présentes dans la base (insertion, suppression, modification)

Le SGBD joue aussi un rôle très important dans la cohérence d'une base de données et sa sécurité.

Toutefois, quand nous parlerons de base de données en général, ce sera pour désigner l'ensemble des données de la base avec le Système de gestion de base de données. Soit (**Base de données = fichiers de Données + SGBD**).

### I.3 – Quelques SGBD (Caractéristiques et évolution au fil des années)

Tableau 1: Caractéristiques et évolution des SGBD

Nom	Année	Éditeur	Caractéristiques	SQL	Licence
Pick	1968	Pick System		✓	BSD
Ingres	1974	<i>Ingres Corporation</i>	Relationnel, spatial, centralisé, distribué.	✓	GPL
MaxDB	1977	SAP AG et MySQL AB	Objet-relationnel, entreprise, centralisé.	✓	GPL
dBase	1978	Ashton-Tate	Relationnel, pour particuliers.		Propriétaire
Oracle Data Base	1979	Oracle Corporation	Relationnel, spatial, centralisé, distribué.	✓	Propriétaire
Firebird	1981	Firebird Foundation	Relationnel, centralisé, embarqué, entreprises.	✓	Interbase
Informix	1981	IBM	Distribué, pour entreprises.	✓	Propriétaire
DB2	1983	IBM	Relationnel, centralisé, distribué.	✓	Propriétaire
PostgreSQL	1985	Michael Stonebraker	Relationnel, centralisé pour entreprises.	✓	BSD
SQL Server	1989	Microsoft	Relationnel, centralisé, distribué, entreprises.	✓	Propriétaire
MySQL	1995	Oracle Corp et MySQL AB	Centralisé, embarqué, distribué, entreprises.	✓	GPL
SQLite	2000	D. Richard Hipp	Embarqué	✓	Public
HSQLDB	2000	<i>Thomas Mueller</i>	Relationnel, embarqué, centralisé, entreprises.	✓	BSD
MariaDB	2009	<i>Monty Program Ab</i>	Centralisé, embarqué, distribué, entreprises.	✓	GPL
MongoDB	2009	MongoDB, Inc	NoSQL, Orienté documents	✓	GNU AGPL

Dans ce tableau on peut voir les multiples systèmes de gestion de base de données, avec leurs dates de création, l'éditeur, les caractéristiques du système, son type (SQL ou non), et en dernière colonne le type de licence qu'il possède.

## I.4 – Architectures fonctionnels d’une base de données (SGBD)

Le fonctionnement d’une base de données peut être présenté sous différents angles, nous allons présenter le fonctionnement d’un point de vue global puis le fonctionnement interne d’une base de données en montrant les principaux processus qui sont mis en jeu.

### A. Architecture standard :

De façon standard, le fonctionnement d’une base de données suit le model **Client-Serveur**, le client constitue toute application qui souhaite exploiter les données de la base et le serveur désigne le système de gestion de base de données (SGBD) qui fournit tous les services nécessaires pour l’exploitation de données.

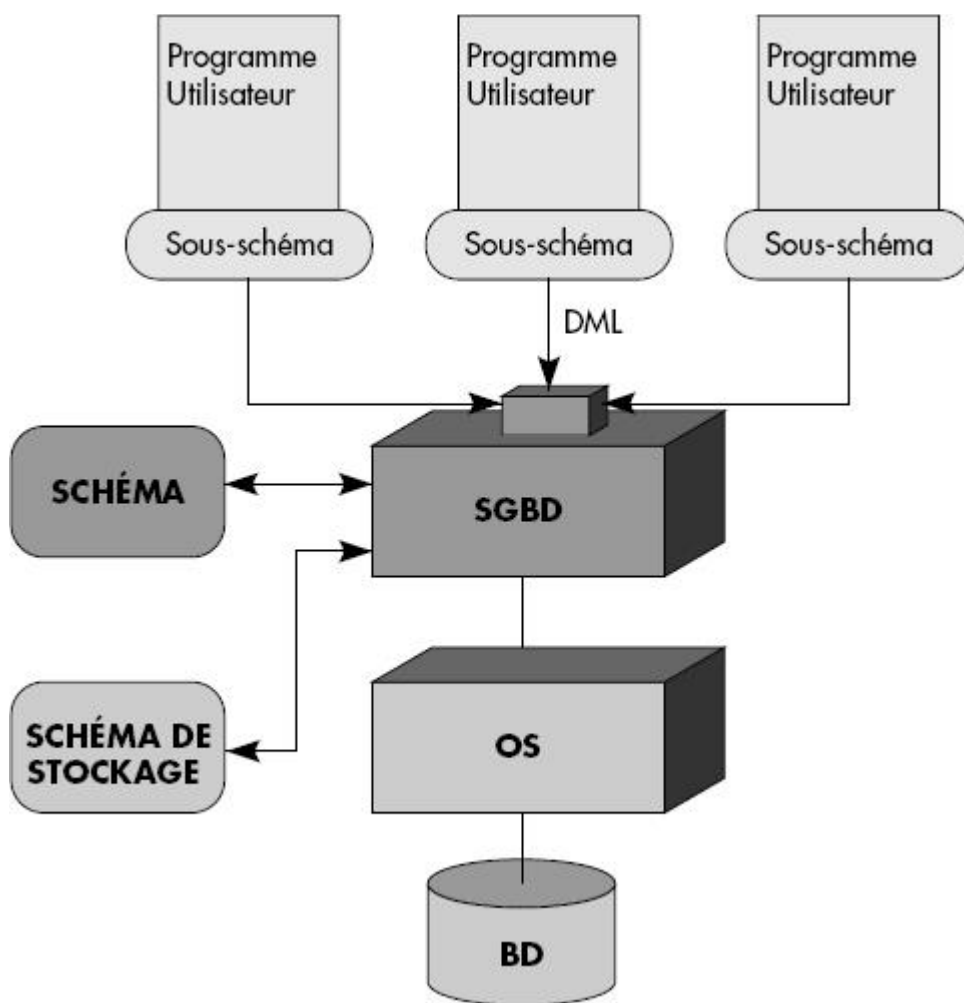


Figure 1: Architecture globale

**OS :** C'est le système d'exploitation, héberge les données ainsi que le SGBD

**BD :** C'est ensemble des fichiers de données

**SGBD :** C'est le logiciel de gestion de données, il crée des schéma de stockage des données

## B. Architecture de fonctionnement interne :

Le fonctionnement interne d'une base de données ou d'un système de gestion de base de données comporte plusieurs processus consécutifs : **Analyseur**, **Contrôleur**, **Optimiseur et Exécuteur**, au centre desquels on retrouve la **Métabase**. Cet ensemble coordonné de processus permet ainsi de manipuler les données de la base (**BD**). La figure ci-dessous illustre l'interaction entre les différents processus.

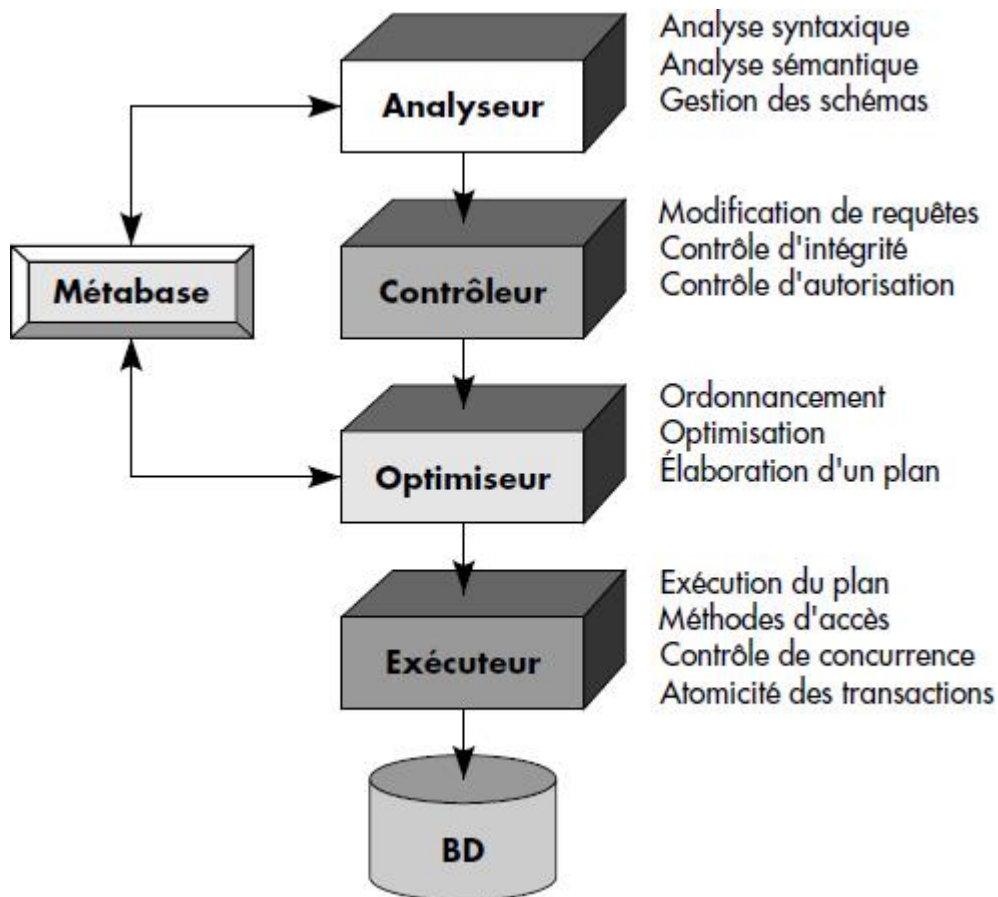


Figure 2: Architecture fonctionnelle interne



## **II – Sécurité et conformité dans les bases e données**

Assurer le fonctionnement continu et la prévention d'un Système d'Information n'est plus aujourd'hui considéré comme un simple exploit mais c'est une nécessité.

Parmi toutes les tâches qui incombent aux Responsables de la Sécurité des Systèmes d'Information (RSSI) dans les organismes privés ou publics, celle qui consiste à bâtir une politique de sécurité cohérente prenant en compte les aspects humains, organisationnels et juridiques est certainement la plus difficile. Une telle politique doit se baser sur une norme bien spécifique. En effet, il existe de nombreuses normes et méthodes sur lesquelles se basent les missions d'audit de la sécurité des systèmes informatiques.

Une norme (qui peut être organisationnelle ou technique) a un objet souvent très vaste et s'appuie généralement sur des concepts ou des notions générales. Le champ d'application de chaque concept doit alors être précisé, pour que la norme puisse être appliquée efficacement. [7]

### **II.1 – Conformité des bases de données : La loi Sarbanes Oxley et la norme COBIT**

- **Sarbanes Oxley :**

En décembre 2001, un événement important fait réagir les acteurs du marché de l'informatique : c'est le scandale Enron ainsi que ses conséquences, à savoir la loi Sarbanes Oxley ou encore les nouvelles règles comptables IAS (International Accounting Standards) et récemment baptisées IFRS (International Financial Reporting Standards). Désormais, les documents financiers doivent être faciles à retrouver et ils ne doivent pas pouvoir être détruits ou corrompus. [8]

De ce fait, la sécurité et la bonne maîtrise du système de gestion de base de données (SGBD) sont devenues indispensables. Car notons que celui-ci contient bien souvent toutes les informations financières ou/et de production d'une entreprise.

- **COBIT :**

La loi Sarbanes-Oxley Act fournit les lignes directrices sur l'élaboration d'un rapport financier. Toutefois, elle n'indique pas comment les appliquer, en particulier du point de vue de la technologie de l'information.

La norme COBIT (Control Objectives for Information and related Technology) est une structure standard de gouvernance et de gestion informatique, développé par l'ISACA (Information Systems Audit and Control Association. En tant que norme acceptée dans le monde entier pour les pratiques d'audit informatique, la COBIT a été choisie par de nombreuses sociétés d'audit pour la mise en conformité avec la loi SOX. [9]

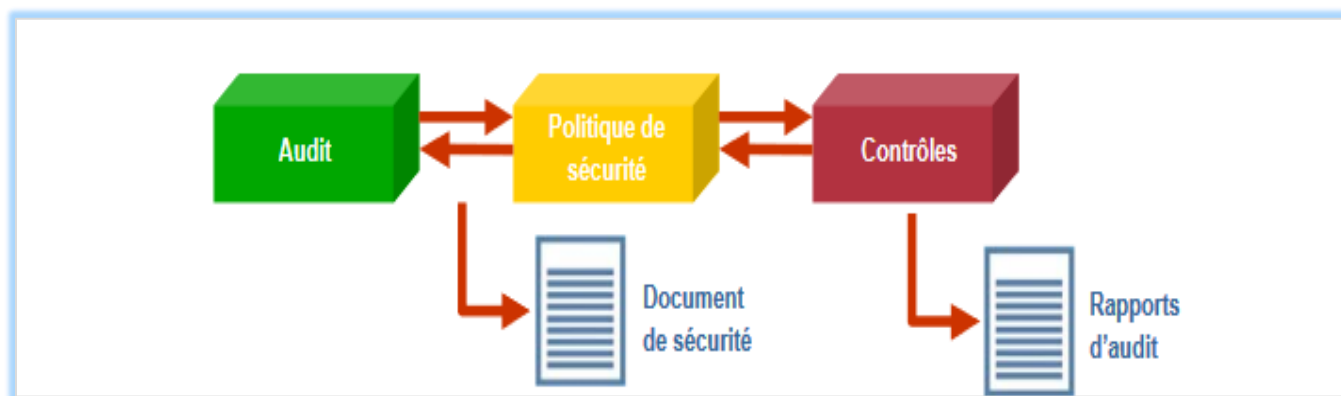


Figure 3 : La norme Cobit

## II.2 – Conformité des bases de données : La norme ISO 27001

### II.2.1 – Cadre d'application et contenu de la norme ISO 27001

#### A. Cadre d'application

La norme ISO 27001, publiée en Novembre 2005, pose le cadre du Management de la Sécurité des systèmes d'Information au sein d'une entreprise. Elle intègre les principes du management de l'ISO 9001 ainsi que le **PDCA** (Plan, Do, Check, Act du cycle de Deming) de l'amélioration continue. Ceci intègre des outils de contrôle de sécurité et de gestion de risque, en particulier les scanners de vulnérabilités. [10]

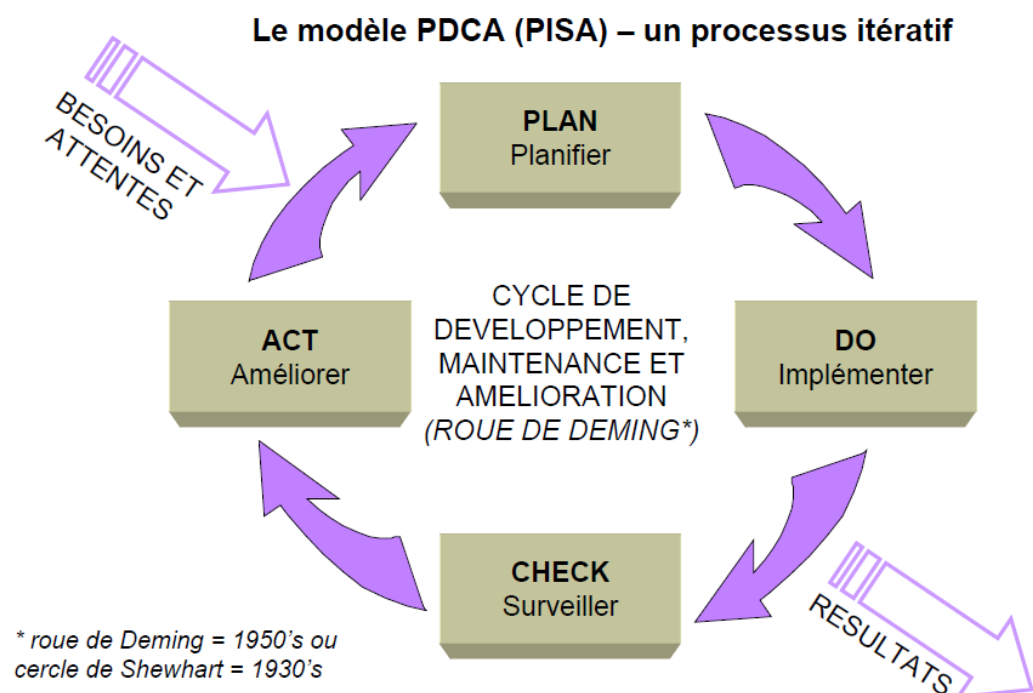


Figure 4: Le modèle PDCA

## **B. Contenu de la norme ISO 27001 :**

La norme ISO 27001 sera composée de 6 familles de processus, lesquelles sont :

1. Définir une politique de la sécurité des systèmes d'informations,
2. Définir le périmètre du Système de Management de la sécurité de l'information,
3. Réaliser une évaluation des risques liés à la sécurité,
4. Gérer les risques identifiés,
5. Choisir et mettre en œuvre les contrôles,
6. Préparer un SoA ("Statement of Applicability").

### ***II.2.2 - Treize (13) contrôles de sécurité efficace pour la conformité ISO 27001***

Gérer la sécurité d'un système d'information tout en respectant les obligations de conformité peut sembler difficile et complexe. Mais en réalité, cela peut être fait simplement avec quelques connaissances de base sur les exigences de sécurité de base implémentées.

Ces 13 contrôles de sécurité efficaces peuvent être facilement mis en œuvre pour aider à répondre aux obligations de conformité à la norme ISO 27001. Bien que nous reconnaissons qu'il est toujours nécessaire de prendre en compte tous les contrôles de la norme ISO 27001, nous nous concentrons sur plusieurs des problèmes rencontrés par la plupart des entreprises. [11] Ces contrôles fournissent des détails sur les principales recommandations suivantes :

1. Activer les solutions d'identité et d'authentification
2. Utilisez les contrôles d'accès appropriés
3. Implémenter et utiliser une solution antimalware recommandée par l'industrie
4. S'assurer qu'une solution efficace d'acquisition et de gestion de certificats est activée
5. Répondre au besoin de chiffrer toutes les données client
6. Examiner les tests de pénétration et les processus de modélisation des menaces
7. Consigner les événements de sécurité
8. Mettre en œuvre des capacités de surveillance et de visualisation pour les événements de sécurité
9. Être capable de déterminer la cause première des incidents
10. Former tout le personnel aux questions de cybersécurité
11. Corrigez tous les systèmes et assurez le déploiement des mises à jour de sécurité
12. Gardez l'inventaire des services et des serveurs actuels et à jour
13. Maintenir la configuration claire du serveur en gardant à l'esprit la sécurité

## II.3 – Sécurité et menaces dans les bases de données

### II.3.1 – Top 10 des menaces de sécurité des bases de données, et leurs préventions

L'infrastructure des bases de données d'une entreprise est sujette à un grand nombre de menaces dont nous pourrions énumérer 10.

Un sondage de l'équipe **@NCCGroupeInfosec** a permis d'évaluer la sécurité sur les systèmes de leurs clients, à la recherche de toute configuration ou absence de durcissement pouvant mettre le système en danger.

Le groupe de la CCN a analysé 20 rapports de révision de la base de données afin de déterminer les failles de sécurité les plus souvent négligées par les administrateurs du système. [12]

Le tableau ci-dessous montre les 10 problèmes les plus fréquemment découverts lors de l'évaluation par le groupe **@NCCGroupeInfosec**.

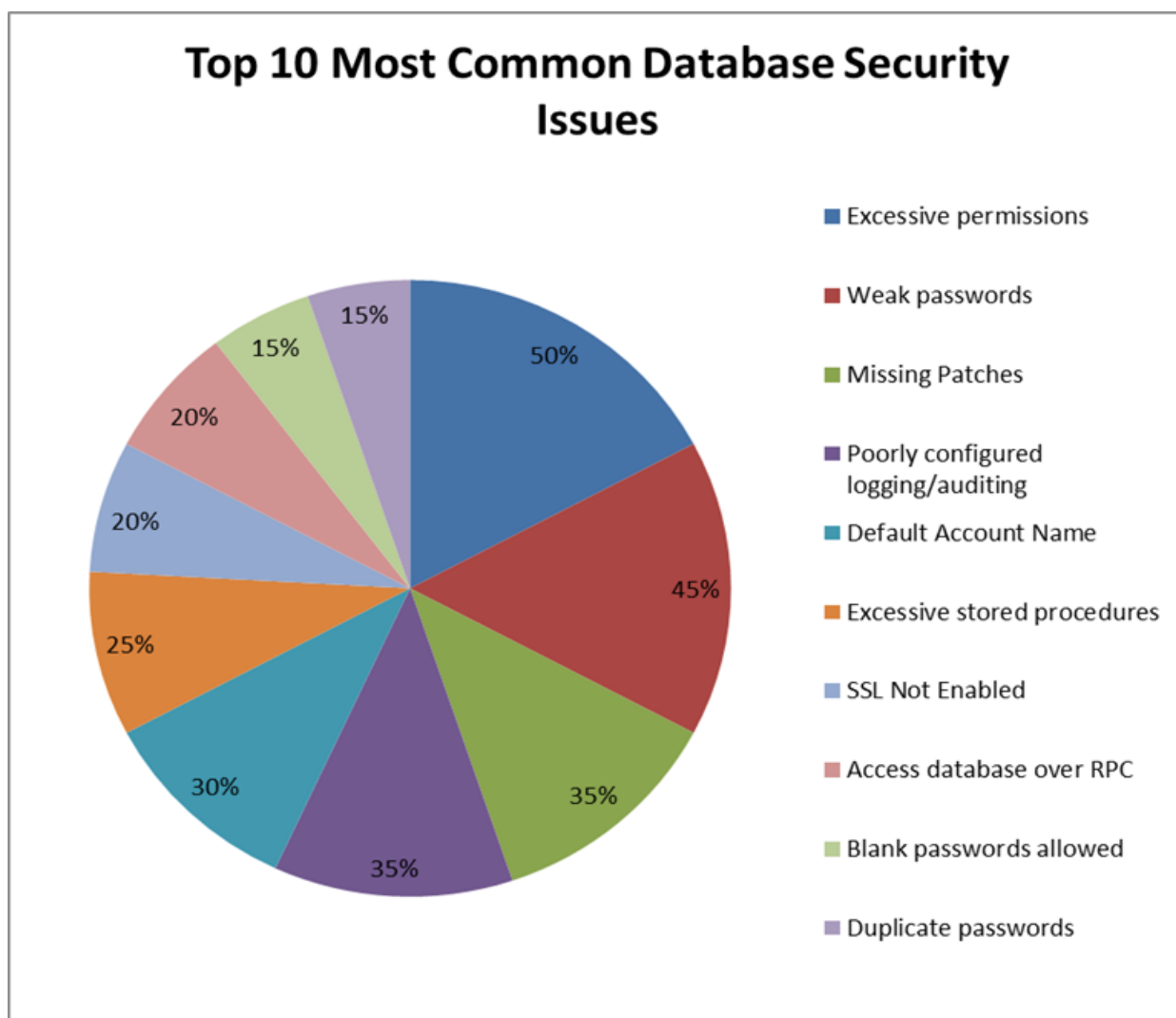


Figure 5: Top 10 des menaces

Par ailleurs, nous avons un autre sondage fait par l'équipe de l'entreprise **IMPERVA** qui propose le TOP 10 des menaces, suivant : [13]

1. Abus de privilège excessif
2. Abus de privilège légitime
3. Elévation de privilège
4. Exploitation de failles des bases de données vulnérables ou mal configurées
5. Injection SQL
6. Faiblesse de l'audit natif
7. Déni de service
8. Vulnérabilités des protocoles de communication des bases de données
9. Copies non autorisées de données sensibles
10. Exposition de données de sauvegarde

**Après avoir analyser ces deux études, nous pouvons retenir le TOP 10 suivant des menaces et leurs préventions :**

#### **Menace n° 1 – Abus de privilège excessif**

Lorsque les utilisateurs (ou les applications) ont des privilèges d'accès à une base de données excédant les exigences de leur fonction professionnelle, ils peuvent abuser de ces privilèges à des fins malveillantes.

#### **Prévention n° 1 – Abus de privilège excessif**

La solution à la menace présentée est l'élimination de tous droits excessifs. Ceci requiert la capacité à identifier les droits excessifs, c'est à dire les droits qui ne sont pas nécessaires à l'utilisateur pour remplir sa fonction. [13] Cela s'effectue par l'extraction des droits depuis les bases de données, la corrélation des droits avec l'utilisateur professionnel et enfin par l'analyse de ces droits. Une solution automatisée peut réduire considérablement le temps et les ressources nécessaires et raccourcir la procédure d'analyse.

#### **Menace n° 2 – Abus de privilège légitime**

Imaginons un éventuel fonctionnaire de la santé malveillant ayant des privilèges pour visualiser les dossiers médicaux des patients grâce à une application Web personnalisée. La structure de l'application Web limite normalement les privilèges des utilisateurs à la visualisation du dossier médical d'un seul patient. [13]

Cependant, le fonctionnaire malveillant peut contourner ces limitations en se connectant à la base de données en utilisant un client alternatif tel que MS-Excel. En utilisant MS-Excel et ses propres identifiants de connexion légitimes, le fonctionnaire peut récupérer et sauvegarder les dossiers médicaux des patients.

### **Prévention n° 2 – Abus de privilège légitime**

La solution à l'abus de privilège légitime est le contrôle d'accès aux bases de données qui s'applique non seulement aux requêtes d'accès spécifiques décrites précédemment, mais aussi au contexte d'accès aux bases de données. En appliquant une règle de contrôle pour les applications client, l'heure et la localisation de la requête d'accès, etc., il est possible d'identifier les utilisateurs qui utilisent des privilèges légitimes d'accès aux bases de données de manière suspecte.

### **Menace n° 3 – Elévation de privilège**

Les auteurs d'attaques peuvent profiter des vulnérabilités des logiciels plate-forme de bases de données pour transformer les privilèges d'accès d'un utilisateur ordinaire en ceux d'un administrateur. Les vulnérabilités peuvent se trouver dans les procédures enregistrées, les fonctions intégrées, les implémentations de protocoles, voire même dans les données SQL. Par exemple, un développeur de logiciels travaillant dans une institution financière peut profiter d'une fonction vulnérable pour s'attribuer des privilèges administrateur d'accès aux bases de données. Avec des privilèges administrateur, le développeur malveillant peut désactiver les mécanismes d'audit, créer des comptes fantômes, transférer des fonds, etc. [13]

### **Prévention n° 3 – Elévation de privilège**

Les abus d'élévation de privilège peuvent être empêchés en combinant un système traditionnel de prévention des intrusions (IPS) et un contrôle d'accès des requêtes (voir la section Abus de privilège excessif décrite précédemment). La technologie IPS inspecte le trafic des bases de données pour identifier les modèles qui correspondent aux vulnérabilités existantes.

### **Menace n° 4 – Failles des bases de données vulnérables ou mal configurées**

Les bases de données sont souvent vulnérables, non corrigées ou disposent de comptes et d'une configuration toujours définis par défaut. L'auteur d'une attaque qui tente d'exploiter la base de données test généralement les systèmes sur ces vulnérabilités, ce qui peut entraîner une violation de sécurité. Des paramètres de compte et de configuration toujours définis par défaut sur une base de données de production peuvent être exploités par l'auteur d'une attaque. [13]

### **Prévention n° 4 – Failles des bases de données vulnérables ou mal configurées**

Pour limiter le risque de menace des bases de données vulnérables, il faut tout d'abord évaluer l'état de sécurité des bases de données et corriger toutes les vulnérabilités et les écarts de sécurité identifiés. Les entreprises devraient effectuer un balayage périodique des bases de données pour découvrir toutes vulnérabilités et correctifs manquants.

### **Menace n° 5 – Protocole SSL/TLS non activé**

Une connexion à la base de données sans chiffrement permet à un tiers malveillant via une simple écoute sur le réseau de récupérer toute information qui surcule à travers le réseau.

### **Prévention n° 5 – Protocole SSL/TLS non activé**

Il faut ajouter au protocole TCP une couche SSL afin que les communications soient chiffrées. Ou encore utiliser un VPN.

### **Menace n° 6 – Mots de passe et comptes d'utilisateurs vulnérables**

Cette menace est très courante dans des nombreuses bases de données. Soit des noms de comptes et mots de passe sont laissés par défaut, soit la politique de création de comptes d'utilisateurs et mots de passe sont faibles.

### **Prévention n° 6 – Mots de passe et comptes d'utilisateurs vulnérables**

Ce qu'il faut faire ici c'est simplement désactiver les comptes et mots de passe par défaut, et créer une politique robuste pour les comptes d'utilisateurs et mots de passe.

### **Menace n° 7 – Déni de service**

Le déni de service (DOS - Denial Of Service) est une catégorie d'attaque générale par laquelle l'accès aux applications réseau est refusé à certains utilisateurs. Les conditions de déni de service peuvent être créées par de nombreuses techniques, parmi lesquelles beaucoup sont liées aux vulnérabilités mentionnées précédemment. Par exemple, le déni de service peut être obtenu en profitant de la vulnérabilité d'une plate-forme de bases de données pour faire tomber un serveur. [13]

### **Prévention n° 7 – Déni de service**

La prévention du déni de service requiert des protections à de multiples niveaux. Des protections au niveau du réseau, de l'application et de la base de données sont toutes nécessaires. Ce document met l'accent sur les protections spécifiques aux bases de données. Dans ce contexte spécifique aux bases de données, le déploiement du contrôle du taux de connexion, la technologie IPS, le contrôle d'accès des requêtes, et le contrôle du temps de réponse sont recommandés.

### **Menace n° 8 – Copies non autorisées de données sensibles (chiffrement)**

De nombreuses entreprises s'efforcent de localiser et maintenir de manière appropriée un inventaire de toutes leurs bases de données. De nouvelles bases de données peuvent être créées sans que l'équipe responsable de la sécurité soit au courant et les données sensibles copiées dans ces bases de données peuvent être exposées si les contrôles nécessaires ne sont pas appliqués.

### **Prévention n° 8 – Copies non autorisées de données sensibles**

Afin de maintenir un inventaire précis des bases de données et une localisation exacte des données sensibles, les organisations devraient identifier toutes les bases de données sur le réseau qui contient des données sensibles. La seconde étape consiste à trouver quels sont les types de données sensibles ou classifiées contenus dans les objets des bases de données. [13]

### **Menace n° 9 – Exposition de données de sauvegarde**

Les dispositifs de sauvegarde de bases de données auxiliaires ne sont généralement pas protégés contre d'éventuelles attaques. Par conséquent, plusieurs violations de sécurité importantes ont vu le jour, y compris le vol de disques durs et de bandes de sauvegarde de bases de données.

### **Prévention n° 9 – Exposition de données de sauvegarde**

Toutes les sauvegardes de bases de données devraient être chiffrées. En réalité, certains fournisseurs ont suggéré que les futurs systèmes de gestion de bases de données ne devraient pas prendre en charge la création de sauvegardes non chiffrées. [13] Le chiffrement des informations des bases de données de production en ligne est souvent suggéré, mais les performances et les inconvénients liés à la gestion des clés cryptographiques rend souvent cette solution peu pratique et sont généralement reconnus pour être un modeste substitut des contrôles de droits d'accès granulaires décrits précédemment.

### **Menace n° 10 – Accès à la base de données via RPC**

Il est possible que sécurité d'une base de données soit compromise à cause de l'accès à un services par un client via RPC (Remote Procedure Call). Elle permet à un attaquant de provoquer une exécution de code arbitraire et une élévation de privilèges.

### **Prévention n° 10 – Accès à la base de données via RPC**

Désactiver les RPC ou veiller à appliquer des correctifs de sécurité de la base.

### **Résumé :**

Il est tout à fait possible de réduire considérablement les risques liés à la sécurité d'une base de données en se basant sur les menaces les plus critiques puis mettre en place des politiques de prévention de risque.

Le Top 10 des menaces décrites dans ce document pourront aider les entreprises à mieux répondre aux exigences de conformité et de limitation des risques d'une industries réglementées.



### ***II.3.2 – Niveaux de sécurité et exigences de sécurité des bases de données***

Il est important de comprendre les différents niveaux de sécurité, ainsi que l'ensemble des exigences liées à la sécurité des bases de données.

#### **A. Niveaux de sécurité**

Pour protéger la base de données, nous devons prendre des mesures de sécurité à plusieurs niveaux :

**1. Au niveau des personnes :**

Les utilisateurs doivent être autorisés avec soin afin de réduire les chances qu'un tel utilisateur donne accès à un intrus contre un échange pour un bien ou d'autres faveurs.

**2. Au niveau du réseau et des applications clientes:**

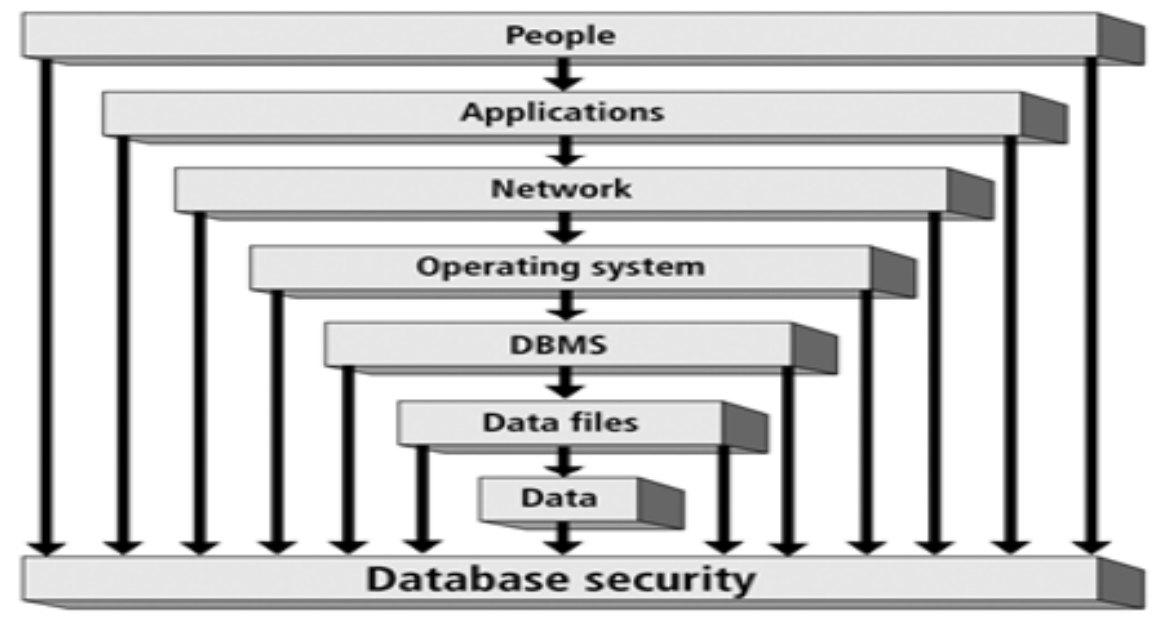
La quasi-totalité des systèmes de gestion de base de données permettent un accès à distance via des terminaux ou des logiciels clients. Ainsi, la sécurité au niveau des configurations réseau du logiciel client est aussi important que la sécurité physique de l'infrastructure réseau.

**3. Au niveau du Système d'exploitation :**

Peu importe à quel point le système de gestion de base de données est sécurisé, la faiblesse de la sécurité du système d'exploitation peut servir de moyen d'accès non autorisé à la base de données.

**4. Au niveau du système de gestion de base de données :**

Certains utilisateurs du système de gestion de base de données peuvent être autorisés à accéder seulement à une partie limitée de la base de données. D'autres utilisateurs peuvent être autorisés à émettre des requêtes, mais il peut être interdit de modifier les données. [14]



*Figure 6: Niveaux de sécurité*

## **B. Exigences de sécurité des bases de données**

Les exigences de sécurité des systèmes des bases de données ne sont pas différentes de ceux des autres systèmes informatiques. Les problèmes basiques de contrôle d'accès aux données, exclusion des données parasites, authentification des utilisateurs, et la fiabilité des données. [14]

### **1. L'intégrité physique de la base de données :**

Les données d'une base de données sont immunisées contre les problèmes physiques, tels que les pannes de courant, et quelqu'un peut reconstruire la base de données si elle est détruite par une catastrophe.

### **2. L'intégrité logique de la base de données :**

La structure de la base de données est conservée. Avec l'intégrité logique d'une base de données, une modification de la valeur d'un champ n'affecte pas d'autres champs.

### **3. La capacité d'audit :**

Il est possible de retracer qui ou quoi a accédé aux éléments de la base de données et quand cela a été fait.

### **4. Le contrôle d'accès :**

Un utilisateur est autorisé à accéder uniquement données qui lui sont autorisées, et différents utilisateurs peuvent être limités sur différents modes d'accès.

### **5. Authentification de l'utilisateur :**

Tout utilisateur est identifié positivement, à la fois pour la vérification et pour la permission d'accéder à certaines données.

### **6. Disponibilité :**

Les utilisateurs peuvent accéder à la base de données en général et à toutes les données pour lesquelles ils sont autorisés.

## **II.4 – Les scanners de vulnérabilités : Fonctionnement de base**

En sécurité informatique, un **scanner (ou scanneur) de vulnérabilités** est un programme conçu pour identifier des vulnérabilités dans une application, un système d'exploitation, ou un réseau.

Les scanners de vulnérabilités effectuent une analyse automatique des vulnérabilités de sécurité qui peuvent être utilisées pour contourner les contrôles de sécurité, pénétrer dans la base de données, compromettre le système, etc., avec des vulnérabilités connues dans le logiciel de base de données.

Les résultats de ces analyses sont ensuite utilisés pour améliorer les contrôles de sécurité de la base de données et fermer les vulnérabilités spécifiques identifiées.

Les scanners de vulnérabilités constituent donc un outil indispensable pour la gestion du risque au niveau des bases de données. [15]

## FONCTIONNEMENT DES SCANNERS DE VULNERABILITE

Le fonctionnement d'un scanner de vulnérabilité comprend trois (3) éléments bien précis : une **cible**, une **méthode de détection**, et la **restitution des résultats**.

### A. La Cible

Un scanner de vulnérabilités est théoriquement capable de tester tout élément joignable par une adresse IP :

- Ordinateur
- Serveur
- Routeur, commutateur, pare-feu
- Smartphone
- Objet connecté
- Site Web
- Automate, robot, machine
- Caméra IP
- IPBX, poste téléphonique sur IP
- etc.

### B. Méthodes de détection

Pour établir la présence d'une vulnérabilité, un scanner dispose de plusieurs méthodes :

- **Le Fingerprinting de version**

Dans cette méthode, l'outil tente de déterminer la nature et la version d'un service présent dans un système, par exemple : **Oracle 12.2.0.1.0**

- **L'exploitation active**

Lorsque des vulnérabilités sont publiquement dévoilées, elles sont parfois accompagnées d'un "**exploit**" qui est un programme permettant de les exploiter automatiquement. Un scanner de vulnérabilités peut donc recourir à cet exploit pour vérifier la présence d'une vulnérabilité. Dans ce cas-là, le scanner n'a pas besoin de se fier à la version du programme qu'il audit, il se fie à la réussite ou à l'échec de l'exploit. [15]

- **Le scan de configuration**

Certaines vulnérabilités ne proviennent pas d'un défaut dans le code source du programme mais simplement d'un mauvais réglage de celui-ci dans un certain contexte. Les scanners de vulnérabilités peuvent donc détecter certaines vulnérabilités en analysant la configuration distante du service audité.

Ceci concerne par exemple les options de sécurité activées sur les cookies, les cipher suite dans la configuration SSL/TLS, les transferts de zone pour un server DNS, les mots de passe par défaut inchangés, les services par défaut laissés activés, etc. [15]

- **Les scans authentifiés**

Bien que la plupart des scanners soient "sans agents", ils fournissent souvent la possibilité d'utiliser un compte renseigné par l'utilisateur pour mener des tests authentifiés.

L'intérêt principal de cette méthode est de ne pas se limiter à la surface visible de la machine depuis le réseau.

### C. Restitution des résultats :

1- La visualisation et la restitution des résultats se fait traditionnellement via deux paradigmes :

- Une vue "**par vulnérabilité**" permettant de lister toutes les vulnérabilités identifiées dans le scan et de donner pour chacune d'elle la liste des machines affectées.
- Une vue "**par machine**" listant les cibles de l'audit associées à la liste de leurs vulnérabilités respectives. [15]

2- Traditionnellement, en sécurité informatique, les vulnérabilités sont restituées par ordre de criticité suivant une échelle à 4 niveaux :

- Critiques : les vulnérabilités permettant généralement une prise de contrôle ou une exécution de commande à distance dont l'exploitation est relativement simple.
- Majeures : les vulnérabilités permettant une prise de contrôle ou une exécution de commande à distance dont l'exploitation demeure complexe.
- Moyennes : les vulnérabilités ayant des impacts limités ou dont l'exploitation nécessite des conditions initiales non triviales.
- Mineures : les vulnérabilités ayant des impacts faibles ou nuls à moins d'être combinées à d'autres vulnérabilités plus importantes. [15]

**3-** L'état de l'art associe à chaque vulnérabilité un score entre 0 et 10 appelé CVSS (*Common Vulnerability Scoring System*) qui dépend des caractéristiques de cette vulnérabilité. La version 3 du CVSS prend en compte, au minimum, les éléments suivants :

- Le vecteur d'attaque : est-ce que l'attaquant peut venir de n'importe où ou bien doit-il avoir une position de départ privilégiée ?
- Complexité : exploiter la vulnérabilité est-il trivial (par exemple si un exploit existe) ou bien hautement techniques ?
- Privilèges requis : l'attaquant doit-il disposer d'accès préalables (un compte utilisateur par exemple) pour pouvoir mener son action ?
- Interaction avec un utilisateur : l'attaquant doit-il amener la victime à effectuer une action pour que son attaque réussisse (comme l'inciter à cliquer sur un lien) ?
- Périmètre : est-ce qu'une exploitation permet à l'attaquant d'avoir accès à de nouvelles cibles ?
- Impacts : une exploitation réussie entraîne-t-elle des pertes de confidentialité/disponibilité/intégrité ? [15]

**4-** Le score CVSS est régulièrement utilisé par les scanners de vulnérabilités pour pondérer les risques associés à une cible.

Les scanners doivent donner à l'utilisateur tous les éléments pertinents pour sa compréhension de la vulnérabilité. Traditionnellement, une description de vulnérabilité comporte les éléments suivants : [15]

- Le nom de la vulnérabilité
- Sa criticité
- La cible touchée
- Une brève description de sa nature
- Une référence à une base de connaissance type CVE, OSVDB, DSA...
- Une mention de la simplicité de l'exploitation
- Une description de l'impact en cas d'exploitation réussie
- Une ou plusieurs préconisations pour la résoudre

Parfois, d'autres éléments y sont ajoutés :

- Le niveau de confiance à accorder à la vulnérabilité : quantification du risque qu'il s'agisse ou non d'un faux positif
- S'il existe ou non un exploit automatique
- Un extrait des données ayant permis au module de sécurité de conclure à la présence de cette faille
- La famille de vulnérabilité (authentification, mise à jour, etc.)
- Des liens pour en savoir plus (notamment pour des explications plus détaillées du fonctionnement technique de la vulnérabilité)

Les rapports sont souvent exportables aux formats PDF, CSV, HTML, etc.

## Exemple de restitution de résultat

### Security Testing Dashboards

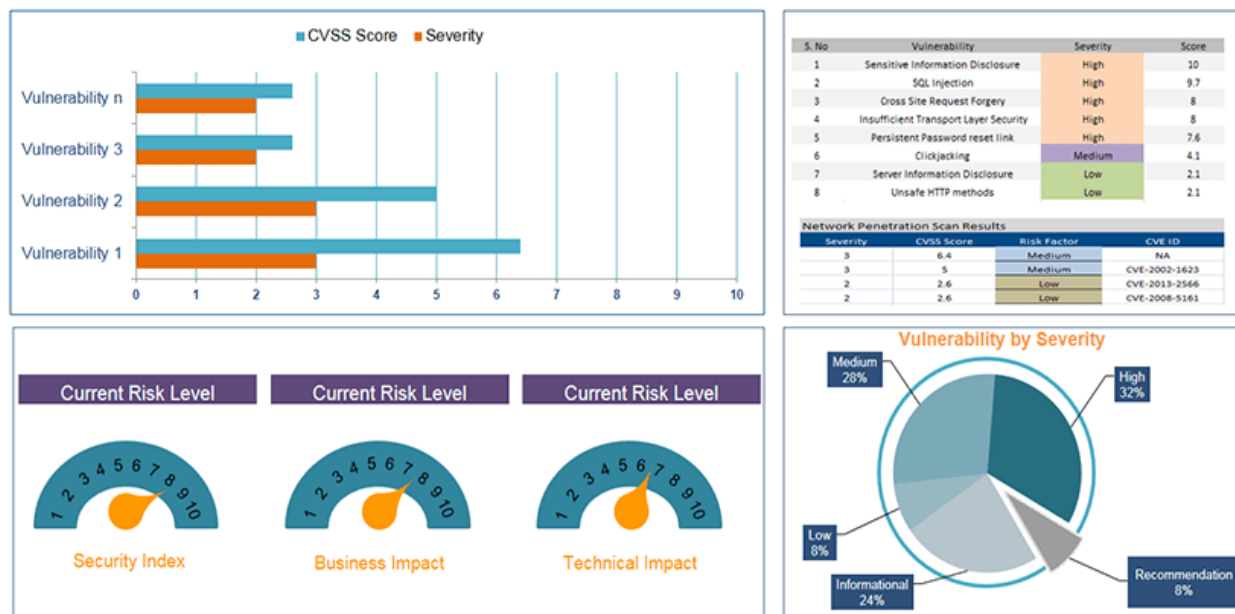


Figure 7: Restitution des résultats

**PARTIE II :**  
**Étude, conception et développement de la solution**

### III – Étude et conception

#### III.1 – Critères recherchés du scanner et état de l’art sur les solutions existantes

##### A. Les principaux critères recherchés pour le scanner de vulnérabilités

Les principaux critères que nous recherchons afin d’obtenir une solution capable de répondre à de nombreux besoins des entreprises en matière d’outil de scan de vulnérabilités de base de données, sont les suivants :

- ✓ Multiplateforme
- ✓ Prise en charge de plusieurs SGBD les plus utiliser
- ✓ Flexibilité (possibilité de choisir les menaces à auditer selon les besoins)
- ✓ Extensibilité (Côté développeurs et côté utilisateurs)
- ✓ Gestion de la mobilité (possibilité de faire un envoi du rapport via mail)

##### B. État de l’art sur les solutions existantes

Après des heures de recherche sur le web, nous constatons que curieusement il y a très peu de solutions concernant les scanners de vulnérabilité de base de données (Oracle DB, Microsoft SQL Server, MySQL, etc.).

Notre étude reposera principalement les outils **Scuba** et **NCC Squirrel Suite**. Il en existe d’autres mais qui ne tiennent compte que de la base de données Oracle. Ces derniers sont donc moins intéressants pour notre étude.

##### ▪ La solution **Scuba Database Vulnerability Scanner** de IMPERVA

**Scuba** est un outil qui permet d’analyser une base de données à la recherche de failles de sécurité et de configuration, y compris des nouveaux correctifs de sécurité. [18]

- **Avantages**
  - Possède une version Gratuite
  - Multiplateforme (Windows, Mac, Linux).
  - Multiples SGBD : Oracle, Microsoft SQL, SAP Sybase, IBM DB2 et MySQL.
  - **Scuba** propose des correctifs pour certaines menaces.
- **Limites de la solution Scuba**
  - Ne gère pas la menace sur le Protocole de communication (SSL/TLS non activé).
  - Ne gère pas la menace sur les données sensibles non chiffrées.
  - Manque de flexibilité.
  - Manque d’extensibilité.
  - Ne gère pas la mobilité de l’utilisateur.



## ▪ La solution Squirrel Suite de NCCGroupeInfosec

**Squirrel** est une gamme de logiciels payant pour garantir la sécurité des bases de données.

### ○ Avantages

- Prend en charge : Oracle, Microsoft SQL, SAP Sybase, IBM DB2 et MySQL.
- Prend en charge les bases de données NoSQL : MongoDB et PostgreSQL.
- Option pour changer la configuration de toutes les vérifications effectuées.
- Vérifie les informations sensibles non chiffrées.
- Reporting comparatif (confirmation des problèmes résolus et alertes sur les nouvelles menaces). [19]

### ○ Limites de la solution Squirrel

- N'est pas multiplateforme (Disponible seulement sur Windows).
- Ne gère pas la menace sur le Protocole de communication (SSL/TLS non activé).
- Ne gère pas la menace sur l'exposition de données de sauvegarde.
- Manque de flexibilité.
- Manque d'extensibilité.
- Ne gère pas la mobilité de l'utilisateur.
















## ▪ Conclusion sur Etat de l'art de l'existant :

Bien que notre étude fût portée essentiellement sur les solutions **Scuba** et **Squirrel** car elles sont les seules qui s'inscrivent dans la liste des outils comparatifs, nous avons aussi étudié le fonctionnement des outils existant suivants : **ODAT**, **Oracle Auditing Tools**, **Omega DB Scanner**, **Oscanner**, **SQL RECON**. Toutefois, hormis différentes limites que l'on peut retrouver sur chacune de ces solutions, un même problème revient, c'est la **flexibilité**.

En effet, un scanner de vulnérabilité de base de données doit avoir pour but auditer la base de données. Cependant l'audit doit être ciblé selon les besoins des responsables de la sécurité des bases de données. La solution que nous présentons a pour but de répondre à ces différents besoins.

## Tableau récapitulatif :

Tableau 2: Critères recherchés dans le scanner

	Multiples SGBD	Multiplateforme	Flexibilité	Extensibilité	Mobilité
Scuba					
Squirrel					
Objectifs de notre solution					

## **III.2 – Analyse des besoins**

### **A. Contexte et définition du problème**

Les responsables de la sécurité des bases de données sont chargés d'auditer les bases de données de leurs entreprises pour la sécurité et la conformité de celles-ci, mais ceux qui lisent simplement le manuel sur la façon d'activer l'audit seront désappointés. Les outils d'audit de base de données ont leurs caprices, et chacun d'eux peut effectuer un audit sans même prendre le temps de planifier de manière appropriée le processus d'audit. Il n'est pas inhabituel pour l'audit ou le traçage de causer plus de 50% de dégradation sur les performances de la base de données. Cela signifie que l'audit apparemment simple finira par ralentir la base de données, et créer des problèmes pour vous-même dans la maintenance et la génération de rapports.

A cet effet, la conception et le développement d'un scanner de vulnérabilité qui pourra être un excellent outil d'audit des bases de données est la tâche attendue.

### **B. Objectif**

L'objectif principal est d'apporter une approche un peu plus révolutionnaire pour la conception et le développement d'un scanner de vulnérabilité. Il est question de mettre au point un outil complet et flexible, qui donnera aux ingénieurs de sécurité des bases de données la possibilité de cibler un audit précis à effectuer sur sa base de données. Par exemple, l'ingénieur doit pouvoir, s'il le souhaite, auditer uniquement les comptes d'utilisateurs et mot de passe, ou bien auditer uniquement les privilèges des utilisateurs dans la base, etc. Ainsi, l'ingénieur pourra prendre le temps de planifier selon sa convenance le processus d'audit. De nombreuses limites constatées dans les outils existants nous ont emmenés à considérer l'importance de ce travail, en vue d'atteindre un taux de satisfaction au-delà des 80%.

Les langages UML et Python seront utilisés respectivement pour la conception et le développement du scanner de vulnérabilité.

### **C. Périmètre**

Pour le cadre du mémoire, le scanner de vulnérabilité prendra en compte dans un premier moment des bases de données SQL. Le cas des bases de données NoSQL sera pris en compte ultérieurement.

### **D. Description fonctionnelle :**

#### **1. Avant le lancement du scan, l'outil doit :**

- a. Permettre à l'utilisateur de choisir le type de BD qu'il souhaite scanner.
- b. Permettre à l'utilisateur d'entrer les informations de connexion à la BD.
- c. Ensuite permettre à l'utilisateur de choisir les menaces qu'il souhaite auditer.
- d. Permettre à l'utilisateur d'effectuer des tests de pénétration de la BD cible.

**2. Après le lancement du scan, l'outil doit pouvoir effectuer principalement de façon indépendantes les tâches suivantes :**

- a. Auditer les performances de la DB (espace disque, RAM, Processeur, etc.).
- b. Auditer les protocoles de communication (Chiffrement des communications).
- c. Auditer les mauvaises configurations ou configurations par défaut de la BD.
- d. Auditer les attaques par force brute ou Déni de services de la base de données.
- e. Auditer les comptes et mots de passe d'un ou plusieurs utilisateurs.
- f. Auditer les privilèges des utilisateurs (Abus, excès, élévation de privilèges.
- g. Auditer l'exposition des données (Chiffrement des données sensible).
- h. Evaluer les faiblesses de l'audit natif de la base de données.
- i. Evaluer les logs du système de gestion de base de données.

**3. Puis après avoir effectué le scan, l'outil doit pouvoir :**

- a. Permettre l'affichage des résultats sous différents formats (HTML par défaut).
- b. Permettre l'envoi des résultats par mail, à une adresse quelconque.
- c. Permettre à l'utilisateur d'envoyer des suggestions des résultats d'audits qu'il souhaite avoir afin de permettre la maintenabilité de l'outil.

## **E. Enveloppe budgétaire**

Les ressources qui doivent être mobiliser sont :

- 1. Un cadre approprié pour les multiples tests et simulations.
- 2. Une connexion internet à débit normal pour les recherches.
- 3. Un travail en équipe serait meilleur pour des échanges d'idées.
- 4. Plusieurs ordinateurs performants pour installer différentes bases de données permettant ainsi de simuler plusieurs niveaux de configurations allant de la moins sécurisée à la plus sécurisée.

### III.3 – Présentation des outils de modélisation

Pour la modélisation nous avons opté pour le langage **UML** (*Unified Modeling Language*) et comme support nous allons utiliser le logiciel **PowerAMC**.

#### A. UML

Le **langage de modélisation unifié**, de l'anglais *Unified Modeling Language* (**UML**), est un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet. [20]

#### B. PowerAMC

**PowerDesigner** (anciennement **PowerAMC**) est un logiciel de conception créé par la société *SAP*, qui permet de modéliser les traitements informatiques et leurs bases de données associées. Il a été créé par SDP sous le nom *AMC\*Designor*, racheté par Powersoft qui lui-même a été racheté par Sybase en 1995. Depuis 2010 Sybase appartient à l'éditeur allemand *SAP*. [21]

### III.4 – Modélisation

Dans cette section nous présentons les différents diagrammes mis en jeu pour la conception du scanner de vulnérabilités avec **UML** (*Unified Modeling Language*). Nous suivrons un modèle de conception orientée objet.

#### A. Spécifications du modèle :

Le modèle de conception de notre scanner de vulnérabilités possède une architecture assez simple, car la quasi-totalité des tâches et des interactions seront effectuées par ce que nous appellerons des "**agents**" du programme principal mais non pas par les utilisateurs du programme. Ainsi, vu la simplicité sur le plan architectural du programme nous aurons à présenter trois diagrammes nous permettant de comprendre la conception de notre système :

##### 1- Le diagramme des cas d'utilisations

Celui-ci permettra d'illustrer les différents cas d'utilisation que les utilisateurs du système pourront avoir.

##### 2- Le diagramme de séquences

Ce diagramme va nous permettre de voir les différentes séquences qui se produiront entre les entités principales.

##### 3- Le diagramme de classes

Ce dernier diagramme présentera les différentes classes qui interagissent dans le programme principal.

▪ **Diagramme des cas d'utilisations :**

Dans ce diagramme nous voyons les cinq cas d'utilisation que nous pouvons avoir :

- 1- L'utilisateur peut **se connecter** en entrant les informations de connexion.
- 2- L'utilisateur peut **sélectionner les audits** (des tâches) qu'il souhaite effectuer.
- 3- L'utilisateur peut **exécuter le scan**.
- 4- L'utilisateur peut **afficher le rapport de scan** selon le type de fichier qu'il veut.
- 5- L'utilisateur a également la possibilité de demander **l'envoi des résultats par mail**.

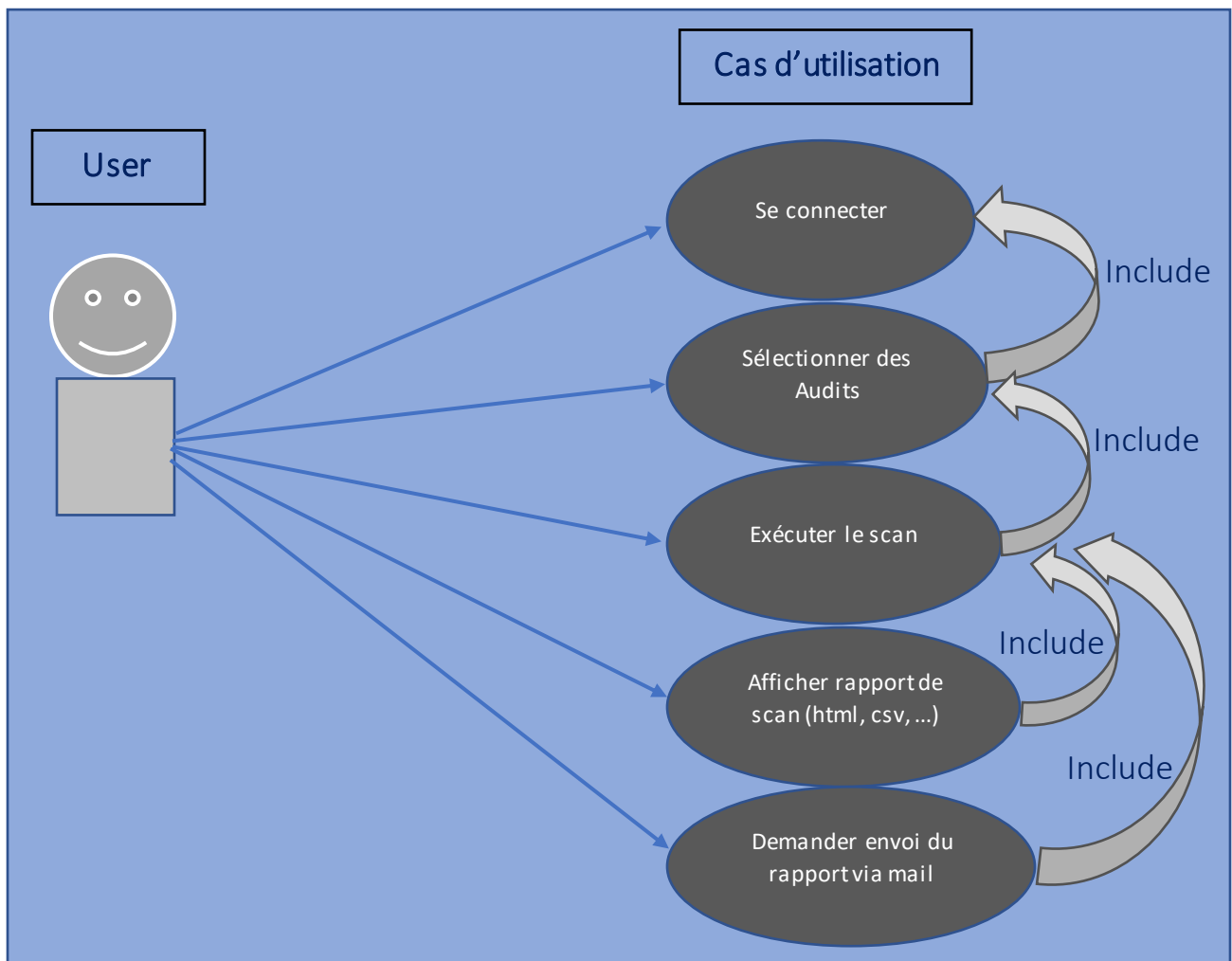


Figure 8: Diagramme des cas d'utilisations

▪ **Diagramme des séquences :**

Dans ce diagramme on peut distinguer 4 entités :

- 1- L'utilisateur qui interagit via l'interface graphique du scanner
- 2- Le scanner (Moteur du scanner)
- 3- La base de données du scanner (SQLite)
- 4- Le SGBD cible (qui doit être scanner)

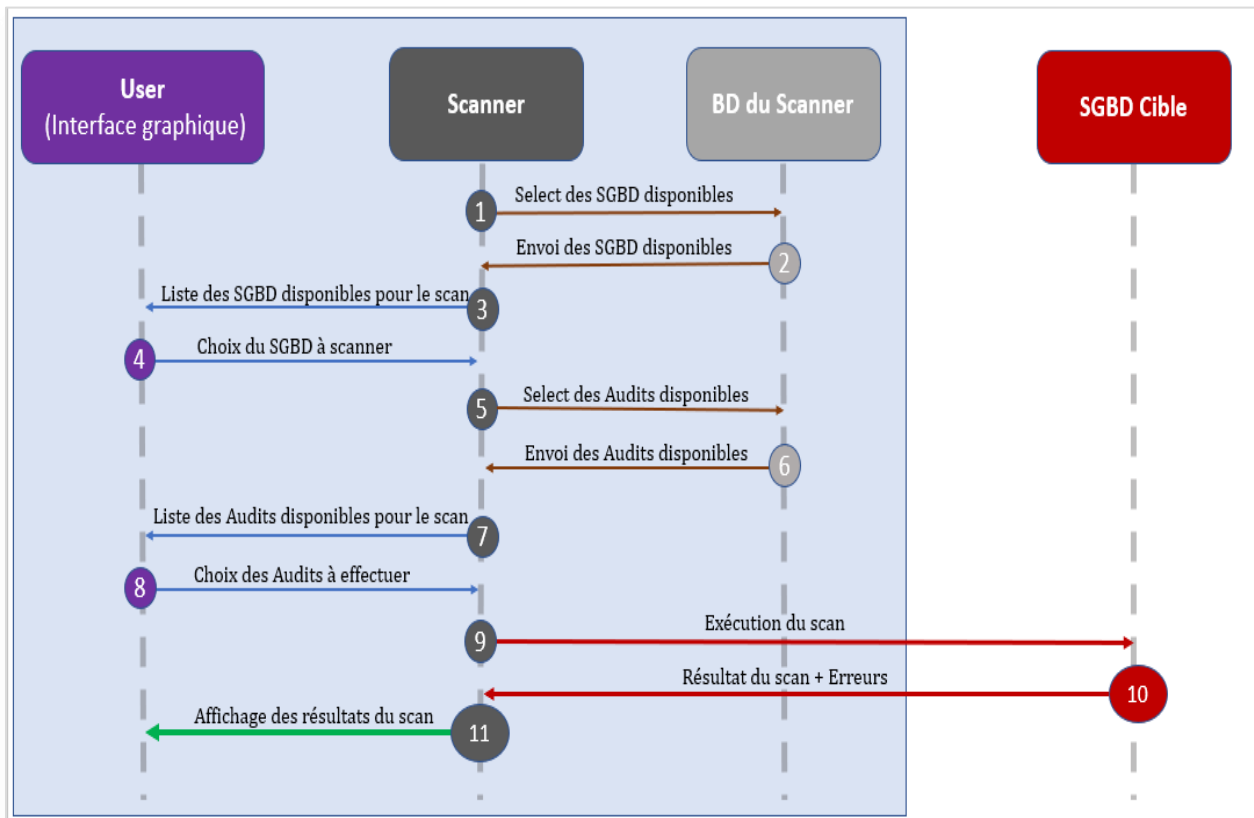


Figure 9: Diagramme des séquences

Résumé des 11 séquences se produisant entre les 4 entités présentées :

1. Le scanner consulte sa base de données pour savoir les SGBD pris en compte
2. La base de données du scanner fournit les SGBD pris en compte
3. Le scanner présente sur l'interface de l'utilisateur les SGBD pris en compte
4. L'utilisateur fait le choix du SGBD
5. Le scanner consulte sa base pour savoir les audits relatifs au SGBD choisit
6. La base de données du scanner fournit tous les audits disponibles
7. Le scanner présente sur l'interface de l'utilisateur les audits nécessaires
8. L'utilisateur sélectionne les audits qu'il souhaite effectuer
9. Le scanner exécute le scan sur la base de données cible (SGBD)
10. La base de données cible fournit des résultats
11. Le scanner interprète et formate les résultats, puis les affiche à l'utilisateur

## ▪ Diagramme de classes

Ce diagramme nous présente uniquement 4 classes, ces classes sont nécessaires.

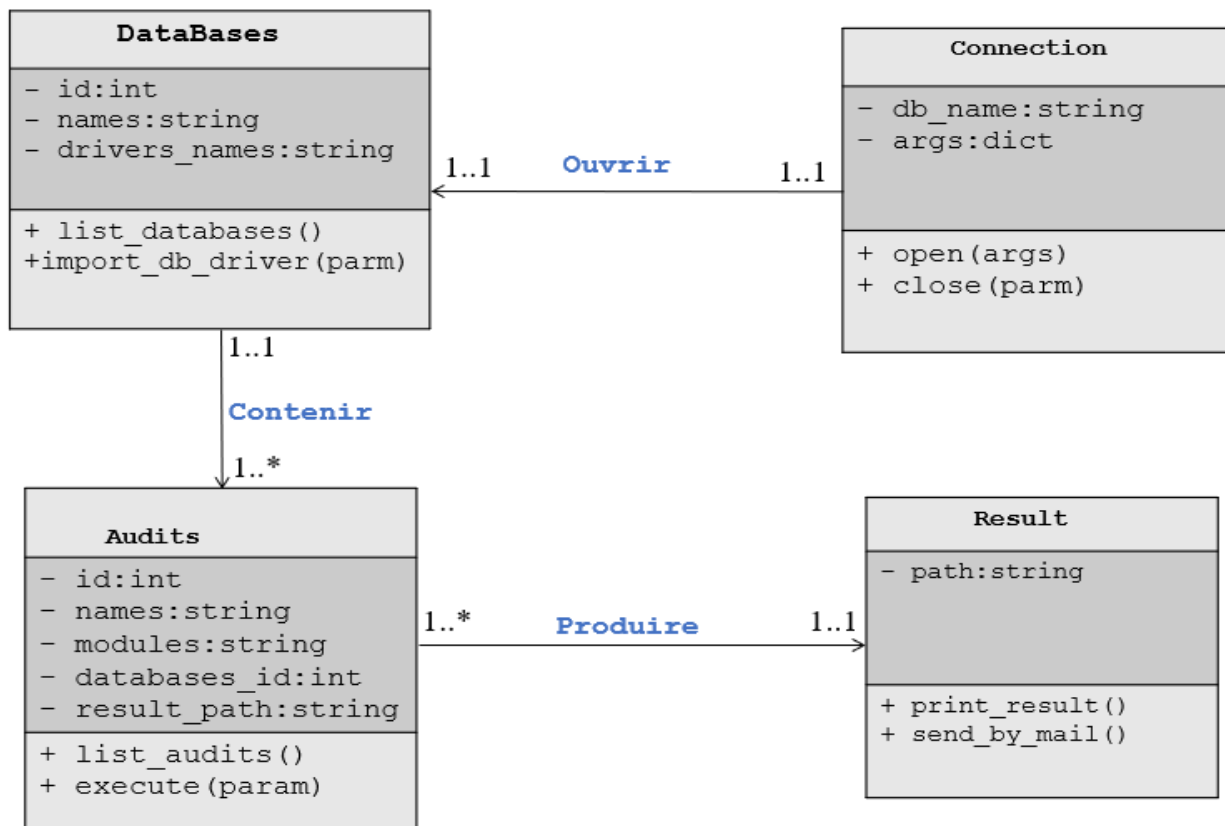


Figure 10 : Diagramme de classes

Les différentes classes sont les suivantes :

### 1- DataBase

Caractérisée par 3 attributs (Id, names, drivers\_names), et possède 2 méthodes (list\_databases(), et import\_db\_driver()).

### 2- Audits

Caractérisée par 5 attributs (id, names, modules, database\_id, result\_path) et constitue le point central du système. Elle possède 2 méthodes (list\_audits(), et execute()).

### 3- Connection

Caractérisée par 2 attributs (db\_name, args) et possède 2 méthodes (open(), close()).

### 4- Result

Caractérisée par 1 attribut (path) et possède 2 méthodes (print\_result(), send\_by\_mail()).

## IV – Développement de la solution

### IV.1 – Choix des outils technologiques

Plusieurs outils auraient pu être choisis pour le développement de ce projet mais tous n'ont pas le même niveau d'apport ou le même niveau de réalisation. Les différents outils technologiques que nous avons choisis répondent mieux aux exigences du scanner de vulnérabilité qu'on souhaite développer. Ces outils sont les suivants :

#### A. L'IDE JetBrains PyCharm Community Edition 2017.3.1

**PyCharm Community Edition** est un environnement de programmation léger et open source dédiée au développement de programmes en Python uniquement. Selon un sondage du site [Developpez.com](http://Developpez.com), plus de 33,33% de développeurs Python choisit PyCharm, certainement pour ses fonctionnalités multiples et son niveau d'intuitivité. De plus PyCharm a l'avantage d'être multiplateformes (Windows, Linux, Mac). La popularité de PyCharm peut être aussi évaluée à partir du fait que de grandes marques comme Twitter, Groupon, Spotify, eBay et Telefonica etc. l'ont également utilisé. Les nombreux avantages de PyCharm ont fait de ce IDE un outil de choix pour ce projet. [16]

#### B. Le langage Python version 3.6

Selon l'Institute of Electrical and Electronics Engineers (IEEE), la plus grande association mondiale de professionnels techniques **Python** est en tête du top 10 des meilleurs langages de programmation en 2017 devant C, Java et C ++. Il est utilisé dans de nombreuses applications et programmes comme Dropbox YouTube (Google) et Instagram, tandis que la NASA, PBS et Reddit l'utilisent sur leurs sites Web. Il est idéal pour l'écriture de scripts et possède une utilisation intensive dans l'industrie de la sécurité de l'information : Exécution des tâches d'administration système, Analyse de logiciels malveillants, Analyse Forensique, etc. [17]

En outre, Python possède les avantages suivants :

1. Optimisé pour la sécurité et les tests d'intrusions
2. Rapidité de développement
3. Portabilité du langage (Windows, Mac, Unix, Linux, Android, ...)
4. Extensibilité du langage
5. Très dynamique
6. Orienté objet

Face à ces multiples avantages, le langage Python présente un intérêt majeur pour le développement d'un scanner de vulnérabilités.



### C. Le moteur de base de données SQLite

**SQLite** en tant que base de données embarquée permettra de stocker les informations dont le scanner de vulnérabilités aura besoin pour effectuer un scan. Le moteur **SQLite** est par défaut intégré à Python. [22]

Contrairement aux serveurs de bases de données traditionnels, comme **MySQL** ou **PostgreSQL**, sa particularité est de ne pas reproduire le schéma habituel client-serveur mais d'être directement intégrée aux programmes. L'intégralité de la base de données (déclarations, tables, index et données) est stockée dans un fichier indépendant de la plateforme. **SQLite** est le moteur de base de données le plus utilisé au monde, grâce à son utilisation dans de nombreux logiciels grand public comme **Firefox**, **Skype**, **Google**, etc.

### D. La plateforme Windows

Enfin la plateforme Windows est celle sur laquelle nous allons déployer notre scanner de vulnérabilités.

Le choix du système d'exploitation Windows est dû à plusieurs raisons, lesquelles sont :

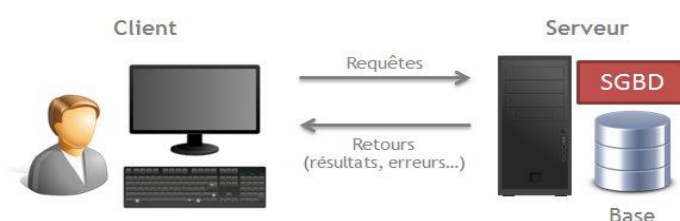
- 1- Constitue la plateforme la plus utilisée par les entreprises
- 2- Accepte facilement les systèmes de virtualisation tel que VMWare pour de nombreux tests.
- 3- Windows permet d'intégrer l'outil Intel VTune Amplifier pour l'optimisation du code Python.
- 4- Windows constitue un environnement idéal pour effectuer des tests de manière un peu plus flexible par rapport à d'autres environnements.

## IV.2 – Principe de fonctionnement de base de notre scanner de vulnérabilités

Comme tout scanner de vulnérabilités, le fonctionnement de base de cette solution comprend trois (3) éléments : **Une cible, la méthode de détection, et la restitution des résultats.**

### A. La Cible

Tout serveur hébergeant la base de données, et joignable via une adresse IP, ou en local constituera la cible. Le scanner sera alors comme un client et la cible comme serveur.



## B. Méthode de détection

Pour établir la présence d'une vulnérabilité dans la base de données, nous avons choisi une méthode de détection hybride qui consiste à en associer trois :

1. **L'exploitation active** pour des vulnérabilités publiquement dévoilées et accompagnées d'un "exploit".
2. **Le scan de configuration** afin de détecter certaines vulnérabilités en analysant la configuration distante du service audité, exemple SSL/TLS.
3. **Le scan authentifié** qui consistera à interroger directement les tables et vues de la base de données.

## C. Restitution des résultats

Nous utiliserons le paradigme de la vue "par vulnérabilité" et par ordre de criticité suivant une échelle à 4 niveaux : **critique, majeure, moyenne, mineure**. Et à chaque vulnérabilité sera associé un score entre **0** et **10**.

**Exemple** : Restitution des résultats

### Security Testing Dashboards

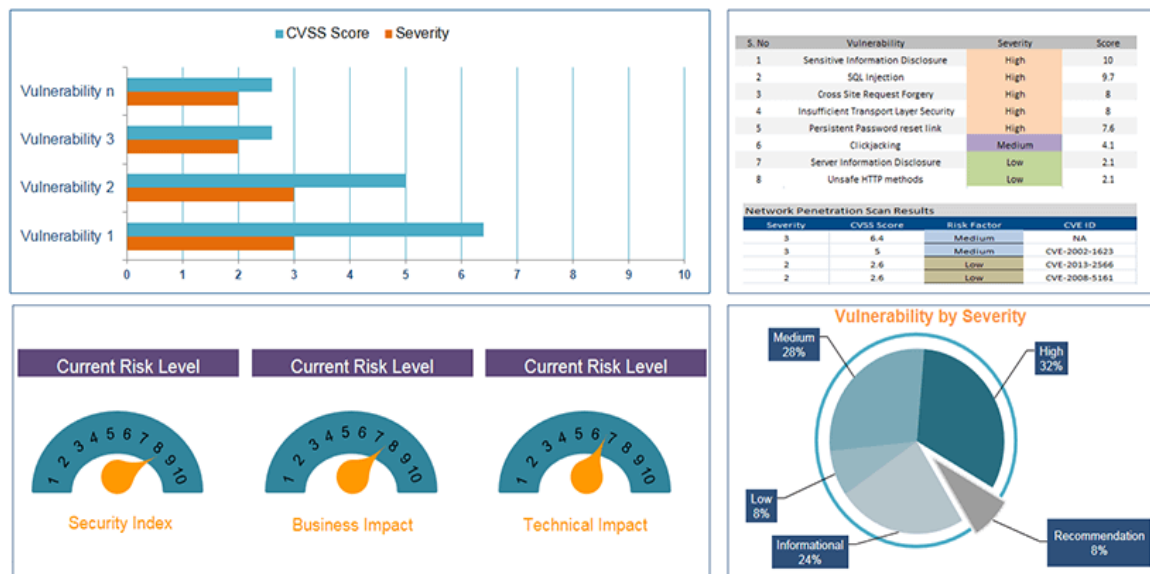


Figure 11: Restitution des résultats

Plusieurs formats seront disponibles pour l'affichage des résultats, (HTML, CSV, PDF, etc.) selon les besoins de l'utilisateur. Toutefois résultats seront affichés par défaut les sous forme de page HTML.

### IV.3 – Architectures, fonctionnement détaillé et sécurité de la solution

Dans cette section nous présenterons un plus en profondeur le fonctionnement de l'outil. Tout cela sera illustré par les différentes architectures de fonctionnement interne, et du réseau du système.

#### A. Architecture fonctionnelle

Du point de vue interne, le système peut être découpé en trois parties :

##### 1- Le noyau du programme :

Il s'agit du noyau de traitement des informations, Il va chercher des informations dans la base de données SQLite et importe les modules Python nécessaires pour le traitement d'une tâche lors du scan.

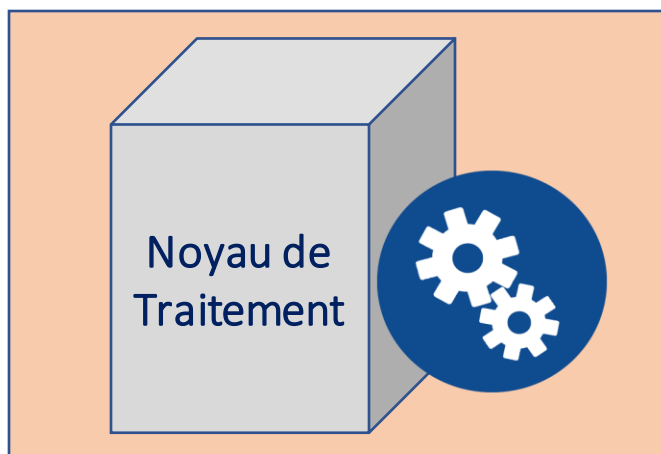


Figure 12: Centre de traitement

##### 2- La base de données SQLite :

Gérée par le moteur de base de données SQLite intégré à Python, elle stock toutes les informations des objets qui sont manipulés par le programme.

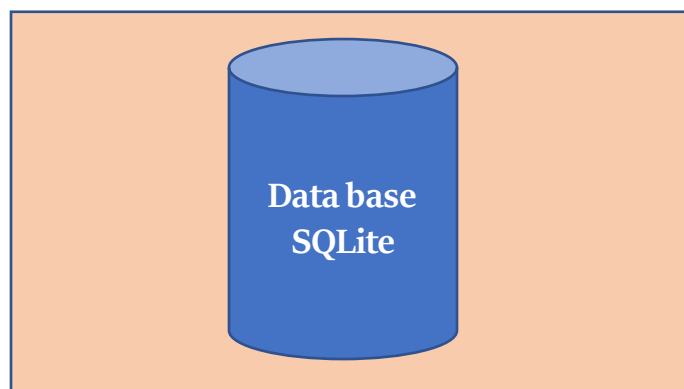


Figure 13: SQLite DB

### 3- La base de données des modules de traitement

Regroupés sous forme de packages (**oracle**, **mssql**, **mysql**, etc.), ces modules sont comme des agents qui reçoivent les ordres du centre de traitement. En réalité ce sont ces agents qui font le vrai boulot. Il s'agit de fichiers **.py** qui contiennent des scripts pour la réalisation d'une tâche précise.

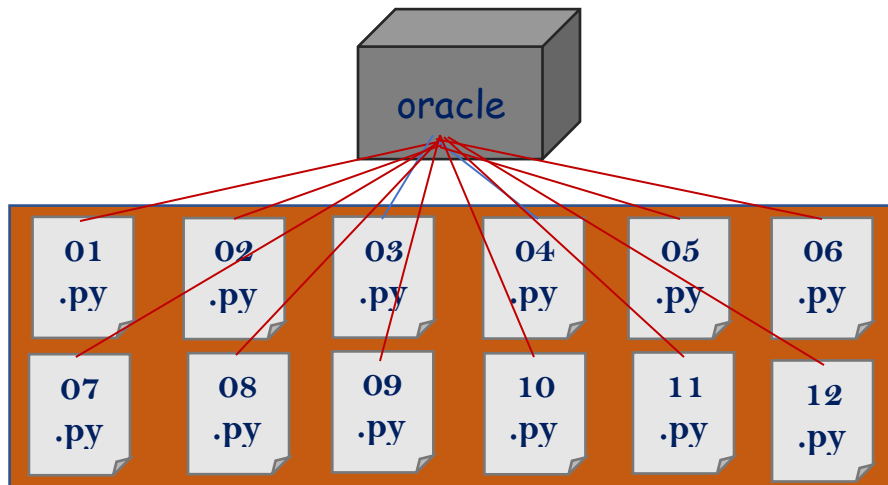


Figure 14: Modules

### B. Architecture interne complète

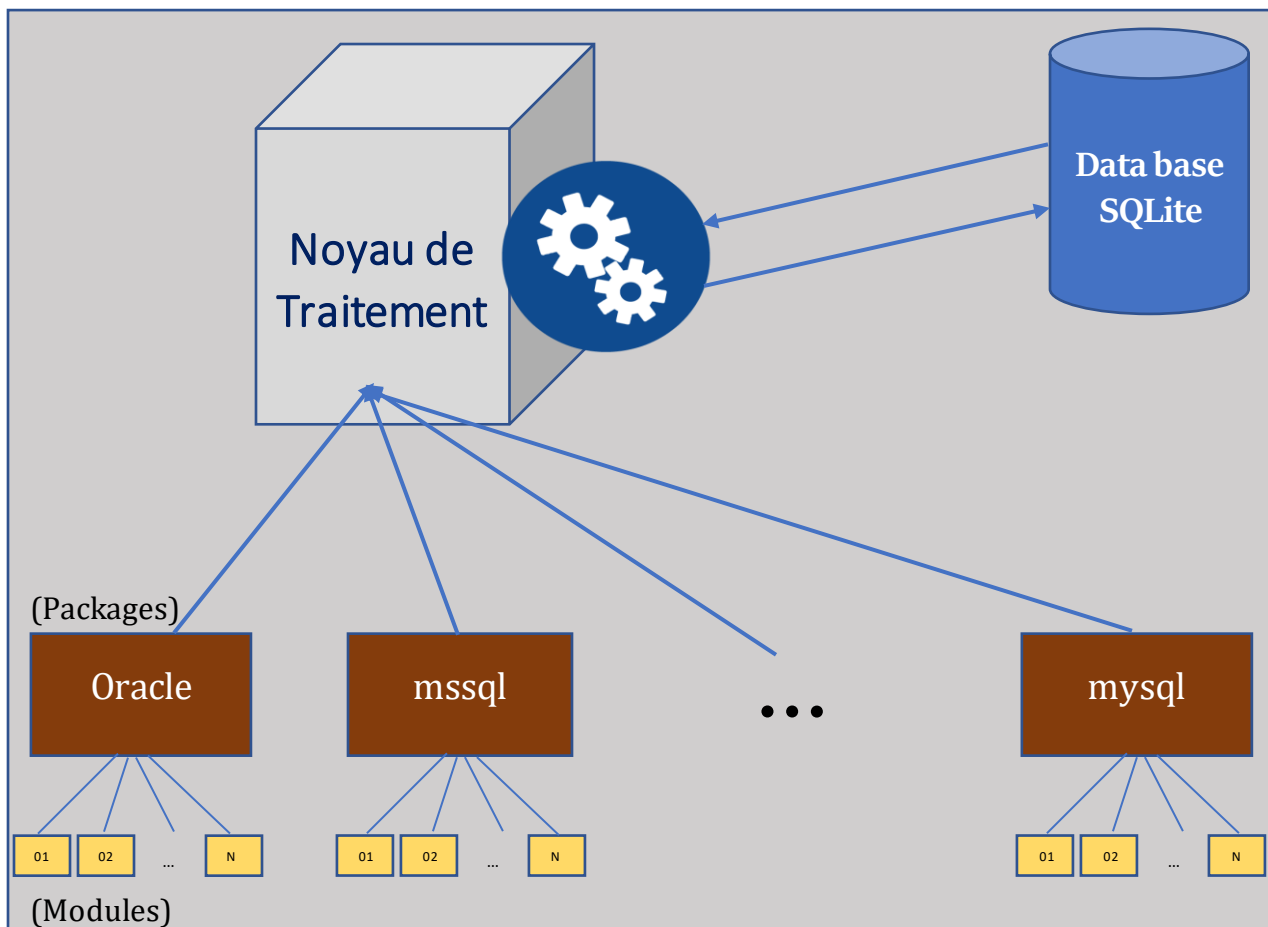


Figure 15: Architecture interne complète du système

### C. Architecture réseau

Du point de vue du réseau on retrouve deux entités communicantes : Le scanner de vulnérabilités qui va interroger la base de données cible.

Pour que notre système puisse communiquer avec la base de données cible, certains éléments sont indispensables. D'abord, il a besoin de connaître le couple (IP, PORT) du serveur. Ensuite, il faut des drivers de connexion Python, et enfin une instance client Oracle, ou un client SQL Server par exemple, qui permettra d'assurer les communications à distance.

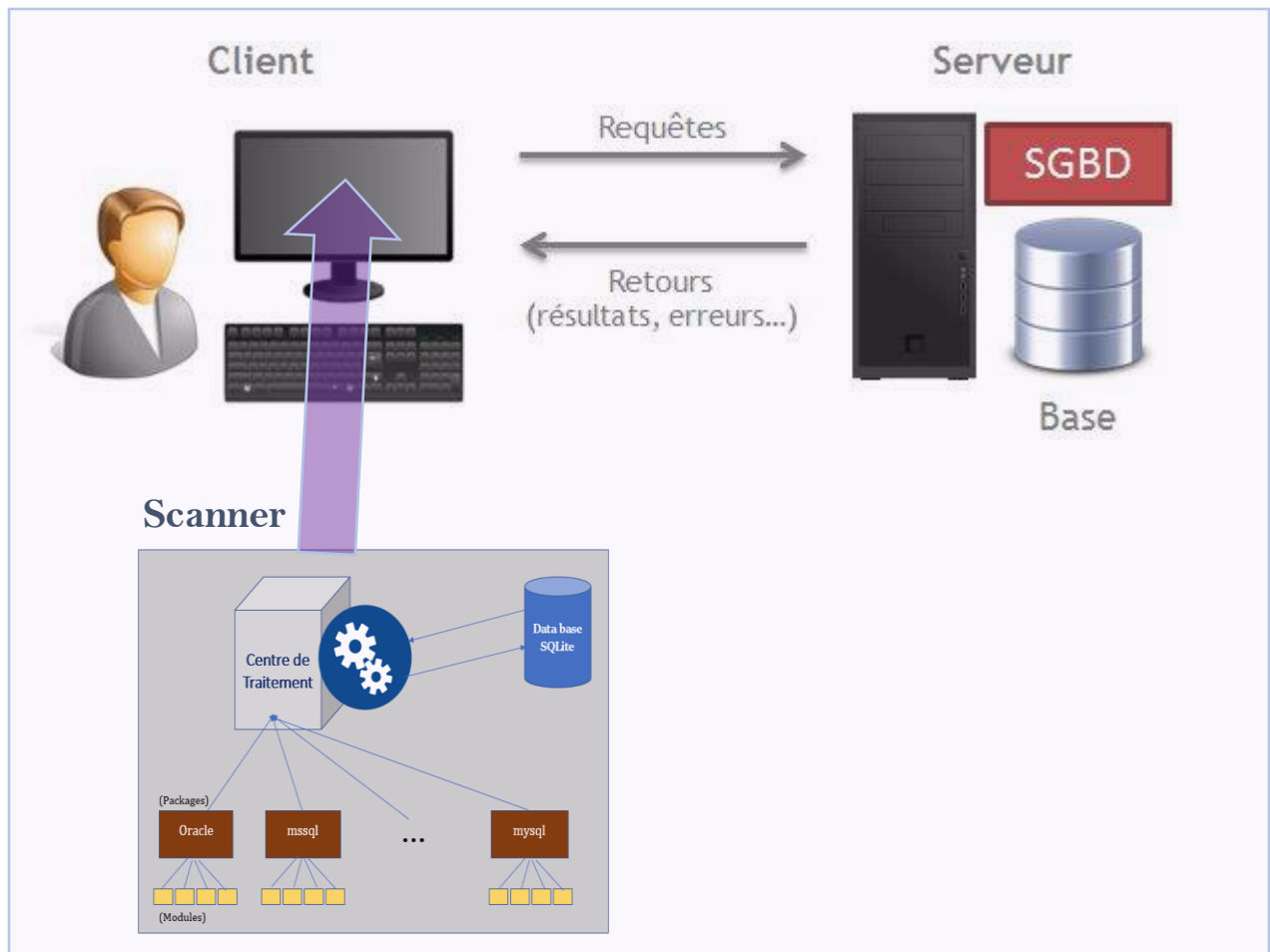


Figure 16: Architecture réseau

En définitive, nous devons retenir que l'architecture interne de cet outil est divisée en trois secteurs dans le but de répondre au besoin de flexibilité de l'audit de la base de données. Et chacune de ces parties joue un rôle bien précis dans le fonctionnement du système tout entier.

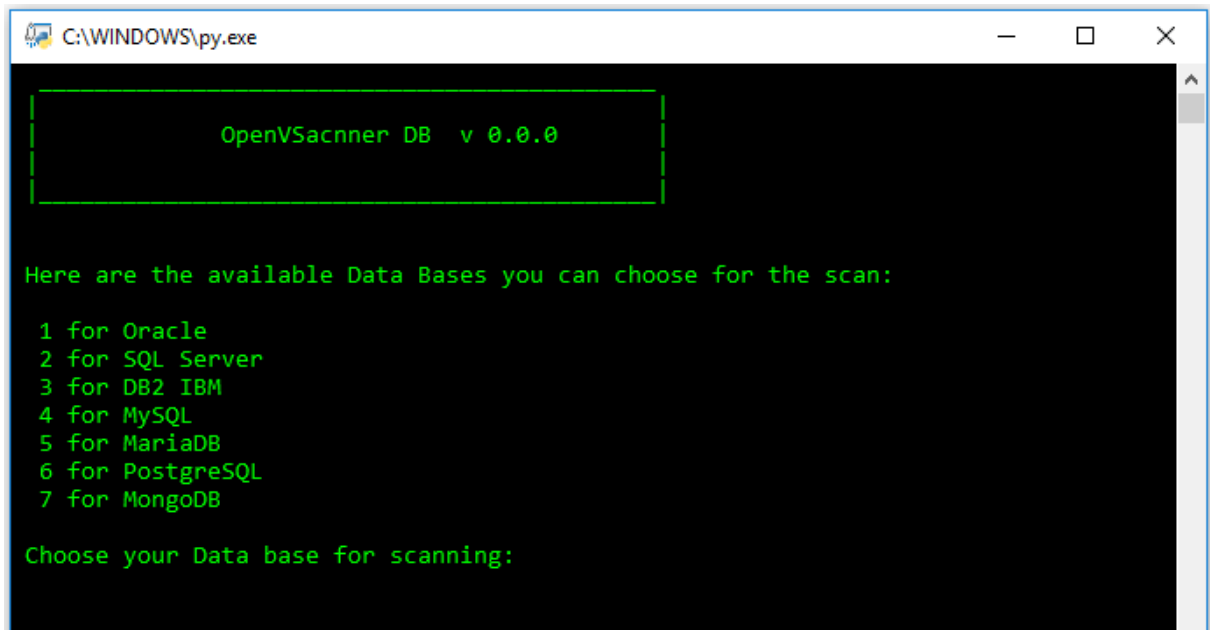
Du point de vu réseau, il existe des dépendances qui s'ajoute au programme afin de permettre la communication avec la base de données cible.

#### IV.4 – Déploiement et tests de la solution

Après avoir déployé la solution, nous avons utilisé comme système de gestion de base de données Oracle pour les tests.

La version de test de l'application est sous forme de Shell.

On démarre donc le programme en mode Shell, alors le programme nous propose directement une liste de SGBD qu'il est capable de scanner pour le moment.



```
C:\WINDOWS\py.exe

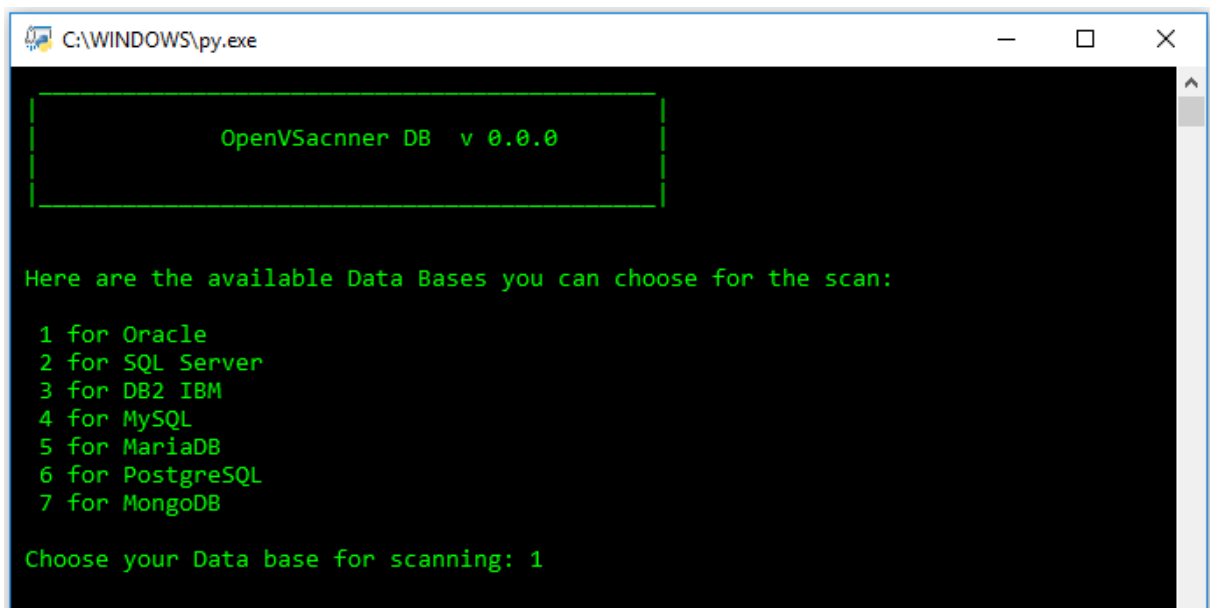
OpenVSacnner DB v 0.0.0

Here are the available Data Bases you can choose for the scan:

1 for Oracle
2 for SQL Server
3 for DB2 IBM
4 for MySQL
5 for MariaDB
6 for PostgreSQL
7 for MongoDB

Choose your Data base for scanning:
```

- 1- On choisit le SGBD qu'on veut scanner (dans cet exemple il s'agit d'une base de données Oracle, donc on tape 1) puis on fait « Entrer ».



```
C:\WINDOWS\py.exe

OpenVSacnner DB v 0.0.0

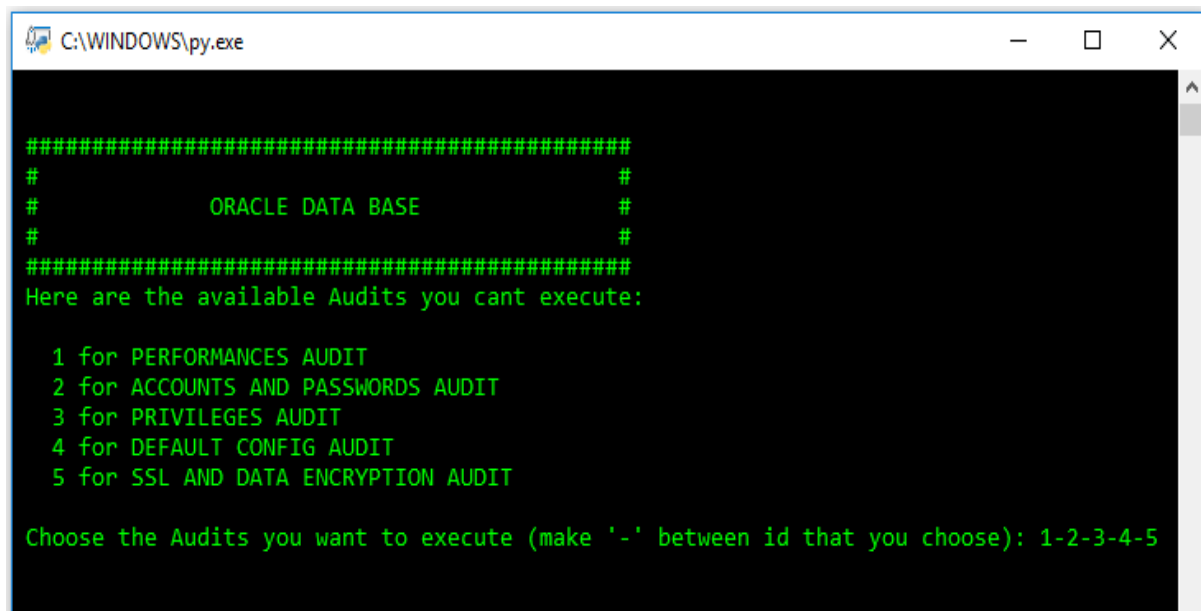
Here are the available Data Bases you can choose for the scan:

1 for Oracle
2 for SQL Server
3 for DB2 IBM
4 for MySQL
5 for MariaDB
6 for PostgreSQL
7 for MongoDB

Choose your Data base for scanning: 1
```

- 2- Après avoir choisi Oracle, le programme nous présente une liste d'audits (pour la base de données Oracle) qu'il sera capable de réaliser selon nos besoins.

On choisit les Audits qu'on souhaite effectuer : Par exemple on entre 1-2-3-4-5 puis on fait « Entrer ».



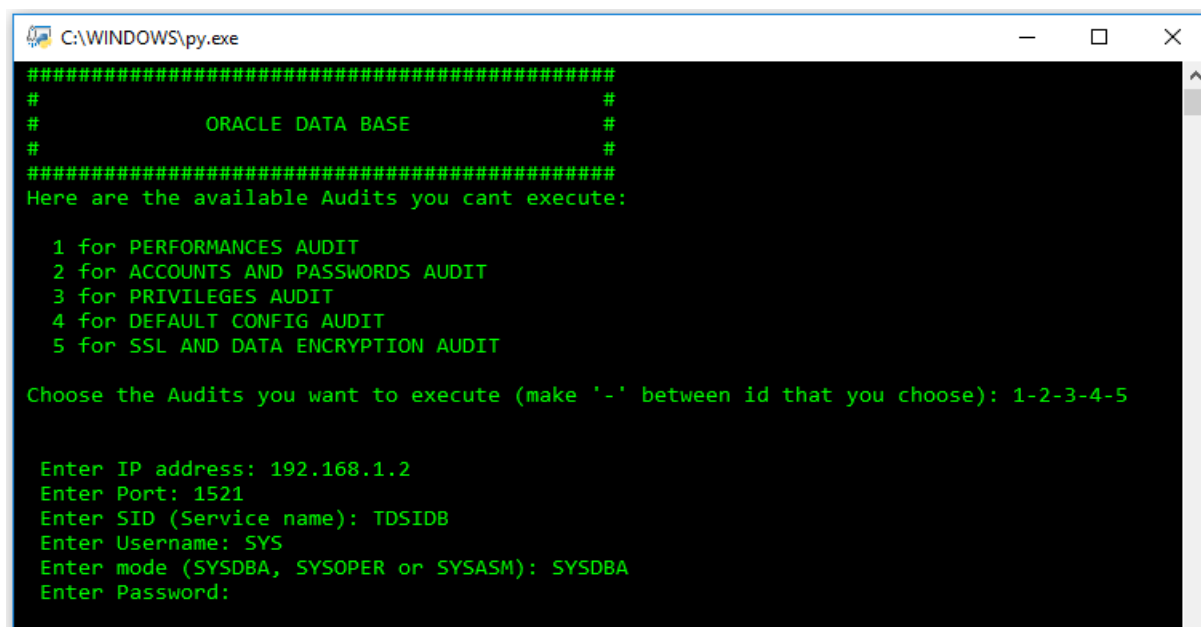
```
C:\WINDOWS\py.exe

#####
#                                     #
#          ORACLE DATA BASE          #
#                                     #
#####
Here are the available Audits you cant execute:

1 for PERFORMANCES AUDIT
2 for ACCOUNTS AND PASSWORDS AUDIT
3 for PRIVILEGES AUDIT
4 for DEFAULT CONFIG AUDIT
5 for SSL AND DATA ENCRYPTION AUDIT

Choose the Audits you want to execute (make '-' between id that you choose): 1-2-3-4-5
```

- 3- Le programme nous demande par la suite d'entre les informations de connexion à la base de données cible. On entre ces informations et on fait « Entrer ».



```
C:\WINDOWS\py.exe

#####
#                                     #
#          ORACLE DATA BASE          #
#                                     #
#####
Here are the available Audits you cant execute:

1 for PERFORMANCES AUDIT
2 for ACCOUNTS AND PASSWORDS AUDIT
3 for PRIVILEGES AUDIT
4 for DEFAULT CONFIG AUDIT
5 for SSL AND DATA ENCRYPTION AUDIT

Choose the Audits you want to execute (make '-' between id that you choose): 1-2-3-4-5

Enter IP address: 192.168.1.2
Enter Port: 1521
Enter SID (Service name): TDSIDB
Enter Username: SYS
Enter mode (SYSDBA, SYSOPER or SYSASM): SYSDBA
Enter Password:
```

On remarquera que le mot de passe n'est pas visible quand on le renseigne, c'est pour des simples raisons de sécurité.

- 4- Le programme établie la connexion avec la base de données cible puis commence le traitement.

```
C:\WINDOWS\py.exe

#####
#
#          ORACLE DATA BASE          #
#
#####
Here are the available Audits you cant execute:

 1 for PERFORMANCES AUDIT
 2 for ACCOUNTS AND PASSWORDS AUDIT
 3 for PRIVILEGES AUDIT
 4 for DEFAULT CONFIG AUDIT
 5 for SSL AND DATA ENCRYPTION AUDIT

Choose the Audits you want to execute (make '-' between id that you choose): 1-2-3-4-5

Enter IP address: 192.168.1.2
Enter Port: 1521
Enter SID (Service name): TDSIDB
Enter Username: SYS
Enter mode (SYSDBA, SYSOPER or SYSASM): SYSDBA
Enter Password:

Connection Established!

Wednesday 07 February 2018 17:14:06 GO!

Running...
```

- 5- Puis en moins d'une second, pour les quelques cas d'audits que nous avons choisi pour le test on a nos résultats affichés quoique brutes.

```
C:\WINDOWS\py.exe

Running...

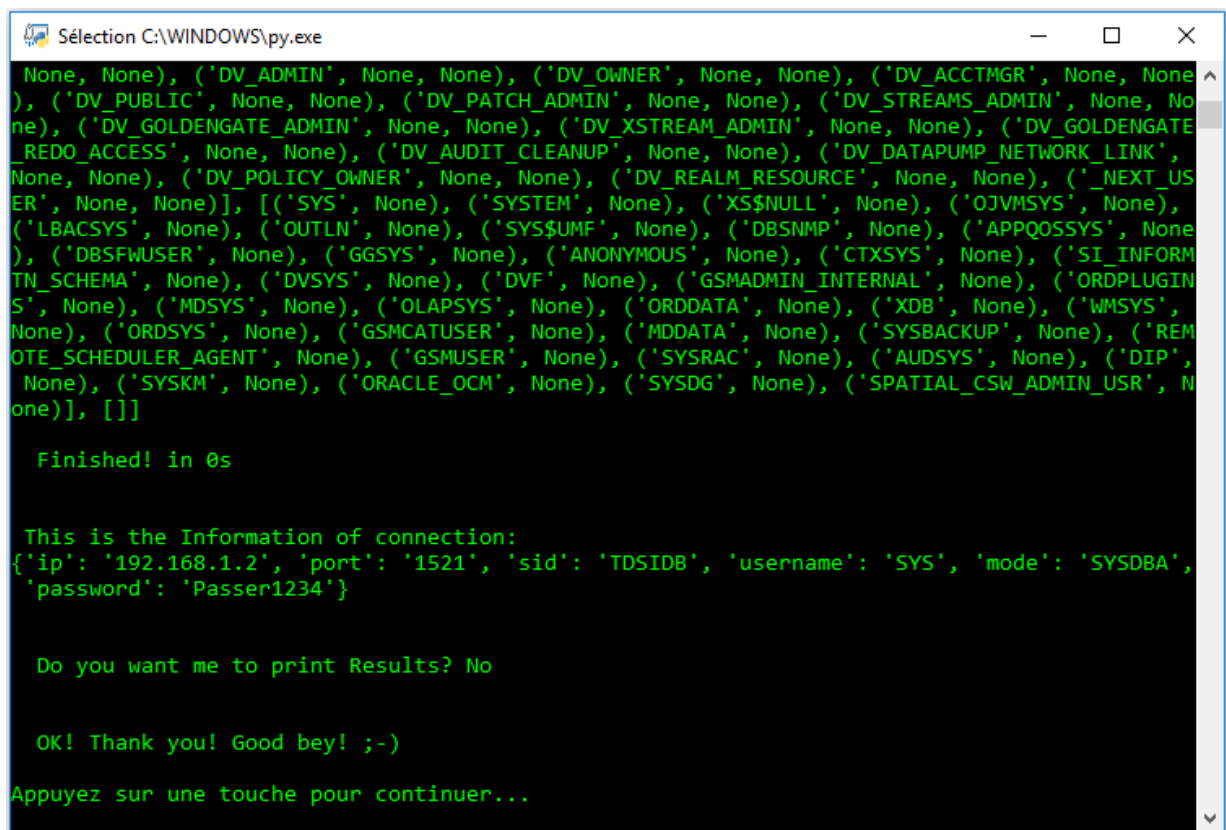
[[('Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production',), ('PL/SQL Release 12.2.0.1.0 - Production',), ('CORE\t12.2.0.1.0\tProduction',), ('TNS for 64-bit Windows: Version 12.2.0.1.0 - Production',), ('NLSRTL Version 12.2.0.1.0 - Production',)], [(8192,), (8192,), (8192,), (8192,)], [('MAXIMUM PERFORMANCE',)], [('NUM_CPUS', 4), ('NUM_CPU_CORES', 2), ('NUM_CPU_SOCKETS', 1)], [(2245184,)]], [('SYSRAC', None), ('ORDSYS', None), ('ORDDATA', None), ('MDSYS', None), ('OLAPSYS', None), ('LBACSYS', None), ('REMOTE_SCHEDULER_AGENT', 'Oracle Scheduler'), ('APPOSSYS', None), ('WMSYS', None), ('GSMCATUSER', None), ('XDB', None), ('OJVM SYS', None), ('SI_INFORMTN_SCHEMA', None), ('ORACLE_OCM', None), ('GGSYS', None), ('CTXSYS', None), ('ORDPLUGINS', None), ('MDDATA', None), ('SPATIAL_CSW_ADMIN_USR', None), ('SYS$UMF', 'Unified Manageability Framework'), ('DBSNMP', None), ('DIP', None), ('OUTLN', None), ('DBSFUSER', 'DB Service Firewall USER'), [('SYS',), ('SYSTEM',)], [('SYS', None, 'S:F9BF35348261387FF0E4677663AD7D80908DAEB2935532A9FE0ADA57F6A;T:D0F5687FE381A60915811AB9C48A3058080F0402E055A2E07C4039325E9F1073387326FA1CDF752388183464F7EEF81C5410E326DFF4658AED5C4437F5B34403A9DA40E73500140CE38233366B130625'), ('PUBLIC', None, None), ('CONNECT', None, None), ('RESOURCE', None, None), ('DBA', None, None), ('PDB_DBA', None, None), ('AUDIT_ADMIN', None, None), ('AUDIT_VIEWER', None, None), ('AUDSYS', None, 'S:3EC58B4728F59AF6F2F5B64048E3E1A54150A3EEAB3DA1CEF67EA7E16A20;T:BAEE2281E04EE40F553352920573CFA9EC8B70B7B0007B61A9DF46865EE80A09336D1F17D6B94997F990A8EEF794656D7EE281C79E52126DB836F557F8CD64B5FBFCF3095DABA749FA70C8EA29C74415'), ('SYSTEM', None, 'S:036F15A0354299B301CAF2F942FEE20AFDF4FE344336F6FBDD82E9492042;T:7A53B15DA8E0E5620271429A78B1AC11D4A58D8231E34C0F694AC4553D510CE7F0E564A3965C9D3534EE021BE7CA4A331FF61EC9CEF30EDCA751BF0C28B77B7CA06D214796542C96F091D994DF863032'), ('SELECT_CATALOG_ROLE', None, None), ('EXECUTE_CATALOG_ROLE', None, None), ('CAPTURE_ADMIN', None, None), ('SYSBACKUP', None, 'S:0230A81F40BDAA9AB844CA683A32796D9FC8487731662D4AE7B9A1F2AE1B;T:A3BC43B22191BFF2584BE34E77DC717F7156D70FF70BCF101B5C3D4CBD58E3E4BF76FBA03F349F0609D9A9BA7DADF91C15C6689E7AC92A9CDF4F03ADEE1C74EE6251D9C130AC438F878687976920E4AA'), ('SYSDG', None, 'S:75B94F2A01F350117A5CDFB5B56024BE1CBCDFCA206695B4C75B47045223;T:383B3A9A52F853B97F41671555D4344BFCAB1F0C09996F5350C8C87110998193
```



Il est évident que ces résultats ne soient pas lisibles pour tout le monde mais il y a pas mal d'informations importantes qu'on peut déjà voir.

[illegible]

```
CF73998D5413BE254971E55CB4021E214C85F158818CE4C'), ('GSM_POOLADMIN_ROLE', None, None), ('OPTIMIZER_PROCESSING_RATE', None, None), ('DBMS_MDX_INTERNAL', None, None), ('ORACLE_OCM', None, None), ('S:D40F3667327868AF6756BA392DA6AC0B5929BDA74A787FD8E007A026AEA;T:428FC22DA6E6A02579CECF5CEFCBF217B2CBE808E949D15187C0AB7482E54B195E89AA23E16233FB2C6308858D75F4F15A9FDAEEF83CF0369171BEC702BAA65933DE0B2E5309115D49D0D597B784D8DFE'), ('RECOVERY_CATALOG_OWNER', None, None), ('RECOVERY_CATALOG_OWNER_VPD', None, None), ('RECOVERY_CATALOG_USER', None, None), ('EM_EXPRESS_BASIC', None, None), ('EM_EXPRESS_ALL', None, None), ('SYSUMF_ROLE', None, None), ('SYSUMF', None, None), ('S:290DA535E9F5E3B9EB726AB3D83A61E05C0293D759202AC65A715680E9FB;T:62A1EE725BD6D0F519A954438CD9391C0DF9DDA4376F1BE7804F9A8CCCAE6E2C452898BF94C1712E857488C143179E62B8BACD9D69456FCB0AADA6E641F2A0DB1EA37A2418A58280DB71C580EF5E7DF06'), ('SCHEDULER_ADMIN', None, None), ('HS_ADMIN_SELECT_ROLE', None, None), ('HS_ADMIN_EXECUTE_ROLE', None, None), ('HS_ADMIN_ROLE', None, None), ('GLOBAL_AQ_USER_ROLE', None, None), ('OEM_ADVISOR', None, None), ('OEM_MONITOR', None, None), ('DBSNMP', None, None), ('S:D0790E24D857211B523B5FD9116CE8CDF06DE201C9EC25FE5B505C1A56A7;T:97646AA0D7352CB739454C32215DA3A8642CBC522014A7BFCA46A6A6F99A6B3B2D62C8423CB40A53D720CC56372D12A942307543901F76A829CE17BD2AE26908FE3578093C8393B1F515932B75E70A3'), ('APPROSYS', None, None), ('S:7BEAD0E96CB89FEFF20621B49EF5B8E162E079FB691108FD600AB62C605;T:A E515D30F32C55261FE6BCB730562F8080AA11C9E71ADE2E322B568D03E12DC70C8C0E0423FF1100A71C2A5AF8C4F8B5EC4C10613BF5823145285EAAF4DDF7C068CDA19D7DF15DBB450E961DF253B4F'), ('GSMADMIN_ROLE', None, None), ('EJBCLIENT', None, None), ('GDS_CATALOG_SELECT', None, None), ('GSMCATUSER', None, None), ('S:23B775952C973D600DD13AE51C95463DE1E8207FC0A8F24AE735455993F;T:2EF4F97CC97A7E19F7F41463697FFD6DD6CD8A553AD73573F5974CAE0D7EF4E3BAA5228B099C22E9B01D918B2B13BDADF79C9479D3F0A1C34C5DF199A8ACCF36424DBB6C8C27FE4143E5568D88A9DFDC'), ('GGSYS', None, None), ('S:7A368690AD44229880773485FE4A55CC3DBD1BAEFF7F8B592F8A48CE876;T:6FE0EE70898666B7C8687F13EDA6A28C5DBB60F961DD1DC96A0A9081BC18BC0811A17D233565004AB4EAB7ED45066E5E9C5AED6DF2EDC837DAE65BAEA05990708017AFE2BBD1C9E2856398070A0F48'), ('GGSYS_ROLE', None, None), ('XDB', None, None), ('S:BA610ED7A665796D86D2ECD18045D23B1F691E51D1A273A890CF33FB6D46;T:94773C49C3F5B96EF47D63A900CE39EDA731BDE312697DED EA66DCDFA5D1C3C8DD19EBBBE3DAF70DB9E67E93B306F857DD3A48CA29FC713528160477C344307F0A8DF61D597D03E1874AD0B061B3312'), ('ANONYMOUS', None, None), ('XDBADMIN', None, None), ('XDB_SET_INVOKER', None, None), ('AUTHENTICATEDUSER', None, None), ('XDB_WEBSERVICES', None, None), ('XDB_WEBSERVICES WITH PUBLIC', None, None), ('XDB_WEBSERVICES OVER HTTP', None, None),
```



```
None, None), ('DV_ADMIN', None, None), ('DV_OWNER', None, None), ('DV_ACCTMGR', None, None), ('DV_PUBLIC', None, None), ('DV_PATCH_ADMIN', None, None), ('DV_STREAMS_ADMIN', None, None), ('DV_GOLDENGATE_ADMIN', None, None), ('DV_XSTREAM_ADMIN', None, None), ('DV_GOLDENGATE_REDO_ACCESS', None, None), ('DV_AUDIT_CLEANUP', None, None), ('DV_DATAPUMP_NETWORK_LINK', None, None), ('DV_POLICY_OWNER', None, None), ('DV_REALM_RESOURCE', None, None), ('_NEXT_USER', None, None)], [('SYS', None), ('SYSTEM', None), ('XS$NULL', None), ('OJVMSYS', None), ('LBACSYS', None), ('OUTLN', None), ('SYS$UMF', None), ('DBSNMP', None), ('APPQOSSYS', None), ('DBSFUSER', None), ('GGSYS', None), ('ANONYMOUS', None), ('CTXSYS', None), ('SI_INFORMTN_SCHEMA', None), ('DVSYS', None), ('DVF', None), ('GSMADMIN_INTERNAL', None), ('ORDPLUGINS', None), ('MDSYS', None), ('OLAPSYS', None), ('ORDDATA', None), ('XDB', None), ('WM SYS', None), ('ORDSYS', None), ('GSMCATUSER', None), ('MDDATA', None), ('SYSBACKUP', None), ('REMOTE_SCHEDULER_AGENT', None), ('GSMUSER', None), ('SYSRAC', None), ('AUDSYS', None), ('DIP', None), ('SYSKM', None), ('ORACLE_OCM', None), ('SYSDG', None), ('SPATIAL_CSW_ADMIN_USR', None)], []]
```

Finished! in 0s

This is the Information of connection:  
{'ip': '192.168.1.2', 'port': '1521', 'sid': 'TDSIDB', 'username': 'SYS', 'mode': 'SYSDBA', 'password': 'Passer1234'}

Do you want me to print Results? No

OK! Thank you! Good bey! ;-)

Appuyez sur une touche pour continuer...

A la fin on remarque que la durée du scan est de moins d'une seconde), on voit aussi les informations de connexion qui ont été expressément affichées.

Et juste après le programme nous demande si on souhaite qu'il nous affiche le résultat. Il s'agit d'un affichage au format HTML.

Mais on a entré « **No** », pour dire non au programme, car nous n'avons pas encore terminer cette section qui traite du formatage des résultats.

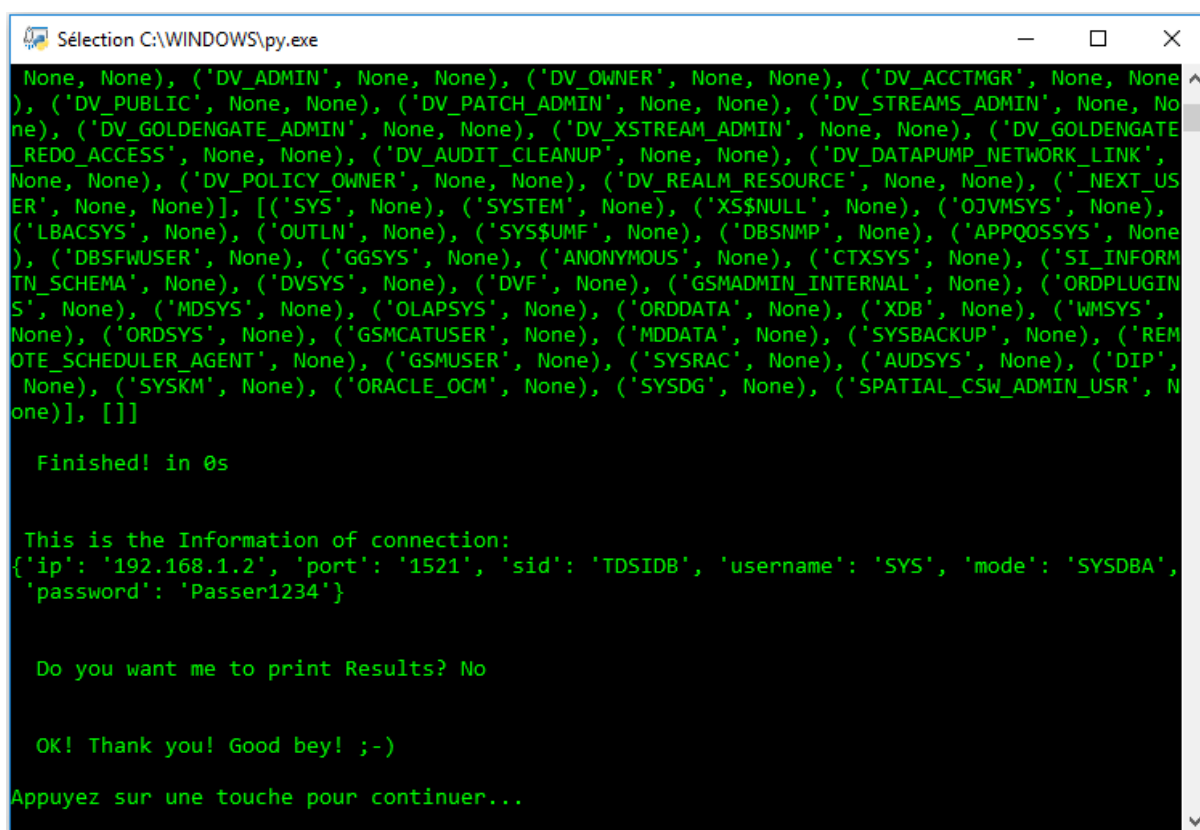
**PARTIE III :**  
**Interprétation des résultats, amélioration et recommandations**

## V – Interprétations des résultats

Quoi que les résultats que nous avons ne sont que partiels car nous n'avons même pas encore exploité le quart des performances de notre système, Il est tout de même possible de donner quelques interprétations superficielles des résultats obtenus.

### V.1 – Interprétation 1 : La rapidité du programme

Après avoir vu dans le chapitre précédant les résultats d'un test effectué sur la base de données Oracle (voir image ci-dessous), on peut constater qu'en moins d'une seconde le programme arrive à obtenir plusieurs résultats. Ceci nous permet de conjecturer la rapidité du programme. Cette rapidité est simplement dû aux performances qu'offre le langage Python.



```
None, None), ('DV_ADMIN', None, None), ('DV_OWNER', None, None), ('DV_ACCTMGR', None, None), ('DV_PUBLIC', None, None), ('DV_PATCH_ADMIN', None, None), ('DV_STREAMS_ADMIN', None, None), ('DV_GOLDENGATE_ADMIN', None, None), ('DV_XSTREAM_ADMIN', None, None), ('DV_GOLDENGATE_REDO_ACCESS', None, None), ('DV_AUDIT_CLEANUP', None, None), ('DV_DATAPUMP_NETWORK_LINK', None, None), ('DV_POLICY_OWNER', None, None), ('DV_REALM_RESOURCE', None, None), ('_NEXT_USER', None, None)], [('SYS', None), ('SYSTEM', None), ('XS$NULL', None), ('OJVM SYS', None), ('LBACSYS', None), ('OUTLN', None), ('SYS$UMF', None), ('DBSNMP', None), ('APPOSSYS', None), ('DBSFUSER', None), ('GGSYS', None), ('ANONYMOUS', None), ('CTXSYS', None), ('SI_INFORMTN_SCHEMA', None), ('DVSYS', None), ('DVF', None), ('GSMADMIN_INTERNAL', None), ('ORDPLUGINS', None), ('MDSYS', None), ('OLAPSYS', None), ('ORDDATA', None), ('XDB', None), ('WMSYS', None), ('ORDSYS', None), ('GSMCATUSER', None), ('MDDATA', None), ('SYSBACKUP', None), ('REMOTE_SCHEDULER_AGENT', None), ('GSMUSER', None), ('SYSRAC', None), ('AUDSYS', None), ('DIP', None), ('SYSKM', None), ('ORACLE_OCM', None), ('SYSDG', None), ('SPATIAL_CSW_ADMIN_USR', None)], []]

Finished! in 0s

This is the Information of connection:
{'ip': '192.168.1.2', 'port': '1521', 'sid': 'TDSIDB', 'username': 'SYS', 'mode': 'SYSDBA', 'password': 'Passer1234'}

Do you want me to print Results? No

OK! Thank you! Good bey! ;-)

Appuyez sur une touche pour continuer...
```

En effet, ces résultats sont non négligeables d'autant plus que les informations recueillies sont bien exploitables pour en tirer des vulnérabilités. On peut par exemple voir que cette base de données possède plusieurs comptes par défaut tels que 'SYS', 'SYSTEM', etc.

## V.2 – Interprétation 2 : La flexibilité du programme

La flexibilité est l'un des caractères les plus importants que nous voulions donner à ce scanner de vulnérabilités de base de données. La modélisation et structuration du système ont été fait pour répondre à ce besoin.

La flexibilité de cet outil intervient à deux niveaux pour :

### 1. Tenir compte de l'évolutivité avec les SGBD :

Pour que le programme soit stable au cours du temps on doit tenir compte de l'évolutivité des Systèmes de Gestion de Base de Données. C'est ce qui justifie le « **Centre de traitement** » qui est en quelque sorte le noyau du scanner. Cette partie du système a été conçue pour être totalement indépendante des SGBD.

Mais il y a des petits programmes qui sont en fait des « **Agents** », ces agents peuvent être codés par plusieurs développeurs et ceci de façon indépendante.

### 2. Tenir compte des besoins des auditeurs :

En essayant de nous plonger dans la tête d'un auditeur d'une base de données, on réalise très vite qu'il est très important que le scanner de vulnérabilités laisse au spécialiste la possibilité de choisir que secteur doit-on auditer. Ceci permet en effet de cibler précisément les tâches qu'il souhaite exécuter dans sa base de données, plutôt que d'avoir un outil qui fait tout même ce qui ne lui est pas demandé.

L'image suivante est une projection futuriste de ce scanner de vulnérabilités avec toute sa flexibilité.


















Figure 17: Image future du scanner

### V.3 – Comparaison

Après évaluations de notre solution, il est clair que celle-ci présente des avantages qui la place bien au-dessus des solutions existantes. Le tableau suivant nous permet d'en tirer une conclusion. Notre solution porte le nom de « **OpenVScanner** »

Tableau 3: Tableau comparatif

	Multiples SGBD	Multiplateforme	Flexibilité	Extensibilité	Mobilité
Scuba					
Squirrel					
OpenVScanner					

### V.4 – Description des principaux avantages de la solution

Le scanner de vulnérabilité de base de données que nous avons conçu, quoique n'étant pas encore arrivé à son aboutissement, on peut tout de même voir qu'il possède de nombreux avantages décrits comme suit :

1. **L'option « Attacks » du logiciel** : Cette option refferme multiples attaques qu'il est possible d'exécuter sur une base de données précise. Cette option donne donc au scanner l'avantage d'être aussi un outil de test d'intrusions.
2. **L'extensibilité du logiciel** : Bien que n'étant pas encore totalement achevé, ce programme est conçu de façon à ce que l'on puisse ajouter des extensions, c'est-à-dire des bouts de code écrits indépendamment du développement du programme lui-même.  
Et ceci est rendu possible grâce au caractère indépendant que possède déjà le **noyau** (Centre de traitement) du programme et grâce au fait que Python est facilement extensible avec de code C/C++/Java et facilement intégrable dans les applications.
3. **La flexibilité du logiciel** : Comme évoqué dans la section « **V.2 Interprétation 2** », ce caractère constitue un avantage majeur de ce logiciel et fait de ce scanner un outil particulier pour l'audit.
4. **Multiplateformes** : Grâce aux avantages multiplateformes que possède Python.
5. **Gestion de la mobilité** : Permettre d'effectuer l'envoi du rapport de scan via mail.

## VI – Quelques idées d'améliorations et les recommandations

Si l'approche que nous présentons s'avère intéressante et que la version actuelle de notre scanner de vulnérabilités est déjà capable de fournir des résultats exploitables, il n'est pas exclu que pour une utilisation optimale de cet outil il y a d'importantes améliorations à envisager. Et aussi quelques recommandations doivent être pris en compte.

### VI.1 – Amélioration 1 : Optimisation en intégrant des outils open source existants

Il est fort possible que les performances de ce scanner de vulnérabilités puissent être améliorées considérablement.

En effet, parmi les principaux avantages que possède cet outil nous avons cité l'**extensibilité**. Il est possible d'intégrer des programmes écrits dans les langages tels que C/C++ ou Java, ceci à cause de la capacité d'extensibilité de Python.

Cependant, bon nombre de programmes libres de droits, gratuits et open source sont très optimisés pour effectuer certaines tâches dont peut avoir besoin notre scanner.

C'est le cas de :

#### 1- Nmap

"**Network Mapper**" est un outil open source (GNU PGL) d'exploration réseau et d'audit de sécurité. Écrit en C++, Python, C, Lua et Java, il a été conçu pour rapidement scanner de grands réseaux, mais il fonctionne aussi très bien sur une cible unique.

**Nmap** peut être intégré à notre scanner de vulnérabilités pour évaluer les risques sur les protocoles de communications, ou encore il peut être intégré dans les options d'attaques notamment pour les pentestes.

#### 2- ODAT

"**Oracle Database Attacking Tool**" est un outil offensif codé en Python permettant de tester la sécurité d'une base de données Oracle 10g, 11g, ou 12c avec ou sans compte. Cet outil implémente un certain nombre d'attaques en un minimum de temps.

#### 3- OSCANNER

**OSCANNER** est un Framework développé en Java, il est disponible dans la plateforme Kali Linux.

#### 4- OAT – Oracle Auditing Tools

#### 5- Etc.

## **VI.2 – Amélioration 2 : Evolutivité de la solution**

### ***VI.2.1 – Migration du scanner vers le model d'une application web***

Dans cette partie on vise à faire évoluer le scanner pour qu'il soit présenté sous forme une application Web, comme c'est cas pour les outils tels que **Nessus**.

Le Framework Python Django est un outil puissant pour réaliser cela.

Plusieurs avantages sont liés à cette idée, à savoir :

1. **L'exploitation de services web** pour une vue encore plus large de l'outil qui consisterait à faire communiquer cet outil avec d'autres applications de façon indépendante.
2. **La présentation des résultats** à ce niveau deviendra encore plus souple et plus élaborée.
3. **Accès à l'interface de l'application via http/https** à partir d'un navigateur.

### ***VI.2.2 – Intégrer l'analyse des journaux pour l'audit de base de données native***

La **journalisation** est une technique importante de sécurité d'une base de données. Elle désigne l'enregistrement chronologique des opérations de la logique métier pendant le fonctionnement du système de gestion de base de données.

Dans ce cadre, les événements enregistrés seront les accès au système, les modifications de fichiers, etc. On consacre typiquement une ligne par événement, en commençant par le moment exact (date, heure, minute, seconde) où il a eu lieu.

Donc analyser minutieusement les fichiers journaux (logs files), permettra de retracer la vie même du système de gestion de base de données, et ainsi fournir des indicateurs de santé de la base de données.

L'idée est d'intégrer une option dans le scanner de vulnérabilités qui traitera uniquement les logs afin d'en ressortir les anomalies dans la base de données, pour produire un audit natif.

### ***VI.2.3 – Implémenter des alertes en temps réel (intégration SIEM)***

Il s'agit de mettre en place un système d'alerte pour la gestion des événements et des informations de sécurité. Cette technique permettra de gérer la menace en temps réel, c'est-à-dire au moment même où la base de données est en production.

Le principe consiste à développer une extension du scanner qui sera déployé lors du premier scan du vulnérabilités dans la base de données cible, à partir de ce moment le programme pourra surveiller l'activité de la base en temps réel, et fournir les alertes nécessaires.



### VI.3 – Recommandation 1 : Compte d'authentification à la base de données cible

Nous avons vu que notre scanner de vulnérabilités de base de données, comme c'est le cas pour les outils existants, a comme **méthode de détection** le **scan authentifié**. L'utilisateur est emmené à fournir un nom d'utilisateur et un mot de passe pour se connecter à la base de données et ensuite effectuer le scan.

Le compte d'utilisateur qui doit être renseigné est d'une grande importance car il doit interroger des éléments importants, voir sensibles de la base de données : **Il faut par conséquent avoir les privilèges nécessaires pour accéder à ces données.**

Le cas le plus simple est de se connecter avec un compte Administrateur et effectuer librement le scan.

Cependant, si nous avons en face une entreprise qui possède des bonnes manières de sécurité, alors les comptes Administrateurs seront bloqués. Dans ce cas deux possibilités sont envisageables :

**1- Activer temporairement le compte Administrateur puis le désactiver après le scan.**

Dans ce cas l'administrateur de base de données devra se charger de cette tâche bien avant de lancer le scan.

**2- Utiliser le compte que le scanner lui-même propose de créer pour effectuer le scan.**

Dans ce cas une option du scanner pourra servir pour créer un compte avec tous les privilèges nécessaires, et pour se faire l'on devra au préalable renseigner un compte d'utilisateur avec les privilèges capables de créer ce nouveau compte que le scanner créera.

### VI.4 – Recommandation 2 : La Sécurité du réseau

Lors de la communication entre le scanner et la base de données cible, les informations échangées à travers le réseau peuvent être interceptées à partir d'une simple attaque passive, en utilisant des outils tels que Wireshark (analyseur de paquets). Dès lors, il est nécessaire d'utiliser le seul moyen contre cette attaque : **Le chiffrement des échanges.**

On peut envisager deux possibilités :

**1- Activer le protocole TCPS et se servir d'une instance client du SGBD cible : C'est le cas d'Oracle et SQL Server.**

**2- Utiliser un VPN simple comme OpenVPN qui est un outil open source, multiplateforme et facile à configurer pour mettre en place un réseau privé virtuel entre le scanner de vulnérabilités et la cible.**

## CONCLUSION :

Il a été question dans ce travail de concevoir et développer un scanner de vulnérabilités de base de données. Pour parvenir au résultat attendu, nous avons tout d'abord montré comment sont structurées les bases de données au travers des différentes architectures présentées. Ensuite nous avons vu comment fonctionnent de façon générale un scanner de vulnérabilités et nous avons présenté les détails du fonctionnement de la solution que nous proposons.

Grâce à la robustesse du langage Python et à la simplicité du model de cette solution qui consiste en un noyau, une base de données locale, et des programmes indépendants que nous appelons des « Agents », nous avons pu répondre aux exigences de flexibilité, et d'extensibilité, voire celles d'un outil complet, que n'offrent pas les solutions existantes.

Au terme donc de ce travail nous pouvons affirmer qu'il est tout à fait possible de produire un outil complet d'audit de base de données capable de répondre aux besoins des entreprises. Ce qui implique une grande avancée des outils d'audits de base de données.

Les difficultés ont été énormes dans ce travail. Premièrement pour la simple raison que le sujet est très peu documenté même en anglais, rappelons qu'il ne s'agit pas d'un scanner de vulnérabilités tout cours (dont il existe une pléthore sur le marché et avec une grande documentation), mais un scanner de vulnérabilités d'une base de données. Deuxièmement la maîtrise du fonctionnement de chaque SGBD dont il fallait tenir compte était strictement nécessaire. Et troisièmement le manque de temps pour les multiples tests à effectuer, car nous ne disposions que de 4 mois pour réaliser tout le travail.

Toutefois, des perspectives intéressantes s'offre à nous, notamment le développement d'un module qui s'occupera uniquement d'appliquer des correctifs nécessaires après avoir obtenu les résultats d'un scan, le passage à l'échelle pour des bases de données contenant de très grands volumes de données (Big Data, NoSQL), et aussi rendre le système intelligent de façon à ce qu'il apprenne tout seul et s'adapte à l'environnement qu'il audite (Machine Learning).

## WEBOGRAPHIE :

- [1] petite-entreprise.net, «Historique des bases de données», P-E.net, [En ligne].  
<https://www.petite-entreprise.net/P-2649-85-G1-historique-des-bases-de-donnees.html>  
[Accès le 18 Janvier 2018].
- [2] WikiPedia, «Base de données», 11 Décembre 2017. [En ligne].  
[https://fr.wikipedia.org/wiki/Base\\_de\\_donnees](https://fr.wikipedia.org/wiki/Base_de_donnees) [Accès le 18 Janvier 2018].
- [3] WikiPedia, «NoSQL», 11 Décembre 2017. [En ligne].  
<https://fr.wikipedia.org/wiki/NoSQL> [Accès le 12 Février 2018].
- [4] G. Press, «A Very Short History Of Big Data», 21 Décembre 2013. [En ligne].  
<https://www.forbes.com/sites/gilpress/2013/05/09/a-very-short-history-of-big-data/#312676b365a1> [Accès le 18 Janvier 2018].
- [5] WikiPedia, «Big Data», 17 Décembre 2017. [En ligne].  
[https://fr.wikipedia.org/wiki/Big\\_data#cite\\_note-19](https://fr.wikipedia.org/wiki/Big_data#cite_note-19) [Accès le 10 Janvier 2018].
- [6] P-E.net, «Définition des bases de données», 27 Mai 2013. [En ligne].  
<https://www.petite-entreprise.net/P-2648-85-G1-definition-des-bases-de-donnees.html>  
[Accès le 11 Janvier 2018].
- [7] ANSI, «Normes d'audit», 16 Mars 2017. [En ligne].  
[https://www.ansi.tn/fr/pages/audit/normes\\_audit.html](https://www.ansi.tn/fr/pages/audit/normes_audit.html) [Accès le 18 Janvier 2018].
- [8] L. Redaction, «Comment securiser ses bases de donnees», 27 Mai 2009. [En ligne].  
<https://www.techniques-ingenieur.fr/actualite/articles/comment-securiser-ses-bases-de-donnees-11033/> [Accès le 11 Janvier 2018].
- [9] C. Group, «Mise en conformité avec la loi Sarbanes-Oxley», Tango/04 , Barcelona, 2005.
- [10] InfoRisque, «LE GUIDE DE LA GESTION DES RISQUES», 12 Mai 2007. [En ligne].  
<https://www.inforisque.info/fiches-pratiques/norme-iso-27001.php> [Accès le 02 Février 2018].
- [11] azure.microsoft.com, «13 effective security controls for iso 27001 compliance», 29 Janvier 2016. [En ligne]. <https://azure.microsoft.com/en-us/blog/13-effective-security-controls-for-iso-27001-compliance/> [Accès le 15 Novembre 2017].
- [12] NCCGROUP, «Top 10 Common Database Security Issues», 21 Juillet 2014. [En ligne].  
<https://www.nccgroup.trust/uk/about-us/newsroom-and-events/blogs/2014/july/top-10-common-database-security-issues/> [Accès le 10 10 2017].
- [13] IMPERVA, «Top 10 Database Vulnerabilities», 27 Septembre 2006. [En ligne].  
[http://schell.com/Top\\_Ten\\_Database\\_Threats.pdf](http://schell.com/Top_Ten_Database_Threats.pdf) [Accès le 17 Novembre 2017].
- [14] A. B. R. Tejashri R. Gaikwad, «A Review on Database Security», 04 Acril 2014. [En ligne].  
<https://www.ijsr.net/archive/v3i4/MDIwMTMxMjc3.pdf> [Accès le 20 Décembre 2017].

- [15] WikiPedia, «Scanneur de vulnérabilité,» 24 Janvier 2018. [En ligne].  
[https://fr.wikipedia.org/wiki/Scanneur\\_de\\_vulnérabilité](https://fr.wikipedia.org/wiki/Scanneur_de_vulnérabilité) [Accès le 24 Janvier 2018].
- [16] 01net, «PyCharm Community Edition,» 24 Avril 2017. [En ligne].  
<http://www.01net.com/telecharger/linux/Programmation/fiches/131017.html> [Accès le 11 Janvier 2017].
- [17] N. D. e. S. Cass, «The Top Programming Languages 2017,» ., 18 Juillet 2017. [En ligne].  
Available: <https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2017> [Accès le 16 Octobre 2017].
- [18] IMPERVA, «Scuba Database Vulnerability Scanner,» 12 02 2016. [En ligne].  
[https://www.imperva.com/lg/lgw\\_trial.asp?pid=213](https://www.imperva.com/lg/lgw_trial.asp?pid=213) [Accès le 08 Janvier 2018].
- [19] NNCGroup, «Information Security Software,» 23 Juin 2015. [En ligne].  
<https://www.nccgroup.trust/uk/our-services/cyber-security/products-and-cloud-services/information-security-software/> [Accès le 13 Janvier 2018].
- [20] WikiPedia, «UML(informatique),» 16 12 2017. [En ligne].  
[https://fr.wikipedia.org/wiki/UML\\_\(informatique\)](https://fr.wikipedia.org/wiki/UML_(informatique)) [Accès le 12 01 2018].
- [21] WikiPedia, «PowerAMC,» 17 Décembre 2017. [En ligne].  
<https://fr.wikipedia.org/wiki/PowerAMC> [Accès le 14 Janvier 2018].
- [22] SQLite, «About SQLite,» ., 22 Janvier 2018. [En ligne].  
<https://www.sqlite.org/about.html> [Accès le 27 Janvier 2018].