MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU.

FACULTE GENIE ELECTRIQUE ET D'INFORMATIQUE

DEPARTEMENT INFORMATIQUE

Réseaux, Mobilités et Systèmes Embarqués

MEMOIRE DE FIN D'ETUDE EN VUE DE L'OBTENTION DU DIPLÔME DE MASTER

Thème :

Réalisation d'un IDS à base d'un réseau LSTM

Realisé par :

Mlle AMIAR Thinhinane

Mr BELKAI Salah

Proposé par :

Mme AOUDJIT Rachida

Année universitaire:

2021/2022

REMERCIEMENTS

Nos louanges vont vers ALLAH qui nous a guidés et aidés pour l'accomplissement de ce travail.

Nous tenons particulièrement à remercier notre promoteur : Mme AOUDJIT pour son soutien, ses conseils et son aide précieuse durant toute la période du travail.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre travail en acceptant de l'examiner et de l'enrichir par leurs propositions. Nous remercions tous les enseignants qui ont contribué à notre formation.

Nous remercions aussi nos familles à qui on doit ce que nous somme aujourd'hui grâce à leur amour, patience et leurs innombrables sacrifices

Nous voudrions aussi exprimer notre gratitude envers tous nos amis, collègues et toutes les personnes qui nous ont accordé leur soutien, tant par leur gentillesse que par leur dévouement.

MERCI

Dédicace :

Du profond de mon cœur, je dédie ce travail à tous ceux qui me sont chers,

A MES CHERS PARENTS

Pour leur patience, leur amour, leur soutien et leurs encouragements. Vous avez tout fait pour mon bonheur et ma réussite,

Aucune dédicace ne serait exprimée mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être, Que dieu vous préserve en bonne santé et vous accorde une longue vie.

A MES CHERES SOEURS

À Kahina Et son époux Ahmed, Fatiha et Kenza, mes piliers dans cette vie,

À mon neveu Aksil adore, source de joie et de bonheur,

Vous étiez toujours présents pour m'aider et m'encourager. Sachez que vous serez toujours dans mon cœur.

A tous mes meilleurs amis qui n'ont cessé de m'encourager et de me soutenir, ainsi à la personne avec qui j'ai un grand honneur de travailler et de partagé une très belle expérience qui est mon binôme « BELKAI Salah », pour son soutien moral et sa patience tout au long de ce travail, je te souhaite une vie plein de bonheur et de succès.

Thinhinane

Dédicace :

Je dédie ce travail tout d'abord à mes très chers parents, eux qui m'ont doté d'une éducation digne, quoi que je fasse ou que je dis, rien ne pourrait exprimer ma reconnaissance envers eux.

Ils ont toujours été là pour moi, ils m'ont soutenu et encouragé dans tout ce que j'ai entretenu dans ma vie et notamment dans mes études.

Leurs présences à mes coter a toujours été ma source de force pour affronter tout obstacle. Je suis et je serais toujours reconnaissant envers eux.

À mes très chers frères et sœur, ainsi que beau-frère et belle-sœur et leur enfant qui sont une source de joie et de bonheur et à toutes les personnes grandes de ma famille.

Sans oublier mes amies de loin comme de près qui partage mon quotidien, qui m'encourage et qui me soutiennent vers le bien de ma réussite.

Et pour finir j'adresse ma reconnaissance tout particulièrement à mon binôme avec qui j'ai eu le plaisir de travailler.

Salah

Résumé

Les systèmes d'informations, Internet et le réseau mondial qui nous interconnecte, jouent un rôle croissant dans la vie quotidienne et dans notre société en général. De ce fait, des attaques réalisées par des utilisateurs malveillants visant à exploiter les vulnérabilités de ces systèmes d'information sont de plus en plus efficaces. Donc la détection de ces attaques émerge comme un problème de recherche intéressant.

Dans ce travail, nous présentons la conception d'un modèle de prédiction d'attaque basé sur l'apprentissage profond (Deep Learning) ou nous avons proposé un modèle LSTM qui joue un rôle dans la classification.

La solution proposée a été implémentée et testée sur la base de données NSL-KDD qui est considérablement complète et riche. Les résultats de simulation sont prometteurs et de très bonne qualité.

Mots clés : attaque, vulnérabilité, prédiction, apprentissage profond, LSTM, classification, NSLKDD.

Summary

Information systems, the Internet and the global network that interconnects us, play an increasing role in daily life and in our society in general. As a result, attacks carried out by malicious users aimed at exploiting the vulnerabilities of these information systems are increasingly effective. So the detection of these attacks emerges as an interesting research problem.

In this work, we present the design of an attack prediction model based on Deep Learning or we have proposed an LSTM model that plays a role in classification.

The proposed solution has been implemented and tested on the NSL-KDD database which is considerably complete and rich. The simulation results are promising and of very good quality.
Keywords: attack, vulnerability, prediction, deep learning, LSTM, classification, NSLKDD.

Table des matières

Table des figures

Liste des tableaux

Introduction génerale

Les réseaux informatiques sont devenus beaucoup plus importants comparés à ce qu'ils étaient quelques années. Aujourd'hui, le monde presque entièrement connecté à un réseau internet, ce qui permet à l'humain de communiquer plus facilement, de bien gérer son temps et ses affaires sans effort, et de même pour les entreprises qui n'hésitent pas à mettre en place des réseaux informatiques pour faciliter la gestion de leurs infrastructures, ce qui rend les données échangées comme des ressources critiques que nous devons protéger à tout prix. C'est ce qui constitue de la sécurité informatique un enjeu crucial.

La sécurité d'un système informatique ou d'un réseau repose sur une définition d'un mécanisme de sécurité que ça soit avec des pare-feu, des antivirus, du cryptage. Cepandant, ces mécanismes ont des limites face au développement rapide des techniques de piratage. Le système de détection d'intrusion (IDS) est l'un des mécanismes important pour répondre à cette exigence. Les administrateurs réseau adaptent ce système afin d'empêcher les attaques malveillantes. Par conséquent, l'IDS est devenu un élément essentiel de la gestion de la sécurité. Grace à l'intelligence artificielle et ses différentes branches, l'IDS est capable de détecter des attaques aux sein d'un réseau, exemple avec des techniques de l'apprentissage automatique et notamment l'apprentissage profond.

En effet, Les techniques traditionnelles d'apprentissage automatique (la régression de gradient, les réseaux neuronaux artificiels et la machine à vecteurs de support) ne parviennent pas à apprendre des modèles séquentiels de données pour des prévisions précises. Elles sont imparfaites pour des scénarios complexes du monde réel. Contrairement aux techniques basées sur l'apprentissage automatique, les modèles de l'apprentissage profond jouent un rôle important et produisent une meilleure précision, ce qui est largement étudié dans différents domaines de la science des données tels que la synthèse vidéo, la classification d'images... etc [18]. Ce qui nous permet de l'utiliser pour concevoir un IDS capable de détecter une attaque et même de prédire sa classe, exemple une attaque de classe DDOS.

Pour répondre au problème de détection d'attaques, nous avons conçu et implémenté un modèle qui utilise un sous-type des réseaux de neurones récurrents (Recurent Neural Networks " RNN ") qui sont les réseaux de mémoire à long terme (Long Short Terme Memory " LSTM ") qui joue un rôle d'un décodeur permettant la classification des attaques. Notre travail est structuré comme suit :

- **Une introduction générale :** pour comprendre le but de notre travail.
- **Chapitre 01 :** qui se focalise sur des définitions de la sécurité informatique, quelques types d'attaques et même quelques mécanismes de sécurité.
- **Chapitre 02 :** où nous allons approfondir la définition des IDS, ses types ainsi que ses différentes architectures.
- **Chapitre 03 :** qui est composé de deux parties, la première partie, nous allons parler sur quelques notions de l'apprentissage automatique, l'apprentissage en pro-

fondeur et son fonctionnement. La deuxième partie nous allons définir l'architecture de notre IDS.

— **Chapitre 04 :** sera consacré pour la réalisation de notre architecture, les outils utilisés et le résultat de notre travail.

— **Conclusion générale :** pour terminer notre travail.

Chapitre 1

Généralités sur la sécurité informatique et les attaques réseaux

1.1 Introduction

L'échange de données dans un réseau engendre d'énormes risques et menaces, c'est pour cela la sécurité informatique existe. Cette dernière consiste à sécuriser contre toute tentative d'accès non autorisé, divulgation ou destruction et de minimiser la vulnérabilité d'un système contre des menaces accidentelles ou intentionnelles. Les ordinateurs connectés à un réseau externe (internet) ont des failles qui peuvent être exploitées par des pirates pour réaliser leurs attaques.

Les conséquences de ces attaques peuvent être lourdes (vol d'information, accès à des informations confidentielles...). Par conséquent, il est important de prendre des mesures de sécurité afin de se protéger contre ces attaques.

Tout au long de ce chapitre, notre intérêt se porte sur les différentes attaques et les moyens mis à la disposition pour sécuriser les données informatiques.

1.2 La sécurité

1.2.1 Définition de la sécurité informatique :

La sécurité informatique est un terme large qui réunit l'ensemble des moyens mis en œuvre afin de réduire la vulnérabilité d'un système contre les menaces accidentelles ou intentionnelles. Il convient d'identifier les exigences fondamentales en sécurité informatique. Elles caractérisent ce à quoi s'attendent les utilisateurs de systèmes informatiques en regard de la sécurité.

1.2.2 Objectif de la sécurité

La sécurité de l'information est définie par " le bien-être de l'information et de l'infrastructure dans la possibilité de vols, modification ou perturbation de l'information, et que les services assurent au minimum ". La sécurité informatique doit contribuer à satisfaire les critères (objectifs) suivants :

- **Confidentialité :** C'est l'assurance que les informations seront accessibles seulement aux personnes autorisées à avoir accès.
- **Intégrité :** C'est l'assurance que les informations restent telles qu'elles sont et ne subissent aucune modification non autorisée.

— **Disponibilité :** La disponibilité est l'assurance que les systèmes responsables de la prestation, le stockage et le traitement des informations sont accessibles en cas de besoin par les utilisateurs autorisés.
— **Authenticité :** Réfère aux caractéristiques d'une communication, un document ou n'importe quelle donnée est d'assuré que ces données sont authentiquées à l'original et non modifie ou falsifié. Son rôle principal est d'assuré que l'utilisateur est celui qui prétend être que les messages sont authentiques.
— **Non répudiation :** C'est l'assurance que si un utilisateur fait une tache ou envoie un message il ne peut pas nier l'authenticité de leur signature.[27]

1.3 La sécurité réseau

La sécurité d'un réseau informatique peut être définie sur trois niveaux :
— **Niveau physique**
— **Niveau logique**
— **Niveau réseau**

1.3.1 Sécurité au niveau physique

La sécurité physique est un aspect fondamental de tout type de sécurité pour garantir l'intégrité, la confidentialité et la disponibilité des informations. Si quelqu'un réussit à accéder au système informatique de l'entreprise, il peut l'endommager ou même le détruire.

La sécurité physique consiste aussi en l'usage de barrières, alarmes, serrures et autres contrôles physiques permettant de conditionner l'accès physique aux locaux, aux ordinateurs et aux équipements. Ces mesures sont nécessaires pour protéger les ordinateurs, leur contenu et les autres ressources matérielles contre l'espionnage, le vol et la destruction accidentelle ou intentionnelle.[13]

1.3.2 Sécurité au niveau logique

La sécurité logique complète la sécurité physique. Sa fonction est de protéger les systèmes, processus et programmes de protection des logiciels et des données, ainsi que l'accès ordonné et autorisé des utilisateurs aux informations. La sécurité logique comprend toutes les mesures établies par la direction pour minimiser les risques de sécurité associés aux opérations quotidiennes effectuées à l'aide des technologies de l'information, comme le vol d'information, la perte de données, l'entrée de virus, la modification non autorisée de données, les attaques de réseaux, etc. La sécurité logique vise à préserver la confidentialité, l'intégrité, l'authenticité et la disponibilité des données.[30]

1.3.3 Sécurité au niveau réseau

Un autre volet très important de la sécurité réseau est l'accès au réseau. Il relève beaucoup d'interrogations :
Qui a accès au réseau ? Quel est son niveau d'accès (droits sur les fichiers et répertoires, privilèges sur les services ...) une fois connecté au réseau ? Que faire pour limiter l'accès à certaines ressources du réseau (serveurs par exemple) ? Il serait très dangereux voire inimaginable d'autoriser l'accès aux ressources internes du réseau à tout le public (quel qu'il soit). Il existe plusieurs mécanismes pour répondre à ces questions :

— Mise en place de serveur d'authentification (exemple serveur radius) pour identifier tout utilisateur voulant se connecter. Il existe deux niveaux d'authentification : authentification simple et forte. L'authentification simple se base sur l'usage de login et de mots de passe. Alors que l'authentification forte se base sur les certificats électroniques, les cartes à puce, l'index du doigt ... avec la création de tunnels sécurisés lors de l'authentification.
— Mise en place de Firewall. Les firewalls permettent de filtrer l'accès au réseau.
— Mise en place de DMZ (zone démilitarisée). Une DMZ permet de partager certaines Ressources du réseau local avec l'extérieur. Cette zone est accessible depuis l'intérieur comme l'extérieur du réseau.

1.4 Les risques et les menaces

1.4.1 Les risques

Les risques se mesurent en fonction de deux critères principaux : la vulnérabilité et la sensibilité. La vulnérabilité : Désigne le degré d'exposition à des dangers. La sensibilité : Désigne le caractère stratégique d'un composant du réseau. Celui-ci peut être très sensible, vu son caractère stratégique mais quasi invulnérable, grâce à toutes les mesures de protection qui ont été prises pour le prémunir contre la plupart des risques.
Deux types de risques :

— **Le risque structurel :** dépend de l'organisation de l'entreprise.
— **Le risque accidentel :** indépendant de tous les facteurs de l'entreprise.

Enfin, selon les niveaux de sensibilité et de vulnérabilité, on distingue souvent quatre niveaux de risques, selon qu'ils sont acceptables, courants, majeurs ou inacceptables.

— **Acceptables :** pas de conséquences graves pour les utilisateurs du réseau. Ex panne électrique, perte de liaison, engorgement...
— **Courants :** pas de préjudices graves au réseau, on peut réparer facilement. Ex gestion du réseau, mauvaise configuration, erreur utilisateur...
— **Majeurs** dus à des facteurs graves et qui causent de gros dégâts mais récupérables. Ex foudre qui tombe sur un routeur...
— **Inacceptables :** fatals pour l'entreprise, ils peuvent entraîner son dépôt de bilan. Ex Perte ou corruption des informations importantes.

1.4.2 Les intrusions

C'est réaliser une attaque ou une menace pour un système d'informatique, pour que ce dernier ne soit plus en sécurité.

1.4.3 Les attaques

1.4.3.1 Définition

Une attaque peut être définie par : n'importe quelle action qui tente d'exploiter une (ou plusieurs) vulnérabilité(s) dans un système pour violer un ou plusieurs besoins de sécurité et il est généralement préjudiciable.

1.4.3.2 Anatomie d'une attaque

Une attaque est souvent décrite à l'aide des 5"p" :

— **Probe (Analyser)** : Dans un premier temps, une personne mal intentionnée va chercher les failles pour pénétrer le réseau.
— **Penetrate (Pénétrer)** : Une fois une ou plusieurs failles identifiées, le pirate va chercher à les exploiter afin de pénétrer au sien du SI.
— **Persist (Pérenniser)** : Le réseau infiltré, le pirate cherchera à y revenir facilement. Pour cela, il installera par exemple des back doors. Cependant, en général, il corrigera la faille par laquelle il s'est introduit afin de s'assurer qu'aucun autre pirate n'exploitera sa cible.
— **Propagate (Propager)** : Le réseau est infiltré, l'accès est pérein. Le pirate pourra alors explorer le réseau et trouver de nouvelles cibles qui l'intéresseraient.
— **Paralyze (Paralyser)** : Les cibles identifiées, le pirate va agir et nuire au sein du SI.[28]

1.4.3.3 Catégorie des attaques

Il existe quatre catégories d'attaques :
— **Interception :** Une tierce partie non autorisée obtient un accès à un actif. C'est une attaque portée à la confidentialité. Il peut s'agir d'une personne, d'un programme ou d'un ordinateur. Une écoute téléphonique dans le but de capturer des données sur un réseau, ou la copie non autorisée de fichiers ou de programmes en sont des exemples.
— **Modification :** Une tierce partie non autorisée obtient accès à un actif et le modifie. Il s'agit d'une attaque portée à l'intégrité. Changer des valeurs dans un fichier de données, altérer un programme de façon à bouleverser son comportement ou modifier le contenu de messages transmis sur un réseau sont des exemples de telles attaques.
— **Fabrication :** Une tierce partie non autorisée insère des faux objets dans un système. C'est une attaque portée à l'authenticité. Il peut s'agir de l'insertion de faux messages dans un réseau ou l'ajout d'enregistrement à un fichier.
— **Interruption :** Un actif du système est détruit ou devient indisponible ou inutilisable. C'est une attaque portée à la disponibilité. La destruction d'une pièce matérielle, la coupure d'une ligne de communication, ou la mise hors service d'un système de gestion de fichiers en sont des exemples.[23]

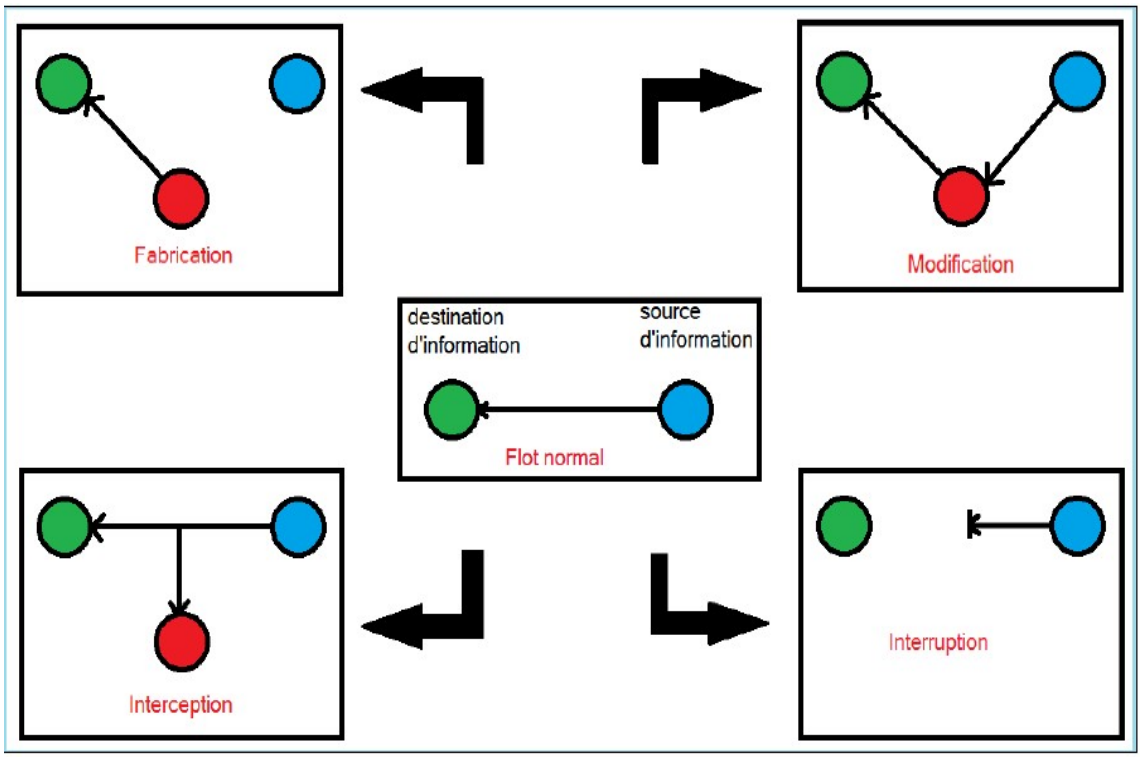

FIGURE 1.1 – Catégorie des attaques [31]

1.4.3.4 Types d'attaques

Les personnes malveillantes utilisent plusieurs techniques d'attaques qui peuvent être regroupées en trois familles différentes :

— **Les attaques directes :**
C'est la plus simple des attaques. Le hacker attaque directement sa victime à partir de son ordinateur. La plupart des "script kiddies" utilisent cette technique. En effet, les programmes de hack qu'ils utilisent ne sont que faiblement paramétrables, et un grand nombre de ces logiciels envoient directement les paquets à la victime. Les attaques directes se produisent uniquement lorsque votre ordinateur est connecté à Internet ou à un réseau local.

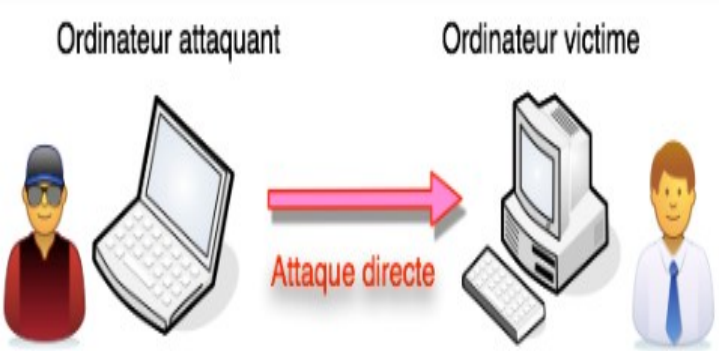

FIGURE 1.2 – Attaque direct

— **Les attaques indirectes par rebond :**
Cette attaque est très prisée des hackers. En effet, le rebond a deux avantages :

- Masquer l'identité (l'adresse IP) de l'hacker.
- Éventuellement, utiliser les ressources de l'ordinateur intermédiaire car il est plus puissant (CPU, bande passante...) pour attaquer.

Le principe en lui-même, est simple : Les paquets d'attaque sont envoyés à l'ordinateur intermédiaire, qui répercute l'attaque vers la victime. D'où le terme de rebond.

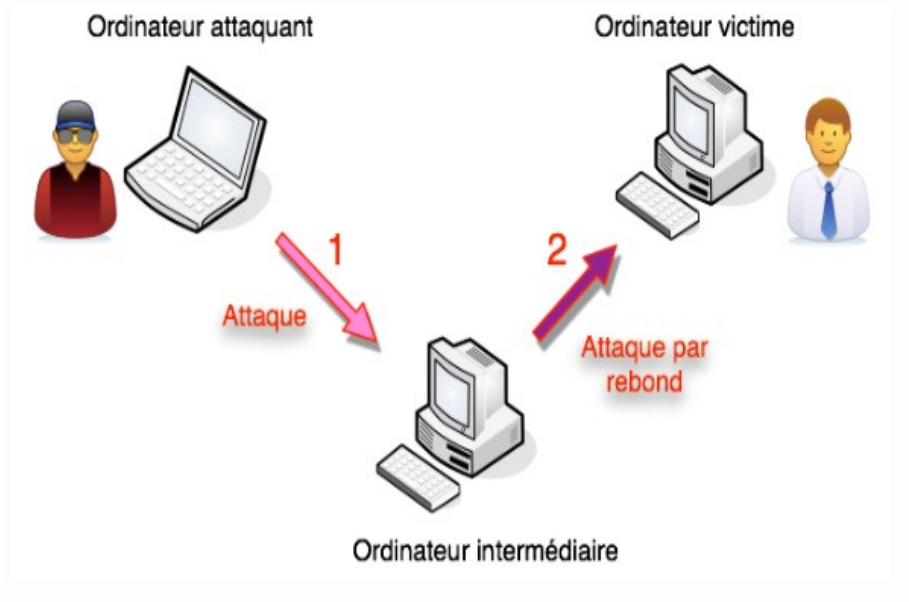

FIGURE 1.3 – Attaque indirect par rebond

— **Les attaques indirectes par réponse :** Cette attaque est un dérivé de l'attaque par rebond. Elle offre les mêmes avantages, du point de vue du hacker. Mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête. Et c'est cette réponse à la requête qui va être envoyée à l'ordinateur victime. C'est idéal pour accéder aux informations confidentielles.[29]

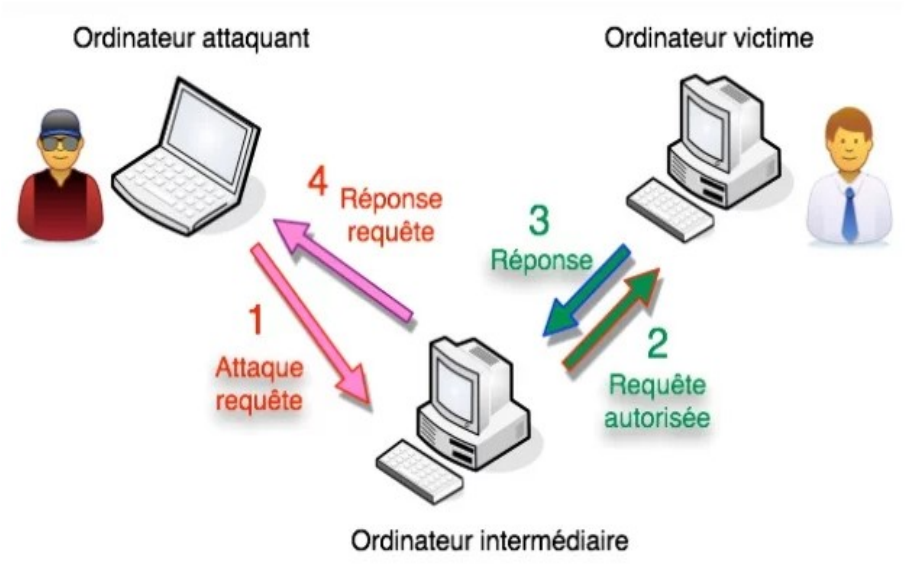

FIGURE 1.4 – Attaque indirect par réponse.[16]

1.4.3.5 Quelques attaques courantes

- **Attaques par déni de service (DoS) et par déni de service distribué (DDoS) :** Une attaque par déni de service submerge les ressources d'un système afin que ce dernier ne puisse pas répondre aux demandes de service. Une attaque DDoS vise elle aussi les ressources d'un système, mais elle est lancée à partir d'un grand nombre d'autres machines hôtes infectées par un logiciel malveillant contrôlé par l'attaquant.
  À la différence des attaques conçues pour permettre à un attaquant d'obtenir ou de faciliter des accès, le déni de service ne procure pas d'avantage direct aux attaquants. Le déni de service est une satisfaction en soi pour certains pirates. Il existe différents types d'attaques DoS et DDoS ; les plus courantes sont les attaques SYN flood, les attaques teardrop, les attaques par rebond, le ping de la mort et les botnets.

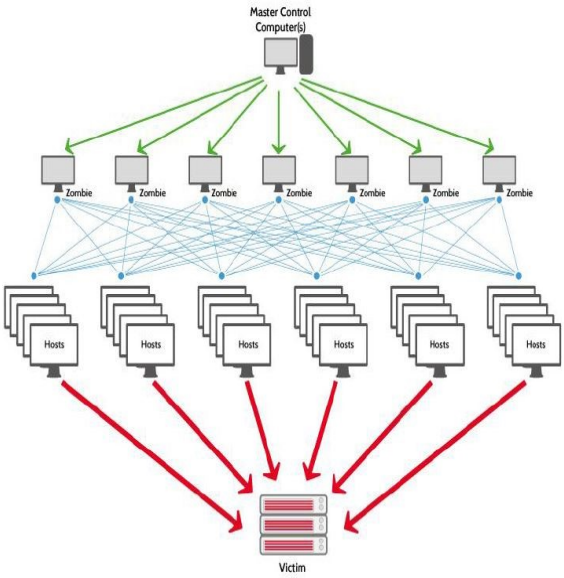

FIGURE 1.5 – Attaques par déni de service (DoS).

- **Attaque de l'homme au milieu (MitM) :** Une attaque de l'homme du milieu est un pirate qui s'insère dans les communications entre un client et un serveur.

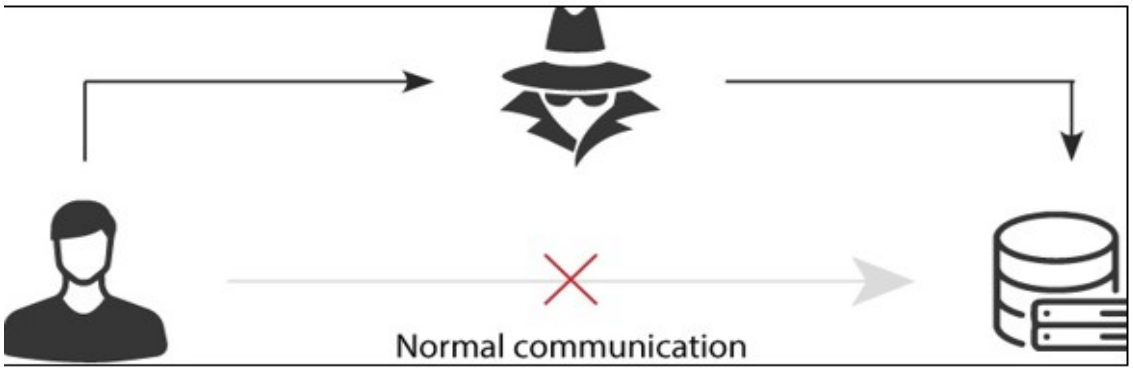

FIGURE 1.6 – Attaque de l homme au milieu (MitM)

Voici quelques types courants d'attaques de l'homme du milieu :

(a) **Détournement de session :** dans ce type d'attaque MitM, un attaquant détourne une session entre un client de confiance et un serveur réseau. L'ordinateur attaquant substitue son adresse IP au client de confiance pendant que le serveur poursuit la session, croyant qu'il communique avec le client. Par exemple, l'attaque pourrait se dérouler ainsi :

i. Un client se connecte à un serveur.

ii. L'ordinateur de l'attaquant prend le contrôle du client.

iii. L'ordinateur de l'attaquant déconnecte le client du serveur.

iv. L'ordinateur de l'attaquant remplace l'adresse IP du client par sa propre adresse IP et son propre nom de domaine et usurpe les numéros de séquence du client.

v. L'ordinateur de l'attaquant poursuit le dialogue avec le serveur, le serveur croit qu'il communique toujours avec le client.

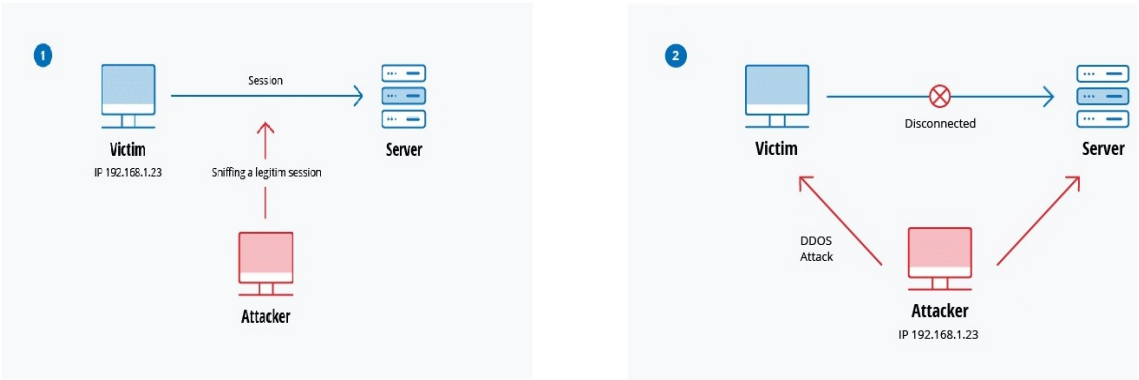

FIGURE 1.7 – Détrounement de session.

(b) **Usurpation d'IP :** Un pirate peut utiliser l'usurpation d'adresse IP pour convaincre un système qu'il communique avec une entité connue et fiable afin de lui donner accès au système. Le pirate envoie à un hôte cible un paquet contenant l'adresse IP source d'un hôte connu et fiable au lieu de sa propre adresse IP source. Il est possible que l'hôte cible accepte le paquet et agisse en conséquence.

(c) **Relecture :** Une attaque par rejeu se produit lorsqu'un attaquant intercepte et enregistre d'anciens messages, puis essaie plus tard de les envoyer, se faisant passer pour l'un des participants. Ce type d'attaque peut facilement être contré avec un horodatage des sessions ou un nonce (nombre ou chaîne aléatoire variant avec le temps).

Il n'existe actuellement pas de technologie unique ni de configuration unique permettant de prévenir toutes les attaques de l'homme du milieu. En général, le chiffrement et les certificats numériques offrent une protection efficace contre les attaques de ce type, assurant à la fois la confidentialité et l'intégrité des communications. Mais une attaque de l'homme du milieu peut être injectée au cœur des communications de telle sorte que le chiffrement ne soit d'aucun secours.

- **Attaques phishing et spear phishing :** L'hameçonnage (phishing) consiste à envoyer des e-mails qui semblent provenir de sources fiables dans le but d'obtenir des informations personnelles ou d'inciter les utilisateurs à faire quelque chose.

Cette technique combine ingénierie sociale et stratagème technique. Elle peut impliquer une pièce jointe à un e-mail, qui charge un logiciel malveillant sur votre ordinateur. Elle peut également utiliser un lien pointant vers un site Web illégitime qui vous incite à télécharger des logiciels malveillants ou à transmettre vos renseignements personnels.

Le harponnage (spear phishing) est un hameçonnage très ciblé. Les attaquants prennent le temps de mener des recherches sur leurs cibles et de créer des messages personnels et pertinents. Pour cette raison, le harponnage peut être très difficile à identifier et encore plus difficile à combattre. L'un des moyens les plus simples pour un pirate de mener une attaque de harponnage est d'usurper une adresse électronique, c'est-à-dire de falsifier la section " De " d'un e-mail, pour vous donner l'impression que le message a été envoyé par une personne que vous connaissez, par exemple votre supérieur ou une entreprise partenaire. Une autre technique que des escrocs utilisent pour ajouter de la crédibilité à leur histoire est le clonage de sites Web : ils imitent des sites Web légitimes pour vous inciter à entrer des informations personnelles identifiables ou des identifiants de connexion.

- **Attaque par Drive by Download :**
  Les attaques par téléchargement furtif sont une méthode courante de propagation des logiciels malveillants. Les pirates recherchent des sites Web non sécurisés et insèrent un script malveillant dans le code HTTP ou PHP de l'une des pages. Ce script peut installer des logiciels malveillants directement sur l'ordinateur d'un visiteur du site, ou rediriger celui-ci vers un site contrôlé par les pirates. Des téléchargements furtifs peuvent survenir lors de la visite d'un site Web ou de l'affichage d'un e-mail ou d'une fenêtre pop-up. À la différence de nombreux autres types d'attaques informatiques, un téléchargement furtif ne nécessite pas qu'un utilisateur déclenche activement l'attaque - nul besoin de cliquer sur un bouton de téléchargement ou d'ouvrir une pièce jointe malveillante pour être infecté. Un téléchargement furtif peut profiter d'une application, d'un système d'exploitation ou d'un navigateur Web contenant des failles de sécurité dues à des mises à jour infructueuses ou à une absence de mise à jour.
- **Attaque par mot de passe :**
  Les mots de passe étant le mécanisme le plus couramment utilisé pour authentifier les utilisateurs d'un système informatique, l'obtention de mots de passe est une approche d'attaque courante et efficace. Le mot de passe d'une personne peut être obtenu en fouillant le bureau physique de la personne, en surveillant la connexion au réseau pour acquérir des mots de passe non chiffrés, en ayant recours à l'ingénierie sociale, en accédant à une base de données de mots de passe ou simplement en devinant.
- **Attaque par injection SQL :**
  L'injection SQL est devenue un problème courant qui affecte les sites Web exploitant des bases de données. Elle se produit lorsqu'un malfaiteur exécute une requête SQL sur la base de données via les données entrantes du client au serveur. Des commandes SQL sont insérées dans la saisie du plan de données (par exemple, à la place du nom d'utilisateur ou du mot de passe) afin d'exécuter des commandes SQL prédéfinies. Un exploit d'injection SQL réussi peut lire les données sensibles de la base de données, modifier (insérer, mettre à jour ou supprimer) les données de la base de données, exécuter des opérations d'administration de la base de données (par exemple la fermer), récupérer le contenu d'un fichier spécifique, et, dans certains cas, envoyer des commandes au système d'exploitation.

  Même si les utilisateurs qui saisissent correctement leur numéro de compte, cela

laisse un passage aux attaquants. Si, par exemple, quelqu'un décidait de fournir un numéro de compte "' or '1' = '1'", cela donnerait une chaîne de requête comme celle-ci : "SELECT * FROM users WHERE account = " or '1' = '1';" Puisque '1' = '1' est toujours évalué comme TRUE, la base de données renvoie les données de tous les utilisateurs au lieu d'un seul.

La vulnérabilité à ce type d'attaque informatique est liée au fait que SQL ne fait aucune distinction réelle entre le plan de contrôle et le plan de données. Pour cette raison, les injections SQL fonctionnent surtout si un site Web utilise SQL dynamique. De plus, les injections SQL sont très courantes avec les applications PHP et ASP en raison de la prévalence des anciennes interfaces fonctionnelles. Les applications J2EE et ASP.NET sont moins sensibles aux injections SQL en raison de la nature des interfaces programmatiques disponibles.

- **Attaque XSS (Cross-site scripting)**

Les attaques XSS utilisent des ressources Web tierces pour exécuter des scripts dans le navigateur Web de la victime ou dans une application pouvant être scriptée. Plus précisément, l'attaquant injecte un JavaScript malveillant dans la base de données d'un site Web. Lorsque la victime demande une page du site Web, ce dernier transmet la page à son navigateur avec le script malveillant intégré au corps HTML. Le navigateur de la victime exécute ce script, qui envoie par exemple le cookie de la victime au serveur de l'attaquant, qui l'extrait et l'utilise pour détourner la session. Les conséquences les plus graves se produisent lorsque XSS sert à exploiter des vulnérabilités supplémentaires. Ces vulnérabilités peuvent non seulement permettre à un attaquant de voler des cookies, mais aussi d'enregistrer les frappes de touches et des captures d'écran, de découvrir et de collecter des informations réseau et d'accéder et de contrôler à distance l'ordinateur de la victime.

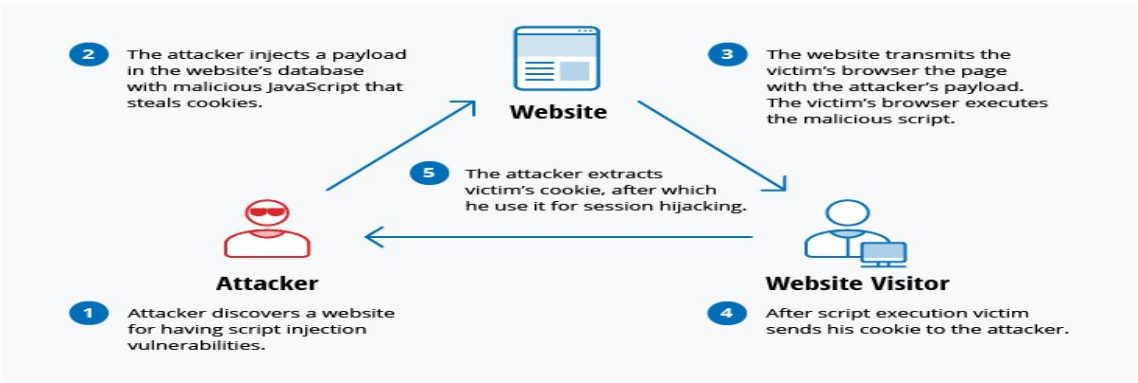

FIGURE 1.8 – Attaque XSS.

- **Attaque par des logiciels malveillants**

Un logiciel malveillant peut être décrit comme un logiciel indésirable installé dans votre système sans votre consentement. Il peut s'attacher à un code légitime et se propager, se cacher dans des applications utiles ou se reproduire sur Internet. Voici quelques-uns des types de logiciels malveillants les plus courants :

(a) **Macro-virus :** Ils infectent des applications comme Microsoft Word ou Excel. Les macrovirus s'attachent à la séquence d'initialisation de l'application. Lorsque l'application est ouverte, le virus exécute ses instructions avant de transférer le contrôle à l'application. Le virus se réplique et s'attache à d'autres codes du système informatique.

(b) **Infecteurs de fichiers :** Ils s'attachent généralement à des codes exécutables, comme les fichiers .exe. Le virus est installé au chargement du code. Une autre

version d'infecteur de fichiers s'associe à un fichier en créant un fichier de virus portant le même nom, mais avec une extension .exe. Ainsi, lorsque le fichier est ouvert, le code du virus s'exécute.

(c) **Infecteurs de système ou de secteur d'amorçage :** Ils s'attachent au secteur d'amorçage principal des disques durs. Quand le système démarre, il consulte le secteur d'amorçage et charge le virus en mémoire, où celui-ci peut se propager à d'autres disques et ordinateurs.

(d) **Chevaux de Troie :** Ce sont des programmes qui se cachent dans un programme utile et qui ont généralement une fonction malveillante. Une différence majeure entre les virus et les chevaux de Troie est que ces derniers ne se répliquent pas d'eux-mêmes. En plus de lancer des attaques contre un système, un cheval de Troie peut établir une porte dérobée qui peut être exploitée par des attaquants. Par exemple, un cheval de Troie peut être programmé pour ouvrir un port à numéro élevé afin que le pirate l'utilise pour écouter puis exécuter une attaque.

(e) **Bombe logique :** Il s'agit d'un type de logiciel malveillant ajouté à une application et qui est déclenché par un événement spécifique, comme une condition logique ou une date et une heure spécifiques.

(f) **Vers :** Ils diffèrent des virus en ce qu'ils ne s'attachent pas à un fichier hôte, ce sont des programmes autonomes qui se propagent sur les réseaux et les ordinateurs. Les vers se propagent généralement via les pièces jointes aux e-mails : l'ouverture de la pièce jointe active le programme du ver. Pour un exploit typique, le ver envoie une copie de lui-même à chaque contact e-mail de l'ordinateur infecté. En plus de mener des activités malveillantes, un ver qui se propage sur Internet et surcharge les serveurs de messagerie peut entraîner des attaques par déni de service contre des nœuds du réseau.

(g) **Rançongiciels (ransomware) :** Il s'agit d'un type de logiciel malveillant qui bloque l'accès aux données de la victime et menace de les publier ou de les supprimer à moins qu'une rançon ne soit versée. Si un rançongiciel simple peut verrouiller le système d'une manière qui n'est pas difficile à réparer pour une personne bien informée, un rançongiciel plus avancé utilisera une technique appelée extorsion crypto virale, qui chiffre les fichiers de la victime de manière à les rendre presque impossible à récupérer sans la clé de déchiffrement.

FIGURE 1.9 – 14 Mars 2021 acer victime de Ransomware.

(h) **Logiciels espions (spyware) :** Ce sont des programmes installés pour recueillir des informations sur les utilisateurs, leurs ordinateurs ou leurs habitudes de navigation. Ils surveillent à votre insu tout ce que vous faites et envoient les données à un utilisateur distant. Ils peuvent également télécharger et installer d'autres programmes malveillants depuis Internet. Les logiciels espions fonctionnent comme les logiciels publicitaires, mais il s'agit généralement d'un pro-

gramme distinct qui s'installe à votre insu lorsque vous installez une application gratuite.[5]

1.4.3.6 Présentation de Quelques Outils D'intrusion.

Après avoir cité quelques attaques nous allons maintenant parler de certains outils d'intrusion permettant de mener ou préparer des attaques.
Les outils d'intrusion sont des moyens (logiciels) qui permettent de préparer une intrusion. Il en existe une variété. Leurs méthodes d'action varient suivant les concepteurs et les OS sur Lesquels ils tournent. Ici Nous nous intéresserons uniquement aux plateformes les plus utilisées : Linux et Windows. Les outils d'intrusion peuvent être classés en deux familles :

- Les Outils intrusion Passifs : les scanners, les sniffer.
- Les Outils intrusion Actifs.

Ces outils sont très simples mais très efficaces.

1.4.3.6.1 Outils d'intrusion passif Le principe de ces outils est de collecter le maximum d'information sans jamais accéder directement au système de la cible et ne pas être détecté par le client. Ils appartiennent à la famille des scanners. L'intérêt du scanner est très simple : il permet de trouver une fois qu'il est lancé, dans un délai très court, tous les ports ouverts sur une machine distante. Il existe différents types de scanners, certains se contentent juste de donner : la liste des ports ouverts, le type et la version de l'OS tournant sur le serveur, d'autres scanners permettent de tester différentes failles connues sur ces services : Nmap, Tcpdump, Netstat, Whireshark.

- **Nmap :** Nmap est un utilitaire qui tourne sous Linux et qui est intégré au système. Nmap donne un aperçu assez complet des différents services s'exécutant sur une machine (même en local) dans un temps assez bref.

  Pour connaître les ports ouverts sur une machine, Nmap procède à l'envoi de paquets sur tous les ports de cette machine et analyse les réponses. Il existe différents types de scans, donc différents types d'envois et de réponses. On peut distinguer les scans utilisant les protocoles TCP, UDP. Suivant le protocole indiqué ce sont les informations relatives aux services tournant avec ce protocole qui sont affichés.
- **Wireshark :** Wireshark est un outil performant permet de capturer et analyser les différents paquets qui circule dans le réseau. Wireshark est un logiciel d'analyse réseau (sniffer) permettant de visualiser l'ensemble des données transitant sur la machine qui l'exécute, et d'obtenir des informations sur les protocoles applicatifs utilisés. Les octets sont capturés en utilisant la librairie réseau PCAP, puis regroupés en blocs d'informations et analysés par le logiciel.

1.4.3.6.2 Outils d'intrusion actif Ces outils bien que souvent, étant des moyens d'administration réseau, peuvent être utilisées pour attaquer des réseaux à distance. " Actifs " par ce qu'ils ont une vocation " offensive ". Il suffit juste pour l'attaquant de connaître entre autres l'adresse IP de la cible, le numéro de port sur lequel tourne le service, un compte d'utilisateur ayant accès au service spécifié et son mot de passe. Certains de ces outils sont souvent intégrés au système d'exploitation (Linux, Windows). Parmi ces outils nous pouvons citer : Telnet, ssh, Ping, Traceroute.

- **Telnet et ssh**
  Ces deux protocoles permettent d'attaquer des machines à distance. Ils sont utilisés pour administrer à distance des systèmes informatiques mais peuvent être utilisés à d'autres fins. Une fois que ces deux services (telnet et ssh) sont démarrés chez la machine cible, il suffit seulement au pirate de connaître un login et son

mot de passe sur la machine ciblée ! Une fois la connexion réussie le pirate peut prendre le contrôle total du système (machine) : il peut se déplacer librement dans l'arborescence, changer le mot de passe de l'administrateur, détruire ou voler des fichiers confidentiels, changer la configuration du système ... Sous Windows il existe plusieurs utilitaires permettant de faire du ssh comme putty.

— **Metasploit**
Ce méta-logiciel bien nommé est comme une arbalète : visez votre cible, choisissez votre exploit, sélectionnez une charge utile et tirez. Son but est de fournir des informations sur les vulnérabilités de systèmes informatiques, d'aider à la pénétration et au développement de signatures pour les systèmes de détection d'intrusion. Metasploit automatise de grandes quantités de tâches fastidieuses. C'est vraiment " le framework de test d'intrusion le plus utilisé au monde ", comme le proclame son site Web. Metasploit est un projet open-source, mais il est possible de bénéficier du support commercial de Rapid7. Il est un incontournable pour les défenseurs qui veulent protéger leurs systèmes des attaques.

1.4.4 Les failles de sécurité.

Une faille de sécurité ou vulnérabilité, désigne en informatique toute faiblesse d'un système sa peut être comme un " trou ", un disfonctionnement ou une insuffisance dans le dispositif de sécurité existant, qui permettrait à une personne potentiellement malveillante d'altérer le fonctionnement normal du système ou encore d'accéder à des données non autorisées. Il existe deux types de failles : actives et passives.

1.4.4.1 Failles actives :

On peut parler de faille active lorsque celle-ci vient directement de l'intérieur du réseau. " Active " car elle est provoquée par l'action des utilisateurs internes du réseau. Elle peut résulter de plusieurs facteurs :

— Non-respect de la politique de sécurité définie dans le réseau, par les utilisateurs internes. Les exemples sont nombreux : installations de programmes non autorisés sur les postes clients ; tentatives de violation de niveau d'accès aux ressources ou services du réseau.

— Erreurs de manipulations commises lors du traitement de certains fichiers. Par exemple fichiers de configuration des services.

1.4.4.2 Failles Passives :

On peut parler de faille Passive lorsque celle-ci est provoquée depuis l'extérieur indépendamment des actions des utilisateurs internes du réseau. " Passive " par ce qu'elle résulte d'une action extérieure au réseau. Elle peut être le résultat de plusieurs actions telles que : l'envoi de programmes malveillants, espions ou de virus vers les machines cibles (du Réseau interne). Ces actions ont pour objectif final de fournir des informations dont a besoin le pirate pour passer à l'attaque. Exemples : Le Cheval de Troyes.

La détection d'une faille dans la sécurité d'un réseau informatique a pour étape suivante L'exploitation de cette faille. L'exploitation de cette faille constitue en quelque sorte le début de la véritable Intrusion réseau et ce qui produit une menace.

1.4.5 Les portes dérobées.

Une porte dérobée ou backdoor en anglais est un programme informatique malveillant installer sur une machine, il s'exécute en arrière-plan pour donner aux pirates un accès à distance non autorisé à un ordinateur infecté en exploitant les vulnérabilités du système. La porte dérobée représente une réelle menace pour la sécurité des systèmes.

1.4.6 Les menaces

Une menace est un événement, d'origine accidentelle ou délibérée, capable s'il se réalise de causer un dommage au sujet étudié. Le réseau informatique comme tout autres réseaux sont en proie à des menaces de toutes sortes qu'il convient de recenser. On peut également classer les menaces en deux catégories.

— **Les menaces passives :** consistent essentiellement à copier ou à écouter l'information sur le réseau, elles nuisent à la confidentialité des données. Dans ce cas, celui qui prélève une copie n'altère pas l'information elle-même.

— **Les menaces actives :** consistent à altérer des informations ou le bon fonctionnement d'un service.

1.4.7 Les hackers

1.4.7.1 Ces quoi un hacker ?

On peut donner plusieurs définitions aux hackers : Celui qui pirate ou une personne inexpérimentée ou non qualifiée dans une activité particulière. Un expert en programmation et en résolution de problèmes avec un ordinateur, une personne qui accède illégalement et parfois falsifie des informations dans un système informatique.

1.4.7.2 Type de hackers.

Principalement on a trois types de hackers qui ont de large connaissance et des compétences extraordinaires :

- **Le White Hat Hacker :**ce sont des gens qui ont l'intention d'aider légalement les gens, connues généralement sous le nom de d'analyste de sécurité. La plupart des organisations possède un analyste pour protéger leur system d'information.

FIGURE 1.10 – White hat hacker

- **Le Black Hat Hacker :** ils utilisent leur compétence pour découvrir les failles des system de défense, ce sont des gens dangereux, ils attaquent illégalement pour des buts précis, ils attaquent généralement au site gouvernemental.

FIGURE 1.11 – Black hat hacker

- **Le Grey Hat Hacker :**sont les individus qui travaillent dans les deux cas le défensive et l'offensive, ils mélangent entre le black hats et le white hats comme il peut aider des hackers à découvrir les failles d'un system de sécurité il peut aussi aider un développeur à sécuriser un produit.

FIGURE 1.12 – Grey hat hacker

Ce n'est seulement ça il existe d'autre types d'hacker dangereux qu'on citera quelques-uns si dessous :

- **Script-kiddies :** ce sont des pirates non qualifie qui compromettent des systèmes en utilisant des utilitaires développés par des vrais hackers
- **Hacktivistes :** Les hacktivistes (contraction d'hacker et activiste) qui agissent pour une cause souvent politique, idéologique, sociale ou religieuse. Ils peuvent par exemple chercher à attirer l'attention du public sur un problème en révélant des informations sensibles sur une cible telle qu'une entreprise ou un gouvernement. Certains groupes d'hacktivists sont très connus. On peut citer Anonymous, LulzSec.

FIGURE 1.13 – Hacktiviste

- **Spy hackers :** sont des employés des organisations qui sont recruté pour pénétrer les systèmes des adversaires et récupérer leur donnée classée secret.
- **Cyber terroriste :** sont des groupes organiser et former par des organisations terroristes motivé par une cause politique, religieuse... etc. c'est le type le plus dangereux par ce qu'ils ne peuvent pas seulement pirater un site web mais commettre des actes plus graves comme meurtre et des attentats...etc.

1.4.7.3 L'objectif des hackers

Un hacker ou un pirate informatique peut être motivé de plusieurs manières :

— Pour certain c'est un défi pour voir jusqu'à quel point il peut aller.
— Pour acquérir plus de connaissance et d'expérience ou juste pour faire des choses illégales.
— Avec des buts tracés à l'avance comme volé des informations critiques, numéro de la carte du crédit, mot de passe... etc.

1.5 Mécanismes de sécurité

1.5.1 Pare-feu(firewall)

Un pare-feu est un logiciel et/ou un matériel permettant de faire respecter la politique de sécurité du réseau, Son objectif est de protéger le réseau interne contre les accès et actions non autorisés en provenance de l'extérieur, en filtrant le trafic entrant. Le firewall peut également filtrer le trafic sortant. C'est comme un mur qui empêche une personne sans autorisation d'accéder à un réseau de données.

1.5.2 Cryptographie

La cryptographie est la science qui utilise les mathématiques pour le cryptage et le décryptage de données. La cryptographie des informations qui transitent par le réseau est utilisée pour assurer la confidentialité, l'intégrité et l'authenticité des transactions et du courrier électronique tout en utilisant un algorithme de chiffrement, Chiffrer un message consiste de le rendre incompréhensible, sauf pour celui qui possède le moyen (une clé) de le déchiffrer.
Il existe deux types de cryptage :
- Cryptage Symétrique : la même clé utiliser pour le cryptage est utiliser pour le décryptage tes le que le AES, DES, 3DES... etc.
- Cryptage Asymétrique : Ce type utilise de clé une qui est public qui est connue par tout le monde est une autre privé, tous ce qui est Crypter par la clé public du destinataire ne peut être décrypté qu'avec la clé privé du destinataire. Voice quelque exemple de cryptage asymétrique : RSA, DSA, DH, El Gamal... etc.
Le principe de ce qui vient d'être énoncé peut se résumer ainsi :

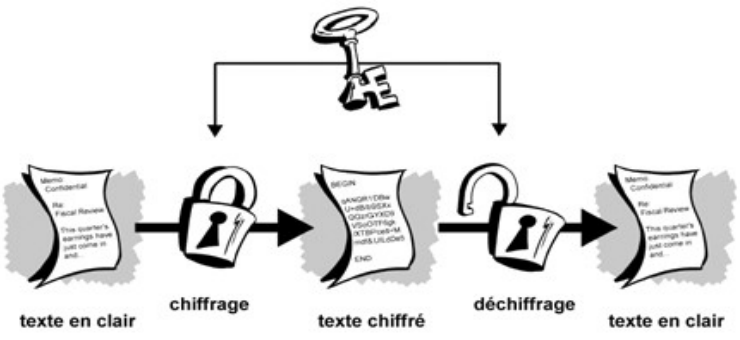

Figure 1.14 – Cryptographie

1.5.3 Mise à jour.

Les mises à jour jouent un rôle important dans la protection de vos appareils car elles permettent de réparer les erreurs ou combler des failles de sécurité. Elles vous permettent aussi de profiter des dernières nouveautés apportées au logiciel, de nouvelles fonctionnalités, d'un design, et d'une ergonomie améliorée.

1.5.4 VPN

Virtual Private Network, est une technique permettant à un ou plusieurs postes distants de communiquer de manière sure, tout en empruntant les infrastructures publiques. Ce type de liaison est apparu suite à un besoin croissant des entreprises de relier les différents sites, et ce de façon simple et économique. Jusqu'à l'avènement des VPN, les sociétés devaient utiliser des liaisons Transpac, ou des lignes louées. Les VPN ont permis de démocratiser ce type de liaison.[14]

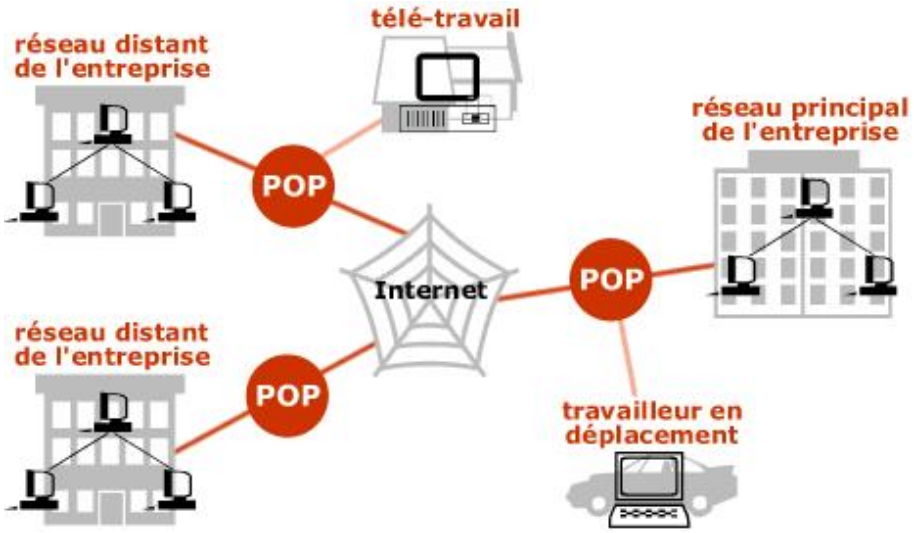

FIGURE 1.15 – Liaison VPN entre déférent siège d'une entreprise.[26]

1.5.5 Antivirus

L'antivirus est un programme qui a pour but principal de détecter, neutraliser ou éradiquer les logiciels malveillants des ordinateurs et autres appareils informatiques qui sont infectés. Il joue également un rôle préventif en empêchant les virus d'infecter les systèmes informatiques et de leur nuire.[22]

1.5.6 L'authentification et contrôle d'accès.

Il s'agit en premier lieu de la sécurité physique des équipements et installations. L'authentification est un mécanisme permettant de prouver l'identité d'un utilisateur (mots de passe, cartes à puce, méthodes biométriques, etc.) et de lui accorder uniquement les privilèges nécessaires pour l'accomplissement de ses tâches.

1.5.7 Système de prévention d'intrusion IPS

Un système de prévention des intrusions (IPS) est une forme de sécurité de réseau qui sert à détecter et prévenir les menaces identifiées. Les systèmes de prévention des intrusions

surveillent en permanence votre réseau, recherchant les éventuels actes de malveillance et capturent des informations à leur sujet. Il est capable d'arrêter et bloquer les attaques, c'est un Dispositif Actif, tout le trafic doit le traverser.

1.5.8 Systèmes de détection d'intrusions IDS

La détection d'intrusions consiste à analyser les informations collectées par les mécanismes d'audit de sécurité, à la recherche d'éventuelles attaques. Il permet ainsi d'avoir une connaissance sur les tentatives réussies comme échouées des intrusions. Les méthodes de détection d'intrusion diffèrent sur la manière d'analyser le journal d'audits.[24]

1.6 Conclusion

Dans ce chapitre, nous avons constaté que la sécurité réseau est un point primordial. Nous avons présenté un aperçu sur la sécurité informatique, et compris que la majorité des menaces et attaques qui se trouvent dans les réseaux locaux et nos appareils personnels, et faire prendre conscience des attaques que nous pouvons rencontrer. De nombreux mécanismes ont été conçus pour tester les vulnérabilités des systèmes, détecter, prévenir et lutter contre les attaques informatiques tels que les IDS qui constituent une bonne alternative pour mieux protéger le réseau informatique. Nous abordons ce système de manière plus détaillée et nous discuterons des déférentes approches des IDS et de leur fonctionnement dans le chapitre suivant.

Chapitre 2

Etude des systèmes de détection des intrusions : IDS

2.1 Introduction

Les systèmes d'information sont aujourd'hui de plus en plus ouverts sur Internet. Il en découle un nombre croissant d'attaques qui sont si rapides qu'avant, et tout le monde est exposé aux pertes des données essentielles. Malheureusement, les systèmes antivirus ou les pares-feux sont la plupart du temps inefficaces face à ces nouvelles menaces. Donc une politique de sécurité autour de ces systèmes est primordiale. C'est pour pallier ce manque que sont apparus récemment des nouveaux composants de sécurité appelés systèmes de détection et de prévention des intrusions.

En effet, après la courte introduction des systèmes de détection d'intrusions présentée au Chapitre I, dans ce chapitre, nous détaillons la notion de ce système ainsi que son architecture. Puis nous détaillons la taxonomie des IDS et présentons une analyse des différentes techniques possibles de la détection, ainsi les différentes efficacités des IDS.

2.2 Définition de l'IDS

Un système de détection d'intrusion (IDS) est un outil de protection censé détecter les attaques contre un réseau ou un hôte en analysant l'activité dans le réseau (Network IDS) ou dans l'hôte (Host IDS). Il est composé d'un agent de collecte de données d'audit sur le système en question. Ensuite, ces données sont soit stockées, soit traitées directement par le détecteur et remis au bureau de sécurité du site (SSO), suivi des étapes qui commencent généralement par une enquête plus approfondie sur les raisons de l'alarme. Lors de l'examen d'un IDS, il est important de différencier quatre fonctions qui existent dans un IDS typique qui sont : l'analyse, la journalisation, la gestion et l'action.

- **Analyse :** Analyse des journaux du système pour identifier des intentions dans la masse de données recueillie par l'IDS. Il y a deux méthodes d'analyses : L'une basée sur les signatures d'attaques, et l'autre sur la détection d'anomalies.
- **Journalisation :** Enregistrement des évènements dans un fichier de log. Exemples d'évènements : arriver d'un paquet, tentative de connexion.
- **Gestion :** Les IDS doivent être administrés de manière permanente. On peut assimiler un IDS à une caméra de sécurité.
- **Action :** Alerter l'administrateur quand une attaque dangereuse est détectée.

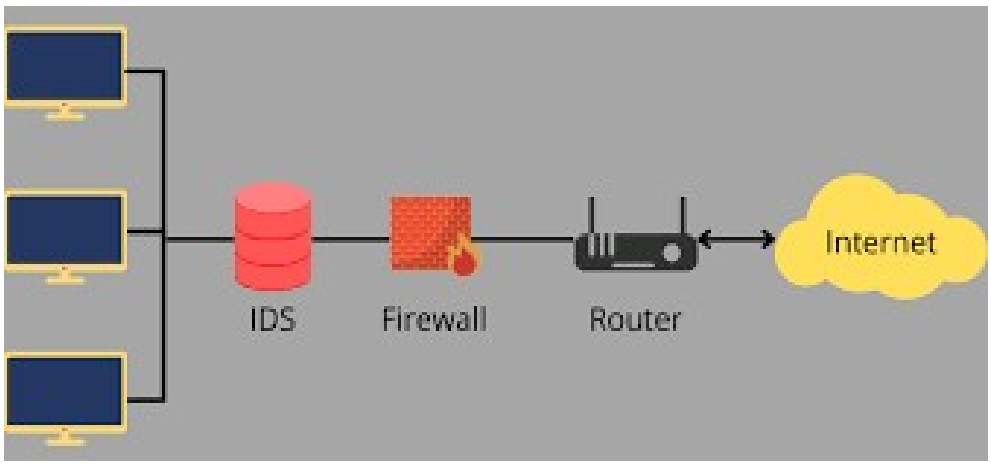

FIGURE 2.1 – IDS [25]

2.3 Caractéristiques d'un système de détection d'intrusion

Pour une détection d'intrusions efficace, il est très important de considérer certaines caractéristiques :

- Résister aux tentatives de corruption, c'est-à-dire, il doit pouvoir détecter s'il a subi lui-même une modification indésirable.
- Utiliser un minimum de ressources de système sous surveillance.
- S'adapter au cours du temps aux changements du système surveillé et du comportement des utilisateurs.
- Être facilement configurable pour implémenter une politique de sécurité spécifique d'un réseau.[19]

2.4 Différents Types IDS

Les IDS peuvent être classés en trois IDS : N-IDS, H-IDS et IDS hybrides

- **La détection d'intrusion basée sur l'hôte (H-IDS) :** IDS basé sur l'hôte (H-IDS) est connecté à un ordinateur et surveille les activités malveillantes se produisant au sein du système. Un H-IDS est placé sur l'hôte, surveille le trafic sur une seule machine. Il a cependant accès à des données fines de ce dernier tels les journaux ou appels système, voire à des données auxiliaires comme la consommation électrique ou la température.

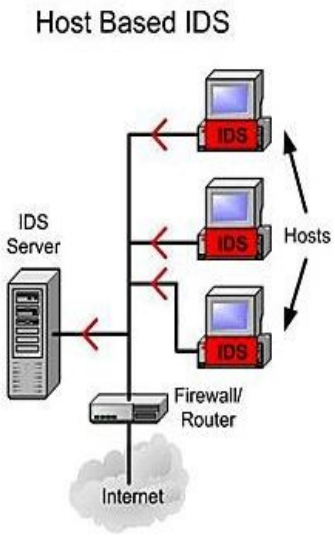

FIGURE 2.2 – Exemple H-IDS

— **La détection d'intrusion réseau (N-IDS) :** Les IDS réseaux (Network IDS), quant à eux, analysent en temps réel le trafic qu'ils aspirent à l'aide d'une sonde. Ils sont des IDS utilisés pour protéger un réseau. Ils disposent en général de capacités de traitement et de stockage importantes, laissant entrevoir une utilisation confortable des méthodes de détection abordées peu après. Ils ont accès à un autre type de données que les H-IDS mais parfois ces données sont chiffrées.

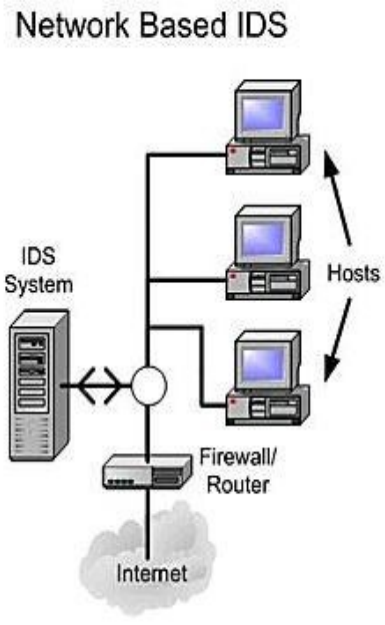

FIGURE 2.3 – Exemple N-IDS

— **Système de détection d'intrusion Hybride :** Il existe un autre type d'IDS qui permet de jumeler les deux types H-IDS et N-IDS pour produire un IDS complet. IDS hybrides rassemblent les caractéristiques des N-IDS et H-IDS. Ils permettent, de surveiller le réseau et les terminaux en même temps. On obtient par exemple : un " placement distribué " en implémentant un H-IDS léger dans chaque objet, un

" placement centralisé " avec un N-IDS exploitant les données d'une ou plusieurs sondes.

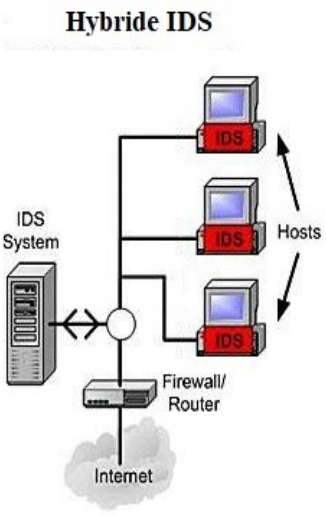

FIGURE 2.4 – Exemple IDS Hybride

2.5 Architecture des IDS

Plusieurs schémas ont été proposés pour décrire les composants d'un système de détection d'intrusions. Nous décrivons dans cette section les composants qui constituent classiquement un système de détection d'intrusion.

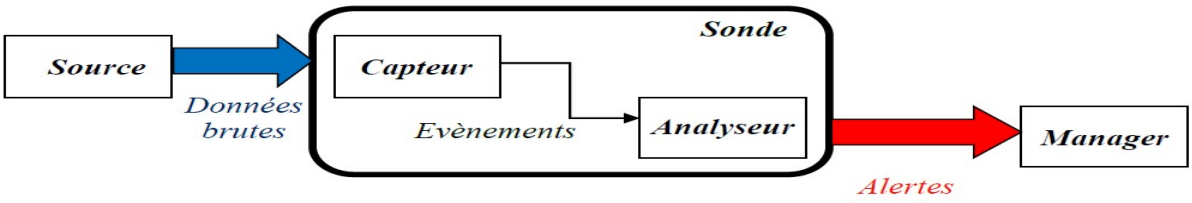

FIGURE 2.5 – Architecture classique d'un IDS[7]

— **Source de données :**dispositif générant de l'information sur les activités des entités du système d'information. Il existe différents types de données provenant de plusieurs sources (réseau, système, application et alertes).
— **Sonde :**un ou plusieurs capteurs couplés avec un analyseur.
— **Capteur :** est un logiciel qui observe l'activité du système et génère des événements en filtrant et formatant les données brutes provenant d'une source de données.
— **Analyseur :**c'est un outil logiciel qui met en œuvre l'approche choisie pour la détection (comportementale ou par scénarios). Son objectif est de déterminer si le flux d'événements fourni par le capteur contient des éléments caractéristiques d'une activité malveillante, et il génère des alertes lorsqu'il détecte une intrusion.
— **Alerte :**message formaté émis par un analyseur s'il trouve des activités intrusives dans une source de données.

— **Manager :** composant d'un IDS qui collecte les alertes produites par le capteur, les met en forme et les présente à l'opérateur. Éventuellement, le manager est chargé de la réaction à adopter qui peut être :

1. Confinement de l'attaque, qui a pour but de limiter les effets de l'attaque.
2. Eradication de l'attaque, qui tente d'arrêter l'attaque.
3. Recouvrement, qui est l'étape de restauration du système dans un état sain.
4. Diagnostic, qui est la phase d'identification du problème.

Un IDS peut analyser les couches suivantes :
— Couche Réseau (IP, ICMP)
— Couche Transport (TCP, UDP)
— Couche Application (HTTP, Telnet)

Selon le type de trafic, l'IDS accomplit certaines actions définies dans les règles. Certains termes sont souvent employés quand on parle d'IDS :
— **Faux positif :** une alerte provenant d'un IDS mais qui ne correspond pas à une attaque réelle (Fausse Alerte).
— **Faux négatif :** une intrusion réelle qui n'a pas été détectée par l'IDS

2.6 Vocabulaire (notion) utilisé dans la détection d'intrusions

La détection d'intrusions utilise un vocabulaire et des notions bien défini qui est comme suit :
— **Attaque ou intrusion :** action qui permet d'enfoncer ou violer la politique de sécurité.
— **Détection d'intrusions :** recherche de traces laissées par une intrusion dans les données produites par une source.
— Vulnérabilité : faille de sécurité due à une fausse implémentation ou de configuration d'un système logiciel ou matériel.
— **Log (trace d'audit) :** c'est un fichier système à analyser.
— **Scénario :** suite constituée des étapes élémentaires d'une attaque.
— **Classification des attaques :** un IDS a la capacité de classifier des attaques.
— **Système de détection d'intrusions :** ensemble constitué d'un ou plusieurs capteurs, un ou plusieurs analyseurs et un ou plusieurs managers.
— **Corrélation :** c'est l'interprétation conceptuelle de plusieurs événements (alertes) visant à leur assigner une meilleure sémantique et à réduire la quantité globale d'événements (d'alertes).
— **La signature :** C'est un ensemble de règles logiques qui décrit un évènement, un paquet. Chaque attaque, virus, vers, etc. Possède des caractéristiques uniques et différentes des autres. Un IDS cherche ces caractéristiques dans le trafic supervisé.
— **Administrateur :** Personne chargée de mettre en place la politique de sécurité, et par conséquent, de déployer et configurer les IDS.[15]

2.7 Emplacement d'un système de détection d'intrusions

Le placement des IDS va dépendre de la politique de sécurité menée. Il existe plusieurs endroits stratégiques où il convient de placer un IDS tels que :

- Dans la zone démilitarisée (attaques contre les systèmes publics),
- Dans le (ou les) réseau privé (intrusions vers ou depuis le réseau interne),
- Sur la patte extérieure du firewall (détection de signes d'attaques parmi tout le trafic entrant et sortant, avant que n'importe quelle protection intervienne).

Il est important de bien définir les zones sensibles du système (réseau), ainsi que les zones les plus attractives pour un pirate.
Le schéma suivant illustre les trois positions que peut y prendre un IDS dans un réseau local :

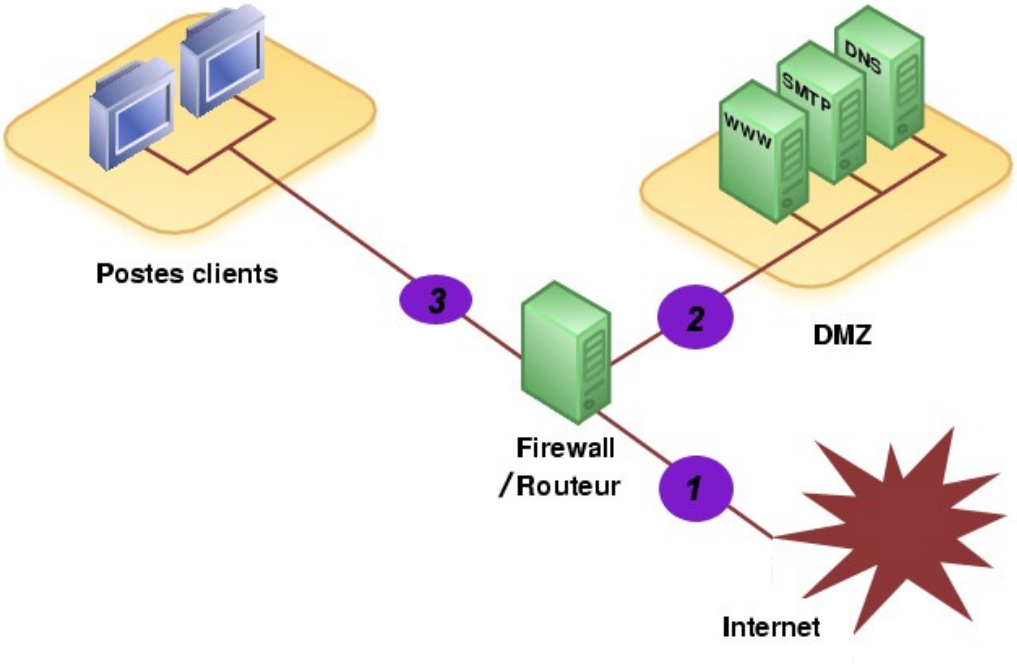

FIGURE 2.6 – Placement d'un IDS

- **Position (1) :** Sur cette position, l'IDS va pouvoir détecter l'ensemble des attaques frontales, provenant de l'extérieur, en amont du firewall. Ainsi, beaucoup d'alertes seront remontées ce qui rendra les logs difficilement consultables.
- **Position (2) :** Si l'IDS est placé sur la DMZ, il détectera les attaques qui n'ont pas été filtrées par le firewall et qui relèvent d'un certain niveau de compétence. Les logs seront ici plus clairs à consulter puisque les attaques bénignes ne seront pas recensées.
- **Position (3) :** L'IDS peut ici rendre compte des attaques internes, provenant du réseau local de l'entreprise. Il peut être judicieux d'en placer un à cet endroit étant donné le fait que 80% des attaques proviennent de l'intérieur. De plus, si des trojans ont contaminé le parc informatique (navigation peu méfiante sur internet) ils pourront être ici facilement identifiés pour être ensuite éradiqués.[9]

2.8 classification des IDS

Les IDS peuvent être classés selon différents critères qui ne sont pas mutuellement exclusif, et ils sont :

2.8.1 Sources et la nature des données à analyser

Parmi les caractéristiques essentielles des systèmes de détection d'intrusions : les sources de données à analyser qui constituent la matière première du processus de détection. Ces données proviennent soit de logs (journaux) générés par le système d'exploitation ou de logs des applications, des informations provenant du réseau, soit encore d'alertes générées par d'autres IDS.

2.8.2 Méthode de détection

Il existe plusieurs méthodes permettant de détecter une intrusion :

2.8.2.1 Approche par scénario ou par signature

Le procédé construit d'abord une collection de signatures basé sur des informations telles que la connaissance du domaine et l'expérience d'experts. Il tente alors de rechercher un modèle particulier dans les données entrantes du réseau qui correspond étroitement à une ou plusieurs signatures dans la base de données. La méthode fondée sur l'abus peut détecter efficacement les intrusions correspondant à au moins un des signatures dans la base de données et dispose ainsi d'un faible taux de faux positifs. Cependant, il ne peut pas détecter intrusions inconnues qui ne correspondent à aucun modèle dans la base de données, et par conséquent peut donner lieu à un taux élevé de faux négatifs, surtout lorsque l'attaquant a connaissance des signatures dans la base de données. Comme un remède, l'IDS basé sur l'abus doit souvent mettre à jour fréquemment les signatures et les règles dans la base de données.

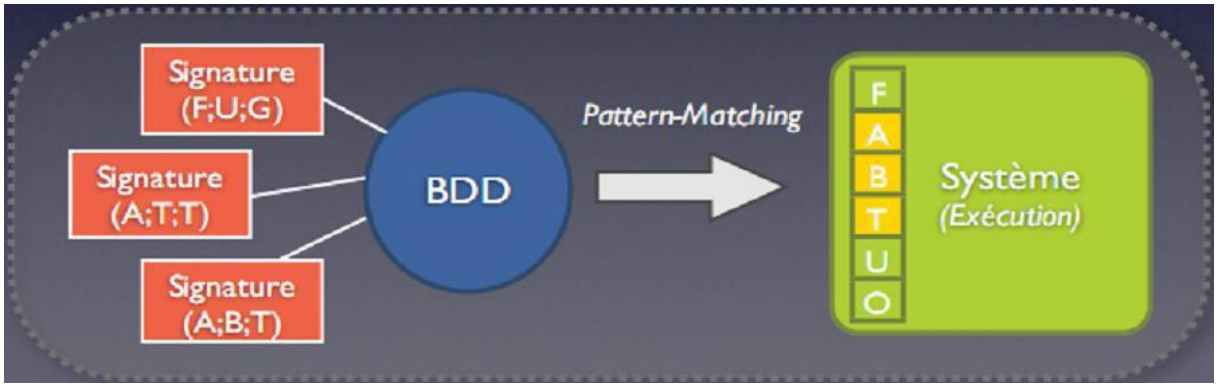

FIGURE 2.7 – Approche par scénario[8]

2.8.2.2 L'approche comportementale (Anomaly Detection)

Cette approche analyse les événements pour détecter une intrusion et est donc connue sous le nom de détection basée sur les événements. Dans les environnements IoT, les IDS basés sur les anomalies identifie les activités malveillantes en analysant l'événement. Premièrement, après une période de surveillance, un comportement régulier ou normal du

réseau est défini. Ensuite si un événement se produit qui n'est pas conforme au comportement habituel, il est qualifié comme une intrusion. Il s'agit d'une approche plus efficace que l'IDS basé sur la signature.
Le principal inconvénient de cette méthode est qu'elle peut générer un énorme volume de fausses alarmes parce que tout un comportement inédit peut être traité comme une anomalie.[32]

2.8.3 Le lieu de l'analyse des données

Les systèmes de détection d'intrusions peuvent être classés aussi en se basant sur la localisation réelle de l'analyse des données :

- **Analyse centralisée :** Certains IDS ont une architecture multi-capteurs (ou multisondes). Ils centralisent les événements (ou alertes) pour analyse au sein d'une seule machine. L'intérêt principal de cette architecture est de faciliter la corrélation entre événements puisque nous disposons alors d'une vision globale. Par contre, la charge des calculs (effectués sur le système central) ainsi que la charge réseau (due à la collecte des événements ou des alertes) peuvent être lourdes et risquent de surcharger le trafic réseau.
- **Analyse locale (Distribué) :** Si l'analyse des événements est effectuée au plus près de la source de données généralement en local sur chaque machine disposant d'un capteur, nous minimisons le trafic réseau et chaque analyseur séparé dispose de la même puissance de calcul. En contrepartie, il est impossible de croiser des événements qui sont traités séparément et nous risquons de passer à côté de certaines attaques distribuées.[21]

2.8.4 Fréquence de l'analyse

La fréquence d'utilisation d'un système de détection d'intrusion peut exister selon deux formes :

- **Surveillance périodique (IDS offline) :**ce type de système de détection d'intrusion analyse périodiquement les différentes sources de données à la recherche d'une éventuelle intrusion ou une anomalie passée. L'avantage de ce type est qu'il ne consomme pas beaucoup de ressources système. L'inconvénient majeur de ce type est sa détection tardive des attaques ce qui risque de provoquer des dégâts dangereux.
- **Surveillance en temps réel (IDS online) :** la détection se fait sur le traitement et l'analyse continue des informations produites par les différentes sources de données pour augmenter le niveau de sécurité dans des contextes sensibles. Ce type d'IDS consomme un taux élevé de ressources systèmes car il faut analyser à la volée tout ce qui se passe sur le système et ce qu'il le rend non préférable en cas de ressources précieuses telle que les serveurs de messagerie par exemple.

2.8.5 Comportement en cas de détection d'une attaque

On peut classer les IDS par type de réaction lorsqu'une attaque est détectée. Il existe deux types de réponses, suivant les IDS utilisés. La réponse passive est disponible pour tous les IDS, la réponse active est plus ou moins implémentée.

- **Réponse passive :**la plupart des systèmes de détection d'intrusions n'apportent qu'une réponse passive à l'intrusion. C'est à dire lorsqu'une attaque est détectée, le système d'intrusion ne prend aucune action, il génère seulement une alarme ou

bien d'un message dans une console en direction de l'administrateur qui contient les informations à propos de chaque attaque. C'est alors lui qui devra prendre les mesures qui s'imposent

— **Réponse active :**La réponse active consiste à répondre directement à une attaque. Donc ces systèmes de détection d'intrusions peuvent, en plus de la notification à l'opérateur, prendre automatiquement des mesures pour stopper l'attaque en cours. Telle que le blocage de son adresse IP.[20]

2.9 Efficacité des IDS

L'efficacité d'un système de détection d'intrusions est déterminée par les mesures suivantes :

— **Exactitude :** Le système de détection d'intrusions n'est pas exact s'il considère les actions légitimes des utilisateurs comme atypiques ou intrusives (faux positif).
— **Performance :** Effectuer une détection en temps réel.
— **Tolérance aux pannes :** Un système de détection d'intrusions doit être résistant aux attaques.
— **Rapidité :** Un système de détection d'intrusions doit exécuter et propager son analyse d'une manière prompte pour permettre une réaction rapide dans le cas d'existence d'une attaque pour permettre à l'agent de sécurité de réagir.
— **Complétude :** La complétude est la capacité d'un système de détection d'intrusion de détecter toutes les attaques.[21]

2.10 Limites et vulnérabilités des IDS

Les IDS comme tout autre système informatique à ses limites malgré ses avantages quel que soit l'architecture N-IDS ou H-IDS :

- Nombreux faux positifs et même des faux négatifs.
- Configuration complexe et longue : la configuration, et l'administration des IDS nécessitent beaucoup de temps, et de connaissances.
- Les attaques applicatives sont difficilement détectables.

  — Injection SQL.
- Des évènements difficilement détectables.
- Pollution des IDS.
  — **Consommation des ressources :** la détection d'intrusion est excessivement gourmande en ressources. En effet un système N-IDS doit générer des journaux de comptes-rendus d'activité anormale ou douteuse sur le réseau.

  — **Perte de paquets :** les vitesses de transmission sont parfois telles qu'elles dépassent largement la vitesse d'écriture des disques durs, ou même la Vitesse de traitement des processeurs.

  — **Vulnérabilité aux dénis de service :** un attaquant peut essayer de provoquer un déni de service au niveau du système de détection d'intrusion, ou pire au niveau du système d'exploitation de la machine supportant l'IDS, une fois l'IDS désactivé (" hors service "), l'attaquant peut tenter tout ce qui lui

convient.

— Une attaque réelle peut passer inaperçue.

■ Ils ne peuvent pas compenser les trous de sécurité dans les protocoles réseaux.

En effet, il existe quelques techniques qui peuvent permettre de détecter un IDS :

— **Usurpation d'adresse MAC :** les N-IDS mettent l'interface de capture en mode promiscuité (promiscuous mode), il est donc possible de détecter l'IDS en envoyant par exemple un ICMP " echo request " à la machine soupçonnée d'être un N-IDS avec une adresses MAC inexistante. Si la machine répond alors elle est en mode promiscuous et peut donc être un N-IDS.

— **Mesure des temps de latence :** puisque l'interface est en mode promiscuous, les temps de réponse sont plus longs. Voici une méthode pour exploiter ces temps de latence :

1. Le pirate génère une série de pings vers l'adresse à tester, puis il mesure et note les temps de réponse.
2. Le pirate sature ensuite le réseau en broadcast dans le but de ralentir l'IDS, qui recevra tous les paquets.

Enfin, le pirate réémet la même série de pings en mesurant les nouveaux temps de réponse. S'ils sont bien plus élevés que les premiers temps obtenus, il est fort possible que la machine soit en mode promiscuous.

— **Observation des requêtes DNS :** Les IDS génèrent souvent des requêtes DNS lors des alertes. En observant le DNS primaire lors de fausses attaques, on peut détecter qu'il y a un IDS.

2.11 Les systèmes de prévention d'intrusion IPS

L'IPS est un système de Prévention/Protection contre les intrusions et non plus seulement de reconnaissance et de signalisation des intrusions comme la plupart des IDS. Un système de prévention d'intrusion est une forme de sécurité de réseau qui sert à détecter et prévenir les menaces identifiées. Ces systèmes de prévention surveillent en permanence le réseau, recherchant les éventuels actes de malveillance et capturent des informations à leur sujet. Il signale ces événements aux administrateurs du système et prend des mesures préventives, telles que la fermeture des points d'accès et la reconfiguration des firewalls pour empêcher de futures attaques. Les solutions IPS peuvent également être utilisées pour identifier les problèmes liés aux politiques de sécurité de l'entreprise

2.12 La similitude et différence entre IDS et IPS

2.12.1 Similitudes entre IDS et IPS :

Les premiers processus pour IDS et IPS sont similaires. Les deux détectent et surveillent le système ou le réseau à la recherche d'activités malveillantes. Voyons leurs points communs :

— **Surveiller :** une fois installées, les IDS et IPS surveillent un réseau ou un système en fonction des paramètres spécifiés. Vous pouvez définir ces paramètres en fonction de vos besoins de sécurité et de votre infrastructure réseau et les laisser inspecter tout le trafic entrant et sortant de votre réseau.

— **Détection des menaces :** les deux lisent tous les paquets de données circulant dans votre réseau et comparent ces paquets à une bibliothèque contenant des menaces connues. Lorsqu'ils trouvent une correspondance, ils signalent ce paquet de données comme malveillant.
— **Apprendre :** ces deux technologies utilisent des technologies modernes telles que l'apprentissage automatique pour s'entraîner pendant une période et comprendre les menaces et les modèles d'attaque émergents. De cette façon, ils peuvent mieux répondre aux menaces modernes.
— **Journal :** lorsqu'ils détectent une activité suspecte, ils l'enregistrent avec la réponse. Il vous aide à comprendre votre mécanisme de protection, à détecter les vulnérabilités de votre système et à former vos systèmes de sécurité en conséquence.
— **Alerte :** dès qu'ils détectent une menace, l'IDS et l'IPS envoient des alertes au personnel de sécurité. Cela les aide à se préparer à toutes les circonstances et à prendre des mesures rapides.

2.12.2 Différence entre IDS et IPS :

La différence principale entre IDS et IPS est que IDS fonctionne comme un système de surveillance et de détection tandis qu'IPS fonctionne comme un système de prévention en dehors de la surveillance et de la détection. Certaines différences sont :

— **Réponse :** Les solutions IDS sont des systèmes de sécurité passifs qui surveillent et détectent uniquement les réseaux pour les activités malveillantes. Ils peuvent vous alerter mais ne prennent aucune mesure par eux-mêmes pour empêcher l'attaque. L'administrateur du réseau ou le personnel de sécurité affecté doit prendre des mesures immédiatement pour atténuer l'attaque. D'autre part, les solutions IPS sont des systèmes de sécurité actifs qui surveillent et détectent sur votre réseau pour les activités malveillantes, alertent et empêchent automatiquement l'attaque de se produire.
— **Placement :** l'IDS est placé à la périphérie d'un réseau pour collecter tous les événements, enregistrer et détecter les violations. Ce positionnement donne à l'IDS une visibilité maximale pour les paquets de données. Le logiciel IPS est placé derrière le pare-feu du réseau et communique en ligne avec le trafic entrant pour mieux prévenir les intrusions.
— **Protection :**Si vous êtes menacé, l'IDS pourrait être moins utile car votre personnel de sécurité doit trouver comment sécuriser votre réseau et nettoyer le système ou le réseau immédiatement. IPS peut effectuer une prévention automatique par lui-même.
— **Faux positifs :** Si IDS donne un faux positif, vous pouvez trouver une certaine commodité. Mais si IPS le fait, l'ensemble du réseau en souffrira car vous devrez bloquer tout le trafic - entrant et sortant du réseau.
— **Les performances du réseau :** Comme IDS n'est pas déployé en ligne, il ne réduit pas les performances du réseau. Cependant, les performances du réseau peuvent être réduites en raison du traitement IPS, qui est en phase avec le trafic.

2.13 Conclusion

Le domaine des systèmes de détection d'intrusions est un sujet très vaste. Donc ce chapitre, nous a permis de constater que les IDS sont de plus en plus fiables, d'où le fait qu'ils soient souvent intégrés dans les solutions modernes de sécurité. Les avantages qu'ils présentent par rapport aux autres outils de sécurité les favorisent, mais d'un autre

côté cela n'empêche pas que les meilleurs IDS présentent aussi des lacunes et quelques inconvénients. Nous comprenons donc que les IDS sont des outils indispensables à la bonne sécurité d'un réseau, et capables de satisfaire les besoins des utilisateurs.

Chapitre 3

Système de détection d'intrusion basée sur LSTM

3.1 Introduction

Un système de détection d'intrusion (ou IDS : Intrusion detection System) est un mécanisme qui vise à détecter des attaques contre les systèmes informatiques et les réseaux ou généralement contre des systèmes d'information. En effet, c'est difficile de fournir des systèmes d'information prouvables sécurisés et pour les maintenir dans un état sécurisé pendant leur vie et utilisation.

Au cours de ces dernières années, divers systèmes basés sur l'apprentissage en profondeur " Deep Learning " ont été mis au point pour améliorer la classification et la prédiction des attaques et logiciels malveillants, et ils dépendent sur des données étiquetées. Dans ce chapitre, nous allons présenter notre approche nommée, qui consiste à prédire les attaques en utilisant LSTM et l'apprentissage par renforcement.

Dans ce qui suit, nous allons tout d'abord donner une présentation générale sur l'apprentissage automatique et sur les réseaux RNN. Ensuite, l'architecture proposée et nous allons expliquer son fonctionnement. Finalement, nous concluons ce chapitre.

3.2 Apprentissage automatique

3.2.1 Notions de base sur l'apprentissage automatique (Machine Learning)

3.2.1.1 Apprentissage automatique

L'apprentissage automatique (ML) est un sous-ensemble de l'intelligence artificielle. Il se concentre sur l'étude d'algorithmes informatiques qui s'améliorent automatiquement par leur propre expérience. Ces algorithmes sont utilisés dans une grande variété d'applications différentes, de la vision par ordinateur aux optimisations mathématiques. Les algorithmes ML sont nécessaires lorsqu'il est difficile ou impossible de développer des algorithmes conventionnels pour effectuer une tâche nécessaire.

3.2.1.2 Types d'apprentissage (Types of Learning)

— **Apprentissage supervisé :** l'apprentissage supervisé consiste à entrainer un modèle à partir de données préalablement étiquetées ou annotées. Il est utilisé aussi bien en traitement du langage qu'en vision par ordinateur ou analyse prédictive.

— **Apprentissage semi-supervisé :** c'est une approche de l'apprentissage automatique qui combine une petite quantité de données étiquetées avec une grande quantité de données non étiquetées pendant la formation. L'apprentissage semi-supervisé se situe entre l'apprentissage non supervisé (sans données de formation étiquetées) et l'apprentissage supervisé (avec uniquement des données de formation étiquetées).
— **Apprentissage non Supervisé :** Découvrez des modèles dans des données sans étiquette. Exemple regrouper des documents similaires basés sur du texte.
— **Apprentissage par renforcement :** En termes simples, nous pouvons dire que l'apprentissage par renforcement est l'une des techniques de l'apprentissage machine. Vous pouvez former un agent d'intelligence artificielle en lui permettant d'effectuer des actions répétitives et en le récompensant. L'agent dans l'expérience de l'apprentissage par renforcement prendra diverses actions. Si les actions sont correctes, l'agent sera récompensé. En revanche, s'il fait de mauvaises actions, il recevra une punition. Cela augmentera la capacité d'apprentissage de l'agent pour la réalisation des actions.

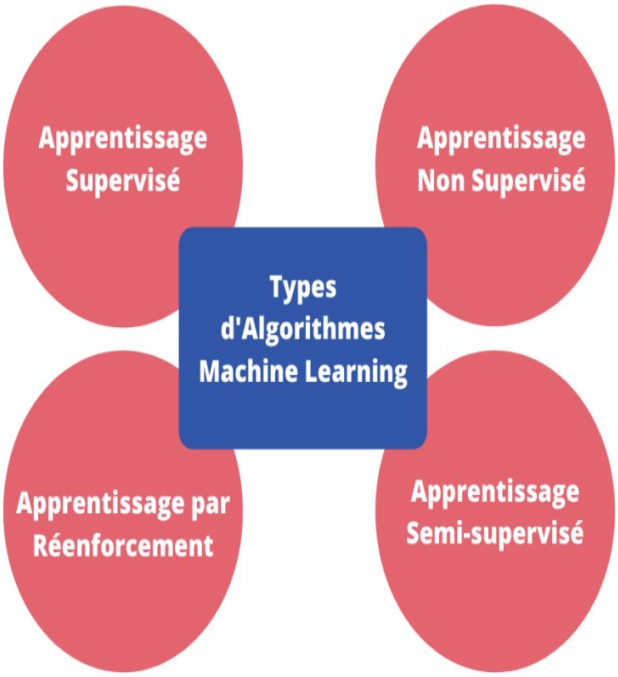

FIGURE 3.1 – Types d'apprentissage automatique

3.2.1.3 Réseaux de neurones et ces types

Un réseau de neurones inspiré du cerveau humain, est un ensemble de neurones formels interconnectés permettant la résolution de problèmes complexes tels que la reconnaissance des formes ou le traitement du langage naturel, grâce à l'ajustement des coefficients de pondération.

Les premiers travaux datent de 1943 et sont l'œuvre de MM. Mac Culloch et Pitts. Ils présentent un modèle assez simple pour les neurones et explorent les possibilités de ce modèle.
L'idée principale des réseaux de neurones "modernes" est la suivante :
On se donne une unité simple, un neurone, qui est capable de réaliser quelques calculs

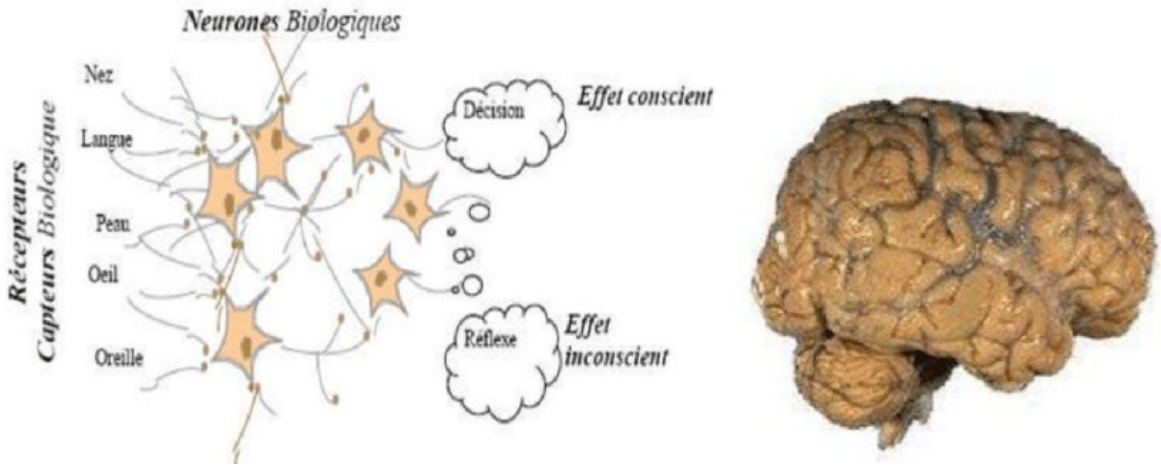

FIGURE 3.2 – Neurone biologique

élémentaires. On relie ensuite entre elles un nombre important de ces unités et on essaye de déterminer la puissance de calcul du réseau ainsi obtenu. Il est important de noter que ces neurones manipulent des données numériques et non pas symboliques.
Il est doté de surcroît de capacités de généralisation et de classification qui lui permettent notamment de réaliser des opérations statistiques très élaborées.[6]

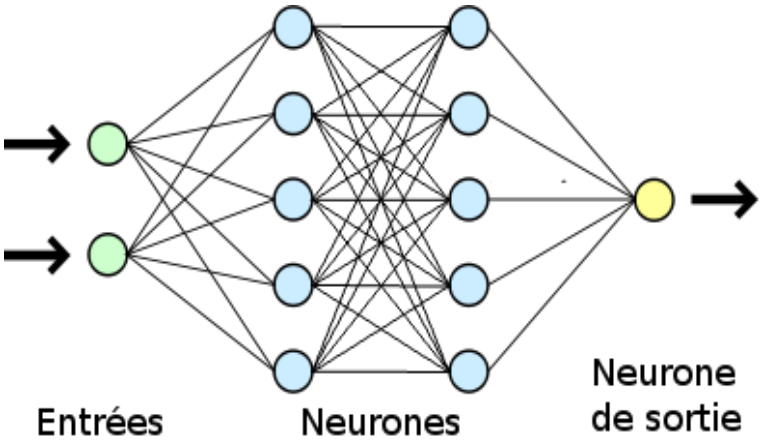

FIGURE 3.3 – Un réseau de neurone [6]

Il est composé d'une couche d'entrées qui fais entrer les données au réseau, une couche de neurone caché qui font des calculs et une couche de sortie qui offre le résultat de calcul. Parmi les types de reseau de neurone on trouve :

— **Les réseaux neurones artificiel (ANN) :** Est un groupe de plusieurs perceptrons ou neurones à chaque couche. ANN est également connue sous le nom de réseau de neurones Feed-Forward, car les entrées ne sont traitées que dans le sens direct. Ce type de réseaux de neurones est l'une des variantes les plus simples des réseaux de neurones. Ils transmettent les informations dans une direction, à travers divers nœuds d'entrée, jusqu'à ce qu'elles parviennent au nœud de sortie. Le réseau peut avoir ou non des couches de nœuds cachés, ce qui rend leur fonctionnement plus interprétable.

— **Réseau de neurones convolutifs (CNN) :** L'un des modèles les plus populaires utilisés aujourd'hui. Ce modèle de calcul de réseau de neurones utilise une variation de perceptrons multicouches et contient une ou plusieurs couches convolutives qui peuvent être soit entièrement connectées, soit regroupées. Ces couches convolutives

créent des cartes de caractéristiques qui enregistrent une région d'image qui est finalement divisée en rectangles et envoyée pour un traitement non linéaire. Ils ont une Très haute précision dans les problèmes de reconnaissance d'images.
— **Réseau de neurones récurrents (RNN) :** Ce type de réseaux sont plus complexes on va le détailler plus dans la section suivante.

3.2.1.4 Apprentissage en profondeur :

Le **Deep Learning**, ou apprentissage profond, est un sous-ensemble du Machine Learning, ou apprentissage automatique, basé sur des réseaux neuronaux artificiels. Le processus d'apprentissage est qualifié de profond parce que la structure des réseaux neuronaux artificiels se compose de plusieurs couches d'entrée, de sortie et masquées. Chaque couche contient des unités qui transforment les données d'entrée en informations que la couche suivante peut utiliser une tâche prédictive spécifique. Grâce à cette structure, une machine est capable d'apprendre au travers de son propre traitement de données.

3.2.2 Les réseaux de neurones récurrents(RNN)

Récemment, la disponibilité des données et l'amélioration considérable des puissances de calcul des GPU ont rendu l'apprentissage en profondeur(Deep Learning) populaire par rapport aux autres techniques d'apprentissage automatiques. Un modèle de Deep Learning est simplement un modèle empilant un large nombre de couches de différents types de réseaux de neurones. Nous allons étudier un type particulier de réseaux de neurones qui sont les réseaux de neurones récurrent ou RNN (Récurrent Neural networks).

3.2.2.1 Qu'est-ce qu'un RNN

Un réseau de neurones récurrent (RNN, récurrent neural network) est un type de réseau de neurones artificiels principalement utilisé dans la reconnaissance automatique de la parole, dans l'écriture manuscrite et dans le traitement automatique du langage naturel, en particulier dans la traduction automatique. Les RNN sont conçus de manière à reconnaître les caractéristiques séquentielles et pour prédire le scénario suivant le plus probable.[12]
Quand on parle de RNN cela peut avoir deux significations :
— Une couche RNN
— Un modèle de Deep Learning composé de couches RNN.
On va donc se concentrer sur les couches RNN classique. Les couches RNN sont récurrentes, c'est à dire que les informations qu'elles calculent vont être stockés en mémoire pour être réutiliser sur un prochain calcul.

Dans une couche classique non récurrente un neurone effectue un produit vectoriel entre l'entrée qu'il reçoit et son poids puis ajoute un biais. Ce calcul va ensuite passer dans sa fonction d'activation. En bref :

fonction-d-activation(prod-vecteur(donnée, poids) + biais)

Dans une couche RNN un deuxième produit vectorielle contenant les données en mémoire vient s'ajouter au calcul :

fonction-d-activation(prod-vecteur(donnée,poids1) + prod-vecteur(donnée en mémoire, poids2) + biais)

Ainsi, lorsqu'on veut prédire la valeur d'une action cotée en bourse à un temps t+1, le modèle RNN prend en compte les précédentes valeurs dans sa prédiction.

En quelques sorte, le modèle comprend la variation du prix qui a eu lieu entre t-2 et t-1, entre t-1 et t et peut prédire la future variation entre t et t+1 !

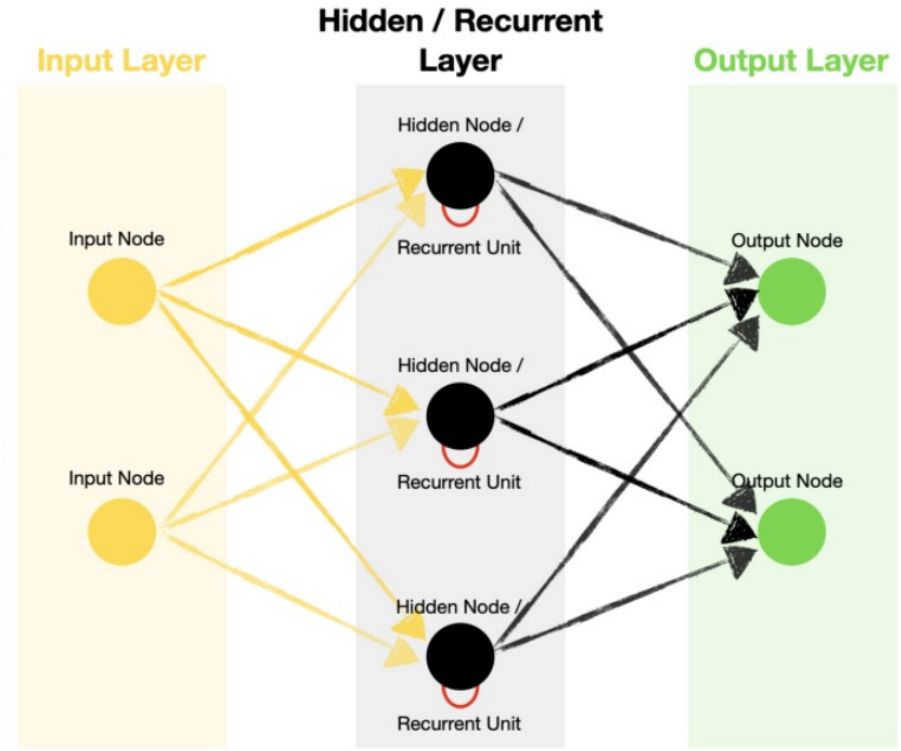

FIGURE 3.4 – Architecture de réseau neuronal récurrent standard

3.2.2.2 Architecture d'un RNN traditionnel

Les RNN utilisent les sorties précédentes comme entrées supplémentaires et sont parfaitement adaptés au traitement de données séquentielles. Un RNN habituel a une mémoire à court terme. Ils sont généralement sous la forme suivante :[10]

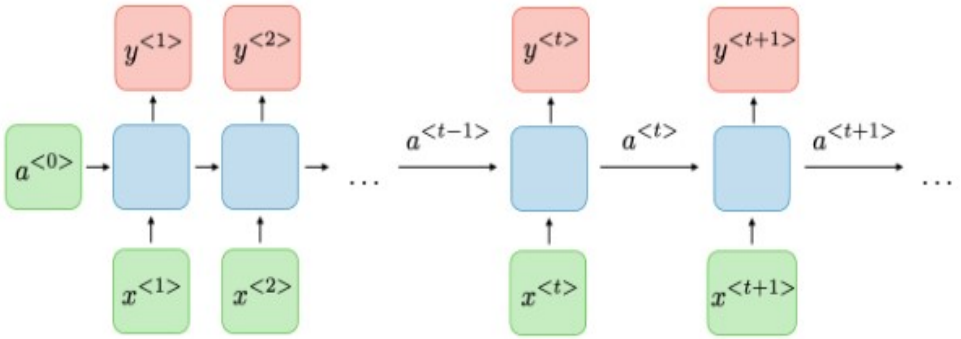

FIGURE 3.5 – Architecture RNN traditionnelle[10]

À chaque instant t, le passage vers l'avant (Forward pass) est modélisé par les équations suivantes :

Où xt et at sont respectivement le vecteur d'entrée du réseau et le vecteur d'activation à l'instant t, g1, g2 sont des fonctions d'activation, les W et b sont respectivement les

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a) \quad (1)$$

$$y^{<t>} = g_2(W_{ya}a^{<t>} + +b_y) \quad (2)$$

poids et les biais à apprendre durant l'entrainement du réseau.

La valeur de sortie à l'instant t y¡t¿ est calculée par l'équation (2) en fonction de la valeur d'activation a¡t¿ calculée par l'équation (1).

Nous constatons clairement l'aspect récurrent dans ces calculs (le calcul à l'instant t est à base de l'information apportée de l'instant t-1, elle-même calculée à partir de l'information apportée de t-2 etc.).

3.2.2.3 Type de RNN

Les réseaux neuronaux Feed-Forward fassent correspondre une entrée à une sortie, les RNN peuvent correspondre à :

- **Un à plusieurs :** Le RNN reçoit une unique entrée et retourne plusieurs sorties. Par exemple : Génération musicale
- **Plusieurs à plusieurs :** on peut prendre plusieurs entrées et obtenir plusieurs sorties. Nous n'avons pas forcément le même nombre de neurones d'entrée et de sortie. Nous pouvons citer ici, la traduction de texte ou reconnaissance de l'entité de nom.
- **Plusieurs à un :** On a plusieurs entrées et il y a une unique sortie. Par exemple : Classification des sentiments
- **Un à un :** Une seule entrée est mappée à une seule sortie. Par exemple : prédiction de mots.

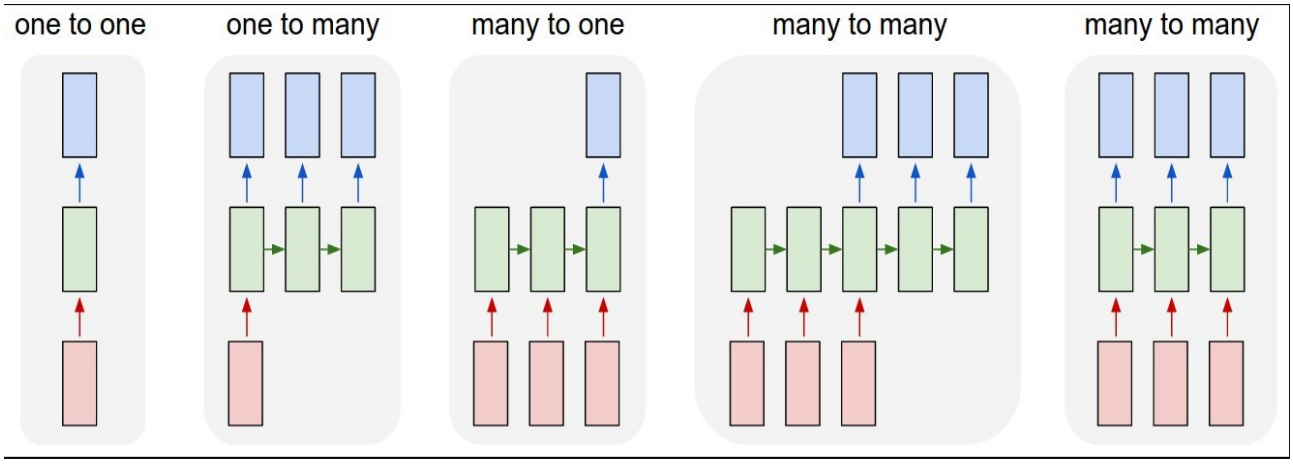

FIGURE 3.6 – Types de réseaux de neurones récurrents[11]

3.2.2.4 Comment fonctionne un RNN

Nous prenons un exemple pour comprendre cette approche :

Il existe une couche cachée dans les réseaux de neurones traditionnels, avec sa propre collection de poids et de biais. Pour le poids et le biais 1 respectivement, disons que ce poids et ce biais sont w1 et b1. De même, pour la troisième couche, nous aurons w2, b2

et w3, b3. Ces couches sont également séparées les unes des autres, ce qui signifie que la sortie précédente n'est pas mémorisée. Supposons cependant qu'un réseau plus profond existe avec une couche d'entrée, trois couches cachées et une couche de sortie.

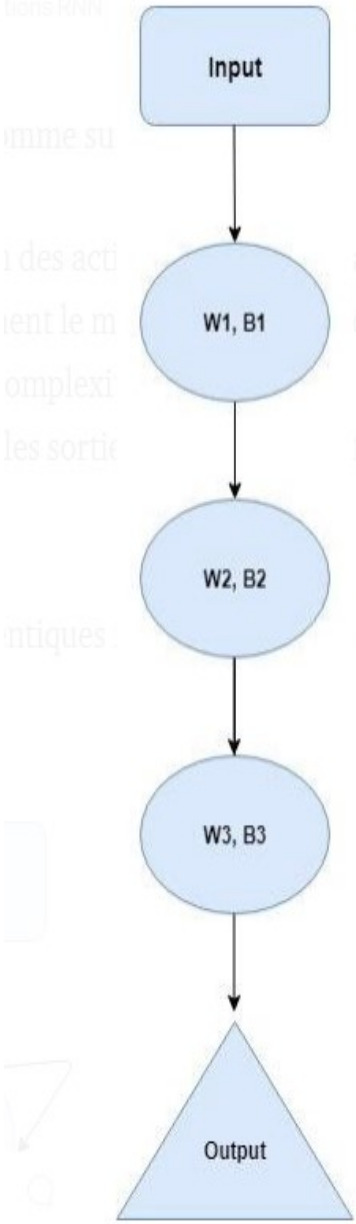

Figure 3.7 – Pondérations RNN

Le réseau neuronal récurrent fonctionne comme suit :

1. Le premier RNN effectuera la conversion des activations indépendantes en activations dépendantes. Il attribue également le même poids et le même biais à toutes les couches, ce qui réduit encore la complexité des paramètres RNN et fournit un cadre cohérent pour mémoriser les sorties précédentes en fournissant une entrée à la couche suivante.
2. Ces trois couches de poids et de biais identiques fusionnent en une seule structure récurrente.

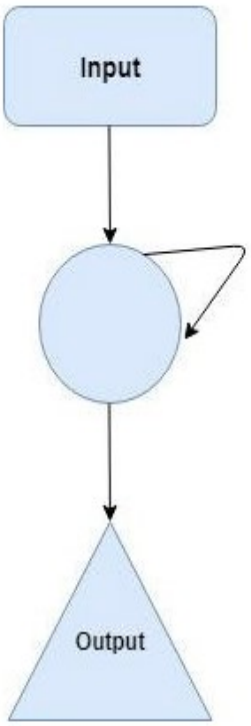

FIGURE 3.8 – Etat RNN

Formule de calcul de l'état actuel : $h_t = f(h(t-1), x_t)$
Où :

- h_t **:** c'est le nouvel état
- $h(t-1)$ **:** est l'état précédent
- x_t **:** est l'entrée actuelle.

Au moment de l'exécution du réseau, une formule de récurrence utilise la même fonction et les mêmes poids et le même biais sur tout le réseau dans chaque horodatage pour avoir une séquence appropriée avec l'entrée.
Voyons maintenant comment les couches cachées sont calculées dans le réseau de neurones récurrent.
Supposons que nous prenions
Une fonction d'activation comme tanh pour notre réseau.

- Le poids pour trois neurones récurrents = Whh.
- Et le poids au neurone d'entrée est égal à Wxh.

Donc, à partir de là, nous pouvons conclure la formule comme :

$$h_t = tanh(W_h hh_(t-1) + w_x hx_t)$$

Le neurone récurrent dans le réseau neuronal récurrent prend en considération l'état immédiatement précédent pour maintenir la séquence.
Nous allons maintenant calculer l'état de la sortie car nous avons déjà l'entrée et la formule de calcul de l'état caché.

La formule mathématique pour calculer l'état de sortie : $y_t = w(hy)h_t$
Un bref résumé pour bien comprendre les étapes de fonctionnement :

- Un seul pas de temps de l'entrée est transmis au réseau, c'est-à-dire que x_t est transmis au réseau pour être transmis à l'étape suivante.

— Ensuite, l'état actuel sera calculé en utilisant une combinaison de l'entrée actuelle et de l'état précédent, c'est-à-dire h_t.
— Le h_t actuel devient $h(t-1)$ pour le prochain pas de temps afin de maintenir la séquence d'entrée.
— Des couches cachées peuvent être ajoutées en fonction de la complexité du programme.
— Une fois que tous les calculs d'entrée et d'étapes cachées sont terminés, l'état actuel final est utilisé pour calculer la sortie y_t.
— Ensuite, nous comparerons la sortie avec la sortie réelle.
— Sur la base des performances du modèle, nous calculerons l'erreur.
— Enfin, nous utiliserons la technique de rétropropagation pour réduire l'erreur et mettre à jour en conséquence le poids du réseau.

3.2.2.5 Rétropropagation dans le temps

Pour un RNN, nous utilisons une version légèrement différente de l'algorithme de rétropropagation du gradient : la rétropropagation dans le temps (Back-Propagation Through Time - BPTT). L'algorithme de BPTT consiste à appliquer récursivement les règles de dérivation des fonctions composées pour tous les pas de temps en parcourant le temps à rebours.

Pour comprendre le concept de rétropropagation à travers le temps, il faut d'abord comprendre les concepts de propagation avant et arrière.
Dans les réseaux de neurones, nous faisons essentiellement la propagation vers l'avant pour obtenir la sortie de notre modèle et vérifier si cette sortie est correcte ou incorrecte, pour obtenir l'erreur.

Remarque : On déduit l'erreur à partir de $(sortie - t)^2$ ou t : prédiction. À partir du dérivé partiel de cette erreur on obtient le gradient après nous utilisons les fonctions d'activations.

Après nous faisons Backward-Propagation, qui n'est rien d'autre que de revenir en arrière à travers notre réseau de neurones pour trouver les dérivées partielles de l'erreur par rapport aux poids, ce qui nous permet de soustraire cette valeur des poids.

Ces dérivées sont ensuite utilisées par **Gradient Descent**, qui est un des algorithmes les plus importants de tout Machine Learning et de tout Deep Learning. Il s'agit d'un algorithme d'optimisation extrêmement puissant qui permet d'entraîner les modèles de régression linéaire, régression logistique ou encore les réseaux de neurones.Et aussi est un algorithme utilisé pour minimiser itérativement une fonction donnée. Ensuite, il ajuste les poids vers le haut ou vers le bas, en fonction de ce qui diminue l'erreur. C'est exactement la façon dont un réseau de neurones apprend pendant le processus de formation.

Donc, avec Back propagation, nous essayons essentiellement de modifier les poids de notre modèle, pendant l'entraînement.
[2]

L'image ci-dessous illustre parfaitement le concept de la propagation vers l'avant et de la propagation vers l'arrière à l'exemple d'un réseau de neurones Feed Forward :

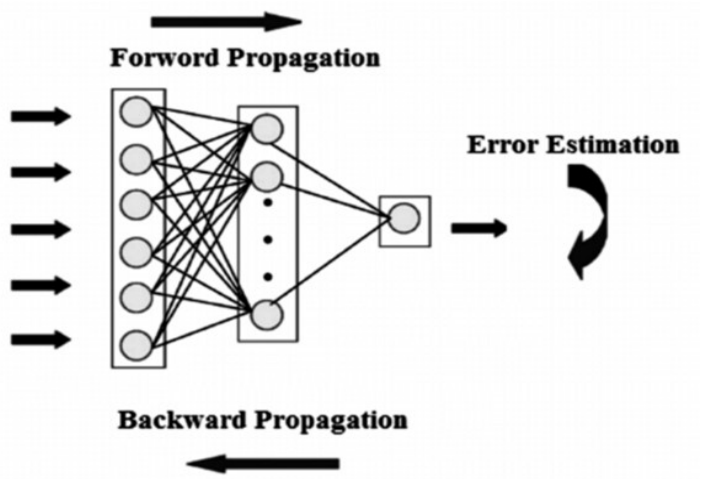

FIGURE 3.9 – Concept de propagation vers l'avant et vers l'arrière

3.2.2.6 Le problème des réseaux de neurones récurrents (RNN)

Les réseaux de neurones récurrents classiques sont exposés au problème de disparition de gradient lors de la rétropropagation qui les empêche de modifier leurs poids en fonction d'évènements passés. Lors de l'entraînement, le réseau essaie de minimiser une fonction d'erreur dépendant en particulier de la sortie.Qu'est-ce que cela signifie ?

Pour " apprendre ", un RNN utilise la méthode de la descente du gradient afin de mettre à jour les poids entre ses neurones. Cela repose sur la formule suivante : $W := w - \alpha F_w$
Avec :

- W un poids du réseau
- α la vitesse d'apprentissage du réseau
- F_w le gradient du réseau par rapport au poids w

Problème : la mise à jour des poids se fait de droite à gauche. A mesure que l'on avance vers la gauche, le produit $\alpha - Fw$ devient très petit et les poids des premières couches de neurones ne sont quasiment pas modifiés ! Ainsi, ces couches n'apprennent strictement rien...

Et par conséquent, le RNN peut facilement oublier des données un petit peu anciennes (ou des mots assez éloignés du mot courant dans un texte) lors de la phase d'apprentissage : **sa mémoire est courte**.[1]

3.2.3 Vers une meilleure architecture : le LSTM

Plusieurs variantes aux RNN standards ont vu le jour pour remédier aux problèmes évoqués précédemment. Nous allons ici décrire les LSTM (Long Short-Term Memory). Ce type de RNN est très utilisé en traitement du langage naturel.

L'idée derrière ce choix d'architecture de réseaux de neurones est de diviser le signal entre ce qui est important à court terme à travers le hidden state (analogue à la sortie d'une cellule de RNN simple), et ce qui l'est à long terme, à travers le cell state, qui sera explicité plus bas.

3.2.3.1 Définition du Long and Short Term Memory (LSTM)

Long Short Term Memory Network est un RNN avancé, un réseau séquentiel, utilisé dans les domaines de l'intelligence artificielle et de l'apprentissage en profondeur qui permet aux informations de persister. Il est capable de gérer le problème de gradient de fuite auquel est confronté RNN.

Les LSTM ont été créés comme méthode permettant de gérer efficacement la mémoire à court et long terme grâce à leurs systèmes de portes. S'il en existe de nombreuses variantes, les versions d'origine sont encore très largement utilisées dans les meilleurs modèles de Deep Learning pour le traitement automatique du langage naturel, ce qui a trait à la reconnaissance/synthèse vocale mais aussi pour la génération de texte ou l'étude de marchés... [1]

3.2.3.2 Où se situe LSTM dans l'univers de l'apprentissage automatique ?

Les réseaux de neurones (NN) se ramifient à partir du cœur de l'univers de l'apprentissage automatique. Les réseaux de neurones récurrents occupent une sous-branche des NN et contiennent des algorithmes tels que les RNN standard, les LSTM et les GRU.

Les figures ci-dessous nous résume les sous-branches de l'apprentissage automatique, les réseaux de neurones et les RNN.

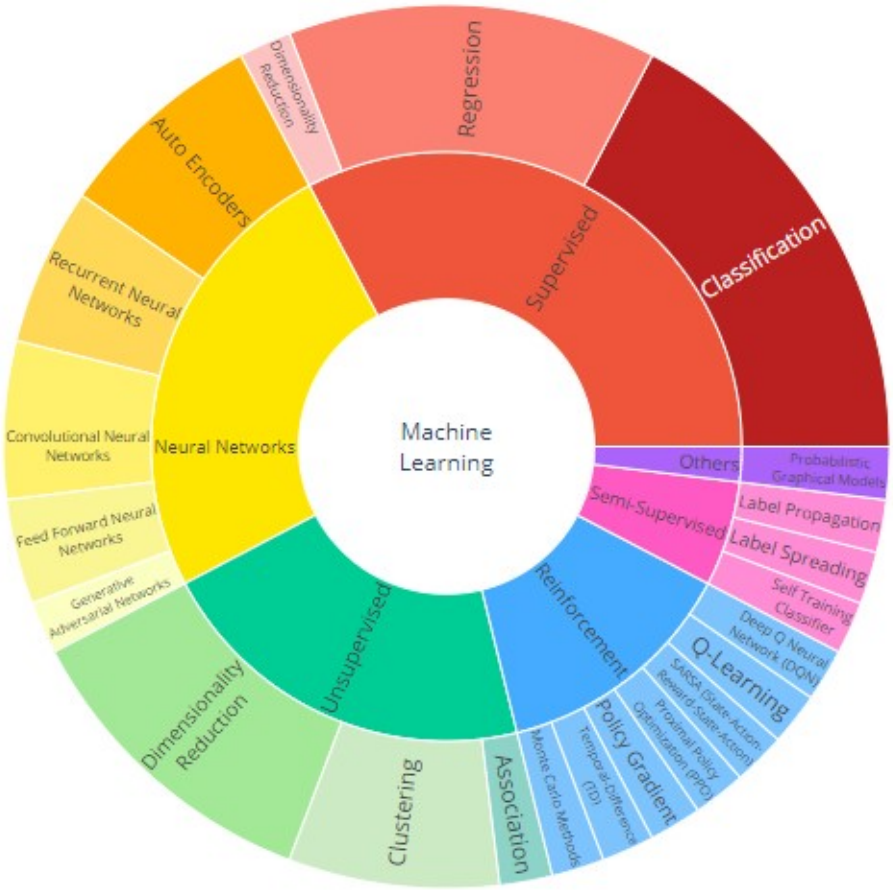

FIGURE 3.10 – Les Sous-branches de Machine Learning

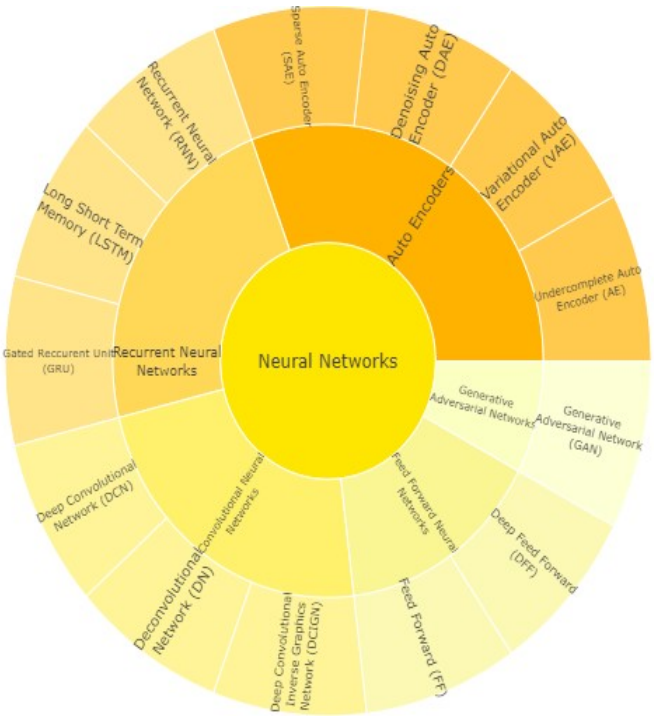

FIGURE 3.11 – Les Sous-Branches des réseaux de neurones

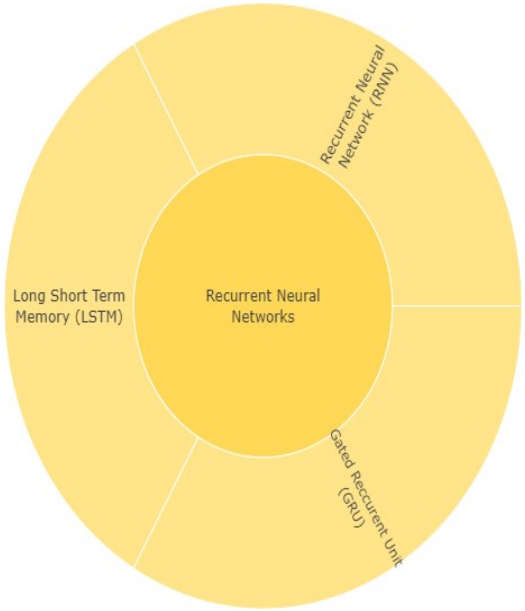

FIGURE 3.12 – Les sous-branches de RNN

3.2.3.3 Qu'est-ce qui différencie LSTM des RNN standard ?

Nous savons que les RNN utilisent des unités récurrentes pour apprendre des données de séquence. Tout comme les LSTM. Cependant, ce qui se passe à l'intérieur de l'unité récurrente est très différent entre les deux.

En regardant à l'intérieur du diagramme d'unités récurrentes simplifié d'un RNN standard (poids et biais non représentés), nous remarquons qu'il n'y a que deux opérations principales : combiner l'état caché précédent avec la nouvelle entrée et la faire passer par la fonction d'activation :

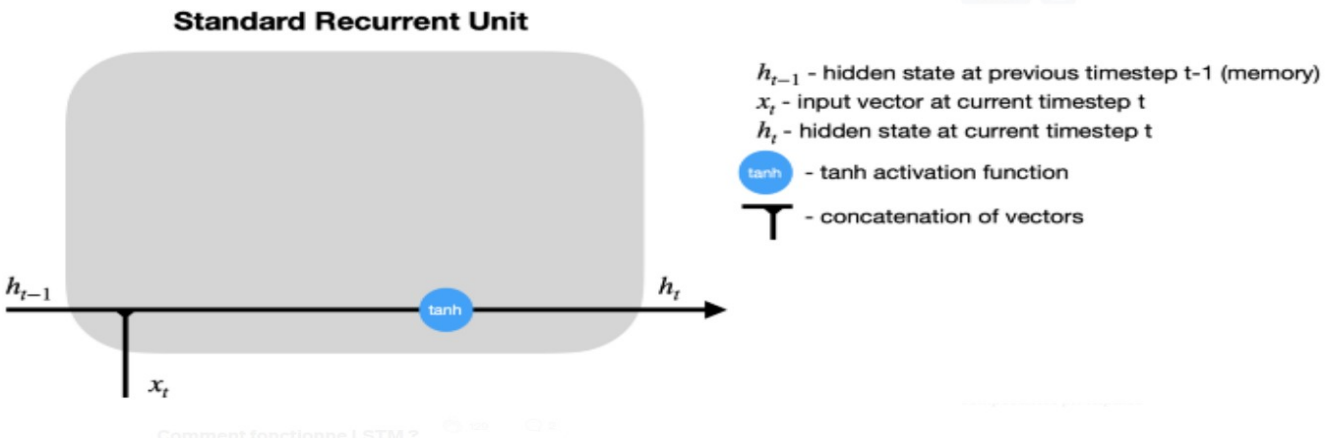

FIGURE 3.13 – unité récurrente RNN standard

Une fois l'état caché calculé au pas de temps t, il est renvoyé à l'unité récurrente et combiné avec l'entrée au pas de temps t+1 pour calculer le nouvel état caché au pas de temps t+1. Ce processus se répète pour t+2, t+3, ..., t+n jusqu'à ce que le nombre prédéfini (n) de pas de temps soit atteint.

Pendant ce temps, LSTM utilise diverses portes pour décider quelles informations conserver ou supprimer. En outre, il ajoute un état de cellule, qui est comme une mémoire à long terme de LSTM. Nous allons étudier ça dans le prochain titre.

3.2.3.4 Comment fonctionne le LSTM

LSTM, est une cellule composée de trois " portes " : porte d'oubli, porte d'entrée et porte de sortie qui sont des zones de calculs qui régulent le flot d'informations (en réalisant des actions spécifiques). On a également deux types de sorties (nommées états) qui sont état caché et état de la cellule.[1]

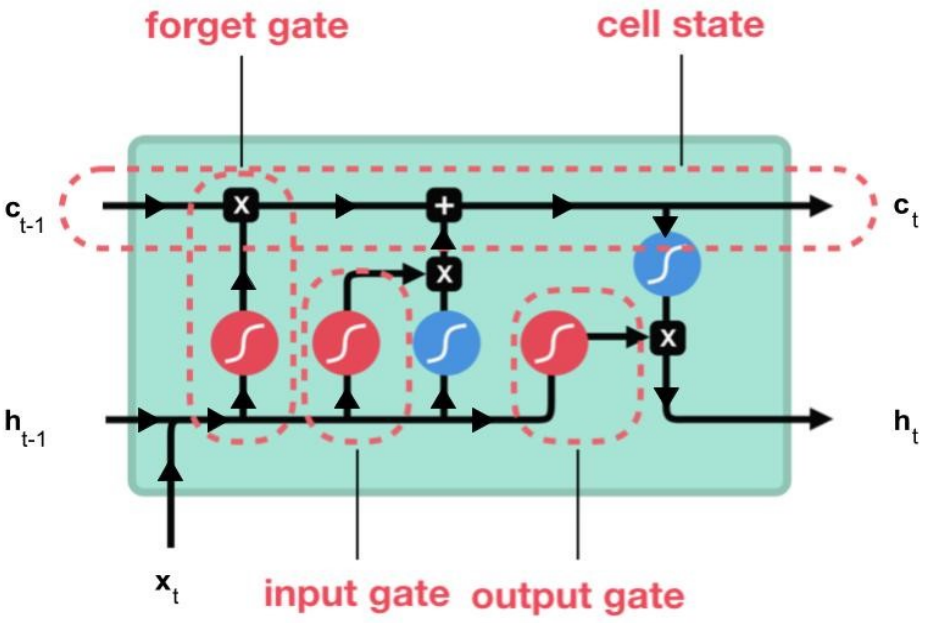

FIGURE 3.14 – Composant d'une cellule LSTM [1]

Qu'est-ce qu'une fonction d'activation ?

La fonction d'activation sert **avant tout à modifier** de manière **non-linéaire** les données. Cette non-linéarité permet de **modifier spatialement leur représentation**. Dit simplement, la **fonction d'activation** permet de **changer notre manière de voir une donnée**.

Chaque neurone d'une couche va **appliquer la fonction d'activation** de la couche sur les données. Cette **transformation** sera **différente selon chaque neurone** car chacun possède un **poids différent**.

Nous avons par exemple : Sigmoid et Tanh comme fonctions d'activations.

— **Sigmoid :** La fonction Sigmoïde donne une valeur entre 0 et 1, une probabilité. Elle est donc très utilisée pour les classifications binaires, lorsqu'un modèle doit déterminer seulement deux labels. Plus elle est proche de 0, plus elle est considérée comme négative.

$$fonction - Sigmoid(x) = 1/(1 + exp(-x))$$

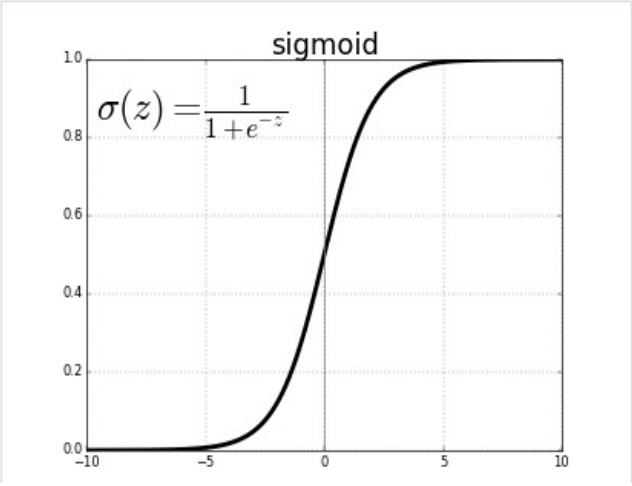

FIGURE 3.15 – Fonction d'activation Sigmoid

La fonction Sigmoïde est très simple a appliqué en Python car il n'y a pas de paramètre autre que la variable d'entrée :
tf.keras.activations.sigmoid(x)

— **Tanh :**La fonction tanh est simplement la fonction de la tangente hyperbolique. Il s'agit en fait d'une version mathématiquement décalée de la fonction sigmoïde :
— Sigmoïde donne un résultat entre 0 et 1
— Tanh donne un résultat entre -1 et 1
Tanh fonctionne mieux que la fonction sigmoïde dans la plupart des cas.

$$fonction - tanh(x) = sinh(x)/cosh(x)$$
$$fonction - tanh(x) = ((exp(x) - exp(-x))/(exp(x) + exp(-x)))$$

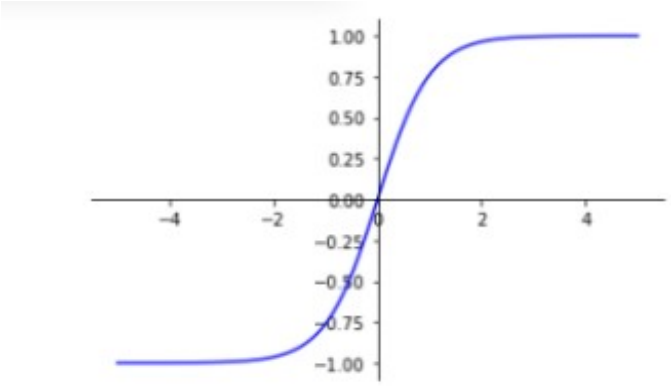

FIGURE 3.16 – Fonction d'activation Tanh

Dans la figure ci-dessous nous allons présenter quelques opérations possibles dans une cellule LSTM avant de passer à son fonctionnement.

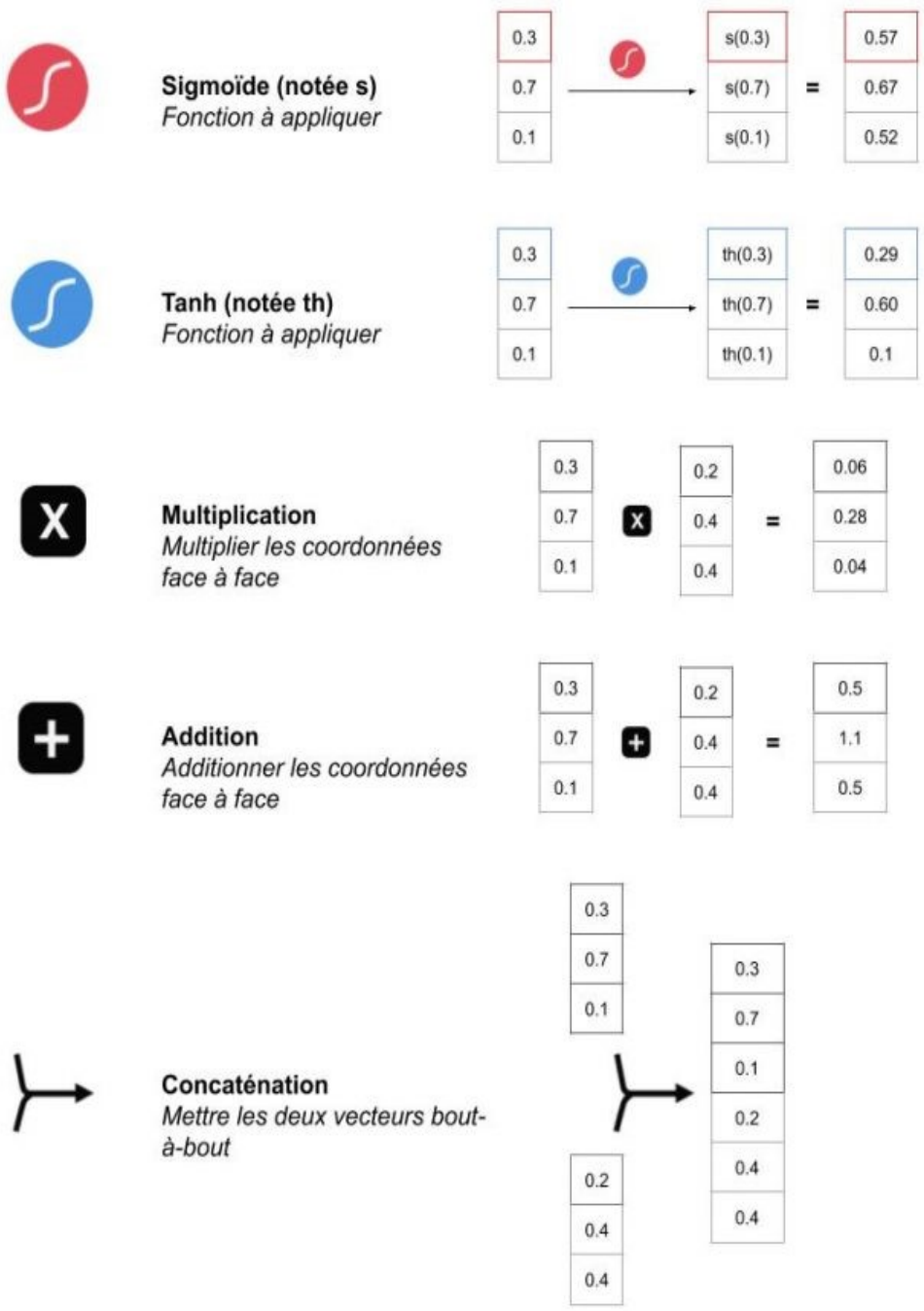

FIGURE 3.17 – Opérations internes que l'on retrouve dans le LSTM[1]

Ces opérations dans les portes permettent au LSTM de conserver ou supprimer des informations qu'il a en mémoire.

Les données stockées dans la mémoire du réseau sont en fait un vecteur noté ct : l'état de la cellule. Comme cet état dépend de l'état précédent ct-1, qui lui-même dépend d'états encore précédents, le réseau peut conserver des informations qu'il a vu longtemps auparavant (contrairement au RNN classique).

3.2.3.5 Comment fait le réseau LSTM pour apprendre, quelles sont ses variables internes ?

Les entrées de chaque porte sont pondérées par des poids liés aux portes ainsi que par un biais. On a 4 matrices de poids (leurs dimensions dépendent des dimensions de ht-1 et xt) :[1]

- **Wf :** pondère l'entrée de la porte d'oubli (forget gate)
- **Wi :** pondère l'entrée de la porte d'entrée (input gate)
- WC : pondère les données qui vont se combiner à la porte d'entrée pour mettre à jour l'état de la cellule (cell state)
- **Wo :** pondère l'entrée de la porte de sortie (output gate)

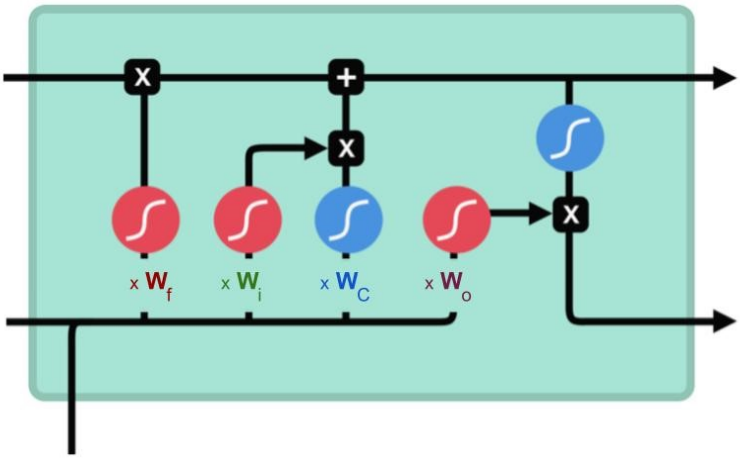

FIGURE 3.18 – Le placement des matrices de poids de chaque porte

Détaillons à présent ce que fait chaque porte, en gardant en mémoire que les données sont pondérées par les poids W (auxquels on ajoute un biais qui dépend de la porte et qui est aussi mis à jour dans la phase d'apprentissage).

- **Porte d'oubli (forget gate) :**Cette porte décide de quelle information doit être conservée ou jetée : l'information de l'état caché précédent est concaténée à la donnée en entrée puis on y applique la fonction sigmoïde afin de normaliser les valeurs entre 0 et 1. Si la sortie de la sigmoïde est proche de 0, cela signifie que l'on doit oublier l'information et si on est proche de 1 alors il faut la mémoriser pour la suite.
  Exemple : dans la prédiction des mots d'une nouvelle phrase, il n'est pas utile de garder le genre du sujet dans la phrase précédente.

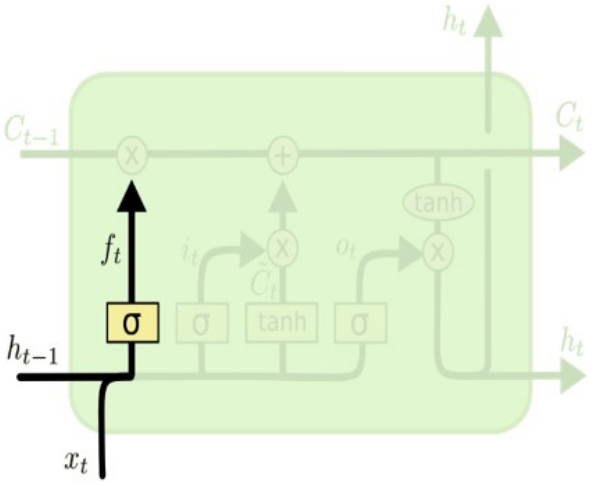

FIGURE 3.19 – Porte d'oublie [1]

— **Porte d'entrée (input gate) :** La porte d'entrée a pour rôle d'extraire l'information de la donnée courante, on va appliquer en parallèle une sigmoïde aux deux données concaténées (cf porte précédente) et une tanh.

1. Sigmoïde va renvoyer un vecteur pour lequel une coordonnée proche de 0 signifie que la coordonnée en position équivalente dans le vecteur concaténé n'est pas importante. A l'inverse, une coordonnée proche de 1 sera jugée importante (i.e. utile pour la prédiction que cherche à faire le LSTM).
2. Tanh va simplement normaliser les valeurs (les écraser) entre -1 et 1 pour éviter les problèmes de surcharge de l'ordinateur en calculs.
3. Le produit des deux permettra donc de ne garder que les informations importantes, les autres étant quasiment remplacées par 0

Exemple : dans la prédiction des mots d'une nouvelle phrase, il est utile de remplacer le genre du sujet de la phrase précédente par le nouveau.

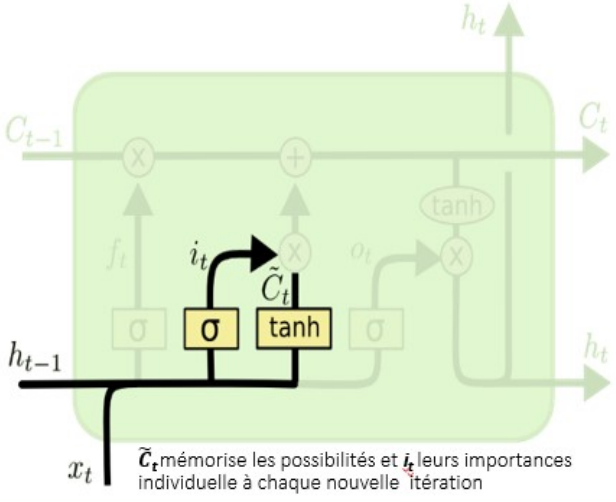

FIGURE 3.20 – Porte d'entrée [1]

— **Etat de la cellule (cell state) :** On parle de l'état de la cellule avant d'aborder la dernière porte (porte de sortie), car la valeur calculée ici est utilisée dedans. L'état de la cellule se calcule assez simplement à partir de la porte d'oubli et de la porte d'entrée : d'abord on multiplie coordonnée à coordonnée la sortie de l'oubli avec l'ancien état de la cellule. Cela permet d'oublier certaines informations de l'état précédent qui ne servent pas pour la nouvelle prédiction à faire. Ensuite, on additionne le tout (coordonnée à coordonnée) avec la sortie de la porte d'entrée, ce qui permet d'enregistrer dans l'état de la cellule ce que le LSTM (parmi les entrées et l'état caché précédent) a jugé pertinent.

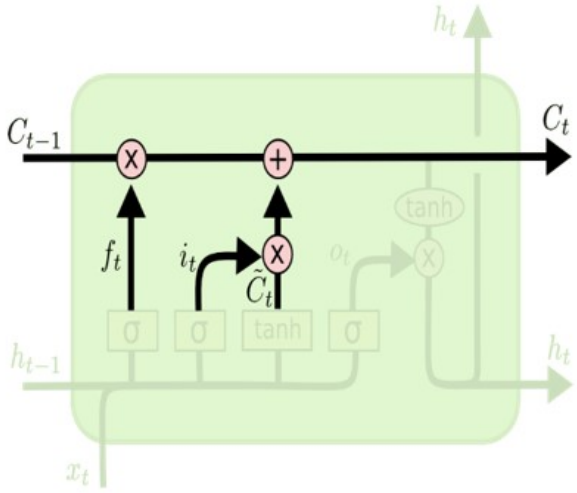

FIGURE 3.21 – Etat de la cellule [1]

— **Porte de sortie (output gate) :**Dernière étape : la porte de sortie doit décider de quel sera le prochain état caché, qui contient des informations sur les entrées précédentes du réseau et sert aux prédictions.Pour ce faire, le nouvel état de la cellule calculé juste avant est normalisé entre -1 et 1 grâce à tanh. Le vecteur concaténé de l'entrée courante avec l'état caché précédent passe, pour sa part, dans une fonction sigmoïde dont le but de décider des informations à conserver (proche de 0 signifie que l'on oublie, et proche de 1 que l'on va conserver cette coordonnée de l'état de la cellule).
Exemple : dans la prédiction des mots d'une nouvelle phrase, il peut être utile de retenir la pluralité du sujet à la vue d'un verbe, ce qui peut simplifier la conjugaison après.

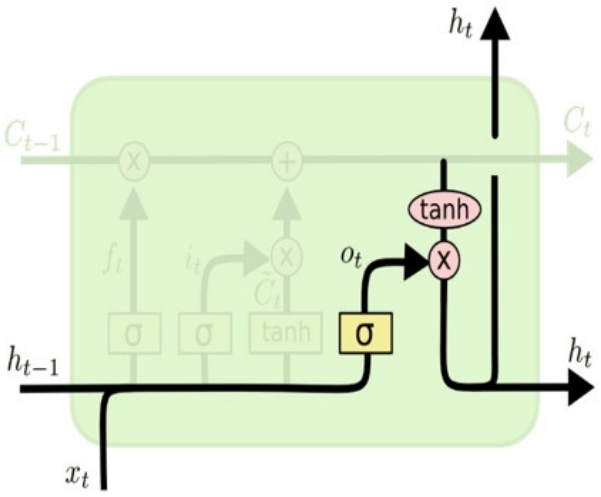

FIGURE 3.22 – Porte de sortie [1]

Tout cela peut sembler magique en ce sens où on dirait que le réseau doit deviner ce qu'il doit retenir dans un vecteur à la volée, mais rappelons bien qu'une matrice de poids est appliquée en entrée. C'est cette matrice qui va, concrètement, stocker le fait que telle information est importante ou non à partir des milliers d'exemples.

3.3 Description et conception de l'architecture de notre IDS

L'objectif de notre travail est de concevoir un système capable de prédire les attaques malveillantes dans un système. Nous présentons une architecture générale de notre systèmes basé sur la méthode LSTM pour détecter les attaques malveillantes dans un scénario IIoT.

3.3.1 Architecture générale du système :

Les données sont générées à partir des appareils IIoT via des capteurs. Le trafic sortant de ces appareils est ensuite collecté et observé via **le renifleur**. Ce dernier détecte les paquets sur un réseau et les stocke dans la base de données de trafic. La base de données peut être physique ou sur un cloud.

Les paquets stockés dans la base de données de trafic sont ensuite envoyés et fournis à un environnement **Sandbox** pour extraire les fonctionnalités statiques et dynamiques, respectivement. Ce travail considère **MobSF** comme un cadre de sécurité tout-en-un et est continuellement mis à jour et maintenu par les auteurs. De plus, MobSF peut être utilisé pour Android, Windows et IOS. Il existe également d'autres cadres de sécurité tels que **DroidBox, DroidScope et CuckooDroid** qui pourraient être utilisés à des fins similaires. Une fois que nous avons extrait les caractéristiques statiques et dynamiques, nous effectuons le processus de transformation **Phase Space Embeding (PSE) et Sparse Auto Encoder SAE**. Notre classificateur actif basé sur LSTM détecterait les attaques malveillantes ou inoffensives lorsqu'elle est en mode prédiction.

Si le modèle demande l'étiquette, la demande est envoyée à l'administrateur ou les informations sont extraites d'un Oracle (si disponible). Les informations récupérées seront ensuite utilisées pour mettre à jour les récompenses (basées sur la prédiction correcte, la prédiction erronée ou la demande d'étiquette) et le modèle est ré-entraîné en conséquence. Dans le cas où une application malveillante est détectée, l'administrateur système est averti par un système d'alerte pour prendre les mesures nécessaires et la signature de comportement du logiciel malveillant est stockée dans la base de données du journal pour une analyse plus approfondie. Encore une fois, la base de données des journaux fait référence au stockage physique ou au cloud.

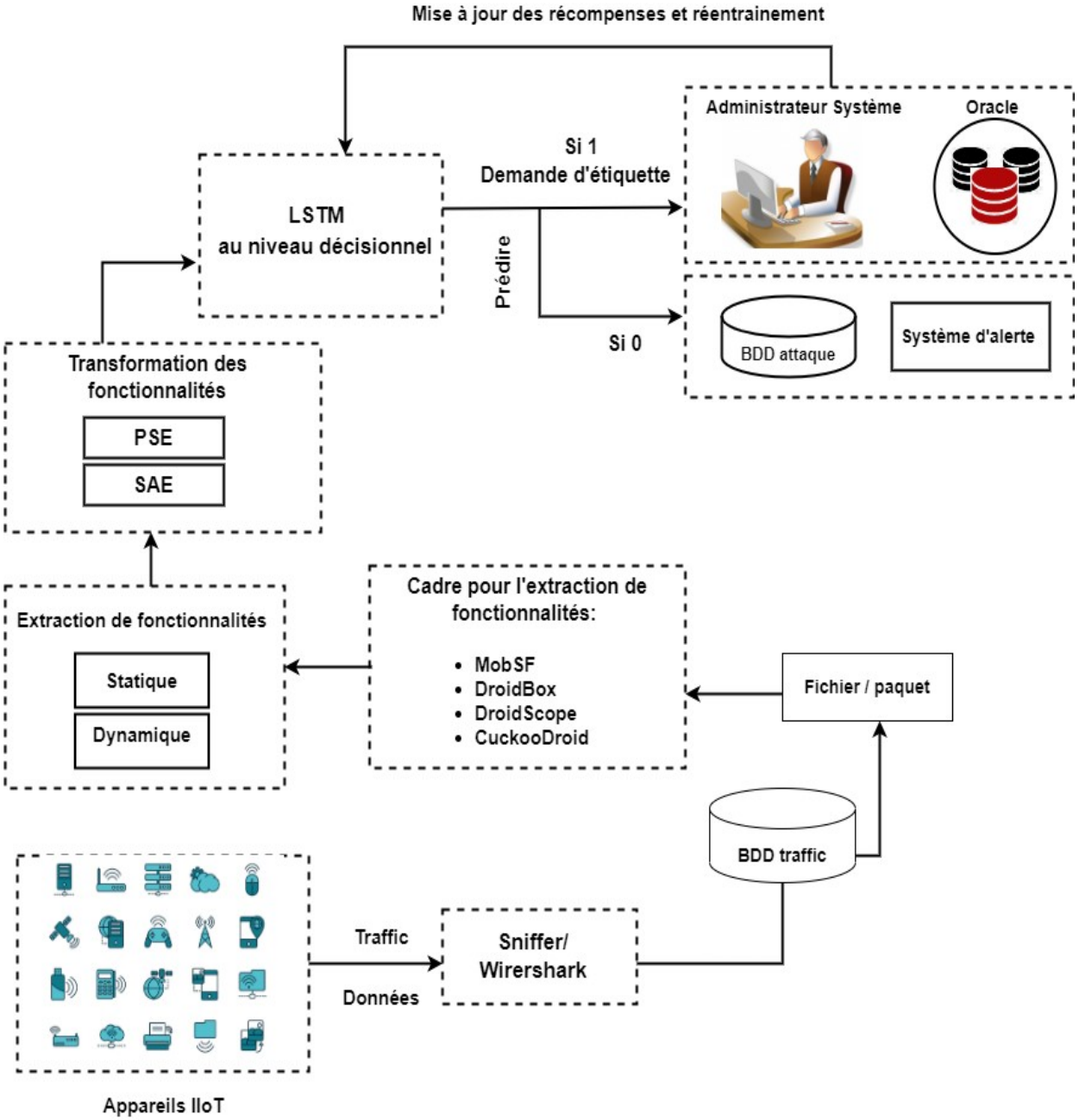

FIGURE 3.23 – Architecture générale du système

Dans notre cas on va s'intéresser sur la partie LSTM et apprentissage par renforcement au niveau décisionnel. Notre LSTM qui est placé dans la partie décision doit être capable de prendre des décisions fiables.
Pour que notre LSTM et l'agent de l'apprentissage par renforcement arrivent à prendre

une décision fiable, ils doivent tout d'abord s'entrainer sur des données déjà existantes qu'on appelle un **DATASET**.

— **Dataset :** Les datasets (ou jeux de données) sont couramment utilisés en Machine Learning. Un dataset est un ensemble cohérent de données produites dans le cadre d'un même projet, sur un même objet d'étude et/ou recueillies sur un même lieu. Toutes les données d'un dataset peuvent donc être décrites avec une majorité de métadonnées communes.

Tous les types de fichiers sont admis (tabulaire, texte, pdf, image, vidéo, audio, SHP, etc.), mais on choisira de préférence des formats ouverts et standards pour faciliter la réutilisation.

Les formats tabulaires RData, SPSS, STATA, CSV, XLSX seulement (xls n'est pas supporté) sont spécifiquement traités pour faciliter l'accès à leurs métadonnées internes et la visualisation de leur contenu.

Un dataset peut être composé d'une centaine de fichiers, sa taille pouvant atteindre quelques dizaines de gigaoctets.

Chaque valeur présente dans un dataset est associée à un attribut et à une observation. Prenons par exemple des données sur différentes personnes atteintes ou non du Covid-19. Les attributs correspondront à différentes caractéristiques telles que l'âge, le poids, la taille, la ville de résidence, les symptômes... Alors que chaque observation sera associée à une personne différente.[3]

Id_patient	Age	taille	poids	Ville de résidence	asymptomatique	symptomatique	symptome_difficulté_respiration	symptome_perte_odorat	symptome_fievre	réanimation
1	34	178	80	Paris	1	0	0	0	0	0
2	58	160	100	Strasbourg	0	1	1	1	1	1
3	26	165	60	Lille	0	1	1	1	1	0
4	19	174	80	Toulouse	0	1	0	1	0	0

FIGURE 3.24 – Dataset sur différentes personnes atteintes ou non du Covid-19

3.3.2 Le prétraitement des données du dataset

Afin de construire un modèle très précis, il est important d'effectuer des analyses exploratoires sur l'ensemble de données et ses caractéristiques. Le pré-traitement de l'ensemble de données est une technique d'exploitation de données brutes qui consiste à transformer ces dernières en un format compréhensible. Sachant que les données du monde réel sont dans la plupart des cas incomplètes, incohérentes ou dépourvues de certaines tendances, le prétraitement de celles-ci est une méthode certaine pour résoudre ces problèmes ainsi que de nombreuses erreurs susceptibles d'exister dans ces données. Il sera effectué avant d'être appliqué au réseau neuronal profond.

Certains algorithmes d'apprentissage (comme par exemple, les réseaux de neurones et les SVM) sont très sensibles à l'échelle des données. Par conséquent, il est standard d'ajuster les variables de façon à obtenir une représentation des données plus adaptée à ces algorithmes.
Le prétraitement des données pour l'apprentissage automatique consiste à suivre les étapes suivantes :

- **Étape 1 :** Importation des bibliothèques.
- **Étape 2 :** Importation de la dataset en question.
- **Étape 3 :** Vérification des valeurs manquantes.
- **Étape 4 :** Vérification des valeurs catégorielles.
- **Étape 5 :** Fractionnement de l'ensemble de données.
- **Étape 6 :** Mise à l'échelle des caractéristiques.

3.3.3 Phase d'apprentissage

Après le prétraitement du dataset, les données seront passées à un modèle pour l'apprentissage. Ce modèle a une superposition de plusieurs couches et couches d'abondons (Dropout). Cette dernière, permet de définir de manière aléatoire les unités d'entrée sur 0 avec une fréquence d'erreur à chaque étape pendant le temps d'entraînement, ce qui permet d'éviter les problèmes de sur ajustement. Lors de la phase d'apprentissage, pour chaque itération, un neurone est gardé avec une probabilité p, sinon il est supprimé.

La phase d'apprentissage consiste à trouver la bonne séquence de test.

3.3.4 Phase prédiction

Après la fin de la phase d'apprentissage de notre système, nous passons à la phase de prédiction ou nous utiliserons des données de test pour voir la précision de notre modèle.

Les données de test subissent les mêmes prétraitements appliqués aux données d'apprentissage. Donc, on commence par le traitement des données manquantes, puis la normalisation. Après le prétraitement de données de test, on charge le modèle et une prédiction peut avoir lieu et les résultats sont enregistrés. Par la suite, on peut utiliser les quatre mesures de performance pour évaluer le modèle à l'aide de la deuxième partie de données de test.

3.4 Conclusion

L'utilisation des RNN s'est avérée une solution optimale vu les résultats promoteurs obtenus dans plusieurs travaux de recherche.

Ce chapitre est un état de l'art sur les concepts reliés à la problématique traitée dans ce mémoire et sa conception. Nous avons abordé l'apprentissage automatique, RNN et LSTM avec plus de détail, pour montrer l'architecture que nous avons choisie pour notre système.

Dans le chapitre suivant, nous passons en revue le travail réalisé dans le cadre de la problématique traitée qui est la prédiction des attaques.

Chapitre 4

Implémentation et résultats

4.1 Introduction

Après une étude détaillée des IDS et des différentes méthodes d'apprentissage profond, nous sommes enfin arrivés à la phase finale de notre travail. Ce chapitre se divise en deux parties : la première partie est consacrée à l'implémentation du modèle proposé. Là où les différents choix d'implémentation sont présentés comme l'environnement de développement, les bibliothèques utilisées et les résultats de paramétrage effectué. La deuxième partie montre les différents tests réalisés, les résultats obtenus avec les discussions et les comparaisons avec d'autres.

4.2 Implémentation

Cette partie est réservée aux détails de l'environnement de développement et le langage de programmation utilisés pour la réalisation de notre système. Nous avons présenté aussi la base d'apprentissage et de test utilisée, les détails de l'implémentation de l'architecture proposée.

4.3 Environnement de développement

4.3.1 Anaconda

Anaconda est une distribution libre et open source des langages de programmation Python et R appliqué au développement d'applications dédiées à l'apprentissage automatique et à la science des données (traitement de données à grande échelle, analyse prédictive, calcul scientifique), qui vise à simplifier la gestion des paquets et de déploiement. Localement, on a fait une exploration et une analyse générale sur l'ensemble des données. La préparation des données aussi a été faite localement par plusieurs étapes et à l'aide des techniques qu'il n'exige pas beaucoup de calculs.

4.4 Langage de programmation et bibliothèques

4.4.1 Python

Python est un langage de programmation open source créé par le programmeur Guido van Rossum en 1991. Il tire son nom de l'émission Monty Python's Flying Circus. Le code open-source Python domine aujourd'hui largement l'informatique car tout en étant

simple, il est relativement puissant et dispose de plus de très nombreuses librairies qui permettent d'appeler des fonctions préprogrammées. La communauté des développeurs en Python est très active. On peut de plus trouver de très nombreux codes (ou parties de code) en open-source, des tutoriels en français, des vidéos explicatives (YouTube) et de très nombreuses aides en ligne sur ce langage.

4.4.2 Bibliothèques utilisées

- **TensorFlow :** nous avons utilisé cette bibliothèque pour définir les composants de base de l'architecture RNN-LSTM. Cette bibliothèque est destinée pour l'implémentation des algorithmes d'apprentissage automatique et profond, elle offre aussi une grande flexibilité dans le cadre de l'utilisation pour le développement d'un réseau des neurones.
- **Keras :**parmi les bibliothèques utilisées avec TensorFlow est Keras. Nous avons utilisé cette bibliothèque pour implémenter les différentes couches, les fonctions d'activation et la préparation de la base d'apprentissage.
- **NumPy :** apporte la puissance de calcul de langages comme C et Fortran à Python. NumPy est une bibliothèque Python optimisée pour le calcul numérique. Il ressemble beaucoup à MATLAB et est tout aussi puissant lorsqu'il est utilisé en conjonction avec d'autres packages tels que SciPy pour diverses fonctions scientifiques, Matplotlib pour la visualisation et Pandas pour l'analyse des données. NumPy est l'abréviation de python numérique. Nous avons utilisé cette bibliothèque pour adapter les types d'entrée selon la configuration du modèles utilisés, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux. Nous avons utilisé cette bibliothèque exactement dans le cas de balayage de l'image et l'extraction des fenêtres.
- **Matplotlib :**est une bibliothèque du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques, nous avons utilisé cette bibliothèque pour mieux visualiser les données sous formes de graphiques.
- **Sklearn :** est l'une des bibliothèques les plus utiles pour l'apprentissage automatique en Python. La bibliothèque sklearn contient de nombreux outils efficaces pour l'apprentissage automatique et la modélisation statistique, notamment la classification, la régression, le clustering et la réduction de la dimensionnalité.
- **Pandas :** est un outil open source rapide, puissant, flexible et facile à utiliser, construit sur le langage de programmation Python. Pandas est une bibliothèque Python spécialisée pour l'analyse de données, en particulier sur d'énormes ensembles de données. Il offre des fonctionnalités faciles à utiliser pour lire et écrire des données, traiter les données manquantes, remodeler l'ensemble de données et masser les données en découpant, indexant, insérant et supprimant des variables de données et des enregistrements. Pandas dispose également d'une importante fonctionnalité groupBy pour agréger des données pour des conditions définies - utile pour tracer et calculer des résumés de données pour l'exploration.

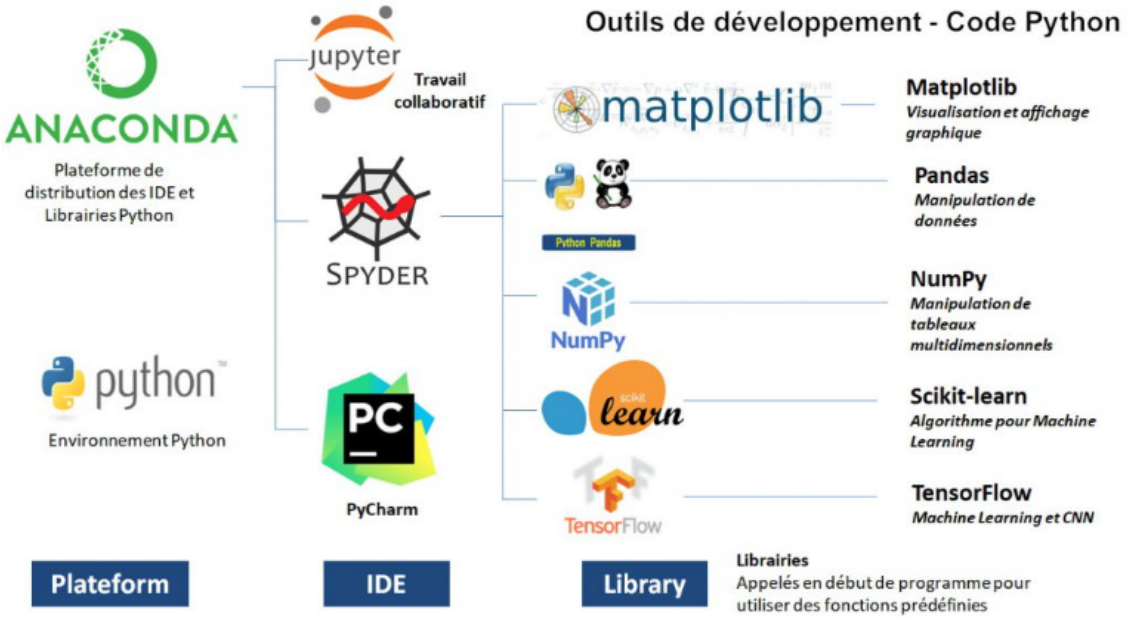

FIGURE 4.1 – Environnement de développement en Python pour le ML.

4.5 Base d'apprentissage et de test : NSL-KDD / Description de la base KDD

La base KDD est représentée sous un format texte ou chaque ligne représente une connexion caractérisée par ses 41 attributs (type de connexion, durée de la connexion, protocole utilisé... etc.) et chaque ligne est représentée comme normale ou une attaque. La base de données est accessible via le site **www.unb.ca/cic/datasets/nsl.html.**

L'ensemble de données d'entraînement se compose d'environ 4 900 000 vecteurs de connexion contenant chacun 41 caractéristiques et est étiqueté comme normal ou comme attaque, avec exactement un type d'attaque spécifique. La base KDD recense 38 attaques. Ces attaques simulées relèvent de l'une des quatre catégories suivantes :

- **Attaque par déni de service (DoS) :** Ce type d'attaque est très simple à mettre en oeuvre, le but de cette attaque est très simple, c'est de compromettre le fonctionnement et la disponibilité des ressources d'une organisation en temps normal.
- **User to Root Attack (U2R) :** est une classe d'exploit où l'attaquant essaye d'avoir les droits d'accès afin d'accéder au système.
- **Remote to Local Attack (R2L) :** se produit lorsqu'un attaquant qui a la capacité d'envoyer des paquets à une machine sur un réseau mais qui n'a pas de compte sur cette machine exploite une vulnérabilité à obtenir un accès local en tant qu'utilisateur de cette machine.
- **Probing Attack :** est une tentative de recueillir des informations sur un réseau d'ordinateurs dans le but apparent de contourner ses contrôles de sécurité.

4.6 Implémentation de l'architecture proposée

Notre application est constituée de trois phases : prétraitement des données, apprentissage et prédiction. Nous présentons dans cette section chaque module avec les détails de sa réalisation :

4.6.1 Phase de prétraitement

Dans cette phase nous préparerons les données pour qu'elles soient utilisées par notre modèle LSTM dans la phase d'apprentissage. Nous suivons plusieurs étapes qui sont les suivantes :

— Tout d'abord l'importation des bibliothèques python qui sont nécessaires pour le traitement des données, on y trouve Numpy, Pandas et matplotlib pour tracer et visualiser des données sous forme de graphiques.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

FIGURE 4.2 – Importation des bibliothèques

— Ensuite importer le dataset d'entrainement et le dataset de test pour qu'ils seront traités.

```
#importer les DATASETS
dataset_train = pd.read_csv('KDDTrain+.txt',header=None,names=col_names)
dataset_test = pd.read_csv('KDDTest+.txt',header=None,names=col_names)
```

FIGURE 4.3 – Importation des deux datasets

Comme mentionné avant, le Dataset est composé de 41 attributs, l'avant dernier attribut qui est " class " contient les types d'attaques que représente chaque paquet et on va les regrouper en 5 classes comme dans le tableau suivant :

Attaque par déni de service (DoS)	User to Root Attack (U2R)	Remote to Local Attack (R2L)	Probing Attack
apache2	buffer_overflow	Dict	illegal-sniffer
arppoison	casesen	framespoofsnmpget	ipsweep
back	eject	ftp_write	lsdomain
desnuke	fdformat	Ftpwrite	mscan
land	ffbconfig	guess_passwd	msscan
mailbomb	loadmodule	Httptunnel	nmap
neptune	ntfsdos	imap	ntinfoscan
pod	nukepw	multihop	portsweep
processtable	perl	named	queso
smurf	ps	ncftp	saint
syslogd	rootkit	netbus	satan
teardrop	sechole	netcat	selfping
tepreset	sqlattack	Phf	
ucpstorm	xterm	ppmacro	
udpstorm	yaga	sendmail	
crashiis	secret	snmpget	
		snmpgetattack	
		snmpguess	
		spy	
		sshtrojan	
		warezclient	
		warezmaster	
		worm	
		xlock	
		xsnoop	

TABLE 4.1 – les attaques regroupées en 5 classes.

Remarque : les paquets de classe " normal " restent comme ils sont.

— " Après le remplacement des types d'attaques avec les classes d'attaques on affiche le contenu de l'attribut " class " du Dataset d'entrainement comme dans la figure ci-dessous.

```
dataset_train['class'].value_counts()
```

```
normal    67343
dos       45927
probe     11656
r2l         995
U2r          52
Name: class, dtype: int64
```

FIGURE 4.4 – Nombre d'attaque par classe

— Comme notre modèle n'accepte pas des données de types String on sera dans l'obligation de coder toutes les données avec des chiffres numériques, le code dans la figure ci-dessous nous permet de coder les attributs qui contiennent des " strings

", les attributs sont " protocol_type, service, flag et class "

```
# Extraction d'étiquettes numériques à partir de données:
def cat_encode(df,col):
    return pd.concat([df.drop(col,axis=1), pd.get_dummies(df[col].values)], axis=1)
def log_trns(df,col):
    return df[col].apply(np.log1p)
cat_lst = ['protocol_type', 'service', 'flag','class']
for col in cat_lst:
    dataset_train = cat_encode(dataset_train,col)
    dataset_test = cat_encode(dataset_test,col)
log_1st = ['duration', 'src_bytes', 'dst_bytes']
for col in log_1st:
    dataset_train[col] = log_trns(dataset_train, col)
    dataset_test[col] = log_trns(dataset_test, col)
```

FIGURE 4.5 – Le code qui nous permet de convertir des strings en données numériques

— Une fois que tous les strings sont codés, on lit le manuel d'utilisation du DataSet pour bien extraire les informations essentielles ensuite on partage les données d'entrainement en X_train qui va contenir 39 attributs du DataSet et Y_train qui va contenir la donnée à prédire ou autrement dit la classe d'attaque qu'il va prédire, et ça sera la même chose pour les données de test avec x_test qui va contenir les données 39 attributs du DataSet et y_test va contenir la donnée à prédire. Par exemple avec le code suivant pour affecter les classes d'attaques à y_train : y_train=dataset_train[['U2r', 'dos', 'normal','probe','r2l']].

	U2r	dos	normal	probe	r2l
0	0	0	1	0	0
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	0	0
4	0	0	1	0	0
...	...	...	...	...	...
125968	0	1	0	0	0
125969	0	0	1	0	0
125970	0	0	1	0	0
125971	0	1	0	0	0
125972	0	0	1	0	0

FIGURE 4.6 – Y_train

— Une fois notre DataSet est partagé, on va ensuite normaliser les données de x_train et x_test. La normalisation est une méthode de prétraitement des données qui permet de réduire la complexité des modèles. C'est également un préalable à l'application de certains algorithmes.
La fonction utilisée est MinMaxScaler () qu'on va importer dans la bibliothèque sklearn.preprocessing. Pour chaque valeur dans le dataset on lui applique automatiquement la formule suivante : $m = (x - xmin)/(xmax - xmin)$ où :

1. m est notre nouvelle valeur,

2. x est la valeur de la cellule d'origine.
3. x min est la valeur minimale de la colonne,
4. x max est la valeur maximale de la colonne.

```
# Nous redimensionnons les fonctionnalités à [0, 1]:
min_max_scaler = MinMaxScaler()
x_train = min_max_scaler.fit_transform(x_train)
x_test = min_max_scaler.transform(x_test)
```

FIGURE 4.7 – Code pour normaliser les données du dataset

4.6.2 Phase de création du modèle et d'apprentissage

Dans cette phase nous allons parler tout d'abord de la création de notre modèle et ses différents paramètres utilisés ensuite de la phase d'entrainement.

- **La phase de création du modèle :** Pour la création d'un modèle LSTM tout d'abords on doit importer les bibliothèques nécessaires pour la manipulation de ce dernier. En premier on importe Tensorflow, ensuite la bibliothèque Keras qui est incluse dans Tensorflow. Et d'autre paramètres nécessaires tel que Sequantial, Dense et même LSTM. Nous expliquons quelques détails :

```
import tensorflow as tf
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
```

FIGURE 4.8 – Importation des bibliothèques nécessaires pour la création de notre modèle

(a) **Sequential :** est une pile de layers ou couche linéaire. Les layers peuvent être décrites de façon très simple.

(b) **Dense :** utilisé pour indiquer la densité de la couche de sortie, il contient 2 paramètres : le nombre de neurones de sortie et la fonction d'activation.

(c) **La couche Embedding :** Vu que le LSTM n'accepte que des données en 3D cette couche nous permet de mettre les données du DataSet en 3D. Il est composé de trois paramètres essentiels

i. **input_dim :** Il s'agit de la taille les données. Par exemple, si vos données sont codées en nombres entiers avec des valeurs comprises entre 0 et 10, la taille du vocabulaire serait de 11 mots.

ii. **output_dim :** C'est la taille de l'espace vectoriel dans lequel les mots seront intégrés. Il définit la taille des vecteurs de sortie de cette couche pour chaque mot. Par exemple, il pourrait être de 32 ou 100 ou même plus. Testez différentes valeurs pour votre problème.

iii. **input_length :** Il s'agit de la longueur des séquences d'entrée, comme vous le définiriez pour n'importe quelle couche d'entrée d'un modèle Keras. Par exemple, si tous vos documents d'entrée sont composés de 1000 mots, ce serait 1000.

(d) **input_shape :** doit être transmis sous forme de tuple (comprenant des entiers ou la valeur None, où None accepte tout entier positif). La dimension du lot n'est pas incluse dans input_shape. Ce paramètre nous permet de fixer la taille des sorties de la couche d'entrer.

Nous allons maintenant procéder à la création de notre modèle LSTM.

```
from keras.layers import Embedding

model = Sequential()
model.add(Embedding(128,39, input_length=39))
model.add(LSTM(39, input_shape=x_train.shape))
model.add(Dense(5,activation="softmax"))
model.summary()
model.compile(loss='mse', optimizer='adam',metrics=["AUC"])
```

FIGURE 4.9 – Création du modèle

- La compilation du modèle : Avant d'entraîner notre modèle, nous devons configurer le processus d'apprentissage en appelant la méthode compile qui accepte trois arguments :

(a) **Optimizer :** Il peut s'agir d'un optimiseur défini par son appellation, par exemple rmsprop ou adagrad, adam ou d'une instance de la classe Optimizer.

(b) **Loss :** Il s'agit de la fonction de coût que le modèle va utiliser pour minimiser les erreurs. Elle peut être définie par son appellation (exemple : Categorical-crossentropy ou Mse) ou bien cela peut être une fonction.

(c) **Metrics :** une liste de métriques. Dans le cas d'un problème de classification, nous utiliserons metrics=['accuracy'] mais cet argument peut spécifier une autre métrique en utilisant l'appellation de la métrique à utiliser ou une fonction de métrique sur mesure. Une métrique est une fonction utilisée pour évaluer les performances de notre modèle.

Grace à **model.summary()** on peut voir un résumé du nombre de paramètres que contient notre modèle ainsi ses couches.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 39, 39)            4992
_________________________________________________________________
lstm_1 (LSTM)                (None, 39)                12324
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 200
=================================================================
Total params: 17,516
Trainable params: 17,516
Non-trainable params: 0
```

FIGURE 4.10 – Le résumé du modèle

Nous pouvons même visualiser grâce à l'outils **graphviz** de python. Voici le résultat :

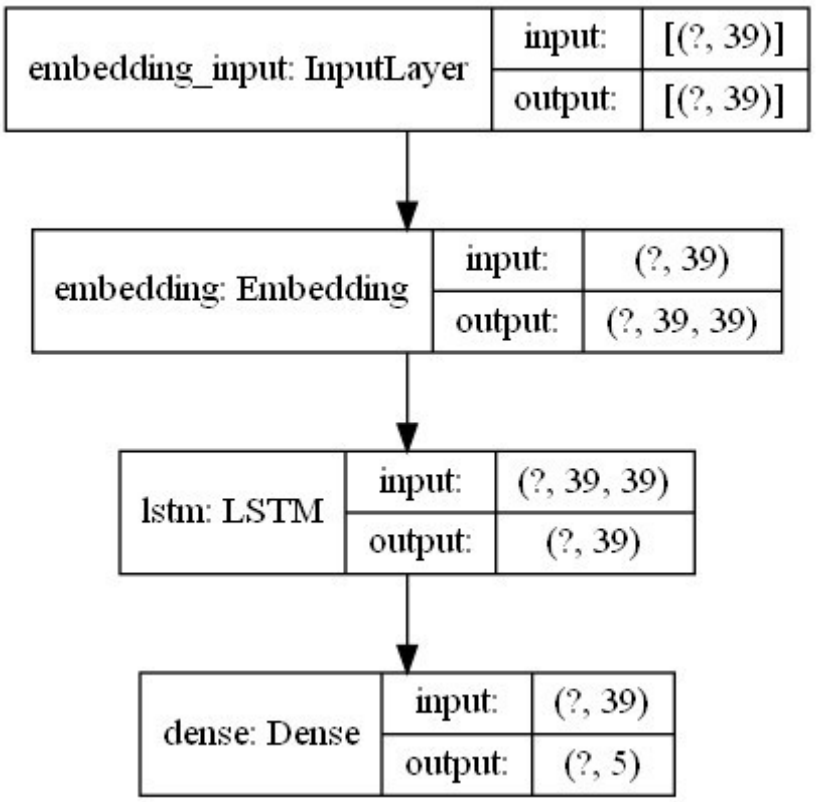

FIGURE 4.11 – Visualisation de notre modèle LSTM

- **La phase d'apprentissage :** Une fois la création de notre modèle est terminée on va entrainer notre modèle et aussi on doit spécifier quelques paramètres qui sont nécessaires ; on y trouve :
  - **Epoch :** Une époque est une itération sur l'ensemble x et y qui sont les données fournies. Il correspond à un apprentissage sur toutes les données, plus ce nombre est grand plus on devrait obtenir une bonne précision, mais bien sûr c'est plus long.
  - **Batch_size :** La taille du batch_size est un hyperparamètre qui définit le

nombre d'échantillons à traiter avant de mettre à jour les paramètres du modèle. S'il n'est pas spécifié, valeur par défaut sera 32.

— **Verbose :** 'auto', 0, 1 ou 2 c'est le mode de verbosité. 0 = silencieux, 1 = barre de progression, 2 = une ligne par époque. 'auto' par défaut est 1 dans la plupart des cas.

— **Validation_data :** pour tester directement notre modèle après chaque epoch, on procède à l'entrainement de notre modèle grâce à l'instruction model.fit. Dans notre cas on a spécifié le nombre d'epochs à 5 et le batch_size à 128.

```
history= model.fit(x_train, y_train, epochs=5,validation_data=(x_test,y_test),batch_size=128, verbose=1)
```

```
Epoch 1/5
985/985 [==============================] - 40s 40ms/step - loss: 0.0301 - auc: 0.9831 - val_loss: 0.1054 - val_auc: 0.8438
Epoch 2/5
985/985 [==============================] - 37s 38ms/step - loss: 0.0141 - auc: 0.9893 - val_loss: 0.1003 - val_auc: 0.8517
Epoch 3/5
985/985 [==============================] - 46s 47ms/step - loss: 0.0125 - auc: 0.9923 - val_loss: 0.0981 - val_auc: 0.8646
Epoch 4/5
985/985 [==============================] - 51s 51ms/step - loss: 0.0118 - auc: 0.9931 - val_loss: 0.1052 - val_auc: 0.8657
Epoch 5/5
985/985 [==============================] - 51s 52ms/step - loss: 0.0117 - auc: 0.9931 - val_loss: 0.1049 - val_auc: 0.8629
```

FIGURE 4.12 – Entrainement de notre modèle.

4.6.3 Phase de prédiction et d'évaluation

Dans cette partie, nous allons aborder comment prédire les valeurs. Donc une fois le modèle est formé, nous pouvons faire la prédiction. Pour prédire les valeurs on utilise **model.predict**.

```
x = model.predict(x_test)
print(x)
```

le code pour prédire les valeurs

```
[[3.6808709e-04 9.8563069e-01 2.2142951e-03 1.0577914e-02 1.2089738e-03]
 [3.6808709e-04 9.8563069e-01 2.2142951e-03 1.0577914e-02 1.2089738e-03]
 [8.4558407e-05 2.7918851e-03 9.8086208e-01 1.2218800e-02 4.0426846e-03]
 ...
 [3.6986382e-04 3.4372311e-04 9.9571437e-01 3.6599187e-04 3.2060510e-03]
 [2.9154000e-05 6.5633893e-04 9.9351358e-01 3.4817364e-03 2.3190391e-03]
 [1.5741575e-04 8.5181044e-04 6.2860197e-01 3.6582881e-01 4.5599351e-03]]
```

valeurs prédites

FIGURE 4.13 – Prédictions des valeurs.

Après avoir prédit les valeurs, on va s'intéresser à comment les modèles de prédiction en Machine Learning seront évalués, et plus précisément dans le cas de la classification. L'étape d'évaluation s'avère être une étape importante dans la démarche de choix du modèle, d'où la nécessité d'avoir un outil (matrice de confusion, classification-report ou accuracy-score) de précision qui nous permet de mener cette étape en bonne et due forme. Nous allons aborder ces modèles dans la section suivante.

4.7 Résultats, discussions et comparaison

4.7.1 Analyse et discussion des résultats de la classification :

Dans cette section, nous présentons et discutons les résultats de classification des données à l'aide des RNN-LSTM. Aussi, nous évaluerons et comparerons les performances utilisées pour estimer les performances de modèle de classification. Le Tableau suivant montre les résultats de cette section. Dans ce tableau, nous avons répertorié la fonction d'erreur, la métrique(metrics), la précision et la perte pour la phase d'évaluation du modèle après avoir entrainé ce dernier.

Fonction de perte (Loss)	Métrique	Précision	Perte
Categorical-crossentropy	Accuracy	10.6%	NAN
Categorical-crossentropy	Categorical-crossentropy	69%	183%
Mean Squared Error (Mse)	Accuracy	69.86%	4.97%
Mean Squared Error (Mse)	Area Under the Curve (AUC)	87%	5.61%

TABLE 4.2 – Performance des différentes méthodes de classification multi-labels.

D'après le tableau, nous remarquerons que la première ligne ne présente pas de bons résultats. Pour la 3ème et la 4ème ligne proposées et étudiées présentent de très bons résultats sur le plan décisionnel de manière générale, car chaque méthode peu être fiable dans un domaine par exemple la fonction de perte mse est parmi les techniques les plus adaptés pour les problèmes de classification.

On note que le taux de classification utilisant la fonction d'erreur MSE avec la métrique ACCURACY ou AUC est le plus élevé (69.86 % / 87%) contrairement à l'utilisation du Categorical-crossentropy comme fonction d'erreur et ACCURACY comme métrique qui avait le taux de classification le plus bas (10.6%).

En effet, le réseau RNN-LSTM proposé nous a permis de faire la classification des attaques de la base données avec une précision (AUC) très satisfaisante. La figure suivante donne les résultats obtenus pour l'entrainement et la validation de la base de données :

```
Epoch 1/5
985/985 [==============================] - 47s 48ms/step - loss: 0.0333 - auc: 0.9802 - val_loss: 0.1027 - val_auc: 0.8439
Epoch 2/5
985/985 [==============================] - 51s 51ms/step - loss: 0.0147 - auc: 0.9909 - val_loss: 0.1064 - val_auc: 0.8448
Epoch 3/5
985/985 [==============================] - 44s 45ms/step - loss: 0.0128 - auc: 0.9927 - val_loss: 0.1055 - val_auc: 0.8862
Epoch 4/5
985/985 [==============================] - 42s 43ms/step - loss: 0.0118 - auc: 0.9957 - val_loss: 0.1047 - val_auc: 0.8937
Epoch 5/5
985/985 [==============================] - 40s 41ms/step - loss: 0.0112 - auc: 0.9959 - val_loss: 0.1054 - val_auc: 0.8726
```

entrainement Test

FIGURE 4.14 – résultats obtenus pour l'entrainement et la validation..

Comme nous pouvons le constater clairement, notre réseau RNN-LSTM proposé a réussi son objectif qui est la classification des attaques de notre base de données, tout en

gardant un taux de précision nettement élevé et une perte réduite. Les figures suivantes présentent les graphes de la fonction ***loss*** ainsi que l'***accuracy*** obtenu pour les différents fichiers de la base de données, pour un nombre d'époques égal à 5 et ceci avec les deux ensembles de données : apprentissage et test (train et test).

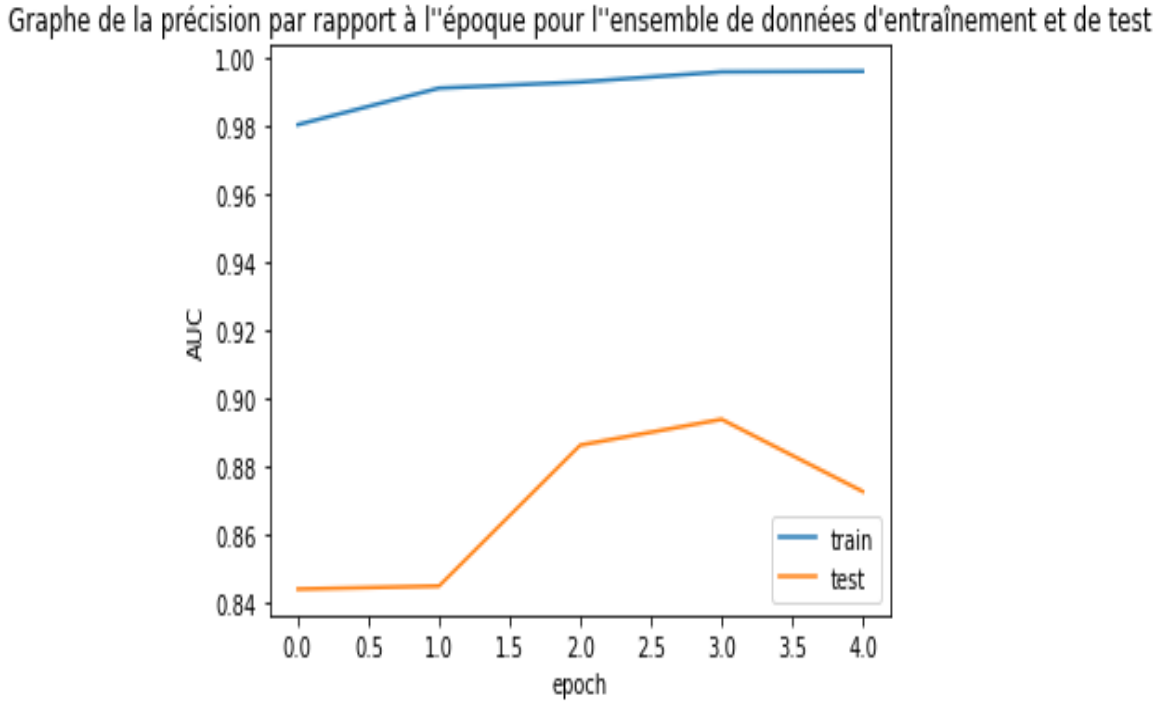

FIGURE 4.15 – graphe de précision pour le modèle.

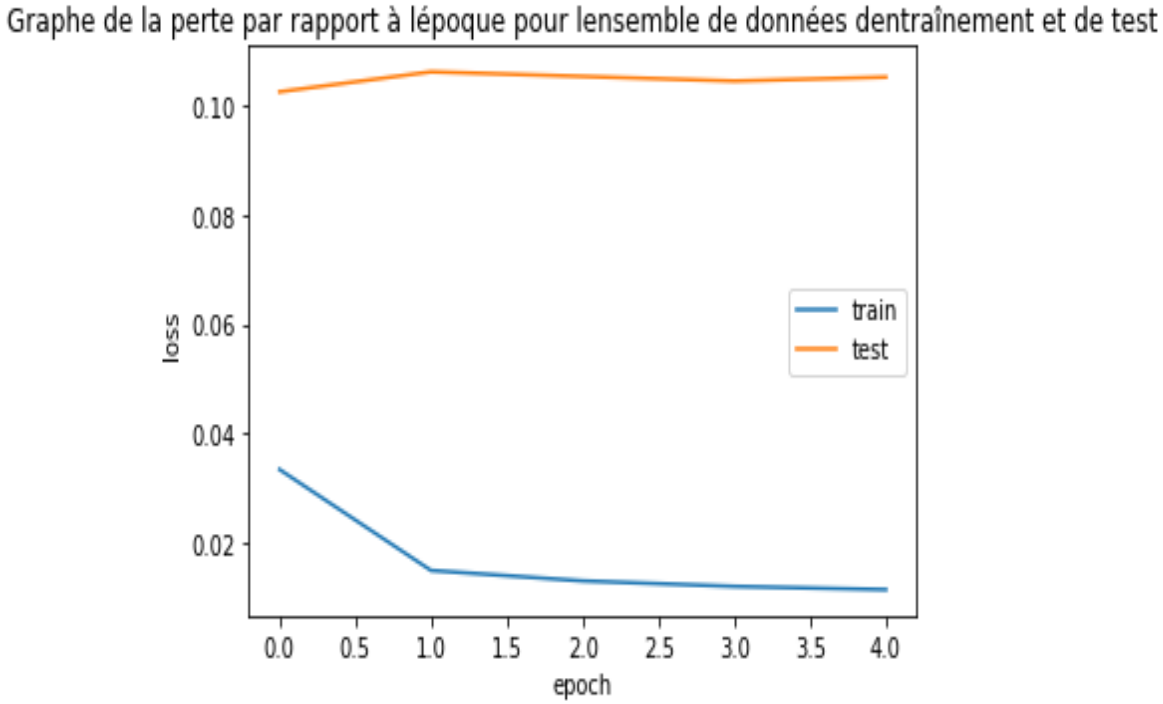

FIGURE 4.16 – graphe d'erreur pour le modèle.

Après l'analyse des résultats obtenus, on constate les remarques suivantes : D'après la

Figure 4.16, la précision de l'apprentissage et de test augmente avec le nombre d'époques, ceci reflète qu'à chaque époque le modèle apprenne plus d'informations. Si la précision est diminuée alors on aura besoin de plus d'informations pour faire apprendre notre modèle et par conséquent on doit augmenter le nombre d'époques et vice versa. De même dans la figure 4.15, l'erreur d'apprentissage et de la validation diminue avec le nombre d'époques.

Nous concluons, un réseau de neurone récurrent avec la méthode LSTM important et profond donne des bons résultats. Les résultats obtenus se sont améliorés à mesure que nous avons modifier la fonction d'erreur, la métrique et les fonctions d'activations et aussi augmenter et diminuer le nombre d'époques et les couches du notre réseau.

```
#prédiction # l'évaluation finale du modele # Final evaluation of the model

accr = model.evaluate(x_test,y_test)
print('Test set\n Perte : {:0.3f}%\n Précision : {:0.3f}%'.format(accr[0]*100,accr[1]*100))
```

```
705/705 [==============================] - 11s 16ms/step - loss: 0.1001 - accuracy: 0.7174
Test set
 Perte : 10.010%
 Précision : 71.740%
```

FIGURE 4.17 – Résultat de la simulation avec une autre fonction de perte et metric.

4.7.2 Comparaison des valeurs prédites avec les valeurs réelles

4.7.2.1 Manuellement

```
x = model.predict(x_test)
```

```
print(" Comparaison manuelle entre les valeurs prédits (x) et les valeurs réelles(y_test)")

print("pour la permier vecteur",np.round(x[0],0))
print("pour le deuxieme vecteur",np.round(x[1],0))
print("pour le 3eme vecteur",np.round(x[2],0))
print("pour le 4eme vecteur",np.round(x[3],0))
print("pour le 5eme vecteur",np.round(x[4],0))
print(" ")
print("... ... ... ... ... ... ... ... ")
print(" ")

print("pour la vecteur 22539 ",np.round(x[22539],0))
print("pour le vecteur 22540",np.round(x[22540],0))


print(" ")
print(y_test)
```

FIGURE 4.18 – code pour prédire les valeurs et les comparer avec les valeurs réelles.

Le résultat :

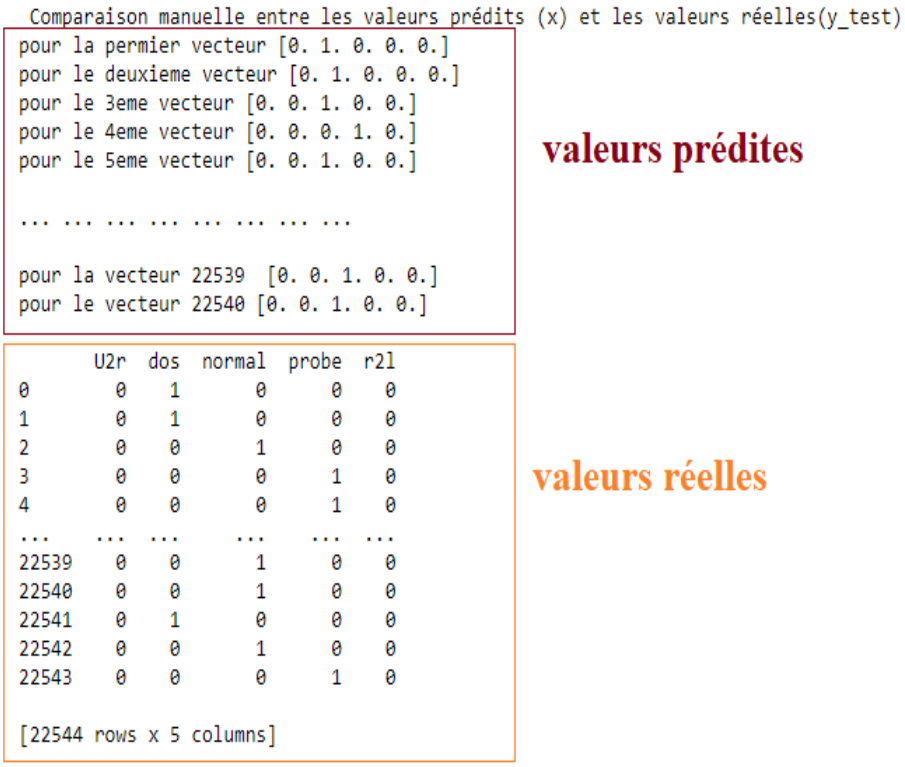

FIGURE 4.19 – résultats de la comparaison.

4.7.2.2 Avec les modèles de classification

Une fois que nous avons terminé la phase de formation du modèle LSTM. Nous pouvons évaluer les performances de classification de notre modèle en utilisant l'une de ces modèles : la classification-report, accuracy-score et la matrice de confusion.
Selon ces modèles, les performances de classification du notre modèle sont plutôt bonnes.

— **Classification-report :**
Un rapport de classification est utilisé pour mesurer la qualité des prédictions à partir d'un algorithme de classification. Combien de prédictions sont vraies et combien sont fausses. Plus précisément, les vrais positifs, les faux positifs, les vrais négatifs et les faux négatifs sont utilisés pour prédire les mesures d'un rapport de classification. Il affiche la précision, le rappel, le score F1 et le support de votre modèle. Il permet de mieux comprendre les performances globales de notre modèle entraîné.
Le code pour générer un rapport similaire à celui ci-dessus est :

```
targets = ['U2r', 'dos', 'normal','probe', 'r2l']
```

```
print(classification_report((y.argmax(axis=1)), y_pred.argmax(axis=1), target_names=targets))
```

FIGURE 4.20 – code pour classification-report.

Le résultat :

	precision	recall	f1-score	support
U2r	0.00	0.00	0.00	67
dos	0.91	0.67	0.77	7458
normal	0.69	0.95	0.80	9711
probe	0.65	0.62	0.63	2421
r2l	0.64	0.30	0.41	2887
accuracy			0.74	22544
macro avg	0.58	0.51	0.52	22544
weighted avg	0.75	0.74	0.72	22544

FIGURE 4.21 – résultat de la classification.

Pour comprendre le rapport de classification d'un modèle d'apprentissage automatique, nous devrons connaître toutes les métriques affichées dans le rapport.
Le rapport montre la précision, le rappel et le score f1 des principales métriques de classification par classe. Les mesures sont calculées en utilisant des vrais et des faux positifs, des vrais et des faux négatifs. Dans ce cas, positif et négatif sont des noms génériques pour les classes prédites. Il existe quatre façons de vérifier si les prédictions sont bonnes ou mauvaises :

1. **TN / True Negative :** lorsqu'un cas était négatif et prédit négatif
2. **TP / True Positive :** lorsqu'un cas était positif et prédit positif
3. **FN / False Negative :** lorsqu'un cas était positif mais prédit négatif
4. **PF / False Positive :** lorsqu'un cas était négatif mais prédit positif

■ **Précision - Quel pourcentage de prédictions était correct ?**
La précision est la capacité d'un classificateur à ne pas étiqueter une instance positive qui est en fait négative. Pour chaque classe, il est défini comme le rapport des vrais positifs à la somme des vrais et des faux positifs.
TP - Vrais positifs
FP - Faux positifs

Précision : Exactitude des prédictions positives.
Précision = TP/(TP + FP)

■ **Rappel - Quel pourcentage de cas positifs avez-vous attrapé ?**
Le rappel est la capacité d'un classificateur à trouver toutes les instances positives. Pour chaque classe, il est défini comme le rapport des vrais positifs à la somme des vrais positifs et des faux négatifs.
FN - Faux négatifs

Rappel : Fraction de positifs qui ont été correctement identifiés.
Rappel = TP/(TP+FN)

■ **Score F1 - Quel pourcentage de prédictions positives étaient correctes ?**
Le score F1 est une moyenne harmonique pondérée de précision et de rappel telle que le meilleur score est de 1,0 et le pire est de 0,0. De manière générale, les scores F1 sont inférieurs aux mesures de précision car ils intègrent la précision et le rappel dans leur calcul. En règle générale, la moyenne pondérée de F1 doit

être utilisée pour comparer les modèles de classificateur, et non la précision globale.

Score F1 = 2*(Rappel * Précision) / (Rappel + Précision)

— **Accuracy-score**

La précision du modèle est une métrique de performance du modèle de classification d'apprentissage automatique qui est définie comme le rapport des vrais positifs et des vrais négatifs à toutes les observations positives et négatives. En d'autres termes, la précision nous indique à quelle fréquence nous pouvons-nous attendre à ce que notre modèle d'apprentissage automatique prédise correctement un résultat sur le nombre total de fois qu'il a fait des prédictions.
Mathématiquement, il représente le rapport de la somme des vrais positifs et des vrais négatifs sur toutes les prédictions.

Score de précision = (TP + TN)/ (TP + FN + TN + FP)

```
from sklearn.metrics import accuracy_score

print(accuracy_score(y.argmax(axis=1),y_pred.argmax(axis=1))*100,"%")
```

```
73.58942512420155 %
```

résultats de précision obtenu

FIGURE 4.22 – code et résultats de l'accuracy-score.

— **Confusion matrix**

La Matrice de confusion est généralement utilisée dans le domaine du Machine Learning et plus précisément dans les problèmes de type classification statistique, également appelée matrice d'erreur.

Cette matrice permet de vérifier la performance d'un classifieur, ainsi que la confusion qui règne entre les différentes classes, sur un ensemble de données de test dont les vraies valeurs sont connues.

Elle résume et englobe les prédictions effectuées par le modèle de classification, les informations qu'elle nous transmet, nous permettent de voir la confusion entre les classes, et de distinguer les erreurs commises par ses dernières, ainsi que leurs natures.

		Classe réelle	
		-	+
Classe prédite	-	True Negatives *(vrais négatifs)*	False Negatives *(faux négatifs)*
	+	False Positives *(faux positifs)*	True Positives *(vrais positifs)*

FIGURE 4.23 – Matrice de confusion .

Le code :

```
conf_matrix = confusion_matrix(y_true=(y.argmax(axis=1)), y_pred=(y_pred.argmax(axis=1)))
#
# Afficher la matrice de confusion en utilisant Matplotlib
#
fig, ax = plt.subplots(figsize=(5, 5))
ax.matshow(conf_matrix, cmap=plt.cm.Oranges, alpha=0.3)
for i in range(conf_matrix.shape[0]):
    for j in range(conf_matrix.shape[1]):
        ax.text(x=j, y=i,s=conf_matrix[i, j], va='center', ha='center', size='xx-large')

plt.xlabel('Predictions', fontsize=18)
plt.ylabel('Actuals', fontsize=18)
plt.title('Confusion Matrix', fontsize=18)
plt.show()
```

FIGURE 4.24 – code pour tracer la matrice de confusion..

Le résultat : Dans notre cas, on est dans le cas multi-classes avec 5 classes.

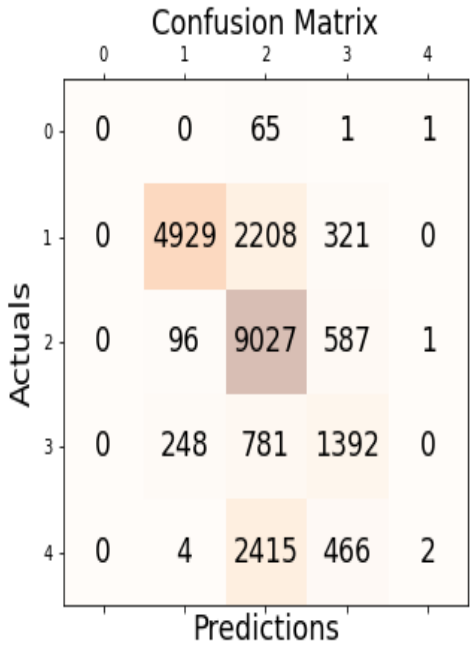

FIGURE 4.25 – résultats obtenus de la matrice de confusion.

4.8 Conclusion

Ce dernier chapitre décrit, d'une manière détaillée, les outils utilisés pour réaliser le travail expérimental et fournit les résultats obtenus avec une analyse et des tests afin d'obtenir des résultats en termes de précision et d'erreur. L'implémentation que nous avons proposée permet de détecter les paquets anormaux (attaques) et les paquets normaux avec un taux de précison élevé et une erreur faible.

Conclusion Générale

Les réseaux informatiques ont évolué au fil du temps à une vitesse vertigineuse. L'interconnexion de ces réseaux à internet les a directement exposés aux menaces informatiques.

Le travail que nous avons réalisé porte essentiellement sur une problématique majeure qui vise à détecter les attaques, et cela en introduisant l'une des plus utilisée des méthodes de réseaux de neurones récurrents qui sont les réseaux à mémoire longue à court terme (LSTM).

L'étude que nous avons menée, nous a conduits à découvrir l'une des mesures de sécurité à déployer, pour assurer la sécurité d'un réseau informatique qui est l'IDS. Nous avons présenté l'aspect théorique sur la sécurité des réseaux informatique, l'IDS et toutes les notions qui s'y rapportent, on a discuté aussi des notions fondamentales de l'apprentissage les plus populaires, des réseaux de neurones en général et des réseaux de neurones récurrents plus précisément et les LSTM en particulier. Nous avons introduit ce réseau LSTM en présentant les différents types de couches utilisées dans la classification.

Le résultat des tests de notre système est satisfaisant, mais cela ne veut pas dire que notre système est parfaitement efficace, car aucun système de sécurité permettant de garantir une sécurité totalement fiable à 100

Bien que les objectifs visés, au préalable, ont été atteints, mais il reste toujours des perspectives et des améliorations possibles qui peuvent encore être réalisés dans le futur, telles que :

• L'amélioration de ce travail en utilisant d'autres méthodes du monde du deep learning et faire une étude comparative entre ces dernières au niveau des résultats, performance et rapidité.

• Penser à la création d'un modèle capable de capturer le trafic réseau en temps réel sans avoir besoin d'utiliser l'ensemble de données NSL-KDD ou autre.

Ce projet nous a permis d'approfondir et enrichir nos connaissances, dans le domaine de la sécurité et l'apprentissage profond, il nous a permis, aussi de connaître et expérimenter de nouvelles bibliothèques python dédiées à ce domaine et de savoir le mode de fonctionnement de plusieurs algorithmes de l'apprentissage automatique tels que les différentes architectures du deep Learning.

Bibliographie

[1] https ://penseeartificielle.fr/comprendre-lstm-gru-fonctionnement-schema/, consulté le 15 juin 2022.

[2] https ://www.datasciencetoday.net/index.php/fr/machine-learning/148-reseaux-neuronaux-recurrents-et-lstm, consulté le 15 juin 2022.

[3] https ://datascientest.com/dataset-definition, consulté le 15 juin 2022.

[4] https ://data.ird.fr/definitions/Quest-ce_quun_dataset_ensemble_ou_jeu_de_donnees, consulté le 15 juin 2022.

[5] Attaques courantes. https ://blog.netwrix.fr/2018/07/04/les-10-types-de-cyberattaques-les-plus-courants, consulté le 20 mai 2022.

[6] https ://www.w3schools.com/ai, consulté le 22 juin 2022.

[7] https ://www.researchgate.net/figure/Architecture-classique-dun-IDS_fig2_30514332, consulté le 30 mai 2022.

[8] http ://igm.univ-mlv.fr/ dr/XPOSE2009/Sonde_de_securite_IDS_IPS/IDS.html, consulté le 30 mai 2022.

[9] http ://www-igm.univ-mlv.fr/ dr/XPOSE2004/IDS/IDSSnort.html, consulté le 30 mai 2022.

[10] https ://stanford.edu/ shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks, consulté le 8 juin 2022.

[11] https ://prograide.com/pregunta/40047/plusieurs–un-et-plusieurs–plusieurs-exemples-de-lstm–keras, consulté le 8 juin 2022.

[12] https ://datavalue-consulting.com/deep-learning-reseaux-neurones-recurrents-rnn/, consulté le 8 juin 2022.

[13] Sécurité au niveau physique. https ://www.securiteinfo.com/conseils/securite-physique-et-logique-du-materiel-informatique.shtml, Mis en ligne en 2004, consulté le 20 mai 2022.

[14] Serge Aumont and Claude Gross. L'impact de la lutte contre le spam et les virus sur les architectures de messagerie. *Actes du congrès JRES*, Mis en ligne en 2005,consulté le 25 mai 2022.

[15] M. TOUATI Azeddine. Master professionnel en informatique : Détection d'intrusions dans les réseaux lan :installation et configuration de l'ids-snort. *Université A /Mira de Bejaïa*.

[16] Le blog de René. Sécuritè informatiques - les techniques d'attaque. https ://www.rene-reyt.fr/documents/informatique/petit-resume-de-securite-informatique/securite-informatique-les-techniques-dattaque/, Mis en ligne en 2004,consulté le 23 mai 2022.

[17] Denis de REYNAL. Présentation sur les vpn. *Actes du congr es JRES*, Mis en ligne en 2004,consulté le 25 mai 2022.

[18] Mr Mohamed Abd Elmoumen DJABALLAH. Mémoire master : Système de prédiction de la consommation d'énergie basé deep learning. *Université de 8 Mai 1945 - Guelma -*.

[19] Melle AMICHE SELYNA. Melle BEN BRAHIM EMBARKA. Mémoire master : Mise en place d'une solution de détection d'intrusion. *UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU*.

[20] Cédric Michel. *Langage de description d'attaques pour la détection d'intrusions par corrélation d'événements ou d'alertes en environnement réseau hétérogène*. PhD thesis, Universite Rennes 1, 2003.

[21] Mr. LAIB Hamza Mr. BABAALI Baligh. Master en informatique : Approche basée sur l'apprentissage profond pour la détection d'intrusion réseau. *UNIVERSITE YAHIA FARES - MEDEA*.

[22] Amor Mrabet. *Les exigences de sécurité informatique de la plateforme de la carte à puce privative pétrolière*. PhD thesis, Université Virtuelle de Tunis, 2015.

[23] Mme Labraoui N. *Sécuritè Informatique, Chapitre 1 : Notions Fondamentales. Master 1 Réseaux et systèmes distribuès*. PhD thesis, Université Abou Bakr Belkaid, 2019/2020.

[24] BIONDI Philippe. *Architecture expérimentale pour la détection d'intrusions dans un système informatique.* PhD thesis, 2001.

[25] Mdme Heddaoui Rebiha. Comment fonctionne un ids ? https ://geekflare.com/fr/ids-vs-ips-network-security-solutions/.

[26] Mdme Heddaoui Rebiha. Mémoire de magister. *Actes du congrès JRES*.

[27] S.Belattaf. 3eme annee : Sécurité informatique. *DEPARTEMENT*, 2017/2018.

[28] s.d. Anatomie d'une attaque. http ://www-igm.univ-mlv.fr/ dr/XPOSE2007/plebacco_ids/A_attaque.html, consulté le 21 mai 2022.

[29] s.d. Types d'attaques. https ://www.securiteinfo.com/attaques/hacking/typesattaques.shtml, Mis en ligne en 2004,consulté le 23 mai 2022.

[30] s.d. Sécuritè au niveau logique. https ://www.departement-ti.com/2019/11/18/les-aspects-de-securite-dun-centre-de-donnees/, Mis en ligne en 2019, consulté le 21 mai 2022.

[31] Slide Serve. Principe de sécuritè. plan. critéres de sécuritè et fonctions associées. https ://www.slideserve.com/miron/plan, Mis en ligne en 2014,consulté le 21 mai 2022.

[32] Kai Yang Jie Ren Yanqiao Zhu and Weiyi Zhang. Active learning for wireless iot intrusion detection.