وزارة التعليم العالي والبحث العلمي

BADJI MOKHTAR UNIVERSITY – ANNABA

UNIVERSITE BADJI MOKHTAR – ANNABA

جامعة باجي مختار – عنابة

Faculté : Sciences de l'Ingéniorat
Département : Informatique

THESE

Présenté en vue de l'obtention du diplôme de : Doctorat 3éme cycle

Intitulé

Système de Détection d'Intrusion Adaptatif et Distribué

Domaine : Math-Informatique
Filière : Informatique
Spécialité : Réseaux et Sécurité Informatique

Par : Mr. Ahmim Ahmed

DIRECTEUR DE THESE : Mme. GHOUALMI-Zine Nacira Pr (MCA)
Université Badji Mokhtar-Annaba

DEVANT Le JURY

PRESIDENT :	Natalia Djellab	Pr.	UBMA
RAPPORTEUR :	Nacira Ghoualmi-Zine	Pr.	UBMA
EXAMINATEUR :	Med Benmohammed	Pr.	U. Constantine 2
EXAMINATEUR :	Abdallah Boukerram	Pr.	U. Bejaia
EXAMINATEUR :	Hayet Farida Merouani	Pr.	UBMA

Année 2014

ملخص

من أجل ضمان تطبيق السياسة الأمنية لأنظمة الإعلام الآلي العديد من الأدوات الأمنية المختلفة طورت، من بين هذه الأدوات نجد أنظمة كشف التسلل (Intrusion detection system). أنظمة كشف التسلل هي كل أداة أو وسيلة تساعدنا على التنبؤ أو تحديد أي نشاط غير مشروع أو غير طبيعي في الشبكة. هذا وقد شهد تطور أنظمة كشف التسلل جيلين. الجيل الأول هو الجيل العشوائي، الذي أظهر العديد من أشكال المحدودية مع الكميات الكبيرة من المعلومات التي تمر عبر الشبكة خاصة مع التطور الحالي للشبكات من حيث القدرة علي نقل المعلومات. أقترح الجيل الثاني للتعامل مع مشاكل الجيل الأول من أنظمة كشف التسلل مثل التصميم على أساس معرفة الخبراء، والعلاقة الوطيدة مع البيئة الهدف، والصعوبة المرتبطة بالتقييم، والنقص في قوة الأداء. ويستند هذا الجيل الثاني على تقنية التنقيب عن البيانات(Data mining) ، ولهذا الجيل العديد من المزايا مثل القدرة على تحليل كميات كبيرة من البيانات، الاستقلالية عن الخبراء، التحكم الذاتي، السرعة والتحسين في قوة الأداء. على الرغم من كل هذه المزايا لهذا الجيل الثاني من أنظمة كشف التسلل لكنه أظهر الكثير من العيوب مثل الحاجة إلى إعداد مجموعة من البيانات، الوقت اللازم للتدريب، مشكلة الانتشار، صعوبة معرفة الأشكال الجديدة من الهجمات الالكترونية والحاجة إلى تحديث منتظم بالإضافة إلى معدل اكتشاف منخفضة للهجمات غير المتكررة. لتعامل مع هذه العيوب اقترحت أنظمة كشف التسلل المتكيفة. في هذه الأطروحة، اقترحنا أربعة حلول لضمان تكيف أنظمة كشف التسلل من الجيل الثاني. الاقتراح الأول هو تطوير نظام كشف تسلل سريع جدا من حيث وقت التعلم ولديه قدرة عالية على التعميم. هذا الحل يعطينا القدرة على إعادة التعلم بسرعة كبيرة لضمان التكيف. الاقتراح الثاني هو تطوير نظام كشف تسلل ذو قدرة عالية جدا على التعميم الشيء الذي يعطيه القدرة على كشف الأشكال الجديدة من الهجمات بشكل مستقل. الاقتراح الثالث هو تطوير نظام كشف تسلل مبنى على أساس التعلم المستمر الشيء الذي يضمن تكيف نظامنا مع البيئة الهدف. الاقتراح الرابع هو أيضا تطوير نظام كشف تسلل ذو قدرة عالية جدا على التعميم لكن بطريقة تختلف عن الاقتراح الثاني. هذا وقد أظهرت أنظمة كشف التسلل المختلفة التي

اقترحنها قدرة عالية في الأداء، حيث أظهرت نتائج جيدة مقارنتا مع الكثير من البحوث المنشورة من قبل الباحثين في مختلف المجلات العلمية الكبرى.

مفاتح البحث: نظام كشف التسلل، IDS، نظام كشف التسلل المتكيف، استخراج البيانات، نظام كشف التسلل الهجين، نظام كشف التسلل الهرمي.

Abstract

To ensure the implementation of the security policy different tools have been developed, from these various tools of computer security we found the intrusion detection systems (IDS). An IDS is any tool, method and resources that help us to predict or identify any unauthorized network activity. The evolution of intrusion detection systems has passed through two generations. The first generation is the ad-hoc generation. It showed many limits with the large volume of traffic of actual networks. The second generation has been proposed to deal with the problems of the first generation of intrusion detection systems such as manual design based on the knowledge of human experts, the close dependence to the target environment, the difficulty of evaluation and the limit of performance. This second generation is based on data mining technique, it gives many advantages like the ability to analyze large volumes of data, independent from the human experts, autonomy, speed and optimization of performance. Despite all these advantages the second generation of intrusion detection systems showed some disadvantage like the need to prepare a set of training data, the time required for training, the deployment problem, the difficulty to know the new forms of attack, the need of regular updating, the low detection rate for non-frequent attacks. To deal with these limits the adaptive intrusion detection systems have been proposed. In this thesis, we have proposed four solutions to ensure the adaptation of the intrusion detection system of the second generation. The first proposal is to develop an intrusion detection system very fast in terms of training time and has a good ability of generalization. This solution gives us the ability to re-train our model very quickly to assure the adaptation. The second proposal is to develop an intrusion detection system with a very high generalization ability which gives us the ability to detect new form of attack. The third proposal is to develop an intrusion detection system based on continuous training method which ensured the adaptation of our system with the target

environment. The fourth proposal is to develop an intrusion detection system with very high generalization ability but with a model different from the model of the second proposal. The different proposed intrusion detection system models have shown a high performance compared to many research published by various researchers in different journals.

Key words: intrusion detection system, IDS, adaptive intrusion detection system, data mining, hierarchical intrusion detection system, hybrid intrusion detection system.

Résumé

Afin d'assurer la mise en œuvre de la politique de sécurité, différents outils ont été développés, parmi ces outils on trouve les systèmes de détection d'intrusion (IDS). Un IDS représente tout outil, méthode et ressource qui nous aident à prévoir ou à identifier toute activité non autorisée dans un réseau. L'évolution des systèmes de détection d'intrusion est passée par deux générations. La première génération est la génération ad hoc, elle a montré beaucoup de limites par rapport au grand volume du trafic réseau actuel. La deuxième génération a été proposée afin de traiter les problèmes de la première génération des systèmes de détection d'intrusion comme la conception manuelle basée sur les connaissances des experts humains, l'étroite dépendance de l'environnement cible, la difficulté liée à l'évaluation et les limites de performance. Cette deuxième génération est basée sur les techniques de data mining et elle nous offre beaucoup d'avantage comme la capacité d'analyser un large volume de données, l'indépendance des experts du domaine, l'autonomie, la rapidité et l'optimisation de la performance. Malgré tous ces avantages, la deuxième génération des systèmes de détection d'intrusion a montré beaucoup d'inconvénients comme la nécessité de préparer un ensemble de données d'apprentissage, le temps nécessaire pour l'apprentissage, le problème de déploiement, la difficulté de connaitre les nouvelles formes d'attaques, la nécessité d'une mise à jour régulière et le faible taux de détection pour les attaques non fréquentes. Afin de traiter ces limites, les systèmes de détection d'intrusion adaptatifs ont été proposés. Dans cette thèse, nous avons proposé quatre solutions pour assurer l'adaptation des systèmes de détection d'intrusion de la deuxième génération. La première proposition consiste à développer un système de détection d'intrusion très rapide en termes de temps d'apprentissage et qui possède une grande capacité de généralisation. Ce qui nous offre la capacité de refaire l'apprentissage très rapidement pour assurer l'adaptation. La deuxième proposition consiste à

développer un système de détection d'intrusion avec une très grande capacité de généralisation ce qui lui donne la capacité de détecter les nouvelles formes d'attaques d'une manière autonome. La troisième proposition consiste à développer un système de détection d'intrusion basé sur un mode d'apprentissage continu, ce qui assure l'adaptation de notre système avec l'environnement cible. La quatrième proposition consiste à créer un système de détection d'intrusion avec une très grande capacité de généralisation, mais avec un modèle différent de celui de la deuxième proposition. Les différents systèmes de détection d'intrusion que nous avons proposés ont montré de bonnes performances par rapport aux performances des travaux de recherche publiés par différents chercheurs dans divers journaux.

Mots clés : sécurité informatique, systèmes de détection d'intrusions, IDS, système de détection d'intrusion adaptatif, data mining, système de détection d'intrusion hiérarchique, système de détection d'intrusion hybride.

Dédicace

Je dédie ce modeste travail très spécialement à mes chers parents, à mes frères et ma sœur, à mon grand-père et toute ma famille.

À tous les professeurs et les collèges d'études.

Remerciement

Je tiens à remercier ma mère et mon père ainsi que toute ma famille de m'avoir encouragé durant toute la durée de mes études et de m'avoir apporté un soutien indispensable à ma réussite.

Je tiens avant tout à exprimer ma profonde gratitude, mes sincères remerciements et ma haute considération à ma directrice de thèse le professeur madame Ghoualmi-Zine Nacira, professeur à l'université Badji-Mokhtar Annaba et directrice du laboratoire LRS-Annaba pour la confiance qu'elle m'a fait en acceptant d'être ma directrice de thèse et pour son soutien et pour ses bons conseils sans lesquels il m'aurait été impossible de mener à terme ce projet de recherche.

Je souhaite exprimer ma gratitude et ma haute considération au professeur madame Natalia Djellab, professeur à l'université Badji-Mokhtar Annaba de m'avoir honoré en acceptant de présider le jury.

Mes vifs remerciements et mes hautes considérations vont également aux membres du jury les professeurs: monsieur Mohammed Benmohammed professeur à l'université de Constantine 2, monsieur Abdallah Boukerram professeur à l'université de Bejaia et madame Hayet Farida Merouani professeur à l'université de Badji-Mokhtar Annaba, pour l'honneur qu'il mon fait en acceptant d'examiner mon travail.

Je remercie aussi tous ceux qui, de près ou de loin, ont contribué à la réussite de cette démarche de recherche.

Liste des figures

Liste des tableaux

Table des matières

Introduction Générale

I. Le cadre scientifique

Actuellement, les outils de piratage et des attaques informatiques sont disponibles aux experts comme aux amateurs avec quelques dollars. Ces outils varient en fonction de leurs dangerosités et leurs performances, certains d'entre eux peuvent contourner les mécanismes de sécurité les plus sophistiqués. Une attaque réussie peut engendrer de très graves pertes, on parle aujourd'hui de plusieurs milliards de dollars de perte, des pays paralysés, des projets stratégiques sabotés, des programmes présidentiels divulgués tout ça à cause d'une attaque informatique qui varie dans le but, l'ampleur, et la dangerosité. La sécurité informatique est devenue une obligation pour toute organisation, pour faire face aux attaques afin de minimiser les risques. La sécurité informatique est la protection de l'information et des systèmes d'information contre les accès, l'utilisation, la divulgation, la perturbation, la modification ou la destruction afin de garantir la confidentialité, l'intégrité et la disponibilité.

Les systèmes et réseaux informatiques contiennent diverses formes de vulnérabilité. Pour faire face à ces problèmes de sécurité informatique, différents mécanismes ont été mis en place pour prévenir toute sorte d'attaques comme les pare-feux, l'authentification, les proxys… etc. Malheureusement, ces mécanismes ont des limites où certains types d'attaques peuvent les contourner pour nuire la confidentialité, l'intégrité ou la disponibilité. Pour faire face à ce problème, un nouveau concept qui s'appelle système de détection d'intrusion a été introduit comme une seconde ligne de défense afin de renforcer la sécurité des systèmes informatiques. Ce concept a été introduit par James Anderson en 1980 (Anderson, 1980). On peut définir un système de détection d'intrusion (IDS) comme tout outil, méthode et ressource qui nous aident à prévoir ou à identifier toute activité non autorisée dans un réseau. Les systèmes de détection d'intrusion se basent généralement sur deux approches : comportementales et par scénario. Les techniques de détection d'intrusion basée sur l'approche comportement supposent que l'intrusion peut être détectée par l'observation de la déviation du comportement par rapport au comportement normal ou prévu du système ou des utilisateurs. Par contre, l'approche par scénario se base sur les connaissances accumulées sur des attaques spécifiques et les vulnérabilités du système.

II. La problématique

Après la publication du premier modèle de détection d'intrusion par Denning (Denning, 1987) plusieurs travaux ont été faits pour créer un système de détection d'intrusion performant et très précis. La conception de cette première génération des systèmes de détection d'intrusion a été basée sur les connaissances des experts de sécurité où les méthodes statistiques et les approches de l'intelligence artificielle ont été utilisées pour créer les noyaux (moteurs) des modèles de détection d'intrusion. Face à des problèmes tels que le grand volume du trafic réseau, la distribution des données très déséquilibrée, la difficulté de prendre une décision entre le comportement normal et anormal, et l'exigence d'une adaptation permanente pour des environnements en constante évolution les techniques de l'intelligence artificielle ont montré beaucoup de limites. Pour faire face à ces nouveaux défis, les techniques de data mining sont utilisés. Grâce à ces techniques de data mining des modèles de détection d'intrusion plus rapides et plus précis ont été développés. Ces modèles représentent la deuxième génération des systèmes de détection d'intrusion.

La deuxième génération des systèmes de détection d'intrusion a montré certaines caractéristiques adaptatives comme l'adaptation aux grands volumes de données, la rapidité de traitement, l'adaptions aux réseaux à grande échelle… etc. Malgré tous ces avantages, les systèmes de détection d'intrusion de la deuxième génération qui représentent les IDSs basés sur les techniques de data mining souffrent de certain nombre de limites qui les empêches d'être très adaptatifs et autonomes. Ces limites peuvent être résumées par la nécessité d'une mise à jour régulière qui est en étroite relation avec les nouvelles formes d'attaques détectées, ainsi que le problème de déploiement qui est généralement causé par le temps nécessaire pour faire l'apprentissage. Un système de détection d'intrusion adaptatif doit traiter les faiblesses et les limites des modèles de détection d'intrusion de la deuxième génération.

Pour traiter ces limites diverses solutions ont été proposées. Ces solutions ont pour but de créer des systèmes de détection d'intrusion adaptatifs avec le sens intégral du mot ou très proche d'un système adaptatif.

III. Le but du travail

L'objectif de notre travail est de créer des systèmes de détection d'intrusion capables de remédier les limites de la deuxième génération des systèmes de détection d'intrusion. Pour atteindre notre objectif, nous avons proposé certaines solutions pour traiter ces limites. Les solutions que nous avons proposées peuvent être résumées par les trois points suivants :

- La création d'un système de détection d'intrusion très rapide en termes du temps d'apprentissage.
- La création d'un système de détection d'intrusion qui possède une grande capacité de généralisation.
- La création d'un système de détection d'intrusion avec un mode d'apprentissage continu.

Ces solutions peuvent être utilisées seules ou combinées avec d'autres solutions en même temps afin d'obtenir le système de détection d'intrusion le plus performant et le plus adaptatif. Dans l'implémentation de la première solution, la création d'un modèle distribué est très recommandé afin d'optimiser le temps d'apprentissage.

IV. Travail réalisé

Quatre systèmes de détection d'intrusion ont été proposés. Le premier modèle représente l'implémentation de notre première solution, qui consiste à créer un système de détection d'intrusion très rapide en termes d'apprentissage. Le second modèle représente l'implémentation de notre deuxième solution qui consiste à créer un système de détection d'intrusion avec une grande capacité de généralisation. Le troisième modèle représente l'implémentation de notre troisième solution qui consiste à créer un système de détection d'intrusion avec un mode d'apprentissage continu. Le quatrième modèle représente l'implémentation de notre deuxième solution qui consiste à créer un système de détection d'intrusion avec une grande capacité de généralisation, mais avec une architecture différente du deuxième modèle.

V. La structure de la thèse

Cette thèse est divisée en deux parties la première partie est un état de l'art concernant les outils de sécurité informatique où nous nous concentrons sur les systèmes de détection d'intrusion avec ces différentes générations, la deuxième partie représente nos contributions pour les systèmes de détection d'intrusion adaptatifs.

Partie I: état de l'art

Cette première partie présente les problèmes actuels de sécurité et les différentes techniques utilisées pour les traiter. Parmi les techniques utilisées pour traiter les problèmes de sécurité informatique nous nous concentrons sur les systèmes de détection d'intrusion (IDS) qui représentent le sujet de notre thèse. Cette partie est composée de quatre chapitres. Le premier chapitre est une introduction à la sécurité informatique. Le deuxième chapitre représente une introduction aux systèmes de détection d'intrusion. Le troisième chapitre présente les différentes techniques utilisées dans la détection d'intrusion. Le quatrième chapitre présente les systèmes de détection d'intrusion adaptatifs.

Chapitre 1 : la sécurité informatique

Ce premier chapitre représente une brève introduction au demain de la sécurité des systèmes informatique, où nous présentons les différents types de vulnérabilité et les différentes formes d'attaques qui exploitent ces vulnérabilités. À la fin de ce chapitre, nous présentons les différentes techniques qui nous permettent de faire face aux problèmes de sécurité et de mettre en œuvre une bonne politique de sécurité.

Chapitre 2 : les systèmes de détection d'intrusion

Ce deuxième chapitre représente une introduction aux systèmes de détection d'intrusion, où nous présentons le concept d'audit de sécurité, puis nous détaillons les systèmes de détection d'intrusion et leurs structures générales. À la fin de ce chapitre, nous présentons les différentes mesures d'efficacité des IDSs.

Chapitre 3 : les techniques de détection d'intrusion

Ce chapitre représente un état de l'art concernant les noyaux des systèmes de détection d'intrusion. Dans ce chapitre, nous abordons les différents techniques et méthodes de détection d'intrusion, où on commence par la première génération des systèmes de détection d'intrusions. Après avoir analysé les faiblesses de cette première génération, nous détaillons les techniques de data mining qui représente la deuxième génération des systèmes de détection d'intrusions.

Chapitre 4 : les systèmes de détection d'intrusion adaptatifs

Dans ce chapitre nous présentons les limites de la deuxième génération des systèmes de détection d'intrusion et l'importance des systèmes de détection d'intrusion adaptative qui traite ces limites. Puis nous analysons certains systèmes de détections d'intrusions adaptatifs existants.

Partie II: contributions

Cette deuxième partie présente nos différentes contributions pour concevoir un système de détection d'intrusion adaptatif. Nos apports se basent sur quatre propositions. La première proposition consiste à créer un système de détection d'intrusion très rapide en termes de temps d'apprentissage avec une grande capacité de généralisation. La deuxième proposition est la création d'un système de détection d'intrusion avec une très haute capacité de généralisation. La troisième proposition représente la création d'un système de détection d'intrusion avec un mode d'apprentissage continu. La quatrième proposition a le même objectif que la deuxième proposition, mais avec un modèle différent. À la fin de cette partie, nous positions nos travaux par rapport aux modèles existants, où nous présentons les critiques de nos différentes propositions ainsi qu'une étude comparative avec les autres systèmes de détection d'intrusion adaptatifs ainsi que certains systèmes de détection d'intrusion récents.

Chapitre 5 : proposition 1 un système de détection d'intrusion très rapide en termes du temps d'apprentissage

Le but de cette première proposition est de créer un système de détection d'intrusion hybride hiérarchique très rapide et de haute performance appelé NFHP-IDS (New Fast Performed Hierarchical Intrusion Detection System) qui possède les caractéristiques suivantes: un temps d'apprentissage de très courte durée, détecte les attaques de faible fréquence, donne un taux élevé pour la détection des attaques fréquentes, donne un faible taux de fausses alarmes. NFHP-IDS contient deux niveaux, le premier niveau comprend les quatre classificateurs rapides suivants: "Random Forest", "Simple Cart", "Best first decision tree", "Naive Bayes". Ces classificateurs sont utilisés pour leurs excellentes performances pour la classification de respectivement comportement normal et DOS, Probe, R2L et U2R. Seules cinq sorties du premier niveau sont

sélectionnées et utilisées comme des entrées du second niveau qui contient le Naïve Bayes comme classificateur final.

Chapitre 6 : proposition 2 un système de détection d'intrusion avec une très grande capacité de généralisation

Le but de cette proposition est de créer un nouveau système de détection d'intrusion hiérarchique basé sur un arbre binaire de différents types de classificateurs. Le modèle de détection d'intrusion proposé doit posséder les caractéristiques suivantes: combiner un taux de détection élevé et un faible taux de fausses alarmes, classer toute connexion dans l'une des cinq catégories de connexion réseau : les attaques d'exploration (Probe), les attaques de déni de service (DoS), les attaques Utilisateur-à-root (U2R), les attaques distance à local (R2L) et le comportement normal (normal). Pour construire l'arbre binaire, nous regroupons les différentes catégories de connexions réseau hiérarchiquement en fonction de la proportion de faux positif et de faux négatif générer entre chaque deux catégories. Le modèle construit est un arbre binaire avec quatre niveaux. Au début, nous utilisons "Repeated Incremental Pruning to Produce Error Reduction" pour classer les connexions réseau en deux catégories : les attaques DOS et G2 qui regroupe Probe, R2L, U2R et Normal. Puis, dans le deuxième niveau, nous utilisons "Naïve Bayes Multinomial" pour classer les connexions réseau de G2 en Probe et G3 qui regroupe R2L, U2R et Normal. Après, nous utilisons "Ripple-down rule learner" pour classer les connexions réseau de G3 en R2L et G4 qui regroupe U2R et Normal. Enfin, nous utilisons "Random Tree" pour classer les connexions réseau de G4 en U2R et Normal.

Chapitre 7 : proposition 3 un système de détection d'intrusion avec un mode d'apprentissage continu

Le but de cette proposition est de créer un nouveau système de détection d'intrusion hiérarchique basé sur un mode d'apprentissage continu. L'idée de cette proposition est d'utiliser deux classificateurs différents itérativement, où chaque itération représente un niveau dans le modèle construit. Pour assurer l'adaptation de notre modèle, nous ajoutons un nouveau niveau chaque fois que la somme des nouvelles attaques et le reste de l'ensemble des données d'apprentissage atteint le seuil. Pour construire notre

modèle, nous avons utilisé "Fuzzy Unordered Rule Induction Algorithm" et "Random Forests" comme classificateurs.

Chapitre 8 : proposition 4 un système de détection d'intrusion avec une très grande capacité de généralisation

Le but de cette proposition est de créer un système de détection d'intrusion hybride, hiérarchique, et de très hautes performances. Le modèle proposé comprend deux niveaux. Le premier niveau contient les cinq classificateurs suivants: "Repeated Incremental Pruning to Produce Error Reduction", "Multilayer Perceptrons", "Self-Organizing Feature Map Network", Arbre de décision C4.5 et Naïve Bayes. Ces classificateurs sont utilisés pour leurs taux élevé de correcte classification de respectivement DOS, comportement normal, Probe, R2L et U2R. Seules cinq prédictions du premier niveau sont sélectionnées et utilisées comme entrées du second niveau qui contient le réseau de neurones RBF comme classificateur final.

Chapitre 9 : positionnement de nos travaux par rapport aux autres systèmes de détection d'intrusion adaptatifs

Dans ce chapitre nous présentons les apports de nos différentes propositions pour remédier les limites des systèmes de détection d'intrusion de la deuxième génération. Pour chaque une des quatre propositions, nous présentons la nouveauté du modèle ainsi que ces avantages et ces inconvénients. La dernière section de ce chapitre représente une comparaison entre la performance de nos modèles et les autres systèmes de détection d'intrusion adaptatifs existants ainsi que certains systèmes de détection d'intrusion récents.

PARTIE I

État de l'art

Chapitre 1

La sécurité informatique

Ce premier chapitre représente une brève introduction au domaine de la sécurité des systèmes informatiques, où nous allons présenter les différents types de vulnérabilité, puis nous détaillons les différentes formes d'attaques qui exploitent ces vulnérabilités. Après avoir expliqué les ennuis de la sécurité informatique, nous détaillons les principaux concepts de sécurité, la gestion des risques et la politique de sécurité. À la fin de ce chapitre, nous présentons les différentes techniques qui nous permettent de faire face aux problèmes de sécurité informatique et de mettre en œuvre une bonne politique de sécurité.

1.1. Introduction

Dans le monde moderne, internet est devenu un outil primordial qu'on utilise pour effectuer diverses activités comme le travail, l'étude, l'achat en ligne, la communication… etc. Cette révolution technologique a été accompagnée par une augmentation phénoménale du nombre d'utilisateurs d'internet, on compte aujourd'hui plus de 2,5 milliards d'utilisateurs. Les informations qui transitent par internet peuvent être importantes, critiques, secrètes et confidentielles alors que les concepteurs d'internet n'ont pas prévu la sécurité de ces informations, leur initial but était d'interconnecter les différents réseaux informatiques. Comme tout système conçu par un être humain internet contient des faiblesses. Certains utilisateurs mal intentionnés peuvent exploiter les vulnérabilités d'internet pour essayer d'accéder à des informations sensibles dans le but de les lire, les modifier ou les détruire. Dès lors que ce réseau est apparu comme cible potentielle d'attaques, leur sécurité est devenue un enjeu incontournable.

Vu que les différentes infrastructures des divers secteurs sociaux, économiques, militaires, gouvernementales sont connectées à internet, une attaque informatique est devenue une arme très dangereuse et très destructive. Grâce à une attaque informatique on peut paralyser tout un pays comme l'attaque contre l'Estonie en avril 2007, on peut aussi retarder ou saboter un projet stratégique comme l'attaque contre le centre nucléaire d'Iran en septembre 2010, sans oublie l'espionnage industriel où la chine occupe la première place mondiale. L'enjeu des attaques informatiques ou les cyberattaques est sorti du cadre d'ambition et de loisir, il est devenu un projet militaire stratégique, on parle aujourd'hui du cyber guerre.

1.2. Les causes de l'insécurité

Au sein d'un réseau informatique, on distingue généralement cinq types de faille qui peuvent causer l'état d'insécurité (Ebel et al, 2009) :

- ***Les failles physiques :*** généralement dans une entreprise ou une administration la sécurité d'accès aux matériels informatiques n'a pas une grande importance. Il suffit de trouver des prétextes comme faire des tests, de la maintenance ou le nettoyage pour accéder. L'exploitation de cet accès physique pour voler un mot de passe, effacer des données, usurper l'identité d'un autre ou injecter des programmes malveillants peut causer des dégâts catastrophiques pour une entreprise.
- ***Les failles réseaux :*** les réseaux informatiques sont fondés sur des normes et des standards bien réfléchis où plusieurs organismes collaborent pour les perfectionner. Malgré tous les efforts faits, il existe certaines failles ou détournements de fonctionnement des standards exploitables. Le problème avec les failles réseau c'est la complexité de leurs corrections qui varie d'après la taille du réseau. À titre d'exemple, corriger les failles réseau d'internet est utopique, c'est la raison pour laquelle on se contente de faire des améliorations comme le passage vers IPV6 ou IPSec.
- ***Les failles systèmes :*** les systèmes d'exploitation sont de plus en plus sophistiqués, ils intègrent différent mécanisme de sécurité comme les mots de passe, les logs, séparation des privilèges…etc. La complexité, la mauvaise configuration ainsi que les faiblesses de certains mécanismes des systèmes d'exploitation représentent un danger pour les utilisateurs. Par exemple la complexité d'un mécanisme de sécurité pousse les utilisateurs à le désactiver, de plus la mauvaise configuration peut engendrer l'arrêt ou la saturation du système.
- ***Les failles applicatives :*** les failles applicatives sont des failles très connues et très répondues. Ils peuvent être causés par la mauvaise conception, non-traitement des exceptions, faille dans le langage de programmation. Ces failles peuvent engendrer beaucoup de problèmes qui influencent le fonctionnement du système.
- ***Les failles Web :*** le monde du web représente la combinaison de différent protocole, réseaux, système et application. Les failles web peuvent être causées par l'une des failles précédemment citées ou par des failles qui résident au niveau des protocoles et des standards du fonctionnement du web.

1.3. Les différents types d'attaque informatique

Une attaque informatique est littérairement définie par toute tentative de détruire, exposer, modifier, désactiver, voler ou obtenir un accès non autorisé ou toute utilisation non autorisée d'une information, logiciels, physique comme un serveur, services, des personnes et de leurs qualifications, et les biens incorporels (ISO/IEC 27000, 2009).

Pour l'aspect technique, on peut définir une attaque par l'exploitation de l'une des failles précédemment citées pour des fins illégales. Il existe cinq formes d'attaque que nous détaillons comme suit (Cole et al, 2005) :

- ***L'attaque passive :*** les attaques passives représentent tout acte qui nous permet de faire l'analyse et le décryptage du trafic, la surveillance des communications, et la capture des informations d'authentification. Les attaques passives peuvent entraîner la divulgation des informations ou des données à un attaquant sans que la victime soit consciente. L'interception du mot de passe, numéros de carte de crédit, des emails représente tous des attaques passives.
- ***L'attaque active :*** les attaques actives comprennent toute tentative a pour but de contourner ou arrêter les fonctions de protection, introduire un code malveillant et de voler ou modifier des informations. Les attaques actives peuvent entraîner la divulgation et la diffusion des données, un déni de service, ou la modification des données.
- ***L'attaque de proximité ou externe :*** les attaques de proximité (externes) représentent l'utilisation de la proximité physique du réseau ou du système qui a été obtenue grâce à l'entrée clandestine ou un accès ouvert afin de modifier, collecter ou refuser l'accès à l'information.
- ***L'attaque interne :*** les attaques internes peuvent être intentionnelles ou non intentionnelles. Les attaques intentionnelles représentent les tentatives d'espionner, de voler ou d'endommager des informations, utiliser l'information de manière frauduleuse, ou interdire l'accès à d'autres utilisateurs autorisés. Les attaques non intentionnelles représentent le résultat d'une mauvaise manipulation, la négligence ou le manque de connaissances.
- ***L'attaque de distribution :*** les attaques de distribution représentent toute modification malveillante du matériel ou du logiciel en usine ou lors de la distribution. Ces attaques consistent à introduire un code malveillant dans un produit

comme un port dérobé pour obtenir un accès non autorisé à des informations ou une fonction système.

1.4. Exemples des attaques informatiques

Il existe plusieurs types des attaques très connues dans le monde de l'informatique, nous détaillons ici trois exemples d'attaques informatiques très réputées par leurs dangerosités et les dégâts qui peuvent causer.

L'attaque man-in-the-middle : l'attaquant s'introduit entre deux systèmes sans que l'un d'entre eux aperçoive l'existence d'un troisième système qui fait passer les échanges réseau. Pour réussir une telle attaque, il faut que la machine de l'attaquant soit physiquement entre les deux machines victimes ou que l'attaquant arrive à modifier le routage réseau afin que sa machine devienne un des points de passage (Liorens et al, 2006). Le schéma suivant illustre le fonctionnement de l'attaque man-in-the-middle.

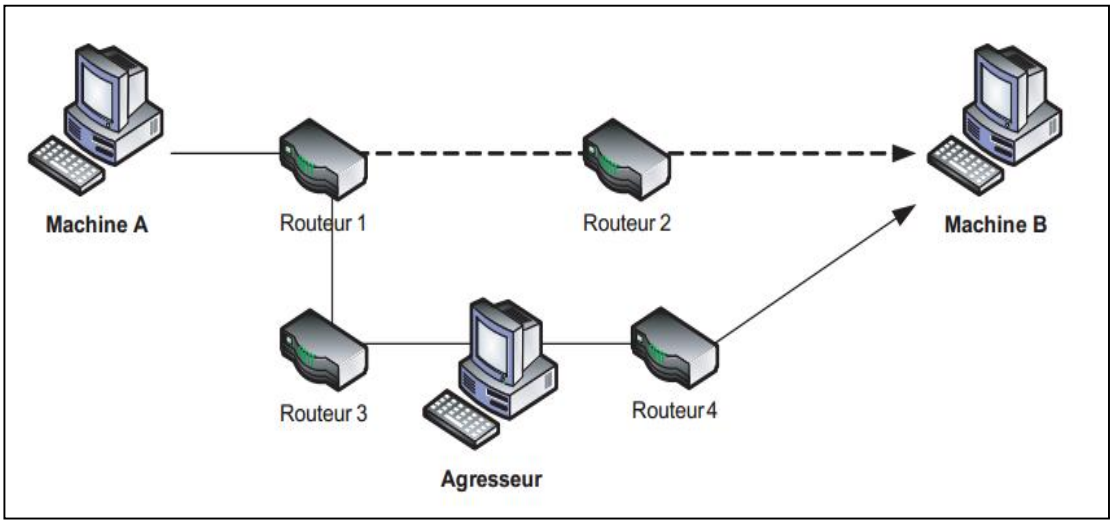

Figure 1 L'attaque man-in-the-middle (Liorens et al, 2006)

L'attaque de déni de service distribué (DDoS) : elle représente la version distribuée de l'attaque de déni de service. Le but de cette variante de l'attaque DoS est que la victime n'arrive pas à isoler les attaquants vu le nombre important des machines utilisées pour réaliser cette attaque. Pour réaliser cette attaque, il faut premièrement pénétrer par diverses méthodes des systèmes dits "handlers" et agents. Où l'attaquant contrôle un ensemble de systèmes "handlers" qui contrôlent eux-mêmes un ensemble de systèmes agents. Le hacker lance l'attaque en ordonnant les systèmes "handlers", qui eux-mêmes ordonnent les agents (Liorens et al, 2006). Le schéma suivant illustre le fonctionnement de l'attaque DDoS.

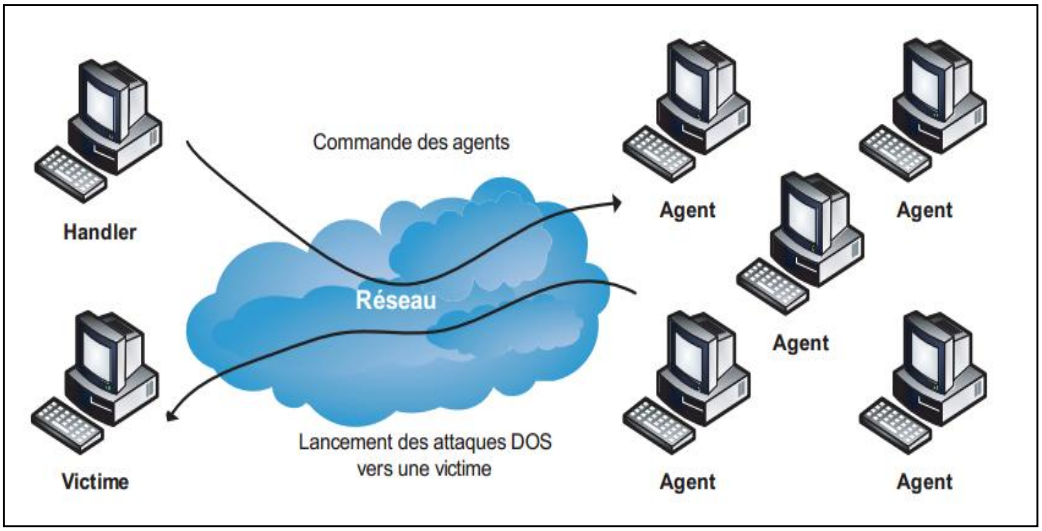

Figure 2 L'attaque DDoS (Liorens et al, 2006)

Attaque par virus : un virus informatique est tout programme capable de se reproduire par lui-même. Un virus informatique peut prendre la forme d'une routine ou d'un programme une fois activé il utilise tous les moyens pour empoisonner la vie de l'utilisateur. Les virus informatiques représentent le type d'attaque le plus fréquent. Le cycle de vie d'un virus commence par la création, puis la reproduction, ensuite l'activation, ensuite le découvrir et en fin le détruire. Il existe plusieurs types de virus qu'on peut les résumer par (Liorens et al, 2006) :

- virus de secteur d'amorçage.
- virus d'infection des fichiers (parasites).
- virus non-résidents mémoire.
- virus résidents mémoire.
- virus multiformes.
- virus furtifs.
- virus polymorphes (mutants).
- virus réseau et vers (worms).
- virus flibustiers (bounty hunters).
- bombes logiques.
- chevaux de Troie.

1.5. L'impact des attaques informatiques

Au début de l'histoire des réseaux informatiques les attaques informatiques ont été munies par des experts, leur nombre a été limité voir très limité. Maintenant les outils de piratage et des attaques informatiques sont disponibles aux amateurs avec quelques dollars. De plus, les pertes qui peuvent être engendrées par une attaque informatique sont de plus en plus graves. On parle aujourd'hui des milliards de dollars de perte, des pays paralysés, des projets stratégiques sabotés, des programmes présidentiels divulgués tout ça à cause des attaques informatiques qui varient dans le but, l'ampleur et la dangerosité.

La figure suivante montre les pertes causées par des attaques informatiques dans une étude munie par le CSI (Computer Security Institue) sur 194 organisations. Le montant total de perte durant l'année 2007 est de 66.930.950 dollars American. Les fraudes financières occupent la première place avec 21.124.750 $USA, la deuxième place est occupée par les virus avec 8.391.800 $USA (Richardson, 2007).

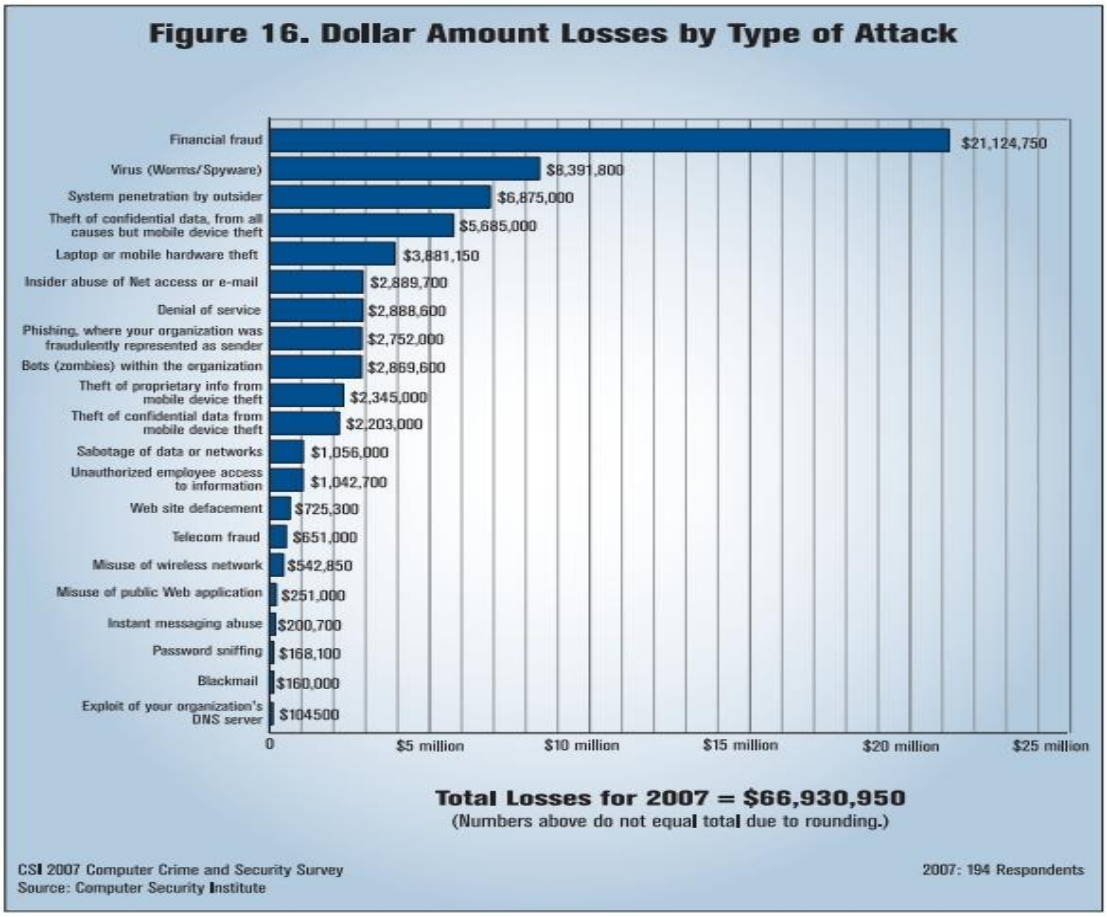

Figure 3 Le rapport des pertes causées par des attaques informatiques sur 194 organisations durant l'année 2007 (Richardson, 2007)

1.6. La sécurité informatique

La sécurité informatique est la protection de l'information et des systèmes d'information contre l'accès, l'utilisation, la divulgation, la perturbation, la modification ou la destruction afin de garantir la confidentialité, l'intégrité et la disponibilité (Johnson et al, 2011).

1.6.1. Les principes de la sécurité informatique

La sécurité des systèmes informatique repose sur trois principes clés: la confidentialité, l'intégrité et la disponibilité. Vu le contexte de l'application de ces principes, certains d'entre eux peuvent avoir plus d'importance que d'autres. Par exemple, la confidentialité est la plus importante dans le cadre d'une transmission des messages secrets entre deux agences de sécurité nationale ou internationale, si quelqu'un arrive à décrypter le message transmis la sécurité sera compromise et l'information sera divulguée. Par contre la disponibilité est la plus importante pour les sites de e-commerce, la non-disponibilité est catastrophique pour des sites comme amazone et eBay (Cole et al, 2005).

La confidentialité : La confidentialité consiste à préserver la révélation non autorisée d'information sensible. La révélation pourrait être intentionnelle comme les attaques qui visent de casser le chiffrement des données et lire les informations, ou involontaire dû au manque de vigilance ou de l'incompétence des individus qui manient les informations (Cole et al, 2005).

L'intégrité : l'intégrité consiste à garantir trois buts principaux :

- Préserver le changement des informations par les utilisateurs non autorisés
- Préserver le changement non autorisé ou involontaire d'information par les utilisateurs autorisés
- Préserver la cohérence interne et la cohérence externe
    - La cohérence interne: consiste à assurer la cohérence des données interne. Par exemple dans une organisation on assure que le nombre total des articles maintenus par cette organisation est égal à la somme des mêmes articles dans la base de données.
    - La cohérence externe: consiste à assurer que la cohérence entre les données dans la base de données et le monde réel est maintenue. Par exemple dans une entreprise on assure que le nombre des articles vendus est le même nombre dans la base de données (Cole et al, 2005).

La disponibilité : La disponibilité assure que les utilisateurs autorisés ont un accès opportun et non interrompu aux informations dans le système et le réseau (Cole et al, 2005).

1.7. La gestion des risques lie aux incidents informatiques

La gestion des risques comprend trois processus: l'estimation des risques, la réduction des risques, et l'évaluation et l'estimation (Cole et al, 2005).

Le processus de l'estimation des risques comprend :

- L'identification et évaluation des risques
- L'identification et évaluation de l'impact des risques
- La recommandation des mesures pour la réduction des risques.

On peut évaluer quantitativement les risques par l'équation suivante : $Risque = \frac{menace \times vulnérabilité}{contre-mesure}$. La menace représente le type d'action susceptible de nuire dans l'absolu, tandis que la vulnérabilité représente le niveau d'exposition face à la menace dans un contexte particulier, la contre-mesure est l'ensemble des actions mises en œuvre pour prévenir les menaces.

Le processus de la réduction des risques comprend les taches suivantes:

- Prioriser les mesures de réduction des risques recommandés par le processus de l'estimation des risques.
- Implémenter les mesures de réduction des risques recommandés par le processus de l'estimation des risques.
- Maintenir les mesures de réduction des risques recommandés par le processus de l'estimation des risques.

Le processus d'évaluation et d'estimation inclut un processus d'évaluation continu. Par exemple, l'autorité approbatrice désignée des États-Unis d'Amérique (DAA) est la responsable de déterminer si le risque résiduel dans le système est acceptable ou que des mesures de contrôle et de protection supplémentaires devraient être implémentées pour accomplir l'accréditation d'un système informatique (Cole et al, 2005).

1.8. La politique de sécurité

Une politique de sécurité est comme une politique étrangère pour un pays, elle définit les buts et les objectifs. Un système informatique sans une politique de sécurité est susceptible

d'avoir un désordre de contre-mesures. Les bonnes politiques sont adaptées aux menaces. S'il n'y avait pas de menaces, il n'y aurait pas de politique.

La politique de sécurité fournit un cadre pour la sélection et la mise en œuvre des contre-mesures contre les menaces. Chaque organisation a besoin d'une politique de sécurité adaptée à son réseau informatique. La politique devrait préciser qui responsable de quoi (mise en œuvre, exécution, vérification, examen), la nature de cette politique de sécurité du réseau et pourquoi elle est de cette nature. La réponse à ces questions est très importante parce qu'une politique claire, concise, cohérente et constante est plus susceptible d'être suivie.

La politique de sécurité est de savoir comment vous déterminez quelles contre-mesures il faut utiliser. Par exemple: Avez-vous besoin d'un pare-feu? Comment devez-vous configurer votre pare-feu? Avez-vous besoin d'un jeton d'accès, ou un mot de passe est suffisant? Les utilisateurs sont autorisés à accéder à la vidéo en streaming à partir de leurs navigateurs Web? S'il n'y a pas de politique, il n'y aura pas de base pour répondre systématiquement à ces questions. Malheureusement, la plupart des organisations n'ont pas une politique de sécurité réseau. Ou bien ils le font, mais personne ne la suit (Schneier, 2004).

Les politiques de sécurité sont différentes d'une organisation à l'autre, mais dans le cas le plus simple une politique de sécurité devrait comprendre les éléments suivants (Trend Micro, 2011) :

- Une explication claire et simple du but de la politique de la sécurité et les objectifs et l'importance de cette stratégie de sécurité pour l'entreprise.
- Une déclaration de soutien pour cette politique de sécurité de la part des cadres supérieurs de l'organisation, ce qui prouve leurs engagements.
- Offrir des formations afin d'aider les employés à comprendre la sécurité de l'information et les dégâts qui peuvent être causés si on enfreint la politique de sécurité.
- Une explication simple et claire des normes minimales de sécurité en mettant l'accent sur les procédures à suivre dans des domaines qui relève une importance particulière pour l'entreprise. Par exemple, toute politique de sécurité doit préciser

les précautions élémentaires concernant les virus informatiques, des consignes pour la mise en place des mots de passe…etc.

- Définir les rôles et les responsabilités de la sécurité des informations au sein de l'organisation.
- Exiger des rapports, réponses, résolutions pour n'importe quel type d'incident de sécurité au sien de l'organisation.
- La nécessité d'un plan de continuité des activités, ce qui explique comment l'entreprise va continuer à fonctionner en cas d'une défaillance catastrophique, comme un incendie ou une inondation.
- Avoir un support bien documenté pour le référencement au sien de l'organisation comme la politique, les guides, les procédures et les standards de sécurité.
    - **Les politiques:** c'est les documents non techniques qui décrivent d'une manière formelle les principes ou les règles auxquelles se conforment les personnes qui reçoivent un droit d'accès au capital technologique et informatif de l'organisation.
    - **Les guides:** c'est les documents qui complètent les documents de la politique où ils détaillent comment implémenter cette politique de sécurité.
    - **Les standards:** c'est les documents de standardisation des normes et des méthodes provenant d'organismes internationaux tels que l'ISO (International Standardization Organization), l'IETF (Internet Engineering Task Force), l'IEEE (Institute of Electrical and Electronics Engineers)… etc.
    - **Les procédures:** c'est les documents techniques qui décrivent d'une manière claire et précise les étapes à suivre pour atteindre un objectif de sécurité donné (Liorens et al, 2006).

Exemple: si nous voulons bien détailler la politique de l'internet, nous pouvons inclure:

- L'utilisation d'internet au sien de l'organisation et les menaces relatives.
- Les services d'internet qu'on peut utiliser sans limites.
- Qui autorise les connexions internet.
- Les normes, les guides et les pratiques à suivre.

- Qui est le contact unique chargé de la politique de sécurité informatique (même si tout le monde est responsable de la mise en œuvre de cette politique).

1.9. Les différentes techniques de sécurité

La mise en œuvre d'une politique de sécurité consiste à déployer les différents moyens et dispositifs visant la sécurisation du système d'information ainsi que l'application des règles définit dans la politique de sécurité adoptée. Ce qui signifie, faire le bon choix de l'ensemble des mécanismes et des techniques les plus simples possible permettant de protéger les ressources d'une manière très efficace avec un faible coût. Il existe différentes techniques utilisées contre les attaques informatiques, ces techniques sont classées en cinq catégories qui sont: la protection des accès réseau, la protection des accès distants, la sécurité des équipements réseau, la protection des systèmes et des applications réseau et la protection de la gestion du réseau (Liorens et al, 2006).

1.9.1. La protection des accès réseau

La protection des accès réseau consiste à maîtriser les flux réseau à l'aide des pare-feux et assurer un niveau de confidentialité des données grâce aux protocoles de sécurité tel que l'IPSec.

1.9.1.1. Contrôler les connexions réseau

Le contrôle du trafic réseau consiste à ne laisser passer que les connexions autorisées. L'objectif de ce contrôle est de créer un périmètre de sécurité, limiter le nombre de points d'accès pour rendre la gestion de la sécurité plus facile et disposer de trace des systèmes en cas d'incident de sécurité. Il existe plusieurs techniques de contrôle et de filtrage de connexion comme:

- ***Le pare-feu :*** c'est le système qui permet de mettre en œuvre la politique du filtrage au sien de l'organisation. Il existe plusieurs principes de filtrage pour les pare-feux: le filtrage des paquets qui filtre les paquets au niveau réseau (IP, etc.), le filtrage à mémoire qui filtre les paquets de manière dynamique, la passerelle de niveau transport qui filtre les paquets en gérant le concept de session et la passerelle de niveau applicatif qui filtre jusqu'aux protocoles du niveau applicatif.
- ***Contrôle de l'accès réseau :*** c'est un nouveau concept développé par Cisco, son objectif et de contrôler les accès les plus près à leurs sources où il permet de vérifier

un certain nombre de points de sécurité avant d'autoriser un système à se connecter au réseau local.

1.9.1.2. Assurer la confidentialité des connexions

Pour garantir la confidentialité des données au sien d'un réseau informatique, on doit utiliser le chiffrement. Où, on crypte les données avant de les envoyer et on les décrypte à la réception. Le schéma suivant montre où il intervient le chiffrement dans l'architecture de communication TCP/IP.

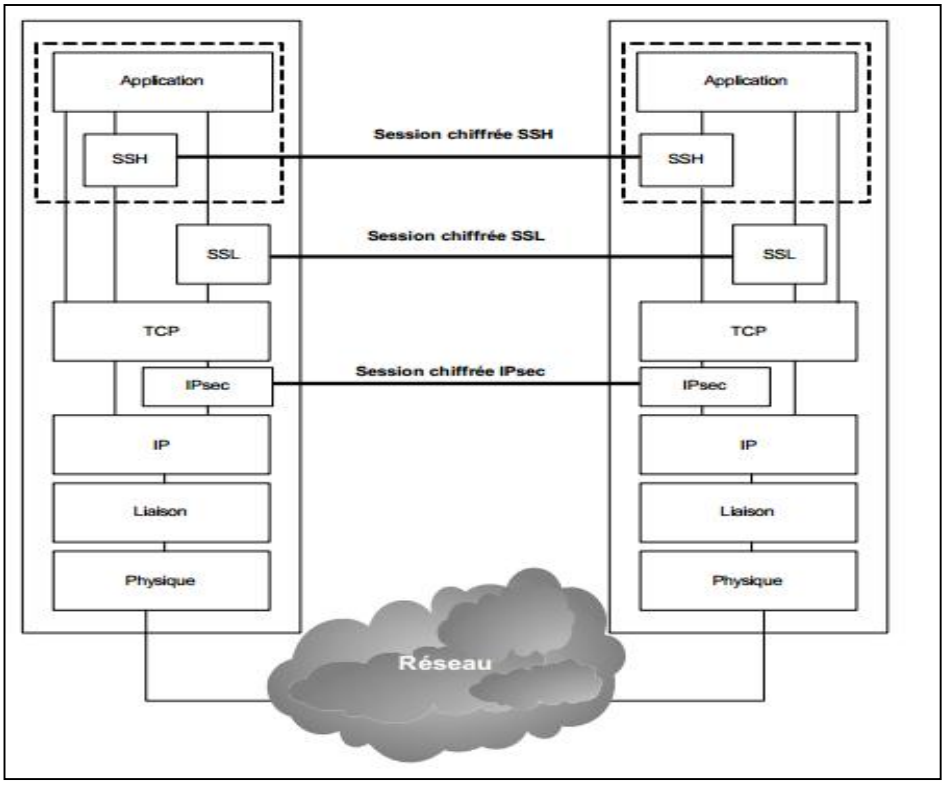

Figure 4 La représentation en couches des protocoles de sécurité (Liorens et al, 2006)

- ***Les algorithmes cryptographiques :*** l'art du chiffrement-déchiffrement des données est effectué par des algorithmes cryptographiques. Ces algorithmes reposent sur des problèmes mathématiques difficiles à résoudre. Il existe deux grandes catégories d'algorithme de cryptographie: les algorithmes cryptographiques à clé secrète ou symétrique qui se basent sur une même clé qui chiffre et déchiffre. Cette clé est partagée par les deux communicants. Les algorithmes cryptographiques à clé publique ou asymétrique qui se basent sur une clé publique de chiffrement et une clé secrète de déchiffrement. Il existe aussi les algorithmes de hachage qui nous permettent d'obtenir une signature numérique à partir des données.

- ***IPSec:*** il est créé pour faire face aux problèmes d'authentification et de confidentialité du protocole IP. IPSec opère au niveau IP et il encapsule nativement tous ces protocoles (TCP, UDP, ICMP, etc.). IPSec offre des services de contrôle d'accès, d'intégrité, d'authentification, de confidentialité de plus il fait face aux attaques de type paquets replay.
- ***SSL (Secure Sockets Layer) :*** opère au-dessus de la couche TCP et offre aux navigateurs internet la possibilité d'établir des sessions authentifiées et chiffrées. Le protocole SSL a été standardisé par le groupe de travail TLS (Transport Layer Security) formé au sein de l'IETF.
- ***SSH (Secure Shell) :*** il opère au niveau application et permet d'obtenir un interprète des commandes (Shell) à distance d'une manière sécurisé.

1.9.2. La protection des accès distants

L'accès distant au réseau d'une organisation offre plusieurs avantages, mais aussi beaucoup de possibilités de pénétration où les lacunes d'authentification des utilisateurs et les failles des protocoles sont bien exploitées. Afin d'assurer la protection des accès distants, on doit assurer l'authentification des connexions distantes, le contrôle des accès physiques à un réseau local, le contrôle des accès distants classiques, le contrôle des accès distants WI-FI.

1.9.2.1. Assurer l'authentification des connexions distantes

L'authentification garantit une protection contre toute sorte d'attaques qui visent l'usurpation d'identité d'un utilisateur légitime, tel que les attaques IP spoofing, les attaques visant à dérober les mots de passe, les attaques par cheval de Troie… etc. Il existe plusieurs solutions qui servent à offrir un service d'authentification nous détaillons ici quelques-unes.

- ***Le mot de passe :*** c'est le moyen d'authentification le plus simple et le plus utilisé. Les faiblesses lie aux mots de passe viennent généralement des protocoles d'accès qui ne les chiffrent pas lors du passage sous réseau comme le Telnet. En second lieu, on trouve les faiblesses dues au fait que les mots de passe sont souvent faciles à deviner comme la suite simple 1234, les dates de naissance, les prénoms …etc.
- ***Le Token RSA :*** Il se base sur la technologie des tickets ou des mots de passe valables pour une courte durée (quelques dizaines de seconds). Au début l'utilisateur se sert d'un authentifiant (le jeton ou carte à puce) plus un code PIN secret. L'authentifiant génère des codes d'identification aléatoires toutes les soixante

secondes. L'identification de l'utilisateur est garantie par la combinaison du code PIN, l'authentifiant et le code aléatoire.

- ***Le certificat électronique:*** Il assure l'identité électronique d'un individu ou d'un système. Il se base sur les infrastructures PKI (Public Key Infrastructure).
- ***La paire de clés PGP (Pretty Good Privacy):*** Elle offre des services de confidentialité et d'authentification pour la messagerie électronique et le stockage des données. La certification ou les niveaux de confiance définis des clés privées/publiques crées sont indépendants des organismes de standardisation. Contrairement à PKI chaque utilisateur possède une ou plusieurs paires de clés privées/publiques. Il communique les clés publiques aux personnes avec lesquelles il veut communiquer.

1.9.2.2. Assurer le contrôle des accès physiques à un réseau local

Le protocole IEEE 802.1X est un standard qui nous permet d'obtenir un mécanisme d'autorisation d'accès physique à un réseau local après authentification. Les composants qui interviennent dans un tel mécanisme sont le système à authentifier, le point d'accès au réseau local (commutateur, routeur, etc.) et le serveur d'authentification.

1.9.2.3. Assurer le contrôle des accès distants classiques

Malgré que les utilisateurs d'accès distants protègent leurs PC portables contre les pénétrations directes, les virus, le cheval de trois l'accès est rapidement usurpé. Les moyens de protection des ordinateurs portables doivent se baser sur un pare-feu local implémentant des règles très restrictives et un système antivirus régulièrement mis à jour. Ces éléments doivent être sous le contrôle de l'administrateur afin d'éviter toute erreur de configuration.

1.9.2.4. Assurer le contrôle des accès distants WI-FI

La norme 802.11 a défini le protocole WEP (Wired Equivalent Privacy) qui assure la confidentialité et l'intégrité des données. Mais la norme 802.11 possède des faiblesses de sécurité comme la petite taille de la clé "maître" utilisée pour le chiffrement des données, la petite taille et la prédictibilité du vecteur d'initialisation. Pour corriger les faiblesses de sécurité du protocole WEP, de nombreuses améliorations ont été proposées afin de renforcer la sécurité de ces accès, comme le WPA (Wi-Fi Protected Access) qui implémente des fonctionnalités comme :

- Le mécanisme de négociation d'authentification fondé sur EAP (Extensible Authentication Protocol) ou PSK (Pre-Shared Key).
- Le mécanisme de gestion et de distribution des clés TKIP (Temporal Key Integrity Protocol).
- Le mécanisme d'intégrité des trames TKIP + algorithme Michael.

1.9.3. La sécurité des équipements réseau

La sécurité d'un réseau informatique se base généralement sur la protection de ses équipements qui recouvre les trois domaines suivants:

- ***La sécurité physique***: c'est la protection physique des équipements face aux menaces physiques externes comme le feu, l'inondation, le survoltage, l'accès illégal à la salle informatique… etc.
- ***La sécurité du système d'exploitation***: c'est la protection des systèmes d'exploitation contre les faiblesses de sécurité ou les bugs.
- ***La sécurité logique***: c'est la configuration de l'équipement réseau, afin de mettre en œuvre la politique de sécurité.

La maîtrise de la sécurité de ces équipements réseau nous permet de se protéger contre les attaques suivantes :

- Les attaques par déni de service visant à exploiter des faiblesses de configuration.
- Les attaques permettant d'obtenir un accès non autorisé à un équipement réseau suite à des faiblesses de configuration.
- Les attaques exploitant un bug référencé du système d'exploitation Cisco, Microsoft, RedHat, …etc.

1.9.4. La protection des systèmes et des applications réseau

Un réseau informatique nous offre un ensemble de services sur des systèmes dédiés. Pour protéger ces systèmes ainsi que les applications qui nous offrent ces services, on doit implémenter tous les services, mais uniquement ces services (Liorens et al, 2006). Pour protéger les systèmes et les applications réseau, il faut séparer les plates-formes, sécuriser les systèmes d'exploitation, configurer les pare-feux, sécuriser le contrôle d'intégrité, maîtriser la sécurité des applications.

1.9.4.1. Séparer les plates-formes

Elle consistant à déployer des services de nature différente sur des plates-formes distinctes. Cette approche implique un surcoût de déploiement et d'administration, mais elle offre aussi une facilité de déploiement et de gestion et une grande résistance contre les attaques.

1.9.4.2. Sécuriser les systèmes d'exploitation

Pour sécuriser un système d'exploitation, on doit faire le déshabillage (strip-down) et le blindage (hardening) du système (Liorens et al, 2006).

- ***« Le déshabillage »:*** pour déshabiller un système d'exploitation, il faut désactiver tous les services réseau inutiles ou dangereux. Cette étape est très importante, car un système qui n'entend pas certaines requêtes est complètement immunisé contre les attaques ciblant ces services. Par exemple, les services tels que Berkeley (rsh, rexec, rlogin) et Telnet doivent être désactivés sur les systèmes Unix.
- ***« Le blindage » :*** il consiste à appliquer systématiquement la règle du privilège minimal. Par exemple: rétrogradation ou redéfinition des privilèges sur les processus, synchronisation de l'horloge du système sur au moins deux sources fiables, installation d'un système de vérification de l'intégrité des répertoires et des fichiers stables.

1.9.4.3. Les pare-feux

On distingue deux grandes catégories des pare-feux: ceux qui visent la protection d'une zone en coupure de ligne et ceux qui contrôlent uniquement les accès au système hôte. Le pare-feu zonal comme le pare-feu embarqué peuvent être "stateless", "stateful" ou "proxy" au niveau applicatif. Il existe plusieurs facteurs qui interviennent dans le choix du type de pare-feu, comme l'isolement topologique du serveur, la différence du type de contrôle nécessaire, une autorité différente… etc.

1.9.4.4. Sécuriser la gestion des droits d'accès

La règle d'or pour la sécurité de la gestion des droits d'accès consiste à authentifier les accès sur une base individuelle et chaque profil doit respecter la règle du plus bas niveau des privilèges, où le niveau des privilèges monte sur une base temporaire pour effectuer une tâche bien précise, puis il redescend à son niveau initial. Il faut distinguer la gestion des droits d'accès à une plate-forme donnée de la gestion des droits d'accès à une application implémentée par un programme. Parmi les gestionnaires des droits d'accès, on peut citer :

- L'annuaire LDAP qui représente une structure centrale qui peut supporter tout type de table, y compris des tables d'authentification.
- Kerberos est un système qui gère les droits d'accès des systèmes distribués. Il est basé sur la notion des tickets. On peut le déployer sur Windows comme Unix.

1.9.4.5. Sécuriser le contrôle d'intégrité

Le contrôle d'intégrité est une partie principale de la politique de sécurité. Pour n'importe quelle politique de sécurité, on doit vérifier l'intégrité de l'implémentation et celle de la configuration du système. Il existe deux méthodes de vérification d'intégrité :

- La première consiste à faire une copie de tous les fichiers système et l'archiver puis comparer la version actuelle avec la version archivée.
- La deuxième consiste à créer une signature numérique et l'archiver puis comparer la signature numérique de la version actuelle avec la signature archivée.

1.9.4.6. Maîtriser la sécurité des applications

Il existe un nombre très important de vulnérabilités dans les applications, malgré toutes ces vulnérabilités, on peut toujours créer une suite de logiciel robuste comme l'a démontré l'équipe de développement du système OpenBSD. Pour arriver à une application robuste, il faut respecter ces quatre pratiques (Liorens et al, 2006):

- ***Codage défensif:*** pour coder d'une manière défensive, il faut appliquer certaines règles simples, mais très importantes comme la validation des entrées, le contrôle de la gestion de la mémoire dynamique, l'application des privilèges minimaux…etc.
- ***Environnements d'exécution sécurisés :*** il consiste à recompiler le code source d'un programme afin d'utiliser d'une manière transparente des environnements d'exécution sécurisés.
- ***Environnements cloisonnés :*** elle consiste à installer un programme relatif à l'application dans une zone cloisonnée. Il existe généralement deux techniques de cloisonnement: les cloisonnements système de type "prison" et "parking" comme la primitive "chroot" de l'Unix et la machine virtuelle.
- ***Tests de validation :*** consiste à faire les tests de logiciel comme les tests d'endurance aux entrées illégitimes, les tests d'endurance avec des entrées aléatoires, analyse rétrospective du code source …etc.

1.9.5. La protection de la gestion du réseau

Une bonne gestion du réseau informatique nous permet de pallier beaucoup de problèmes de sécurité et de faire face à un nombre très important d'attaques. Cette gestion comprend la gestion du routage réseau, supervision réseau, gestion du service des noms de domaine, gestion du service de mise à l'heure, et la gestion de la zone d'administration (Liorens et al, 2006).

La gestion du routage réseau (IS-IS, OSPF, BGP, etc.). S'articule sur les pratiques suivantes (Liorens et al, 2006):

- Décrire les règles de configuration du protocole IGP qui nous permet d'assurer un périmètre de sécurité du processus de routage.
- Décrire les règles de configuration du protocole EGP qui nous permet d'assurer un périmètre de sécurité du processus de routage.
- Mettre en œuvre un mécanisme de supervision des indicateurs relatifs aux tables de routage.
- Décrire les procédures d'intervention en cas de perturbation du routage du réseau.

La gestion de la supervision réseau (SNMP). S'articule sur les pratiques suivantes (Liorens et al, 2006):

- Consacrer des serveurs pour la supervision SNMP.
- Renforcer au maximum la sécurité des systèmes d'exploitation des serveurs.
- Localiser les serveurs SNMP dans la zone d'administration.
- Implémenter au maximum les options de sécurité disponibles (authentification).
- Suivre et appliquer tous les patchs de sécurité relatifs aux serveurs et aux services SNMP.
- Migrer vers une administration reposant sur le protocole IPSec.

La gestion du service des noms de domaine (DNS). S'articule sur les pratiques suivantes (Liorens et al, 2006) :

- Consacrer des serveurs pour la résolution des noms de domaine.
- Renforcer au maximum la sécurité des systèmes d'exploitation des serveurs DNS.
- Localiser les serveurs DNS dans la zone d'administration.
- Implémenter au maximum les options de sécurité disponibles (authentification).

- Suivre et appliquer tous les patchs de sécurité relatifs aux serveurs et aux services DNS.

La gestion du service de mise à l'heure (NTP). S'articule sur les pratiques suivantes (Liorens et al, 2006):

- Consacrer des serveurs pour la mise à jour des horloges des équipements réseau.
- Renforcer au maximum la sécurité des systèmes d'exploitation des serveurs NTP.
- Localiser les serveurs NTP dans la zone d'administration.
- Implémenter au maximum les options de sécurité disponibles (authentification).
- Suivre et appliquer tous les patchs de sécurité relatifs aux serveurs et aux services NTP.

La gestion de la zone d'administration. S'articule sur les pratiques suivantes (Liorens et al, 2006):

- Consacrer une zone d'administration pour le réseau.
- Renforcer la sécurité du périmètre de sécurité par des contrôles de filtrage très stricts.
- Authentifier tous les accès à la zone d'administration.
- Générer les traces des accès et les commandes passées à des fins d'investigation de sécurité.
- Installer des systèmes de détection d'intrusion au sein de la zone d'administration
- Consacrer un plan d'adressage spécifique.

1.10. Conclusion

La sécurité des systèmes informatiques est un domaine très vaste qui nécessite beaucoup de prudence et de vigilance. Il existe beaucoup de vulnérabilités auxquelles il faut faire face en utilisant les différents outils et techniques de sécurité informatique. Bien configure et protéger votre réseau, système et application est la clé de la bonne conduite d'une politique de sécurité au sien d'une organisation. Un bon administrateur réseau et système doit toujours prévoir toute sorte d'attaque en suivant les différentes étapes nécessaires afin de sécuriser les données de l'organisation parce que la plupart du temps les attaques sont irréversibles et comme le proverbe dit: « mieux vaut prévenir que guérir ».

Chapitre 2

Les systèmes de détection d'intrusion

Ce deuxième chapitre représente une introduction aux systèmes de détection d'intrusion où nous allons présenter la notion d'audit de sécurité, puis nous détaillons le concept des systèmes de détection d'intrusion et leurs structures générales. Après avoir fait la description des systèmes de détection d'intrusion, nous détaillons la taxonomie des IDSs. À la fin de ce chapitre, nous présentons les différentes mesures d'efficacité des IDSs.

2.1. Introduction

Les systèmes et réseaux informatiques contiennent diverses formes de vulnérabilité. Pour faire face à ces problèmes de sécurité, différents mécanismes ont été mis en place pour prévenir toute sorte d'attaque comme les pare-feux, l'authentification, les proxys… etc. Malheureusement, ces mécanismes ont des limites où certains types des attaques peuvent les contourner pour nuire la confidentialité, l'intégrité ou la disponibilité. C'est la raison pour laquelle un nouveau concept appelé système de détection d'intrusion a été introduit comme une seconde ligne de défense afin de renforcer la sécurité des systèmes informatiques. Ce concept a été introduit par James Anderson en 1980 dans le fameux rapport « COMPUTER SECURITY THREAT MONITORING AND SURVEILLANCE » (Anderson, 1980), où il a montré l'importance des traces d'audit pour relever toute violation potentielle de la politique de sécurité. Bien que l'idée d'Anderson soit très originale, le sujet n'a pas eu beaucoup de succès. Il a fallu attendre la concrétisation de cette idée par Denning en 1987 qui a proposé le premier modèle de détection d'intrusions dans son article « An intrusion detection model » (Denning, 1987) pour marquer réellement le départ du domaine de détection d'intrusion.

2.2. L'audit de sécurité

Un événement c'est toute activité système produite suite à un ensemble d'action effectué à un moment donné par un utilisateur, processus ou application. Le journal d'audit de sécurité c'est le fichier qui enregistre par ordre chronologique tout ou une partie des événements produits dans un système donné. Généralement le journal d'audite nous permet de connaitre l'opération faite, l'utilisateur qui la fait, quand il la fait, les ressources système affectés par cette opération, l'utilisateur a pu terminer l'opération sinon pourquoi l'opération a échoué (Mé and Alanou, 1996).

Pour réaliser l'audite de sécurité il faut répondre aux trois questions suivantes : Quoi audité ? Comment le collecter ? Comment l'analysé ?

2.2.1. Les activités auditables du système

D'après la politique de sécurité et le niveau de sécurité souhaité, l'administrateur peut définir des événements auditables correspondants à certaines informations qui peuvent être (Mé and Alanou, 1996) :

- **Un accès au système:** comprend toutes les informations qui nous permettent de détecter toute violation de sécurité comme: qui a accédé au système? Quand? Où ? et comment?
- **Un usage des ressources système:** comprend toutes les informations relatives à l'utilisation des ressources système comme l'utilisation des commandes système, l'utilisation de CPU, RAM, les entrées/sorties.
- **Un usage des fichiers :** comprend toutes les informations concernant l'accès aux fichiers comme l'horodatage de l'accès, type d'accès, source d'accès…etc.
- **Des événements liés aux applications:** comprend toutes les informations relatives aux événements engendrer par des applications qui peuvent influencer la sécurité du système comme le lancement et l'arrêt des applications, les entrées utilisés et les sorties produites…etc.
- **Les violations éventuelles de la sécurité:** comprend toutes les informations relatives aux tentatives d'accès non autorisé à des ressources système comme l'exécution d'une application ou d'une commande en mode privilégie, changement des droits d'accès …etc.
- **Les statistiques du système:** comprends toutes les informations de nature statistique qui peuvent nous aider à repérer toute activité anormale comme les statistiques sur le nombre de tentatives d'accès refusés.

2.2.2. La collecte des événements

Les systèmes d'exploitation actuels possèdent un mécanisme d'audit capable de générer certains types d'événement. Où le noyau garantit la génération et la collecte de ces événements. Concernant les applications on doit fournir aux développeurs un ensemble de primitives de génération et de collecte des événements. De cette manière on aura l'ensemble d'audit système et application (Mé and Alanou, 1996).

2.2.3. L'analyse du journal d'audit

L'objectif de l'analyse de l'audit de sécurité et de détecter toutes violations potentielles de la politique de sécurité qui peuvent atteindre la confidentialité, l'intégrité ou la disponibilité. La fréquence d'analyse des journaux d'audit peut être faite soit en temps réel ou en temps différé. Afin de minimiser les dégâts qui peuvent être engendrés par la violation de la politique de sécurité, il est recommandé de surveiller le système en temps quasi réel. Dans le cas d'un réseau, il faut créer un fichier d'audit global qui regroupe les différents audits collectés des différentes machines du réseau.

Certain type d'attaque modifier le journal d'audit afin d'effacer toute trace qui peut révéler cette attaque. Pour traiter ce problème, il est nécessaire de protéger le journal d'audit contre toute tentative de modification par des utilisateurs non autorisés. Dans le cas d'un réseau informatique, il faut protéger non seulement le fichier global d'audit, mais aussi le transfert des informations auditées.

2.3. Les systèmes de détection d'intrusion

Avant d'entamer les systèmes de détection d'intrusion, il faut éclaircir la notion d'intrusion qu'on puisse la définir par toute séquence active d'événement en relation qui tente de causer du tort comme interrompre le fonctionnement d'un système, usurper l'identité d'un utilisateur ou modifier des informations. Cette définition comprend toutes les tentatives qui réussissent ou celles qui échouent (Endorf et al, 2004).

2.3.1. Définition d'un système de détection d'intrusion

On peut définir un système de détection d'intrusion (IDS) comme tout outil, méthode et ressource qui nous aident à prévoir ou identifier toute activité non autorisée dans un réseau. Une partie du nom du système de détection d'intrusion est trompeuse, les systèmes de détection d'intrusion actuels ne détectent pas les intrusions, mais ils détectent les activités réseau qui peuvent être une intrusion ou non. La détection d'intrusion est typiquement une partie d'un système de protection total installé autour d'un système ou appareil. Il n'est pas une mesure de protection autonome (Endorf et al, 2004).

2.3.2. Les avantages d'un système de détection d'intrusion

Les systèmes de détection d'intrusion offrent beaucoup d'avantages comme (Endorf et al, 2004) :

- Une efficacité plus grande que celle de la détection manuelle des intrusions.

- L'utilisation d'une base de connaissance plus grande pour prédire les intrusions.
- La capacité de traiter un large volume de données.
- Produit une alerte presque en temps réel ce qui réduit le dommage potentiel des attaques.
- Des mesures de contre-attaque automatique comme la fermeture des sessions, désactivation des comptes utilisateur, lancement des scripts automatiques.
- L'ajout d'une valeur préventive forte.
- La création automatique des rapports et le jugement de la suite d'événements.

2.3.3. Le modèle de base d'un système de détection d'intrusion

Il existe plusieurs outils de détection d'intrusion, chaque outil utilise sa propre technique de détection et ses propres sources de données ce qui rend la comparaison entre ces outils très difficile voire impossible. Il est très intéressant de se disposer d'un modèle général qui englobe et standardise la structure d'un système de détection d'intrusion. Ce sujet a été le centre d'intérêt du groupe IDWG (Intrusion Detection Working Group) de l'IETF. IDWG a proposé le modèle général des systèmes de détection d'intrusion qui se compose de senseur (collecteur), analyseur, manager (administrateur). La figure suivante montre en détail les composants d'un système de détection d'intrusion.

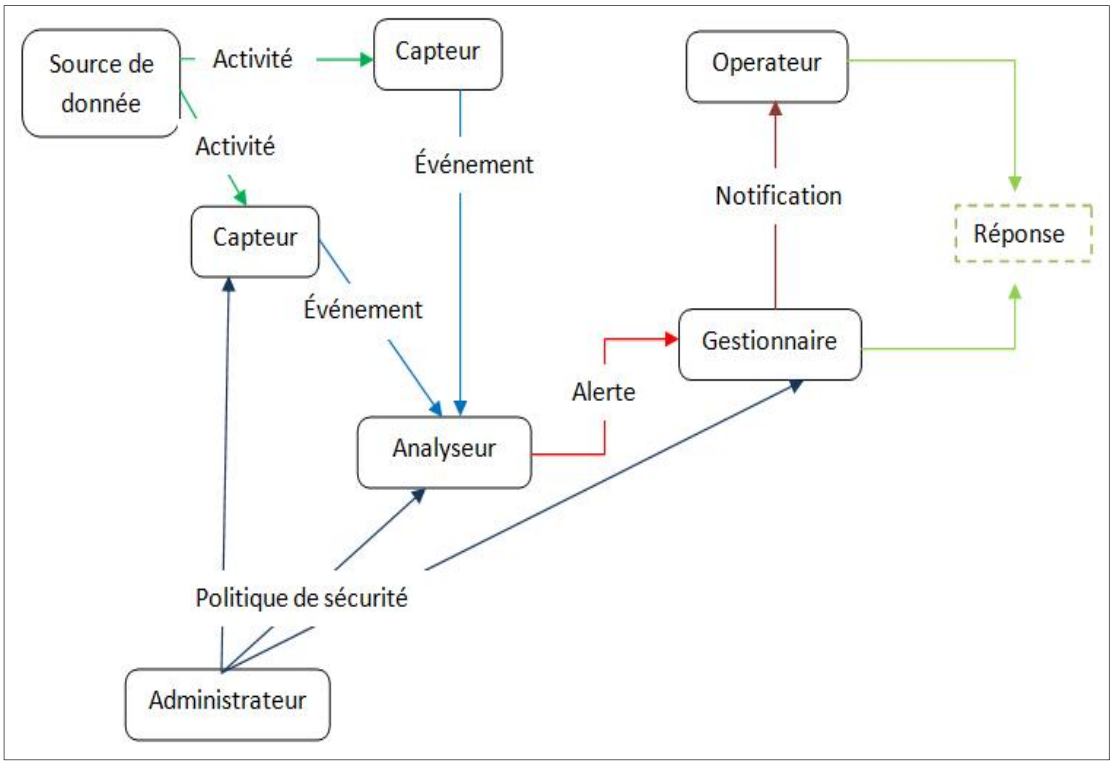

Figure 5 Le modèle générique de la détection d'intrusions proposé par l'IDWG (Wood and Erlinger, 2012)

- ***L'activité:*** c'est les éléments de la source ou les occurrences au sein de la source de données qui sont identifiés par le capteur ou l'analyseur comme étant à intérêt pour l'opérateur. Par exemple les sessions réseau montrant une activité inattendue de Telnet, les entrées des fichiers journaux du système d'exploitation montrant un utilisateur qui tente d'accéder à des fichiers auxquels il n'est pas autorisé, les fichiers journaux d'application montrant des échecs de connexion persistants… etc.
- ***L'administrateur:*** c'est le responsable de l'établissement de la politique de sécurité de l'organisation, donc celui qui déploie et configure l'IDS. Cette personne peut ou peut ne pas être l'opérateur de l'IDS. Dans certaines organisations l'administrateur est associé à un réseau ou à des groupes d'administration de système. Dans d'autres organisations, c'est une position indépendante.
- ***L'alerte :*** c'est un message qui passe de l'analyseur au gestionnaire pour lui informer qu'un événement d'intérêt a été détecté. Une alerte contient généralement des informations sur l'activité inhabituelle qui a été détectée ainsi que ces détails.
- ***L'analyseur :*** c'est le composant clé, il analyse les données recueillies par le capteur pour signaler les activités non autorisées ou indésirables ou les événements qui pourraient avoir un intérêt pour l'administrateur de sécurité. Dans la plupart des IDSs existants, le capteur et l'analyseur font partie d'un même composant.
- ***La source de données:*** c'est les informations brutes utilisées par le système de détection d'intrusion pour détecter les activités non autorisées ou non désirées. Les sources de données communes incluent (mais ne sont pas limités à) les paquets bruts du réseau, les journaux d'audit du système d'exploitation, les journaux d'audit d'applications et les données de contrôle générées par le système.
- ***L'événement:*** c'est toute occurrence détectée dans la source des données par un capteur et qui peut donner lieu à une alerte. Par exemple une attaque.
- ***Le gestionnaire:*** c'est l'élément clé ou le processus à partir de laquelle l'opérateur gère les différents composants du système. Les fonctions du gestionnaire comprennent généralement (mais ne sont pas limités à) la configuration du capteur, la configuration de l'analyseur, la gestion de la notification d'événements, la consolidation des données et la gestion des rapports.
- ***La notification:*** c'est la méthode avec laquelle le gestionnaire de l'IDS informe l'opérateur de la survenance d'une alerte. Dans de nombreux IDSs, la notification se fait via l'affichage d'une icône colorée sur l'écran du gestionnaire de l'IDS, la

transmission d'un e-mail ou un message, ou la transmission d'un Simple Network Management Protocol (SNMP) trap…etc.

- ***L'opérateur***: c'est l'utilisateur principal du gestionnaire de l'IDS. L'opérateur surveille souvent la sortie du système de détection d'intrusion et déclenche ou recommande d'autres actions.
- ***La réponse :*** c'est les mesures prises comme réponse à un événement. Les réponses peuvent être effectuées automatiquement par une entité dans l'architecture de l'IDS ou peuvent être initiées par un humain. L'envoi d'une notification à l'opérateur est une réponse très commune. Autres réponses incluent (mais ne sont pas limités à) la journalisation de l'activité, l'enregistrement des données brutes (à partir de la source de données) qui ont caractérisé l'événement, l'arrêt du réseau ou de l'utilisateur ou la session de l'application, la modification des contrôles d'accès réseau ou système.
- ***Le capteur:*** c'est le composant qui collecte des données à partir de la source de données. La fréquence de la collecte des données varie selon la configuration de l'IDS. Le capteur est mis en place pour transférer des événements à l'analyseur.

2.4. Taxonomie des systèmes de détection d'intrusion

Il existe de nombreux systèmes de détection d'intrusion, ces IDSs peuvent être classifiés d'après plusieurs critères. Cinq critères de classification des systèmes de détection d'intrusion ont été introduits par Hervé Debar, Marc Dacier et Andreas Wespi (Debar et al, 2000). La figure 2 résume ces cinq critères de classification des IDSs.

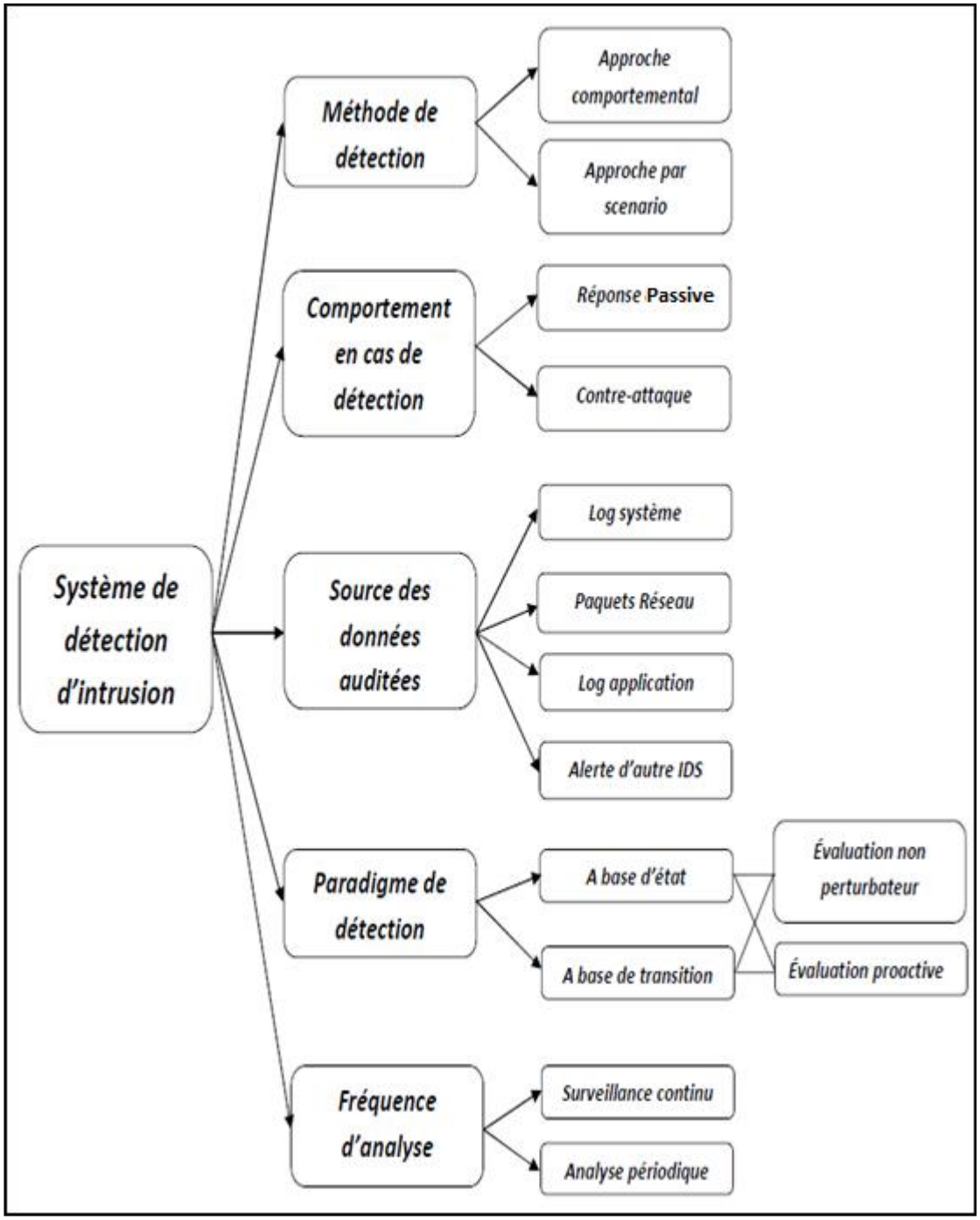

Figure 6 Les critères de classification des IDSs (Debar et al, 2000)

2.4.1. Les méthodes de détection d'intrusion

Il existe deux méthodes de détection, la première consiste à utiliser des connaissances accumulées sur les attaques puis les exploiter afin de prouver l'existence de d'autres attaques. La seconde consiste à créer un modèle basé sur le comportement habituel du système et surveiller toute déviation de ce comportement. La première méthode est appelée approche par scénario et la seconde est l'approche comportementale (Debar et al, 2000).

2.4.1.1. L'approche par scénario

Elle se base sur les connaissances accumulées sur des attaques spécifiques et les vulnérabilités du système. Le système de détection d'intrusion contient les informations sur les vulnérabilités et cherche toute tentative de les exploiter. Si l'IDS détecte une tentative, une alarme est déclenchée. En d'autres termes, toute action qui n'est pas explicitement reconnue comme une attaque est considérée comme acceptable. Par conséquent, la précision des systèmes de détection d'intrusion basée sur l'approche par scénario est bonne. Cependant, cette précision dépend toujours de la mise à jour des connaissances sur les attaques qui doit être régulière (Debar et al, 2000).

Avantage: les principaux avantages de cette approche sont : le taux très faible de fausse alarme et l'analyse contextuelle très détaillée. Donc, la compréhension des problèmes de sécurité et la prise des mesures préventives ou correctives sont devenues plus faciles pour l'administrateur de sécurité à l'aide de ce système de détection d'intrusion.

Inconvénient: les inconvénients comprennent la difficulté de rassembler les informations nécessaires sur les attaques connues et de garder la base à jour avec les nouvelles vulnérabilités. La maintenance de la base de connaissance du système de détection d'intrusion nécessite une analyse approfondie de chaque vulnérabilité, donc c'est une tâche fastidieuse. Cette approche est fondée sur des connaissances, donc elle doit faire face à la question de la généralisation de ces connaissances. La connaissance des attaques est très liée au système d'exploitation, la version, la plateforme et l'application. Par conséquent, le système de détection d'intrusion sera étroitement lié à un environnement donné. En outre, la détection des attaques internes impliquant un abus des privilèges est plus difficile à détecter, car il n'y a aucune vulnérabilité exploitée par l'attaquant.

Il existe plusieurs systèmes de détection d'intrusion qui utilisent l'approche par scénario où différent technique ont été utilisé pour les implémentés comme les systèmes experts, l'analyse de la signature, les réseaux de pétri, l'analyse de l'état transition.

2.4.1.2. L'approche comportementale

Les techniques de détection d'intrusion basées sur l'approche comportementale supposent que l'intrusion peut être détectée par l'observation de la déviation par rapport au comportement normal ou prévu du système ou des utilisateurs. Au début, le modèle du comportement normal est extrait à partir des informations de référence recueillies par divers moyens. Puis, le système de détection d'intrusion compare ce modèle avec l'activité actuelle. Si une déviation est détectée, une alerte sera déclenchée. D'une manière générale, on peut

dire que cette approche considère tout comportement qui n'est pas précédemment enregistré comme intrusion. Par conséquent, cette approche peut être complète, mais la précision reste son plus grand souci.

Avantage: le point fort de cette approche est qu'elle arrive à détecter les nouvelles formes d'attaques qui exploitent les nouvelles formes de vulnérabilités non connues auparavant. Cette approche est moins dépendante du système d'exploitation par rapport à l'approche par scénario. Elle peut aussi détecter les attaques d'abus de privilège qui n'exploite aucune vulnérabilité.

Inconvénient: le principal inconvénient de cette technique est le taux de fausses alarmes très élevé parce que l'ensemble du périmètre du comportement d'un système d'information ne peut pas être complètement couvert pendant la phase d'apprentissage. En outre, le comportement peut changer au fil du temps. Ce qui nous oblige à refaire l'apprentissage du comportement normal, ce qui cause soit l'indisponibilité temporaire du système de détection d'intrusion ou des fausses alarmes supplémentaires. De plus le système d'information peut subir des attaques au moment d'apprentissage. Par conséquent, le profil de comportement normal contiendra des comportements intrusifs qui ne seront pas détectés comme anormale. Il existe certains systèmes de détection d'intrusion qui utilisent l'approche comportementale, où différents techniques ont été utilisés pour les implémenter comme les systèmes experts, la méthode statistique, les réseaux de neurones (Debar et al, 2000).

Les systèmes de détection d'intrusion commerciaux actuels utilisent une seule approche et la plupart d'entre eux utilisent l'approche par scénario pour les raisons suivantes:

- L'approche par scénario est plus facile que l'approche comportementale dans l'implémentation. De plus, le taux élevé de fausse alarme pour l'approche comportementale lui rend inapproprié pour des IDSs commerciaux.
- La vitesse de traitement des audits est un facteur très important c'est la raison pour laquelle les signatures sont utilisées à la place des règles.

2.4.2. Le comportement de l'IDS en cas de détection (IDS actif VS IDS passif)

La plupart des systèmes de détection d'intrusion sont des systèmes passifs, donc lorsqu'une attaque est détectée, une alarme est générée, mais aucune contre-mesure ne sera appliquer activement pour arrêter ou limiter les dégâts d'une attaque. Ce qui signifie que dans un contexte de recherche, les IDSs passifs n'impactent pas la disponibilité du système en cas de grand nombre de fausses alarmes.

Certains systèmes de détection d'intrusion basés sur l'analyse périodique ont une capacité active additionnelle lorsqu'un problème de sécurité est détecté dans la configuration du système. Ces outils génèrent des scripts qui peuvent supprimer la vulnérabilité (par exemple en changeant les permissions sur un système de fichiers) et restaurer le système à son état antérieur. Par conséquent, l'application d'une contre-mesure est devenue plus sécurisée par la capacité de revenir rapidement à un état antérieur en cas d'anomalie. Avec l'arrivée des produits de détection d'intrusion, l'élément de contre-mesure est devenu de plus en plus prééminent où plusieurs IDSs inclurent la capacité de couper les connexions qui transportent les attaques, bloquant la connexion des hôtes à partir desquels les attaques proviennent ou la reconfiguration des autres équipements tels que les pare-feux ou les routeurs.

Grâce à des stratégies de sécurité proactives, les produits de détection d'intrusion sont de plus en plus réputés comme outils fiables (Debar et al, 2000).

2.4.3. La source des données auditées

Les systèmes de détection d'intrusion basés système sont les premières IDSs qui ont vu le jour. Quand le premier système de détection a été inventé, l'environnement cible été le système sur lequel tous les utilisateurs opèrent. Les interactions avec l'extérieur du système ont été vraiment rares, ce qui a simplifié beaucoup la tâche du système de détection d'intrusion. Le système de détection d'intrusion analyse les informations d'audit fourni par l'ordinateur central, soit localement ou sur une machine séparée, et signale tous les événements suspects.

Avec l'arrivée des réseaux informatiques, plusieurs prototypes de systèmes de détection d'intrusion ont été développés afin de répondre à ces nouveaux besoins. Le premier essai dans ce domaine a été de créer un IDS communicant à partir des IDSs basés système. Dans un environnement distribué, les utilisateurs passent d'une machine à l'autre, en changeant éventuellement leurs identités lors de leurs déplacements. Donc, ils lancent leurs attaques sur plusieurs systèmes. Par conséquent, le système de détection d'intrusion local doit disposer d'un échange des informations avec ses égaux. Cet échange des informations se fait à plusieurs niveaux, que ce soit un échange des enregistrements d'audit brut sur le réseau, ou d'émettre des alarmes déduites d'une analyse locale. Les deux solutions proposées engendrent des coûts. Le transfert des audits peut potentiellement avoir un impact énorme sur la bande passante du réseau, alors que le traitement localement affecte les performances du poste de travail.

Après la généralisation de l'utilisation de l'internet, les systèmes de détection d'intrusion doivent faire face à des attaques contre le réseau lui-même. Les attaques réseau (DNS Spoofing, TCP détournement, balayage des ports, Ping de la mort, etc.) ne peuvent pas être détectées par l'analyse d'audit de sécurité. Par conséquent, des outils spécifiques ont été développés pour snifer les paquets réseau en temps réel et rechercher des attaques réseau. De plus, un certain nombre d'attaques contre les serveurs classiques peuvent également être détectées par l'analyse de la charge utile du paquet et la recherche des commandes suspectes. Ces outils sont souvent intéressants pour les administrateurs système, car certains outils peuvent être installés à des endroits stratégiques du réseau pour couvrir la plupart des attaques actuelles.

Les approches hybrides ont également été développées. Elles utilisent les outils de détection d'intrusion basés réseau et système dans un environnement multi-systèmes (Debar et al, 2000).

2.4.3.1. Les informations à base de système

Les données d'audit système sont le seul moyen pour recueillir des informations sur les activités des utilisateurs d'une machine donnée. D'autre part, cette source de données est également vulnérable à des altérations dans le cas d'une attaque qui termine avec succès. Cela crée une contrainte importante, où le système de détection d'intrusion doit analyser l'audite de sécurité en temps réel et générer des alarmes avant que l'attaquant essaye de modifier l'audite ou arrêter le système de détection d'intrusion (Debar et al, 2000).

2.4.3.1.1. Les commandes système

Tout système d'exploitation possède des primitives qui nous permettent d'obtenir une image sur l'état du système au moment de son exécution (à l'instant t). Par exemple sur Unix on a les primitifs ***ps, pstat, vmstat, getrlimit***. Ces commandes fournissent des informations précises et ciblées sur les événements, car ils examinent directement la mémoire noyau du système. Par contre, il est très difficile d'utiliser ces commandes pour une collecte d'information continue, car ils n'offrent pas des données bien structurées (Debar et al, 2000).

2.4.3.1.2. Les statistiques relatives aux systèmes

Les statistiques relatives aux systèmes représentent l'une des plus anciennes sources d'information pour le comportement du système. Elles fournissent des informations sur la consommation des ressources partagées par les utilisateurs du système. Les ressources sont

le temps processeur, mémoire, l'utilisation du disque ou du réseau, les applications lancées…etc. Cette omniprésence a conduit certains concepteurs des prototypes de détection d'intrusion de tenter à utiliser ces statistiques comme source d'audit. Par contre, les statistiques relatives aux systèmes possèdent un certain nombre d'inconvénients, ce qui les rend peu fiables comme source d'audit. Par exemple, les fichiers des statistiques "stats" sont parfois situés dans la même partition de disque comme le répertoire /temp. Les utilisateurs peuvent alors simplement remplir la partition du disque jusqu'à 90%, donc le processus de calcul des statistiques s'arrête. Il existe encore d'autres inconvénients plus importants comme: le manque de paramétrage, absence d'identification précise de commande, le retard pour obtenir des informations.

En raison de ces inconvénients, les statistiques relatives au système n'ont jamais été utilisées dans l'approche de détection par scénario, et rarement utilisées pour la détection d'intrusion basée sur le comportement (Debar et al, 2000).

2.4.3.1.3. Le syslog

Le syslog est un service d'audit fourni aux applications par le système d'exploitation. Ce service reçoit une chaîne de caractère des applications, cette chaîne contient le temps et le nom du système sur lequel s'exécute l'application. Ces informations sont archivées soit localement ou à distance. Le syslog est très facile à utiliser, ce qui a incité de nombreux développeurs d'applications à l'utiliser comme piste de vérification. Un certain nombre d'applications et de services réseau utilisent ce service, tels que login, sendmail, nfs, http, et cela inclut également des outils liés à la sécurité tels que sudo, klaxon ou TCP wrappers. Par conséquent, des outils de détection d'intrusion qui utilisent les informations fournies par le Syslog ont été développés, par exemple Swatch (Stephen et al, 1993). Bien que le syslog soit une source d'audit légère qui ne génère pas une grande quantité de données d'audit par machine, un grand réseau peut générer un grand nombre de messages qui ont que peu d'entrées importantes pour la sécurité (Debar et al, 2000).

2.4.3.1.4. Les traces d'audit de sécurité C2

Toutes les traces d'audit de sécurité ont le même principe de base. Ils enregistrent le passage des instructions exécutées par le processeur dans l'espace utilisateur et les instructions exécutées dans l'espace « Trusted Computing Base » (TCB). Ce modèle de sécurité repose sur le fait que le TCB est fiable, et que les actions dans l'espace utilisateur ne peuvent pas nuire la sécurité du système, ainsi que les actions liées à la sécurité peuvent influer le système uniquement lorsque les utilisateurs demandent des services de la TCB. Les traces

d'audit de sécurité C2 sont les principales sources des informations d'audit pour la majorité des prototypes et des outils de détection d'intrusion basés système, car elles sont actuellement le seul mécanisme fiable pour recueillir des informations détaillées sur les actions prises sur un système d'information.

Beaucoup de travaux ont été menés par plusieurs groupes de recherche pour définir les informations qui devraient être figurées dans le journal de trace d'audit de sécurité ainsi que le format commun de ces fichiers (Debar et al, 2000).

2.4.3.2. Les données réseau

2.4.3.2.1. Les informations SNMP

Le « Simple Network Management Protocol (SNMP) Management Information Base (MIB) » est un répertoire d'information utilisé pour des raisons de gestion du réseau. Il contient les informations de configuration (table de routage, adresses, noms) ainsi que les informations liées à la performance du réseau et les conteurs qui mesurent le trafic sur les différentes interfaces réseau et aux différentes couches. SNMP MIB représente une intéressante source d'audit pour les systèmes de détection d'intrusion (Debar et al, 2000).

2.4.3.2.2. Les paquets réseau

Les sniffeurs réseau sont plus en plus utilisés chez les hackers, mais ils représentent aussi une source de données très importantes qui recueille les informations relatives aux événements qui se produisent sur le réseau. La plupart des accès aux ordinateurs sensibles se font via les réseaux informatiques, alors que la capture des paquets avant qu'ils entrent au serveur est le moyen le plus efficace pour les contrôler. Les attaques de type déni de service (DoS) provient dans la plupart du temps du réseau, alors qu'un IDS basé système n'est pas capable de détecter ces attaques. L'analyse des paquets représente le moyen le plus efficace pour détecter les attaques de type Déni de Service (DoS).

2.4.3.3. Les fichiers logs des applications

Avec la grande augmentation de l'utilisation des serveurs d'application, les fichiers logs des applications sont devenus une source d'information pour les systèmes de détection d'intrusion. La comparaison entre cette source de données et les traces d'audit de sécurité C2 ou les paquets réseau montre trois avantages (Debar et al, 2000) :

- **La précision:** Les traces d'audite de sécurité C2 ou les paquets réseau nécessitent un traitement avant que le système de détection d'intrusion arrive à comprendre quelle information est actuellement reçue par l'application. Par

contre le fait de recevoir l'information directement du log nous rend presque sûr que l'information est précise.

- **La complétude:** Les traces d'audit de sécurité C2 ou les paquets réseau exigent le montage de plusieurs sources d'audit ou plusieurs paquets réseau potentiellement sur plusieurs hôtes, ce qui est très difficile à réaliser. Par contre le journal des applications contient toutes les informations pertinentes. En outre, l'application peut fournir des données internes qui ne se présentent ni dans les audits de sécurité ni dans les paquets réseau.
- **La performance:** En laissant l'application sélectionne les informations pertinentes pour la sécurité, la surcharge induite par le mécanisme de collecte est fortement réduite par rapport à l'audit de sécurité C2.

En revanche, il existe deux inconvénients pour l'utilisation des fichiers logs d'application pour la détection d'intrusion:

- **La potentielle absence d'information:** Les attaques ne sont détectées que lorsque le log d'application est écrit. Si l'attaque arrive à empêcher l'écriture dans le log de l'application (ce qui est le cas dans des nombreuses attaques par déni de service), alors les informations requises par le système de détection d'intrusion seront absentes.
- **Les attaques de bas niveau:** Certains nombres d'attaques comme les attaques par déni de service peuvent viser les niveaux inférieurs du système, tels que les pilotes de réseau. Comme ces attaques n'exécutent pas du code d'application, ils ne peuvent pas être visibles dans les logs d'application (Debar et al, 2000).

2.4.4. Les alertes de la détection d'intrusion

La détection d'intrusion est devenue très populaire et les entreprises l'utilisent sur une grande échelle. Face à cette grandeur d'utilisation, les systèmes de détection d'intrusion doivent faire face au nombre élevé d'alarmes générées. Par conséquent, une nouvelle génération des systèmes de détection d'intrusion qui ne détecte pas directement les attaques, mais plutôt des informations corrélées provenant de plusieurs outils de niveau inférieur sont développés. Cette nouvelle génération des systèmes de détection d'intrusion utilise des techniques de corrélation et les techniques de fouille de données pour présenter une image plus condensée de l'activité anormale découverte. Cela présente un grand avantage vu que les systèmes de détection d'intrusion ont tendance d'avoir des taux de fausse alarme élevés.

En corrélant les sorties de plusieurs outils qu'on connaît leurs forces et leurs faiblesses, le système de détection d'intrusion peut éviter automatiquement plusieurs fausses alarmes (Debar et al, 2000).

2.4.5. Le paradigme de détection

2.4.5.1. Les IDSs basés État VS les IDSs basés transaction

Il existe essentiellement deux paradigmes pour les moteurs de détection d'intrusion. La première classe des moteurs de détection d'intrusion se base sur les états, la deuxième classe se base sur les transitions entre états. La figure 7 montre ces paradigmes en se basant sur les termes de la fiabilité. NORMAL représente l'état désiré du système. ERROR1 et ERROR2 représentent deux étapes d'une chaîne qui finit par conduire le système à un état d'échec «FAILURE ». L'objectif du système de détection d'intrusion est de détecter que le système a quitté l'état normal avant qu'il atteigne l'état de défaillance (Debar et al, 2000).

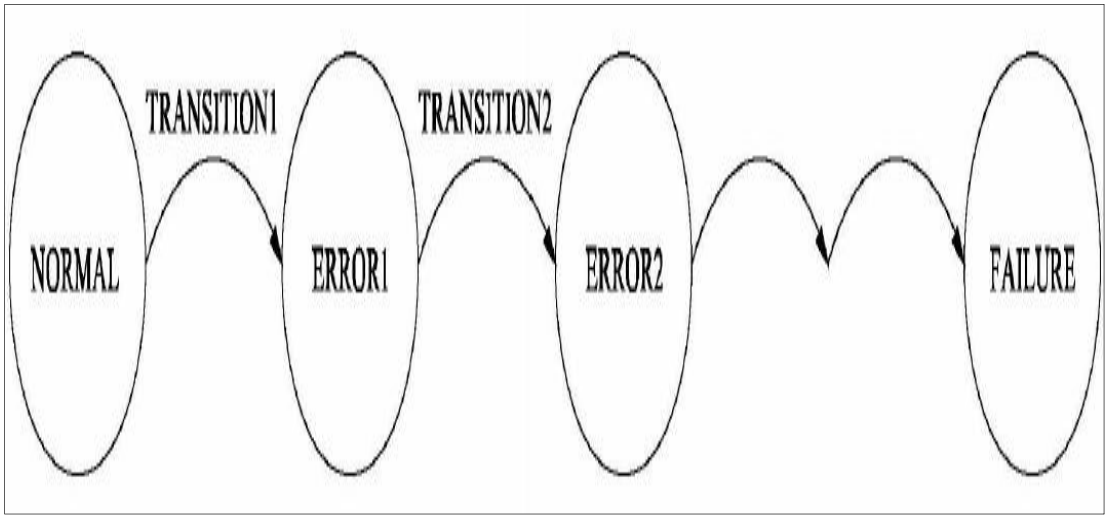

Figure 7 État VS Transition (Debar et al, 2000)

Les outils de détection d'intrusion peuvent faire plusieurs choses pour atteindre cet objectif (Debar et al, 2000):

- **Reconnaître l'état normal:** C'est moyennement utile, car il ne donne aucune information sur l'état du système dans la chaîne vers l'état de défaillance. Il est également extrêmement difficile de caractériser la normalité, comme le montre la détection d'intrusion basée sur l'approche comportementale.
- **Reconnaître un état d'erreur sur le chemin:** Il est généralement possible de caractériser les états d'erreur en chemin.
- **Reconnaître certaines transitions conduisant à un état d'erreur**

2.4.5.2. L'analyse non perturbatrice VS proactive

L'analyse de l'état ou de la transition peut être effectuée de deux façons, soit par une observation de la non-perturbation du système ou d'essayer d'une manière proactive d'évaluer l'état ou la transition qui peut modifier l'état du système par la suite (Debar et al, 2000).

2.4.5.2.1. L'analyse non perturbatrice

Une observation non perturbatrice consiste à évaluer la partie des vulnérabilités des versions des applications ou des bannières, puis les comparer avec une base des vulnérabilités connue. Si la version de l'application est dans la base, le système est étiqueté comme étant dans l'état vulnérable, sinon il sera marqué comme étant à l'état sécurisé. Ce type d'analyse tente de minimiser l'impact sur le système quand il examine son état ou sa transition (Debar et al, 2000).

2.4.5.2.2. L'analyse proactive

Cette classe d'outils effectue une analyse proactive en déclenchant explicitement des événements sur l'environnement pour déterminer les états ou créer des transitions (Debar et al, 2000).

2.4.6. La fréquence de l'analyse

Il existe deux façons dont les systèmes de détection d'intrusion effectuent leurs analyses qui sont: analyse continue et analyse périodique. Un outil de détection d'intrusion dynamique effectue une analyse continue et en temps réel par l'acquisition d'informations sur les mesures prises sur l'environnement dès qu'ils se produisent. Un outil de détection d'intrusion statique prend périodiquement un aperçu de l'environnement et analyse ce cliché à la recherche des logiciels vulnérables, des erreurs de configuration,…etc. (Debar et al, 2000).

2.5. L'efficacité des IDSs

Il existe cinq mesures qui nous permettent d'évaluer l'efficacité des systèmes de détection d'intrusion (Debar et al, 2000) :

- **La précision:** Un système de détection d'intrusion précis est un système qui détecte les attaques sans faire des fausses alarmes. La non-précision survient lorsqu'un système de détection d'intrusion déclare comme anormale ou intrusive une action légitime dans l'environnement.
- **La performance de traitement**: La performance de traitement d'un système de détection d'intrusion est mesurée par la vitesse avec laquelle les événements d'audit

sont traités. Si la performance de traitement du système de détection d'intrusion est faible, alors la détection en temps réel n'est pas possible.

- **La complétude:** La complétude est la capacité d'un système de détection d'intrusion de détecter toutes les attaques. La non-complétude se produit lorsque le système de détection d'intrusion ne parvient pas à détecter une attaque. Cette mesure est beaucoup plus difficile à évaluer par rapport aux autres mesures, car il est impossible d'avoir une connaissance globale sur les attaques ou les abus des privilèges.
- **La tolérance aux pannes:** Un système de détection d'intrusion devrait être résistant aux attaques en particulier les attaques de déni de service, donc un IDS devrait être conçu avec cet objectif. Ceci est particulièrement important parce que la plupart des systèmes de détection d'intrusion s'exécutent dans des systèmes d'exploitation ou des matériels qui sont connus pour être vulnérables aux attaques.
- **La rapidité:** Un système de détection d'intrusion doit exécuter et propager son analyse le plus rapidement possible pour permettre à l'agent de sécurité de réagir, afin de minimiser les dégâts possibles, et aussi pour empêcher l'attaquant d'altérer la source de vérification ou interrompre le fonctionnement du système de détection d'intrusion. Cela ne se limite pas à la mesure de performance, car il englobe non seulement la vitesse de traitement, mais aussi le temps nécessaire pour la propagation de l'information et le temps de réagir.

2.6. Conclusion

Les systèmes de détections d'intrusion représentent une seconde ligne de défense contre les attaques informatiques. Ce mécanisme nous permet de faire face aux attaques qui peuvent contourner les mécanismes de sécurité classique comme les pare-feux, l'authentification, le proxy…etc. Grâce aux différentes sources d'information et aux différents modes d'analyse et la différente localisation des IDSs, ce mécanisme représente un atout dans la guerre contre la cybercriminalité.

Chapitre 3

Les techniques de détection d'intrusion

Ce chapitre représente un état de l'art sur les noyaux des systèmes de détection d'intrusion. Dans ce chapitre, nous abordons les différentes techniques et méthodes de détections d'intrusion où on commence par la première génération des systèmes de détection d'intrusions. Après avoir analysé les faiblesses de cette première génération, nous présentons les techniques de data mining qui représentent la deuxième génération des systèmes de détection d'intrusions.

3.1. Introduction

Depuis la publication de l'article de Denning qui a proposé le premier modèle de détection d'intrusion (Denning, 1987), plusieurs travaux ont été faits pour construire un système de détection d'intrusion performant et très précis. La conception de cette première génération se base sur les connaissances des experts de sécurité où les méthodes statistiques et les approches de l'intelligence artificielle sont utilisées pour construire les noyaux (moteurs) des modèles. Face à des problèmes tels que le grand volume du trafic réseau, la distribution des données très déséquilibrée, la difficulté de prendre une décision entre le comportement normal et anormal et l'exigence d'une adaptation permanente pour des environnements en constante évolution, les techniques de l'intelligence artificielle ont montré beaucoup de limites. Pour faire face à ces nouveaux défis les techniques de data mining sont utilisés (Xiaonan and Banzhaf, 2010). Grâce à ces techniques de data mining, des modèles de détection d'intrusion plus rapides et plus précis ont été développés. Ces modèles représentent la deuxième génération des systèmes de détection d'intrusion.

3.2. La première génération des systèmes de détection d'intrusion

Le modèle proposé par Denning (Denning, 1987) représente la source d'inspiration sur laquelle se base le développement des systèmes de détection d'intrusion. Les méthodes de détection d'intrusion représentent le cerveau du système de détection d'intrusion. Les techniques ou les méthodes utilisées dans la première génération des systèmes de détection d'intrusion dépendent de l'approche adoptée qui peut être comportementale ou par scénario.

3.2.1. Les méthodes utilisées pour la détection comportementale

L'approche comportementale consiste à observer la déviation par rapport au comportement normal ou prévu du système ou des utilisateurs. Il existe plusieurs méthodes de détection d'intrusion utilisées pour implémenter cette approche. Les principales méthodes utilisées

sont : la méthode statistique, les systèmes experts, les réseaux de neurones, l'immunologie (Debar et al, 2000).

3.2.1.1. La méthode statistique

La méthode la plus utilisée pour construire les systèmes de détection d'intrusion basés comportemental est la méthode statistique (Javitz et al, 1993) (Helman et al, 1992) (Helman and Liepins, 1993). Elle consiste à mesurer le comportement de l'utilisateur ou du système par un nombre de variables échantillonnées dans le temps. Ces variables comprennent le temps de connexion et de déconnexion de chaque session, l'utilisation de la mémoire, l'occupation du processeur, l'accès aux fichiers. Le temps de la période d'échantillonnage varie de très court (quelques minutes) à long (un mois ou plus).

Le modèle de base conserve les moyennes de toutes ces variables, puis il les compare avec les valeurs des variables, où il détecte si les seuils sont dépassés. Ce modèle de base est très simple pour représenter les données d'une manière fiable. C'est la raison pour laquelle un modèle plus complexe a été développé (Javitz et al, 1993) (Javitz and Valdes, 1991), où il compare les profils des activités des utilisateurs à long terme et à court terme. Les profils doivent être régulièrement mis à jour à chaque fois que le comportement des utilisateurs évolue. Ce modèle statistique est maintenant utilisé dans un certain nombre de systèmes de détection d'intrusion ainsi que dans des nombreux prototypes (Debar et al, 2000).

3.2.1.2. Le système expert

Un système expert est capable de reproduire les mécanismes de reconnaissance d'un expert dans un domaine particulier. Il se compose d'une base des faits, une base des règles et un moteur d'inférence. Le système expert a été utilisé pour les systèmes de détection d'intrusion comportementale, où la base des règles peut être conçue de deux façons :

- La première s'appuie sur un ensemble de règles qui décrivent statistiquement le comportement des utilisateurs, où elle utilise les enregistrements de leurs activités sur une période de temps donnée. L'activité courante est comparée de ces règles afin de détecter un comportement incohérent. La base des règles est reconstruite régulièrement pour tenir compte des nouvelles utilisations. Wisdom et Sense (Vaccaro and Liepins, 1989) sont des IDSs basés sur ce mode de fonctionnement.

- Dans la deuxième approche, on vérifie les actions des utilisateurs en fonction d'un ensemble de règles qui décrivent la politique du bon usage, et on signale toute action

qui ne correspond pas aux modèles acceptables. AT&T's Computer Watch (Dowell and Ramstedt, 1990) est un IDS basé sur ce mode de fonctionnement.

Cette approche est utile pour des profils d'utilisation fondés sur les politiques de sécurité, mais elle est moins efficace que l'approche statistique pour le traitement des grandes quantités d'informations d'audit.

3.2.1.3. Les réseaux de neurones

Un réseau de neurones artificiels est un modèle de calcul inspiré du mode de fonctionnement des neurones biologiques. Les réseaux neuronaux sont utilisés pour apprendre la relation entre deux ensembles des informations, puis généraliser cette relation. Dans le domaine de la détection d'intrusion, les réseaux de neurones ont été principalement utilisés pour apprendre le comportement des utilisateurs du système. Certaines équivalences entre les modèles des réseaux de neurones et statistiques ont été illustrées dans (Gallinari et al, 1988) et (Sarle et al, 1994). L'avantage d'utiliser des réseaux de neurones par rapport à l'approche statistique réside dans le moyen simple d'exprimer des relations non linéaires entre les variables, et dans le fait que l'apprentissage / réapprentissage du réseau de neurones est automatique.

Des expériences ont été effectuées en utilisant un réseau de neurones pour prédire le comportement des utilisateurs (Debar et al, 1992). Ces expériences ont montré que le comportement des utilisateurs root UNIX est extrêmement prévisible (en raison de l'activité très régulière générée par des actions automatiques du système, des démons, etc.), et que le comportement de la plupart des utilisateurs est également prévisible, et qu'il y a une très petite fraction des utilisateurs dont le comportement est imprévisible.
Les réseaux de neurones sont une technique de calcul intensif, et ils ne sont pas très utilisés par la communauté de détection d'intrusion.

3.2.1.4. L'immunologie

L'immunologie dans le domaine de l'informatique a été introduite par Forrest et al. (Forrest et al, 1997). Un système immunitaire artificiel (SIA) est inspiré des principes du fonctionnement du système immunitaire naturel. Ces algorithmes utilisent les caractéristiques du système immunitaire qui apprend et mémorise pour résoudre les problèmes. Cette technique vise à construire un modèle de comportement normal des services réseau UNIX, plutôt que le comportement des utilisateurs. Ce modèle utilise des

courtes séquences d'appels système effectués par les processus. L'outil rassemble tout d'abord un ensemble de traces d'audit de référence qui représente le comportement approprié du service, puis extrait un tableau de référence contenant toutes les bonnes séquences connues des appels système. Ces modèles (Pattern) sont ensuite utilisés pour la surveillance en temps réel pour vérifier si les séquences générées sont répertoriées dans le tableau, sinon, le système de détection d'intrusion génère une alarme. Cette technique a un taux de fausses alarmes potentiellement très faible si la table de référence est suffisamment exhaustive.

L'inconvénient de cette méthode réside dans ses faiblesses pour les erreurs de configuration dans un service, c'est-à-dire quand les attaques utilisent des mesures légitimes prises par le service pour accéder sans autorisation.

3.2.2. Les méthodes utilisées pour la détection par scénario

L'approche par scénario consiste à utiliser les connaissances accumulées sur des attaques spécifiques et les vulnérabilités du système. Il existe plusieurs méthodes de détection utilisées pour implémenter cette approche. Les principales méthodes utilisées sont : les systèmes experts, l'analyse de la signature, les réseaux de Pétri, l'analyse de l'état transition et les algorithmes génétique.

3.2.2.1. Le système expert

Les systèmes experts (Fan et al, 2004) sont utilisés principalement par des systèmes de détection d'intrusion basés scénario. Le système expert contient un ensemble de règles qui décrivent les attaques. Les événements de l'audit de sécurité sont ensuite traduits en des faits portant leurs significations sémantiques pour le système expert. Le moteur d'inférence tire des conclusions à l'aide de ces règles et des faits. Les langages basés sur des règles (Habra et al, 1992) sont un outil naturel pour la modélisation de la connaissance que les experts ont recueilli sur les attaques. Cette approche nous permet de naviguer systématiquement dans les traces d'audit à la recherche des preuves d'une tentative d'exploitation des vulnérabilités connues. Ils sont également utilisés pour vérifier la bonne application de la politique de sécurité d'une organisation.

Les systèmes experts demeurent toujours comme l'une des techniques de détection d'intrusion les plus utilisées pour l'approche par scénario surtout pour les IDS commerciaux.

3.2.2.1. L'analyse de la signature

L'analyse de la signature suit exactement les mêmes démarches d'acquisition des connaissances faites par les systèmes experts, mais les connaissances sont exploitées d'une manière différente. La description sémantique des attaques se transforme en des informations que l'on trouve dans les traces d'audit d'une manière simple. Par exemple, des scénarios d'attaque pourraient être traduits en séquences d'événements d'audit qu'ils génèrent ou en des modèles de données qui peuvent être recherchés dans la trace d'audit généré par le système.

Cette technique permet une mise en œuvre très efficace et applicable pour les produits commerciaux de détection d'intrusion (Haystack Labs, 1997), (Internet Security Systems, 1997), (Wheel Group Corporation).

Le principal inconvénient de cette technique comme toutes les approches basée sur l'approche par scénario est la nécessité de faire des mises à jour fréquentes. Cette situation est aggravée par l'obligation de représenter toutes les facettes possibles des attaques par des signatures. Cela nous conduit à représenter une attaque par un certain nombre de signatures, au moins une pour chaque système d'exploitation pour que le système de détection d'intrusion devient portable.

3.2.2.2. Les réseaux de Pétri

Le réseau de Pétri a été utilisé pour représenter les signatures des intrusions. IDIOT (Kumar and Spafford, 1994) est un IDS qui utilise le réseau de Pétri coloré, il a été développé par l'université de Purdue. Les avantages des réseaux de Pétri colorés résident dans leurs capacités de généralisation, leurs simplicités conceptuelles et leurs représentations graphiques. En raison de la grande capacité de généralisation du réseau de Pétri coloré, même les signatures complexes peuvent facilement être écrites. Cependant, la comparaison entre cette signature complexe et les traces d'audit peut devenir très coûteuse en termes de calcul.

3.2.2.3. L'analyse de l'état transition

L'analyse de l'état transition est une technique proposée par Porras et Kemmerer, elle a été implémentée dans un système UNIX au premier temps (Porras and Kemmerer, 1992), puis dans d'autres environnements. Cette technique consiste à décrire les attaques avec un

ensemble d'objectifs et de transitions, puis les représenter sous forme de diagrammes de transition d'état.

3.2.2.4. Les algorithmes génétiques

Les algorithmes génétiques ont été proposés par John Holland en 1975 (Holland, 1975). Les algorithmes génétiques simulent la théorie de darwinienne pour le processus de l'évolution naturelle. Ils utilisent un vocabulaire similaire à celui de la génétique naturelle. Le but de ces algorithmes est de trouver une solution proche de la solution optimale d'un problème donnée.

Les algorithmes génétiques ont été utilisés dans l'approche par scénario afin de trouver les signatures des attaques prédéfinies dans les traces d'audit de sécurité (Mé, 1995). L'approche de Ludovic Mé (Mé, 1995) consiste à traduire le problème de la recherche de la signature des attaques dans l'audit de sécurité en un problème d'optimisation, où on recherche la meilleure solution qui maximise le nombre des attaques détectées toute en respectant la contrainte de la correspondance entre le nombre des événements de chaque signature d'attaque et le nombre des événements audités.

3.2.3. Un exemple d'IDS de première génération (la résolution génétique de PASFAS)

3.2.3.1. L'analyse simplifiée du fichier d'audit de sécurité (PASFAS)

L'analyse de l'audit de sécurité est comme le diagnostic médical, elle vise à déterminer l'ensemble des conditions qui peut expliquer la présence des symptômes observés. Pour cette raison, l'expert utilise des connaissances spécifiques (les scénarios des attaques) de type causes à effet. L'expert utilise ses connaissances pour élaborer des hypothèses qui confrontent la réalité observée. S'il y a des symptômes non expliqués par l'hypothèse alors l'hypothèse faite est présumée mauvaise et une nouvelle hypothèse plus pertinente doit être élaborée (Mé, 1995). Dans cette approche, les scénarios des attaques sont modélisés par un ensemble de couples (E_i, N_i) où le E_i est le type d'événement et N_i est le nombre des occurrences de ce type d'événement dans le scénario. Cette approche est appelée «Analyse simplifiée du fichier d'audit de sécurité». Formellement, le problème de l'analyse simplifiée du fichier d'audit de sécurité peut être exprimé par la figure suivante.

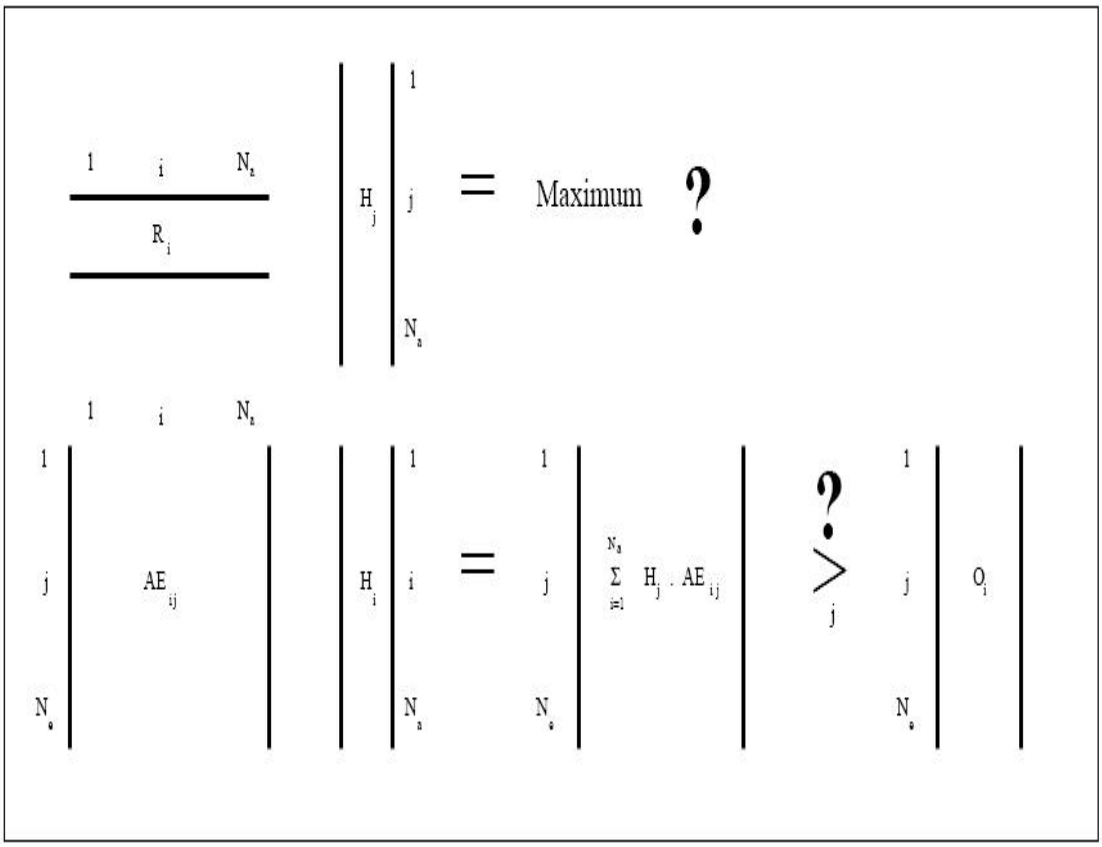

Figure 8 Le problème de l'analyse simplifiée du fichier d'audit de sécurité (Mé, 1995)

- N_e: le nombre de types des événements audités.
- N_a: le nombre potentiel des attaques connues.
- AE: représente la matrice attaque événement $N_a \times N_e$ elle définit l'ensemble des événements générés par chaque attaque. AE_{ij} représente le nombre des événements de type i généré par le scénario d'attaque j où :

$$AE_{ij} \geq 0 \qquad (3.1)$$

- R: est un vecteur de taille N_a il représente le vecteur des poids des attaques où:

$$(R_i > 0) \qquad (3.2)$$

R_i représente le poids associé à l'attaque i. Ce poids R_i représente le risque potentiel engendré par l'attaque I.

- O: est un vecteur de taille N_e où :

O_i Comptabilise le nombre d'occurrences des événements de type i dans les traces d'audit (O est le vecteur d'observation).

- H : est de taille N_a, il représente le vecteur d'hypothèse où:

$$H_i = 1 \qquad (3.3)$$

Signifier que l'attaque i est supposée présente dans l'audit de sécurité.

$$H_i = 0 \qquad (3.4)$$

Signifier que l'attaque i est supposée non présente dans l'audit de sécurité.

L'analyse du fichier d'audit de sécurité consiste à déterminer la matrice H maximisant le produit matriciel $R \times H$, tout en respectant les contraintes:

$$(AE \times H)_i \leq O_i, (1 \leq i \leq N_a) \qquad (3.5)$$

PASFAS est un problème NP-complet. Le temps de traitement nécessaire pour résoudre PASFAS ne permet pas d'envisager une solution algorithmique non polynomiale quant N_a atteindre quelque centaine. C'est pourquoi il faut envisager l'utilisation d'une méthode heuristique telle que les algorithmes génétiques (Mé, 1995). L'approche heuristique choisie par Ludovic Mé (Mé, 1995) pour résoudre ce problème NP-complet est la suivante: une hypothèse est faite (par exemple, parmi l'ensemble des attaques possibles, les attaques i, j et k sont supposées présentes dans l'audite de sécurité), la faisabilité de l'hypothèse est évaluée selon cette évaluation, une hypothèse améliorée est proposée, jusqu'à ce qu'une solution faisable soit trouvée.

Afin d'évaluer les hypothèses correspondantes à un sous-ensemble des attaques présentes, nous comptons le nombre d'occurrences des événements générés par toutes les attaques de cette hypothèse (Vecteur H). Si ce nombre d'occurrences est moins que ou égal au nombre d'enregistrements des traces d'audite alors cette hypothèse est réalisable. Sinon elle est non réalisable.

Après la formulation du problème, nous devons trouver un algorithme permettant de dériver une nouvelle hypothèse basée sur l'hypothèse passé. L'algorithme génétique est l'un des algorithmes les plus efficaces pour réaliser cette tâche.

3.2.3.2. La résolution génétique de PASFAS (Mé, 1995)

Les algorithmes génétiques (GA) sont des algorithmes de recherche optimaux basés sur le mécanisme de la sélection naturelle dans une population. Une population est un ensemble d'individus ou de chromosomes artificiels. Ces individus sont des chaînes de longueur I qui codent une solution potentielle d'un problème à résoudre, le plus souvent en utilisant un alphabet binaire. La taille L de la population est constante. La population n'est rien d'autre qu'un ensemble de points dans un espace de recherche. La population est générée aléatoirement, puis évolue à chaque génération, une nouvelle série des individus artificiels

est créée en utilisant les plus forts individus ou les morceaux des individus les plus forts de la précédente génération.

L'aptitude de chaque individu est tout simplement la valeur de la fonction à optimiser (la fonction sélective). Le processus itératif de création d'une nouvelle population est réalisé par les trois opérateurs génétiques de base: la sélection (pour sélectionner les individus les plus forts), la reproduction ou croisement (favorise l'exploration des nouvelles régions de l'espace de recherche) et la mutation (protège la population contre la perte irrémédiable de l'information). Deux défis se posent lors de l'application des algorithmes génétiques à un problème particulier. Le premier est le codage d'une solution adapté à un problème avec une chaîne de bits. Le deuxième est de trouver une fonction sélective pour évaluer chaque individu de la population.

3.2.3.2.1. Le codage de la solution

Un individu est une chaîne de longueur *l* codant une solution potentielle pour le problème à résoudre. Dans notre cas, le codage est un codage binaire simple où chaque individu de la population correspond à un vecteur particulier H, la longueur d'un individu est notée N_a.

3.2.3.2.2. La fonction sélective

Nous devons rechercher parmi tous les sous-ensembles des attaques possibles, celui qui présente le plus grand risque pour le système. Cela se traduit par la maximisation du produit R × H. Vu que les algorithmes génétiques sont des algorithmes de recherche optimale, le faite de trouver le maximum d'une fonction sélective peut être traduit facilement par le produit R × H. Par conséquent nous avons :

$$F = \sum_{I=1}^{Na} R_i \, I_i \qquad (3.6)$$

Où I est un individu.

Cette fonction sélective ne tient pas compte de la contrainte de notre problème qui suppose que certains individus au sein du 2^{N_a} cas possible ne sont pas réalistes. C'est le cas pour un certain type i des événements lorsque:

$$(AE \times H)_i > O_i \qquad (3.7)$$

Comme un grand nombre des individus ne respectent pas la contrainte, Ludovic Mé (Mé, 1995) a décidé de les pénaliser en réduisant leurs valeurs de fitness. Donc, nous calculons une fonction de pénalité (P), où T_e est le nombre de types des événements pour lesquels : $(AE \times H)_i > O_i$

La fonction de pénalité appliquée à un individu H est:

$$P = T_e^p \qquad (3.8)$$

Une fonction de pénalité quadratique (soit p = 2) permet une bonne discrimination des individus qui violent les contraintes. La fonction sélective proposée par Ludovic Mé (Mé, 1995) est la suivante:

$$F(I) = \alpha + \left(\sum_{i=1}^{Na} R_i\, I_i - \beta . T_e^p\right) \qquad (3.9)$$

Le paramètre β permet de modifier la pente de la fonction de pénalité et α fixe un seuil en rendant la formule positive.

3.2.4. Notre apport pour la résolution génétique de PASFAS

Nous avons proposé deux optimisations pour la résolution génétique de PASFAS. La première a été l'objet de notre participation à la conférence MISC 2010 avec notre article «Intrusion Detection by optimized GASSATA » (AHMIM et al, 2010). La deuxième représente la version distribuée de notre première proposition, cette deuxième amélioration a été l'objet de notre deuxième communication intitulé «Improved Off-Line Intrusion Detection Using a Genetic Algorithm and RMI » (AHMIM et al, 2011).

3.2.4.1. Notre première optimisation de la résolution génétique de PASFAS (AHMIM et al, 2010)

Notre proposition est inspirée du travail de Ludovic Mé (Mé, 1995), où nous avons utilisé les mêmes spécifications en termes de codage et de fonction sélective. Notre contribution consiste à éliminer les attaques certainement existantes ainsi que les attaques certainement non existantes. Nous nous intéressons aux attaques auxquelles on n'est pas sûr de leur existence qui représente le vrai problème (le réel PASFAS), puis on divise ce réel PASFAS en sous-problèmes indépendants plus faciles à résoudre. Enfin, on applique l'algorithme génétique avec l'opérateur de croisement proposé dans (Mahmoudi and Ghoualmi, 2010).

3.2.4.1.1. La filtration des attaques

Dans cette étape de filtration nous utilisons la matrice d'observation "O" et la matrice attaque-événement "AE". Si on réduit l'espace de recherche, le temps de traitement sera plus court et la probabilité d'obtenir la solution optimale sera plus élevée. Par conséquent, nous éliminons les attaques certainement existantes et certainement non existantes, puis nous divisons le reste des attaques qui représentent le vrai problème en sous-problèmes indépendants.

3.2.4.1.1.1. L'élimination des attaques certainement non existantes

Nous éliminons les attaques qui ont une probabilité d'existence égale à 0%. Ces attaques ont un nombre d'occurrences pour l'un des événements plus grands que le nombre d'occurrences audités pour cet événement. Donc, nous éliminons toute attaque i qui vérifie la formule suivante:

$$\exists\, j \in N_e\ (AE_{ij} > O_j) \qquad (3.10)$$

Pour éliminer ces attaques, nous comparons la matrice attaque-événement "AE" avec la matrice d'observation "O". Suite à cette opération, les matrices auront un nombre d'attaques noté N_{ap}. Après avoir éliminé ces attaques, nous éliminons les événements qui ont une valeur égale à 0 dans la matrice d'observation "O". Le nombre d'événements utilisés pour les attaques sélectionnées est noté N_{ep}.

Les résultats de cette première étape sont les matrices avec les dimensions suivantes: $AE_{(N_{ap},N_{ep})}$, $O_{(N_{ep})}$, $H_{(N_{ap})}$, $R_{(N_{ap})}$.

La complexité de cette première étape est: $\theta(N_a \times N_e)$.

3.2.4.1.1.2. L'élimination des attaques certainement existantes

Nous éliminons les attaques qui ont une possibilité d'existence égale à 100%. Ces attaques n'ont pas un événement commun avec d'autres attaques ou dans le ce cas d'existence, la somme de leur nombre d'occurrences est inférieur ou égal au nombre d'occurrences auditées pour cet événement. Pour éliminer ces attaques, nous comparons la matrice $AE_{(N_{ap},N_{ep})}$ avec la matrice $O_{(N_{ep})}$ où nous éliminons toute attaque i qui satisfait la formule suivante:

$$\forall j \in N_e \left((AE_{ij} > 0) \rightarrow \left(\left(\sum_{i=0}^{i=N_a} AE_{ij} \right) \leq O_j \right) \right) \qquad (3.11)$$

À la fin de cette opération, le nombre des attaques sera N_{ha}. Par conséquent, nous redimensionnons la matrice "AE" elle devient d'une taille (N_{ha}, N_{ep}). Après cette étape, nous éliminons les événements j qui vérifient la formule suivante :

$$\left(\sum_{i=0}^{i=N_a} AE_{ij} \right) \leq O_j \qquad (3.12)$$

Le nombre des événements retenus est noté N_{he}.

Les résultats sont les matrices avec les dimensions suivantes: $AE_{(N_{ha},N_{he})}$, $O_{(N_{he})}$, $H_{(N_{ha})}$, $R_{(N_{ha})}$.

La complexité de la deuxième étape est: $\theta(N_{ap} \times N_{ep})$.

3.2.4.1.1.3. La division de PASFAS en sous-PASFASs

Pour effectuer cette étape, nous utilisons le reste des attaques auxquelles nous avons des doutes pour leurs présences. Cette étape consiste à regrouper les attaques qui ont un événement en commun avec des autres attaques (> 0) où la somme du nombre d'occurrences dépasse le nombre d'occurrences des événements audités. Chaque groupe des attaques est associé à un groupe des événements dont ils ont un certain nombre d'occurrences plus élevé que le nombre des occurrences auditées. Après cette étape, nous créons les Sous-PASFASs où chaque sous-$PASFAS_i$ contient les attaques du groupe I avec les événements associés. Le pseudo algorithme ci-dessous décrit le processus de regroupement.

```
Procedure grouping

Begin
for i=1 to N_ha  do
     if (not-marked attack(i)) then
                                   Create group() ;
                                   Mark(i) ;
                                   add-element(i) ;
end.


Procedure add-element(i)

begin
for  j = 1 to N_ep do
   if AE_ij ≠ 0 then
              for x = i + 1 to N_ha do
                 if  AE_xj ≠ 0 then
                    if (not-marked attack (x)) then
                                                add-to-group(x);
                                               Mark(i);
                                               add-element(x);
                    else  fusion-group-where-belong(x,i);
end .
```

La complexité de cette première étape est: $\theta(N_{ha} \times N_{he})$

3.2.4.1.2. Le croisement

Le croisement utilisé est un croisement fortement aléatoire (Mahmoudi and Ghoualmi, 2010). Toutes les possibilités d'héritage sont faisables dès la première génération, ce qui réduit le nombre de générations nécessaires pour trouver la solution optimale. Ce croisement consiste à faire un clonage de l'un des deux parents. Puis, hériter aléatoirement les gènes du second parent, et les posés dans les locus correspondants dans le parent cloné comme le montre la figure 9.

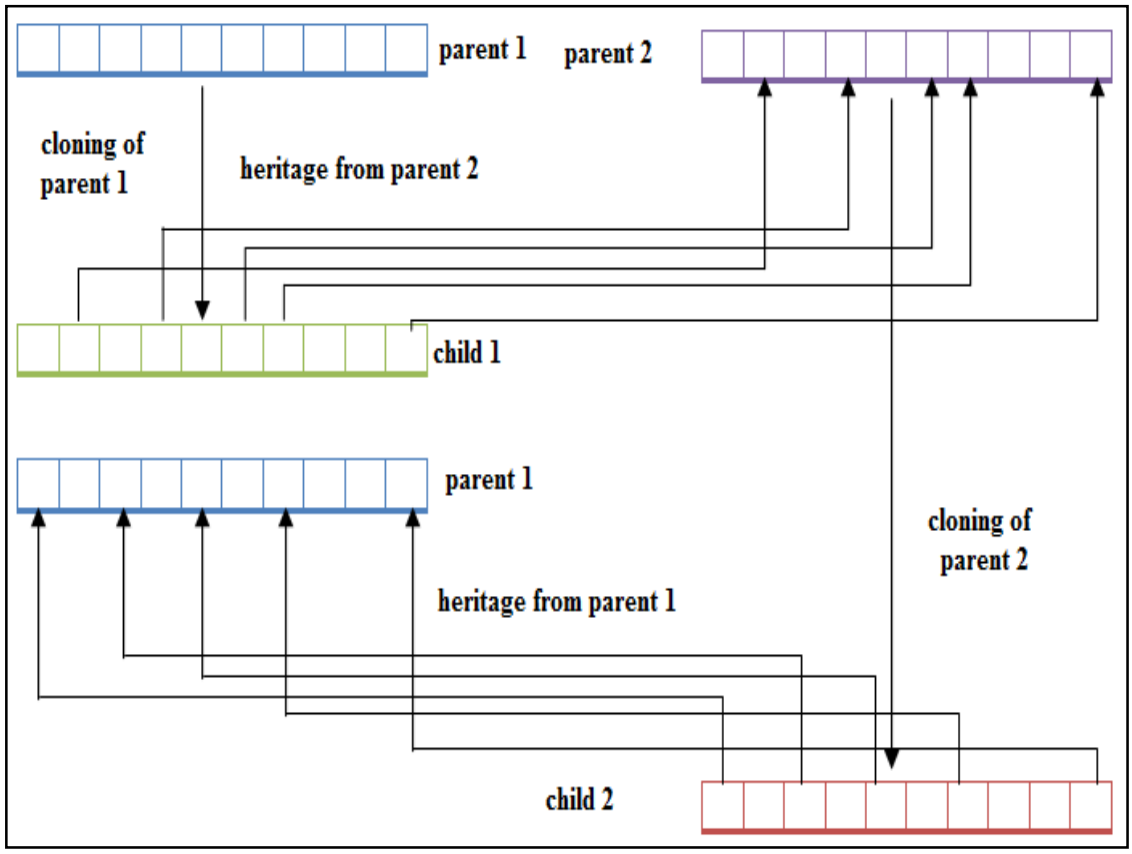

Figure 9 Le croisement utilisé dans notre proposition

3.2.4.2. La version distribuée de notre proposition (AHMIM et al, 2011)

Cette étape consiste à résoudre les sous-PASFASs simultanément en utilisant l'invocation de méthode à distance (Remote Method Invocation). Afin d'implémenter cette proposition, notre réseau doit avoir deux types d'unité (une seule unité maitre et plusieurs unités esclaves). Comme le montre la figure 10, l'unité maitre effectuer la tache de filtration des attaques, ensuite elle devise le réel PASFAS en plusieurs Sous-PASFASs puis d'après les performances des unités esclaves qui implémentent la résolution génétique de PASFAS, elle associe à chacune d'entre elles l'un des sous-PASFASs à résoudre d'après leurs performances où la meilleure unité en terme de performance résoudre le plus grand sous-PASFAS.

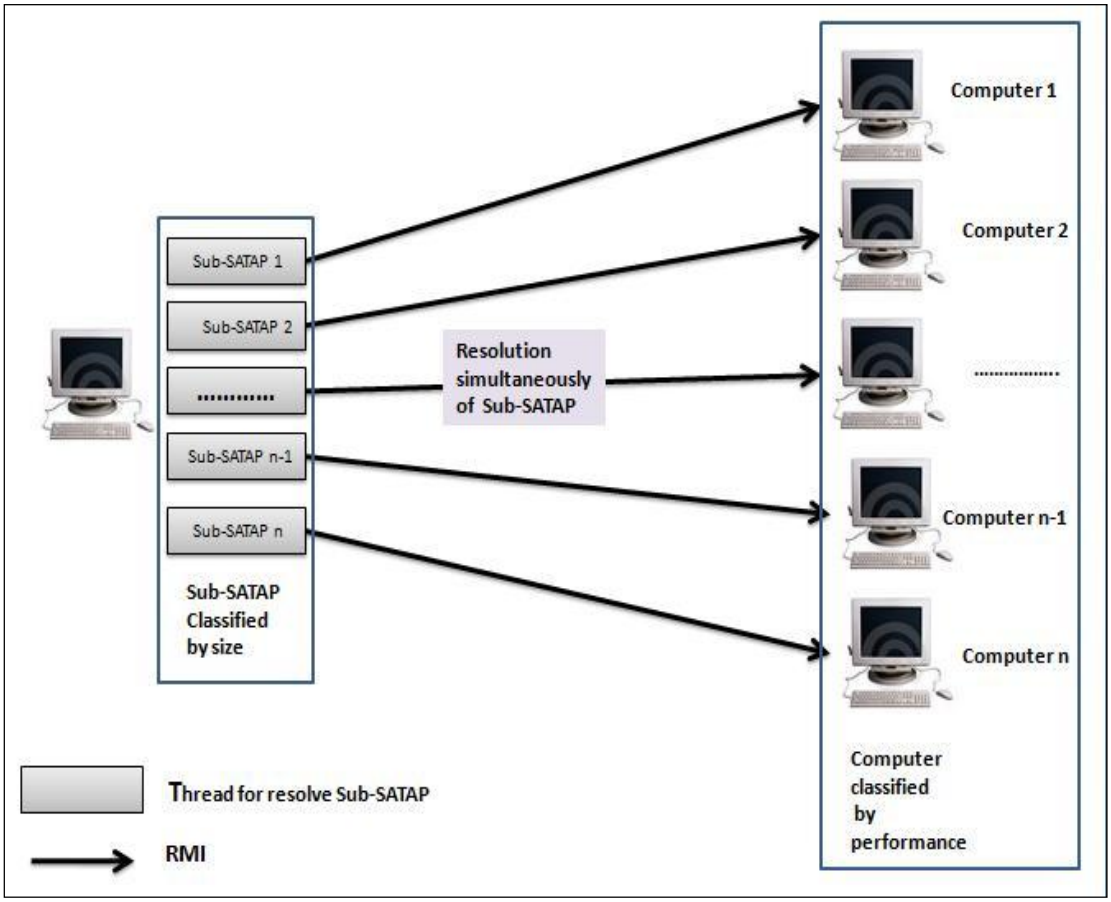

Figure 10 Le mécanisme de la résolution simultanée des sous-PASFASs (AHMIM et al, 2011)

3.2.4.2.1. La comparaison entre notre proposition (AHMIM et al, 2010), (AHMIM et al, 2011) et celle de Ludovic Mé (Mé, 1995)

Afin de montrer les avantages de notre proposition, nous comparons nos résultats avec celles du travail de Mé (Mé, 1995) en utilisant les mêmes benchmarks. Les métriques suivantes sont utilisées dans la comparaison: le nombre d'attaques détectées, le nombre de contraintes violées pour chaque génération, la vitesse de convergence vers la meilleure solution, le nombre de générations et le temps nécessaire pour la résolution.

Les résultats illustrés dans la figure 11 et la figure 12 ainsi que le tableau 1 montre que nos travaux et celui de Mé ont détecté le même nombre d'attaques qui représentent les attaques réelles. Cependant, il existe quelques différences qu'on peut les résumer dans les points suivants:

- La durée: le temps de traitement de notre première proposition est inférieur à celui de Mé (AHMIM et al, 2010) en raison de la réduction de la taille du problème. La version distribuée de notre modèle a encore optimisé le temps de traitement grâce au traitement simultané des sous-PASFASs.

- Le nombre de générations: le nombre de générations nécessaires pour résoudre PASFAS dans notre modèle est inférieur au nombre de générations dans le modèle de Mé (Mé, 1995), car la taille de PASFAS filtré ou le plus grand sous-PASFAS à traiter est inférieur ou égal (dans le pire des cas) à la taille de PASFAS.
- La vitesse de convergence: la vitesse de convergence de notre proposition est plus rapide que Mé (Mé, 1995), car la taille de PASFAS filtré ou le plus grand sous-PASFAS à traiter est inférieure ou égale (dans le pire des cas) à la taille de PASFAS.
- La violation des contraintes : en raison de l'opération de filtrage et la division de PASFAS en sous-PASFAS, la violation des contraintes de notre contribution est moins que (quasi inexistante) celle de Mé (Mé, 1995).

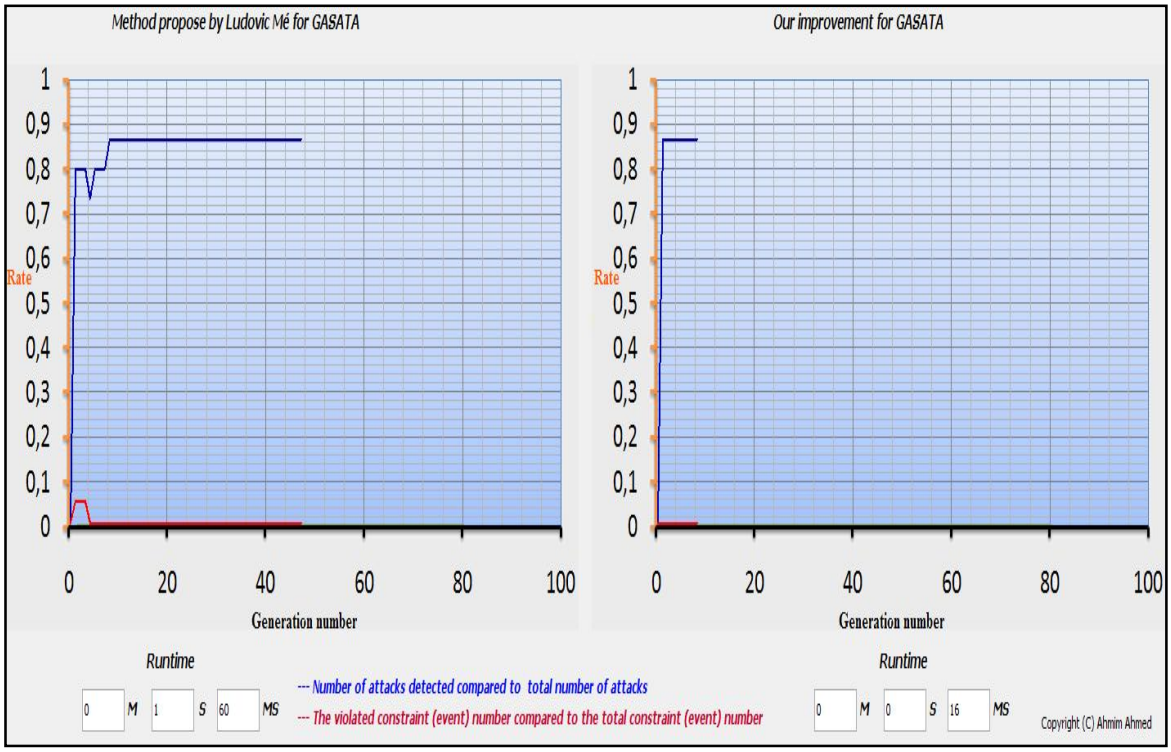

Figure 11 La comparaison entre notre proposition et celle de Mé pour le benchmark (15, 19)

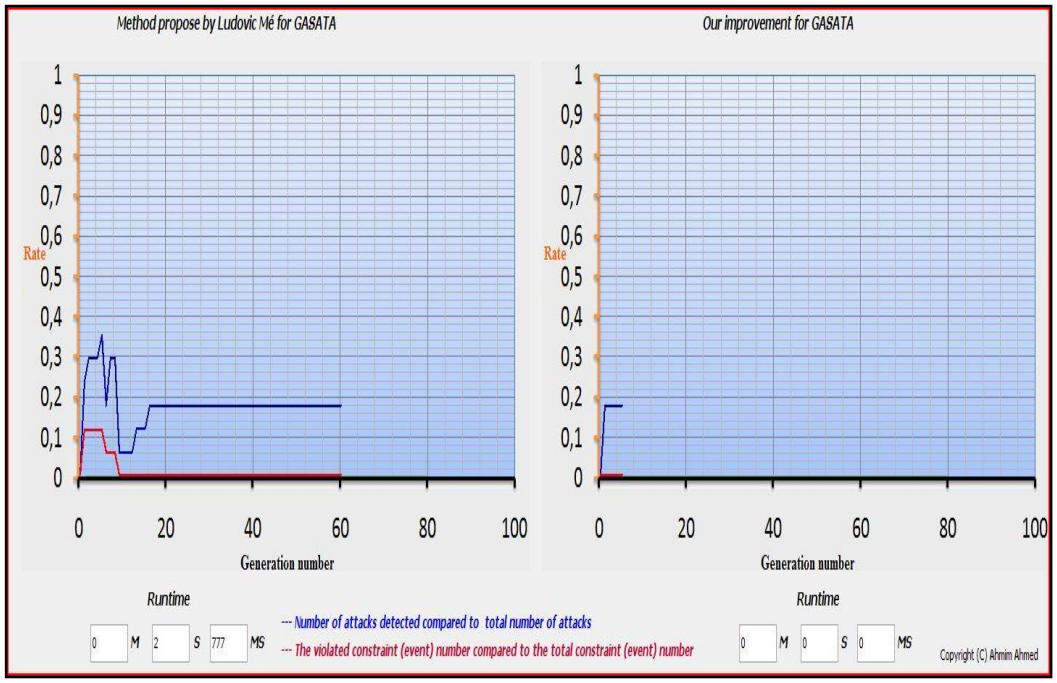

Figure 12 La comparaison entre notre proposition et celle de Mé pour le benchmark (25, 51)

	Résolution classique	Notre proposition	
		Sans RMI	Avec RMI
benchmark(2,4)	≈16 ms	≈0 ms	≈0 ms
benchmark(5,9)	≈62 ms	≈0 ms	≈0 ms
benchmark(6,9)	≈125 ms	≈7 ms	≈7 ms
benchmark(9,11)	≈234 ms	≈0 ms	≈0 ms
benchmark(10,12)	≈312 ms	≈0 ms	≈0 ms
benchmark(15,19)	≈1s 60 ms	≈16 ms	≈13 ms
benchmark(15,20)	≈1s 139 ms	≈0 ms	≈0 ms
benchmark(17,35)	≈2s 777 ms	≈0 ms	≈0 ms
benchmark(21,40)	≈4s 771 ms	≈0 ms	≈0 ms
benchmark(24,50)	≈7s 909 ms	≈0 ms	≈0 ms
benchmark(25,51)	≈8s 502 ms	≈265 ms	≈232 ms

Tableau 1 La comparaison entre notre proposition et celle de Mé (Mé, 1995) pour la résolution de PASFAS

3.2.4.3. Conclusion

L'utilisation des algorithmes génétiques pour la résolution du problème de l'analyse simplifié des fichiers d'audit de sécurité a donné des bons résultats. Notre contribution consiste à classer les attaques l'audite de sécurité en trois classes et diviser la 3e classe en sous-problèmes. Ensuite, appliquer l'algorithme génétique avec les mêmes spécifications de Ludovic Mé (Mé, 1995). Concernant l'opérateur de croisement, nous avons utilisé le croisement fortement aléatoire proposé par Mahmoudi et Ghoualmi (2010). La seconde contribution consiste à optimiser le temps de résolution génétique. Pour cette raison, nous avons utilisé RMI (Remote Method Invocation) pour résoudre tous les sous-problèmes en même temps.

Notre contribution a apporté les avantages suivants:

- 0% faux +.
- 0% faux -.
- Le taux de détection est 100%.
- La minimisation du temps d'exécution.
- L'augmentation de la vitesse de convergence.
- La réduction de la violation des contraintes.
- La réduction du nombre des générations nécessaire pour la résolution du problème.

3.2.5. Les limites de la première génération des systèmes de détection d'intrusion

Malgré la diversité des techniques utilisées pour les systèmes de détection d'intrusion de la première génération pour l'approche comportementale ou par scénario, cette première génération a montré plusieurs limites face aux problèmes suivants (Bensefia, 2009) :

- **La conception manuelle basée sur les connaissances des experts humains.** Pour toutes les techniques utilisées dans les systèmes de détection d'intrusion de la première génération, l'intervention de l'expert humain est indispensable dans les différentes étapes du développement de ces systèmes de détection d'intrusion. Le rôle de cet expert consiste généralement à spécifier les informations à auditer, créer la base de signature, établir le profil du comportement normal, spécifier l'encodage utilisé, suivre et mettre à jour les systèmes de détection d'intrusion. L'intervention des experts humains dans les différentes étapes de développement des systèmes de détection d'intrusion rend ces systèmes très associés aux experts

et leurs performances dépendent de la compréhension des experts du système d'information et des attaques.

- **L'étroite dépendance de l'environnement cible.** Vu que les systèmes de détection d'intrusion de cette première génération sont développés en étroite relation avec l'environnement cible, et que pendant ce développement on utilise généralement les spécificités de cet environnement cible rend la portabilité de ces systèmes de détection d'intrusion même pour des environnements similaire très difficile voire impossible.
- **La difficulté liée à l'évaluation.** Pour évaluer les performances des systèmes de détection d'intrusion de cette première génération chaque système utilise ses propres données d'évaluation, son propre format. Ce manque d'une base commune d'évaluation, d'un format standard pour présenter les traces d'audit représentent un obstacle qui nous empêche de faire une vraie évaluation des différentes méthodes et de les comparer afin de voir leurs avantages et leurs inconvénients.
- **Les limites de performance.** Face à la grande croissance du trafic réseau en volume et en capacité, ainsi que la variété des attaques qui ne cessent pas de changer, les systèmes de détection d'intrusion de la première génération ont montré beaucoup de limites. Ces limites sont engendrées par l'incapacité de traiter un large volume de données d'audite, la difficulté et le coût très élevé de l'analyse d'une signature d'une attaque dans des larges données d'audites.

3.3. Les systèmes de détection d'intrusion basés sur les techniques de data mining

Pour faire face à la grande quantité de données dans les cyber infrastructures, le nombre très important de cybercriminels qui ne cessent pas de tenter d'accéder aux systèmes et aux données et les différentes méthodes et techniques de piratage qui n'arrêtent pas d'évoluer et de varier les chercheurs en sécurité informatique ont utilisé les différents techniques d'apprentissage automatique, de statistique, et de data mining, afin de relever les défis de la cyber sécurité.

3.3.1. Le data mining

Le data mining est l'extraction de la connaissance à partir d'une grande quantité de données. Les modèles ou les règles détectées par les techniques de data mining peuvent être utilisés

pour la prédiction non triviale des nouvelles données. En prédiction non triviale, l'information est implicitement présente dans les données, mais auparavant inconnu est découverte. Les techniques de data mining utilisent les statistiques, l'intelligence artificielle et la reconnaissance des formes des données dans le but de regrouper ou extraire des comportements ou des entités. Donc le data mining est un domaine interdisciplinaire qui se base sur l'utilisation des outils d'analyse des modèles statistiques, des algorithmes mathématiques et des méthodes d'apprentissage automatique afin de découvrir des modèles valides et des relations précédemment inconnues dans un grand ensemble de données. Ces modèles et ces relations découvertes sont très utiles pour créer des mécanismes de sécurité capable de trouver toute violation de la politique de sécurité et de préserver de la vie privée des utilisateurs des systèmes informatiques.

Le data mining est utilisé dans de nombreux domaines comme la finance, l'ingénierie, la biomédecine et le cyber sécurité. Il existe deux catégories de méthodes de data mining: supervisées et non supervisées. Les techniques de data mining supervisées prédisent une fonction cachée en utilisant les données d'apprentissage. Les données d'apprentissage sont des paires de variables d'entrée/sortie (des étiquettes ou des classes). La sortie de la méthode prévoit l'étiquette de la classe des variables d'entrée. La classification et la prédiction sont des exemples de data mining supervisé. Le data mining non supervisée est l'identification des modèles cachés des données sans l'introduction de données d'apprentissage (c.-à-d. les paires entrées et étiquettes des classes). Le regroupement (clustering) et les règles associatives (associative rule mining) sont des exemples typiques de data mining non supervisé. Le data mining est également une partie intégrante de la découverte de connaissances dans les bases de données (KDD) qui représente un processus itératif non trivial de l'extraction de l'information à partir des données. Le KDD comprend plusieurs étapes de la collecte des données brutes à la création des nouvelles connaissances. Le processus itératif comprend les étapes suivantes: nettoyage des données, l'intégration des données, la sélection des données, la transformation des données, l'extraction des données (data mining), l'évaluation du modèle, et la représentation des connaissances.

Étape 1. Pendant le nettoyage des données, le bruit et les données non pertinentes sont supprimés de la collection.

Étape 2. L'intégration des données consiste à combiner les données provenant des sources multiples et hétérogènes dans une base de données.

Étape 3. Les techniques de sélection des données permettent à l'utilisateur d'obtenir une représentation réduite de l'ensemble des données afin de maintenir l'intégrité de l'ensemble des données d'origine dans un volume réduit.

Étape 4. Dans la transformation des données, les données sélectionnées sont transformées en un format souhaitable.

Étape 5. Le data mining est l'étape dans laquelle les outils d'analyse sont appliqués pour découvrir des modèles qui pourraient être utiles.

Étape 6. L'évaluation du modèle consiste à identifier des modèles intéressants et utiles en utilisant des mesures de validation des données.

Étape 7. La représentation des connaissances est la phase finale du processus de découverte des connaissances où le savoir découvert est présenté aux utilisateurs dans des formes visuelles.

Les techniques de data mining sont utilisés pour aider à élaborer des modèles prédictifs qui permettent une réponse en temps réel après une séquence de processus qui comprennent l'échantillonnage des données en temps réel, la sélection, l'analyse et la recherche, et le data mining qui sert à classer et détecter les attaques et les intrusions sur un réseau informatique (Dua and Xian , 2011).

3.3.2. Les défis de data mining

Les défis de data mining en terme de cyber sécurité sont classés en quatre domaines d'application qui sont : la modélisation des réseaux à grande échelle, la découverte de l'intrusion, le dynamisme du réseau et les cyberattaques, et la préservation de la vie privée (Dua and Xian, 2011).

3.3.2.1. La modélisation des réseaux à grande échelle

La modélisation d'un cyber infrastructure est difficile, car de nombreuses mesures des graphes communs sont difficiles à calculer pour des réseaux sous-jacents. Il est difficile de construire le modèle explicatif des réseaux en raison des exigences en matière de précision d'apprentissage et de prédiction: des réseaux réalistes à différentes échelles sont simulés pour tester des algorithmes pour la défense, des anomalies qui ne sont pas conformes au modèle, et potentiellement une intrusion ou d'autres problèmes de réseau sont détectés.

Un modèle de réseau peut être extrait en partie et avec attention pour l'analyse avancée, et un réseau peut être construit dans un monde réel de façon significative, mais il ne peut pas suivre l'hypothèse de variables aléatoires. De plus, il y a les difficultés résidantes dans le calcul des mesures graphiques du modèle de réseau. Des exemples de ces modèles graphiques ont la dynamique du réseau de télécommunications, des réseaux de communications électroniques de courrier par lequel les virus propagent, ou du réseau de liens hypertextes entre les sites Web. Le diamètre de graphe, la distance maximale entre deux nœuds dans un graphe sont des exemples d'une mesure graphique. Les difficultés de calcul nous pousse à appeler les modèles de data mining qui peuvent découvrir la nature des données réelles en utilisant un modèle plus simple.

3.3.2.2. La découverte des menaces

L'utilisation de data mining dans des cybers infrastructure pour la découverte de menaces souffre du volume et des données hétérogènes du réseau, le changement dynamique des menaces et les graves déséquilibres des classes de comportements normaux et anormaux. Ces défis nous poussent à appeler des méthodes qui peuvent agréger les informations des réseaux dynamiquement et localement pour détecter les attaques complexes en plusieurs étapes et prévoir les menaces potentielles et rares sur la base de l'analyse du comportement des données et des événements du réseau. Les méthodes les plus employées pour détecter le code ou le comportement malveillant sont les modèles à base de règles ou statistiques pour identifier les menaces en temps réel en utilisant la détection adaptative de la menace avec la modélisation des données temporelles et les données manquantes. L'échantillonnage des données sur le réseau à grande échelle doit être adaptatif aux incertitudes de l'évolution physique des réseaux, des codes malveillants et des comportements malveillants. La modélisation adaptative et dynamique est nécessaire pour l'évolution temporelle de la structure et les caractéristiques des données.

3.3.2.3. Le dynamisme du réseau et les cyberattaques

Beaucoup de cyber attaques propagent des programmes malveillants pour les ordinateurs vulnérables. En raison des conditions de déclenchement inconnu des logiciels malveillants, ces logiciels malveillants peuvent infecter les ordinateurs dans un réseau à des degrés divers. Une fois que l'administrateur réseau détecte les logiciels malveillants, la propagation des infections des logiciels malveillants doit être étudiée pour construire un système de protection. Les nouvelles méthodes de data mining sont nécessaires pour prévoir les futures

attaques en se basant sur l'évolution des logiciels malveillants. Cependant, la structure détaillée du réseau est inconnu, ce qui limite les connaissances de l'évolution de l'infection.

3.3.2.4. La préservation de la vie privée en data mining

L'extraction de données peut être utilisée avec malveillance dans les cybers infrastructure pour violer la vie privée. En principe, plus les données complètes sont disponibles pour l'exploration des données, plus le résultat obtenu sera précis. Toutefois, les données complètes et précises peuvent également soulever des questions d'atteinte à la vie privée. En outre, le résultat d'exploration des données peut potentiellement révéler des informations privées. Le concept de la préservation de la vie privée dans le data mining (Privacy preserving data mining PPDM) protège les données privées d'être volées ou mal utilisées par des utilisateurs malveillants, tout en permettant d'extraire les données pour être utilisées (Dua and Xian, 2011).

3.3.3. Les techniques de data mining utilisées pour la détection d'intrusion

L'utilisation des techniques de data mining pour la détection d'intrusion a passé par différents chemins où on trouve l'utilisation des classificateurs simple, hybride et ensemble (Tsaia et al, 2009).

3.3.3.1. Les classificateurs simples

Le problème de détection d'intrusion peut être abordé en utilisant une seule technique de data mining. Dans la littérature, il existe deux catégories de techniques d'apprentissage automatique qui ont été utilisés dans la détection d'intrusion. La première catégorie représente les techniques d'auto apprentissage supervisé et la deuxième représente les techniques d'auto apprentissage non supervisé (Tsaia et al, 2009).

3.3.3.1.1. Les techniques d'apprentissage supervisé

Dans l'apprentissage supervisé, on fournit à un algorithme d'apprentissage un ensemble de données complètement étiqueté. L'algorithme utilise les échantillons étiquetés pour la formation afin de créer un modèle. Ensuite, le modèle d'apprentissage automatique qualifié étiquète les données qui n'ont jamais été utilisées par l'algorithme. L'objectif est d'aider l'algorithme d'apprentissage automatique supervisé à obtenir la précision de la classification la plus élevée. Les méthodes les plus populaires dans l'apprentissage automatique supervisé comprennent le réseau de neurones artificiels (ANN), la machine à vecteurs de support (SVM), les arbres de décision, les réseaux bayésiens (BNS), les K plus proches voisins (KNN), et le modèle de Markov caché (HMM) (Tsaia et al, 2009).

3.3.3.1.1.1. La machine à vecteurs de support

La machine à vecteurs de support (SVM) a été proposée par Vapnik en 1998 (Vapnik, 1997). SVM comme le montre la figure 13 trace un vecteur d'entrée dans un espace de caractéristiques de grande dimension, afin d'obtenir l'hyperplan optimal de séparation pour cet espace. De plus, une limite de décision est déterminée par des vecteurs de soutien plutôt que des échantillons d'apprentissage entiers, par conséquent la machine à vecteurs de support est extrêmement robuste aux valeurs extrêmes. En particulier, SVM est conçu pour la classification binaire afin de séparer un ensemble de vecteurs d'apprentissage qui appartiennent à deux classes différentes. Il faut noter que les vecteurs de support sont des échantillons d'apprentissage à proximité d'une frontière de décision. Le SVM fournit également un paramètre spécifié d'utilisation appelé facteur de pénalité. Ce paramètre permet aux utilisateurs de faire un compromis entre le nombre des échantillons mal classés et la largeur d'une frontière de décision (Tsaia et al, 2009).

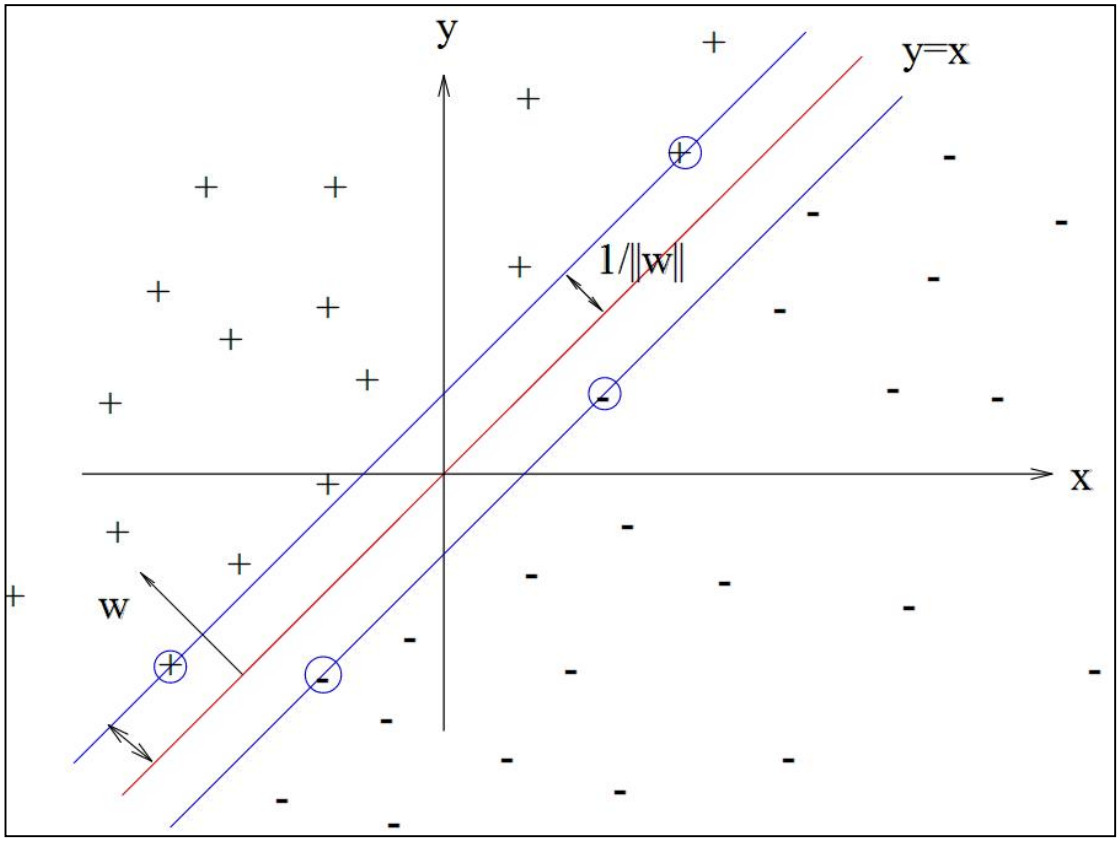

Figure 13 La machine à vecteurs de support

3.3.3.1.1.2. Le réseau de neurones artificiels

Le réseau de neurones est un ensemble des unités de traitement de l'information qui a pour but d'imiter les neurones du cerveau humain (Haykin, 1999). Le Perceptron multicouche (MLP) est l'architecture de réseau de neurones la plus utilisée dans de nombreux problèmes de reconnaissance des formes. Comme le montre la figure 14 un réseau MLP se compose

d'une couche d'entrée qui contient un ensemble de nœuds sensoriels comme des nœuds d'entrées, une ou plusieurs couches cachées de nœuds de calcul, et une couche de sortie de nœuds de calcul. Chaque interconnexion est associée à une pondération scalaire qui est ajustée pendant la phase d'apprentissage. L'algorithme d'apprentissage de rétropropagation est généralement utilisé pour former un MLP. Au début, des poids aléatoires sont attribués. Ensuite, l'algorithme de rétropropagation effectue le réglage des poids, pour définir laquelle des représentations des unités cachées est plus efficace pour minimiser l'erreur de classification erronée (Tsaia et al, 2009).

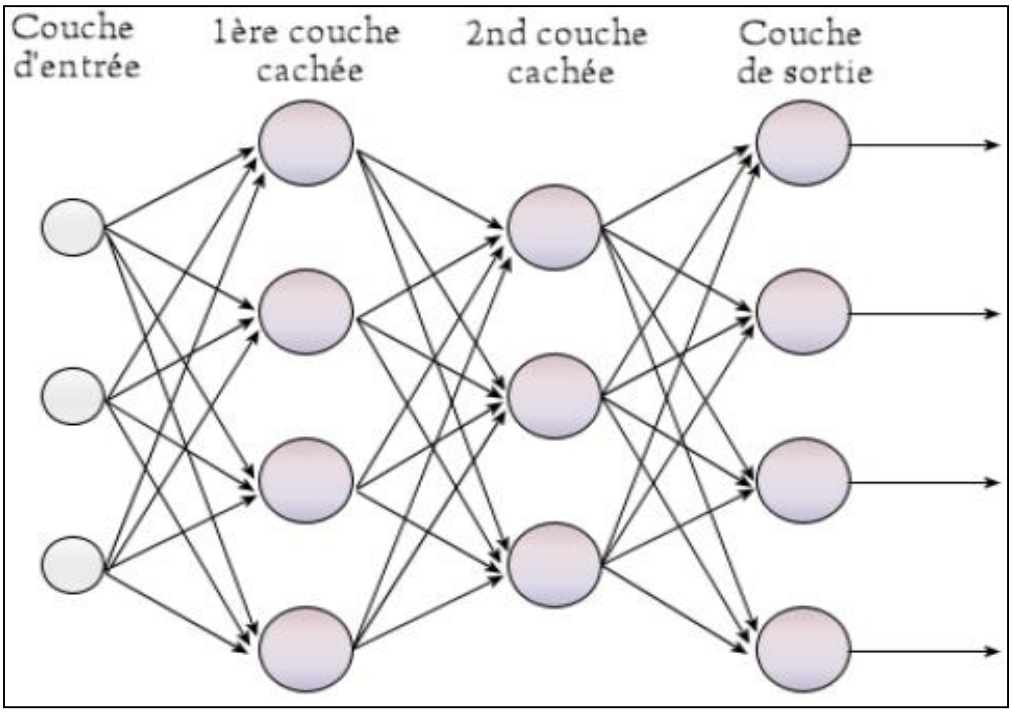

Figure 14 Le perceptron multicouche

3.3.3.1.1.3. Le Naïve Bayes

Il existe de nombreux cas où nous connaissons les dépendances statistiques ou les relations causales entre les variables du système. Toutefois, il pourrait être difficile d'exprimer avec précision les relations probabilistes entre ces variables. En d'autres termes, la connaissance préalable du système est tout simplement représentée par le fait que certaines variables peuvent influencer les autres. Pour exploiter cette relation structurelle ou cette causale dépendance entre les variables aléatoires d'un problème, on peut utiliser un modèle de graphe probabiliste appelé réseau Bayésien Naïve (NB). Le modèle offre une réponse à des questions comme "Qu'elle est la probabilité de l'existence d'un certain type d'attaque, compte tenu de certains événements observés dans le système?" En utilisant la formule de probabilité conditionnelle. Comme le montre la figure 15 suivante, la structure d'un NB est

généralement représentée par un graphe acyclique (DAG), où chaque nœud représente l'une des variables du système et chaque lien encode l'influence d'un nœud sur un autre (Pearl, 1988). Donc, s'il y a un lien entre le nœud A et le nœud B, alors A influé directement B (Tsaia et al, 2009).

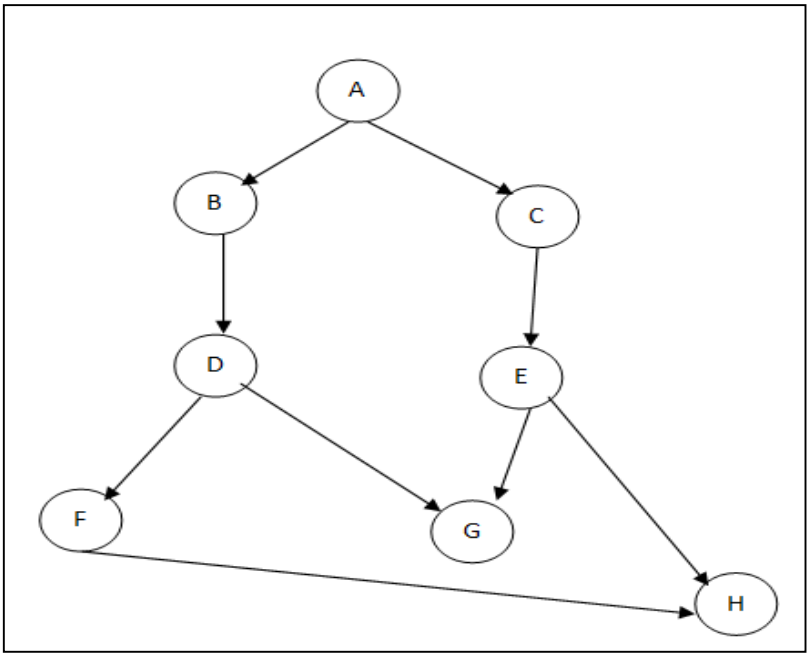

Figure 15 réseau bayésien naïve

3.3.3.1.1.4. L'arbre de décision

Un arbre de décision classifie un échantillon à travers une série de décisions, dont la décision actuelle contribue à la décision ultérieure. Comme l'illustre la figure 16, la série de décisions est représentée sous forme d'une structure arborescente. La classification de l'échantillon se fait à partir du nœud racine à un souhaitable nœud feuille, où chaque nœud feuille représente une catégorie de classification. Les attributs des échantillons sont assignés à chaque nœud, et la valeur de chaque branche est correspondante aux attributs (Mitchell, 1997). CART (Classification And Regression Trees) est un programme bien connu pour la construction des arbres de décision (Breiman et al, 1984). Un arbre de décision avec des étiquettes discrètes de classe (symbolique) est appelé un arbre de classification, tandis qu'un arbre de décision avec une plage de valeurs continues (numérique) est appelé un arbre de régression (Tsaia et al, 2009).

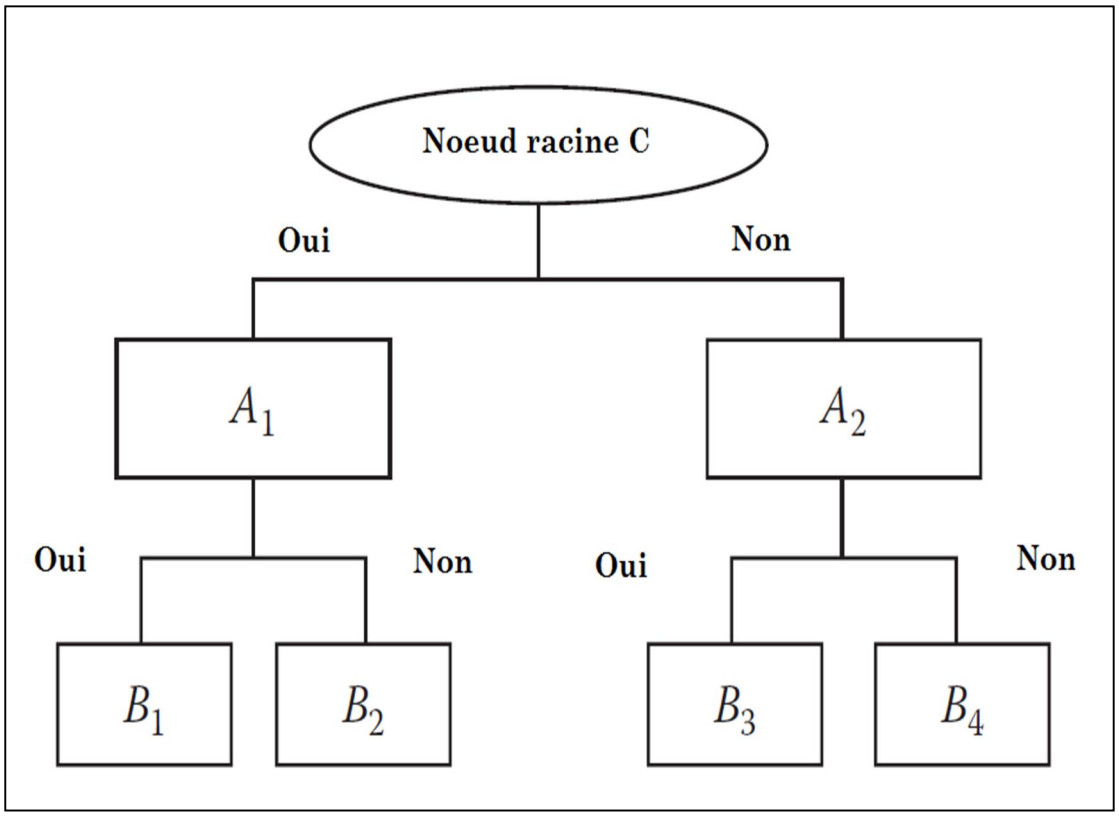

Figure 16 L'arbre de décision

3.3.3.1.1.5. La logique floue

La logique floue est basée sur le concept du phénomène flou qui se produit fréquemment dans le monde réel. La théorie des ensembles floue considère que l'ensemble des valeurs d'appartenance et la plage des valeurs sont compris entre 0 et 1. Autrement dit, en logique floue le degré de vérité peut varier entre 0 et 1, et il n'est pas limité aux deux valeurs de vérité vraie et fausse. Par exemple, "pleut" est un phénomène communément naturel, et il peut y avoir des changements très féroces. Pleuvoir peut se convertir d'après les circonstances de légère à violent (Zimmermann, 2001).

3.3.3.1.1.6. Les algorithmes génétiques

Les algorithmes génétiques (GA) utilisent l'ordinateur pour appliquer la sélection naturelle et l'évolution (Koza, 1992). Ce concept vient de la "survie adaptative dans les organismes naturels". L'algorithme commence par la génération aléatoirement d'une population importante de candidats. Certains types de mesure d'aptitude (fonction sélective) sont utilisés pour évaluer la performance de chaque individu dans une population. Un grand nombre d'itérations est réalisé pour remplacer les individus les moins performants par des recombinaisons génétiques des individus performants. Autrement dit, un individu avec une

fonction sélective faible est supprimé et ne survit pas pendant la prochaine itération (Tsaia et al, 2009).

3.3.3.1.2. Les techniques d'apprentissage non supervisé

Dans l'apprentissage non supervisé, les algorithmes utilisent des échantillons non étiquetés pour la formation, et obtiennent un modèle adapté à cet échantillon de données. L'apprentissage non supervisé est pour but de connaitre la distribution des données, et les relations entre les variables sans distinction entre les variables observées et les variables à prédire. Les méthodes les plus populaires dans l'apprentissage automatique non supervisé sont le K-plus proches voisins (k-NN), Cartes auto-organisées (SOM) et K-moyennes (Tsaia et al, 2009).

3.3.3.1.2.1. Le K-plus proches voisins

Le K-plus proches voisins (K-NN) est l'une des techniques non paramétriques la plus simple et la plus traditionnellement utilisée pour classer les échantillons (Bishop, 1996), (Manocha and Girolami, 2007). Comme l'illustre la figure 17, K-NN calcule les distances approximatives entre les différents points sur les vecteurs d'entrée, puis affecte le point non marqué à la classe de ses K-plus proches voisins. Dans le processus de création du classificateur K-NN, K est un paramètre très important et le changement des valeurs de k affectera les performances de notre classificateur. Si k est considérablement grand alors les voisins utilisés pour la prédiction vont prendre beaucoup de temps pour la classification. K-NN est appelé aussi algorithme d'apprentissage par l'exemple, et il est différent de l'approche d'apprentissage inductive (Mitchell, 1997). K-NN ne contient pas une étape d'apprentissage du modèle, il ne cherche que les exemples des vecteurs des entrées et il classifie les nouvelles instances. Par conséquent, K-NN apprend d'une manière "en ligne" les exemples et découvre le K plus proches voisins de la nouvelle instance (Tsaia et al, 2009).

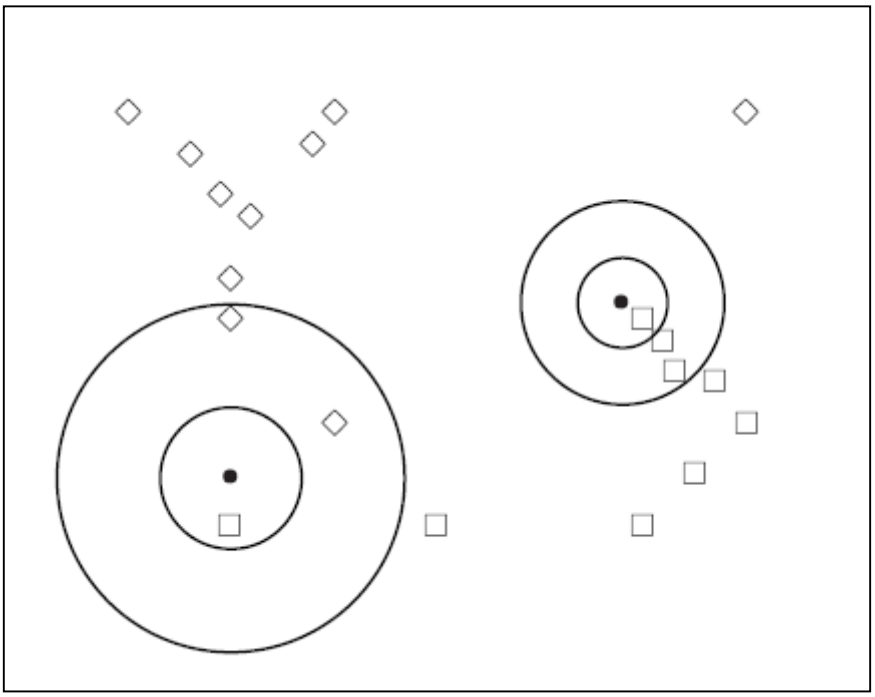

Figure 17 Le K Plus proches voisins

3.3.3.1.2.2. Les cartes auto-organisées

La carte auto-organisée (SOM) (Kohonen, 1982) est formée par un algorithme d'apprentissage non supervisé compétitif qui représente le processus d'auto-organisation. Le but de SOM est de réduire la dimension de la visualisation des données. Comme le montre la figure 18, SOM projette et regroupe les vecteurs d'entrée de grandes dimensions sur une carte visualisée de faible dimension (généralement deux dimensions pour la visualisation). SOM est généralement constituée d'une couche d'entrée et la couche de Kohonen qui est conçue comme arrangement bidimensionnel de neurones qui mappe les entrées de n dimensions à deux dimensions. SOM associe chacun des vecteurs d'entrée à une sortie représentative. Le réseau trouve le nœud le plus proche pour chaque cas d'entraînement et déplace le nœud gagnant qui est le neurone le plus proche (le neurone avec une distance minimale) pour le cas de l'apprentissage. Donc, SOM trace des vecteurs d'entrée similaires sur les mêmes ou les similaires unités de sortie sur une carte bidimensionnelle. Par conséquent, les unités de sortie vont s'auto-organiser un plan ordonné, de plus ces unités de sortie avec des poids similaires sont également placées à proximité après l'apprentissage (Tsaia et al, 2009).

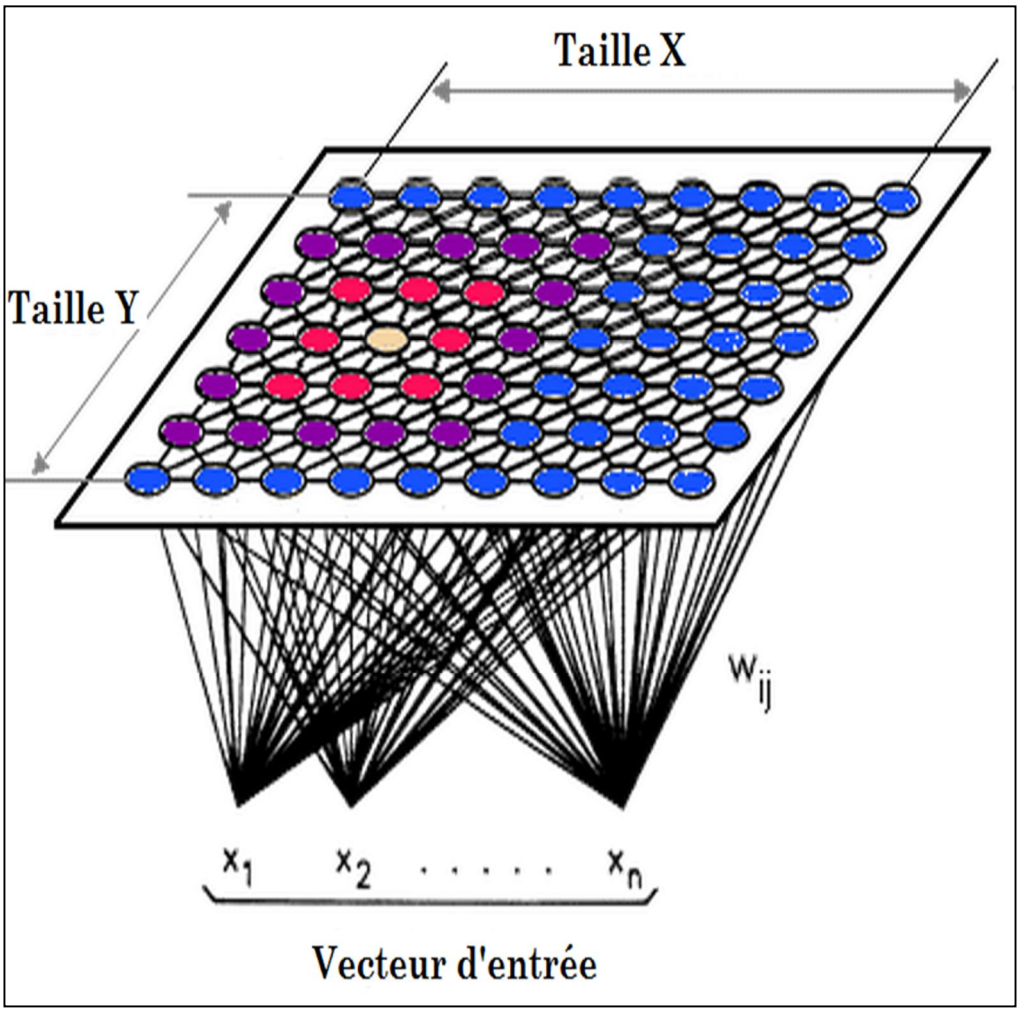

Figure 18 La carte auto-organisée

3.3.3.2. Les classificateurs hybrides

Dans le développement d'un IDS, le but ultime est de parvenir à la meilleure précision possible. Cet objectif nous conduit naturellement à la conception des approches hybrides pour résoudre le problème de détection d'intrusion. L'idée derrière un classificateur hybride est de combiner plusieurs techniques d'apprentissage automatique, afin d'améliorer considérablement la performance du système. Plus précisément, une approche hybride se compose généralement de deux éléments fonctionnels. Le premier traite les données brutes en entrée et génère des résultats intermédiaires. Le second prendra les résultats intermédiaires comme des entrées et il produit les résultats finaux (Jang et al, 1996). En particulier, les classificateurs hybrides peuvent être basés sur différents classificateurs en cascade. D'autre part, les classificateurs hybrides peuvent utiliser une approche basée sur le regroupement pour prétraiter les échantillons d'entrée afin d'éliminer les exemples d'apprentissages non représentatifs de chaque classe. Ensuite, les résultats de regroupement sont utilisés comme des exemples d'apprentissage pour le classificateur. Par conséquent, le

premier niveau des classificateurs hybrides peut être basé sur des techniques d'apprentissage supervisé ou non supervisé. Finalement, les classificateurs hybrides peuvent également se baser sur l'intégration de deux techniques différentes dont la première vise à optimiser les performances du deuxième modèle de la prédiction (Tsaia et al, 2009).

3.3.3.3. Les classificateurs d'ensemble

Les classificateurs d'ensemble ont été proposés pour améliorer les performances de classification des classificateurs simple (Kittler et al, 1998). Le terme «ensemble» fait référence à la combinaison de plusieurs algorithmes d'apprentissage faibles ou des apprenants faibles. Les apprenants faibles sont formés sur différents échantillons d'apprentissage afin que la performance globale puisse être efficacement améliorée. Parmi les stratégies utilisées pour combiner les apprenants faibles on trouve le «vote majoritaire» qu'il est sans doute le plus couramment utilisé dans la littérature. D'autres méthodes de combinaison telle que le renforcement et l'ensachage sont basés sur la formation de ré-échantillonnage des données, puis à un vote à la majorité sur les résultats des apprenants faibles (Tsaia et al, 2009).

3.3.4. Comparaison entre les IDSs basés sur les techniques de data mining

3.3.4.1. Par rapport au design du classificateur

Les méthodes de détection d'intrusion peuvent généralement être divisées en trois catégories simples, hybrides et ensembles. Le tableau suivant montre plus d'une cinquantaine d'articles basés sur des classificateurs simples, hybrides et ensemble.

	Simple	**Hybride**	**Ensemble**
Nombre d'articles	**22**	**27**	**7**
Référence des articles	(Balajinath and Raghavan, 2000), (Bouzida et al, 2004), (Chen et al, 2005), (Chimphlee et al, 2005) (Depren et al, 2005), (Eskin et al, 2002), (Fan et al, 2004), (Heller et al, 2003) (Li and Guo, 2007), (Liao and Vemuri, 2002), (Mukkamala et al, 2004), (Peddabachigari et al, 2004) (Ramos and Abraham, 2005), (Schultz et al, 2001), (Scott, 2004), (Shyu et al, 2003) (Tian et al, 2004), (Wang and Stolfo, 2004), (Wang and Battiti, 2006), (Wang et al, 2004) (Wang et al, 2006), (Zhang and Shen, 2005)	(Abadeh et al, 2007), (Bridges and Vaughn, 2000), (Chavan et al, 2004), (Chen et al, 2007) (Florez et al, 2002), (Giacinto and Roli, 2003), (Jiang et al, 2006), (Joo et al, 2003) (Kayacik et al, 2007), (Khan et al, 2007), (Lee and Stolfo, 1998), (Lee and Stolfo, 2000) (Liu and Yi, 2006), (Liu et al, 2004), (Liu et al, 2007), (Luo and Bridgest, 2000) (Moradi and Zulkernine, 2004), (Ozyer et al, 2007), (Peddabachigari et al, 2007), (Shon et al, 2006) (Shon and Moon, 2007), (Stein et al, 2005), (Toosi and Kahani, 2007), (Tsang et al, 2007) (Xiang and Lim, 2007), (Zhang et al, 2005), (Zhang et al, 2004), (Depren et al, 2005), (Eskin et al, 2002)	(Giacinto et al, 2006), (Han and Cho, 2003) (Kang et al, 2005), (Mukkamala et al, 2005), (Abadeh et al, 2007), (Giacinto and Roli, 2003), (Peddabachigari et al, 2007)

Tableau 2 La distribution du nombre d'articles par rapport aux types du design du classificateur (Tsaia et al, 2009)

Comme le montre le tableau 1, les classificateurs simples et hybrides représentent le plus grand nombre d'articles publie entre 2000 et 2007. En revanche, très peu d'études se basent sur les classificateurs d'ensemble bien qu'ils puissent surpasser les classificateurs simples en termes de précision de classification. La figure 19 présente la répartition des articles en fonction de la conception du classificateur par rapport aux années de publication. Le nombre d'articles qui se basent sur les méthodes simples a atteint le sommet en 2004 puis a diminué progressivement par la suite (Tsaia et al, 2009).

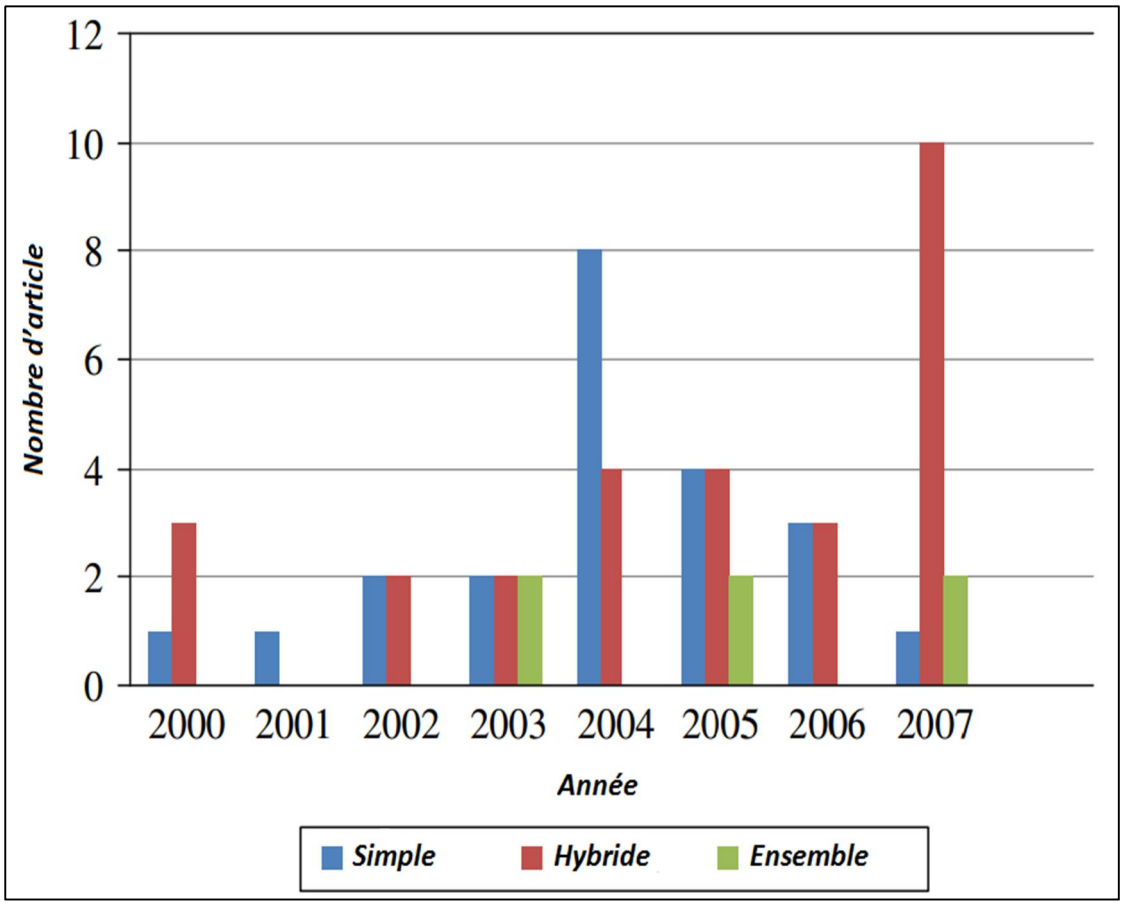

Figure 19 La distribution des articles par rapport aux types du design du classificateur entre l'année 2000 et 2007 (Tsaia et al, 2009)

En raison du développement récent du domaine de la détection d'intrusion, il est très difficile de concevoir une approche simple qui surpasse celles existantes. D'autre part, les approches hybrides sont passées de la marginalisation aux plus importantes approches dans les dernières années. La preuve est qu'il y a 10 publications basées sur des approches hybrides en 2007. Il faut noter qu'il n'y avait qu'un papier basé sur la méthode simple pour la même année. Sans aucun doute, les approches hybrides offrent une meilleure flexibilité et donc gagnent plus en plus de popularité dans les années à suivre (Tsaia et al, 2009).

3.3.4.2. Les classificateurs simples

Le tableau suivant illustre le nombre total des travaux basés sur des classificateurs simples en utilisant différentes techniques de classification.

	Fuzzy logic	**K-NN**	**SVM**	**NB**	**MLP**	**DT**	**SOM**	**GA**
Nombre d'articles	1	6	7	3	2	4	1	2
référence des articles	(Chimphlee et al, 2005)	(Bouzida et al, 2004), (Eskin et al, 2002), (Li and Guo, 2007), (Liao and Vemuri, 2002), (Wang and Stolfo, 2004), (Wang et al, 2004)	(Chen et al, 2007), (Eskin et al, 2002), (Heller et al, 2003), (Peddabachigari et al, 2004), (Tian et al, 2004), (Wang and Battiti, 2006), (Zhang and Shen, 2005)	(Schultz et al, 2001), (Scott, 2004), (Wang et al, 2006)	(Chen et al, 2007), (Shyu et al, 2003)	(Bouzida et al, 2004), (Depren et al, 2005), (Fan et al, 2004), (Peddabachigari et al, 2004)	(Ramos and Abraham, 2005)	(Balajinath and Raghavan, 2000), (Mukkamala et al, 2004)

Tableau 3 Le nombre total des articles qui se base sur les classificateurs simples entre l'année 2000 et 2007 (Tsaia et al, 2009)

La figure 20 présente la distribution de ces articles en fonction de leurs classificateurs par rapport aux années de publication. Comme le montre la figure 20, K-NN et SVM sont les techniques les plus utilisées pour l'approche simple dans le domaine de la détection d'intrusion. Ce résultat montre que SVM est le plus utilisé pour la conception des classificateurs simple. D'autre part, la logique de floue et SOM ne sont pas beaucoup utilisés dans la détection d'intrusion (Tsaia et al, 2009).

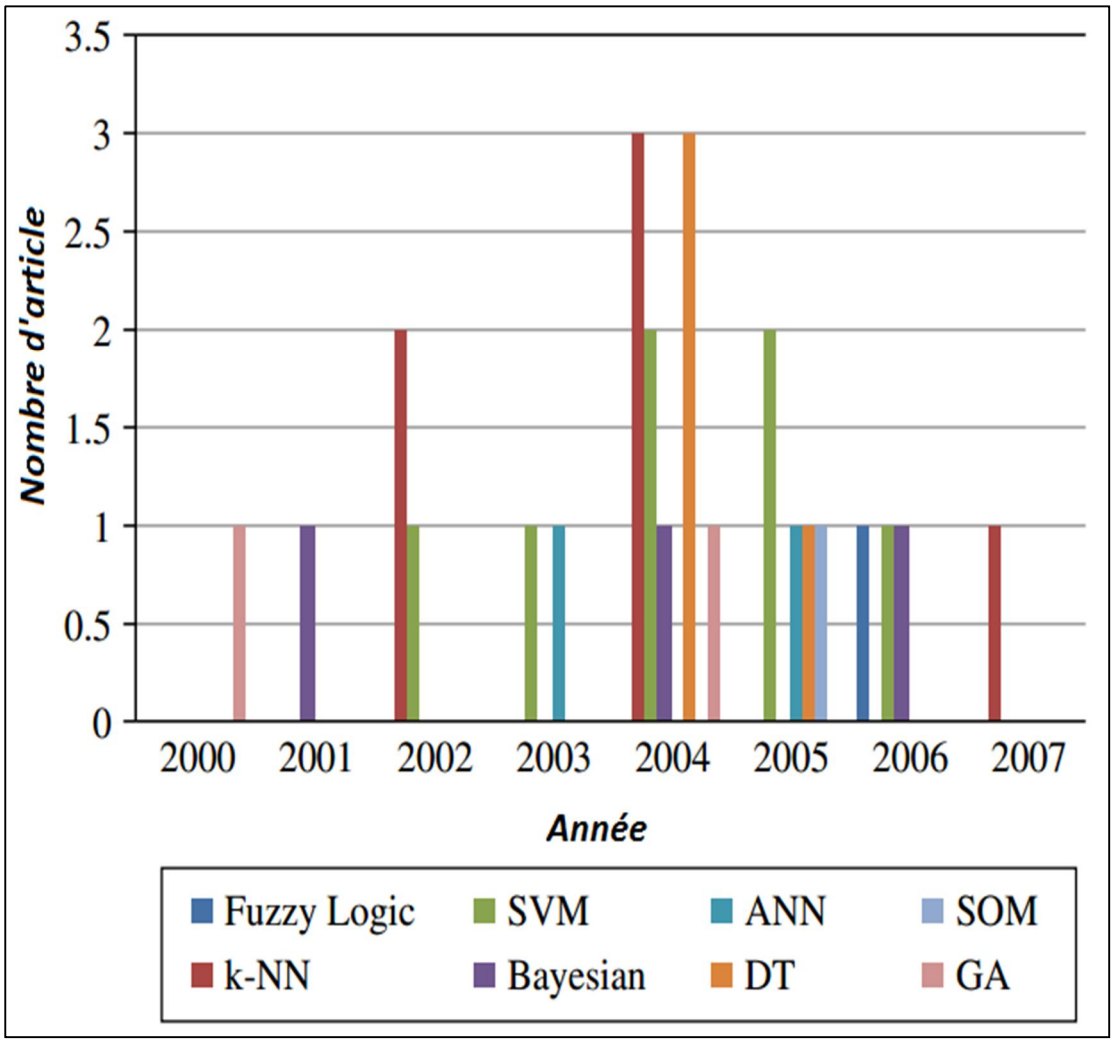

Figure 20 La distribution des articles basés sur les classificateurs simples entre l'année 2000 et 2007 (Tsaia et al, 2009)

3.3.4.3. Les classificateurs hybrides

Il y a trois stratégies pour concevoir des classificateurs hybrides qui sont : Cluster + méthodes simples, Méthodes hybrides en cascade, Méthodes hybrides intégrées. Le tableau 4 montre le nombre total des articles basés sur ces trois types de classificateurs hybrides (Tsaia et al, 2009).

	Cluster + méthodes simples	Méthodes hybrides en cascade	Méthodes hybrides intégrées
Nombre d'articles	5	8	9
Référence	(Chavan et al, 2004), (Khan et al, 2007), (Liu and Yi, 2006), (Liu et al, 2007), (Liu et al, 2004)	(Chen et al, 2005), (Depren et al, 2005), (Giacinto and Roli, 2003), (Joo et al, 2003), (Kayacik et al, 2007), (Moradi and Zulkernine, 2004), (Stein et al, 2005), (Zhang et al, 2005)	(Abadeh et al, 2007), (Bridges and Vaughn, 2000), (Depren et al, 2005), (Luo and Bridgest, 2000), (Ozyer et al, 2007), (Peddabachigari et al, 2007), (Shon et al, 2006), (Toosi and Kahani, 2007), (Xiang and Lim, 2007)

Tableau 4 Le nombre total des articles pour les classificateurs hybrides (Tsaia et al, 2009)

La figure 21 illustre la distribution de ces articles par rapport aux années en fonction de la conception du classificateur hybride.

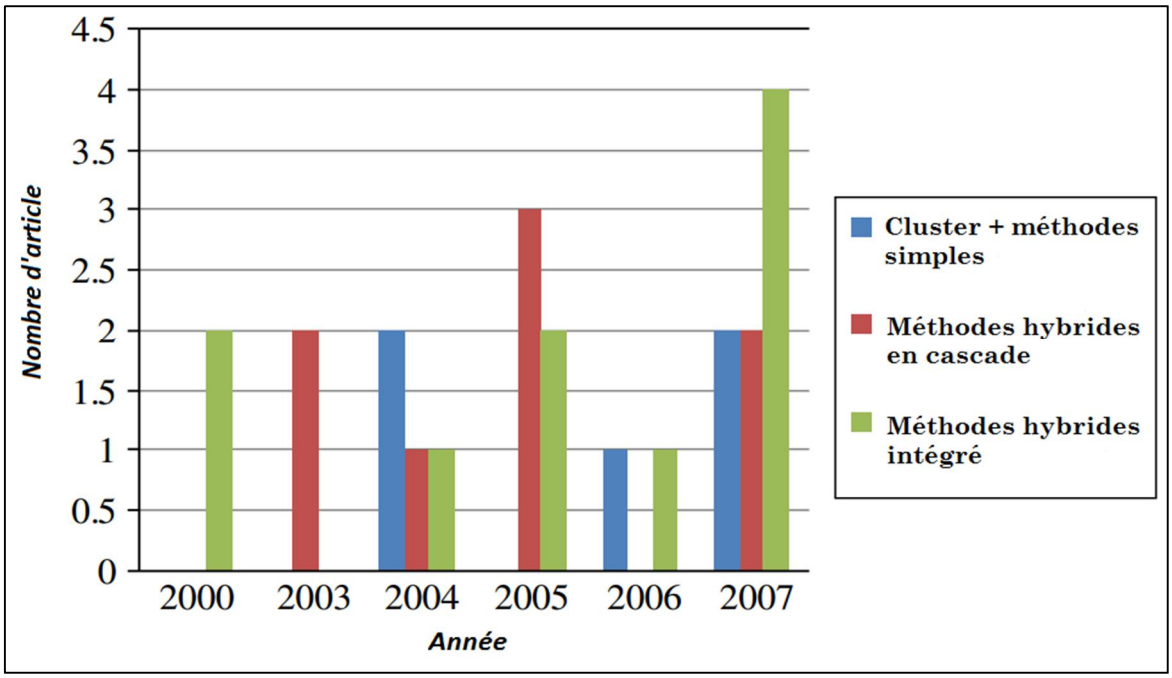

Figure 21 La distribution par année des articles pour les classificateurs hybrides (Tsaia et al, 2009)

Les résultats montrent que les classificateurs hybrides intégrés sont les approches de conception des classificateurs hybrides les plus utilisés pour la détection d'intrusion surtout en 2007. D'autre part, les classificateurs hybrides en cascade sont également largement utilisés dans la littérature.

Il faut noter que le nombre des articles basés sur les classificateurs de type ensemble pour la détection d'intrusion est très petit.

3.3.4.4. Les techniques de base utilisées dans la comparaison

Il existe plusieurs approches de base par rapport auxquelles des méthodes plus complexe ou hybride sont comparées dans la littérature. Autrement dit, chaque œuvre choisit généralement différents classificateurs de base pour valider leurs systèmes de détection d'intrusion. Le tableau 5 montre la distribution par année des techniques de data mining utilisées dans la comparaison avec les différents travaux publiés entre 2000 et 2007.

	'07	**'06**	**'05**	**'04**	**'03**	**'02**	**'01**	**'00**	**Totale**
K-Means	-	2	1	-	-	-	-	-	3
SVM	7	3	5	-	4	-	2	-	21
MLP	2	2	3	-	4	2	-		13
K-NN	2	1	2	-	-	1	3	-	9
LR	-	1		-	-	-	-	-	1
SOM	1	-	2	-	-	-	-	-	3
DT	4	-	5	-	1	-	-	2	12
GA	1	-	-	-	1	-	-	1	3
Bayesian	1	-	1	-	-	-	-	-	2

Tableau 5 La distribution par année des techniques de data mining utilisées dans la comparaison avec les différents travaux publiés entre 2000 et 2007 (Tsaia et al, 2009)

D'après le tableau 5, SVM est la technique de référence la plus utilisée. En second place, on trouve le réseau de neurones MLP suivi par l'arbre de décision.

3.3.5. Les ensembles de données utilisés

Le tableau 6 montre la distribution des bases de données utilisées pour l'évaluation des systèmes de détection d'intrusion entre 2000 et 2007.

	'07	'06	'05	'04	'03	'02	'01	'00	Totale
KDD'99	7	4	5	7	3	2	1	1	30
DARPA1998	5	3	4	2		2	1	1	18
DARPA1999	1	-	-	-	-	1	-	-	3
UNM	-	-	1	-		-	-	-	1
SSCNNJU	-	-	-	1	-	-			1
CUCS	-	-	-	1		-	-	-	1
RWND	-	-	-	-	1	-	-	-	1
PACCT	-	-	-	-	1	-	-	-	1
Windows System	-	-	-	-	1	-	-	-	1
Network tcpdump data	-	-		-	-	-	-	1	1

Tableau 6 La distribution des bases de données utilisées entre 2000 et 2007 (Tsaia et al, 2009)

En raison de l'existence de seulement quelques ensembles de données publics comme KDD'99, DARPA 1998 et DARPA 1999, beaucoup de travaux utilisent que ces ensembles de données pour leurs expérimentations. Très peu d'études utilisent des ensembles de données non publiques ou leurs propres bases de données. Ce résultat montre que ces jeux de données publiques sont reconnus comme des ensembles de données standard dans la détection d'intrusion.

NSL-KDD est une nouvelle base de données qui a été dérivée en 2010 du KDD'99. Dans cette nouvelle base de données certains problèmes desquels le KDD'99 souffre ont été traités. Actuellement KDD'99 (KDD'99, 1999) et NSL-KDD (NSL-KDD, 2009) représentent les bases de données les plus utilisées pour l'évaluation des systèmes de détection d'intrusion (pour plus de détaille voir annexe1 et annexe 2).

3.3.6. L'évaluation des techniques de data mining

L'efficacité d'une technique de data mining est évaluée par rapport à sa capacité de faire des prévisions correctes. Selon la nature réelle d'un événement donné par rapport à la prévision de la technique de data mining, les quatre résultats présentés dans le tableau 7 sont possibles, ce tableau est connu comme la matrice de confusion.

Les vrais négatifs ainsi que les vrais positifs correspondent à un fonctionnement correct de la technique de data mining, ce qui signifie que la technique de data mining a prédit avec succès respectivement le comportement normal et les attaques. Les faux positifs sont des comportements normaux prédits comme des attaques. Les faux négatifs sont des attaques incorrectement prédites comme des comportements normaux.

		Prédiction de la classe	
Classe actuelle		Classe négative (normal)	Classe positive (attaque)
	Classe négative (normal)	Vrai négative (VN)	Faux positif (FP)
	Classe positive (attaque)	Faux négative (FN)	Vrai positif (VP)

Tableau 7 La matrice de confusion

Les métriques traditionnelles de classification comprennent le taux d'exactitude et le taux d'erreur de la classification, elles sont définies comme suit:

$$\text{Exactitude} = \frac{VP+VN}{VP+VN+FP+FN} \qquad (3.13)$$

$$\text{Erreur} = 1\text{- exactitude} \qquad (3.14)$$

Ces paramètres sont sensibles au changement de l'ensemble de données. Par exemple, nous avons un ensemble de données qui a une distribution dans laquelle 95% des échantillons sont négatifs et 5% des échantillons sont positifs. Si 5 d'un ensemble de 100 échantillons de données d'essai sont positifs et 95 échantillons sont négatifs, alors, même si tous les résultats des tests sont classés comme négatifs, le taux d'exactitude sera de 95%. Cette valeur est conservée lorsque le nombre de vrai négatif (VN) augment alors que le nombre de vrai positif (VP) diminue du même montant. Lorsque le résultat positif est plus important pour les chercheurs, les mesures ci-dessus ne peuvent pas fournir l'information exacte sur la classification des classes. Pour évaluer d'une manière complète l'apprentissage déséquilibré surtout pour la classification de la minorité, d'autres mesures sont utilisées, ces mesures comprennent la précision, rappel, F-score, Q-score, G-moyenne, caractéristiques

opérationnelles du récepteur (ROC), zones sous caractéristiques opérationnelles du récepteur, courbes des rappels de précision et courbes de coûts (Dua and Xian, 2011).

Ces métriques sont définies comme suit:

$$précision = \frac{VP}{VP+FP} \qquad (3.15)$$

$$Rapel = \frac{VP}{VP+FN} \qquad (3.16)$$

$$\text{F} - \text{score} = \frac{(1+\beta)^2 * rapel * précision}{\beta^2 * rapel * précision} \qquad (3.18)$$

$$\text{F} - \text{score} = \frac{(1+\beta)^2 * VP}{(1+\beta)^2 * VP * \beta^2 * FN * FP} \qquad (3.19)$$

$$\text{G} - \text{moyenne} = \sqrt{\frac{\text{VP}}{\text{VP+FN}} * \frac{\text{VN}}{\text{VN+FP}}} \qquad (3.20)$$

La précision mesure l'exactitude de l'étiquetage positif qui représente la couverture des étiquettes positives correctes parmi tous les échantillons positifs marqués. Le rappel mesure la complétude de l'étiquetage positif qui représente le pourcentage des échantillons positifs correctement étiquetés parmi tous les échantillons positifs de la classe. La précision est sensible à la distribution des données, alors que le rappel non. Le rappel ne reflète pas combien d'échantillons sont étiquetés à tort comme positif, et la précision ne fournit aucune information sur le nombre d'échantillons positifs qui sont étiquetés correctement. Le F-mesure combine les deux mesures ci-dessus et attribue une pondération d'importance pour chaque précision ou rappel en utilisant le coefficient β. Par conséquent, le F-mesure donne une meilleure idée sur la précision d'un classificateur par rapport au rappel et à la précision, tout en restant sensible à la distribution des données. Le G-moyen évalue le biais inductif du classificateur en utilisant le ratio de positive à négative exactitude.

Les courbes ROC fournissent plus de perspicacité dans l'équilibre relatif entre les gains (vrai positif) et les coûts (faux positif) de classification sur un ensemble de données. Les mesures d'évaluation qui sont utilisées dans les courbes ROC sont comme suit:

$$\text{Taux VP} = \frac{VP}{VP+FN} \qquad (3.21)$$

$$\text{Taux FP} = \frac{FP}{FP+VN} \qquad (3.22)$$

Les courbes ROC comme le montre la figure 22 sont composée des valeurs combinatoires Taux de VP et Taux de FP. Chaque point de la courbe ROC correspond à la performance d'un classificateur sur un ensemble de données. Le point A est le résultat parfait de classification avec aucune erreur. Le point B est le résultat de classification le plus mauvais dont lequel toutes les étiquettes positives sont incorrectes. Le point situé plus près au point A donne un meilleur résultat de classification que le point plus proche au point B.

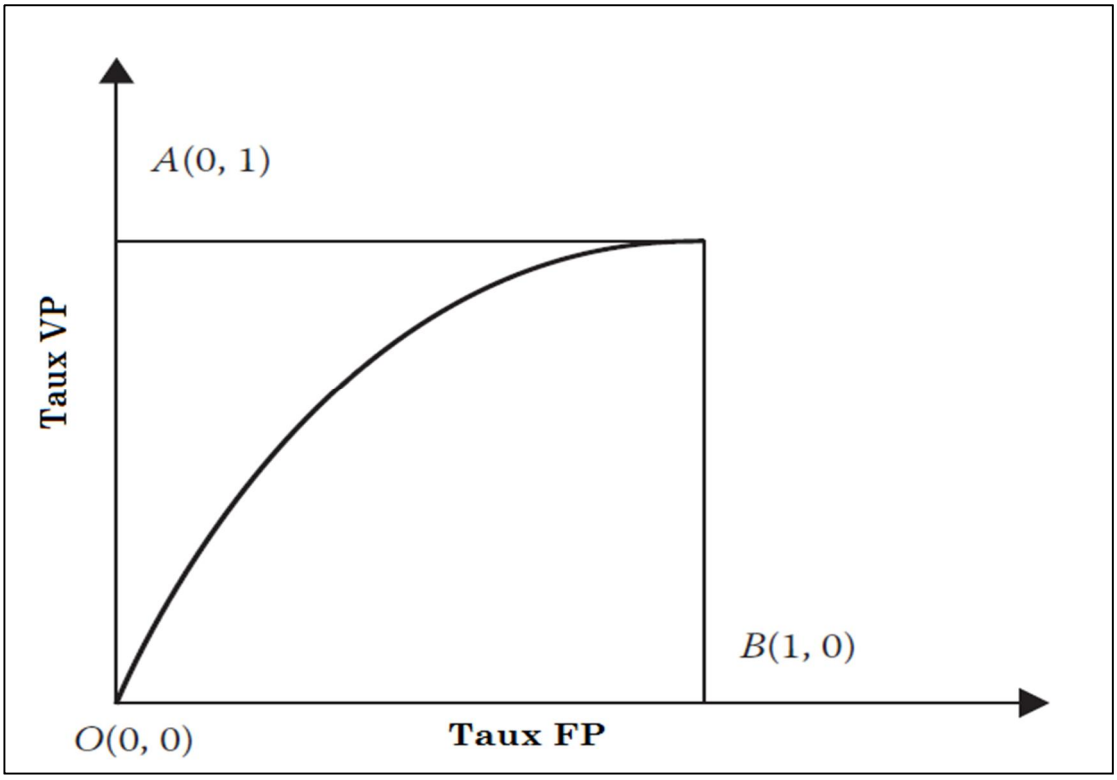

Figure 22 La courbe ROC

3.3.7. Les avantages des systèmes de détection d'intrusion basés data mining

Les systèmes de détection d'intrusion basés data mining ont montré une grande efficacité dans le traitement des problèmes de détection d'intrusion auxquels la première génération souffre. On peut résumer les avantages offerts par les techniques de data mining par les points suivants (Bensefia, 2009):

3.3.7.1. La capacité d'analyser un large volume de données

La première génération des systèmes de détection d'intrusion est limitée par rapport au large volume de données vu qu'elle ne peut pas extraire des modèles de comportement normal ou intrusifs avec les grands volumes de données. Cette limite représente un défi très important vu l'évolution des réseaux informatiques dans la dernière décennie. Les techniques de data

mining ont résolu ce problème grâce à leurs capacités de généralisation automatique et la rapidité des modèles de détection d'intrusion comportementales ou par scénario.

3.3.7.2. L'indépendance par rapport aux experts du domaine

Dans les systèmes de détection d'intrusion de la première génération, l'expert de sécurité représente la clé de l'efficacité et la performance du système vu qu'il spécifie les scénarios sur lesquels le modèle se base. La capacité des techniques de data mining de générer des modèles automatiques sans l'intervention des experts humains rend ces systèmes de détection d'intrusion plus autonome. Ce qui rend les performances de nos systèmes de détection d'intrusion indépendantes des experts humains.

3.3.7.3. L'autonomie et la rapidité

Dans la première génération des systèmes de détection d'intrusion, la conception et la construction des modèles se font d'une manière manuelle où on code manuellement les profils et les formes d'intrusion. Les techniques de data mining ont remplacé ce caractère ad hoc par une construction automatique des modèles de détection d'intrusion. De plus les techniques de data mining ont montré une rapidité dans la génération des modèles ainsi que dans le traitement des données, alors que la première génération souffre du coût élevé de ces étapes.

3.3.7.4. L'optimisation de la performance

Les techniques de data mining possèdent la capacité de détecter les nouvelles formes d'attaques non détectées par la première génération basée sur la signature des attaques. Cette capacité de détection des nouvelles formes d'attaques est acquisse grâce à la génération automatique du modèle de détection d'intrusion et la capacité de généralisation à partir d'un petit échantillon de données. La capacité de détecter des nouvelles attaques augmente le taux de vrai positif ce qui augmente considérablement les performances de notre système.

3.4. Conclusion

Le développement des systèmes de détection d'intrusion a passé par deux générations. La première génération représente le lancement des IDSs où les méthodes statistiques et les techniques de l'intelligence artificielle ont été utilisées. Cette première génération a montré beaucoup de limites comme les limites de performance, la conception ad hoc, l'intervention des experts humains…etc. Pour traiter les limites de la première génération, les techniques

de data mining ont été utilisés, ce qui représente la deuxième génération des systèmes de détection d'intrusion.

Chapitre 4

Les IDSs Adaptatifs

Ce chapitre montre les limites de la deuxième génération des systèmes de détection d'intrusion et l'importance des systèmes de détection d'intrusion adaptatifs qui traitent ces limites. Dans la première section, nous présentons les limites de la deuxième génération des IDSs, puis dans la deuxième section nous présentons les différentes solutions proposées pour traiter ces limites. Dans la troisième section, nous présentons les avantages et les inconvénients de certains IDSs adaptatifs existants.

4.1. Introduction

La deuxième génération des systèmes de détection d'intrusion a montré beaucoup de caractéristiques adaptatives comme l'adaptation au grand volume de données à traiter, la rapidité de traitement, l'adaptation aux réseaux à grande échelle… etc. Malgré tous ces avantages les systèmes de détection d'intrusion de la deuxième génération qui représentent les IDSs basés sur les techniques de data mining souffrent de certain nombre de limites qui les empêchent d'être adaptatifs et autonomes. Un système de détection d'intrusion adaptatif doit traiter les faiblesses et les limites des modèles de la deuxième génération des systèmes de détection d'intrusion. Ces limites peuvent être résumées par la nécessité d'une mise à jour régulière qui est en étroite relation avec les nouvelles formes d'attaques détectées ainsi que le problème de déploiement qui y est généralement causé par le temps nécessaire pour faire l'apprentissage de notre modèle.

Pour traiter ces limites, il existe certaines solutions qui ont été proposées ainsi que d'autres solutions que nous avons nous même proposées dans nos travaux effectués durant les années de notre thèse. Ces solutions ont pour but de rendre ces systèmes de détection d'intrusion adaptatifs avec le sens intégral du mot ou très proche d'un système adaptatif.

4.2. Les limites des systèmes de détection d'intrusion basés data mining

Les systèmes de détection d'intrusion basés data mining ont traité les limites de la première génération des IDSs comme l'étroite dépendance de l'environnement cible, la difficulté liée à l'évaluation, les limites de performance, le temps de traitement, l'intervention des experts humains. De plus, cette deuxième génération a montré beaucoup d'avantages qui ont un caractère adaptatif comme l'adaptation au grand volume de données et au réseau à grande échelle. Malgré tous ces grands avantages et ces bonnes caractéristiques, cette deuxième

génération des systèmes de détection d'intrusion souffre de certaines limites qu'on peut les résumer par les points suivants (Bensefia, 2009):

- **La nécessité de préparer un ensemble de données d'apprentissage:** Cette étape est indispensable pour l'apprentissage d'un modèle de détection d'intrusion basé data mining. Les données doivent être collecté du différent fichier de trace d'audité puis formater dans une forme spécifique adaptée à notre modèle. Dans le cas où on utilise une approche par scénario et que le mode d'apprentissage est un apprentissage supervisé, ces données doivent être étiquetées par un expert de sécurité. Dans le cas où on utilise une approche comportementale, les instances des attaques doivent être supprimées. Avec les grands volumes de données de trace d'audité générées par les différents réseaux et machines cette étape de collecte et traitement manuel des traces d'audite est devenu fastidieuse et très couteuse en temps et en effort.
- **Le temps nécessaire pour l'apprentissage:** Les systèmes de détection d'intrusion basés data mining ont besoin du temps pour faire l'apprentissage, ce temps varie de quelque milliseconde à des heures d'après la technique de data mining utilisée. Ce temps d'apprentissage peut devenir presque nul dans le cas d'un apprentissage continu. Le temps d'apprentissage jeux un rôle très important dans le déploiement et la mise à jour d'un système de détection d'intrusion. Un temps d'apprentissage très court rend le déploiement plus rapide, de plus le système sera plus disponible vu que le temps de mise à jour sera très court.
- **Le problème de déploiement:** Les systèmes de détection d'intrusion basés data mining dépends fortement des données d'apprentissage. Un système de détection d'intrusion entrainé par un ensemble de données qui provient d'un environnement donné ne fonctionne pas avec des données qui proviennent d'une autre source de données. Donc, il est indispensable de refaire l'étape de collection et de formatage des données afin de réformer notre système de détection d'intrusion. Ce coût élevé de collection et de formatage des traces d'audit est associé à chaque déploiement d'un IDS.
- **La difficulté de détecter les nouvelles formes d'attaques:** Malgré que les systèmes de détection d'intrusion de la seconde génération se basent sur divers techniques de data mining, la plupart d'entre elles se posent à la difficulté de la reconnaissance des nouvelles formes d'attaques. Généralement une nouvelle

forme d'attaque peut être soit une variation de la forme d'une attaque déjà connue, une représentation ultérieure d'une ancienne attaque ou une attaque complètement nouvelle qui vise un nouveau service ou un nouveau protocole. La capacité de généralisation des techniques de data mining a donné aux systèmes de détection d'intrusion de la seconde génération la capacité de détecter les attaques qui ressemblent aux formes des attaques utilisées pendant la phase d'apprentissage. Malgré cette capacité, les systèmes de détection d'intrusion basés data mining ont montré des limites face aux attaques complément nouvelles qui ne ressemblent pas aux attaques déjà rencontrées. Malgré que l'approche comportementale peut détecter ces nouvelles attaques, mais le taux très élevé de faux positif nous pousse à l'écarter comme une solution de ce problème.

- **La nécessité d'une mise à jour régulière:** À chaque fois qu'une nouvelle attaque est détectée, notre système de détection d'intrusion devient obsolète et la nécessité de mettre à jour notre IDS devient primordiale. Afin de mettre à jour notre système de détection d'intrusion, on doit :
  - Premièrement, mettre à jour notre ensemble de données d'apprentissage en ajoutant les nouvelles attaques.
  - Deuxièmement, refaire l'apprentissage de la technique de data mining utilisée.

  Cette mise à jour peut être immédiate à chaque fois qu'une nouvelle attaque est détectée ou périodique chaque heure ou chaque jour. Cette fonction de mise à jour est très couteuse en temps et en effort.

4.2.1. Le traitement des limites des systèmes de détection d'intrusion de la deuxième génération

Afin de traiter les limites de la deuxième génération des systèmes de détection d'intrusion, plusieurs solutions ont été proposées ainsi que d'autres solutions nous avons nous même proposées. Chaque une de ces solutions a été apportée afin de résoudre l'une des limites de la deuxième génération des systèmes de détection d'intrusion. Malgré tout l'avancement technologique certains problèmes restent persistants. On peut présenter ces solutions comme suit :

- **La solution pour la nécessité de préparer un ensemble de données d'apprentissage:** La préparation des données pour faire l'apprentissage est une

étape indispensable pour les systèmes de détection d'intrusion basés data mining. Malheureusement, le formatage et l'étiquetage des différents enregistrements restent toujours une étape qui demande l'intervention d'un expert en sécurité informatique. Si on utilise un système de détection qui se base sur l'approche comportementale, on peut connaitre si la connexion est une attaque ou non, mais on va rencontrer le problème de taux élevé de fausse alarme, de plus on ne peut pas connaitre le type exact de cette attaque. Pour créer un IDS avec un taux acceptable de fausse alarme et réussir à trouver le type de l'attaque on doit utiliser un IDS basée scénario ce qui rend les étapes de collecte, formatage et étiquetage inévitable pour n'importe quel type de système de détection d'intrusion performant.

- **La solution pour le temps d'apprentissage:** Le temps d'apprentissage joue un rôle très important dans l'apprentissage, la mise à jour et le déploiement d'un système de détection d'intrusion. Si on utilise un système de détection avec un mode d'apprentissage continu ce temps sera presque nul, mais le système ne sera pas performant dès sa mise en œuvre, c'est avec le temps qu'il sera plus en plus performant. Pour traiter ce problème du temps d'apprentissage, il faut que le système soit très rapide en termes de temps d'apprentissage. Par exemple un temps d'apprentissage de quelque second au maximum. Un système de détection d'intrusion basé sur un mode d'apprentissage continu avec un temps d'apprentissage initial très court sera la solution idéale.
- **La solution pour le problème de déploiement:** Un système de détection d'intrusion sans coût de déploiement est un système de détection portable qui sera compatible avec n'importe quel système ou plateforme, donc il faut que les données collectées de la trace d'audit de tous les enivrements soient standardisées. Cette problématique a été abordée par Michel and Mé (Michel and Mé, 2001) mais ce sujet reste très difficile à réaliser vu qu'aucun standard n'a été publié. De plus chaque système de détection d'intrusion utilise un ensemble de données différent des autres. En outre, les systèmes de détection d'intrusion sont de plus en plus liés à des systèmes spécifiques ce qui rend les efforts de standardisation moins en moins fructueux.
- **La solution pour la difficulté de détecter les nouvelles formes d'attaques:** Les nouvelles attaques qui ne sont que des modifications des formes des attaques déjà

détectées ou utilisées dans la phase d'apprentissage ne représentent pas un grand problème parce qu'un système de détection d'intrusion de la deuxième génération avec une grande capacité de généralisation a la capacité de les détecter. Le vrai problème est avec les attaques émergentes et complètement nouvelles. Ces attaques peuvent être détectées avec un IDS de type comportemental, mais à cause du taux très élevé de faux positif nous écartons cette solution. Les solutions qu'on peut utiliser sont :

- ❖ mettre à jour notre système de détection d'intrusion à chaque fois qu'une attaque complètement nouvelle est détectée.
- ❖ l'utilisation d'un mode d'apprentissage continu.
- ❖ l'utilisation des techniques d'apprentissage adaptatif.

La grande capacité de généralisation de notre système nous permet de détecter toutes les attaques drivées de cette nouvelle attaque.

- **La solution pour la nécessité de la mise à jour régulière:** La solution idéale pour la mise à jour est l'apprentissage continu où notre système peut apprendre à tout moment, donc l'apprentissage des nouvelles formes d'attaques ne sera pas couteux en termes de temps. La deuxième solution est de développer un IDS très rapide en termes de temps d'apprentissage où le réapprentissage ne sera pas très couteux. Il nous reste toujours le problème incontournable de la préparation des enregistrements pour faire la mise à jour.

On peut résumer la discussion précédente concernant les résolutions des problèmes des systèmes de détection d'intrusion de la deuxième génération par les solutions suivantes :

- L'utilisation des techniques d'apprentissage adaptatif.
- La création d'un système de détection d'intrusion très rapide en termes d'apprentissage.
- La création d'un système de détection d'intrusion qui possède une grande capacité de généralisation.
- La création d'un système de détection d'intrusion avec un mode d'apprentissage continu.

Ces solutions peuvent être utilisées seules ou combinées avec d'autres solutions dans le même modèle afin d'obtenir le système de détection d'intrusion adaptatif le plus performant.

4.3. Étude de certains systèmes de détection d'intrusion publiés

Dans cette section nous présentons quelques systèmes de détection d'intrusion adaptatifs existants. Pour chaque un de ces systèmes de détection d'intrusion nous montrons sa structure, son mode de fonctionnement ainsi que les critiques apportées par Bensefia et Ahmed-Nacer (Bensefia and Ahmed-Nacer, 2008) pour certains d'entre eux. De plus nous apportons nos propres critiques. Les systèmes de détection d'intrusion qui seront présentés dans cette section sont: SHIDS, PHIDS, HPCANN-IDS, FCANN-IDS.

4.3.1. SHIDS et PHIDS

SHIDS (Zhang et al, 2005) est l'abréviation de système de détection d'intrusion hiérarchique en série (Serial Hierarchical Intrusion Detection System). Il représente un système de détection d'intrusion hiérarchique en série construit progressivement. Au début, il n'y a que la couche du classificateur basé sur l'approche comportementale. Les connexions normales sont autorisées à passer, mais les connexions malicieuses sont détectées et stockées dans une base de données. Quand le nombre des attaques atteint le seuil, l'algorithme de regroupement est utilisé pour regrouper ces attaques dans différents groupes. Chaque groupe est utilisé pour l'apprentissage d'un nouveau classificateur basé sur un réseau de neurones de type RBF (Radial Basis Function). Ces classificateurs RBF représentent les couches de détection des attaques, ils sont basés sur l'approche par scénario. La figure suivante montre la structure ainsi que le mode de fonctionnement de SHIDS.

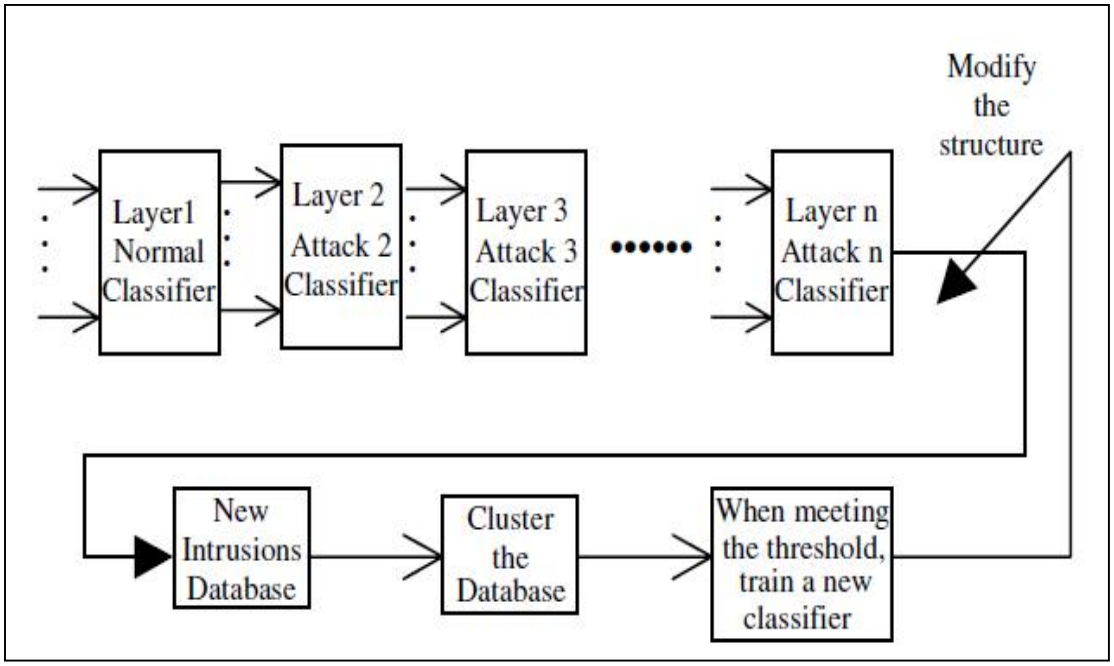

Figure 23 La structure de SHIDS

Le principal inconvénient de SHIDS est l'influence des erreurs des couches au-dessus sur les performances des couches au-dessous. Pour traiter ce problème, PHIDS (Zhang et al, 2005) a été proposé. PHIDS est l'abréviation de système de détection d'intrusion hiérarchique en parallèle (Parallel Hierarchical Intrusion Detection System). Comme le montre la figure 24 PHIDS contient trois niveaux: le premier niveau est un classificateur basé sur l'approche comportementale, le second est un classificateur basé sur l'approche par scénario, son rôle est d'identifier la catégorie principale de l'intrusion. Finalement le troisième niveau possède quatre classificateurs associés aux quatre types d'attaques: DOS, PROB, R2L et U2R.

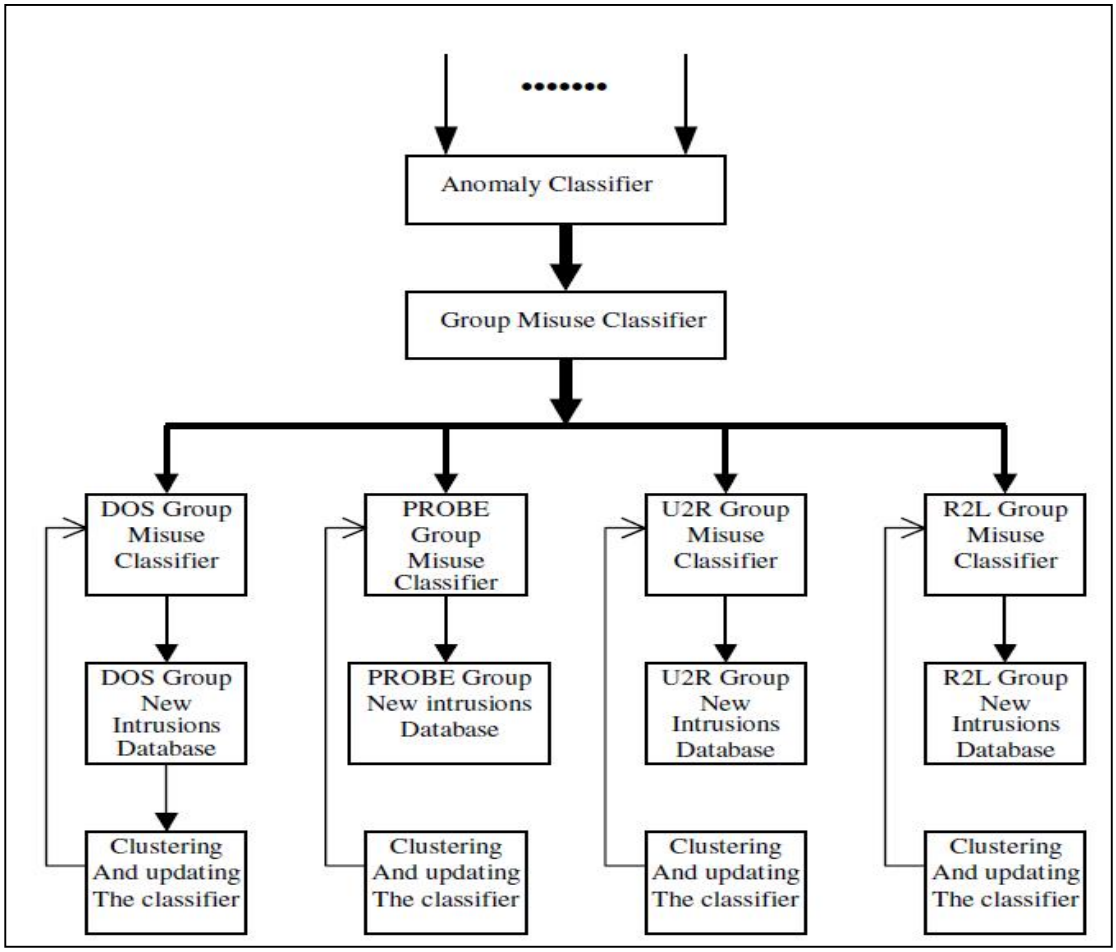

Figure 24 La structure de PHIDS

L'expérimentation en KDD'99 (KDD Cup 99, 1999) a montré que les deux modèles SHIDS et PHIDS basés RBF ont donné de très hautes performances et ont maintenu l'adaptabilité du modèle. Par contre, PHIDS est plus rapide et plus souple que SHIDS (Zhang et al, 2005).

4.3.1.1. Les avantages de SHIDS et PHIDS

On peut résumer les avantages du SHIDS et PHIDS par les points suivants :

- La combinaison de deux approches comportementale et par scénario, ce qui donne à ce système de détection d'intrusion la capacité de détecter les nouvelles formes d'attaque grâce à l'approche comportementale. De plus ce système possède la capacité de connaitre le type exact de l'attaque grâce à l'approche par scénario.
- SHIDS décompose le processus de détection des attaques en une série de classificateurs ce qui rend la phase de détection moins complexe.
- PHIDS décompose le processus de détection des attaques en une hiérarchie parallèle de classificateur ce qui rend la phase de détection moins complexe.
- Une grande flexibilité en termes d'apprentissage et de mise à jour.

4.3.1.2. Les inconvénients de SHIDS et PHIDS

On peut résumer les inconvénients du SHIDS et PHIDS par les points suivants :

- L'influence de la décision des couches au-dessus sur les couches au-dessous dans SHIDS, ce qui a poussé les chercheurs à développer le PHIDS.
- Vu que la première couche est basée sur l'approche comportementale, une connexion considérée comme attaque peut être un nouveau comportement normal.
- Le manque d'un mécanisme de mise à jour pour les nouveaux comportements normaux.
- SHIDS et PHIDS utilisent les réseaux de neurones qui sont très couteux en termes d'apprentissage/réapprentissage.
- Le réapprentissage dépend d'un seuil fixé à l'avance, qui peut être longue comme courte. Donc, la capacité de détecter les nouvelles formes d'attaques dépend de ce seuil.

4.3.2. HPCANN-IDS

HPCANN-IDS (Liu et al, 2007) est l'abréviation de système de détection d'intrusion hiérarchique basé réseau de neurones d'analyse de composent principale (Hierarchical Principal Component Analysis Neural Network Intrusion Detection System). HPCANN-IDS utilise les réseaux de neurones de type PCA afin de réduire la dimension des propriétés des connexions réseau. Comme le montre la figure 25, le premier niveau est un classificateur de type comportemental, son rôle est d'identifier les connexions anormales. Le deuxième niveau est constitué d'une série de trois classificateurs basés sur l'approche par scénario où chaque classificateur correspond à une catégorie principale d'attaque (DOS, R2L, PROBE). Le troisième niveau se compose de trois séries de classificateurs où chacun d'entre eux est relié à un classificateur de deuxième niveau. Les classificateurs d'une série correspondent aux différentes sous-classes des attaques appartenant à une classe principale d'attaque.

HPCANN-IDS fonctionne comme suit: si une connexion anormale n'a pas été identifiée par le deuxième niveau, le système initialise le champ flag comme une nouvelle classe principale d'attaque. Si une connexion anormale est identifiée par le deuxième niveau sans être identifiée par le troisième niveau, le système initialise le champ flag comme une nouvelle sous-classe d'une attaque qui doit appartenir à la classe principale identifiée.

Les signatures de la nouvelle attaque sont stockées dans une base de données. Lorsque le nombre des enregistrements atteint un seuil fixé à l'avance, le système déclenche la fonction de regroupement des enregistrements stockés en fonction du contenu du champ flag. Ensuite, il initie automatique le réapprentissage des nouveaux classificateurs qui seront connectés au niveau indiqué par le champ flag. L'expérimentation en DARPA (DARPA 98, 1998) montre la haute performance de ce modèle pour détecter les attaques DOS, Probe et R2L.

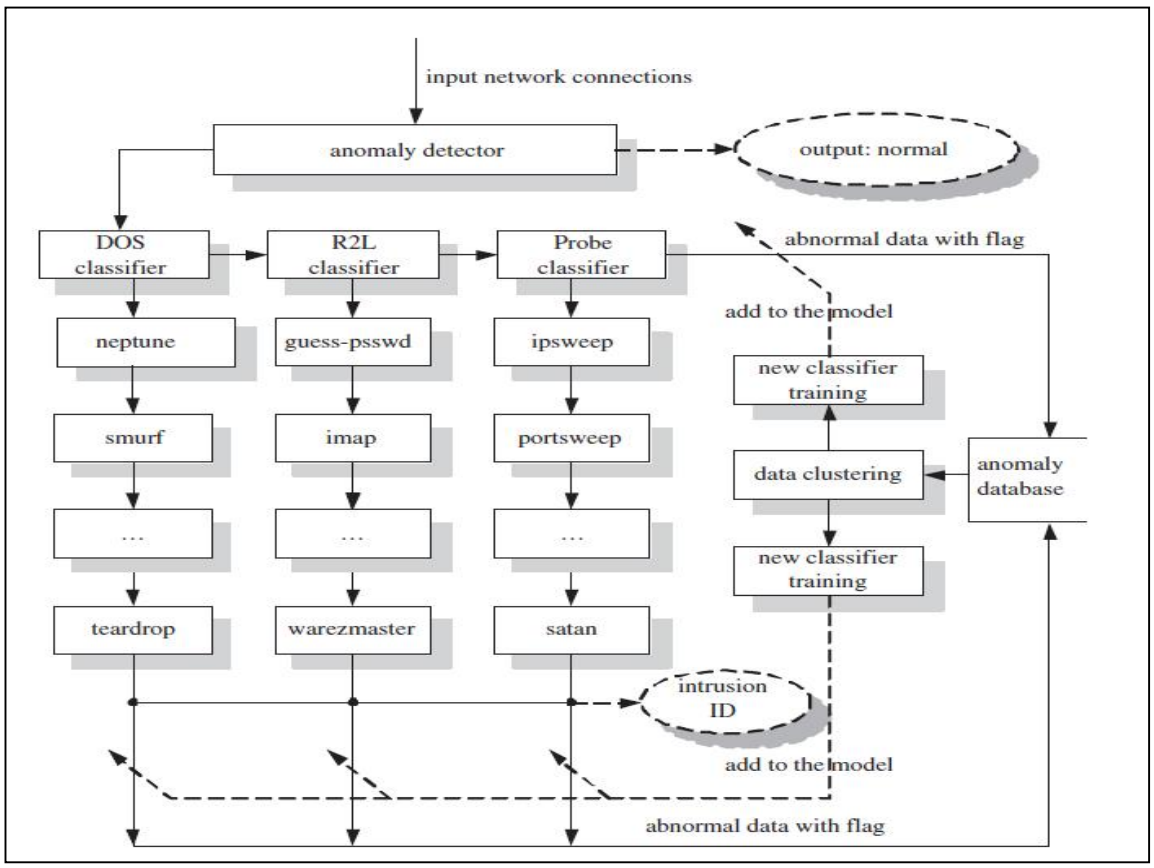

Figure 25 La structure de HPCANN-IDS

4.3.2.1. Les avantages de HPCANN-IDS

On peut résumer les avantages du HPCANN-IDS par les points suivants :

- La combinaison de deux approches comportementale et par scénario ce qui donne à ce système de détection d'intrusion la capacité de détecter les nouvelles formes d'attaques grâce à l'approche comportementale. De plus ce système possède la capacité d'identifier le type exact de l'attaque grâce à l'approche par scénario.
- Une classification détaillée où il identifie la classe principale puis la sous-classe.

- Extensibilité du HPCANN-IDS pour des nouvelles classes principales et sous-classes des attaques par l'ajout de nouveaux classificateurs dans le deuxième et le troisième niveau.

4.3.2.2. Les inconvénients de HPCANN-IDS

On peut résumer les inconvénients du HPCANN-IDS par les points suivants :

- Vu que le premier niveau est basé sur l'approche comportementale, une connexion considérée comme attaque peut être un nouveau comportement normal.
- Le manque d'un mécanisme de mise à jour pour les nouveaux comportements normaux.
- Le manque de la classe U2R ainsi que ses sous-classes.
- Afin d'utilisé PCA les trois propriétés de connexion: Protocol, flag et service ont été supprimées. Ces trois propriétés sont très importantes dans le processus de classification des attaques d'après les études effectuées par Zhang et Zulkernine (Zhang et Zulkernine, 2005).

4.3.3. FC-ANN-IDS

FC-ANN-IDS (Wanga et al, 2010) est l'abréviation de système de détection d'intrusion basé sur l'algorithme de regroupement flou et réseau de neurones (Fuzzy Clustering and Neural Network Intrusion Detection System). FC-ANN-IDS est un IDS hiérarchique basée sur les réseaux de neurones et la logique floue. Comme le montre la figure 26, FC-ANN-IDS est composé de trois couches: la première couche est un classificateur flou qui génère les différents sous-ensembles d'apprentissage. La deuxième couche représente les différents réseaux de neurones qui sont formés pour élaborer les différents modèles de base. La dernière couche est un module d'agrégation floue utilisé pour agréger les résultats et réduire les erreurs détectées. FC-ANN-IDS a été testé avec le KDD'99 (KDD Cup 99, 1999) et il a donné une bonne performance, surtout pour les attaques de faible fréquence (Wanga et al., 2010).

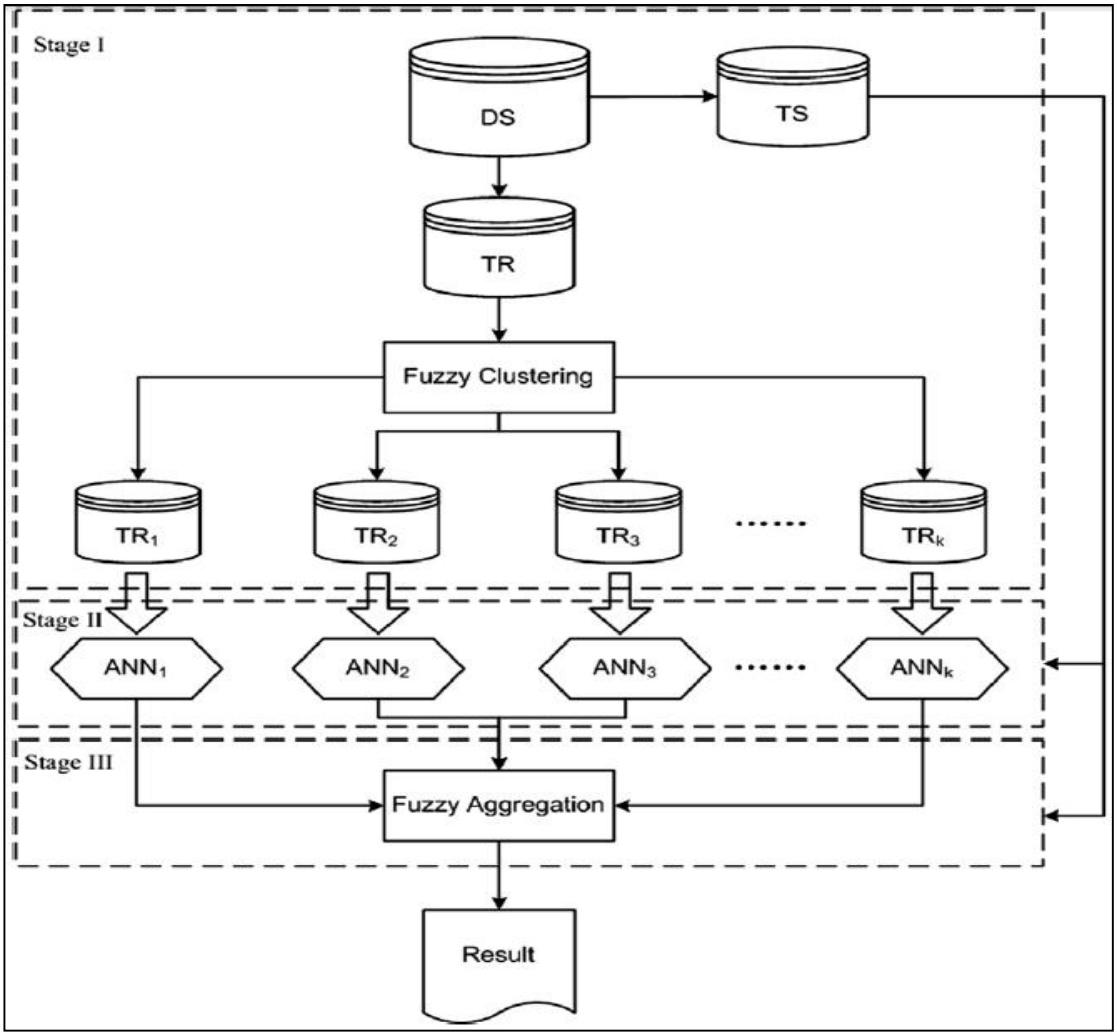

Figure 26 La structure de FC-ANN-IDS

4.3.3.1. Les avantages de FC-ANN-IDS

On peut résumer les avantages du FC-ANN-IDS par les points suivants :

- La combinaison de trois types différents d'algorithme: le clustering, la classification (réseau de neurones) et l'agrégation.
- Une autonomie totale grâce au module de regroupement
- L'agrégation des décisions des différents classificateurs pour les différents types de connexion.
- La structure hiérarchique qui décompose le processus de détection des attaques en une hiérarchie parallèle de classificateurs, ce qui rend la phase de détection moins complexe.
- La grande capacité de généralisation ce qui donne au système des bonnes performances dans la détection des nouvelles attaques.

4.3.3.2. Les inconvénients de FC-ANN-IDS

- La complexité de l'étape de mise à jour
- Le manque de flexibilité
- L'influence des erreurs de l'algorithme de clustering sur l'apprentissage des réseaux de neurones
- La lenteur de l'apprentissage/ réapprentissage à cause de l'utilisation des réseaux de neurones comme classificateurs.

4.4. Conclusion

Les systèmes de détection d'intrusion de la deuxième génération ont montré beaucoup de caractéristiques adaptatives. Malgré tous ces avantages, les systèmes de détection d'intrusion de la deuxième génération souffrent de certains nombres de limites. Les systèmes de détection d'intrusion adaptatives sont inventés afin de remédier ces limites où plusieurs stratégies ont été adopté comme l'apprentissage continu, les techniques adaptatifs de data mining, l'augmentation de la capacité de généralisation …etc. Quatre systèmes de détection adaptatifs ont été présentés dans ce chapitre afin de montrer quelques stratégies adoptées pour créer un système de détection adaptatif. Ces quatre IDSs ont traité certaines limites de la deuxième génération, mais ils souffrent toujours de d'autres limites.

PARTIE II

Contributions

Chapitre 5

Proposition 1 un système de détection d'intrusion très rapide en termes d'apprentissage

Le but de cette première proposition est de créer un système de détection d'intrusion très rapide en termes d'apprentissage/réapprentissage, afin de remédier le problème de réapprentissage des systèmes de détection d'intrusion de la deuxième génération. Le système de détection d'intrusion proposé est un IDS hiérarchique, hybride, très rapide en termes d'apprentissage et de haute performance appelé NFHP-IDS (New Fast Performed Hierarchical Intrusion Detection System) qui possède les caractéristiques suivantes: un temps d'apprentissage de très courte durée, détecte les attaques de faible fréquence, donne un taux élevé de détection pour les attaques fréquentes, et donne un faible taux de fausses alarmes. NFHP-IDS contient deux niveaux. Le premier niveau comprend les quatre classificateurs rapides suivants : "Random Forest", "Simple Cart", "Best first Decision Tree", "Naive Bayes" qui sont utilisés pour leurs excellentes performances dans la détection de respectivement comportement normal et DOS, Probe, R2L, U2R. Seules cinq sorties du premier niveau sont sélectionnées et utilisées comme des entrées du second niveau qui contient le "Naive Bayes" comme classificateur final. L'expérimentation avec le KDD'99 a montré la haute performance de notre modèle par rapport aux résultats obtenus par certains classificateurs bien connus. Ce travail a été l'objet de notre article «A New Fast and High Performance Intrusion Detection System » (Ahmim and Ghoualmi-Zine, 2013).

5.1. Description du modèle

5.1.1. La structure de NFHP-IDS

Dans cette section, nous présentons les différents composants de NFHP-IDS et leurs utilités. Comme le montre la figure 27, notre modèle contient deux niveaux:

- Le premier niveau: ce niveau contient les différents types de classificateurs. Ces classificateurs sont sélectionnés pour leurs courts temps d'apprentissage et leurs hautes performances dans la détection d'une ou de plusieurs classes de connexion. Comme le montre la figure 27, chaque classificateur donne cinq prédictions relatives aux quatre catégories d'attaques et du comportement normal. Nous ne maintenons que les prédictions des classes pour lesquelles les classificateurs sont sélectionnés. Ces cinq prédictions sont utilisées comme des entrées du second niveau.

- Le deuxième niveau: ce niveau contient un seul classificateur utilisé pour sa haute performance et sa rapidité d'apprentissage en tant que classificateur final. Il analyse les prédictions sélectionnées des différents classificateurs du premier niveau et prend la décision finale. Cette décision peut être soit une attaque soit un comportement normal.

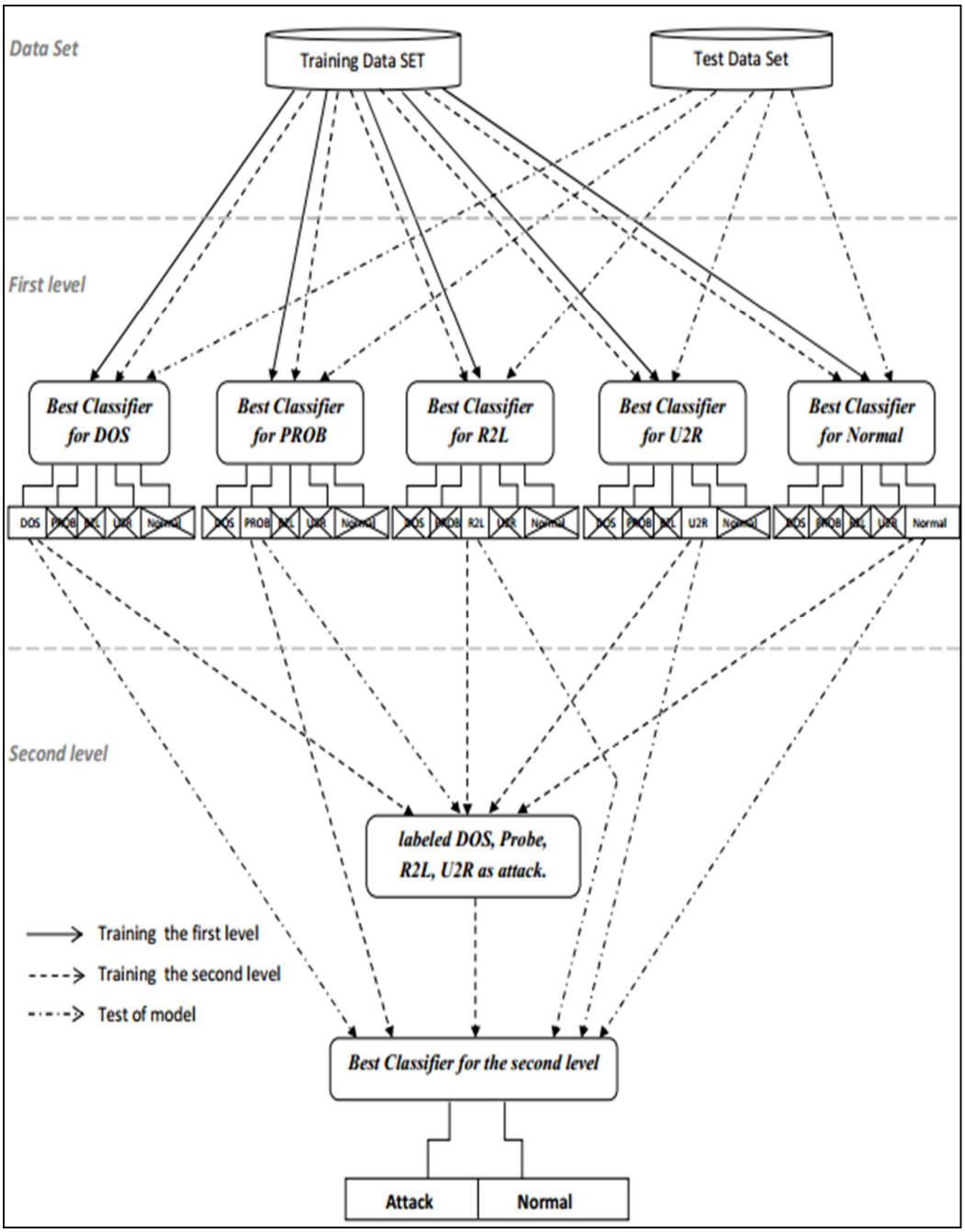

Figure 27 La structure générale de NFHP-IDS

5.1.2. Le mode de fonctionnement de NFHP-IDS

Le mode de fonctionnement de NFHP-IDS se compose de trois étapes: la phase de sélection des classificateurs du premier et second niveau, la phase d'apprentissage et la phase de test.

5.1.2.1. Sélectionner les différents classificateurs du premier et du second niveau

Dans le but de sélectionner les meilleurs classificateurs pour NFHP-IDS, nous effectuons deux études comparatives entre différents types de classificateurs. Dans la première partie, nous comparons les différents classificateurs par rapport à leurs rapidités d'apprentissage et leurs performances pour la classification des connexions dans l'une des cinq classes de connexions (DOS, Probe, R2L, U2R, et le comportement normal). Nous ne sélectionnons que les cinq classificateurs qui donnent un temps d'apprentissage très court et un bon taux de correcte classification pour les cinq catégories de connexions. De plus, chaque classificateur sélectionné doit donner le taux de correcte classification le plus élevé pour au moins l'une des cinq classes. Pour effectuer la deuxième étude comparative, nous générons un nouvel ensemble de données à partir des cinq prédictions sélectionnées du premier niveau. Ensuite, nous utilisons les données d'apprentissage pour comparer les différents classificateurs dans la classification des connexions en deux classes: attaque ou comportement normal. Nous sélectionnons le classificateur qui donne un temps d'apprentissage très court, le taux de vrai positif le plus élevé et le taux de fausse alarme le plus bas.

5.1.2.2. La phase d'apprentissage

Dans cette phase, nous effectuons l'apprentissage de notre modèle dans le but de le préparer pour la phase de test. Cette phase est composée de deux étapes:

- Former le premier niveau: nous formons les différents classificateurs du premier niveau avec l'ensemble de données d'apprentissage, où chaque élément de l'ensemble de données d'apprentissage représente une entrée pour les classificateurs.
- Former le second niveau: un nouvel ensemble de données est créé à partir des prédictions des classificateurs du premier niveau. Pour générer ce nouvel ensemble de données d'apprentissage, nous associons les prédictions sélectionnées du premier niveau avec l'étiquette correcte comme le montre le tableau suivant. Le nouvel ensemble de données d'apprentissage est utilisé pour former le classificateur sélectionné pour le deuxième niveau.

La prédiction pour DOS	La prédiction pour Probe	La prédiction pour U2R	La prédiction pour R2L	La prédiction pour Normal	étiquette
0.94	0.25	0.17	0.38	0.18	Attaque
0.15	0.34	0.18	0.36	0.94	Normal
0.28	0.28	0.89	0.22	0.15	Attaque
0.35	0.99	0.38	0.14	0.36	Attaque
0.16	0.13	0.25	0.89	0.32	Attaque

Tableau 8 Le nouvel ensemble de données d'apprentissage

5.1.2.3. La phase de test

Dans cette phase, nous testons la performance de notre modèle après l'achèvement de la phase d'apprentissage, où nous utilisons l'ensemble de données de test. Nous traitons chaque enregistrement de l'ensemble de données de test par les différents classificateurs du premier niveau. Ensuite, nous utilisons les prédictions sélectionnées des différents classificateurs du premier niveau comme entrées pour le classificateur du deuxième niveau.

5.1.2.4. Optimisation du temps d'apprentissage et du test

Pour optimiser le temps d'apprentissage et de test, nous avons proposé l'architecture distribuée détaillée dans la figure suivante.

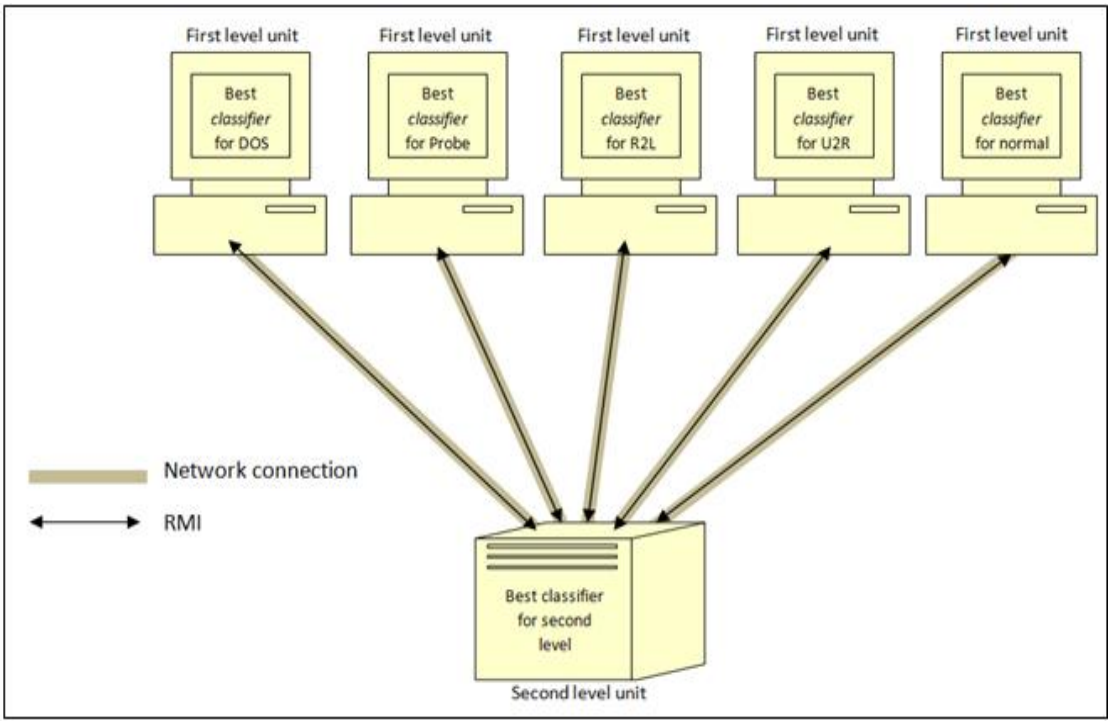

Figure 28 L'architecture distribuée de NFHP-IDS

Cette architecture distribuée contient deux types d'unités, où toutes les unités du premier niveau sont connectées à l'unité de deuxième niveau. Chaque unité du premier niveau

contient l'ensemble de données d'apprentissage et l'un des classificateurs sélectionnés. Toutes les unités du premier niveau forment leurs classificateurs simultanément en utilisant leurs ensembles de données d'apprentissage local. L'unité du second niveau génère le nouvel ensemble de données d'apprentissage où elle utilise l'appel de méthode à distance (RMI) pour demander les prédictions des différentes unités du premier niveau pour chaque enregistrement de l'ensemble de données d'apprentissage. Ensuite, elle utilise le nouvel ensemble de données d'apprentissage pour former son classificateur. Cette architecture nous permet de réduire le temps d'apprentissage, qui devient le temps d'apprentissage du plus lent classificateur du premier niveau plus la durée d'apprentissage du classificateur de deuxième niveau.

Pour l'étape du test, l'unité de second niveau appelle toutes les unités du premier niveau simultanément en utilisant RMI. Ensuite, elle traite leurs réponses par son classificateur. Les requêtes simultanées réduire le temps de test de NFHP-IDS.

5.2. Expérimentation

Cette section est divisée en trois sous-sections. Dans la première sous-section, nous détaillons l'ensemble de données d'apprentissage et de test. La deuxième sous-section représente une étude comparative entre huit classificateurs dans le but de sélectionner les classificateurs les plus performants et les plus rapides pour le premier et le deuxième niveau. La troisième sous-section représente une étude comparative entre notre nouveau modèle hiérarchique et d'autres classificateurs bien connus.

Nous avons effectué une série d'expérimentations avec le KDD'99 Cup (KDD'99, 1999), qui représente l'ensemble de données d'évaluation des systèmes détection d'intrusion le plus utilisé dans la dernière décennie. Weka Data Mining (Witten et al, 2011) est utilisé pour l'implémentation des différents classificateurs. Les résultats sont obtenus sur un PC Windows avec Core 2 Duo 2,0 GHz et 2 Go de RAM.

5.2.1. Les données d'apprentissage et de test

En raison de la grande taille du KDD'99_10%, nous avons créé notre ensemble de données d'apprentissage qui contient 30.000 enregistrements. Pour réduire la taille du KDD'99_10% tous les enregistrements redondants ont été supprimés. Ensuite, la sélection aléatoire est utilisée pour sélectionner les enregistrements Normal et DOS (Neptune). Le tableau 9 présente la répartition des attaques et du comportement normal de notre ensemble de

données d'apprentissage. L'ensemble des données KDD'99 Test est utilisé pour évaluer la performance de notre modèle. Les deux caractéristiques (num_outbound_cmds, is_host_login) sont supprimées en raison de leurs valeurs identiques dans l'ensemble de données d'apprentissage. Pour normaliser les ensembles de données, le codage ASCII est utilisé pour convertir les données symboliques à des valeurs numériques. Ensuite, chacune des données x_i de la caractéristique j est normalisée en utilisant l'équation suivante:

$$\overline{x_i(j)} = \frac{x_i(j) - min(x(j))}{max(x(j)) - min(x(j))} \qquad (5.1)$$

Type de Connexion	Nombre d'enregistrements	Description	Pourcentage
Normal	8000	8000 enregistrements normaux distincts et aléatoirement sélectionnés du KDD'99 10%	26.67%
DOS	18819	Tous les enregistrements distincts de Pod, Land, Back, Teardrop, Smurf plus 16067 enregistrements distincts et aléatoirement sélectionnés de Neptune. Tous les enregistrements sont extraits du KDD'99 10%.	62.73%
Probe	2130	Tous les enregistrements distincts de Probe extraits du KDD'99 10%	7.10%
R2L	999	Tous les enregistrements distincts de R2L extraits du KDD'99 10%	3.33%
U2R	52	Tous les enregistrements distincts de U2R extraits du KDD'99 10%	0.17%

Tableau 9 La répartition des attaques et du comportement normal de notre ensemble de données d'apprentissage

5.2.2. Étude comparative entre les différents types de classificateurs

Dans le but de choisir les meilleurs classificateurs pour les deux niveaux de notre nouveau IDS hiérarchique, nous avons réalisé deux études comparatives. La première consiste à sélectionner les différents classificateurs du premier niveau qui donnent un temps d'apprentissage très court et le meilleur taux de correcte classification pour DOS, Probe, U2R, R2L et le comportement normal. La seconde consiste à sélectionner le classificateur du deuxième niveau qui donne un temps d'apprentissage très court et le meilleur taux de classification des connexions en attaque et comportement normal. Les huit classificateurs

comparés sont les suivants: " Multilayer Perceptrons (MLP) " (Bishop, 1996), "Naïve Bayes (NB)" (John and Langley, 1995), "C4.5 Decision Tree (DT)" (Quinlan, 1993), "Support Vector Machine (SVM)" (Chang and lin, 2001), "Simple Cart (SC)" (Breiman et al., 1984), "Random Forests (RF)" (Breiman, 2001), "Best First Decision tree (BFTree)" (Shi, 2007) et "Repeated Incremental Pruning to Produce Error Reduction (RIPPER)" (Cohen, 1995). Dans cette étude comparative, nous avons utilisé les données d'apprentissage détaillé dans le tableau ci-dessus.

5.2.2.1. Étude comparative entre les huit classificateurs pour le premier niveau

Pour comparer les différents classificateurs par rapport au premier niveau, nous avons effectué une série d'expérimentation, où chaque classificateur dispose de 39 entrées qui représentent les 41 caractéristiques du KDD'99 sans num_outbound_cmds et is_host_login. Chaque classificateur donne ses prédictions pour les quatre catégories d'attaques (DOS, Probe, U2L, R2L) et le comportement normal. Le tableau suivant résume le taux de correcte classification pour chaque catégorie de connexion des huit classificateurs ainsi que le temps nécessaire pour faire l'apprentissage.

	DOS	**Normal**	**Probe**	**R2L**	**U2R**	**Le temps d'apprentissage**
NB	90,9%	94,3%	89,6%	0,7%	21,9%	1,62
SVM	97,1%	98,5%	79,6%	9,8%	8,8%	61,62
MLP	97,5%	98,5%	76,9%	6,9%	6,6%	1074,77
BFTree	97,3%	77,9%	90,1%	62,3%	5,3%	28,79
DT	97,4%	87,2%	85,3%	8,5%	7,0%	12.32
RIPPER	98,1%	98,8%	83,1%	13,7%	15,8%	196,31
RF	97,5%	99,3%	84,1%	7,0%	11,4%	14,85
SC	97,4%	99,2%	90,4%	29,1%	8,8%	29,21

Tableau 10 Une étude comparative entre les huit classificateurs pour le premier niveau

Ripper et MLP ne seront pas sélectionnés en raison de leurs temps d'apprentissage trop long. Comme le montre la figure suivante, les meilleurs classificateurs pour le comportement normal et DOS, Probe, R2L, U2R sont respectivement "Random Forests", "Simple Carte", "Best First decision tree", "Naïve Bayes".

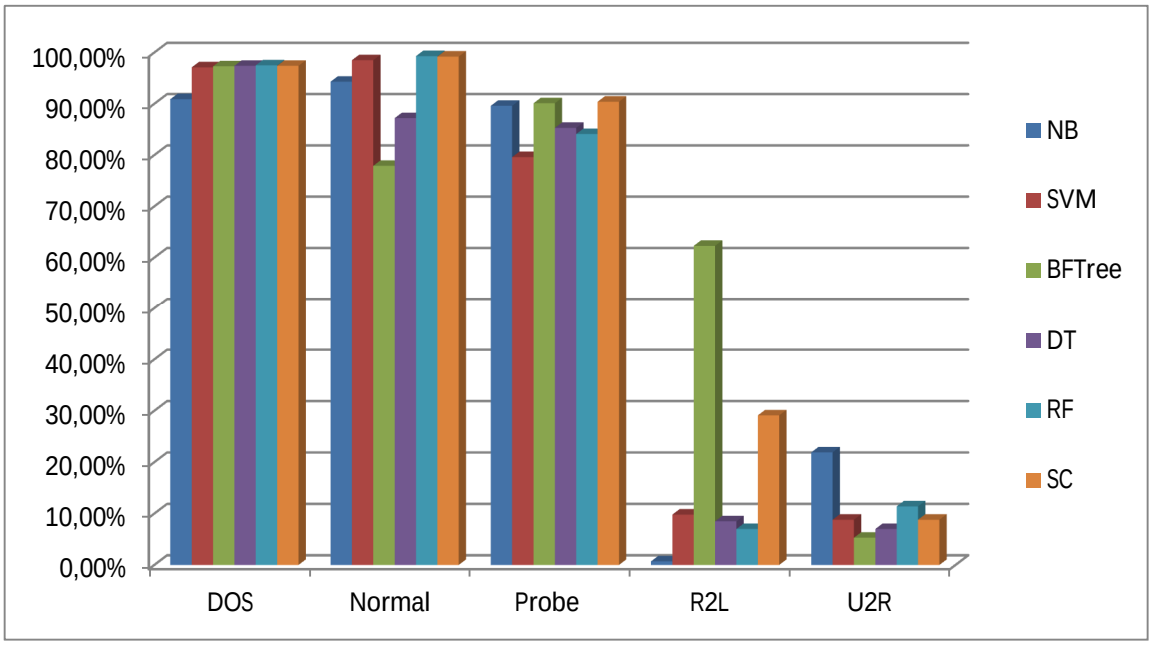

Figure 29 Etude comparative entre les huit classificateurs pour le premier niveau

5.2.2.2. Étude comparative entre les huit classificateurs pour le deuxième niveau

Pour choisir le meilleur classificateur pour le deuxième niveau, nous avons fait une série d'expérimentation, où chaque classificateur dispose de 5 entrées qui représentent les cinq sorties sélectionnées du premier niveau. Chaque classificateur donne sa prédiction qui peut être soit une attaque ou un comportement normal. Le tableau suivant résume le taux de correcte classification du comportement normal et attaque pour les huit classificateurs ainsi que leurs temps d'apprentissages.

	Attaque	**Normal**	**Tout**	**Temps d'apprentissage**
NB	94,3%	98,7%	95,1%	0,22s
SVM	92,2%	99,2%	93,5%	1,28s
MLP	92,0%	99,1%	93,4%	70,49s
BFTree	92,1%	99,1%	93,5%	0,88s
DT	92,1%	99,1%	93,5%	0,3s
JRIP	92,1%	99,1%	93,5%	0,7s
RF	92,0%	99,1%	93,4%	2,41s
SC	92,1%	99,1%	93,5%	1,54s

Tableau 11 Etude comparative entre les huit classificateurs pour le deuxième niveau

Comme l'illustre la figure 30, le meilleur classificateur pour la classification du comportement normal et attaque est Naïve Bayes qui donne le plus haut taux de correcte classification et le temps d'apprentissage le plus court.

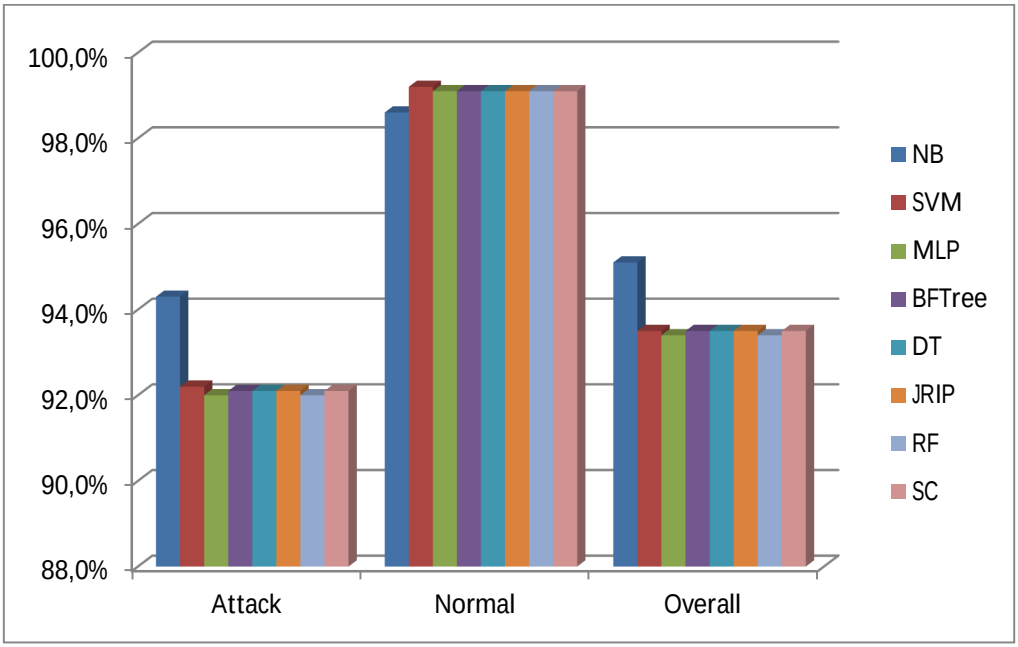

Figure 30 Etude comparative entre les huit classificateurs pour le deuxième niveau

5.2.3. Evaluation de notre nouveau IDS hiérarchique

Après avoir analysé la performance des différents types de classificateurs, nous avons exploité leurs points forts afin d'atteindre notre objectif. Dans le premier niveau, nous avons utilisé "Random Forests", "Simple Cart", "Best First Tree", "Naïve Bayes" pour leurs rapidités d'apprentissage et leurs taux de correcte classification le plus élevés pour respectivement comportement normal et DOS, Probe, R2L, U2R. Dans le deuxième niveau, nous avons sélectionné Naïve Bayes qui donne le meilleur taux de correcte classification et le temps d'apprentissage le plus court. La figure suivante montre la structure pratique de NFHP-IDS.

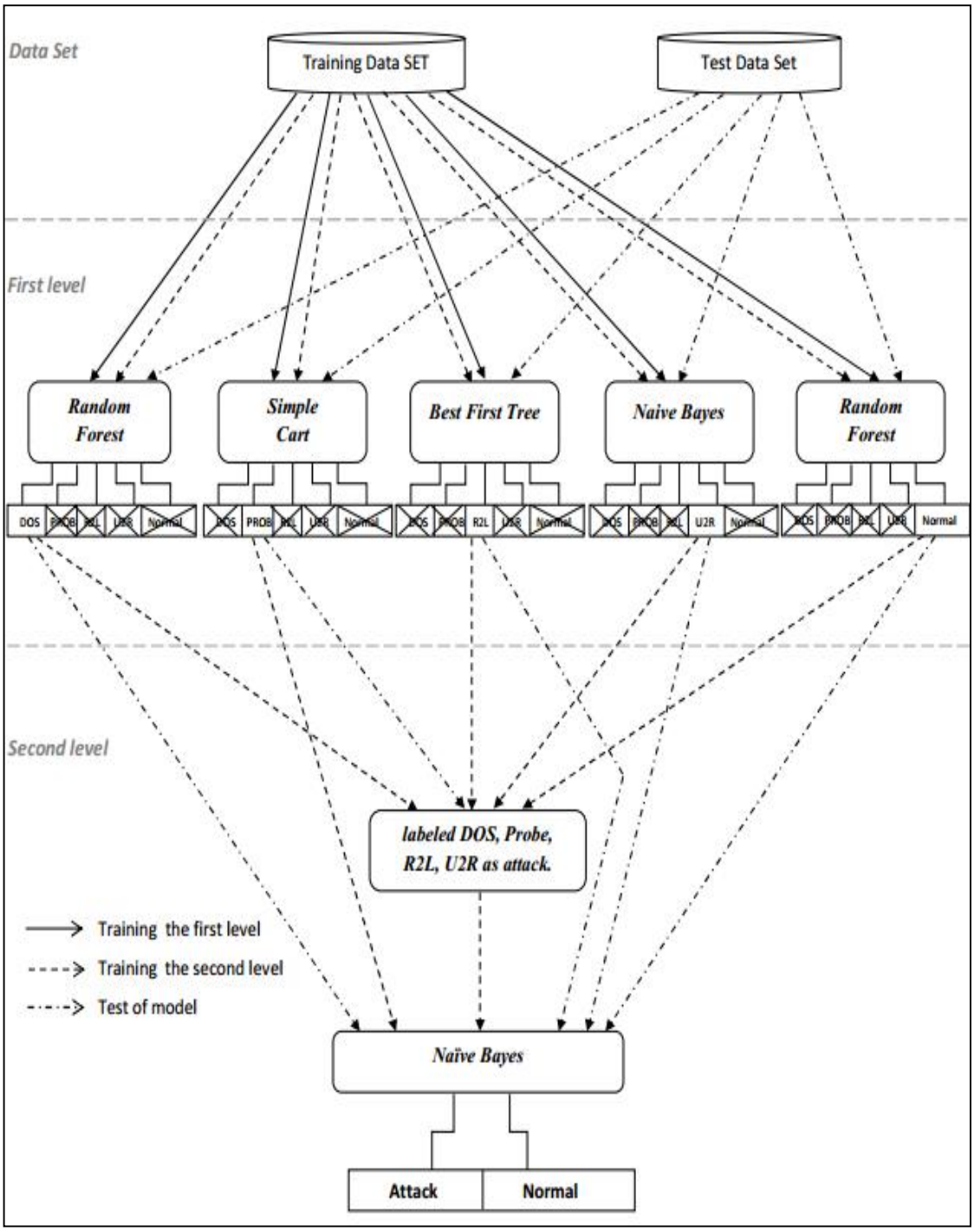

Figure 31 La structure pratique de NFHP-IDS

Pour évaluer la performance de NFHP-IDS, nous l'avons comparé avec certains classificateurs bien connus telles que: SVM, MLP, RIPPER, arbre de décision C4.5. Dans cette étude comparative, nous avons utilisé les données d'apprentissage détaillé dans le tableau 9 ci-dessus comme un ensemble de données d'apprentissage et le KDD'99 Test comme un ensemble de données de test. Les résultats de cette comparaison sont présentés dans le tableau ci-dessous.

		Notre modèle		MLP NN		SVM		RIPPER		DT	
		Attaque	Normal	Attaque	Normal	Attaque	Normal	Attaque	Normal	Attaque	Normal
Normal		1,35%	98,65%	0,93%	99,07%	3,75%	96,25%	7,71%	92,29%	2,11%	97,89%
Attaque	DOS	97,81%	2,19%	97,40%	2,60%	97,10%	2,90%	97,46%	2,54%	97,64%	2,36%
	PROB	98,13%	1,87%	88,24%	11,76%	84,40%	15,60%	88,36%	11,64%	98,42%	1,58%
	R2L	43,15%	56,85%	8,61%	91,39%	5,09%	94,91%	43,50%	56,50%	12,50%	87,50%
	U2R	72,81%	27,19%	65,35%	34,65%	56,14%	43,86%	78,51%	21,49%	85,96%	14,04%
	ALL	94,26%	5,74%	91,48%	8,52%	90,90%	9,10%	93,80%	6,20%	92,14%	7,86%
DR		94,26%		91,48%		90,90%		93,80%		92,14%	
FAR		1,35%		0,93%		3,75%		7,71%		2,11%	
Taux d'exactitude		95,12%		92,96%		91,95%		93,51%		93,26%	
Temps d'apprentissage		29,43 s		839,89 s		55,49 s		106,88 s		9,41 s	

Tableau 12 Etude comparative entre NFHP-IDS et d'autres classificateurs bien connus

NFHP-IDS a montré sa capacité à mieux détecter les attaques de faibles fréquences comme U2R et R2L sans perdre son taux élevé pour la correcte classification du comportement normal et des autres attaques fréquentes, ce qui représente un grand avantage. Comme le montre la figure 32, NFHP-IDS donne le taux de détection le plus élevé, le taux d'exactitude le plus élevé et le deuxième plus faible taux de fausse alarme. Le temps d'apprentissage de notre modèle représente le deuxième plus court temps d'apprentissage. En outre, le temps nécessaire pour tester tous l'ensemble de test de KDD'99 est de 15 secondes, ce qui signifie que le temps nécessaire pour tester un enregistrement est de 48 microsecondes. L'amélioration de la précision représente 4992 enregistrements correctement classés de plus par rapport au meilleur classificateur utilisé dans cette étude comparative.

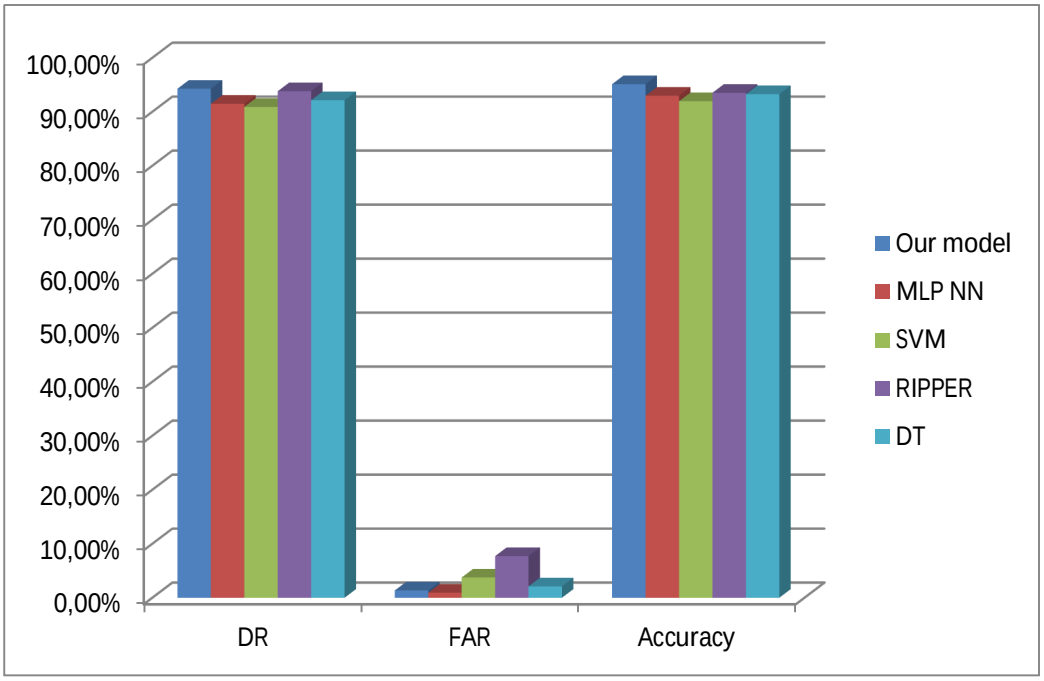

Figure 32 Etude comparative entre NFHP-IDS et d'autres classificateurs bien connus

Le temps d'apprentissage de NFHP-IDS est très court, ce qui signifie qu'on peut refaire l'apprentissage de notre modèle d'une manière très rapide. Donc, notre système de détection d'intrusion peut s'adapter aux nouvelles formes d'attaques d'une manière très rapide grâce à sa grande capacité de généralisation (95,12%) ainsi que son temps d'apprentissage/réapprentissage très court (29,43 seconds).

5.3. Conclusion

Dans ce chapitre, nous avons proposé un nouveau système de détection d'intrusion hybride, hiérarchique, très rapide et de haute performance appelé NFHP-IDS. Notre modèle est basé sur la combinaison de différents classificateurs rapide. Il est composé de deux niveaux. Le premier niveau contient les quatre classificateurs suivants : "Random Forests", "Simple Cart", "Best first decision tree" et "Naive Bayes", qui sont utilisées pour leurs excellentes performances dans la classification de respectivement comportement normal et DOS, Probe, R2L, U2R. Le deuxième niveau contient Naïve Bayes comme classificateur final, qui représente le classificateur le plus rapide et il offre le meilleur taux de correcte classification. Les expérimentations en KDD'99 montrent que NFHP-IDS est un modèle très rapide et donne une très bonne performance par rapport à d'autres techniques utilisés pour la détection d'intrusion, où il donne le taux de détection le plus élevé et le taux d'exactitude le plus élevé. En outre, NFHP-IDS a montré une capacité à mieux détecter les attaques de faibles fréquences. De plus, il a gardé ses hautes performances pour la détection

des attaques fréquentes et le comportement normal. Par conséquent, NFHP-IDS s'adapter aux nouvelles formes d'attaques grâce à sa grande capacité de généralisation ainsi que son temps d'apprentissage/ réapprentissage très court.

Chapitre 6

Proposition 2 un système de détection d'intrusion avec une très grande capacité de généralisation

Le but de cette proposition est de créer un nouveau système de détection d'intrusion hiérarchique avec une très grande capacité de généralisation, ce qui permet à notre système de détecter les nouvelles formes d'attaques. Notre IDS est basé sur un arbre binaire de différents types de classificateurs. Le modèle de détection d'intrusion proposé doit avoir les caractéristiques suivantes: combiner un taux de détection élevé et un taux de fausses alarmes faible, classer toute connexion dans l'une des cinq catégories de connexion réseau (attaques d'exploration (Probe), attaques de déni de service (DoS), attaques Utilisateur-à-root (U2R), attaques distant à local (R2L), et le comportement normal (normal)). Pour construire l'arbre binaire, nous regroupons les différentes catégories de connexions réseau hiérarchiquement en fonction de la proportion de faux positifs et de faux négatifs générer entre chaque deux catégories. Le modèle crée est un arbre binaire avec quatre niveaux. Dans le premier niveau, nous utilisons "Repeated Incremental Pruning to Produce Error Reduction (RIPPER)" pour classer les connexions réseau en deux catégories: les attaques DOS et G2 qui regroupe Probe, R2L, U2R et Normal. Puis, dans le deuxième niveau, nous utilisons "Naïve Bayes Multinomial (NBM)" pour classer les connexions réseau de G2 en Probe et G3 qui regroupe R2L, U2R et Normal. Après, nous utilisons "Ripple-down rule learner (Ridor)" pour classer les connexions réseau de G3 en R2L et G4 qui regroupe U2R et Normal. Finalement et comme dernier classificateur, nous utilisons "Random Tree (RT)" pour classer les connexions réseau de G4 en U2R et Normal. L'expérimentation avec NSL-KDD et KDD'99 a montré la haute performance de notre modèle par rapport aux résultats obtenus par des modèles de détection d'intrusion récents et certains classificateurs bien connus. L'étude comparative a montré que notre modèle donne un faible taux de fausse alarme et le taux de détection le plus élevé. De plus, notre modèle est plus efficace que certains classificateurs bien connus comme SVM, Arbre de décision C4.5, Réseau de neurones MLP, Naïve Bayes où il donne un taux d'exactitude égal à 83,26% pour le NSL-KDD. En outre, il est plus efficace que le meilleur des modèles récents de détection d'intrusion avec un taux d'exactitude égale à 95,72% pour le KDD'99. Ce travail a été l'objet de notre article «A new hierarchical intrusion

detection system based on a binary tree of classifiers » (Ahmim and Ghoualmi-Zine, 2015).

6.1. Description du modèle

Notre travail a pour but de construire un IDS hiérarchique de très grande capacité de généralisation qui combine un taux de détection élevé et un taux de fausses alarmes faible. En outre, chaque connexion doit être classée dans l'une des cinq classes de connexion réseau : les attaques d'exploration (Probe), les attaques de déni de service (DoS), les attaques Utilisateur-à-Root (U2R), les attaques distant-à-local (R2L) et le comportement normal (normal). Pour créer notre modèle, nous avons développé un algorithme de regroupement hiérarchique, pour former les groupes dont on a besoin afin d'entrainer les classificateurs des différents niveaux de l'arbre binaire, où nous sélectionnons le meilleur classificateur pour chaque niveau. Pour regrouper les différentes catégories de connexions réseau, nous avons utilisé deux mesures de performance: faux positifs et faux négatifs.

6.1.1. L'algorithme hiérarchique de clustering

Dans cette sous-section, nous présentons la façon avec laquelle nous regroupons les catégories des connexions réseau hiérarchiquement afin de définir les données d'apprentissage nécessaire pour former les différents niveaux de l'arbre binaire. Il existe cinq catégories de connexion: les attaques d'exploration (Probe), les attaques de déni de service (DoS), les attaques Utilisateur-à-Root (U2R), les attaques distant-à-local (R2L) et le comportement normal (normal). Pour regrouper hiérarchiquement les catégories de connexion, nous avons créé un algorithme basé sur la somme des faux positifs et faux négatifs générés entre chaque deux catégories de connexion. Dans un premier temps, nous évaluons l'ensemble de classificateurs avec le 10-fois-cross-validation, où nous utilisons les données d'apprentissage. Le résultat de cette évaluation est illustré par l'exemple du tableau 13.

Prédiction / Classe réelle	DOS	Probe	R2L	Normal	U2R
DOS	99,96%	0,00%	0,01%	0,03%	0,00%
Probe	0,33%	98,03%	0,28%	1,31%	0,05%
R2L	0,30%	0,00%	96,60%	3,10%	0,00%
Normal	0,08%	0,13%	0,23%	99,55%	0,01%
U2R	0,00%	5,77%	5,77%	17,31%	71,15%

Tableau 13 L'évaluation d'un classificateur avec 10 fois cross validation

Ensuite, en fonction des résultats de classification, on regroupe les deux catégories qui génèrent entre eux la plus grande proportion de taux de mauvaise classification dans la même nouvelle catégorie (groupe). Pour notre exemple, les classes normales et U2R génèrent le maximum de taux de mauvaise classification entre eux (17,31% + 0,01%). Après le regroupement de la classe normale et U2R dans la même nouvelle classe, on obtient le tableau suivant.

Prédiction / Classe réelle	DOS	Probe	R2L	Groupe4
DOS	99,96%	0,00%	0,01%	0,03%
Probe	0,33%	98,03%	0,28%	1,36%
R2L	0,30%	0,00%	96,60%	3,10%
Group4	0,07%	0,16%	0,26%	99,51%

Tableau 14 Le résultat de cross validation après le regroupement des deux classes

Nous répétons ce processus jusqu'à ce que toutes les catégories des connexions réseau soient regroupées dans le même groupe. À la fin de ce processus, les différents niveaux de l'arbre binaire seront formés en fonction des résultats de clustering. Les deux premières catégories regroupées représentent l'ensemble de données utilisées pour l'apprentissage du niveau le plus bas, et les deux dernières catégories regroupées sont utilisées pour l'apprentissage du niveau le plus haut. L'application de notre algorithme de clustering sur l'ensemble de données d'apprentissage du KDD'99 et NSL_KDD (Tavallaee et al., 2009) avec différents types de classificateur comme arbre de décision, réseaux de neurones, Machine à vecteurs de support donne la même hiérarchie de clustering détaillée dans la figure suivante.

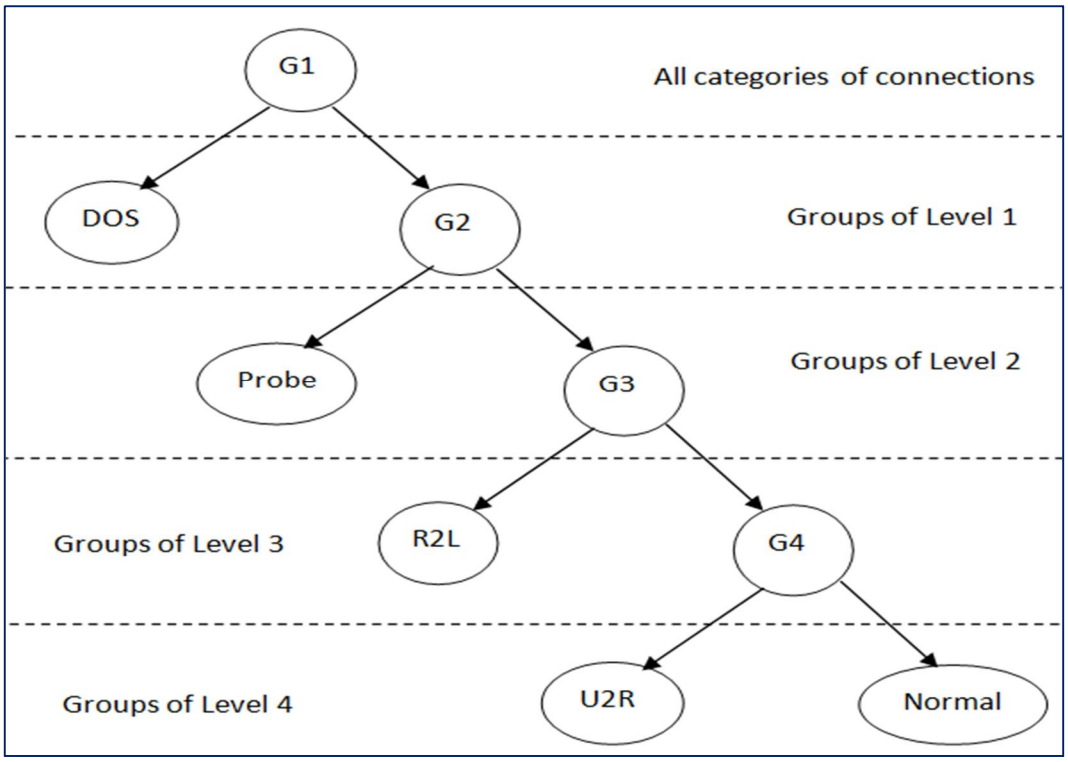

Figure 33 Le résultat de notre algorithme hiérarchique de clustering

Cette figure montre la ressemblance entre le comportement normal et les attaques R2L et U2R, ce qui justifie le faible taux de détection de ces deux catégories d'attaques dans la plupart des travaux de détection d'intrusion.

6.1.2. Sélectionner le meilleur classificateur pour chaque niveau de l'arbre binaire

Après avoir défini les différents groupes de chaque niveau de l'arbre binaire, nous sélectionnons le meilleur classificateur pour chaque niveau de notre arbre binaire. Comme le montre la figure 34, dans le premier niveau, nous sélectionnons le meilleur classificateur pour la classification des connexions réseau en deux catégories: DOS et G2 qui regroupe Probe, R2L, U2R et Normal. Dans le deuxième niveau, nous sélectionnons le meilleur classificateur pour la classification de G2 en G3 et Probe. Au troisième niveau, nous sélectionnons le meilleur classificateur pour la classification de G3 en G4 et R2L. Dans le dernier niveau, nous sélectionnons le meilleur classificateur pour la classification de G4 en U2R et Normal.

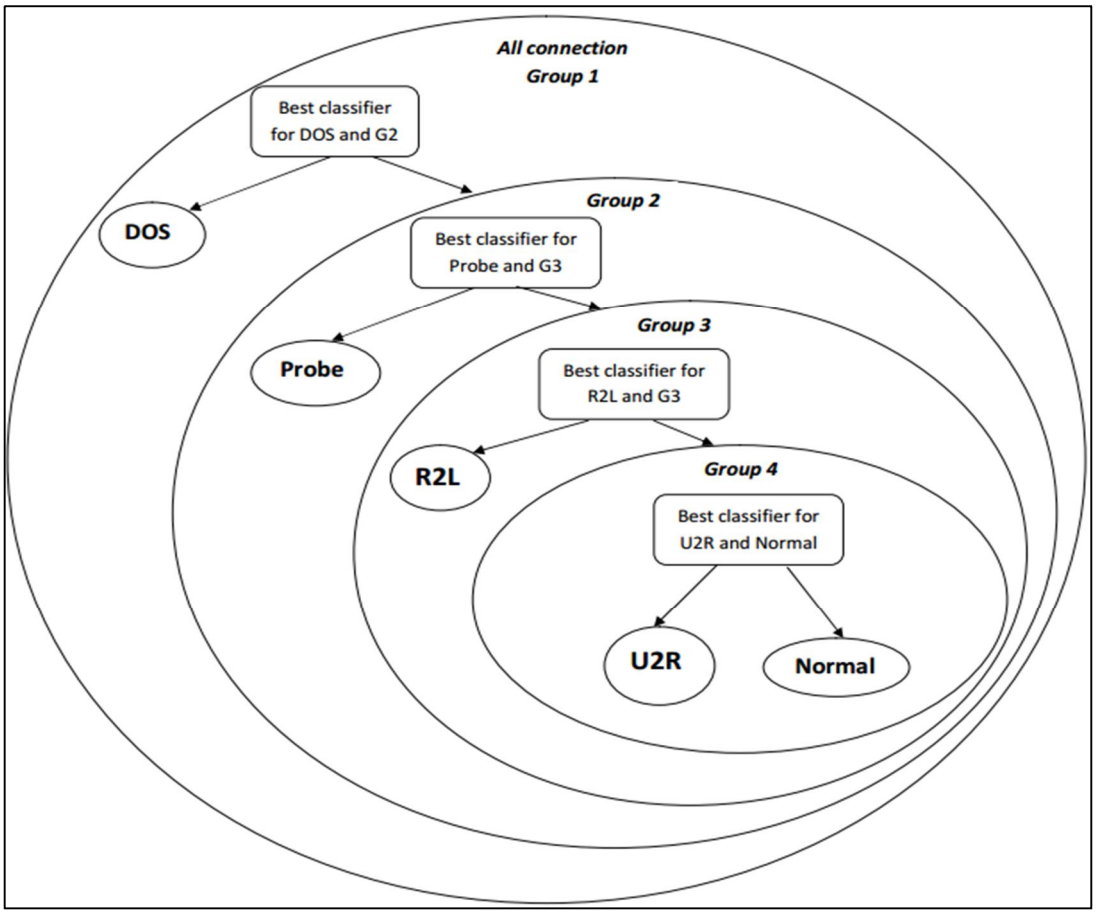

Figure 34 La structure générale de notre nouveau modèle hiérarchique

6.1.3. Le mode de fonctionnement de notre nouveau modèle hiérarchique

Le mode de fonctionnement de notre nouveau modèle hiérarchique se compose de deux phases: la phase d'apprentissage et la phase de test.

6.1.3.1. La phase d'apprentissage

Dans cette phase, on forme notre modèle dans le but de le préparer pour la phase de test. Différents ensembles de données d'apprentissage sont utilisés, où nous adaptons l'étiquette de chaque ensemble de données pour le niveau approprié comme suit:

- **Former le premier niveau:** nous formons le classificateur du premier niveau avec tout l'ensemble de données d'apprentissage, où toutes les attaques DoS sont étiquetées comme "DoS" et toutes les autres connexions réseau comme "G2".
- **Former le deuxième niveau:** pour former le classificateur du deuxième niveau, nous éliminons toutes les attaques DoS de l'ensemble de données d'apprentissage. Ensuite, nous étiquetons toutes les attaques d'exploration comme "Probe" et le reste des connexions réseau comme "G3".

- **Former le troisième niveau:** pour former le classificateur du troisième niveau, nous éliminons toutes les attaques DoS et Probe de l'ensemble de données d'apprentissage. Ensuite, nous étiquetons toutes les attaques R2L comme "R2L" et le reste des connexions réseau comme "G4".
- **Former le quatrième niveau:** le classificateur du quatrième niveau est formé avec uniquement les connexions U2R et Normal, où nous étiquetons toutes les attaques U2R comme "U2R" et les connexions normales comme "Normal".

6.1.3.2. La phase de test

Dans cette phase, nous testons la performance de notre modèle après l'achèvement de la phase d'apprentissage. Seul l'ensemble de données de test est utilisé pour accomplir cette étape. Ce processus est similaire à la recherche dans un arbre binaire. Au début, nous testons la connexion avec le classificateur du premier niveau de l'arbre binaire, si elle est classée comme DoS, nous arrêtons le processus, sinon nous testons cette connexion avec le classificateur du deuxième niveau. Si le classificateur du deuxième niveau classe cette connexion en tant qu'attaque Probe nous arrêtons le processus, sinon nous la testons avec le classificateur du troisième niveau. Si le classificateur du troisième niveau classe la connexion comme R2L nous arrêtons le processus, sinon nous testons cette connexion avec le classificateur du quatrième niveau. Enfin, le classificateur du quatrième niveau classe cette connexion soit comme comportement normal ou attaque U2R.

6.2. Expérimentation

Cette section est divisée en quatre sous-sections. Dans la première sous-section, nous détaillons l'ensemble de données d'apprentissage et de test. Dans la deuxième sous-section, nous présentons la structure pratique de notre modèle. Dans la troisième sous-section, nous donnons une brève analyse expérimentale avec NSL-KDD. La quatrième sous-section présente une analyse expérimentale détaillée avec KDD'99.

L'outil de data mining Weka (Witten et al., 2011) est utilisé pour l'implémentation de notre modèle. Les résultats sont obtenus dans un PC Windows avec Core 2 Duo 2,0 GHz et 2 Go de RAM.

Pour évaluer la performance de notre modèle, deux types de métrique de performance sont utilisés. Le premier dépend de la catégorie de connexion, il représente le taux de détection des quatre catégories de connexions réseau (DOS, Probe, U2R, R2L) ainsi que le taux du

vrai négatif du comportement normal. Pour calculer le taux de détection on utilise l'équation suivante: $TD_A = \frac{VP_A}{VP_A+FN_A}$ (6.1). Pour calculer le taux de vrai négatif on utilise l'équation suivante: $TVN_A = \frac{VN_A}{VN_A+FP_A}$ (6.2). Le deuxième type de métrique représente les indicateurs de performance globaux. Il comprend le taux de fausses alarmes (TFA), le taux de détection global (TD) et l'exactitude (Accuracy). Pour calculer ces mesures, nous utilisons les équations suivantes:

$$TFA = \frac{FP_{Normal}}{VN_{Normal}+ FP_{Normal}} \qquad (6.3)$$

$$TD = \frac{VP_{DOS}+VP_{Probe}+VP_{R2L}+VP_{U2R}}{VP_{DOS}+VP_{Probe}+VP_{R2L}+VP_{U2R}+ FN_{DOS}+ FN_{Probe}+ FN_{R2L}+ FN_{U2R}} \qquad (6.4)$$

$$\text{Exactitude } \textit{(Accuracy)} = \frac{VP_{DOS}+VP_{Probe}+VP_{R2L}+VP_{U2R}+ VN_{Normal}}{VP_{DOS}+VP_{Probe}+VP_{R2L}+VP_{U2R}+ FN_{DOS}+ FN_{Probe}+ FN_{R2L}+ FN_{U2R}+VN_{Normal}+FP_{Normal}} \qquad (6.5)$$

6.2.1. Les ensembles de données d'apprentissage et de test

6.2.1.1. KDD'99

En raison de la grande taille de l'ensemble de données d'apprentissage KDD'99_10%, nous avons créé quatre ensembles de données d'apprentissage qui contiennent respectivement 40.000, 30.000, 20.000, 10.000 d'enregistrements. Pour réduire la taille des données d'apprentissage du KDD'99_10% tous les enregistrements redondants sont supprimés. Puis, la sélection aléatoire est utilisée pour sélectionner les enregistrements Normal et DOS (Neptune). Le tableau suivant résume la distribution des attaques et du comportement normal de nos quatre ensembles de données d'apprentissage.

	DOS	**Normal**	**Probe**	**R2L**	**U2R**
Training Data 1	24819	12000	2130	999	52
Training Data 2	18819	8000	2130	999	52
Training Data 3	12819	4000	2130	999	52
Training Data 4	4819	2000	2130	999	52

Tableau 15 La distribution des connexions réseau de nos quatre ensembles de données d'apprentissage extraits du KDD'99

L'ensemble de données KDD'99 Test est utilisé pour évaluer la performance de classification de notre modèle. Les deux caractéristiques num_outbound_cmds et is_host_login sont éliminées en raison de leurs valeurs identiques dans l'ensemble de

données d'apprentissage. Pour normaliser ces ensembles de données, le codage ASCII est utilisé pour convertir les données symboliques en des valeurs numériques. Ensuite, chaque donnée x_i de la propriété J est normalisée en utilisant l'équation suivante:

$$\overline{x_i(j)} = \frac{x_i(j) - min(x(j))}{max(x(j)) - min(x(j))} \qquad (6.6)$$

6.2.1.2. NSL-KDD

En raison de la grande taille de KDDTrain+, nous avons créé notre propre ensemble de données d'apprentissage qui contient 40.000 enregistrements. Pour réduire la taille de KDDTrain+ la sélection aléatoire est utilisée pour sélectionner les enregistrements normaux et DOS (Neptune). Le tableau suivant résume la répartition des attaques et des comportements normaux de notre ensemble de données d'apprentissage NSL-KDD.

DoS	Probe	R2L	U2R	Normal
23953	3000	995	52	12000

Tableau 16 La distribution des connexions réseau de notre ensemble de données d'apprentissage extrait du NSL-KDD

L'ensemble de données KDDTest+ est utilisé pour évaluer la performance de notre modèle. Comme pour le KDD'99, pour normaliser les ensembles de données du NSL-KDD, le codage ASCII est utilisé pour convertir les données symboliques à des valeurs numériques. Ensuite, chaque donnée x_i de la caractéristique J est normalisée en utilisant l'équation suivante:

$$\overline{x_i(j)} = \frac{x_i(j) - min(x(j))}{max(x(j)) - min(x(j))} \qquad (6.6)$$

En se basant sur la classification des attaques fourni dans l'annexe 2, notre ensemble de données d'apprentissage du KDD'99, notre ensemble de données d'apprentissage du NSL-KDD, l'ensemble de données de test de KDD'99 et l'ensemble de données KDDTest+ sont étiquetés d'après les cinq catégories de connexions. Où toutes les attaques DOS sont étiquetées comme "DOS", toutes les attaques d'exploration sont étiquetées comme "Probe", toutes les attaques R2L sont étiquetées comme "R2L", toutes les attaques U2R sont étiquetées comme "U2R" et toutes les connexions normales sont étiquetées comme "Normal".

6.2.2. La structure pratique de notre modèle

Le choix des meilleurs classificateurs pour notre modèle est une phase très critique à cause de son impact sur la performance de notre modèle. Ce choix peut suivre différents principes. Le premier consiste à adapter notre modèle aux risques engendrés par les attaques. Le second est d'adapter notre modèle à la fréquence d'une catégorie d'attaque. Le troisième est d'adapter notre modèle à tous types de trafic réseau, où nous sélectionnons les classificateurs qui donnent les meilleurs résultats pour toutes les catégories des attaques de n'importe quel trafic réseau.

Si nous suivons le premier principe, le choix de nos classificateurs dépend fortement de la nature et les risques engendrés par des attaques. Par exemple, dans le réseau de commerce qui comprend l'hébergement web, e-commerce et des services de marketing en ligne la disponibilité représente le point le plus important dans la politique de sécurité. Par conséquent les attaques DoS et DDoS représentent la catégorie des attaques la plus dangereuse pour notre réseau. Dans ce cas, il est fortement recommandé d'utiliser le classificateur qui donne le plus haut taux de détection pour la catégorie DOS. Dans un autre cas, si nous avons un réseau militaire ou gouvernemental la confidentialité représente le point le plus important dans la politique de sécurité. Par conséquent, les attaques U2R représentent la catégorie la plus dangereuse. Donc, nous devons trouver les trois classificateurs qui donnent le meilleur taux de correcte classification pour respectivement G2, G3 et la catégorie U2R.

Le deuxième principe consiste à construire un IDS qui donne le taux d'exactitude le plus élevé pour un réseau particulier. Par exemple, si nous avons un réseau de commerce, les attaques les plus fréquentes sont les attaques DoS. Donc, notre IDS détecte plus d'attaques et donne le meilleur taux d'exactitude si nous utilisons le meilleur classificateur dans la détection de la catégorie des attaques DoS.

Dans notre travail, nous adoptons le troisième principe, où nous essayons de construire un système de détection d'intrusion qui donne le meilleur taux d'exactitude avec tout trafic réseau. Notre modèle ne dépend pas des risques ou de la fréquence des attaques. Pour cette raison, chaque classificateur sélectionné doit donner le taux de détection le plus élevé pour l'un des quatre niveaux de notre modèle à la fois pour KDD'99 et NSL-KDD.

Dans le but de sélectionner les meilleurs classificateurs pour les différents niveaux de notre nouveau modèle hiérarchique, nous avons effectué une série d'expérimentation. Ces expérimentations sont pour but de sélectionner les classificateurs qui donnent les meilleurs taux de correcte classification pour les différents niveaux de notre modèle à la fois pour KDD'99 et NSL-KDD. Les différents classificateurs comparés sont: "Naive Bayes (NB)" (John and Langley, 1995), "C4.5 Decision Tree (DT)" (Quinlan, 1993), "Support Vector Machine (SVM)" (Chang and lin, 2001), "Repeated Incremental Pruning to Produce Error Reduction (RIPPER)" (Cohen, 1995), "Random Forests (RF)" (Breiman, 2001), "Multilayer Perceptrons (MLP)" (Bishop, 1996), "Self-Organizing Feature Map Network (SOFM)" (Kohonen, 2001), "Naive Bayes Multinomial (NBM)" (Mccallum and Nigam, 1998), "Ripple-down rule learner (Ridor)" (Gaines and Compton, 1995), "RBF Network (RBFN)" (Bugmann, 1998), "Random Tree (RT)" (Witten et al., 2011), "Sequential Minimal Optimization (SMO)" (Platt,1999).

Pour l'expérimentation en NSL-KDD, nous avons utilisé les données d'apprentissage détaillé dans le tableau 16 ci-dessus. Chaque classificateur a 37 entrées qui représentent les 41 caractéristiques de NSL-KDD sans num_outbound_cmds, is_host_login, land et wrong_fragment.

Pour l'expérimentation en KDD'99, nous avons utilisé les différents ensembles de données d'apprentissage détaillées dans le tableau 15 ci-dessus. Chaque classificateur dispose de 39 entrées qui représentent les 41 caractéristiques de KDD'99 sans num_outbound_cmds et is_host_login.

Après avoir analysé les résultats de la classification des différents classificateurs utilisés dans nos expérimentations, nous avons sélectionné les classificateurs qui donnent une grande précision pour le NSL-KDD et le KDD'99. Les classificateurs sélectionnés pour le premier niveau, le deuxième niveau, le troisième niveau, le quatrième niveau sont respectivement : "Repeated Incremental Pruning to Produce Error Reduction", "Naive Bayes Multinomial", "Ripple-down rule learner" et "Random Tree".

Comme le montre la figure suivante, dans le premier niveau, nous utilisons "Repeated Incremental Pruning to Produce Error Reduction" pour classer les connexions réseau en deux catégories DoS et G2. Dans le deuxième niveau, nous utilisons "Naive Bayes Multinomial" pour classer les enregistrements de G2 en G3 et Probe. Au troisième niveau,

nous utilisons "Ripple-down rule learner" pour classer les enregistrements de G3 en G4 et R2L. Dans le dernier niveau, nous utilisons "Random Tree" pour classer les connexions de G4 en U2R et Normal.

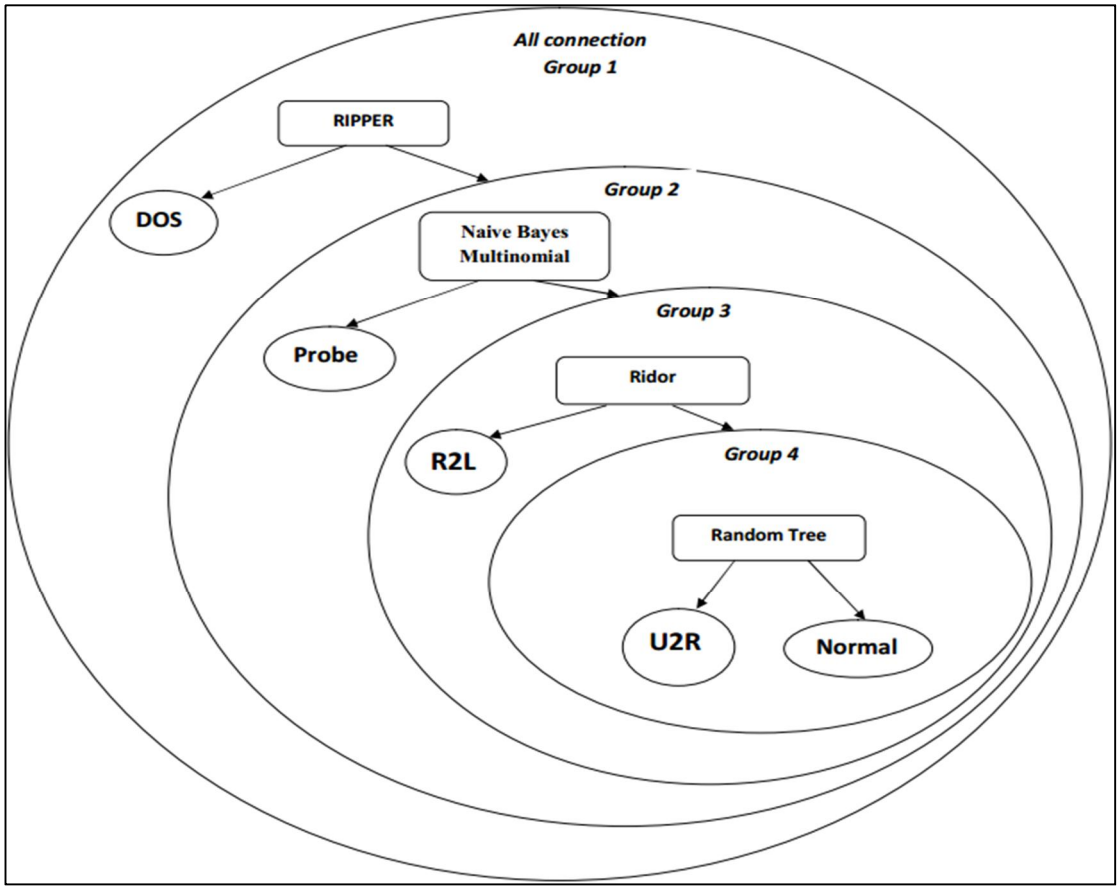

Figure 35 La structure pratique de notre modèle

6.2.3. L'analyse expérimentale avec NSL-KDD

Dans cette sous-section, nous détaillons la mise en œuvre de notre modèle avec NSL-KDD. Ensuite, nous comparons nos résultats avec les résultats de certains classificateurs bien connus. Dans cette étude comparative, nous utilisons les données détaillées dans le tableau 16 ci-dessus comme données d'apprentissage, et tout le KDDTest+ comme ensemble de données de test.

6.2.3.1. Les performances des classificateurs de chaque niveau

Les détails des différents paramètres des classificateurs utilisés pour la mise en œuvre de notre modèle dans l'outil de data mining Weka (Witten et al, 2011) sont résumés dans le tableau ci-dessous.

RIPPER		Naive Bayes Multinomial		Ridor		Random Tree	
Paramètre	**Valeur**	**Paramètre**	**Value**	**Paramètre**	**Valeur**	**Paramètre**	**Valeur**
Folds	3	-	-	Folds	4	KValue	8
minNo	3	-	-	majorityClass	false	allowUnclassifiedInstances	false
optimizations	3	-	-	minNo	2	MaxDepth	0
seed	1	-	-	seed	4	MinNum	3
usePruning	true	-	-	shuffle	2	seed	1
-	-	-	-	wholeDataErr	false	numFolds	0

Tableau 17 Les paramètres des classificateurs de notre modèle avec NSL-KDD

Les performances de ces classificateurs pour la classification des connexions de KDDTest+ sont détaillées dans le tableau ci-dessous.

RIPPER				Naive Bayes Multinomial				Ridor				Random Tree		
	DOS	G2			Probe	G3			R2L	G4			Normal	U2R
DOS	6448	1010		Probe	2258	163		R2L	894	1860		Normal	9697	14
G2	287	14799		G3	415	12250		G4	27	9884		U2R	71	129

Tableau 18 Les performances des différents classificateurs de notre modèle pour la classification des connexions de KDDTest +

Comme le montre la figure 35, nous avons utilisé RIPPER comme classificateur du premier niveau, où il donne un taux d'exactitude égal à 94,24%, "Naïve Bayes Multinomial" comme classificateur du deuxième niveau avec un taux d'exactitude égal à 96,16%, Ridor comme classificateur du troisième niveau avec un taux d'exactitude égal à 85,10%, et pour le dernier niveau, nous avons utilisé "Random Tree" avec un taux d'exactitude égale à 99,10%.

6.2.3.2. Étude comparative

Pour évaluer les performances de notre modèle, nous l'avons comparé avec certains classificateurs bien connus tels que: Naïve Bayes (NB), Arbre de décision C4.5 (DT), Machine à vecteur de support (SVM), "Repeated Incremental Pruning to Produce Error Reduction (RIPPER)", "Perceptron multicouche (MLP)". Dans cette comparaison, nous avons utilisé l'ensemble de données d'apprentissage détaillé dans le tableau 16 ci-dessus comme un ensemble de données de formation et tout KDDTest + comme un ensemble de données de test. Ces classificateurs doivent classer chaque enregistrement de KDDTest+ dans l'une des cinq catégories de connexions réseau (DOS, Probe, R2L, U2R et Normal).

Les paramètres TD_{DOS}, TD_{Probe}, TD_{U2R}, TD_{R2L}, TVN_{Normal}, TFA, TD et Exactitude sont utilisés pour comparer les performances de notre modèle et les autres classificateurs. Le tableau suivant détaille les performances de notre modèle et les autres classificateurs.

	Notre modèle	**DT**	**RIPPER**	**NB**	**SVM**	**MLP**
TD_{DOS}	86,46%	86,70%	85,75%	73,45%	85,91%	86,30%
TD_{Probe}	88,19%	59,77%	56,38%	83,11%	63,28%	71,71%
TD_{R2L}	29,63%	15,54%	15,76%	13,51%	28,98%	17,50%
TD_{U2R}	18,00%	3,50%	10,00%	26,00%	0,00%	8,50%
TVN_{Normal}	96,13%	92,66%	96,22%	86,53%	96,61%	96,89%
Exactitude	83,26%	76,94%	77,88%	72,38%	80,37%	80,20%
TD	73,52%	65,05%	64,01%	61,67%	68,08%	67,57%
TFA	3,87%	7,34%	3,78%	13,47%	3,39%	3,11%

Tableau 19 La performance de notre modèle et les autres classificateurs pour le KDDTest+

Comme le montre la figure suivante, notre modèle donne le plus haut taux de détection des attaques Probe avec un taux égal à 88,19% et R2L avec un taux égal à 29,63%. De plus, il donne le deuxième plus élevé taux de détection des attaques DoS avec un taux égal à 86,46% et le deuxième plus élevé taux de vrai négatif pour le comportement Normal avec un taux égal à 96,13%. Globalement, notre modèle hiérarchique donne le plus élevé taux global de détection (TD) avec 73,52%, le plus élevé taux d'exactitude avec 83,26% et le deuxième plus faible taux de fausse alarme avec 3,87%. L'amélioration de la classification représente 653 enregistrements correctement classés de plus par rapport au meilleur classificateur utilisé dans cette étude comparative.

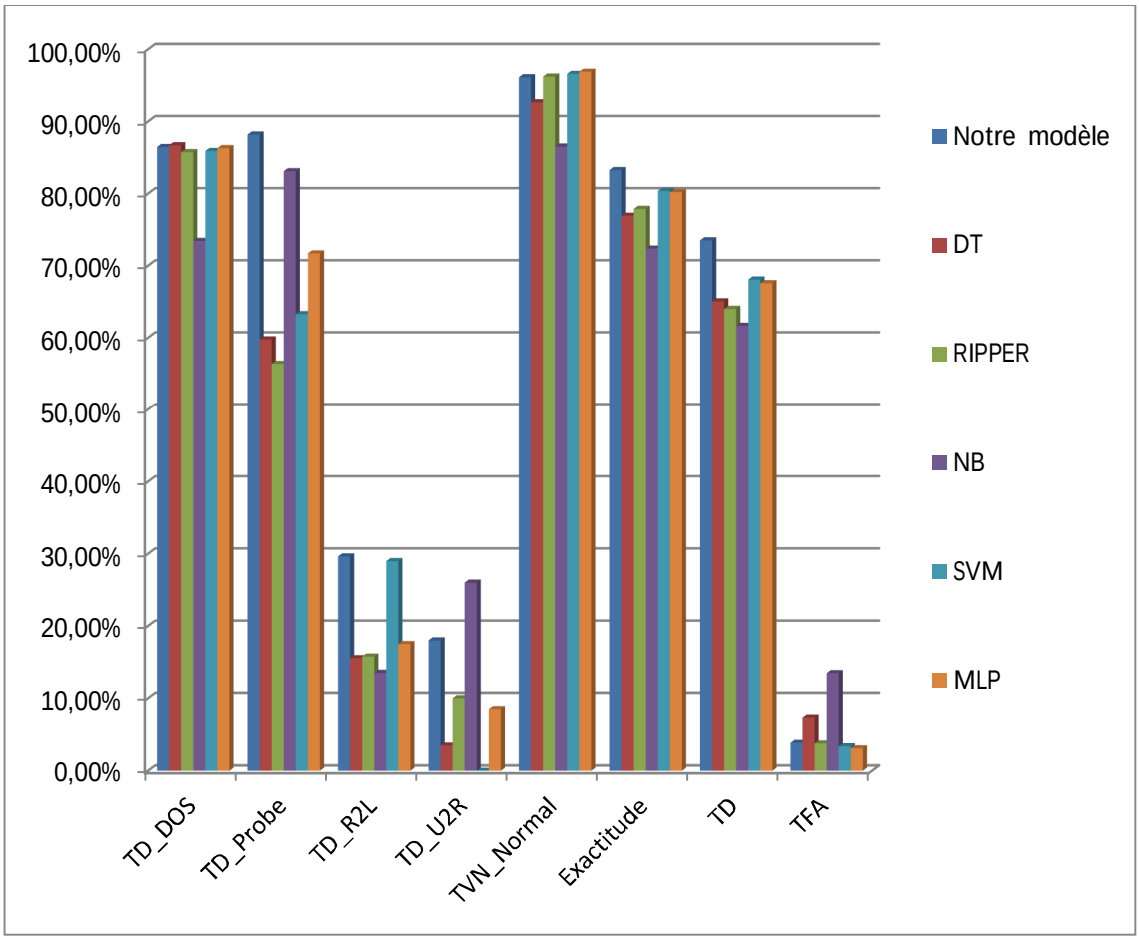

Figure 36 La performance de notre modèle et les autres classificateurs pour le KDDTest +

L'étude comparative entre notre modèle et les autres classificateurs montre que notre modèle a atteint les objectifs de notre travail avec NSL-KDD, où il donne le meilleur taux d'exactitude avec 83,26%, il combine un taux de détection élevé et un taux de fausse alarme faible, et il classe tout connexion dans une des cinq catégories de connexion réseau (DOS, Probe, R2L, U2R et normal).

6.2.4. L'analyse expérimentale avec KDD'99

Dans cette sous-section, nous détaillons la mise en œuvre de notre modèle avec le KDD'99 qui représente l'ensemble de données d'évaluation des méthodes de détection d'intrusion le plus utilisé. Ensuite, nous comparons les résultats obtenus par notre modèle avec les résultats des travaux récents qui ont utilisé tout l'ensemble de données KDD'99 Test comme ensemble de données de test. De plus nous comparons notre modèle avec certains classificateurs bien connus.

6.2.4.1. La performance de classification pour chaque niveau

Les détails des différents paramètres des classificateurs utilisés pour la mise en œuvre de notre modèle dans Weka sont résumés dans le tableau ci-dessous.

RIPPER		Naïve Bayes Multinomial		Ridor		Random Tree	
Paramètre	**Valeur**	**Paramètre**	**Valeur**	**Paramètre**	**Valeur**	**Paramètre**	**Valeur**
Folds	3	-	-	Folds	3	KValue	12
minNo	2	-	-	majorityClass	false	allowUnclassifiedInstances	false
optimizations	2	-	-	minNo	2	MaxDepth	0
seed	1	-	-	seed	1	MinNum	1
usePruning	true	-	-	shuffle	1	seed	1
-		-	-	wholeDataErr	false	numFolds	0

Tableau 20 Les paramètres des classificateurs de notre modèle avec KDD'99

Les données d'apprentissage utilisé avec RIPPER, "Naïve Bayes Multinomial", Ridor, "Random Tree" sont respectivement "Training Data1", " Training Data1" (nous éliminons les deux caractéristiques Land et wrong_fragment), "Training Data4" (nous éliminons les deux caractéristiques Land et wrong_fragment) et "Training Data1". La performance de ces couples (classificateur, ensemble de données d'apprentissage) pour la classification de l'ensemble de données KDD'99 Test est détaillée dans le tableau ci-dessous.

RIPPER			Naïve Bayes Multinomial			Ridor			Random Tree		
	DOS	G1		Probe	G3		R2L	G4		Normal	U2R
DOS	228596	1257	Probe	3722	444	R2L	5890	10299	Normal	60568	25
G1	351	80825	G3	570	76440	G4	584	60237	U2R	160	68

Tableau 21 La performance des différents classificateurs de notre modèle pour l'ensemble de données KDD'99 Test

Comme le montre le tableau 21, nous avons utilisé RIPPER comme classificateur du premier niveau où il donne un taux d'exactitude égal à 99,48%, "Naïve Bayes Multinomial" comme classificateur du deuxième niveau avec un taux d'exactitude égal à 98,75%, Ridor comme classificateur du troisième niveau avec un taux d'exactitude égal à 85,86%, et dans le dernier niveau, nous avons utilisé "Random Tree" avec un taux d'exactitude égal à 99,69%.

6.2.4.2. Le temps d'apprentissage et de test

Le temps d'apprentissage de notre modèle est égal à la somme des temps d'apprentissage des quatre classificateurs. Dans notre expérimentation, le temps d'apprentissage de RIPPER, "Naïve Bayes Multinomial", Ridor, "Random Tree" est respectivement 75,84 secondes, 0,1 seconde, 0,62 seconde et 0,77 seconde. Par conséquent, le temps nécessaire pour l'apprentissage de notre modèle est 77,33 secondes ce qui représente un temps très court par rapport à certains travaux connexes comme Wanga et al. (2010) qui a besoin de 2125,4 secondes. Donc, notre modèle a besoin de très peu de temps pour être déployé. Le temps nécessaire pour tester tout l'ensemble de données KDD'99 Test est de 14,71 secondes. Vu que l'ensemble de données KDD'99 Test contient 311029 enregistrements, alors le temps moyen nécessaire pour traiter un enregistrement est 47,30 microsecondes. Ce qui prouve la rapidité de notre modèle dans le traitement du trafic réseau. La rapidité de traitement du trafic réseau est obtenue grâce à la structure d'arbre binaire qui représente la meilleure et la plus rapide structure de recherche.

6.2.4.3. Étude comparative

Afin de positionner notre travail par rapport aux modèles de détection d'intrusion existants, nous présentons une étude comparative. Dans cette étude comparative, nous comparons les résultats de notre modèle avec les résultats de certains IDSs adaptatifs existants ainsi que certains modèles récents qui ont utilisé tout le KDD'99 Test comme un ensemble de données de test pour leurs modèles. De plus, nous comparons notre modèle avec quelques classificateurs bien connus comme l'arbre de décision C4.5 (DT), la machine à vecteur de support (SVM), " Repeated Incremental Pruning to Produce Error Reduction (RIPPER)" , "Perceptron multicouche (MLP)" où nous utilisons le "Training Data 1" détaillé précédemment comme ensemble de données d'apprentissage, et tout l'ensemble de données KDD'99 Test comme ensemble de données de test.

Pour l'implémentation de notre modèle, nous avons utilisé les ensembles de données d'apprentissage détaillé dans la section précédente comme ensembles de données d'apprentissage et tout le KDD'99 Test comme ensemble de données de test. Les paramètres TD_{DOS}, TD_{Probe}, TD_{U2R}, TD_{R2L}, TVN_{Normal}, TFA, TD et Exactitude sont utilisés pour comparer les performances de notre modèle et les autres travaux. Dans un premier temps, nous comparons les taux de correcte classification des catégories de connexions. Ensuite, nous comparons les mesures globales de performance. Le tableau suivant détaille la

performance de notre modèle et les autres modèles de détection d'intrusion pour la classification des cinq catégories de connexion.

	TVN_{Normal}	TD_{DOS}	TD_{Probe}	TD_{R2L}	TD_{U2R}
Toosi and Kahani (2007)	98,20%	99,50%	84,10%	31,50%	14,10%
Xiang et al. (2008)	96,80%	98,66%	93,40%	46,97%	71,43%
Wanga et al. (2010)	99,08%	96,70%	80,00%	58,57%	76,92%
Horng et al. (2011)	99,30%	99,50%	97,50%	28,80%	19,70%
Khor and Ting (2012)	97,40%	97,80%	73,30%	48,20%	87,30%
Badran and Rockett (2012)	99,50%	96,99%	78,01%	5,59%	11,40%
Koc et al. (2012)	-	99,60%	-	-	-
SVM	98,95%	97,10%	74,84%	9,69%	8,77%
RIPPER	98,13%	97,42%	80,17%	7,00%	10,53%
DT	99,10%	97,42%	85,50%	10,77%	4,82%
MLP	98,13%	97,42%	80,17%	7,00%	10,53%
Notre modèle	98,57%	99,45%	84,11%	36,17%	8,77%

Tableau 22 La performance de notre modèle et les autres modèles et classificateurs pour le KDD'99 Test

Comme le montre la figure 37, le meilleur modèle pour la classification des comportements normaux est le modèle de Badran et Rockett (2012) avec un taux égal à 99,50%, les attaques DoS est le modèle de Koc et al. (2012) avec un taux égal à 99,60%, les attaques Probe est le modèle de Horng (2011) avec un taux égal à 97,50%, les attaques R2L est le modèle de Wanga et al. (2010) avec un taux égal à 58,57%, et les attaques U2R est le modèle de Khor et Ting (2012) avec un taux égal à 87,30%. Notre modèle ne donne pas le taux de correcte classification le plus élevé pour aucune catégorie de connexion, mais il est proche du taux le plus élevé pour la plupart d'entre eux, où il donne un taux égal à 98,57% pour Normal, 99,45% pour DOS, 84,11% pour Probe, 36,17% pour R2L et 8,77% pour U2R.

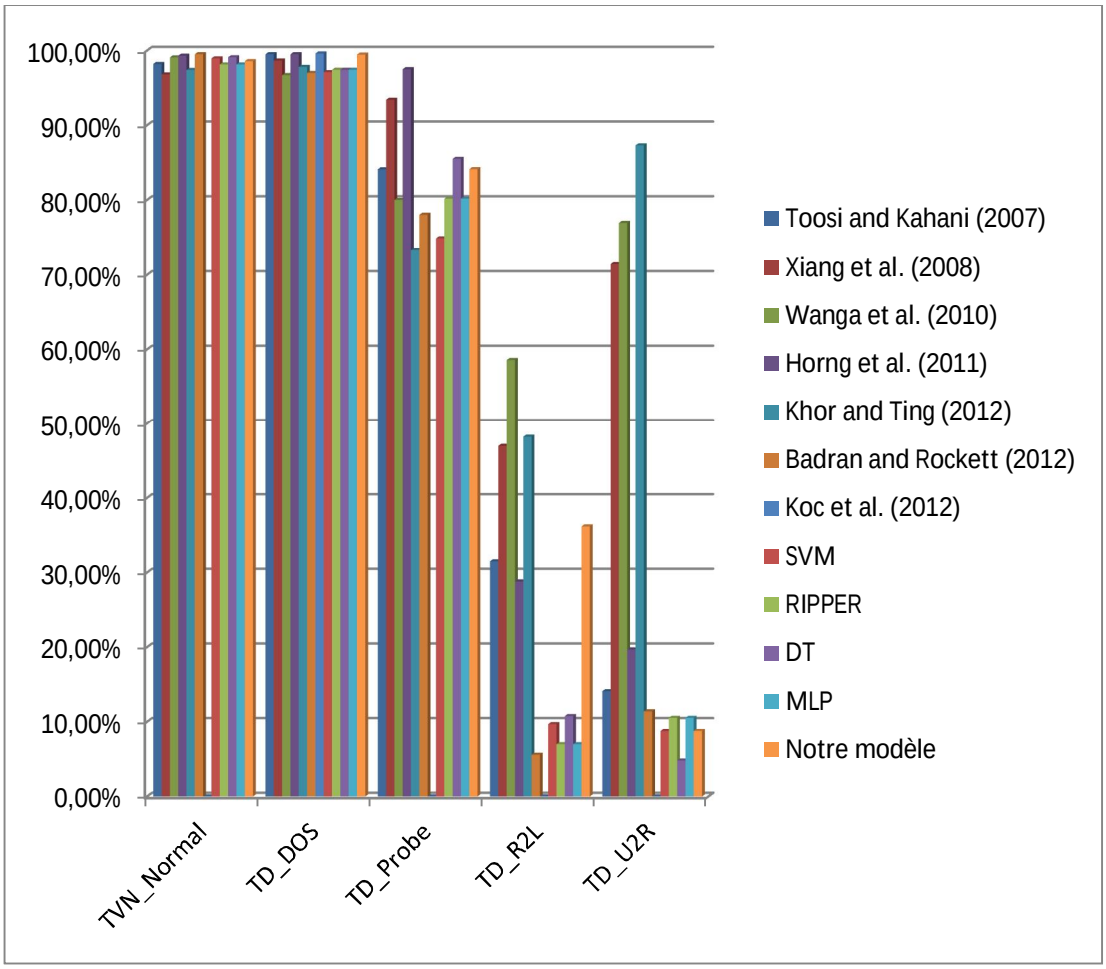

Figure 37 La performance de notre modèle et les autres modèles et classificateurs pour le KDD'99 test

Après avoir comparé les performances de notre modèle avec les autres modèles et classificateurs pour la classification des différentes catégories de connexion, nous comparons les mesures globales de performance: le taux global de détection (TD), le taux de fausses alarmes (TFA) et le taux d'exactitude. Le tableau suivant résume les métriques de performance globales de notre modèle ainsi que les autres modèles et classificateurs.

	TD	TFA	Exactitude
Toosi and Kahani (2007)	94,77%	1,90%	95,30%
Xiang et al. (2008)	93,93%	3,20%	94,49%
Wanga et al.2010)	93,94%	0,92%	94,94%
Horng et al. (2011)	94,82%	0,70%	95,70%
Khor and Ting (2012)	94,18%	2,60%	94,80%
Badran and Rockett (2012)	90,69%	0,50%	92,41%
Koc et al. (2012)	-	-	93,72%
SVM	91,00%	1,05%	92,55%
RIPPER	91,21%	1,87%	92,56%
DT	91,54%	0,90%	93,01%
MLP	91,21%	1,87%	92,56%
Notre modèle	95,02%	1,43%	95,72%

Tableau 23 Les métriques globales de performance de notre modèle ainsi que les autres modèles et classificateurs pour le KDD'99 Test

Comme l'illustre la figure ci-dessous, notre modèle fournit le taux de détection le plus élevé avec un taux égal à 95,02%, le taux d'exactitude le plus élevé avec 95,72% et un faible taux de fausses alarmes égal à 1,43%. Ces métriques de performance globales montrent la haute performance de notre modèle. Notre modèle est le modèle le plus précis par rapport aux autres modèles et classificateurs avec un taux d'exactitude égal à 95,72%. Ce qui signifie que notre modèle a correctement classé 63 connexions de plus par rapport aux meilleurs classificateurs et modèles utilisés dans cette étude comparative.

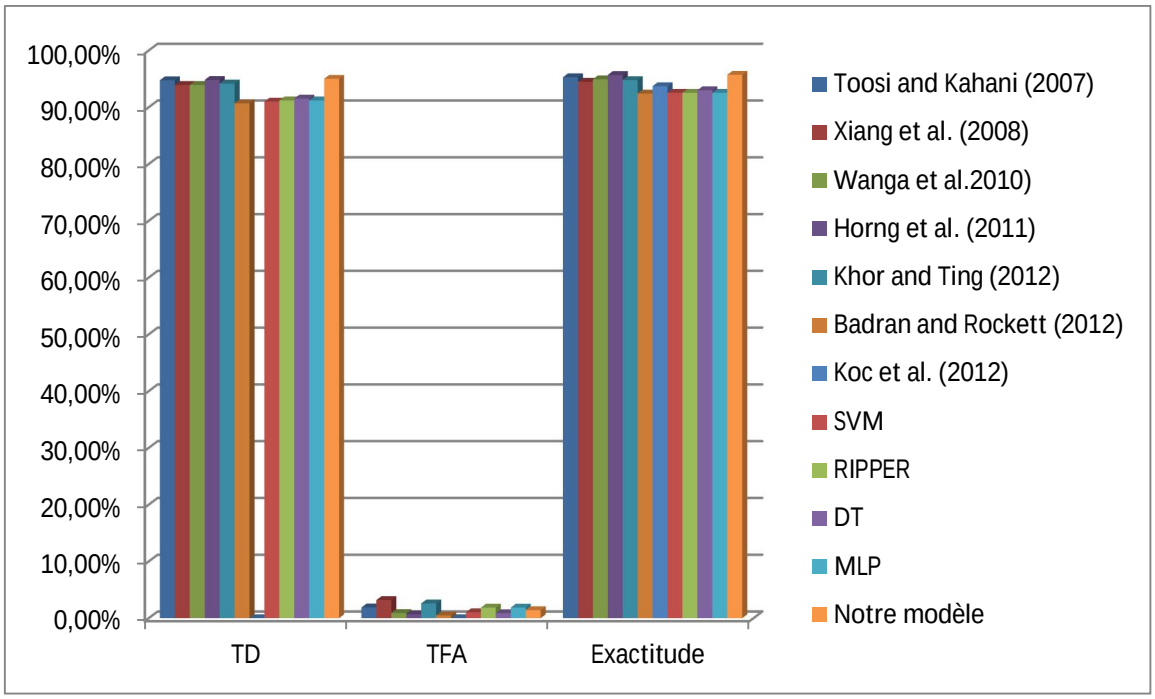

Figure 38 Les métriques globales de performance de notre modèle et les autres modèles et classificateurs pour le KDD'99 Test

L'étude comparative entre notre modèle et les autres modèles et classificateurs montre que notre modèle a atteint les objectifs de notre travail avec l'ensemble de données KDD'99, où il donne la meilleure performance avec un taux d'exactitude égal à 95,72%, il combine un taux de détection élevé et un taux de fausses alarmes faible, et il classe toute connexion dans l'une des cinq catégories de connexion réseau (DOS, Probe, R2L, U2R, et Normal).

6.3. Conclusion

Dans ce chapitre, nous avons proposé un nouveau système de détection hiérarchique basé sur un arbre binaire de classificateurs. Le but de notre travail est de concevoir un modèle de détection d'intrusion qui possède une très grande capacité de généralisation, ce qui lui permet de détecter les nouvelles formes d'attaques. Pour construire l'arbre binaire, nous avons regroupé les catégories de connexions hiérarchiquement en fonction de la proportion de faux positifs et faux négatifs générés entre chaque deux catégories. Le modèle crée est un arbre binaire avec quatre niveaux. Dans le premier niveau, nous avons utilisé RIPPER pour classer les connexions réseau en deux catégories de connexion: DOS et G2 qui regroupe Probe, R2L, U2R et Normal. Puis, dans le deuxième niveau, nous avons utilisé "Naïve Bayes Multinomial" pour classer les connexions réseau de G2 en Probe et G3 qui regroupe R2L, U2R et Normal. Après, nous avons utilisé Ridor pour classer les connexions réseau de G3 en R2L et G4 qui regroupe U2R et Normal. Pour le dernier niveau, nous avons utilisé "Random Tree" pour classer les connexions réseau de G4 en U2R et Normal. Les expérimentations avec NSL-KDD ont montré la haute performance de notre modèle par rapport à certains classificateurs bien connus, où il donne le taux de détection le plus élevé pour R2L et Probe. De plus, il donne le taux de détection globale le plus élevé, le taux d'exactitude le plus élevé et un taux de fausses alarmes faible. Les expérimentations avec KDD'99 ont confirmé la haute performance de notre modèle par rapport aux travaux connexes, certains modèles récents de détection d'intrusion, ainsi que certains classificateurs bien connus. Notre modèle donne le taux global de détection le plus élevé, le taux d'exactitude le plus élevé et un taux de fausses alarmes faible. De plus, notre nouveau modèle hiérarchique est très rapide dans le traitement du trafic réseau, ce qui représente un autre avantage de plus.

Chapitre 7

Proposition 3 un système de détection d'intrusion avec un mode d'apprentissage continu

Le but de cette proposition est de créer un nouveau système de détection d'intrusion hiérarchique (HIDS) basé sur un mode d'apprentissage continu afin d'assurer l'adaptation de notre modèle. L'idée de cet article est d'utiliser deux classificateurs différents d'une manière itérative, où chaque itération représente un niveau dans le modèle créé. Pour assurer l'adaptation de notre modèle, nous ajoutons un nouveau niveau chaque fois que la somme des nouvelles attaques et le reste de l'ensemble de données d'apprentissage atteignent le seuil. Pour construire notre modèle, nous avons utilisé "Fuzzy Unordered Rule Induction Algorithm" et "Random Forests" comme classificateurs. L'expérimentation avec l'ensemble de données KDD'99 a montré la haute performance de notre modèle qui a montré sa capacité de combine un taux de détection élevé et un faible taux de fausse alarme. En outre, notre modèle donne le taux de détection le plus élevé et le plus grand taux d'exactitude par rapport à certains modèles bien connus dans la littérature de la détection d'intrusion. Ce travail a été l'objet de notre article « A new adaptive intrusion detection system based on the intersection of two different classifiers » (AHMIM and Ghoualmi-Zine, 2014).

7.1. La description du modèle

La plupart des travaux de détection d'intrusion utilisent les classificateurs du même niveau de façon isolée. Dans cet article, nous proposons une nouvelle approche qui combine deux différents classificateurs dans chaque niveau, où les itérations nécessaires pour construire notre modèle représentent le nombre de niveaux.

7.1.1. La base théorique de notre approche

Dans le domaine de data mining, nous avons différents types de classificateur basés sur différentes méthodes et techniques. Chaque classificateur peut classer chaque connexion réseau comme un comportement normal ou une attaque avec divers taux d'erreur comme l'illustre la figure suivante.

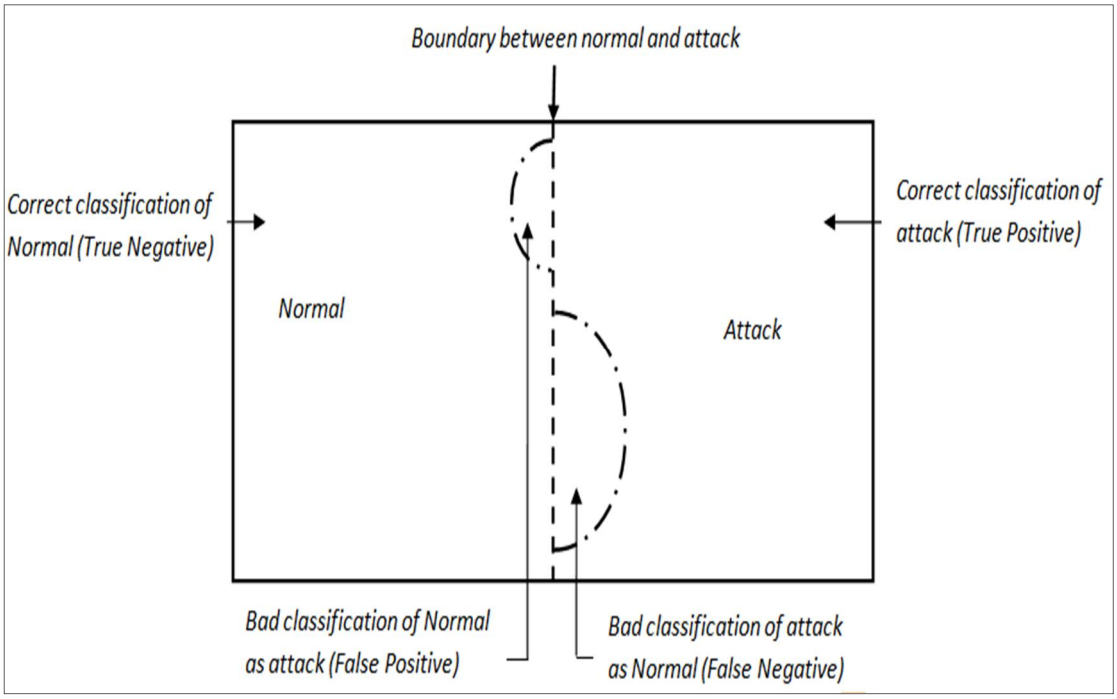

Figure 39 Le résultat de la classification d'une connexion réseau avec les techniques de data mining

La performance des différents types de classificateurs est mesurée par leurs capacités de classer chaque connexion dans la bonne catégorie. Comme le montre la figure 39, les quatre cas possibles sont: vrai négatif, faux négatif, faux positif et vrai positif. Le vrai négatif (VN) est la classification correcte de la classe normale. Le faux négatif (FN) est la mauvaise classification de la classe d'attaque comme une classe normale. Le faux positif (FP) est la mauvaise classification d'une classe normale comme une classe d'attaque. Le vrai positif (VP) est la classification correcte de la classe d'attaque.

La modification de l'ensemble de données d'apprentissage ou l'utilisation d'un autre type de classificateur peut causer des modifications dans les résultats de la classification. Si nous utilisons deux classificateurs de type différent ou un classificateur formé avec deux ensembles de données d'apprentissage différents, l'intersection des résultats de classification peut nous permettre de diminuer le FP et FN. Ce qui nous permet de construire un modèle de détection d'intrusion plus efficace. La figure suivante montre le résultat de l'intersection de deux classificateurs différents.

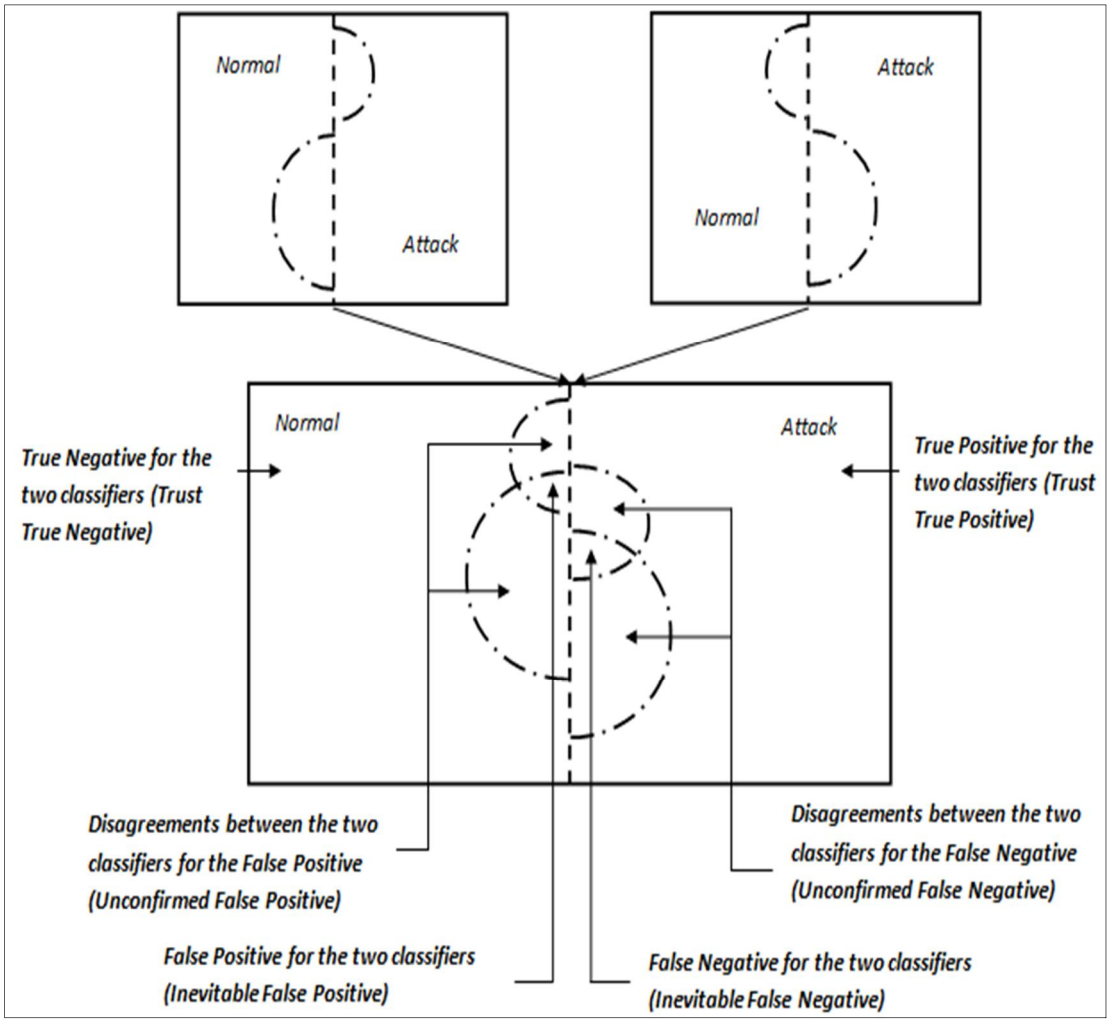

Figure 40 L'intersection des résultats de classification de deux différentes techniques de data mining

Comme le montre la figure 40, nous pouvons classer l'intersection entre les résultats de classification de deux différents classificateurs en six catégories:

- Le véritable vrai négatif (TTN): TTN représente toutes les connexions normales classées par les deux classificateurs comme normales.
- Le véritable vrai positif (TTP): TTP représente toutes les attaques classées par les deux classificateurs comme attaque.
- L'inévitable faux positif (IFP): IFP représente toutes les connexions normales mal classées par les deux classificateurs comme attaque. Il est égal dans le pire des cas au minimum de FP généré par l'un des deux classificateurs.
- L'inévitable faux négatif (IFN): IFN représente toutes les attaques mal classées par les deux classificateurs comme normales. Il est égal dans le pire des cas au minimum de FN généré par l'un des deux classificateurs.

- Le faux positif non confirmé (UFP): UFP représente toutes les connexions normales classifiées comme des attaques par l'un des deux classificateurs et classifiés comme normales par l'autre classificateur.
- Le faux négatif non confirmé (UFN): UFN représente toutes les attaques classifiées comme normales par l'un des deux classificateurs et classifiées comme des attaques par l'autre classificateur.

7.1.1.1. La structure générale de notre modèle adaptatif

Dans cette sous-section, nous présentons les différentes étapes nécessaires pour construire notre modèle adaptatif. Notre modèle est basé sur la minimisation du FP et FN par l'intersection de deux différents classificateurs. Comme le montre la figure ci-dessous, notre modèle adaptatif est composé de N niveaux, chaque niveau contient deux classificateurs. Si le nombre de connexions de l'ensemble de données d'apprentissage (UFP union UFN) est plus grand que ou égal au seuil, nous construisons un nouveau niveau. Sinon, nous regroupons les nouvelles attaques avec le reste de l'ensemble de données d'apprentissage du niveau précédent. Quand la somme devient supérieure ou égale au seuil nous créons un nouveau niveau. Nous pouvons résumer ce processus dans les trois fonctions suivantes: la première fonction consiste à construire le modèle initial, la seconde montre comment tester les connexions, et la troisième assure l'adaptation de notre modèle.

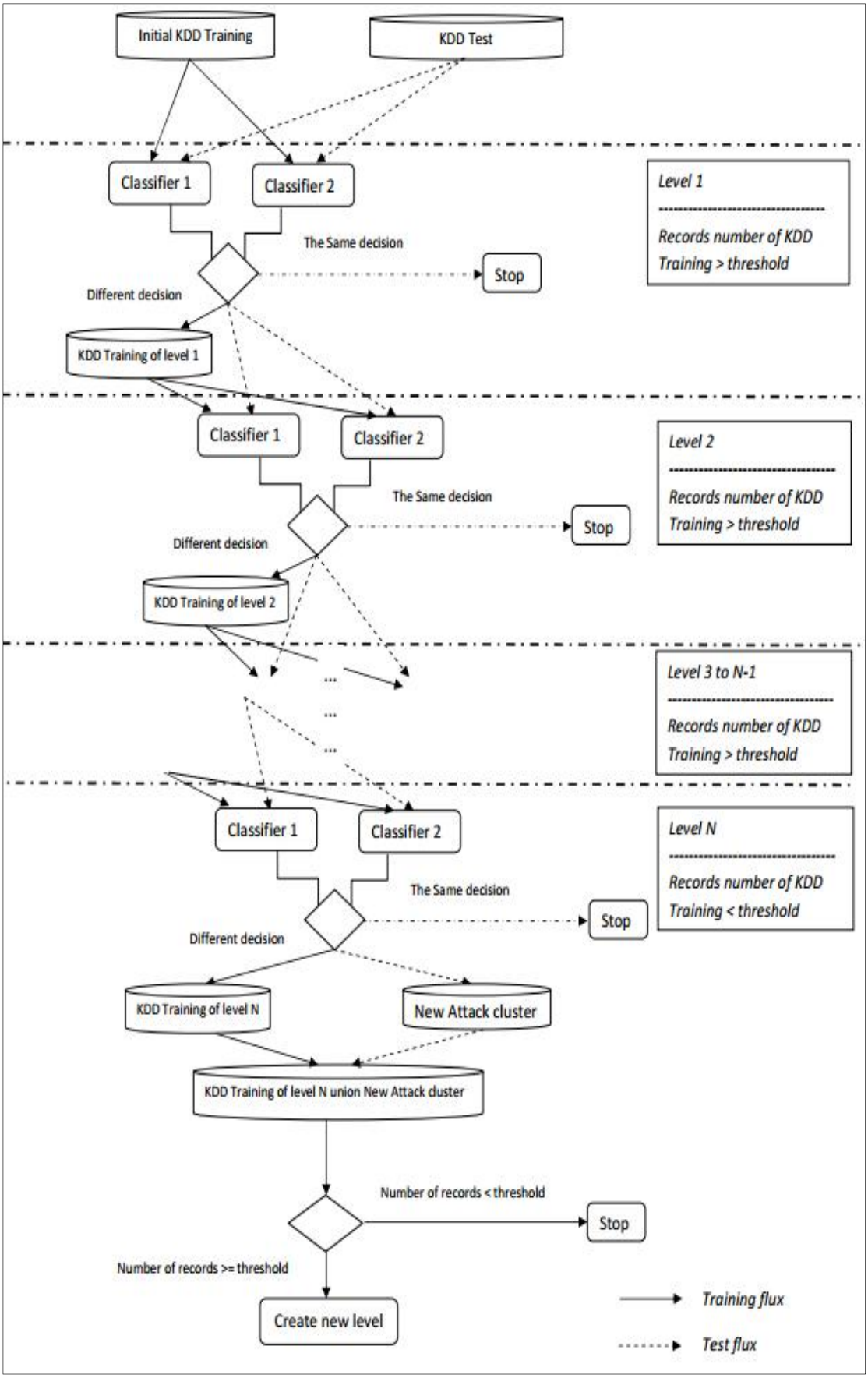

Figure 41 La structure générale de notre modèle adaptatif

7.1.1.1.1. Construire le modèle initial

L'UFN et UFP représentent la clé de notre approche, où nous les utilisons comme nouvelles données d'apprentissage pour l'apprentissage d'un nouveau niveau. Ensuite, les nouveaux UFN et UFP (générer à partir de l'intersection des résultats de la classification des classificateurs du nouveau niveau) sont utilisés comme nouvelles données d'apprentissage pour l'apprentissage d'un autre nouveau niveau. Nous répétons ce processus jusqu'à ce que le nombre des enregistrements de UFN union UFP deviennent inférieur au seuil. Nous pouvons résumer le processus de construction du modèle initial par le pseudo code suivant:

Algorithm *Build_Model: IT*

```
1:   T = IT; // IT is the initial training data set;
2:   I =1;
3:   built_model = null;
     /* Repeat these instructions until the number of records becomes less than the
     threshold */
4:   While (count_records (T) >= threshold) do
5:   Begin
6:     A = Proportion of T;
7:     B = Proportion of T;
8:     Training C1 with A; // Training classifier C1 with A data set
9:     Training C2 with B; // Training classifier C2 with B data set
       //set the two trained classifiers as classifiers of level I
10:    built_model→add(the trained C1 and C2 as classifier of level I);
11:    I++;
12:    D1 = Test T with C1; // Test all T records with the classifier C1
13:    D2 = Test T with C2; // Test all T records with the classifier C2
14:    {UFN} = all UFN generated from the intersection of D1 with D2;
15:    {UFP} = all UFP generated from the intersection of D1 with D2;
16:    T = {UFN} union {UFP};
17:  End_While
     /*Model= built model with the different levels*/
18:  Result→ Model= built_model;
     /*Rest= {UFN} union {UFP} of the last level that is less than the threshold*/
19:  Result→ Rest=T;
20:  Return Result;
```

7.1.1.1.2. Le processus du test

Pour tester les différents enregistrements, chacun d'entre eux doit se propage du haut vers le bas. Pour chaque niveau, nous testons si la décision des deux différents classificateurs est la même décision nous nous arrêtons le processus et nous classons l'enregistrement. Sinon, nous testons l'enregistrement par le niveau suivant. Si tous les classificateurs de tous les niveaux du modèle donnent deux décisions différentes, cette connexion est considérée comme une nouvelle attaque. Le processus de test est résumé par le pseudo code suivant:

Algorithm *TestProcess: CT*

```
1:   SC = False; // SC represents the stop condition
2:   I=1;
3:   L= the number of Levels of the built model;
4:   TS= CT // TS=the record to be tested;
5:   While ((I <= L) and Not (SC)) do
6:     Begin
7:       D1 = the Decision of Classifier (C1) of Level (I)
                                    for the connection (TS);

8:       D2 = the Decision of Classifier (C2) of Level (I)
                                    for the connection (TS);

9:       SC = (D1==D2);
10:      I++;
11:   End_While
12:   If (SC)  then return D1; // the same decision
13:  Else return "New_Attack";
```

7.1.1.1.3. L'adaptation du modèle

Pour assurer l'adaptation de notre modèle, nous regroupons les nouvelles attaques et le reste de l'ensemble des données d'apprentissage du dernier niveau pour former un nouveau niveau. Nous pouvons résumer le processus d'adaptation qui représente le programme principal de notre modèle par le pseudo code suivant:

Algorithm *AdaptProcess*

```
1:    IT= Training Data Set; // initial data set
2:    Result= BuildModel(IT); // build the initial model
3:    Imodel = Result→Model; // Imodel is the initial Model
      // NA is the training data set of the last level of built model
4:    NA= Result→Rest;
5:    Adaptive_Model=Imodel;
6:    AL = connection_list; // AL is the list of records
7:    L= length (AL);
8:    I=0;
      // while the list of the records is not empty
9:    While ( I < L) do
10:    Begin
        /*while the list of the records is not empty and the sum of the new attacks and the rest of
        training data is less than the threshold*/
11:     While ( I< L) and (count(NA)< threshold) do
12:       Begin
13:         if (TestProcess(AL[i])== "New_Attack")
            // add the new attack to the list of new attack
14:          NA→add(AL[i]);
15:         i++;
16:       End_while
        /* if the sum of the new attacks and the rest of training data is equal to the threshold*/
17:     If (count(NA)== threshold) then
18:     Begin

          // build a new model with only one level (new level)

19:       Result= BuildModel(NA);
20:       AModel= Result→model;
21:       NA=Result→Rest;

           /* add the level of AModel (new level ) to our Adaptive Model as new level*/

22:       Adaptive_Model→Add_level_of(AModel);
23:    End_if
24:  End_while
```

7.2. Expérimentation

Cette sous-section est divisée en trois parties. Dans la première de cette sous-section, nous détaillons l'ensemble de données d'apprentissage et de test. La deuxième partie de cette sous-section décrit les classificateurs et les paramètres utilisés pour construire et former notre modèle. La troisième partie de cette sous-section représente une étude comparative entre notre modèle et d'autres modèles récents de détection d'intrusion.

Nous avons effectué une série d'expérimentation avec le KDD'99 qui représente l'ensemble de données de détection d'intrusion le plus utilisées (Xiaonan et Banzhaf, 2010), (Tsaia et al, 2009). Weka Data Mining Tools (Witten et al, 2011) est utilisé pour la mise en œuvre de notre modèle. Les résultats sont obtenus sur un PC Windows avec Core 2 Duo 2,0 GHz et 2 Go de RAM. La performance d'un IDS est mesurée par sa capacité de classer chaque connexion dans la bonne catégorie. Les indicateurs de performance les plus utilisés pour évaluer les systèmes de détection d'intrusion sont:

$$Exactitude\ (Accuracy) = \frac{\text{VP+VN}}{VP+VN+FP+FN} \qquad (7.1)$$

$$Taux\ de\ Detection\ (TD\ ) = \frac{\text{VP}}{VP+FN} \qquad (7.2)$$

$$Taux\ de\ Faux\ Alarm\ (TFA\ ) = \frac{\text{FP}}{VN+FP} \qquad (7.3)$$

7.2.1. Les ensembles de données d'apprentissage et de test

En raison de la taille très grande de l'ensemble de données d'apprentissage du KDD'99_ten_porcent (KDD'99 10%), nous avons créé notre ensemble de données d'apprentissage qui contient 40.000 enregistrements. Pour réduire la taille du KDD'99 10% tous les enregistrements redondants ont été supprimés puis la sélection aléatoire est utilisée pour sélectionner les enregistrements Normal et DOS (Neptune). Le tableau suivant résume la répartition des attaques et du comportement normal de notre ensemble de données d'apprentissage. L'ensemble de données KDD'99 Test (KDD'99, 1999) est utilisé pour évaluer la performance de notre modèle. Les deux caractéristiques num_outbound_cmds et is_host_login sont supprimées en raison de leurs valeurs identiques dans l'ensemble de données d'apprentissage. Pour normaliser les ensembles de données, le codage ASCII est utilisé pour convertir les données symboliques en valeurs numériques. Ensuite, chaque donnée x_i de la caractéristique J est normalisée en utilisant l'équation suivante:

$$\overline{x_i(j)} = \frac{x_i(j) - min(x(j))}{max(x(j)) - min(x(j))} \quad (7.4)$$

Type de Connexion	Nombre d'enregistrements	Description	%
Normal	10.000	10.000 enregistrements distincts sélectionnés aléatoirement de la catégorie Normal, tous les enregistrements sont extraits du KDD'99 10%	25%
DOS	2.6819	Tous les enregistrements distincts des types d'attaque Pod, Land, Back, Teardrop, Smurf plus 24067 enregistrements distincts et aléatoirement sélectionnés du type Neptune. Tous les enregistrements sont extraits du KDD'99 10%.	67,05%
Probe	2.130	Tous les enregistrements distincts de la catégorie Probe. Tous les enregistrements sont extraits du KDD'99 10%	5.33%
R2L	999	Tous les enregistrements distincts de la catégorie R2L. Tous les enregistrements sont extraits du KDD'99 10%	2.5%
U2R	52	Tous les enregistrements distincts de la catégorie U2R. Tous les enregistrements sont extraits du KDD'99 10%	0.13%

Tableau 24 La répartition des attaques et du comportement normal de notre ensemble de données d'apprentissage

7.2.2. Les classificateurs et les paramètres utilisés pour construire et entraîner notre modèle

Pour construire notre modèle, nous avons sélectionné deux techniques de data mining très différentes dans leur mode d'opération. Ces deux techniques nous permettent d'avoir un bon taux de classification correcte et une bonne hétérogénéité dans la classification des différentes connexions. Ces deux classificateurs sont: "Fuzzy Unordered Rule Induction Algorithm (FURIA)" (Hühn and Hüllermeier, 2009) et "Random Forests (RF)" (Breiman, 2001).

7.2.2.1. Fuzzy Unordered Rule Induction Algorithm (FURIA)

FURIA est l'abréviation de "Fuzzy Unordered Rule Induction Algorithm" qui représente une méthode de classification fondée sur des règles floues. FURIA est basée sur l'algorithme RIPPER (Cohen, 1995). FURIA a conservé les avantages de RIPPER avec quelques

modifications, où il utilise des règles floues plutôt que des règles conventionnelles et un ensemble de règles non ordonnées au lieu des listes de règles (Hühn et Hüllermeier, 2009).

7.2.2.2. Random Forests (RF)

RF combine les facteurs prédictifs des arbres, où chaque arbre dépend des valeurs d'un vecteur aléatoire échantillonnées d'une manière indépendante et avec la même distribution pour tous les arbres de la forêt. L'erreur de classification de la forêt des arbres dépend de la robustesse des différents arbres de la forêt et de la corrélation entre eux (Breiman, 2001).

7.2.2.3. Les paramètres utilisés pour former notre modèle

L'ensemble de données d'apprentissage initial (IT) représente notre ensemble de données d'apprentissage précédemment présent dans le tableau 24 ci-dessus. Les parties (A) et (B) utilisés pour former respectivement C1 (FURIA) et C2 (RF) sont définis dans le tableau ci-dessous.

	T	
	Normal	Attaque
A	30%	10%
B	10%	30%

Tableau 25 Les valeurs des parties A et B

Nous avons utilisé deux parties différentes A et B pour former respectivement FURIA et RF afin d'augmenter le UFP et le UFN. Cette hétérogénéité des classificateurs et des données d'apprentissage nous permet de construire un modèle multi-niveau. Dans notre implémentation nous avons fixé le seuil à 220 enregistrements.

7.2.3. Étude comparative

Pour évaluer les performances de notre modèle, nous avons comparé ses performances avec des travaux connexes et certains modèles récents de détection d'intrusion qui utilisaient tout l'ensemble de données KDD'99 Test (KDD'99, 1999) pour tester leurs modèles. Nous avons utilisé les paramètres détaillés dans la sous-section précédente pour créer et entraîner notre modèle. Tous les enregistrements du KDD'99 Test sont utilisés comme ensemble de données de test. Le résultat de cette étude comparative est résumé dans le tableau suivant.

	Normal	DOS	Probe	R2L	U2R	TD	TFA	Exactitude
Toosi and Kahani(2007)	98,20%	99,50%	84,10%	31,50%	14,10%	94,77%	1,90%	95,30%
Xiang et al.(2008)	96,80%	98,66%	93,40%	46,97%	71,43%	93,93%	3,20%	94,49%
Wanga et al.(2010)	99,08%	96,70%	80,00%	58,57%	76,92%	93,94%	0,92%	94,94%
Khor and Ting(2012)	97,40%	97,80%	73,30%	48,20%	87,30%	94,18%	2,60%	94,80%
Badran and Rockett(2012)	99,50%	96,99%	78,01%	5,59%	11,40%	90,69%	0,50%	92,41%
Koc et al.(2012)	-	99,60%	-	-	-	-	-	93,72%
Notre modèle	97,77%	98,32%	95,82%	45,48%	73,68%	94,84%	2,23%	95,41%

Tableau 26 La comparaison entre notre approche et les travaux connexes ainsi que certains modèles de détection d'intrusion récents

Comme le montre la figure suivante, les meilleurs classificateurs pour la classification des comportements normaux, des attaques DoS, des attaques Probe, des attaques R2L, des attaques U2R sont respectivement le modèle de Badran et Rockett (2012), le modèle de Koc et al. (2012), notre modèle, le modèle de Wanga et al. (2010), le modèle de Khor et Ting (2012). Notre modèle a montré sa haute performance pour la classification de DOS, PROB, R2L, U2R et le comportement normal par rapport aux travaux connexes et certains modèles récents de détection d'intrusion. Globalement, notre modèle donne le taux de détection le plus élevé et un faible taux de fausses alarmes. En outre, notre modèle est plus précis que le meilleur des modèles utilisés dans cette étude comparative avec un taux d'exactitude égal à 95,41%.

Le temps d'apprentissage de notre modèle initial est de 19 secondes, ce qui prouve la rapidité de déploiement de notre modèle.

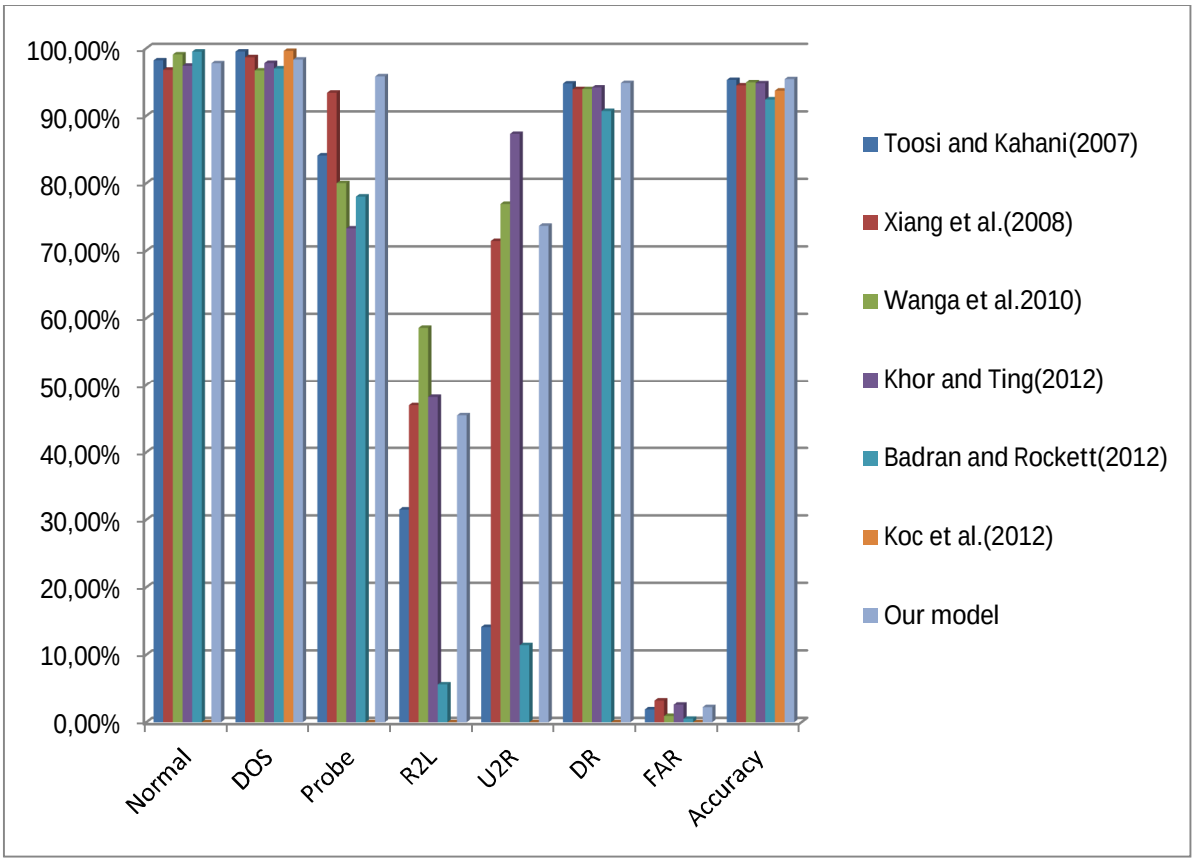

Figure 42 La comparaison entre notre approche et les travaux connexes ainsi que certains modèles de détection d'intrusion récents

7.3. Conclusion

Dans ce chapitre, nous avons proposé une nouvelle approche pour la création d'un système détection d'intrusion adaptatif basée sur l'intersection de deux différents classificateurs. Pour créer un modèle de détection d'intrusion rapide et performant, nous avons utilisé "Fuzzy Unordered Rule Induction Algorithm" et "Random Forests" comme classificateurs. Les expérimentations en KDD'99 ont montré la haute performance de notre modèle par rapport aux travaux connexes et certains modèles récents de détection d'intrusion. Notre modèle donne le taux de détection le plus élevé et le taux le plus élevé d'exactitude par rapport aux travaux connexes ainsi que certains modèles de détection d'intrusion récents. En outre, le temps d'apprentissage de notre modèle initial est de 19 secondes, ce qui prouve la rapidité de déploiement de notre modèle.

Chapitre 8

Proposition 4 un système de détection d'intrusion avec une grande capacité de généralisation

Le but de cette proposition est de créer un système de détection d'intrusion qui possède une grande capacité de généralisation. L'IDS proposé est un modèle hybride, hiérarchique qui comprend deux niveaux. Le premier niveau contient les cinq classificateurs suivants: "Repeated Incremental Pruning to Produce Error Reduction", "Multilayer Perceptrons", "Self-Organizing Feature Map Network", Arbre de décision C4.5 et Naïve Bayes. Ces classificateurs sont utilisés pour leurs taux élevés de correcte classification pour respectivement DoS, comportement normal, Probe, R2L et U2R. Seules cinq prédictions du premier niveau sont sélectionnées et utilisées comme des entrées du second niveau qui contient le réseau de neurones RBF comme classificateur final. L'expérimentation avec le KDD'99 a montré que notre modèle donne le taux le plus élevé d'exactitude et le taux de détection le plus élevé par rapport aux résultats obtenus par certains modèles de détection d'intrusion bien connus.

8.1. La description du modèle

8.1.1. La structure de notre modèle

Dans cette sous-section, nous présentons les différents composants de notre modèle et ses utilités. Comme le montre la figure 43, notre modèle contient deux niveaux:

- Le premier niveau: ce niveau contient différents types de classificateurs. Ces classificateurs sont sélectionnés pour leurs hautes performances dans la classification d'une ou de plusieurs classes de connexion. Comme le montre la figure 43, chaque classificateur donne cinq prédictions relatives aux quatre catégories d'attaques et du comportement normal. Nous ne maintenons que les prédictions des classes pour lesquelles les classificateurs sont sélectionnés. Ces cinq prédictions sont utilisées comme des entrées pour le second niveau.
- Le deuxième niveau: ce niveau contient un seul classificateur utilisé pour sa haute performance en tant que classificateur final. Il analyse les prédictions sélectionnées des différents classificateurs du premier niveau et prend la décision finale. Cette décision peut être soit une attaque soit un comportement normal.

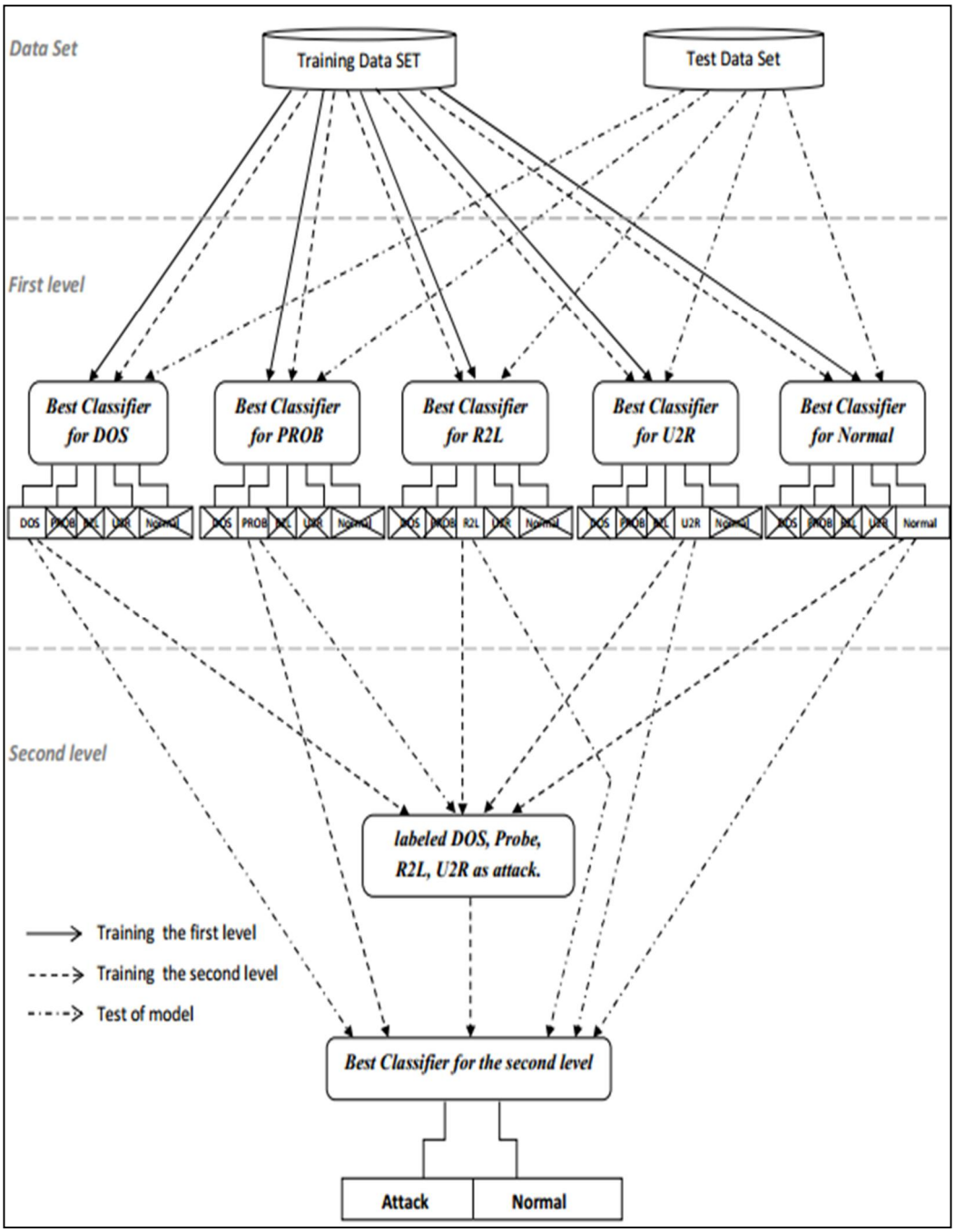

Figure 43 La structure générale de notre modèle hiérarchique

8.1.2. Le mode de fonctionnement de notre modèle

Le mode de fonctionnement de notre modèle hiérarchique se compose de trois phases: la phase de sélection des classificateurs du premier et du second niveau, la phase d'apprentissage et la phase de test.

8.1.2.1. Sélectionner les différents classificateurs du premier et du second niveau

Dans le but de sélectionner les meilleurs classificateurs pour notre modèle hiérarchique, nous effectuons deux études comparatives entre les différents types de classificateurs. Dans la première partie, nous comparons les différents classificateurs par rapport à leurs performances dans la classification des connexions dans l'une des cinq classes de connexions (DOS, Probe, R2L, U2R et le comportement normal). Nous ne sélectionnons que les cinq classificateurs qui donnent un bon taux de détection pour les cinq types de connexions. De plus, chaque classificateur sélectionné doit donner le taux de correcte classification le plus élevé pour au moins l'une des cinq classes. Pour effectuer la deuxième étude comparative, nous générons un nouvel ensemble de données à partir des cinq prédictions sélectionnées du premier niveau. Ensuite, nous utilisons ces données d'apprentissage pour comparer les différents classificateurs pour la classification des connexions en deux classes: attaque ou comportement normal. Nous sélectionnons le classificateur qui donne le taux de vrai positif le plus élevé et le taux de fausse alarme le plus bas.

8.1.2.2. La phase d'apprentissage

Dans cette phase, nous effectuons l'apprentissage de notre modèle dans le but de le préparer pour la phase de test. Cette étape est composée de deux étapes:

- Former le premier niveau: nous formons les différents classificateurs du premier niveau avec l'ensemble de données d'apprentissage, où chaque élément de l'ensemble de données d'apprentissage représente une entrée pour le classificateur.
- Former le second niveau: un nouvel ensemble de données est créé à partir des prédictions des classificateurs du premier niveau. Pour générer ce nouvel ensemble de données d'apprentissage, nous associons les prédictions sélectionnées avec l'étiquette correcte comme le montre le tableau 27. Le nouvel ensemble de données d'apprentissage est utilisé pour former le classificateur sélectionné pour le deuxième niveau.

Prédiction du DOS	Prédiction du Probe	Prédiction du U2R	Prédiction du R2L	Prédiction du Normal	étiquette
0.94	0.25	0.17	0.38	0.18	Attaque
0.15	0.34	0.18	0.36	0.94	Normal
0.28	0.28	0.89	0.22	0.15	Attaque
0.35	0.99	0.38	0.14	0.36	Attaque
0.16	0.13	0.25	0.89	0.32	Attaque

Tableau 27 Les nouvelles données d'apprentissage pour le deuxième niveau

8.1.2.1. La phase de test

Dans cette phase, nous testons la performance de notre modèle après l'achèvement de la phase d'apprentissage, où nous utilisons l'ensemble de données de test. Nous traitons chaque enregistrement de l'ensemble de données de test par les différents classificateurs de premier niveau. Ensuite, nous utilisons les sorties des prédictions sélectionnées des différents classificateurs du premier niveau comme des entrées pour le classificateur du deuxième niveau.

8.2. Expérimentation

Cette section est divisée en trois parties. Dans la première partie, nous détaillons l'ensemble de données d'apprentissage et de test. La deuxième partie présente une étude comparative entre huit classificateurs, dans le but de sélectionner les classificateurs les plus performants pour le premier et le deuxième niveau. La troisième partie représente une étude comparative entre notre nouveau modèle hiérarchique et d'autres classificateurs bien connus.

Nous avons effectué une série d'expérimentation avec le KDD'99 Cup (KDD'99, 1999), qui représente l'ensemble de données de détection d'intrusion le plus utilisé dans la dernière décennie. Weka Data Mining (Witten et al, 2011) et NeuroSolutions Software (Curt et al, 2012) sont utilisés pour la mise en œuvre des différents classificateurs. Les résultats sont obtenus sur un PC Windows avec Core 2 Duo 2,0 GHz et 2 Go de RAM.

8.2.1. Les données d'apprentissage et de test

En raison de la grande taille du KDD'99_10%, nous avons créé notre ensemble de données d'apprentissage qui contient 20.000 enregistrements. Pour réduire la taille du KDD'99_10% tous les enregistrements redondants sont supprimés. Ensuite, la sélection aléatoire est utilisée pour sélectionner les enregistrements Normal et DOS (Neptune). Le tableau 28

représente la répartition des attaques et du comportement normal de notre ensemble de données d'apprentissage. L'ensemble de données KDD'99 Test est utilisé pour évaluer la performance de notre modèle. Les deux caractéristiques num_outbound_cmds et is_host_login sont supprimées en raison de leurs valeurs identiques dans l'ensemble de données d'apprentissage. Pour normaliser les ensembles de données, le codage ASCII est utilisé pour convertir les données symboliques à des valeurs numériques. Ensuite, chacune des données xi de la caractéristique j est normalisée en utilisant l'équation suivante:

$$\overline{x_i(j)} = \frac{x_i(j) - min(x(j))}{max(x(j)) - min(x(j))} \qquad (8.1)$$

Type de Connexion	**Nombre d'enregistrement**	**Description**	**Pourcentage**
Normal	4.500	4.500 enregistrements normaux distincts sélectionnés aléatoirement du KDD'99 10%	22,50%
DOS	12.319	Tous les enregistrements distincts de Pod, Land, Back, Teardrop, Smurf plus 9.567 enregistrements distincts et aléatoirement sélectionnés de Neptune. Tous les enregistrements sont extraits du KDD'99 10%.	61,60%
Probe	2.130	Tous les enregistrements distincts de Probe extraits du KDD'99 10%	10,65%
R2L	999	Tous les enregistrements distincts de R2L extraits du KDD'99 10%	5,00%
U2R	52	Tous les enregistrements distincts de U2R extraits du KDD'99 10%	0,26%

Tableau 28 La répartition des attaques et du comportement normal de notre ensemble de données d'apprentissage

8.2.2. Étude comparative entre les différents classificateurs

Dans le but de sélectionner les meilleurs classificateurs pour les deux niveaux de notre nouveau IDS hiérarchique, nous avons effectué deux études comparatives. La première consiste à sélectionner les différents classificateurs du premier niveau qui donnent le meilleur taux de correcte classification pour DOS, Probe, U2R, R2L et le comportement normal. La seconde consiste à sélectionner le classificateur de deuxième niveau qui donne le meilleur taux de correcte classification des connexions en attaque et comportement normal. Les huit classificateurs comparés sont les suivants: "Multilayer Perceptrons (MLP)" (Bishop, 1996), "Naïve Bayes (NB)" (John and Langley, 1995), "C4.5 Decision Tree (DT)" (Quinlan, 1993), "Support Vector Machine (SVM)" (Chang and lin, 2001), "RBF Neural

Network (RBFN)" (Bugmann, 1998), "Recurrent neural network (RNN)" (Tutschku, 1995), "Sequential Minimal Optimization (SMO)" (Platt,1999), "Repeated Incremental Pruning to Produce Error Reduction (RIPPER)" (Cohen, 1995). Dans cette étude comparative, nous avons utilisé les données d'apprentissage détaillé dans le tableau 28.

8.2.2.1. Étude comparative entre les huit classificateurs pour le premier et le second niveau

Pour comparer les différents classificateurs par rapport au premier niveau, nous avons effectué une série d'expérimentation, où chaque classificateur dispose de 39 entrées qui représentent les 41 caractéristiques de KDD9 sans num_outbound_cmds et is_host_login. Chaque classificateur donne sa prédiction pour les quatre catégories d'attaques (DOS, Probe, U2L, R2L) et le comportement normal. Le tableau suivant résume le taux de correcte classification des huit classificateurs pour chaque catégorie de connexion.

	DOS	**Normal**	**Probe**	**R2L**	**U2R**
DT	97.05%	86.59%	92.05%	**28.23%**	8.77%
SVM	97.04%	96.53%	84.47%	9.81%	8.77%
NB	96.33%	94.17%	89.53%	0.78%	**22.81%**
RIPPER	**97.92%**	97.34%	84.21%	13.95%	10.53%
MLP	97.49%	**98.99%**	83.20%	7.80%	0.00%
RBF	96.31%	95.14%	77.96%	3.45%	0.00%
SOFM	96.65%	95.17%	**97.10%**	1.93%	0.00%
RNN	97.59%	97.22%	84.61%	8.21%	0.00%

Tableau 29 Etude comparative entre les huit classificateurs pour le premier niveau

Comme le montre la figure 44, les meilleurs classificateurs dans la classification de comportement normal, DOS, R2L, Probe, U2R sont respectivement MLP avec TVN égal à 98,99%, RIPPER avec DR égal à 97,49%, DT avec DR égal à 28,23%, SOFM avec DR égal à 97,10%, NB avec DR égal à 22,81%.

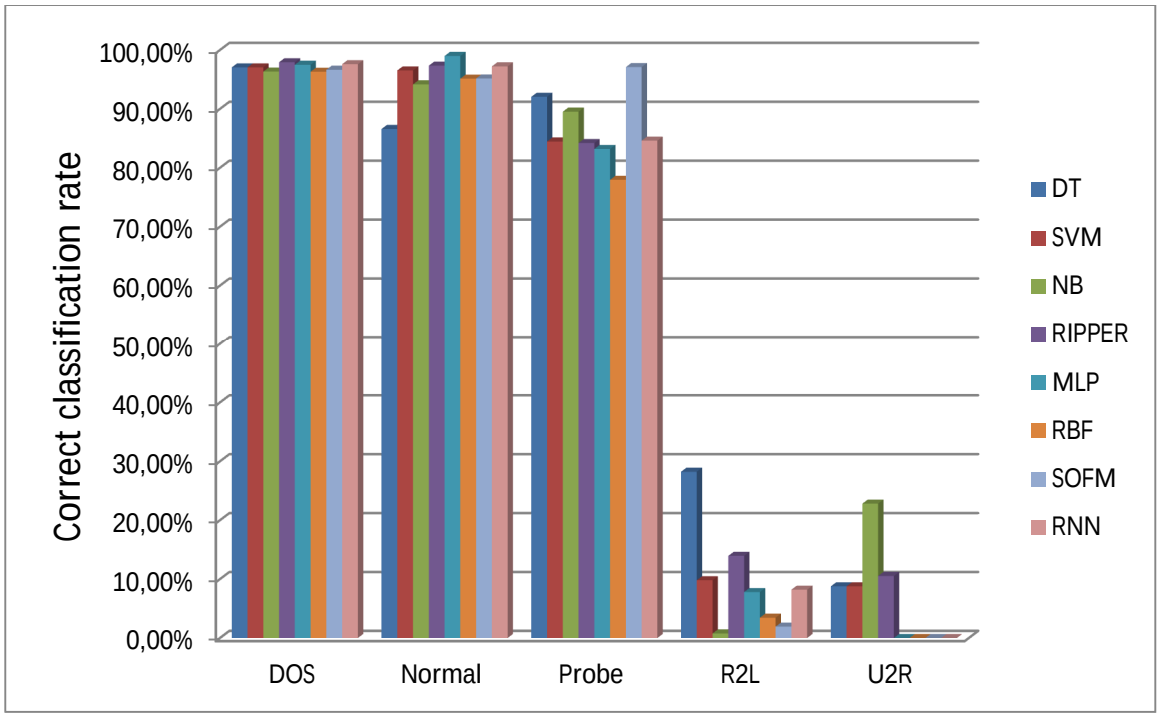

Figure 44 Etude comparative entre les huit classificateurs pour le premier niveau

Pour le deuxième niveau, nous sélectionnons le réseau de neurones RBF qui donne le plus grand taux d'exactitude par rapport aux autres classificateurs.

8.2.3. L'évaluation de notre nouveau IDS hiérarchique

Après avoir analysé la performance des différents types de classificateurs, nous avons exploité leurs points forts afin d'atteindre notre objectif. Comme le montre la figure 45, notre nouveau modèle contient dans le premier niveau les classificateurs suivants RIPPER, SOFM, DT et NB qui ont été sélectionné pour leurs taux de vrai positif le plus élevés dans la détection de respectivement DOS, Probe, R2L et U2R. En outre, MLP est sélectionné pour son taux de vrai négatif le plus élevé dans la détection du comportement normal. Dans le deuxième niveau, nous utilisons le réseau de neurones RBF comme un classificateur final afin de combiner les prédictions des différents classificateurs et donner la décision finale.

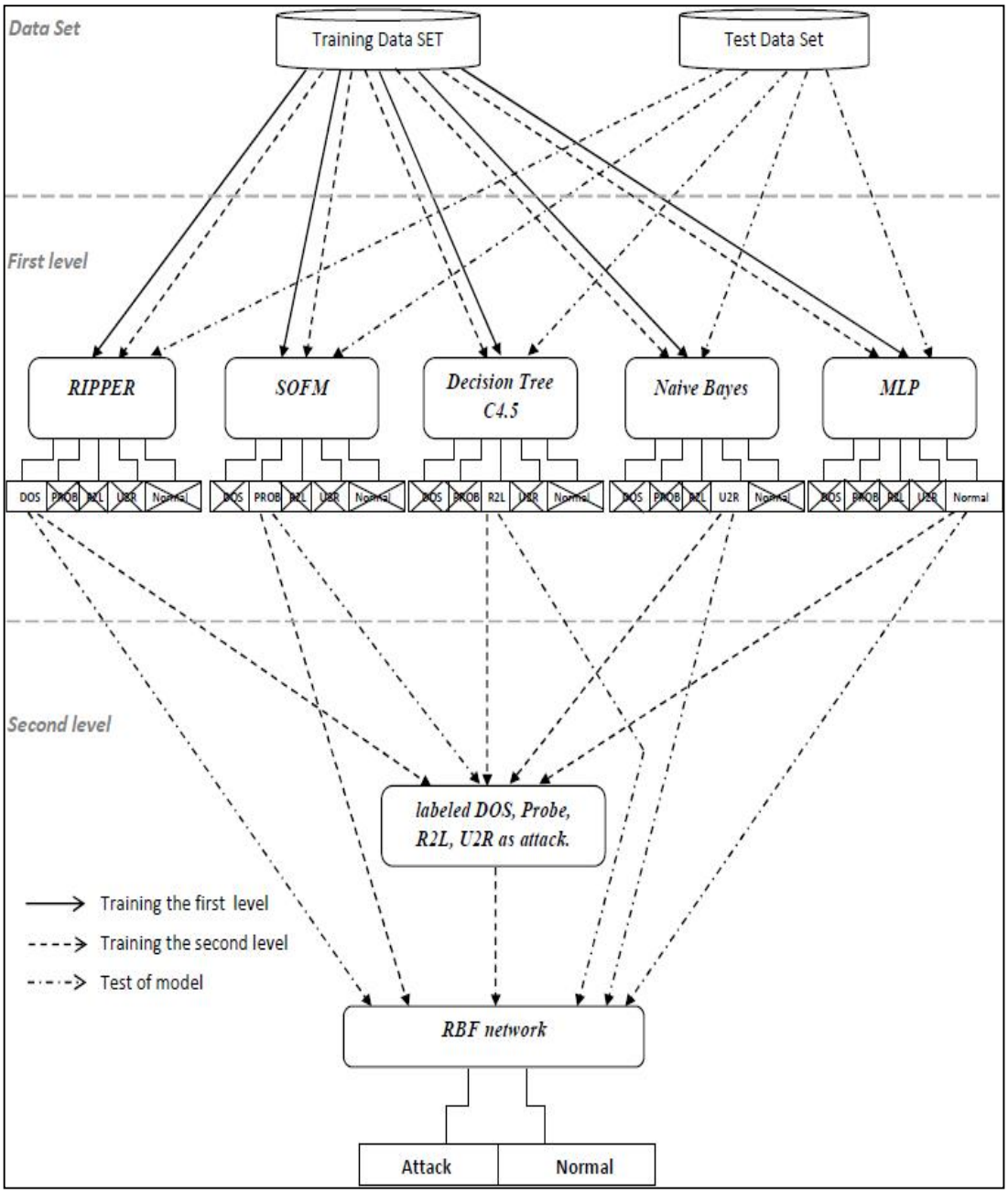

Figure 45 La structure pratique de notre modèle hiérarchique

Pour implémenter notre modèle, nous avons utilisé l'outil Weka (Witten et al, 2011) pour l'apprentissage et le test des différents classificateurs, et le logiciel NeuroSolutions (Lynn et al, 2012) pour l'apprentissage et le test des différents réseaux de neurones. Java SE Development Kit 7 et la base de données MySQL sont utilisés pour traiter l'ensemble de données et combiner les différentes sorties du premier niveau. Le tableau 30 résume les principaux paramètres de l'arbre de décision C4.5 et RIPPER dans l'outil Weka. Les tableaux 31 et 32 résument les principaux paramètres des réseaux de neurones du niveau 1 (MLP, SOFM) et du niveau 2 (RBF) dans NeuroSolutions.

RIPPER		C4.5 DT	
paramètre	Valeur	paramètre	Valeur
Folds	3	BinarySplits	false
minNo	2	CollapseTree	true
optimizations	2	confidenceFactor	0.25
seed	1	minNumObj	2
usePruning	true	numFolds	3
-	-	seed	1

Tableau 30 Les différents paramètres de RIPPER et de l'arbre de décision C4.5

MLP		**SOFM**	
Paramètre	**valeur**	**Paramètre**	**valeur**
Hidden layers	3	Hidden layers	3
Transfer function	ThanAxon	Rows	5
Momentum	0,7	Columns	5
Hidden Layer 1 Elements	38	Neighborhood Shape	SquareKohonenFull
Hidden Layer 2 Elements	19	Hidden Layer Elements	38
Hidden Layer 3 Elements	12	Hidden Layer Elements	19
Threshold	0,001	Hidden Layer Elements	12
-	-	Transfer function	ThanAxon
-		Threshold	0,001

Tableau 31 Les différents paramètres de MLP et SOFM

RBF Neural Network	
paramètre	***Valeur***
Hidden layers	2
Transfer function	ThanAxon
learning Rule	LevenbergMarqua
Threshold	0,001
Cluster Centers	25
Competitive Rule	ConscienceFull
Metric	Euclidean
Hidden Layer 1 Processing Elements	15
Hidden Layer 3 Processing Elements	7

Tableau 32 Les paramètres du réseau de neurones RBF

Pour évaluer la performance de notre nouvelle approche, nous avons comparé ses résultats avec les résultats des travaux connexes et des autres travaux récents qui utilisent tout l'ensemble de données KDD'99 Test data set pour tester leurs modèles. Pour notre modèle, nous avons utilisé l'ensemble de données d'apprentissage détaillé dans le tableau 28 ci-dessus comme un ensemble de données de formation, et toutes les données de KDD'99 comme ensemble de données de test. Le résultat de la comparaison est résumé dans le tableau 33.

	Notre modèle	**Toosi (Tossi and Kahani, 2007)**	**FC-ANN (Wanga et al., 2010)**	**Xiang (Xiang et al, 2008)**	**Horng (Horng et al,2011)**
Normal	98,85%	98,20%	99,08%	96,80%	**99,30%**
DoS	98,68%	99,50%	96,70%	98,66%	**99,50%**
Probe	96,88%	84,10%	80,00%	93,40%	**97,50%**
R2L	42,77%	31,50%	**58,57%**	46,97%	28,80%
U2R	67,11%	14,10%	**76,92%**	71,43%	19,70%
DR	**95,01%**	94,77%	93,94%	93,93%	94,82%
FAR	1,15%	1,90%	0,92%	3,20%	**0,70%**
Accuracy	**95,76%**	95,30%	94,94%	94,49%	95,70%

Tableau 33 Comparaison entre notre modèle et certains travaux connexes et récents

Comme le montre la figure 46, le meilleur modèle pour la classification des comportements normaux est le modèle de Horng avec 99,30 %, les attaques DoS est le modèle de Horng avec 99,50 %, les attaques d'exploration est le modèle de Horng avec 97,50%, les attaques R2L est le modèle FC- ANN avec 58,57 % et les attaques U2R est le modèle FC- ANN avec 76,92%. Notre modèle ne donne pas le meilleur taux de correcte classification pour aucune catégorie, mais il est proche du plus haut taux de correcte classification pour la plupart d'entre eux, où il donne 98,85 % pour Normal, 98,68 % pour DOS, 96,88 % pour Probe, 42,77 % pour R2L et 67,11 pour U2R.

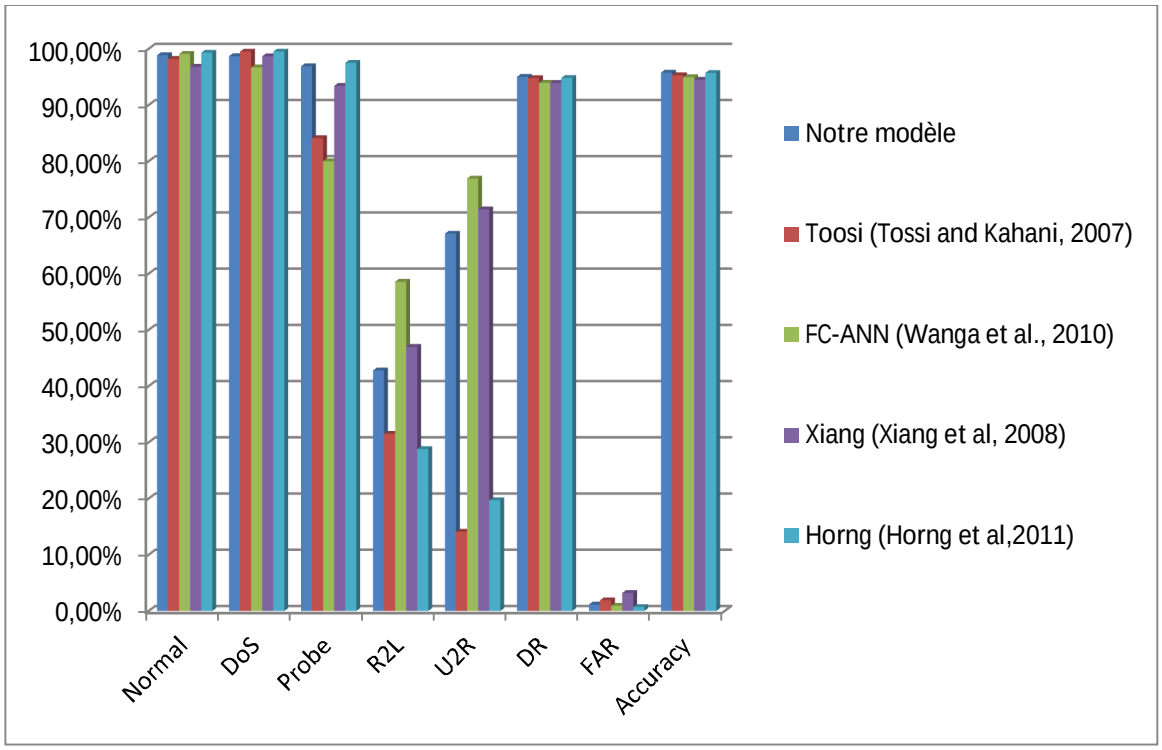

Figure 46 Comparaison entre notre modèle et les travaux connexes

Globalement, notre modèle donne le taux de détection le plus élevé, le taux d'exactitude le plus élevé et un taux de faux positif faible qui représente le troisième plus bas taux de fausse alarme. L'amélioration de la précision représente 186 enregistrements correctement classifiés plus que le meilleur modèle des travaux connexes et récents. De plus, notre modèle a montré sa capacité à mieux détecter les attaques non fréquentes et nouvelles de type U2R et R2L sans diminuer le taux de détection des attaques fréquentes ou d'augmenter le taux de fausses alarmes, ce qui représente un grand avantage.

Le temps nécessaire pour tester tout l'ensemble de données KDD'99 Test est 78,53 secondes, donc le temps moyen nécessaire pour traiter un enregistrement est 252,48 microsecondes. Ce qui prouve la rapidité de notre modèle dans le traitement du trafic réseau.

L'étude comparative entre notre modèle et les autres travaux connexes et récents montre que notre modèle a atteint les objectifs de notre travail, où il donne la plus grande précision avec 95,76 %, il combine un taux de détection élevé et un taux de fausse alarme faible, et il détecte mieux les attaques de faible fréquence sans diminuer le taux de détection des attaques fréquentes ou d'augmenter le taux de fausse alarme.

8.3. Conclusion

Dans ce chapitre, nous avons proposé un nouveau modèle de détection d'intrusion basé sur la combinaison de différents classificateurs qui possède une très grande capacité de généralisation. Notre modèle proposé répond aux exigences suivantes: mieux détecter les attaques non fréquentes et nouvelles, donne un taux de vrai positif élevé pour les attaques fréquentes et donne un taux de fausse alarme faible. Notre modèle comprend deux niveaux. Le premier contient cinq classificateurs RIPPER, MLP, SOFM, Arbre de décision C4.5 et Naïve Bayes utilisés pour leurs taux élevés de correcte classification de respectivement DOS, comportement normal, Probe, R2L et U2R. Seules cinq prédictions du premier niveau sont sélectionnées et utilisées comme entrées du second niveau qui contient le réseau de neurones RBF comme classificateur final. Les expérimentations en KDD'99 ont montré que notre modèle donne le taux de détection le plus élevé, le taux d'exactitude le plus élevé et un taux de fausse alarme faible par rapport à certains modèles de détection d'intrusion bien connus. En outre, notre modèle a montré sa capacité à mieux détecter les attaques de faible fréquence sans diminuer le taux de détection des attaques fréquentes ou d'augmenter le taux de fausse alarme.

Chapitre 9

Positionnement de nos travaux par rapport aux autres systèmes de détection d'intrusion adaptatifs

Dans ce chapitre nous présentons les apports de nos solutions pour remédier les limites des systèmes de détection d'intrusion de la deuxième génération. Pour chacun des quatre modèles proposés, nous présentons la nouveauté du modèle ainsi que ces avantages et ces inconvénients. La dernière partie de ce chapitre représente une comparaison entre la performance de nos modèles et les autres systèmes de détection d'intrusion adaptatifs existants ainsi que certains IDSs récents.

9.1. Les nouveautés des modèles proposés

Dans cette section, nous présentons les nouveautés de nos propositions par rapport aux modèles existants. Où nous présentons brièvement les idées qui nous ont poussés à proposer ces modèles, les clés de leurs implémentations ainsi que les apports de ces modèles.

9.1.1. Les nouveautés du premier modèle

Ce modèle a pour but de créer un système de détection d'intrusion très rapide en termes d'apprentissage afin de garantir un temps d'apprentissage et de réapprentissage très court sans négliger l'aspect de la capacité de généralisation. Ce modèle est composé de deux niveaux, le premier niveau contient cinq classificateurs sélectionnés pour leurs temps d'apprentissage très court et leurs grandes capacités de généralisation. Le second niveau ne contient qu'un seul classificateur très rapide qui combine les sorties des classificateurs du premier niveau et donne la décision finale. Pour réduire le temps d'apprentissage et de test, une version distribuée de ce modèle a été créée, où les classificateurs sont distribués sur différentes machines. Ce modèle nous a permis de créer un système de détection d'intrusion qui possède une bonne capacité de généralisation et qu'on peut le mettre à jour d'une manier très rapide. Ce qui facilite l'adaptation de notre modèle avec l'environnement cible et les nouvelles formes d'attaques.

9.1.2. Les nouveautés du deuxième modèle

Ce modèle vise à construire un système de détection d'intrusion avec une très grande capacité de généralisation afin de pouvoir détecté les nouvelles formes d'attaques et de s'adapter avec n'importe qu'elle environnement cible. Ce modèle a été créé sous forme d'un arbre binaire de classificateur. Pour chaque niveau de l'arbre, nous avons sélectionné le meilleur classificateur en termes de taux d'exactitude. Ce modèle possède une grande

capacité de généralisation ce qui lui permet de détecter les nouvelles formes d'attaques et d'avoir un taux de détection très élevé pour n'importe quel environnement cible.

9.1.3. Les nouveautés du troisième modèle

Ce modèle vise à construire un système de détection d'intrusion avec un mode d'apprentissage continu afin de s'adapter au fur et à mesure au système cible de plus ce modèle peut détecter d'une manière autonome les nouvelles formes d'attaques et d'intégrer ces nouvelles connaissances pour détecter leurs dérivées. La conception de ce modèle se base sur le croisement des décisions de classification de deux classificateurs différents, où les éléments qui représentèrent un désaccord entre les deux classificateurs sont utilisés pour la création d'une nouvelle couche dans notre modèle. Ce modèle nous a permis de construire un système de détection d'intrusion très rapide avec une grande capacité de généralisation qui s'adapte d'une manière autonome avec l'environnement cible.

9.1.4. Les nouveautés du quatrième modèle

Ce modèle a le même objectif du deuxième modèle, mais avec une structure similaire à la structure du premier modèle. Ce modèle possède une très grande capacité de généralisation ce qui lui a permis d'avoir un taux de détection très élevé et une capacité de détecter les nouvelles formes d'attaques.

9.2. Les avantages de nos modèles

Dans cette section nous présentons les avantages de nos différents modèles

9.2.1. Les avantages du premier modèle

On peut résumer les avantages de notre premier modèle par les points suivants :

- La structure hiérarchique du modèle où nous avons décomposé le processus de détection des attaques et du comportement normal en une hiérarchie de processus, ce qui rend la phase de détection moins complexe.
- L'utilisation de différents types de classificateurs d'après leurs performances dans la détection des différents types d'attaques et du comportement normal, nous a permis d'avoir une capacité de généralisation élevée, ce qui a été traduit par la capacité de détecter les nouvelles formes d'attaques.
- L'utilisation des classificateurs rapides ce qui nous a permis d'avoir un système de détection d'intrusion rapide en termes de temps d'apprentissage et de

réapprentissage. Donc, notre système peut s'adapter rapidement à l'environnement cible.

- La combinaison des décisions de différent classificateur pour les différents types de connexions ce qui augmente notre chance de prendre la bonne décision.

9.2.2. Les avantages du deuxième modèle

On peut résumer les avantages de notre deuxième modèle par les points suivants:

- Une structure sous forme d'un arbre binaire de classificateur qui offre beaucoup d'avantages comme la facilite de classification en deux classes pour chaque niveau, la rapidité d'apprentissage, la rapidité du test.
- La combinaison de plusieurs types de classificateurs performants pour la détection des différents types de connexions.
- Une classification détaillée en cinq classes de connexions (Probe, DoS, U2R, R2L, Normal).
- Une très grande capacité de généralisation.

9.2.3. Les avantages du troisième modèle

On peut résumer les avantages de notre troisième modèle par les points suivants :

- La structure hiérarchique du modèle qui décompose le processus de détection en une hiérarchie de processus, ce qui rend la phase de détection plus facile.
- La rapidité d'apprentissage ce qui facilite la phase de déploiement.
- Une autonomie pour la détection des nouvelles formes d'attaques.
- Une grande capacité de généralisation.
- La combinaison des décisions de deux différents types de classificateur ce qui minimise la possibilité de faire une mauvaise classification.

9.2.4. Les avantages du quatrième modèle

On peut résumer les avantages de notre quatrième modèle par les points suivants:

- La structure hiérarchique du modèle où nous avons décomposé le processus de détection des attaques et du comportement normal en une hiérarchie, ce qui rend la phase de détection moins complexe.
- Une très grande capacité de généralisation.
- L'utilisation de différent type de classificateur d'après leurs performances dans la détection des différents types d'attaques et du comportement normal, ce qui nous a

permis d'avoir une capacité de généralisation très élevée, qui a été traduit par la capacité de détecter les nouvelles formes d'attaques.

9.3. Les inconvénients de nos modèles

Dans cette section nous présentons les inconvénients de nos différents modèles

9.3.1. Les inconvénients du premier modèles

On peut résumer les inconvénients de notre premier modèle par les points suivants:

- La diminution de la performance de classification à cause du choix des classificateurs rapides plutôt que les classificateurs qui possèdent la plus grande capacité de généralisation.
- La complexité de l'étape de mise à jour.

9.3.2. Les inconvénients du deuxième modèle

On peut résumer les inconvénients de notre deuxième modèle par les points suivants:

- La complexité de l'étape de mise à jour.
- L'influence de la décision de la couche supérieure sur la performance de la couche inférieure.

9.3.3. Les inconvénients du troisième modèle

On peut résumer les inconvénients de notre troisième modèle par les points suivants:

- L'ajout d'une nouvelle couche qui assure l'adaptation de notre modèle dépend d'un seuil fixé à l'avance qui peut être long comme court.

9.3.4. Les inconvénients du quatrième modèle

On peut résumer les inconvénients de notre quatrième modèle par les points suivants:

- La complexité de l'étape de mise à jour.
- La lenteur de l'apprentissage et du réapprentissage à cause de l'utilisation de certains classificateurs qui demande un temps d'apprentissage long.

9.4. Étude comparative

Dans cette étude comparative, nous comparons les performances de nos modèles avec d'autres modèles de détection d'intrusion adaptatifs ainsi que d'autres modèles de détection d'intrusion récents qui utilisent tout la base de test KDD'99 Test pour l'évaluation de leurs modèles. Cette étude comparative est composée de deux parties. La première partie consiste à comparer les taux de correcte classification des cinq catégories de connexions. La

deuxième partie consiste à comparer le taux de détection globale, le taux de fausse alarme ainsi que le taux d'exactitude.

Le tableau suivant résume les résultats de notre modèle et les autres modèles de détection d'intrusion pour la classification des cinq catégories de connexions.

	Normal	DoS	Probe	R2L	U2R
Proposition 1	98,65%	97,81%	**98,13%**	43,15%	72,81%
Proposition 2	98,57%	99,45%	84,11%	36,17%	8,77%
Proposition 3	97,77%	98,32%	95,82%	45,48%	73,68%
Proposition 4	98,85%	98,68%	96,88%	42,77%	67,11%
Toosi and Kahani (2007)	98,20%	99,50%	84,10%	31,50%	14,10%
Wanga et al (2010)	99,08%	96,70%	80,00%	**58,57%**	**76,92%**
Xiang et al (2008)	96,80%	98,66%	93,40%	46,97%	71,43%
Horng et al (2011)	99,30%	99,50%	97,50%	28,80%	19,70%
Badran and Rockett (2012)	**99,50%**	96,99%	78,01%	5,59%	11,40%
Koc et al (2012)	-	**99,60%**	-	-	-

Tableau 34 Comparaison entre nos modèles et les autres modèles de détection d'intrusion pour la classification des différentes catégories de connexions

Comme le montre la figure suivante, les meilleurs modèles dans la classification du comportement normal, les attaques DoS, les attaques Probe, les attaques R2L, les attaques U2R sont respectivement le modèle de Badran and Rockett (2012), le modèle de Koc et al (2012), le modèle de notre première proposition, le modèle de Wanga et al (2010) et le modèle de Wanga et al (2010).

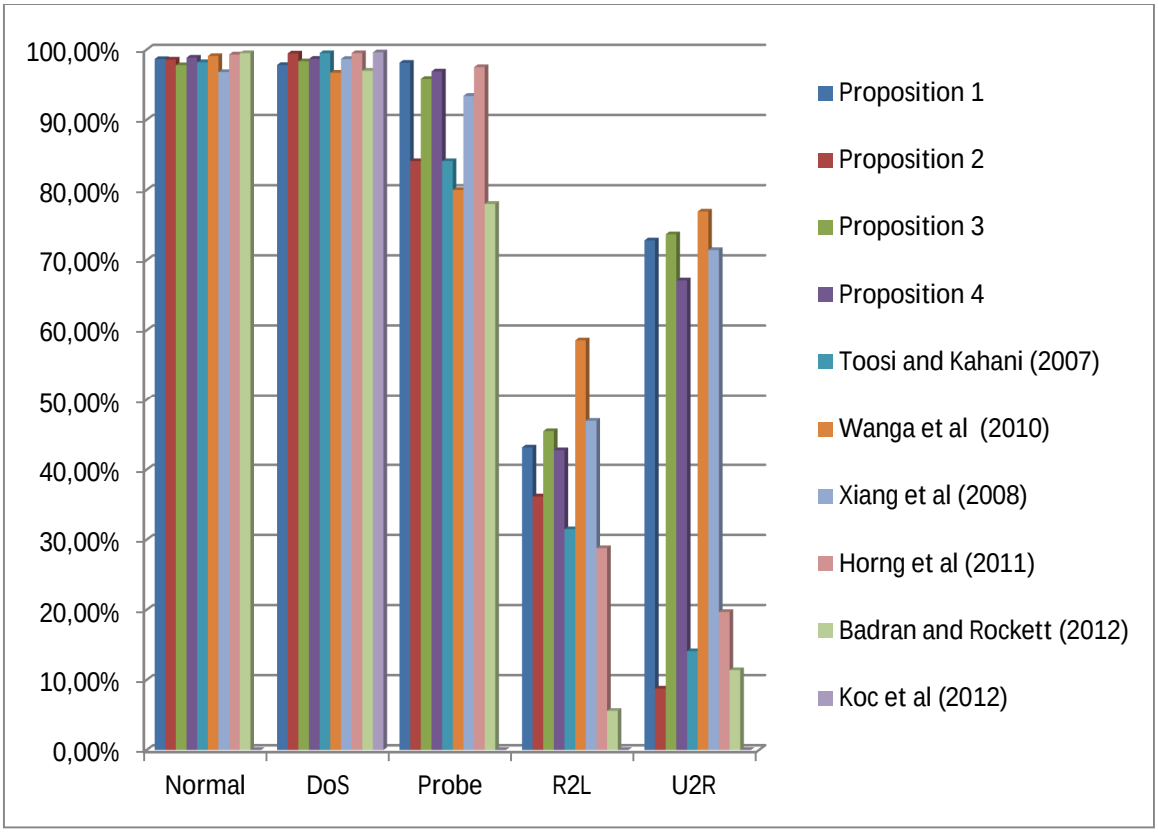

Figure 47 Comparaison entre nos modèles et les autres modèles de détection d'intrusion pour la classification des différentes catégories de connexions

Le tableau 35 résume les métriques de classification globales de nos modèles ainsi que les autres modèles adaptatifs et récents.

	TD	TFA	Exactitude
Proposition 1	94,26%	1,35%	95,12%
Proposition 2	**95,02%**	1,43%	95,72%
Proposition 3	94,84%	2,23%	95,41%
Proposition 4	95,01%	1,15%	**95,76%**
Toosi and Kahani (2007)	94,77%	1,90%	95,30%
Wanga et al (2010)	93,94%	0,92%	94,94%
Xiang et al (2008)	93,93%	3,20%	94,49%
Horng et al (2011)	94,82%	0,70%	95,70%
Badran and Rockett (2012)	90,69%	**0,50%**	92,41%
Koc et al (2012)	-	-	93,72%

Tableau 35 Comparaison entre nos modèles et les autres modèles de détection d'intrusion par rapport à TD, TFA et Exactitude

Comme le montre la figure 48 les meilleurs modèles par rapport au TD, TFA, Exactitude sont respectivement le modèle de notre deuxième proposition, le modèle de Badran and Rockett (2012), le modèle de notre quatrième proposition. Nos différents modèles

représentent les quatre modèles les plus performants, où ils donnent les quatre plus élevés taux de détection et les quatre plus élevés taux d'exactitude.

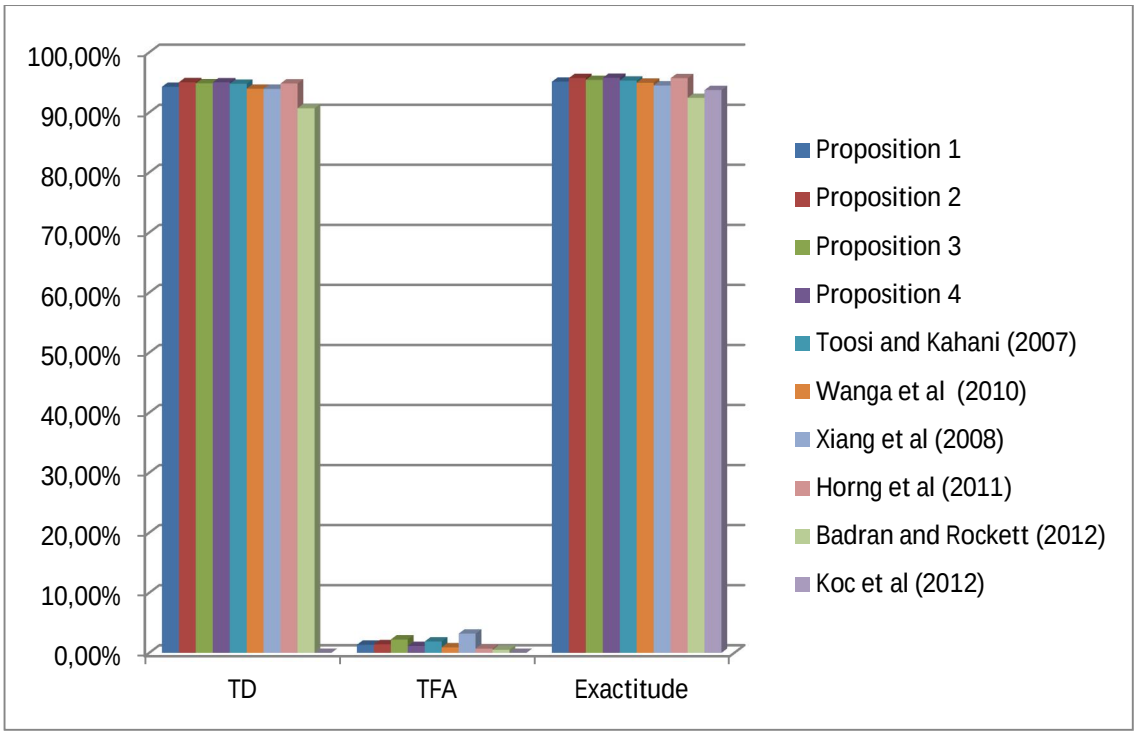

Figure 48 Comparaison entre nos modèles et les autres modèles de détection d'intrusion par rapport à TD, TFA et Exactitude

Cette étude comparative a montré que nos différents modèles sont très efficaces et très performants par rapport aux modèles de détection d'intrusions existants.

9.5. Conclusion

Dans ce chapitre nous avons montré le positionnement nos solutions pour créer des systèmes de détection d'intrusion adaptatifs par rapport aux solutions existantes. Pour chacun de nos quatre modèles proposés, nous avons présenté ses nouveautés, ses avantages ainsi que ses inconvénients. À la fin de ce chapitre, nous avons présenté une étude comparative entre nos modèles et d'autres IDSs adaptatifs ainsi que certains IDSs récents. Cette étude comparative a prouvé la grande efficacité et la haute performance de nos modèles par rapport aux IDSs existants.

Conclusion générale

I. Introduction

Parmi les différents outils de sécurité informatique, on trouve le système de détection d'intrusion. Cet outil est devenu très indispensable pour tout réseau informatique, il nous permet de connaitre toutes activités anormales qui peuvent présenter un danger pour notre réseau. Le développement des systèmes de détection d'intrusion a passé par deux générations. La première génération est la génération ad hoc, cette génération a montré beaucoup de limites avec la rapidité et l'augmentation du trafic réseau. La deuxième génération a été proposée afin de traiter les problèmes de la première génération, où les techniques de data mining ont été utilisé. Malgré la puissance et l'efficacité des techniques de data mining, les systèmes de détection d'intrusion de la deuxième génération souffrent de certaines limites comme la nécessité de faire une mise à jour régulière, la nécessité de préparer les données d'apprentissage, la difficulté de détecter les nouvelles formes d'attaques… etc. Les systèmes de détection d'intrusion nommés adaptatifs sont proposés afin de traiter ces limites.

II. Contribution

Dans cette thèse nous avons créé quatre modèles pour traiter les problèmes de la deuxième génération des systèmes de détection d'intrusion. Le premier modèle est un système de détection d'intrusion très rapide en termes de temps d'apprentissage, ce qui nous garantit un apprentissage initial et un réapprentissage dans le plus bref délai tout en conservant une bonne capacité de généralisation. Le deuxième modèle est un système de détection d'intrusion avec une très grande capacité de généralisation ce qui lui permet de détecter les nouvelles formes d'attaques. Le troisième modèle est un système de détection d'intrusion avec un mode d'apprentissage continu ce qui garantit l'adaptation de notre modèle avec l'environnement cible. Le quatrième modèle a le même but que le deuxième modèle, mais avec une structure qui ressemble au premier modèle. Nos quatre modèles représentent les différentes implémentations de nos trois solutions proposées: la création d'un système de détection d'intrusion très rapide en termes d'apprentissage, la création d'un système de

détection d'intrusion qui possède une grande capacité de généralisation, la création d'un système de détection d'intrusion avec un mode d'apprentissage continu. Les différents systèmes de détection d'intrusion que nous avons proposés ont montré des très hautes performances par rapport aux travaux de recherche publies par des grands chercheurs dans des journaux renommés.

III. Perspectives

Dans nos futurs travaux nous allons tester l'efficacité de d'autres solutions comme l'utilisation des techniques de data mining adaptatif ainsi que la combinaison de plusieurs solutions dans le même modèle. La combinaison de deux approches d'apprentissage comportementale et par scénario représente une autre perspective sur laquelle nous allons travailler. L'utilisation d'un autre ensemble de données de test et d'apprentissage comme le KDD ISCX 2012 (Shiravi et al, 2012) est aussi l'une de nos principales perspectives.

Références

(Abadeh et al, 2007) Abadeh, M. S., Habibi, J., Barzegar, Z., Sergi, M. (2007). A parallel genetic local search algorithm for intrusion detection in computer networks. Engineering Applications of Artificial Intelligence, Vol. 20, N. 8, pp.1058–1069.

(Ahmim and Ghoualmi-Zine, 2013) AHMIM, A., GHOUALMI-Zine, N. (2013), A New Fast and High Performance Intrusion Detection System, International Journal of Security and Its Applications, vol. 7, No. 5, pp.67-80.

(Ahmim and Ghoualmi-Zine, 2014) AHMIM, A., GHOUALMI-Zine, N. (2014), A new adaptive intrusion detection system based on the intersection of two different classifiers, international journal of security and network, vol. 9, No. 3, pp 125-132.

(Ahmim and Ghoualmi-Zine, 2015) AHMIM, A., GHOUALMI-Zine, N. (2015), A new hierarchical intrusion detection system based on a binary tree of classifiers, international journal of information management and computer security.

(AHMIM et al, 2010) AHMIM, A., GHOUALMI, N., NOUDJOUD, K. (2010) Intrusion Detection by optimized GASSATA, International Symposium on Modeling and Implementation of Complex Systems MISC 2010, Constantine, Algeria, May 30-31, 2010.

(AHMIM et al, 2011) AHMIM, A., GHOUALMI, N., NOUDJOUD, K. (2010) Improved Off-Line Intrusion Detection Using a Genetic Algorithm and RMI, the 1st international conference on information systems and technologies ICIST 2011, Tébessa, Alegria, April, 2011.

(Anderson, 1980) Anderson J. (1980), Computer security threat monitoring and surveillance.

(Badran and Rockett, 2012) Badran, K., Rockett, P. (2012), Multi-class pattern classification using single, multi-dimensional feature-space feature extraction evolved by multi-objective genetic programming and its application to network intrusion detection, Genetic Programming and Evolvable Machines, Vol. 13, N.1, pp.33-63.

(Balajinath and Raghavan, 2000) Balajinath, B. and Raghavan, S. V. (2000). Intrusion detection through learning behavior model, Computer Communication, Vol. 24, No. 12, pp. 1202–1212.

(Bensefia and Ahmed-Nacer, 2008) Bensefia, H., Ahmed-Nacer, M. (2008), Towards an Adaptive Intrusion Detection System: a Critical and Comparative Study, 2008 International Conference on Computational Intelligence and Security, Guangzhou, China, 13-17 Dec 2008.

(Bensefia, 2009) Bensefia, H. (2009), vers un modèle de détection d'intrusion adaptatif intégrant les systèmes connexionniste évolutif et systèmes de classeurs apprenants, mémoire de magistère, Badji Mokhtar-Annaba University, Algeria.

(Bishop, 1996) Bishop, C, M. (1996), Neural Networks for Pattern Recognition, OXFORD University press, England, ISBN-13: 978-0198538646.

(Bouzida et al, 2004) Bouzida, Y., Cuppens, F., Cuppens-Boulahia, N., Gombault, S. (2004), Efficient intrusion detection using principal component analysis, la 3eme conférence sur la sécurité et architectures réseaux (SAR). La Londe, France, June 2004.

(Breiman et al, 1984) Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C J. (1984), Classification and Regression Trees. Wadsworth International Group, Belmont, California.

(Breiman, 2001) Breiman, L. (2001), Random Forests, Machine Learning, Vol. 45, No.1, pp. 5-32.

(Bridges and Vaughn, 2000) Bridges, S. M., Vaughn, R. B. (2000), Intrusion detection via fuzzy data mining, the twelfth annual Canadian information technology security symposium. Ottawa, USA, June 19-23, 2000.

(Bugmann, 1998) Bugmann, G. (1998), "Normalized Gaussian Radial Basis Function networks", Neurocomputing, Vol.20, No. 1–3, pp. 97–110.

(Chang and lin, 2001) Chang, C., Lin, C. (2001). LIBSVM - A Library for Support Vector Machines. URL http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

(Chavan et al, 2004) Chavan, S., Shah, K. D. N., Mukherjee, S. (2004). Adaptive neuro-fuzzy intrusion detection systems, the international conference on information technology: Coding and computing (ITCC'04), Las Vegas, NV, USA, 5-7 April 2004

(Chen et al, 2005) Chen, W.H., Hsu, S.H., Shen, H.P. (2005), Application of SVM and ANN for intrusion detection, Computer and Operations Research, Vol.32, No. 10, pp. 2617–2634.

(Chen et al, 2007) Chen, Y., Abraham, A., & Yang, B. (2007), Hybrid flexible neural-tree-based intrusion detection systems, International Journal of Intelligent Systems, Vol.22, No. 4, pp. 337–352.

(Chimphlee et al, 2005) Chimphlee, W., Addullah, A. H., Sap, M. N. M., Srinoy, S., Chimphlee, S. (2006), Anomaly-based intrusion detection using fuzzy rough clustering, the international conference on hybrid information technology (ICHIT'06), Cheju Island, 9-11 Nov. 2006.

(Cohen, 1995) Cohen, W., (1995), Fast effective rule induction, the 12th international conference on machine learning, ICML. Morgan Kaufmann, Tahoe City, pp 115–123.

(Cole et al, 2005) : Cole, E., Krutz, R., Conley, J. (2005), Network Security Bible, Wiley Publishing, Inc, ISBN13:978-0-7645-7397-2.

(Curt et al, 2012) Curt, L., Samson, P., Wooten, D., Geniesse, G., Lucas, M., Fancourt, G., Gerstenberger, J., Euliano, N., Lynn, G., Allen, M., Marossero, D. (2012), NeuroSolutions, version 6.02. Copyright. NeuroDimension Inc., www.nd.com 1994–2012, available at: http:// www.nd.com (accessed June 2012).

(DARPA, 1999) MIT Lincoln Labs, 1998 DARPA Intrusion Detection Evaluation. Available on:

http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/index.html, February 2008.

(Debar et al, 1992) Debar, H., Becker, M., Siboni, D., (1992), A neural network component for an intrusion-detection system. the 1992 IEEE Computer Society Symposium on Research in Security and Privacy, pages 240-250, Oakland, CA, May 1992.

(Debar et al, 2000) Debar, H., Dacier, M., Wespi, A. (2000), A Revised Taxonomy for Intrusion-Detection Systems, Annales des Télécommunications, Vol. 55, No. 7-8, pp. 361-378.

(Denning, 1987) Denning, D. E. (1987), An intrusion-detection model. IEEE Transactions on software engineering, Vol. SE-13, No. 2, pp. 222–232.

(Depren et al, 2005) Depren, O., Topallar, M., Anarim, E., Ciliz, M. K. (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. Expert Systems with Applications, Vol. 29, No. 4, pp.713–722.

(Dowell and Ramstedt , 1990) Dowell, C., Ramstedt P. (1990), The ComputerWatch data reduction tool, the 13th National Computer Security Conference, pp. 99-108, Washington, DC, October 1990.

(Dua and Xian, 2011) Dua, S., Xian, D. (2001), Data Mining and Machine Learning in Cybersecurity, Taylor and Francis Group, LLC, ISBN: 13: 978-1-4398-3943-0.

(Ebel et al, 2009) Ebel, F., Baudru, S., Crocfer, R., Puche, D., Hennecart, J., Lasson, S., Agé, M. (2009), Sécurité informatique (Ethical Hacking), Editions ENI, Octobre 2009, ISBN: 978-2-7460-5105-8

(Endorf et al, 2004) Endorf, C., Schultz, E., Mellander, J. (2004), Intrusion Detection and Prevention, ISBN:0072229543.

(Eskin et al, 2002) Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S. (2002). A geometric framework for unsupervised anomaly detection, Applications of Data Mining in Computer Security, Advances in Information Security, Vol. 6, pp. 77-101.

(Fan et al, 2004) Fan, W., Lee, W., Miller, M., Stolfo, S. J., Chan, P. K. (2004), Using artificial anomalies to detect unknown and known network intrusions, Knowledge and Information Systems, Data Mining, 2001. ICDM 2001, pp.507–527, San Jose, CA, 29 Nov 2001-02 Dec 2001.

(Florez et al, 2002) Florez, G., Bridges, S. M., Vaughn, R. B. (2002), An improved algorithm for fuzzy data mining for intrusion detection, the North American fuzzy information processing society conference (NAFIPS 2002). New Orleans, LA, 2002.

(Forrest et al, 1997) Forrest, S., Hofmeyr, S. A., Somayaji, A. (1997), Computer immunology, Communications of the ACM, Vol. 40, No.10, pp.88-96.

(Gaines and Compton, 1995) Gaines, B, R., Compton, P. (1995), Induction of Ripple-Down Rules Applied to Modeling Large Databases, Journal of Intelligent Information Systems, Vol. 5, No. 3, pp. 211 - 228.

(Gallinari et al, 1988) Gallinari P, Thiria S, and Fogelman-Soulie F. (1988), Multilayer perceptrons and data analysis, the IEEE Annual International Conference on Neural Networks (ICNN88), Vol. I, pages 391-399, San Diego, CA, July 1988.

(Giacinto and Roli, 2003) Giacinto, G., Roli, F. (2003). Intrusion detection in computer networks by multiple classifier systems, the proceeding of ICPR 2002, 16th international conference on pattern recognition. Quebec City, Canada, 2003.

(Giacinto et al, 2006) Giacinto, G., Perdisci, R., Rio, M. D., Roli, F. (2006). Intrusion detection in computer networks by a modular ensemble of one-class classifiers. Information Fusion, Vol. 9, No.1, pp.69–82.

(Habra et al, 1992) Habra, N., Le Charlier, B., Mounji, A., Mathieu, I. (1992), Asax: Software architecture and rule-based language for universal audit trail analysis, the Second European Symposium on Research in Computer Security (ESORICS), Toulouse, France, November 1992.

(Han and Cho, 2003) Han, S.J., Cho, S.B. (2003), Detecting intrusion with ruled-based integration of multiple models. Computers and Security, Vol. 22, No. 7, pp. 613–623.

(Haykin, 1999) Haykin, S. (1999). Neural networks: A comprehensive foundation (2nd ed.), New Jersey: Prentice Hall, ISBN-10: 0132733501.

(Haystack Labs, 1997) Haystack Labs, Inc. Stalker (1997), Available from the company's website at http://www.haystack.com/stalk.htm.

(Heller et al, 2003) Heller, K. A., Svore, K. M., Keromytis, A. D., Stolfo, S. J. (2003). One class support vector machines for detecting anomalous window registry accesses, the 3rd IEEE conference data mining workshop on data mining for computer security. Florida, 2003.

(Helman and Liepins, 1993) Helman, P., Liepins, G. (1993), Statistical foundations of audit trail analysis for the detection of computer misuse, IEEE Transactions on Software Engineering, Vol. 19, No. 9, pp.886-901.

(Helman et al, 1992) Helman, P., Liepins, G., Richards, W. (1992), Foundations of intrusion detection, the Fifth Computer Security Foundations Workshop, pages 114-120, Franconic, NH, June 1992.

(Holland, 1975) Holland, J.H. (1975), Adaptation in natural and artificial system, Ann Arbor, The University of Michigan Press, 1975.

(Horng et al, 2011) Horng, S., Su, M., Chen, Y., Kao, T., Chen, R., Lai, J., Perkasa, C, D. (2011), A novel intrusion detection system based on hierarchical clustering and support vector machines, Expert Systems with Applications, Vol.38, No.1, pp.306–313.

(Hühn and Hüllermeier, 2009) Hühn, J., Hüllermeier, E. (2009), FURIA: an algorithm for unordered fuzzy rule induction, Data Mining and Knowledge Discovery, Vol. 19, No. 3, pp. 293-319.

(Internet Security Systems, 1997) Internet Security Systems, Inc. (1997), RealSecure. Internet http://www.iss.net/prod/rsds.html.

(ISO/IEC 27000, 2009): ISO/IEC 27000. (2009), Information technology — Security techniques — Information security management systems — Overview and vocabulary, http://standards.iso.org/ittf/licence.html

(Jang et al, 1996) Jang, J.-S., Sun, C.-T., Mizutani, E. (1996). Neuro-fuzzy and soft computing: A computational approach to learning and machine intelligence. New Jersey: Prentice Hall, ISBN-10: 0132610663

(Javitz and Valdes, 1991) Javitz, H., Valdes, A. (1991), The SRI IDES statistical anomaly detector, the IEEE Symposium on Research in Security and Privacy, pp. 316-326, Oakland, CA, May 1991.

(Javitz et al, 1993) Javitz, H. S., Valdez, A., Lunt, T. F., Tamaru, A., Tyson, M., Lowrance, J. (1993), Next generation intrusion detection expert system (NIDES) - 1. Statistical algorithms rationale - 2. Rationale for proposed resolver. Technical Report A016-Rationales, SRI International, 333 Ravenswood Avenue, Menlo Park, CA, March 1993.

(Jiang et al, 2006) Jiang, S. Y., Song, X., Wang, H., Han, J.-J., Li, Q.H. (2006). A clustering-based method for unsupervised intrusion detections. Pattern Recognition Letters, Vol.27, No. 7, pp. 802–810.

(John and Langley, 1995) John, G. H., Langley, P. (1995), Estimating Continuous Distributions in Bayesian Classifiers, the eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338-345, San Mateo, 1995.

(Johnson et al, 2011) Johnson, A., Dempsey, K., Ross, R., Gupta, S., Bailey, D. (2011), Guide for Security-Focused Configuration Management of Information Systems, national Institute of standard and Technology.

(Joo et al, 2003) Joo, D., Hong, T., Han, I. (2003). The neural network models for IDS based on the asymmetric costs of false negative errors and false positive errors, Expert System with Applications, Vol. 25, No. 1, pp. 69–75.

(Kang et al, 2005) Kang, D. K., Fuller, D., Honavar, V. (2005), Learning classifiers for misuse and anomaly detection using a bag of system calls representation, Information Assurance Workshop 2005, IAW '05 ,15-17 June 2005.

(Kayacik et al, 2007) Kayacik, H. G., Nur, Z.-H., Heywood, M. I. (2007). A hierarchical SOM-based intrusion detection system, Engineering Applications of Artificial Intelligence, Vol. 20, No. 4, pp.439–451.

(KDD'99, 1999) The KDD CUP 1999 Data. (1999), available at: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed June 2012).

(Khan et al, 2007) Khan, L., Awad, M., Thuraisingham, B. (2007), A new intrusion detection system using support vector machines and hierarchical clustering, The VLDB Journal, Vol. 16, No. 4, pp.507–521.

(Khor and Ting, 2012) Khor, K., Ting, C., Phon-Amnuaisuk, S. (2012), A cascaded classifier approach for improving detection rates on rare attack categories in network intrusion detection , Applied Intelligence, Vol. 36, No. 2, pp. 320-329.

(Kittler et al, 1998) Kittler, J., Hatef, M., Duin, R. P. W., Matas, J. (1998), On combining classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No. 3, pp. 226–239.

(Koc et al, 2012) Koc, L., Mazzuchi, T, A., Sarkani, S. (2012), A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier, Expert Systems with Applications, Vol. 39, No. 18, pp. 13492–13500.

(Kohonen, 1982) Kohonen, T. (1982), Self-organized formation of topologically correct feature maps, Biological Cybernetics, Vol. 43, No.1, pp. 59–69.

(Kohonen, 2001) Kohonen, T. (2001), Self-Organizing Maps, Springer Series in Information Sciences, Vol. 30, ISBN 978-3-642-56927-2.

(Koza, 1992) Koza, J. R. (1992), Genetic programming: On the programming of computers by means of natural selection, Massachusetts: MIT.

(Kumar and Spafford, 1994) Kumar, S., Spafford, E. (1994), A pattern matching model for misuse intrusion detection, the 17th National Computer Security Conference, pp. 11-21, Baltimore, Md, October 1994.

(Lee and Stolfo, 1998) Lee, W., and Stolfo, S. (1998), Data mining approaches for intrusion detection, the proceedings of the seventh USENIX security symposium (SECURITY'98). San Antonio, TX, January 26-29, 1998.

(Lee and Stolfo, 2000) Lee, W., & Stolfo, S. (2000), A framework for constructing features and models for intrusion detection systems, ACM Transactions on Information and System Security, Vol. 3, No. 4, pp. 227–261.

(Li and Guo, 2007) Li, Y., Guo, L. (2007), An active learning based TCM-KNN algorithm for supervised network intrusion detection, Computer and Security, Vol. 26, No. 7-8, pp. 459–467.

(Liao and Vemuri, 2002) Liao, Y., and Vemuri, V. R. (2002), Use of K-nearest neighbor classifier for intrusion detection, Computer and Security, Vol. 21, No. 5, pp. 439–448.

(Liorens et al, 2006) Liorens, C., Levier, L., Valois, D. (2006), Tableaux de bord de la sécurité réseau, éditions eyrolles, ISBN: 2-212-11973-9.

(Lippmann et al, 2000) Lippmann R. P. , Fried D. J., Graf I., Haines J. W., Kendall K. R., McClung D. , Weber D. , Webster S. E. , Wyschogrod D. , Cunningham R. K. , and Zissman M. A. (2000), Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation, DARPA Information Survivability Conference and Exposition, Vol. 02, pp. 12 – 26, Hilton Head, SC, 25-27 Jan 2000.

(Liu and Yi, 2006) Liu, G., Yi, Z. (2006), Intrusion detection using PCASOM neural networks. Third International Symposium on Neural Networks, Chengdu, China, May 28 - June 1, 2006 .

(Liu et al, 2004) Liu, Y., Chen, K., Liao, X., Zhang, W. (2004), A genetic clustering method for intrusion detection, Pattern Recognition, Vol. 37, No. 5, pp. 927–942.

(Liu et al, 2007) Liu G., Z. Yi, Yang S. (2007), A hierarchical intrusion detection model based on the PCA neural networks, Neurocomputing, Vol.70, No. 7-9, pp. 1561-1568.

(Luo and Bridgest, 2000) Luo, J., Bridgest, S. M. (2000), Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection. International Journal of Intelligent Systems, Vol. 15, No. 8, pp. 687–703.

(Mahmoudi and Ghoualmi, 2010) Mahmoudi, R., Ghoualmi, N. (2010), Crossover and mutation based cloning parent for degree Constrained Minimum Spanning Tree Problem (d-MSTP), Second International Conference on Engineering Systems Management and Its Applications (ICESMA), March 30 2010-April 1 2010, Sharjah.

(Manocha and Girolami, 2007) Manocha, S., Girolami, M. A. (2007), An empirical analysis of the probabilistic Knearest neighbour classifier, Pattern Recognition Letters, Vol. 28, No. 13, pp. 1818–1824.

(Mccallum and Nigam, 1998) Mccallum, A., Nigam, K. (1998), A Comparison of Event Models for Naive Bayes Text Classification, the AAAI-98 Workshop on Learning for Text Categorization.

(McHugh, 2000) McHugh J. (2000), Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory, ACM Transactions on Information and System Security, Vol. 3, No. 4, pp. 262–294, 2000.

(Mé and Alanou, 1996) Mé, L., Alanou, V. (1996), Détection d'intrusions dans un système informatique: méthodes et outils, TSI Journal, Vol. 96, No. 4, pp. 429-450.

(Mé, 1995) Mé, L. (1995), Un algorithme génétique pour détecter des intrusions dans un système informatique. Valgo Journal, Vol 95, pp. 68-78.

(Michel and Mé, 2001) Michel, C., Mé, L. (2001), Adele: an Attack Description Language for Knowledge-based Intrusion Detection , the 16th International Conference on Information Security. Kluwer. June 2001.

(Mitchell, 1997) Mitchell, T. (1997), Machine learning, New York: McGraw Hill.

(Moradi and Zulkernine, 2004) Moradi, M., Zulkernine, M. (2004), A neural network based system for intrusion detection and classification of attacks, the 2004 IEEE international conference on advances in intelligent systems – Theory and applications. Luxembourg, November 15-18, 2004.

(Mukkamala et al, 2004) Mukkamala, S., Sung, A. H., Abraham, A. (2004), Modeling intrusion detection systems using linear genetic programming approach, the 17th

International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2004, Ottawa, Canada, May 17-20, 2004.

(Mukkamala et al, 2005) Mukkamala, S., Sung, A. H., Abraham, A. (2005), Intrusion detection using an ensemble of intelligent paradigms. Network and Computer Applications, Vol. 28, No. 2, pp.167–182.

(NSL-KDD, 2009) NSL-KDD (2009), available at: http://nsl.cs.unb.ca/NSL-KDD/, (accessed March 2013).

(Ozyer et al, 2007) Ozyer, T., Alhajj, R., Barker, K. (2007). Intrusion detection by integrating boosting genetic fuzzy classifier and data mining criteria for rule pre-screening. Journal of Network and Computer Applications, Vol. 30, No. 1, pp. 99–113.

(Pearl, 1988) Pearl, J. (1988), Probabilistic reasoning in intelligent systems: Networks of Plausible Inference, Morgan Kaufmann, ISBN-10: 1558604790.

(Peddabachigari et al, 2004) Peddabachigari, S., Abraham, A., Thomas, J. (2004), Intrusion detection systems using decision trees and support vector machines, International Journal of Applied Science and Computations, available at : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.60.4079&rep=rep1&type=pdf.

(Peddabachigari et al, 2007) Peddabachigari, S., Abraham, A., Grosan, C., Thomas, J. (2007), Modeling intrusion detection system using hybrid intelligent systems. Journal of Network and Computer Applications, Vol. 30, No. 1, pp. 114–132.

(Platt, 1999) Platt, J. (1999), Fast training of support vector machines using sequential minimal optimization, Advances in kernel methods, MIT Press Cambridge, MA, USA, pp.185 – 208, ISBN:0-262-19416-3.

(Porras and Kemmerer, 1992) Porras, P., Kemmerer, R. (1992), Penetration state transition analysis : A rule-based intrusion detection approach, the Eighth Annual Computer Security Applications Conference, pp. 220-229, San Antonio, TX, 30 Nov 1992-04 Dec 1992.

(Quinlan, 1993) Quinlan, R. (1993), C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA, ISBN:1-55860-238-0.

(Ramos and Abraham, 2005) Ramos, V., and Abraham, A. (2005), ANTIDS: Self organized ant based clustering model for intrusion detection system, the proceedings of the fourth IEEE international workshop on soft computing as transdisciplinary science and technology (WSTST'05).

(Richardson, 2007) Richardson R, (2007), the 12th Annual CSI, COMPUTER CRIME AND SECURITY SURVEY 2007.

(Sarle et al, 1994) Sarle, W. S. (1994), Neural networks and statistical models, the Nineteenth Annual SAS Users Group International Conference, pp. 1538-1550, Cary, NC, April 1994.

(Schneier, 2004) Schneier, B. (2004), Secrets and Lies: DIGITAL SECURITY IN A NETWORKED WORLD, Wiley Publishing, Inc, ISBN: 0471453803.

(Schultz et al, 2001) Schultz, M. G., Eskin, E., Zadok, E., Stolfo, S. J. (2001), Data mining methods for detection of new malicious executables, the 2001 IEEE symposium on security and privacy (SP'01), Oakland, CA, 14-16 May 2001.

(Scott, 2004) Scott, S. L. (2004), A Bayesian paradigm for designing intrusion detection systems, Computational Statistics and Data Analysis, Vol. 45, No. 1, pp. 69–83.

(Shi, 2007) Shi, H. (2007). Best-first decision tree learning. University of Waikato, Hamilton, NZ, available at : http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=507277E62C0B339FCB5A27A7990197EE?doi=10.1.1.149.2862&rep=rep1&type=pdf

(Shiravi et al, 2012) Shiravi, A., Shiravi, H., Tavallaee, M., Ghorbani, A. A. (2012) Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Computers and Security, Vol. 31, No. 3 pp.357-375.

(Shon and Moon, 2007) Shon, T., Moon, J. (2007), A hybrid machine learning approach to network anomaly detection, Information Sciences, Vol. 177, No. 18, pp. 3799–3821.

(Shon et al, 2006) Shon, T., Kovah, X., Moon, J. (2006), Applying genetic algorithm for classifying anomalous TCP/IP packets, Neurocomputing, Vol. 69, No. 16-18, pp. 2429–2433.

(Shyu et al, 2003) Shyu, M., Chen, S., Sarinnapakorn, K., Chang, L. (2003). A novel anomaly detection scheme based on principal component classifier, IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with ICDM'03, 2003.

(Stein et al, 2005) Stein, G., Chen, B., Wu, A. S., Hua, K. A. (2005), Decision tree classifier for network intrusion detection with GA-based feature selection, the 43rd annual Southeast regional conference. Kennesaw, Georgia, 2005.

(Stephen et al, 1993) Stephen, E., Hansen, E., Todd A. (1993), Automated system monitoring and notification with swatch, the Seventh Systems Administration Conference (LISA '93), Monterey, CA, November 1993.

(Stolfo et al, 2000) Stolfo S. J., Fan W., W., Prodromidis, L, A., Chan P. K., (2000), Cost-based modeling for fraud and intrusion detection: Results from the jam project, DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00, Vol. 02, p. 1130, Hilton Head, SC, 25-27 Jan 2000.

(Tavallaee et al, 2009) Tavallaee, M., Bagheri, E., Lu, W., Ghorbani A., A Detailed Analysis of the KDD CUP 99 Data Set, IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009, pp. 1-6, Ottawa, ON, 8-10 July 2009.

(Tian et al, 2004) Tian, M., Chen, S. -C., Zhuang, Y., Liu, J. (2004). Using statistical analysis and support vector machine classification to detect complicated attacks, the third international conference on machine learning and cybernetics. Shanghai, 26-29 Aug. 2004.

(Toosi and Kahani, 2007) Toosi, A. N., Kahani, M.(2007), A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers, Computer Communications,Vol.30, No.10, pp.2201–2212.

(Trend Micro, 2011) Trend Micro. (2011), IT Security FOR DUMMIES, North American Small Business Edition, Wiley Publishing, Inc, 2011, ISBN: 978-1-118-08410-6.

(Tsaia et al, 2009) Tsaia, C., Hsub, Y., Linc, C., Lin, W. (2009), Intrusion detection by machine learning: A review, Expert Systems with Applications, Vol. 36, No. 10, pp 11994–12000.

(Tsang et al, 2007) Tsang, C.H., Kwong, S., Wang, H. (2007), Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. Pattern Recognition, Vol. 40, No. 9, pp. 2373–2391.

(Tutschku, 1995) Tutschku, K. (1995), Recurrent Multilayer Perceptrons for Identification and Control: The Road to Applications, Report No. 118, Institute of Computer Science, University of Wurzburg, June 1995.

(Vaccaro and Liepins, 1989) Vaccaro H. S., Liepins G. E. (1989), Detection of anomalous computer session activity, the 1989 IEEE Symposium on Research in Security and Privacy, pages 280-289, Oakland, CA, 1-3 May 1989.

(Vapnik, 1998) Vapnik, V. (1998), Statistical Learning Theory (Adaptive and Learning Systems for Signal Processing, Communications and Control Series), New York: John Wiley, ISBN-10: 0471030031

(Wang and Battiti, 2006) Wang, W., Battiti, R. (2006), Identifying intrusions in computer networks with principal component analysis, the first international conference on availability, reliability and security (ARES'06), 20-22 April 2006.

(Wang and Stolfo, 2004) Wang, K., Stolfo, S. J. (2004), Anomalous Payload-based network intrusion detection, the 7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15 - 17, 2004.

(Wang et al, 2004) Wang, W., Guan, X., & Zhang, X. (2004). A novel intrusion detection method based on principle component analysis in computer security, the international symposium on neural networks. Dalian, China, August 19-21, 2004

(Wang et al, 2006) Wang, Y., Kim, I., Mbateng, G., and Ho, S.-Y. (2006), A latent class modeling approach to detect network intrusion, Computer Communications, Vol. 30, No. 1, pp. 93–100.

(Wanga et al, 2010) Wanga, G., Hao, J., Mab, J., Huanga, L. (2010), A new approach to intrusion detection using artificial neural networks and fuzzy clustering, Expert Systems with Applications, Vol. 37, No. 9, pp. 6225–6232.

(WheelGroup Corporation) WheelGroup Corporation. Brochure of the Netranger intrusion detection system. Available from the company's website at http://www.wheelgroup.com/netrangr/netranger_broch.html.

(Witten et al, 2011) Witten, I., Frank, E., Hall, M. (2011), Data Mining: Practical Machine Learning Tools and Techniques, Elsevier Inc, (2011).

(Wood and Erlinger, 2012) Wood, M., Erlinger, M. (2012) Intrusion detection message exchange requirements, The RFC Archive: http://www.ietf.org/rfc/rfc4766.txt, Consulté le 09 mai 2012.

(Xiang and Lim, 2007) Xiang, C., & Lim, S. M. (2005). Design of multiple-level hybrid classifier for intrusion detection system, the IEEE workshop machine learning for signal processing, pp. 117 – 122, Mystic, CT, 28-28 Sept. 2005.

(Xiang et al, 2008) Xiang, C., Yong, Chin, P., Meng, Lim, S. (2008), Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees, Pattern Recognition Letters, Vol. 29, No. 7, PP. 918-924.

(Xiaonan and Banzhaf, 2010) Xiaonan Wu, S., Banzhaf, W. (2010), The use of computational intelligence in intrusion detection systems: A review, Applied Soft Computing, Vol. 10, No.1, PP 1–35, 2010.

(Zhang and Shen, 2005) Zhang, Z., Shen, H. (2005). Application of online-training SVMs for real-time intrusion detection with different considerations, Computer Communications, Vol. 28, No. 12, pp. 1428–1442.

(Zhang et al, 2004) Zhang, L.-H., Zhang, G.-H., Yu, L., Zhang, J., Bai, Y.-C. (2004). Intrusion detection using rough set classification. Journal of Zhejiang University Science, Vol. 5, No.9, pp.1076–1086.

(Zhang et al, 2005) Zhang, C., Jiang, J., Kamel, M. (2005). Intrusion detection using hierarchical neural network. Pattern Recognition Letters, Vol. 26, No. 6, pp. 779–791.

(Zhang et Zulkernine, 2005) Zhang J., Zulkernine M., Network Intrusion Detection using Random Forests, Third Annual Conference on Privacy, Security and Trust, pp. 53-61, Canada, October 2005.

(Zimmermann, 2001) Zimmermann, H. (2001). Fuzzy set theory and its applications. Kluwer Academic Publishers, ISBN-10: 0792374355.

Annexes

Annexe 1 Liste des abréviations

ASCII: American Standard Code for Information Interchange

BGP: Border Gateway Protocol

CART: Classification And Regression Trees

CSI: Computer Security Institute

DAA: Designated Approval Authority

DAG: Directed Acyclic Graph

DARPA: Defense Advanced Research Projects Agency

DDoS: Distributed Denial of Service

DNS: Domain Name System

DoS: Denial of Service

DR: Detection Rate

DT: Decision tree

EAP: Extensible Authentication Protocol

EGP: Exterior Gateway Protocol

FN: False negative

FP: False positive

GA: Genetic algorithm

H-IDS: Host based Intrusion Detection System

ICMP: Internet Control Message Protocol

ID: Intrusion Detection

IDS: Intrusion Detection System

IEC: International Engineering Consortium

IEEE: Institute of Electrical and Electronics Engineers

IETF: Internet Engineering Task Force

IGP: Interior gateway protocol

IP: Internet Protocol

IPS: Intrusion Prevention System

IPsec: Internet Protocol security

IPV6: Internet Protocol version 6

IS-IS: Intermediate system to intermediate system

ISO: International Organization for Standardization

KDD: Knowledge Discovery in Databases

K-NN: K-Nearest Neighbors algorithm

LDAP: Lightweight Directory Access Protocol

MLP: Multilayer perceptron

NB: Naive Bayes

N-IDS: Network based Intrusion Detection System

N-IPS: Network-Intrusion Prevention System

NN: Neural Network

NTP: Network Time Protocol

OSI: Open Systems Interconnection

OSPF: Open Shortest Path First

PASFAS: Problème de l'analyse simplifier de fichier d'audite de sécurité

PCA: Principal component analysis

PGP: Pretty Good Privacy

PIN: Personal Identification Number

PKI: Public Key Infrastructure

PPDM: Privacy Preserving Data Mining

Probe: Probing

PSK: Pre-shared Key

R2L: Remote to Local

RMI: Remote method invocation

RSA: algorithm of Ron Rivest, Adi Shamir, and Leonard Adleman

SNMP: Simple Network Management Protocol

SOM: Self-Organizing Map

SSH: Secure Shell

SSL: Secure Sockets Layer

SVM: Support Vector Machine

TCB: Trusted Computing Base

TCP/IP: Transmission Control Protocol/ Internet Protocol

TCP: Transmission Control Protocol

TKIP: Temporal Key Integrity Protocol

TLS: Transport Layer Security

U2R: User to Root

UDP: User Datagram Protocol

VN: Vrai négative

VP: Vrai positif

WEP: Wired Equivalent Privacy

WI-FI: WIreless Fidelity

WPA: Wi-Fi Protected Access

Annexe 2 La description du KDD'99

Depuis 1999 l'ensemble de données KDD'99 (KDD'99, 1999) est devenu la base de données la plus utilisée pour l'évaluation des systèmes de détection d'intrusion. Cet ensemble de données a été préparé par Stolfo et al. (2000), où ils ont utilisé des données extraites de l'ensemble de données d'évaluation des systèmes de détection d'intrusion le DARPA'98 (Lippmann et al, 2000).

Le DARPA'98 contient environ 4 gigabits d'enregistrements binaires compressés des données tcpdump collectées durant 7 semaines de surveillance de trafic réseau qui ont été transformées en environ 5 millions d'enregistrements de connexion, où chaque connexion est composée d'environ 100 octets. Les deux semaines des données de test représentent environ 2 millions d'enregistrements de connexion. L'ensemble de données d'apprentissage « KDD training data set » se compose d'environ 4,9 millions d'enregistrements de connexion dont chacun contient 41 caractéristiques ainsi qu'une étiquette qui indique le type de connexion normal ou une attaque (le type spécifique d'attaque). Les attaques simulées appartiennent à l'une des quatre catégories suivantes:

- **L'attaque de déni de service « Denial of Service Attack (DoS) » :** est une attaque dans laquelle l'attaquant essaye de rendre la mémoire d'un système ou la bande passante d'un réseau trop occupée ou trop chargée pour gérer les demandes légitimes, afin d'empêcher les utilisateurs légitimes d'accès à une machine.
- **L'attaque de passage d'un utilisateur à un super utilisateur « User to Root Attack (U2R) » :** est une attaque dans laquelle l'attaquant commence par un accès à un compte utilisateur normal sur le système (peut-être acquise par des mots de passe capturés, une attaque par dictionnaire… etc.) dans le but d'exploiter certaines vulnérabilités pour obtenir un accès Root sur le système.
- **L'attaque distance à local « Remote to Local Attack (R2L) » :** se produit quand un attaquant qui a la capacité d'envoyer des paquets vers une machine sur un réseau, mais qui n'a pas de compte sur cette machine. Il exploite certaines vulnérabilités afin d'obtenir un accès local en tant qu'utilisateur de cette machine.
- **L'attaque d'exploration « Probing Attack » :** elle vise de rassembler des informations sur un réseau d'ordinateurs dans le but de contourner les contrôles de sécurité.

Il faut noter que les données de test n'ont pas la même distribution de probabilité que les données d'apprentissage, et elles incluent des types d'attaques spécifiques qui n'existent pas dans les données d'apprentissage ce qui rend les données plus réalistes.

Les ensembles de données contiennent un nombre total de 38 attaques, 24 types d'attaques dans les données d'apprentissage, et 14 autres types d'attaques existent dans les données de test seulement. Le nom et la description détaillée des types d'attaques sont répertoriés dans (DARPA, 1999).

KDD'99 représente l'ensemble de données standard ouvert le plus utilisé pour l'évaluation des approches de détection d'intrusion. KDD'99 inclut 3 échantillons de données qui sont: toutes les données d'apprentissage, 10% des données d'apprentissage, les données de test. Le tableau suivant présente la distribution des enregistrements de chaque échantillon.

échantillon	**Nombre d'enregistrement**
Toutes les données d'apprentissage	4 898 431
10% des données d'apprentissage	494 021
Les données de test	311 029

Tableau 36 La distribution des enregistrements de chaque échantillon de données du KDD'99

Pareille que DARPA et comme le montre le tableau 37 le KDD'99 contient 41 propriétés qui peuvent être classées en trois groupes:

1) **Les caractéristiques de base:** cette catégorie englobe tous les attributs qui peuvent être extraits à partir d'une connexion TCP/IP. La plupart de ces caractéristiques conduisent à un retard implicite dans la détection.
2) **Les caractéristiques du trafic:** cette catégorie comprend des caractéristiques qui sont calculées par rapport à un intervalle de fenêtre et se divise en deux groupes:
    a) **Les caractéristiques du « Même hôte »:** examine uniquement les connexions durant les 2 dernières secondes qui ont le même hôte de destination que la connexion en cours, et calcule les statistiques relatives au protocole, service, etc.
    b) **Les caractéristiques des « Mêmes services»:** examine uniquement les connexions durant les 2 dernières secondes qui ont le même service que la connexion actuelle.

Les deux types de caractéristique du trafic mentionnés ci-dessus sont appelés en fonction du temps. Cependant, il y a plusieurs attaques de type exploration (Probe)

lentes qui scannent les hôtes (ou ports) avec un intervalle de temps beaucoup plus grand que 2 secondes, par exemple, une exploration chaque minute. En conséquence, ces attaques ne produisent pas des modèles (Paterns) d'intrusion avec un intervalle de temps de 2 secondes.

Pour résoudre ce problème, les caractéristiques du "même hôte" et "même service" sont recalculées, mais basées sur une fenêtre de connexion de 100 connexions plutôt qu'une fenêtre de temps de 2 secondes. Ces caractéristiques sont appelées caractéristiques de trafic basé sur les connexions.

3) **Caractéristiques de contenu:** contrairement à la plupart des attaques DoS et Probe, les attaques R2L et U2R n'ont pas un modèle d'intrusion séquentiel fréquents. Parce que les attaques DoS et Probe impliquent de nombreuses connexions à un hôte (s) dans un très court laps de temps, mais les attaques R2L et U2R sont intégrés dans les portions de données des paquets, et impliquent normalement une seule connexion. Pour détecter ces types d'attaques, nous avons besoin de certaines caractéristiques pour examiner les comportements suspects dans des parties de données, par exemple, le nombre de tentatives des connexions ayant échouées. Ces caractéristiques sont appelées fonctions de contenu.

#	**Nom de la propriété**	**Type de la propriété**	**Description de la propriété**
1	Duration	Continu	Longueur de la connexion (second)
2	Protocol_type	Discret	Type de protocole, e.g. tcp, udp, etc.
3	Service	Discret	Service réseau de destination, e.g., http, telnet, etc.
4	Flag	Discret	Statut normal ou erreur de la connexion
5	Src_bytes	Continu	Nombre d'octets de données de la source à la destination
6	Dst_bytes	Continu	Nombre d'octets de données de la destination à la source
7	Land	Discret	1 si une connexion est de / vers le même hôte / port; 0 sinon
8	Wrong_fragment	Continu	Nombre de fragments « erronées »

9	Urgent	Discret	Nombre de paquets urgents
10	Hot	Discret	Nombre d'indicateurs "hot"
11	Num_failed_logins	Discret	Nombre de tentatives de connexion échouées
12	Logged_in	Discret	1 si un succès de se connecter, 0 sinon
13	Num_compromised	Discret	Nombre de conditions compromises
14	Root_shell	Discret	1 si le root Shell est obtenu; 0 autrement
15	Su_attempted	Discret	1 si la commande " su root " a été tentée, sinon 0
16	Num_root	Discret	Nombre de " root " ont accédé
17	Num_file_creations	Discret	Nombre d'opérations de création de fichiers
18	Num_shells	Discret	Nombre d'invités du shell
19	Num_access_files	Discret	Nombre d'opérations sur les fichiers de contrôle d'accès
20	Num_outbound_cmds	Discret	Nombre de commandes sortantes dans une session FTP
21	Is_host_login	Discret	1 si la connexion appartient à la liste du ‘hot’ ; 0 sinon
22	Is_guest_login	Discret	1 si le login est un login "guest", sinon 0
23	Count	Discret	nombre de connexions vers la même machine que la connexion en cours dans les deux dernières secondes
24	Srv_count	Discret	Nombre de connexions pour le même service que la connexion en cours dans les deux dernières secondes
25	Serror_rate	Discret	% Des connexions qui ont des erreurs "SYN" (connexions de la même machine)
26	Srv_serror_rate	Discret	% Des connexions qui ont des erreurs "SYN" (connexions du même service)

27	Rerror_rate	Discret	% Des connexions qui ont des erreurs "REJ" (connexions de la même machine)
28	Srv_rerror_rate	Discret	% Des connexions qui ont des erreurs "REJ" (connexions du même service)
29	Same_srv_rate	Discret	% des connexions aux mêmes services
30	Diff_srv_rate	Discret	% de connexions aux différents services
31	Srv_diff_host_rate	Discret	% de connexions aux différentes machines
32	Dst_host_count	Discret	compteur pour la machine de destination
33	Dst_host_srv_count	Discret	Srv_count pour la destination host
34	Dst_host_same_srv_rate	Discret	Same_srv_rate pour la destination host
35	Dst_host_diff_srv_rate	Discret	Diff_srv_rate pour la destination host
36	Dst_host_same_src_port_rate	Discret	Same_src_port_rate pour la destination host
37	Dst host srv diff host rate	Discret	Diff_host_rate pour la destination host
38	Dst_host_serror_rate	Discret	Serror_rate pour la destination host
39	Dst_host_srv_serror_rate	Discret	Srv_serror_rate pour la destination host
40	Dst_host_rerror_rate	Discret	Rerror_rate pour la destination host
41	Dst_host_srv_rerror_rate	Discret	Srv_serror_rate pour la destination host

Tableau 37 Les propriétés du KDD'99

Comme le DARPA les enregistrements du KDD'99 sont correctement étiquetés soit comme un type spécifique d'attaque ou comme normal. Normal représenté toute connexion non intrusive (les comportements habituels d'un utilisateur) telle que la consultation d'une page web ou le téléchargement des applications et des fichiers...etc. Toutes les attaques du KDD'99 appartiennent à l'une des quatre classes d'attaques précédemment présentées dans la description du DARPA (DoS, Probing, R2L, U2R). Le KDD'99 contient 39 attaques où 17 attaques n'existent que dans les données de test. Le tableau suivant résume la classification des différentes attaques du KDD'99 en quatre catégories d'attaques.

Les catégories des attaques	Type d'attaque		
	Attaque existe dans les données d'apprentissage et de test	**Attaque existe que dans les données d'apprentissage**	**Attaque existe que dans les données de test**
DOS	back, land, neptune, pod, smurf, teardrop		apache2, mailbomb, processtable, udpstorm
Probe	ipsweep, nmap, portsweep, satan		mscan, saint
R2L	ftp_write, guess_passwd, imap, multihop, phf, warezmaster	spy, warezclient	named, sendmail, snmpgetattack, snmpguess, worm, xlock, xsnoop
U2R	buffer_overflow, loadmodule, perl, rootkit		httptunnel, ps, sqlattack, xterm

Tableau 38 La classification des attaques du KDD'99

Le tableau suivant présente la distribution des attaques et du comportement normal dans l'ensemble de données d'entrainement (le KDD'99 10%) et l'ensemble de données de test (le KDD'99 Test).

10% Training			**KDD Test**			
étiquette	**toute**	**distincte**	**étiquette**	**toute**	**distincte**	**Class**
			apache2	794	794	**DOS**
back	2203	968	back	1098	386	**DOS**
buffer_overflow	30	30	buffer_overflow	22	22	**U2R**
ftp_write	8	8	ftp_write	3	3	**R2L**
guess_passwd	53	53	guess_passwd	4367	1302	**R2L**
			httptunnel	158	145	**U2R**
imap	12	12	imap	1	1	**R2L**
ipsweep	1247	651	ipsweep	306	155	**Probe**
land	21	19	land	9	9	**DOS**
loadmodule	9	9	loadmodule	2	2	**U2R**
			mailbomb	5000	308	**DOS**
			mscan	1053	1049	**Probe**
multihop	7	7	multihop	18	18	**R2L**

			named	17	17	**R2L**
neptune	107201	51820	neptune	58001	20332	**DOS**
nmap	231	158	nmap	84	80	**Probe**
normal	97278	87832	normal	60593	47913	**Normal**
perl	3	3	perl	2	2	**U2R**
phf	4	4	phf	2	2	**R2L**
pod	264	206	pod	87	45	**DOS**
portsweep	1040	416	portsweep	354	174	**Probe**
			processtable	759	744	**DOS**
			ps	16	16	**U2R**
rootkit	10	10	rootkit	13	13	**U2R**
			saint	736	364	**Probe**
satan	1589	906	satan	1633	860	**Probe**
			sendmail	17	15	**R2L**
smurf	280790	641	smurf	164091	936	**DOS**
spy	2	2				**R2L**
			snmpgetattack	7741	179	**R2L**
			snmpguess	2406	359	**R2L**
			sqlattack	2	2	**U2R**
teardrop	979	918	teardrop	12	12	**DOS**
			udpstorm	2	2	**DOS**
warezclient	1020	893				**R2L**
warezmaster	20	20	warezmaster	1602	1002	**R2L**
			worm	2	2	**R2L**
			xlock	9	9	**R2L**
			xsnoop	4	4	**R2L**
			xterm	13	13	**U2R**
Total	**494021**	**145586**	**Total**	**311029**	**77291**	

Tableau 39 La distribution détaillée des attaques et du comportement normal dans l'échantillon 10% des données d'apprentissage et les données de test

Le tableau suivant montre la distribution des catégories des attaques et du comportement normal pour l'ensemble de données d'apprentissage (le KDD'99 10 %) et l'ensemble de données de test (le KDD'99 Test).

	Ensemble de données d'apprentissage le KDD'99 10 %		Ensemble de données de test le KDD'99 Test	
Normal	97278	87832	60593	47913
Dos	391458	54572	229853	23568
Probe	4107	2131	4166	2682
R2L	1126	999	16189	2913
U2R	52	52	228	215
All	494021	145586	311029	77291

Tableau 40 La distribution des catégories d'attaques et du comportement normal dans le KDD'99 10% et le KDD'99 Test

Annexe 3 La description du NSL-KDD

L'ensemble de données NSL-KDD a été proposé pour résoudre certains problèmes inhérents de l'ensemble de données KDD'99 qu'ils ont été mentionnés dans l'article de Tavallaee et al, (2009). Bien que cette nouvelle version de données souffre encore de certains problèmes évoqués par McHugh (McHugh, 2000) et ne peut pas être un représentant parfait des véritables réseaux existants, cet ensemble de données peut encore être appliqué en tant qu'ensemble de données de référence efficace pour aider les chercheurs à comparer les différentes méthodes de détection d'intrusion.

Le nombre d'enregistrements dans les données d'apprentissage et de test du NSL-KDD sont raisonnable. Cet avantage rend abordable d'exécuter les expérimentations sur l'ensemble complet sans la nécessité de choisir au hasard une petite partie. Par conséquent, les résultats de l'évaluation des différents travaux de recherche seront cohérents et comparables.

L'ensemble de données NSL-KDD présente les avantages suivants par rapport à l'ensemble original de données (KDD'99):

- Il n'inclut pas les enregistrements redondants dans les données d'apprentissage, de sorte que les classificateurs ne seront pas biaisés en faveur des enregistrements les plus fréquents.
- Il n'y a pas des enregistrements redondants dans les ensembles de données de test proposées, par conséquent, les performances des classificateurs ne seront pas biaisées par les méthodes qui ont un meilleur taux de détection pour les enregistrements fréquents.
- Le nombre d'enregistrements sélectionnés dans chaque groupe de niveau de difficulté est inversement proportionnel au pourcentage d'enregistrements dans l'ensemble original de données. En conséquence, les taux de classification des méthodes d'apprentissage automatique varient dans une plage plus large, ce qui lui rend plus efficace pour avoir une évaluation précise des différentes techniques d'apprentissage.
- Le nombre d'enregistrements des données d'apprentissage et de test est raisonnable, ce qui rend abordable d'exécuter les expérimentations sur l'intégralité des données sans la nécessité de choisir au hasard une petite partie. Par conséquent, les résultats de l'évaluation des différents travaux de recherche seront cohérents et comparables.

NSL-KDD contient 4 échantillons de données qui sont: KDDTrain+, KDDTrain+_20Percent, KDDTest+, KDDTest-21.

- KDDTrain+ : représente toutes les donnes d'apprentissage du NSL-KDD
- KDDTrain+_20Percent : représente 20% des donnes d'apprentissage du NSL-KDD
- KDDTest+ : représente toutes les donnes du test du NSL-KDD
- KDDTest-21 : représente toutes les donnes du test du NSL-KDD qui ne contient pas les enregistrements avec le niveau de difficulté de 21 sur 21 (Tavallaee et al, 2009).

Pareil que le DARPA et KDD'99, chaque enregistrement du NSL-KDD contient 41 propriétés. Ces enregistrements sont correctement étiquetés soit comme un type spécifique d'attaque ou comme un comportement normal. Toutes les attaques du NSL-KDD appartiennent à l'une des quatre catégories précédemment présentées (DoS, Probing, R2L, U2R).

Le tableau suivant montre la distribution de toutes les connexions réseau de KDDTrain + et KDDTest+.

Catégorie	Nombre d'enregistrement en KDDTrain+	Nombre d'enregistrement en KDDTest+
Normal	67343	9711
DOS	45927	7458
Probe	11656	2421
R2L	995	2754
U2R	52	200
All	125973	22544

Tableau 41 La distribution des connexions du KDDTrain+ et KDDTest

Nos productions scientifiques

(AHMIM et al, 2010) AHMIM, A., GHOUALMI, N., NOUDJOUD, K. (2010) Intrusion Detection by optimized GASSATA, International Symposium on Modeling and Implementation of Complex Systems MISC 2010, Constantine, Algeria, May 30-31, 2010.

(AHMIM et al, 2011) AHMIM, A., GHOUALMI, N., NOUDJOUD, K. (2010) Improved Off-Line Intrusion Detection Using a Genetic Algorithm and RMI, the 1st international conference on information systems and technologies ICIST 2011, Tébessa, Algeria, April, 2011.

(AHMIM and GHOUALMI-ZINE, 2013) AHMIM, A., GHOUALMI-Zine, N. (2013), A New Fast and High Performance Intrusion Detection System, International Journal of Security and Its Applications, vol. 7, No. 5, pp.67-80.

(AHMIM and Ghoualmi-Zine, 2014) AHMIM, A., GHOUALMI-Zine, N. (2014), A new adaptive intrusion detection system based on the intersection of two different classifiers, international journal of security and network, vol. 9, No 3, pp.125-132.

(AHMIM and Ghoualmi-Zine, 2015) AHMIM, A., GHOUALMI-Zine, N. (2015), A new hierarchical intrusion detection system based on a binary tree of classifiers, international journal of information management and computer security.