

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI

BOUMEDIENE

USTHB / ALGER

Faculté d'Electronique et Informatique



Mémoire

Présenté pour l'obtention du Diplôme de :

MAGISTER

Spécialité : Systèmes Intelligents et Ingénierie du Logiciel

Par

Mr. DEBBAB MOHAMED

SUJET

***La Conception et la Réalisation d'un
Système Intelligent de Détection de
Fausses alerte réseau***

Soutenu publiquement, le 22/12/2010 Devant le jury composé de :

Mr. Amar AISSANI	Professeur, à l'USTHB	Président
Mr. Ahmed Riadh BABA-ALI	M.C/B, à l'USTHB	Directeur de Mémoire
Mr. Hamid AZZOUNE	M.C/A, à l'USTHB	Examineur
Mme. Haned Faiza KHELLAF	M.C/A, à l'USTHB	Examinatrice

Remerciements

Le travail présenté dans ce mémoire a été effectué à l'U.E.R Informatique à l'Ecole Militaire Polytechnique (E.M.P), sous la direction du Mr A.R.Baba-Ali maître de conférence à l'USTHB à qui j'exprime ma profonde gratitude de m'avoir proposé ce sujet, pour l'attention portée afin d'élaborer ce mémoire, son soutien permanent et sa disponibilité.

Je tiens tout d'abord à exprimer ma reconnaissance envers les membres du jury pour l'intérêt qu'ils ont porté à ce mémoire en acceptant d'en être examinateurs. Je remercie Monsieur A. AISSANI, d'avoir accepté de présider le jury. Je remercie également Monsieur H. AZZOUNE et Madame F. KHELAF, d'avoir accepter la charge d'évaluer en qualité d'examineurs les travaux réalisés.

Nous n'oublierons pas également de remercier nos enseignants, particulièrement ceux du département d'informatique (U.S.T.H.B), pour leur contribution à notre formation.

J'adresse un remerciement particulier à Col KESSAL, qui grâce à sa relecture très attentive de mon rapport, m'a permis d'en améliorer sensiblement la qualité.

Mes remerciements vont également à l'ensemble du personnel de l'UER Informatique pour sa collaboration et son amabilité.

Enfin, je tiens à remercier tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail.

Dédicaces

A ma famille

et amis

Résumé

Le système de détection d'intrusion (*IDS*) est considéré comme une approche pour protéger les systèmes informatiques. Il est de plus en plus utilisé contre les attaques des réseaux d'entreprises. Cependant, il existe un problème difficile lié à l'utilisation des *IDSs*. Ils déclenchent généralement un grand nombre de messages d'alerte qui peut atteindre jusqu'à mille alertes ou plus par jour.

Un problème important considéré par les systèmes de détection d'intrusion aujourd'hui est celui des faux positifs, c'est-à-dire des alertes qui indiquent de manière erronée des problèmes de sécurité et exigent l'attention de l'analyste de détection d'intrusion. D'après [Julisch 03], il est estimé que plus de 99 % d'alertes enregistrées par les *IDSs* ne sont pas liées aux problèmes de sécurité. Dans ce contexte, nous avons présenté une classification globale des approches existantes afin de proposer une approche pour la détection de faux positifs basée sur la notion de *decision list*. Diverses simulations numériques sur la base de données *DARPA* 1998 montrent que l'approche proposée est performante.

Mots clés : Système de détection d'intrusion, *IDS*, faux positif, *DARPA* 1998.

Abstract

The intrusion detection system (*IDS*) is considered as an approach to protect computer systems. It is increasingly being used against attacks toward corporation networks. However, there appears a difficult problem with the use of *IDSs*, ie they generally trigger a large number of alerts which can reach up to a thousand or more alerts per day.

One of the most important issues considered by the intrusion detection systems today are the false positives, that are alerts that indicate incorrectly safety problems and require the attention of the intrusion detection analyst. According [Julisch 01] [Julisch 02] [Julisch 03], it was estimated that over 99% of alerts recorded by the *IDSs* are not related to security issues. In this context, we have a comprehensive classification of existing approaches in order to propose an approach for the detection of false positives based on the notion of decision list. Various numerical simulations based on data *DARPA* 1998 show that the proposed approach is efficient.

Keywords: Intrusion Detection System, *IDS*, false positive, *DARPA* 1998.

Sommaire

INTRODUCTION GENERALE.....	1
I. LA SECURITE INFORMATIQUE.....	4
I.1 INTRODUCTION	4
I.2 DEFINITION DE LA SECURITE INFORMATIQUE	5
I.3 MENACES, RISQUES ET POLITIQUE DE SECURITE	5
I.3.1 <i>Risque</i>	5
I.3.2 <i>Menaces</i>	6
I.3.3 <i>Politique de sécurité</i>	7
I.4 LES CRITERES DE SECURITE	8
I.4.1 <i>Confidentialité des données</i>	8
I.4.2 <i>Authentification</i>	9
I.4.3 <i>Intégrité des données</i>	9
I.4.4 <i>Disponibilité</i>	9
I.4.5 <i>Non répudiation</i>	9
I.5 MECANISMES DE SECURITE	9
I.6 MISE EN PLACE D'UNE POLITIQUE DE SECURITE	11
I.7 QUELQUES ATTAQUES CONNUES.....	12
I.7.1 <i>Le déni de service (DoS, Denial of Service)</i>	12
I.7.2 <i>L'inondation de requêtes d'ouverture (SYN Flooding)</i>	12
I.7.3 <i>Dissimulation d'adresse IP (IP Spoofing)</i>	12
I.7.4 <i>Autres attaques</i>	13
I.8 CONCLUSION	13
II. LES SYSTEMES DE DETECTION D'INTRUSION	15
II.1 INTRODUCTION.....	15
II.2 DEFINITIONS	15
II.3 MODELE DE PROCESSUS DE LA DETECTION D'INTRUSION.....	16
II.3.1 <i>La source d'informations (sonde)</i>	16
II.3.2 <i>L'analyse</i>	17
II.3.3 <i>La réponse</i>	17
II.4 UN SYSTEME DE DETECTION D'INTRUSION DANS LE RESEAU.....	17
II.5 OBJECTIFS	18
II.6 TECHNIQUES DE MISE EN ŒUVRE.....	19
II.6.1 <i>Par scénarios</i>	19
II.6.2 <i>L'approche comportementale</i>	20
II.7 L'ÉVALUATION DES IDS	20
II.7.1 <i>Faux négatif</i>	20
II.7.2 <i>Faux positif</i>	21
II.8 LES LIMITATIONS DES SYSTEMES DE DETECTION D'INTRUSION	21
II.9 CLASSIFICATION DES SYSTEMES DE DETECTION D'INTRUSION	21
II.10 LA METHODE D'ANALYSE	22
II.10.1 <i>La détection par scénario</i>	22
II.10.2 <i>La détection d'anomalie (comportementale)</i>	23
II.11 LE COMPORTEMENT DE LA DETECTION (LA REPONSE).....	24
II.11.1 <i>Les réponses actives</i>	25
II.11.2 <i>Les réponses passives</i>	26
II.12 L'EMPLACEMENT DES SOURCES D'AUDITS.....	27
II.12.1 <i>NIDS (Network-Based IDS)</i>	27

II.12.2	<i>HIDS (Host-Based IDS)</i>	27
II.12.3	<i>IDS d'application</i>	28
II.12.4	<i>IDS hybrides</i>	28
II.13	LA FREQUENCE D'UTILISATION (LA SYNCHRONISATION)	28
II.13.1	<i>En temps différé (périodique)</i>	28
II.13.2	<i>En temps réel (continu)</i>	28
II.14	L'ARCHITECTURE	29
II.14.1	<i>Host Target Colocation (Cohabitation de la cible et de l'hôte)</i>	29
II.14.2	<i>Host Target Separation (Séparation entre la cible et l'hôte)</i>	29
II.15	LA STRATEGIE DE CONTROLE	29
II.15.1	<i>Centralisée</i>	29
II.15.2	<i>Partiellement distribuée</i>	30
II.15.3	<i>Entièrement distribuée</i>	30
II.16	CONCLUSION	30
III.	LA DETECTION DE FAUX POSITIFS	33
III.1	INTRODUCTION	33
III.2	LES RAISONS DE FAUX POSITIFS	33
III.3	LA DETECTION DE FAUX POSITIFS	34
III.3.1	<i>Des méthodes basées sur la classification</i>	34
III.3.2	<i>Des méthodes basées sur les causes d'origine</i>	38
III.3.3	<i>Des méthodes basées sur l'hypothèse "Fréquence d'alertes"</i>	43
III.4	ÉTUDE COMPARATIVE ENTRE LES TROIS METHODES	44
III.5	LES AVANTAGES ET LES INCONVENIENTS DE CHAQUE METHODE	45
III.6	CONCLUSION	45
IV.	LES BENCHMARKS	47
IV.1	INTRODUCTION	47
IV.2	LES BENCHMARKS	47
IV.2.1	<i>L'évaluation par test</i>	48
IV.2.2	<i>L'évaluation analytique</i>	49
IV.3	L'ÉVALUATION DU DARPA 98	49
IV.4	LES ELEMENTS PRINCIPAUX DE DARPA 1998	52
IV.5	L'ENSEMBLE DES DONNEES UTILISEES	53
IV.5.1	<i>Le système de détection d'intrusion utilisé</i>	54
IV.5.2	<i>La représentation des alertes</i>	55
IV.5.3	<i>L'étiquetage des alertes</i>	56
IV.6	CONCLUSION	57
V.	L'APPROCHE PROPOSEE POUR LA DETECTION DE FAUX POSITIF	59
V.1	INTRODUCTION	59
V.2	DEFINITION	59
V.2.1	<i>Concepts fondamentaux</i>	60
V.2.2	<i>La présentation des alertes</i>	61
V.3	L'ARCHITECTURE DE L'APPROCHE PROPOSEE	62
V.3.1	<i>La technique d'apprentissage utilisée</i>	63
V.3.2	<i>Le codage de règle</i>	64
V.3.3	<i>L'algorithme de l'approche</i>	64
V.4	CONCLUSION	66

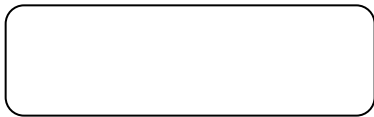
VI.	LES TESTS.....	68
VI.1	INTRODUCTION.....	68
VI.2	OUTILS UTILISES.....	68
VI.2.1	<i>Le langage de programmation utilisé</i>	<i>68</i>
VI.2.2	<i>La bibliothèque externe.....</i>	<i>68</i>
VI.3	LES BENCHMARKS DARPA 98.....	69
VI.3.1	<i>Les données d'apprentissage</i>	<i>69</i>
VI.3.2	<i>Le choix du classifieur PART.....</i>	<i>70</i>
VI.3.3	<i>Les mesures utilisées pour évaluer la qualité de l'approche</i>	<i>71</i>
VI.4	RESULTATS DE L'EXPERIMENTATION	71
VI.5	CONCLUSION.....	75
	CONCLUSION GENERALE ET PERSPECTIVES	77
	RÉFÉRENCES BIBLIOGRAPHIQUES	80

Liste des figures

Figure I.1. Les principaux services de la sécurité informatique [Florin et al, 99].....	8
Figure II.1. Modèle simplifié d'un système de détection d'intrusions [Papini 05]	17
Figure II.2. L'emplacement d'un IDS [Nassih 06].....	18
Figure II.3. Taxonomie des systèmes de détection d'intrusion [Debar et al, 00].....	22
Figure III.1. Machine learning (ALAC) [Pietraszek et al, 05]	35
Figure III.2. Architecture ALAC en mode recommander [Pietraszek 06]	36
Figure III.3. Pseudo-code de l'algorithme (ALAC) en Mode recommander [Pietraszek 06]	37
Figure III.4. Architecture ALAC en mode agent [Pietraszek 06]	37
Figure III.5. Pseudo-code de l'algorithme (ALAC) en Mode agent [Pietraszek 06]	38
Figure III.6. Diagramme entité-association utilisé par CLARAty [Pietraszek 06]	39
Figure III.7. Hiérarchie pour les cibles des attaques [Debar et al, 04]	40
Figure III.8. Data-mining (CLARAty) [Pietraszek et al, 05]	42
Figure III.9. Pseudo-code de l'algorithme CLARAty [Pietraszek 06]	43
Figure IV.1. Le processus d'évaluation d'IDS [Richard et al, 98].....	50
Figure IV.2. La capture des trames sur Ethereal	53
Figure IV.3. Une règle de Snort	55
Figure IV.4. Un message d'alerte généré par Snort	55
Figure IV.5. Représentation d'une alerte en format CSV	57
Figure V.1. Illustration du processus d'apprentissage.....	62
Figure V.2. Illustration du processus de classification.....	62
Figure V.3. Le schéma du système.....	63
Figure V.4. Exemple de codage de règles	64
Figure V.5. L'algorithme de l'approche	65
Figure VI.1. Les règles générées par la base de données utilisée.....	71
Figure VI.2. Variation de précision, FP et FN suivant le nombre d'alerte	73
Figure VI.3. Comparaison entre le taux de FP original et le taux de FP obtenu	73
Figure VI.4. Variation de précision, FP et FN suivant le nombre d'alertes.....	74
Figure VI.5. Comparaison entre le taux de FP original et le taux de FP obtenu	75

Liste des tableaux

Tableau III.1. La généralisation d'un fichier log d'IDS [Pietraszek et al, 05]	41
Tableau III.2. Étude comparative entre les trois approches	44
Tableau III.3. Avantages et inconvénients de chaque méthode	45
Tableau IV.1. Les attaques disponibles aux données DARPA 98 [Kendall 99]	52
Tableau V.1. La présentation d'un ensemble d'alertes	61
Tableau VI.1. Résumé sur la base de données utilisée.....	70
Tableau VI.2. Evaluer la précision pour chaque jour	70
Tableau VI.3. Résultats obtenus par un ensemble alertes de la base de tests jeudi.....	72
Tableau VI.4. Résultats obtenus par toutes les alertes de la base de tests jeudi.....	74



INTRODUCTION GÉNÉRALE

Introduction générale

La sécurité des systèmes d'information et des réseaux est devenue une préoccupation majeure des entreprises. Beaucoup de méthodes ont été développées pour sécuriser les infrastructures réseaux et la communication sur Internet, parmi lesquelles l'utilisation des firewalls, le cryptage et les réseaux privés virtuels (VPN). La détection d'intrusion est un élément supplémentaire, en plus de ces techniques. On peut résumer en disant qu'un système de sécurité se compose d'outils multiples, incluant :

- Un firewall utilisé pour bloquer le trafic entrant et sortant non désiré ;
- Des outils d'évaluation de vulnérabilité, utilisés pour repérer d'éventuelles failles de sécurité existant sur le réseau ;
- Un système de détection d'intrusion (*IDS*) utilisé pour savoir si un intrus est entré ou essaye d'entrer dans le réseau.

Les systèmes de détection d'intrusion (*IDS*) ont un rôle crucial dans le schéma de sécurité des systèmes informatiques. Pourtant, il reste des désavantages dans les *IDSs* courants. Les *IDSs* produisent souvent un gros volume d'alertes dues à des processus d'attaques automatisés et fréquents. Cela rend l'analyse embarrassante et compliquée. Les *IDSs* peuvent également générer de nombreux faux positifs. Il est donc intéressant de développer des techniques pour la détection de faux positifs.

Malgré la réputation de ces outils, le déploiement croissant et multiple de ces outils dans des environnements de plus en plus complexes a mis en évidence plusieurs problèmes qui sont actuellement fortement posés. Le problème majeur de ces outils est le faux positif dû à plusieurs raisons telles que : des signatures d'attaques non précises qui déclarent malicieux un comportement normal, ou des défaillances dans l'implémentation de ces outils qui ne permettent pas de détecter certaines attaques. Donc la problématique des systèmes de détection d'intrusion actuels se résume dans le taux de faux positif très élevé qui peut atteindre jusqu'à 99%.

L'objectif de ce travail, dans un premier temps, est l'étude des approches existantes qui traitent le problème de faux positif. Dans un second temps, c'est la proposition d'une approche pour la détection de faux positif basée sur la notion de *decision list*. L'approche proposée est basée sur deux phases, la première consiste à utiliser un algorithme d'apprentissage général *PART* intégré dans l'outil *weka* pour extraire des règles généralisées, la deuxième est un processus de classification basée sur la notion de *decision list*. Ce processus utilise les règles extraites dans la première phase. Finalement, nous validons l'approche proposée via des données de tests *DARPA* 1998. Diverses simulations numériques sur la base de données *DARPA* 1998 montrent que l'approche proposée est performante.

Organisation du mémoire

Le présent mémoire est organisé comme suit :

Nous avons commencé par une introduction générale dans laquelle nous avons défini la problématique et notre contribution à la solution.

Le premier chapitre représente une introduction générale au domaine de la sécurité informatique d'une façon générale.

Le deuxième chapitre constitue une brève description des systèmes de détection d'intrusion. Dans ce chapitre nous allons définir le processus de la détection d'intrusion ainsi que les performances d'un *IDS*.

Dans le troisième chapitre, nous discutons : les différentes méthodes et techniques utilisées dans la détection de faux positif, ainsi qu'une étude comparative entre ces approches afin de présenter les avantages et les inconvénients de chacune d'elles.

Dans le quatrième chapitre, nous avons présenté les benchmarks utilisés dans le domaine de la détection de faux positif, dans ce contexte nous présentons également en détail les benchmarks *DARPA* 1998.

Dans le cinquième chapitre, nous détaillons l'architecture globale de l'approche proposée. Ce chapitre est constitué d'une description de *decision list*, et du codage de règle afin de présenter l'algorithme de l'approche.

Le dernier chapitre contient les tests qui ont été menés. Ce système a été validé par une expérimentation sur les données de test *DARPA* 1998.

Enfin, cette thèse se termine par une conclusion générale et des perspectives qui pourront enrichir ce travail.

Chapitre 1

LA SÉCURITÉ INFORMATIQUE

Chapitre 1

La sécurité informatique

I.1 Introduction

Dans un contexte de connectivité croissante des réseaux informatiques et de développement de la technologie des réseaux, la sécurisation des systèmes d'information est devenue un enjeu majeur.

Avec l'avènement de l'ère Internet à la fin des années 90, le développement des échanges électroniques et l'évolution des interconnexions font que les risques en matière de sécurité informatique sont de plus en plus avérés et nombreux. Aussi, la sécurité des réseaux prend son essor avec le besoin de sécurité des utilisateurs, face aux hackers.

L'évolution de la technologie des systèmes informatiques a permis l'utilisation des systèmes automatisés pour accélérer les traitements, de partager de plus en plus d'information, pour gagner en économie et en efficacité. Ceci implique la nécessité de canaux de distribution de l'information performants grâce à l'usage d'Internet. En plus, Internet est un moyen de communication de très bon marché qui est présent partout, faisant de notre planète un village. Outre ses avantages indiscutables, il présente d'énormes contraintes en matière de sécurité, du fait qu'aucune garantie ou certitude n'est assurée quant aux informations qui y transitent. De plus, le concept de personnel nomade permettant l'utilisation d'ordinateurs portables et *PDA* (Personnel Data Assistant) qui accèdent en temps réel aux informations de part le monde complique d'avantage sa sécurisation.

De plus, le développement du e-commerce, courrier électronique et autres outils de téléchargement rend le système d'information de plus en plus accessible depuis des machines de moins en moins contrôlées, d'où le problème qui se pose de comment rendre les systèmes informatiques fiables ou comment les sécuriser?

I.2 Définition de la sécurité informatique

La sécurité est actuellement devenue une composante primordiale dans tout système d'information. Cependant les entreprises sont peu, ou pas, protégées contre tous les types d'attaques informatiques. De plus leurs réseaux sont exposés aux différentes menaces (externes ou internes).

La sécurité informatique est l'ensemble des moyens matériels, logiciels et humains mis en œuvre pour minimiser la vulnérabilité d'un système informatique contre des menaces. Elle vise à réduire le risque des menaces qui peuvent provenir de l'intérieur ou de l'extérieur de l'entreprise [Florin et al, 99].

La sécurité informatique est l'ensemble des moyens mis en œuvre pour minimiser la vulnérabilité d'un système informatique contre des menaces [Baudru 09].

I.3 Menaces, risques et politique de sécurité

Risques et menaces sont deux concepts fondamentaux pour la compréhension des techniques utilisées dans le domaine de la sécurité. Le risque est une fonction de paramètres qu'on peut maîtriser à la différence de la menace qui est liée à des actions ou des opérations. Dans un grand réseau, la sécurité concerne non seulement les éléments physiques (câbles, modems, routeurs, commutateurs...) mais aussi les éléments logiques, voire volatiles, que représentent les données qui circulent. Le responsable de la sécurité doit analyser l'importance des risques encourus, les menaces potentielles et définir un plan général de protection appelé politique de sécurité.

I.3.1 Risque

Les risques se mesurent en fonction de deux critères principaux: la vulnérabilité et la sensibilité. La vulnérabilité désigne le degré d'exposition à des dangers. Un des points de vulnérabilité d'un réseau est un point facile à approcher. Un élément de ce réseau peut être très vulnérable tout en présentant un niveau de sensibilité très faible: le poste de travail de l'administrateur du réseau, par exemple, dans la mesure où celui-ci peut se connecter au système d'administration en tout point du réseau.

La sensibilité désigne le caractère stratégique d'un composant du réseau. Celui-ci peut être très sensible, vu son caractère stratégique mais presque invulnérable, grâce à toutes les mesures de protection qui ont été prises pour le prémunir contre la plupart des risques.

Exemples: le câble constituant le média d'un réseau local lorsqu'il passe dans des espaces de service protégés, l'armoire de sauvegarde des logiciels de tous les commutateurs du réseau...

On peut classer les risques en deux catégories: structurels, ils sont liés à l'organisation et la démarche d'une entreprise ; accidentels, ils sont indépendants de l'entreprise.

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

I.3.1.1 Acceptables

Ils n'induisent aucune conséquence grave pour les entités utilisatrices du réseau. Ils sont facilement rattrapables: pannes électriques de quelques minutes, perte d'une liaison...

I.3.1.2 Courants

Ce sont ceux qui ne portent pas un préjudice grave. Ils se traduisent, par exemple, par une congestion d'une partie du réseau. La mauvaise configuration d'un équipement peut causer la répétition des messages émis, un opérateur peut détruire involontairement un fichier de configuration...

I.3.1.3 Majeurs

Ils sont liés à des facteurs rares. Ils causent des préjudices ou des dégâts importants, mais ils peuvent encore être corrigés, exemple d'un incendie ayant ravagé le centre de calcul d'une entreprise.

I.3.1.4 Inacceptables

Ils sont, en général, fatals pour l'entreprise. Ils peuvent entraîner son dépôt de bilan. Exemple: la destruction du centre informatique et de l'ensemble des sauvegardes des programmes et données.

I.3.2 Menaces

On peut également classer les menaces en deux catégories selon qu'elles ne changent rien (menaces passives) ou qu'elles perturbent effectivement le réseau (menaces actives).

I.3.2.1 Menaces passives

Les menaces passives consistent essentiellement à copier ou à écouter l'information sur le réseau, elles nuisent à la confidentialité des données. Dans ce cas, celui qui prélève une copie n'altère pas l'information elle-même. Il en résulte des difficultés à détecter ce type de malveillance, car elles ne modifient pas l'état du réseau. La méthode de prélèvement varie suivant le type de réseau. Sur les réseaux câblés, on peut imaginer un branchement en parallèle grâce à des appareils de type analyseurs de protocole ou une induction (rayonnement électromagnétique). Sur les faisceaux hertziens, des antennes captent les lobes secondaires des faisceaux; dans les transmissions par satellites, des antennes avec systèmes de poursuite.

I.3.2.2 Menaces actives

Les menaces actives nuisent à l'intégrité des données. Elles se traduisent par différents types d'attaques. On distingue le brouillage, le déguisement (modification des données au cours de leur transmission, modification de l'identité de l'émetteur ou du destinataire), l'interposition (création malveillante de messages en émission ou en réception).

I.3.3 Politique de sécurité

La définition d'une politique de sécurité nécessite d'abord l'analyse des informations qui circulent ou qui sont stockées (analyse de leur importance pour l'entreprise, analyse du coût que représenterait leur perte) et celles des menaces que l'on peut objectivement envisager [Briffaut 08].

Une politique de sécurité est un ensemble de règles qui fixent les actions autorisées et interdites dans le domaine de la sécurité informatique. Pour atteindre un bon niveau de sécurité, il faut l'aborder à travers plusieurs approches dont [Florin et al, 99]:

- *L'approche préventive coercitive*: Obliger les usagers à utiliser la sécurité.
- *L'approche préventive analytique*: Identifier les menaces.
- *L'approche curative*: Enregistrement de tout ou d'une partie des actions accomplies sur le système informatique pour pouvoir l'analyser.

Les priorités de l'entreprise et sa stratégie influent sur le choix des procédures internes que devront respecter tous les utilisateurs. Il faut définir les mécanismes de protection à mettre en œuvre (les outils antivirus, les pare-feu, les patches ou programmes de correction

des systèmes et des applications utilisés) puis tous les outils de surveillance (depuis l'audit jusqu'au journal historique et la détection des intrusions).

I.4 Les critères de sécurité

Dans l'entreprise, la sécurité informatique doit remplir certains objectifs, qui sont un ensemble de services, ils représentent les fondements et les pièces maîtresses de la sécurité sur les SI. La Figure I.1 présente les différents services assurés par la sécurité informatique.

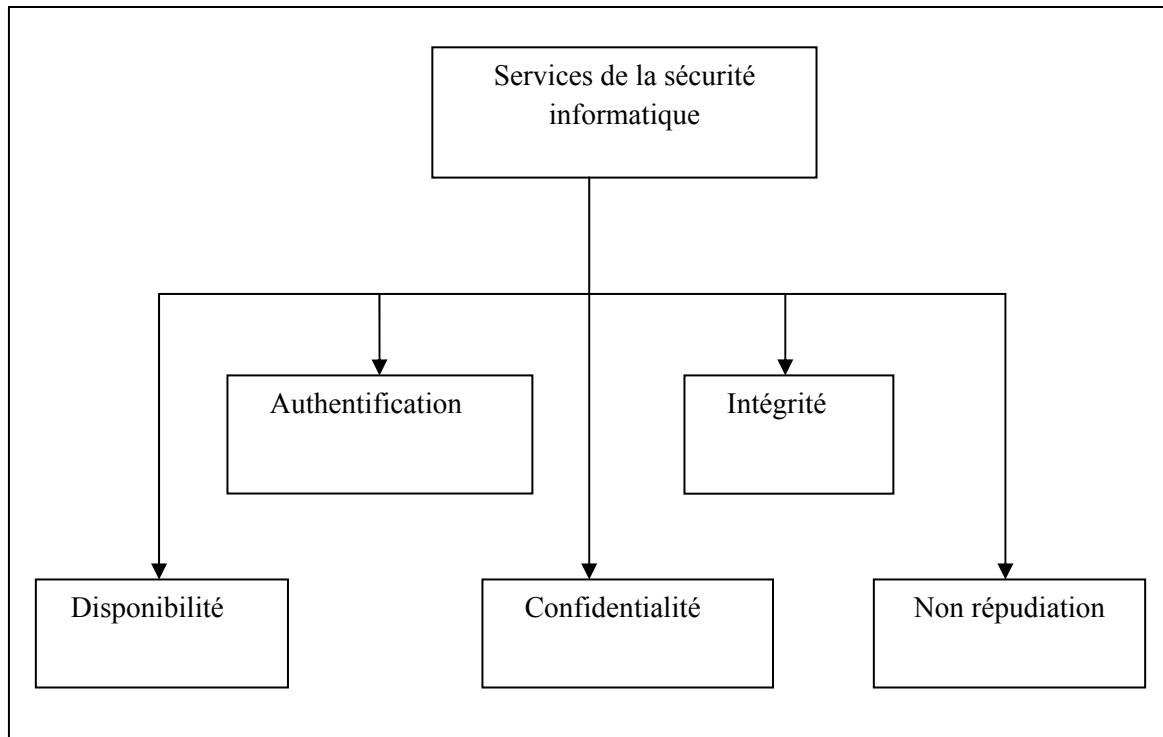


Figure I.1. Les principaux services de la sécurité informatique [Florin et al, 99]

La teneur des critères de sécurité à envisager peut être décrite de la façon suivante [Laurent et al, 09] [SOUBRIER 98] [Leopld et al, 99]:

I.4.1 Confidentialité des données

La confidentialité des données représente le fait que les données informatiques ne sont accessibles que par les personnes autorisées. L'objectif de ce service est d'empêcher les données d'être compréhensibles par une entité tierce non autorisée, le plus souvent en état de fraude passive, c'est-à-dire en écoute de l'information sur le réseau.

I.4.2 Authentification

Le critère d'authentification assure que le message provient de l'endroit d'où il prétend venir. Dans le cas d'un échange bidirectionnel, deux aspects sont présents. Il faut s'assurer que les deux entités sont bien celles qu'elles affirment être. De plus, le service d'authentification doit montrer que la connexion ne peut être brouillée par une troisième entité essayant de se faire passer pour un des deux correspondants (l'homme au milieu).

I.4.3 Intégrité des données

L'intégrité des données signifie que l'information ne peut être modifiée que par les personnes autorisées ou seulement par les moyens autorisés. L'intégrité reste un domaine très large couvrant à la fois les modifications, les moyens de modification mais également l'après modification et donc la consistance. Contre les fraudes actives (brouillage, modification des données ou de l'identité, déguisement en émission ou en réception), ce service détecte les altérations partielles ou intégrales des données entre émetteur et récepteur.

I.4.4 Disponibilité

La disponibilité se reflète dans l'information et dans les services. Ces services permettent d'assurer qu'un objet est accessible et utilisable sur demande par une entité autorisée.

I.4.5 Non répudiation

Un mécanisme de non répudiation permet d'empêcher une personne de nier le fait qu'elle a effectué une opération. En fait, c'est surtout le récepteur qui est visé, il ne peut pas « répudier » son message, dire qu'il ne l'a pas reçu.

I.5 Mécanismes de sécurité

La prolifération des formes de malveillance informatique s'accomplit parallèlement à la convergence de leurs méthodes: néanmoins l'utilisateur n'est pas sans défense contre les attaques de plus en plus nombreuses et de plus en plus puissantes, il existe des armes défensives. Nous examinerons ici la plus nécessaire [Laurent et al, 09]:

- **Chiffrement:** algorithme généralement basé sur des clefs et transformant les données. Sa sécurité est dépendante du niveau de sécurité des clefs.
- **Signature:** données ajoutées pour vérifier l'intégrité ou l'origine des données.

- **Bourrage de trafic :** données ajoutées pour assurer la confidentialité, notamment au niveau du volume du trafic.
- **Notarisation:** utilisation d'un tiers de confiance pour assurer certains services de sécurité.
- **Contrôle d'accès:** vérifie les droits d'accès d'un acteur aux données. N'empêche pas l'exploitation d'une vulnérabilité.
- **Antivirus:** logiciel censé protéger ordinateur contre les logiciels (ou fichiers potentiellement exécutables) néfastes. Ne protège pas contre un intrus qui emploie un logiciel légitime, ou contre un utilisateur légitime qui accède à une ressource alors qu'il n'est pas autorisé à le faire.
- **Le pare-feu:** un élément (logiciel ou matériel) du réseau informatique contrôlant les communications qui le traversent. Il a pour fonction de faire respecter la politique de sécurité du réseau, celle-ci définissant quels sont les communications autorisés ou interdites. N'empêche pas un attaquant d'utiliser une connexion autorisée pour attaquer le système. Ne protège pas contre une attaque venant du réseau intérieur (qui ne le traverse pas).
- **Détection d'intrusion:** repère les activités anormales ou suspectes sur le réseau surveillé. Ne détecte pas les accès incorrects mais autorisés par un utilisateur légitime. Mauvaise détection: taux de faux positifs, faux négatifs.
- **Journalisation ("*logs*"):** Enregistrement des activités de chaque acteur. Permet de constater que des attaques ont eu lieu, de les analyser et potentiellement de faire en sorte qu'elles ne se reproduisent pas.
- **Analyse de la vulnérabilité ("*security audit*"):** identification des points de vulnérabilité du système. Ne détecte pas les attaques ayant déjà eu lieu, ou lorsqu'elles auront lieu.
- **Contrôle du routage:** sécurisation des chemins (liens et équipements d'interconnexion).
- **Contrôle d'accès aux communications:** le moyen de communication n'est utilisé que par des acteurs autorisés. Par *VPN* ou tunnels.
- **Horodatage:** marquage sécurisé des instants significatifs.
- **Certification:** preuve d'un fait, d'un droit accordé.

- **Distribution de clefs:** distribution sécurisée des clefs entre les entités concernées.
- **Authentification :** Authentifier un acteur, l'authentification est nécessaire au bon fonctionnement des autres mécanismes.
- **La protection physique :** peut fournir une protection totale, mais qui peut être excessive. Par exemple isoler complètement son système est une solution qui peut être trop radicale.

I.6 Mise en place d'une politique de sécurité

La sécurité des systèmes informatiques se cantonne généralement à garantir les droits d'accès aux données et ressources d'un système en mettant en place des mécanismes d'authentification et de contrôle permettant d'assurer que les utilisateurs des dites ressources possèdent uniquement les droits qui leur ont été octroyés.

Les mécanismes de sécurité mis en place peuvent néanmoins provoquer une gêne au niveau des utilisateurs et les consignes et règles deviennent de plus en plus compliquées au fur et à mesure que le réseau s'étend. Ainsi, la sécurité informatique doit être étudiée de telle manière à ne pas empêcher les utilisateurs de développer les usages qui leur sont nécessaires, et de faire en sorte qu'ils puissent utiliser le système d'information en toute confiance.

C'est la raison pour laquelle il est nécessaire de définir dans un premier temps une politique de sécurité, dont la mise en œuvre se fait selon les quatre étapes suivantes:

- Identifier les besoins en terme de sécurité, les risques informatiques pesant sur l'entreprise et leurs éventuelles conséquences;
- Elaborer des règles et des procédures à mettre en œuvre dans les différents services de l'organisation pour les risques identifiés;
- Surveiller et détecter les vulnérabilités du système d'information et se tenir informé des failles sur les applications et matériels utilisés;
- Définir les actions à entreprendre et les personnes à contacter en cas de détection d'une menace;

La politique de sécurité est donc l'ensemble des orientations suivies par une organisation (à prendre au sens large) en termes de sécurité. A ce titre, elle se doit d'être élaborée au niveau de la direction de l'organisation concernée, car elle concerne tous les utilisateurs du système [ISO 05] [Marcus 09].

I.7 Quelques attaques connues

Les utilisateurs aux intentions malveillantes ont déployé de nombreuses attaques sur des sites commerciaux ou des sites de grandes sociétés et organismes. Les principales sont le déni de service (*DoS, Denial of Service*), l'inondation de requêtes d'ouverture TCP (*SYN flooding*), la dissimulation ou l'usurpation d'adresses IP (*IP Spoofing*) [Berthomier 05].

I.7.1 Le déni de service (*DoS, Denial of Service*)

Une attaque en déni de service consiste à bloquer une machine cible en lui envoyant des requêtes inutiles. Cela l'empêche de rendre le service pour lequel elle est installée. L'attaque la plus simple est l'inondation par des ping (messages ICMP *Echo Request*) ou des messages ICMP avec beaucoup de données forçant les différents intermédiaires à traiter la fragmentation. La machine cible passe son temps à répondre aux sollicitations reçues et n'a plus de disponibilité pour son propre service.

I.7.2 L'inondation de requêtes d'ouverture (*SYN Flooding*)

Une demande d'ouverture de connexion TCP (segment avec drapeau SYN mis à 1) provoque une réponse avec les drapeaux SYN et ACK mis à 1 puis une attente du troisième segment avec seulement le drapeau ACK mis à 1. L'attaque par inondation de requêtes d'ouverture consiste à envoyer à une machine cible un grand nombre de segments avec drapeaux SYN mais sans jamais transmettre le troisième segment. La machine cible réserve vainement des ressources à chaque requête d'ouverture et passe son temps à gérer les temporisateurs d'attente du troisième segment qui confirme l'ouverture.

I.7.3 Dissimulation d'adresse IP (*IP Spoofing*)

Le datagramme IP transporte l'adresse IP de l'émetteur et, en l'absence d'un mécanisme d'authentification de l'adresse, il est impossible de vérifier qui a émis avec cette adresse. Un pirate veut attaquer un réseau dont il connaît l'adresse IP: il usurpe l'une de ces adresses et l'utilise comme adresse source. Il y a toutes les chances pour que son datagramme soit considéré comme un datagramme normal du réseau... sauf s'il se présente, venant d'Internet, à la porte d'entrée du réseau et que l'administrateur a prévu qu'un message avec une adresse IP d'émetteur interne ne puisse pas provenir de l'extérieur.

I.7.4 Autres attaques

Les pirates ont toujours beaucoup d'imagination: utiliser un port (ouvert) proposé pour un protocole donné avec un autre protocole, ce qui donne des possibilités de manipulations sur la machine cible; voler des sessions (*hijacking*) TCP ouvertes de l'intérieur, profiter des failles de sécurité sur une machine pour l'utiliser ensuite comme source et profiter des droits d'accès de celle-ci. Le rejeu est également une attaque possible, il consiste à réinjecter dans le réseau des messages corrects (chiffrés, signés...) qui ont déjà été transmis. Nous pourrions ranger dans cette catégorie les chevaux de Troie ou les vers...

I.8 Conclusion

Au cours de ce chapitre, nous avons abordé les différents aspects liés à la sécurité dans les réseaux. L'ISO a défini le vocabulaire des services et des mécanismes de sécurité: l'authentification, l'intégrité, la non-répudiation, etc. Les solutions retenues actuellement pour faire face aux différents risques et menaces foisonnent. Ainsi nous avons présenté les différents mécanismes pour sécuriser les systèmes informatiques (pare-feu, antivirus, etc.).

La panoplie des protections est très vaste, elle s'accroît avec la créativité des attaquants; par ailleurs, la technologie évolue et leur fournit des capacités de traitement toujours plus puissantes.

La sécurité du système d'information et des réseaux nécessite donc des équipes compétentes et rigoureuses et une bonne information des utilisateurs parce qu'avec internet nous sommes entrés dans une ère où le régime de menaces est de haute intensité, et les menaces sont permanentes.

Enfin, notons que les outils de sécurité informatique n'assurent pas la protection à cent pourcent. Cependant, ils présentent des mécanismes efficaces, dans le chapitre suivant, nous étudions un type de mécanisme de sécurité, l'IDS ou système de détection d'intrusions.

Chapitre 2

LES SYSTÈMES DE DÉTECTION D'INTRUSION

Chapitre 2

Les systèmes de détection d'intrusion

II.1 Introduction

Dans ce chapitre, nous présentons les différents concepts fondamentaux sur les systèmes de détection d'intrusion, objet principal de notre travail. Nous définissons tout d'abord le système de détection d'intrusion, ainsi que le mode de fonctionnement et les performances. La fin de ce chapitre est consacrée aux limitations rencontrées par l'IDS.

II.2 Définitions

La détection d'intrusion est le processus qui consiste à surveiller les événements se produisant dans un ordinateur ou dans un réseau informatique et de les analyser pour découvrir des signes d'intrusions, intrusions définies comme des tentatives de compromission de la confidentialité, l'intégrité, la disponibilité et la responsabilité, ou pour dévier des mécanismes de sécurité [Bace 00, Bsi 00, Ubizen 00].

Les intrusions sont provoquées par: l'accès d'attaquants externes aux systèmes via des réseaux ouverts comme Internet, des utilisateurs autorisés qui tentent de gagner des privilèges additionnels pour lesquels ils ne sont pas autorisés, ou des utilisateurs autorisés qui abusent de leurs privilèges [Bace 00].

Les systèmes de détection d'intrusions (IDS pour *Intrusion Detection System*) ont pour objectif de révéler, généralement via des alertes, toute activité pouvant être considérée comme intrusive, depuis ou vers un système d'information, par analyse de données. Les sources de ces données correspondent à des événements générés par différents services ou utilisateurs. Les premiers travaux en détection d'intrusions ont débuté avec Anderson en 1980 et Denning en 1987. Aujourd'hui, il existe plus de 140 systèmes de détection d'intrusions différents [Yves 09].

Philip dans [Phillip et al, 98] définit trois critères pour évaluer l'efficacité des systèmes de détection d'intrusion:

- **L'exactitude (*accuracy*):** on parle de l'exactitude quand le système de détection d'intrusion déclare comme malicieux une activité légale.
- **La performance (*performance*):** la performance de système de détection d'intrusion est la vitesse de traitement des événements. Si cette vitesse est faible, la détection en temps réel est donc impossible.
- **La complétude (*completeness*):** on parle de la complétude quand le système de détection d'intrusion rate la détection d'une attaque. Ce critère est le plus difficile parce qu'il est impossible d'avoir une connaissance globale sur les attaques.

Debar dans [Debar et al, 00] a rajouté également les deux critères suivants:

- **La tolérance aux fautes (*Fault tolerance*):** le système de détection d'intrusion doit lui-même résister aux attaques, particulièrement au déni de service. Ceci est important, parce que plusieurs systèmes de détection d'intrusion s'exécutent sur des matériels ou logiciels connus comme vulnérables aux attaques.
- **La réaction à temps (*Timeliness*):** le système de détection d'intrusion doit s'exécuter et propager les résultats de l'analyse le plus tôt possible, pour permettre à l'autorité de réagir avant que des graves dommages n'aient lieu. Ceci implique plus qu'un calcul de performance, parce qu'il ne s'agit pas seulement de temps de traitement des événements, mais aussi du temps nécessaire pour la propagation et la réaction à cet événement.

II.3 Modèle de processus de la détection d'intrusion

La majorité des systèmes de détection d'intrusion peuvent être décrits en termes de trois composants fonctionnels fondamentaux (voir Figure II.1) [Papini 05]: la source d'information, l'analyse et la réponse.

II.3.1 La source d'informations (sonde)

Les différentes sources d'événements utilisées pour déterminer les intrusions qui ont eu lieu, elles peuvent être fournies par les différents niveaux du système d'information: les réseaux, les hôtes, et les applications.

II.3.2 L'analyse

Il s'agit de la partie du système de détection d'intrusions qui organise et donne un sens aux événements dérivés des sources d'informations, décidant quand ces événements indiquent que des intrusions se produisent ou ont déjà eu lieu. Les principales approches communes d'analyse sont: la détection par scénario (*The misuse detection*) ou encore dite approche par scénarios et la détection d'anomalie (*Anomaly detection*) ou encore dite approche comportementale lesquelles seront expliquées par la suite.

II.3.3 La réponse

C'est l'ensemble des contre-mesures que le système prend une fois qu'il détecte des intrusions. Celles-ci sont typiquement groupées dans des mesures actives et passives, les mesures actives comportent une certaine interposition automatisée de la part du système, alors que les mesures passives rapportent des résultats issus de l'analyse aux responsables, qui sont alors prévenus pour agir et prendre une action basée sur ces rapports.

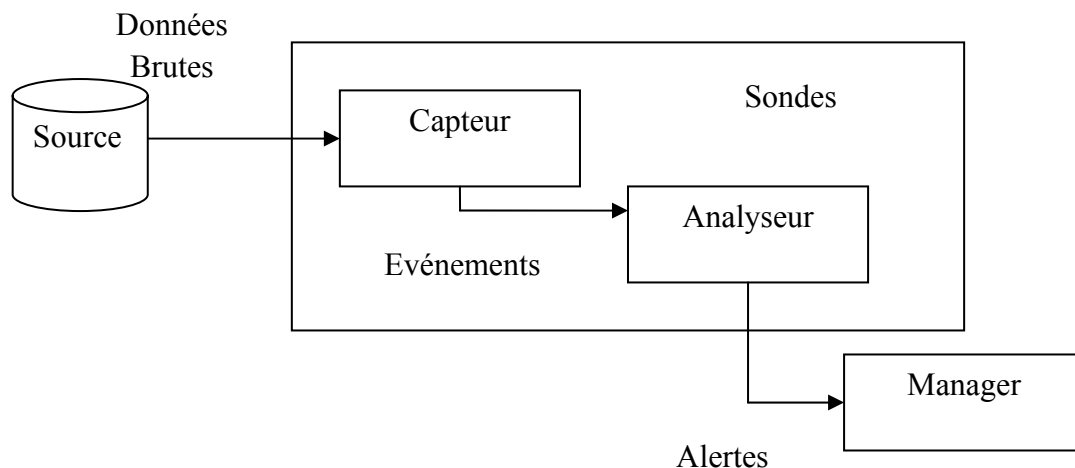


Figure II.1. Modèle simplifié d'un système de détection d'intrusions [Papini 05]

II.4 Un système de détection d'intrusion dans le réseau

L'emplacement d'un IDS dépend de types d'activités d'intrusion que l'on veut détecter. Si l'entreprise a une seule connexion WAN alors le meilleur emplacement peut être juste après le routeur. Dans le cas où l'entreprise a plusieurs connexions, on peut placer un SDI sur chaque liaison. La Figure II.2 montre un exemple de réseau avec un IDS. Dans ce cas

il est connecté à un concentrateur entre le réseau local et le pare-feu. Le trafic est donc visible dans les deux directions [Nassih 06].

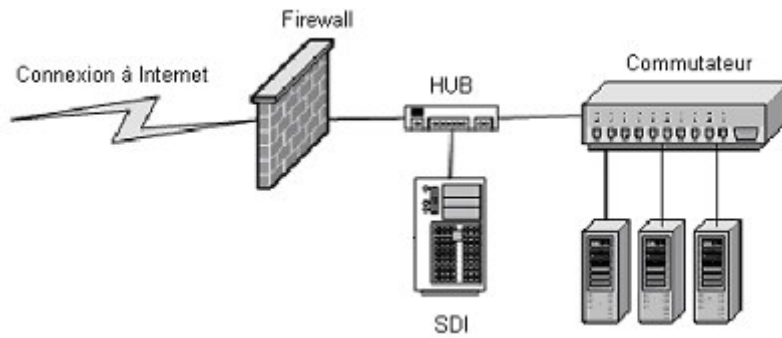


Figure II.2. L'emplacement d'un IDS [Nassih 06]

Dans le cas d'un réseau avec une zone démilitarisée (*DMZ*), un IDS peut être placé dans cette zone, de cette façon on pourra détecter les attaques qui visent les serveurs de l'entreprise. Les attaques peuvent être lancées de l'intérieur de l'entreprise, il est donc préférable d'avoir un IDS qui contrôle le trafic interne et signale les anomalies.

II.5 Objectifs

La détection d'intrusion permet aux organisations de protéger leurs systèmes contre les menaces qui ne cessent de croître. Ces menaces sont en augmentation à cause de l'extension de la connectivité du réseau public (Internet), et de l'utilisation aux systèmes informatiques qui comportent parfois des bugs. La question pour les professionnels de la sécurité ne devraient pas être s'il faut utiliser la détection d'intrusion, mais quels dispositifs utiliser et quelles sont leur capacités de détection d'intrusion.

Les systèmes de détection d'intrusion sont considérés comme un élément nécessaire dans l'infrastructure de la sécurité informatique de chaque organisation. En effet, il y a plusieurs raisons pour acquérir et utiliser les systèmes de détection d'intrusion:

- Pour détecter les attaques et autres violations de sécurité qui ne sont pas empêchées par d'autres outils de sécurité.
- Pour documenter les menaces existantes dans une organisation, c'est-à-dire découvrir les vulnérabilités avant qu'elles ne soient exploitées par un attaquant.
- Pour agir en tant que contrôle de qualité pour la conception de sécurité, particulièrement dans les grandes et complexes entreprises.
- Pour fournir des informations utiles au sujet des intrusions qui ont eu lieu, et faire des diagnostics, recouvrement, et corrections des facteurs causatifs.

- Pour arrêter les intrusions afin de limiter les dégâts. Malheureusement cela n'est pas toujours possible à cause de la complexité et la diversité des intrusions, et la naissance de nouveaux types d'intrusions liées au développement des nouvelles technologies d'information.

II.6 Techniques de mise en œuvre

Parmi les techniques utilisées dans les deux approches [Tabia 08]:

II.6.1 Par scénarios

Les techniques de modélisation et d'implémentation les plus utilisées pour élaborer des IDSs par scénarios ayant connu un succès dans la pratique sont:

II.6.1.1 Systèmes experts

Un système expert pour la détection d'intrusions comprend une base de règles codant des signatures d'intrusions, les informations d'audit sont traduites en base de faits suivant la sémantique du système expert. Le moteur d'inférence cherche alors les attaques à partir de la base de faits (données d'audits) et de règles (signatures).

II.6.1.2 Pattern matching

La plus utilisée, cette technique décrit les signatures à plusieurs niveaux sémantiques (appels systèmes, commandes systèmes, etc.) et implémente des algorithmes de localisation des patterns de ces signatures dans les informations d'audit.

II.6.1.3 Analyse état-transition

Dans cette technique, une intrusion est décrite comme un ensemble d'objectifs et d'états sous forme d'un diagramme d'état-transition. Tout événement pouvant atteindre l'état attaque est considéré comme une intrusion.

II.6.1.4 Fouille de données et apprentissage automatique

Les techniques de fouille de données et d'apprentissage automatique sont très prometteuses dans la mesure où ces techniques peuvent, à partir de données d'audit, apprendre ou extraire, des signatures d'intrusions ou des modèles de comportements malveillants.

II.6.2 L'approche comportementale

Concernant les techniques de modélisation et d'implémentation utilisées dans le cadre de l'approche comportementale, ci-après les plus utilisées:

II.6.2.1 Méthodes statistiques

En utilisant des méthodes statistiques, la modélisation du profil de l'activité normale se base sur les distributions de certaines variables aléatoires relatives à certains paramètres des activités normales (utilisation CPU, requêtes lancées sur un serveur, etc.).

II.6.2.2 Systèmes experts

Ici, le profil normal basé sur des paramètres statistiques, ils sont traduit sous forme de règles. Les données d'audit constituent après certaines adaptations une base de faits. Le moteur d'inférence tente d'identifier en utilisant la base de règles des anomalies dans la base de faits.

II.6.2.3 Fouille de données

Les techniques de fouille de données sont utilisées soit pour l'extraction et l'apprentissage des profils des activités normales à partir de données d'audit normales, soit pour détecter des comportements très déviants et distants des comportements normaux connus.

II.7 L'évaluation des IDS

La performance d'un système de détection d'intrusions, et notamment de sa méthode d'analyse, est liée à deux notions importantes qui permettent d'évaluer ces performances [Bishop 03]:

II.7.1 Faux négatif

Idéalement, toute intrusion doit donner lieu à une alerte. Une intrusion non détectée, c'est-à-dire n'ayant pas généré d'alerte, constitue alors un faux négatif. La fiabilité (ou couverture) d'un Analyseur est liée à son taux de faux négatifs qui représente alors le pourcentage d'intrusions non détectées, ce taux devant être le plus bas possible.

II.7.2 Faux positif

Toute alerte doit correspondre à une intrusion effective. Lorsqu'un système de détection d'intrusions génère une alerte qui n'a pas lieu d'être, cette alerte est qualifiée de faux positif. La pertinence (ou crédibilité) d'un Analyseur est liée à son taux de faux positifs qui représente alors le pourcentage de fausses alertes.

II.8 Les limitations des systèmes de détection d'intrusion

Les systèmes de détection d'intrusion, souffrent généralement de limitations communes, on peut citer les points suivants [Yves 09]:

- Le nombre important de faux positifs et faux négatifs.
- L'identification faible et imprécise: même lorsque l'IDS détecte des attaques, il les identifie parfois mal.
- La corrélation des alertes.
- Les nouvelles attaques: sont les attaques qui jamais n'ont été observées avant ou précédemment. Ces attaques essayent d'exploiter des vulnérabilités.
- Variation d'attaque.

II.9 Classification des systèmes de détection d'intrusion

Le domaine de la détection d'intrusions est encore jeune mais en plein développement. Nous dénombrons à l'heure actuelle environ une centaine de systèmes de détection d'intrusions, que ce soit des produits commerciaux ou du domaine public [Bace 00]. Il est donc devenu très utile d'utiliser des critères pour classer ces systèmes de détection d'intrusion, c'est ce que nous allons présenter dans ce chapitre.

Il existe plusieurs critères qu'on peut utiliser pour classer les différents systèmes de détection d'intrusion, dont les principaux sont résumés dans Figure II.3.

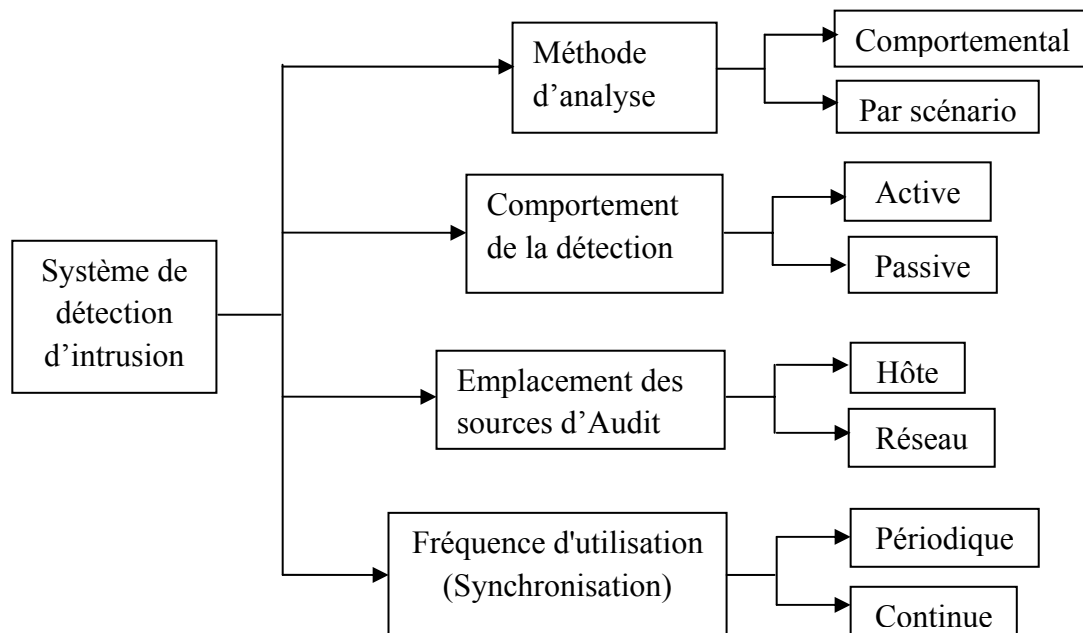


Figure II.3. Taxonomie des systèmes de détection d'intrusion [Debar et al, 00]

II.10 La méthode d'analyse

La méthode d'analyse définit l'ensemble des techniques utilisées par les systèmes de détection d'intrusion dans le processus de la détection. L'approche est dite par scénario si le détecteur analyse les informations relatives aux attaques, et elle est dite comportementale si le détecteur analyse les informations relatives au comportement normal du système [Debar et al, 00] [Briffaut 08].

Dans la première méthode, l'analyse vise quelque chose de connu et qualifié de "mauvais". Cette technique est utilisée par la plupart des systèmes commerciaux. Dans la deuxième méthode, l'analyse cherche les modèles anormaux de l'activité. D'une autre façon, la détection par scénario se base sur les caractéristiques d'une attaque connue pour la détecter. Par contre, la détection comportementale se base sur la définition d'un modèle d'utilisation normal pour détecter tout ce qui est anormal. Chaque approche présente des avantages et des inconvénients.

II.10.1 La détection par scénario

La détection par scénario (mauvaise utilisation) considère comme normal tout ce qui n'est pas hostile, et elle adopte la politique suivante: " si ce n'est pas dangereux, alors c'est normal ". Donc, il est impératif de bien connaître les attaques possibles, la détection par scénarios (ou *misuse detection*) permet de détecter une attaque connue via la définition d'un scénario. Cette approche utilise une base de connaissances, appelée base de signatures d'attaques et une méthode de recherche de motifs permettant de reconnaître les signatures définies. Un détecteur d'intrusions par scénario est alors composé de:

- Un ensemble de sondes produisant un flux d'évènements.
- Une base de signatures d'attaques.
- Un algorithme de recherche de motifs, comparant le flux d'évènements aux signatures contenues dans la base.

➤ **Avantages**

- ✓ Très efficace pour détecter des attaques sans produire un grand nombre de fausses alertes.
- ✓ Fiabilité pour les attaques connues.
- ✓ Peut rapidement et sûrement diagnostiquer l'utilisation d'un outil spécifique ou une technique d'attaque. Ceci peut aider les responsables de sécurité à donner la priorité aux mesures correctives.

➤ **Inconvénients**

- ✓ Peut seulement détecter les attaques connues, dont les signatures sont introduites dans le système, donc le système de détection doit être constamment mis à jour avec les signatures des nouvelles attaques.
- ✓ Beaucoup de systèmes adoptant cette approche sont conçus pour employer un nombre limité de signatures qui peuvent être définis, ce qui les empêchent de détecter des variantes de ces attaques.

II.10.2 La détection d'anomalie (comportementale)

Cette approche comporte deux phases:

La première étant la construction d'un profil pertinent, la seconde correspond à l'évaluation de la déviation du comportement observé par rapport à ce profil. La définition du profil détermine le comportement attendu du système. Ainsi toute déviation par rapport à ce profil donnera lieu à une alerte sans aucune connaissance de la cause de cette déviation.

Cette approche permet donc de détecter une attaque sans aucune connaissance du fonctionnement de celle-ci et donc de pouvoir potentiellement détecter des attaques inconnues. De plus, en supposant que ce profil n'évolue que rarement, cette approche ne requiert qu'une maintenance minimale. Cependant, la définition du profil et l'évaluation des déviations sont deux points critiques qui influent sur la "qualité" de la détection. En plus cette approche produit généralement un taux de faux positifs élevé. En outre, une des hypothèses forte de cette approche est que, durant la phase d'observation, aucune attaque ou intrusion n'a eu lieu. Ainsi, la phase d'observation doit être effectuée sur des systèmes sûrs et avec des utilisateurs de confiance. Dans le cas contraire, on se heurte à un problème de faux négatifs, certaines attaques ou intrusions n'étant pas détectées car faisant partie du profil. Et le choix des paramètres du profil représentatif des entités surveillées est un problème majeur.

La détection des déviations pose aussi un problème. Ainsi, un seuil de déclenchement trop bas (haut) peut donner lieu à un taux de faux positifs (négatifs) élevé. Les mises-à-jour des profils posent aussi problème.

➤ **Avantages**

- ✓ Les systèmes de détection d'intrusion basés sur la détection d'anomalie détectent le comportement peu commun, et ils ont ainsi la capacité de détecter des symptômes des attaques connues et inconnues sans la connaissance spécifique des détails.
- ✓ Cette approche permet de produire l'information utile pour la définition des signatures pour les systèmes de détection d'intrusion à base de signatures.

➤ **Inconvénients**

- ✓ Le point noir de cette approche est le grand nombre de faux positif dues aux comportements imprévisibles des utilisateurs du réseau.
- ✓ Elle exige souvent l'historique à long terme des événements enregistrés afin de caractériser les modèles normaux de comportement. Les systèmes basés sur cette approche doivent être dotés d'une certaine intelligence pour raison d'apprentissage automatique en utilisant par exemple les réseaux de neurones.
- ✓ Risque d'attaque lors de la construction des profils.
- ✓ Evolution des profils au cours du temps peut être vue comme une faille.

II.11 Le comportement de la détection (la réponse)

Le comportement de la détection décrit la réponse du système de détection d'intrusion à une attaque. Elle est qualifiée d'active, si le détecteur réagit activement par des actions correctives, ou proactives (changer les règles de filtrage de Firewall, arrêter des connexions TCP, ou encore attaquer l'attaquant, etc.). Si le système de détection d'intrusion génère simplement des alertes (afficher un message sur l'écran, générer un son spécifique, envoi d'un email, archivage dans un fichier ou dans une base de données, etc.), la réponse est qualifiée de passive.

II.11.1 Les réponses actives

Les réponses actives des IDSs sont des actions automatisées prises quand certains types d'intrusions sont détectés. Il y a trois catégories de réponses actives:

II.11.1.1 Rassembler des informations additionnelles

Il est très important de rassembler des informations additionnelles sur une attaque afin de l'identifier avec précision. Chacun de nous a fait probablement l'équivalent de cela une fois réveillé par un bruit étrange pendant la nuit. La première chose à faire dans une telle situation est d'écouter d'avantage, recherchant l'information additionnelle qui nous permet de décider si on doit agir ou non. Dans le cas des IDSs, cela se traduira par l'exigence d'analyse des informations additionnelles, faire des corrélations, ou bien communiquer avec d'autres types d'IDSs installés sur le réseau.

II.11.1.2 Changer l'environnement

Une autre réponse active doit stopper une attaque en progression et puis bloquer l'accès de l'attaquant. Typiquement, les IDSs n'ont pas les capacités de bloquer l'accès d'une personne spécifique, mais ils peuvent uniquement rompre des connexions ou bloquer certains paquets spécifiques en s'appuyant sur les mécanismes des protocoles Internet, cela est dû à la capacité du hacker expert de construire des paquets falsifiés (*forging packets*). Parmi ces actions on trouve:

- L'envoi des paquets TCP de type Reset ou des paquets ICMP au système de l'attaquant pour arrêter la connexion.
- La configuration des routeurs et des Firewalls pour bloquer les paquets provenant de l'adresse IP de l'attaquant.

- La configuration des routeurs et des Firewalls pour bloquer les paquets selon le numéro de port, le protocole, ou le service utilisé par l'attaquant.

II.11.1.3 Agir contre l'intrus

La première option dans la réponse active est d'agir contre l'intrus. En effet, la forme la plus agressive de cette réponse implique le lancement des contre-attaques ou d'essayer d'obtenir activement les informations sur le hôte ou l'emplacement de l'attaquant. Cependant, à cause des ambiguïtés légales au sujet de la responsabilité civile, cette option peut représenter un grand risque qu'une contre-attaque réussisse. La première question concernant le choix de cette option même avec beaucoup d'attention est: « est ce que notre action peut être illégale? ». Beaucoup d'attaquants emploient de fausses adresses de réseau quand ils attaquent les systèmes, ce qui peut être la cause d'endommagement des sites Internet ou de torts causés aux utilisateurs innocents. En conclusion, il faut prendre ces actions avec plus de prudence.

II.11.2 Les réponses passives

Les réponses passives des IDSs fournissent l'information nécessaire aux administrateurs réseau et aux responsables de la sécurité pour les aider à prendre des mesures basées sur cette information. Beaucoup d'IDSs se fondent seulement sur des réponses passives dont les principales sont:

II.11.2.1 L'alerte

Les alertes sont produites par les IDSs pour informer les administrateurs réseau quand des attaques sont détectées. La forme la plus commune est d'afficher un message d'alerte contenant des informations détaillées de l'intrusion détectée sur la console du responsable de la sécurité réseau. Une autre option très utile consiste à envoyer ces alertes au téléphone du responsable, on peut aussi envoyer des e-mails, ou générer des alertes sonores.

II.11.2.2 SNMP Trap

Certains IDSs sont conçus pour produire des alertes et envoyer les rapports au système de gestion de réseau (*network management system*). Ils utilisent le protocole SNMP (*Simple Network Management Protocol*), qui est un protocole dédié à la gestion du réseau.

II.11.2.3 L'archivage

L'archivage (*logging*) permet aux analystes de faire des analyses approfondies, et de faire des corrélations avec l'historique dont ils disposent concernant les événements qui se sont produits auparavant.

II.12 L'emplacement des sources d'audits

La manière la plus connue pour classer les IDSs est de les grouper par sources d'informations (sondes). Certains IDSs analysent des paquets capturés à partir du réseau, en plaçant des *sniffers* sur les différents segments du réseau local. D'autres IDSs analysent des informations produites par le système d'exploitation ou par des applications pour la recherche des signes d'intrusions.

II.12.1 NIDS (*Network-Based IDS*)

Ces outils analysent le trafic réseau, ils comportent généralement une sonde qui "écoute" sur le segment de réseau à surveiller et un moteur qui réalise l'analyse du trafic afin de détecter les signatures d'attaques. Les IDSs réseau à base de signatures sont confrontés actuellement à deux problèmes majeurs qui sont: l'utilisation grandissante du cryptage, et les réseaux commutés. En effet, d'une part, le cryptage rend l'analyse du contenu des paquets presque impossible, d'autre part il est plus difficile "d'écouter" sur les réseaux commutés. La plupart des NIDS sont aussi dits IDS on-line car ils analysent le flux en temps réel. Pour cette raison, la question des performances est très importante. De tels IDSs doivent être de plus en plus performants afin d'analyser les volumes de données de plus en plus importants pouvant transiter sur les réseaux.

II.12.2 HIDS (*Host-Based IDS*)

Les IDSs de ce type analysent le fonctionnement et l'état des machines sur lesquelles ils sont installés afin de détecter les attaques. Pour cela ils ont pour mission l'analyse des journaux système (*logs*), le contrôle d'accès aux appels systèmes, la vérification d'intégrité des systèmes de fichiers, etc. Ils sont très dépendants de système sur lequel ils sont installés. Il faut donc employer des outils spécifiques en fonction des systèmes déployés. Ces IDSs peuvent s'appuyer sur des fonctionnalités d'audit propres ou non au système d'exploitation, pour en vérifier l'intégrité, et générer des alertes. Il faut cependant noter qu'ils sont incapables

de détecter les attaques exploitant les faiblesses de la pile TCP/IP du système, typiquement les Dénis de Service.

II.12.3 IDS d'application

Similaires aux HIDSs, ils sont installés sur un serveur ou une machine pour détecter les attaques relatives à une application donnée. Par exemple un IDS installé sur un serveur Oracle pour détecter les intrusions relatives à Oracle.

II.12.4 IDS hybrides

Les IDS hybrides rassemblent les caractéristiques de plusieurs IDS différents. En pratique, on ne retrouve que la combinaison de NIDS et HIDS. Ils permettent, en un seul outil, de surveiller le réseau et l'hôte. Les sondes sont placées dans des points stratégiques, et agissent comme NIDS et/ou HIDS suivant leurs emplacements. Toutes ces sondes remontent alors les alertes à une machine qui va centraliser, agréger, et lier les informations d'origines multiples.

II.13 La fréquence d'utilisation (la synchronisation)

La synchronisation se rapporte au temps écoulé entre les événements qui sont surveillés et l'analyse de ces événements. Elle est réalisée en: temps réel ou différé:

II.13.1 En temps différé (périodique)

Dans cette classe, le flux d'informations émanant des points de surveillance vers les détecteurs n'est pas continu. En effet, l'information est traitée dans un mode semblable au principe "emmagasiner et expédier". Cette approche est employée surtout dans les Host-IDSs qui scrutent les logs du système d'exploitation dans des intervalles de temps réguliers.

II.13.2 En temps réel (continu)

Les IDSs en temps réel traitent des flux continus d'informations à partir des différentes sources d'informations. C'est la technique prédominante de synchronisation pour les IDSs

réseau, qui recueillent l'information du trafic réseau. Par conséquent Les IDSs peuvent prendre des actions pour affecter la progression d'une attaque détectée.

Dans [Bace 00] Rebbeca a étudié d'autres critères de classifications moins connus, tels que l'architecture et la stratégie de contrôle:

II.14L'architecture

L'architecture des IDSs se rapporte à la manière d'ajuster leurs composants architecturaux. Les composants architecturaux de base sont l'hôte (*Host*) où l'IDS s'exécute et la cible (*Target*), qu'il soit hôte ou réseau que l'IDS doit protéger. Les principales architectures types sont:

II.14.1 Host Target Colocation (Cohabitation de la cible et de l'hôte)

Les premiers IDSs fonctionnaient sur les systèmes qu'ils étaient sensés protéger. Ceci était dû au fait que la plupart des systèmes étaient des systèmes centraux (*mainframe*), et le coût élevé d'ordinateurs faisait d'une architecture séparée un mauvais choix. Ceci présente un problème de point de vue de la sécurité. En effet, n'importe quel attaquant qui réussit une attaque sur le système peut neutraliser l'IDS, étant donné que l'emplacement de ce dernier est connu.

II.14.2 Host Target Separation (Séparation entre la cible et l'hôte)

Avec l'arrivée des postes de travail et des ordinateurs individuels, la plupart des architectures des IDS ont orienté les systèmes d'analyse et de commande vers un système séparé, par conséquent, séparant la machine d'IDS et la cible. Ceci a amélioré la sécurité des IDSs parce qu'il est devenu plus facile de cacher leurs existences.

II.15La stratégie de contrôle

La stratégie de contrôle décrit comment les éléments d'IDS sont commandés, et comment les entrées et les sorties d'IDS sont contrôlées. Elle peut être: centralisée, partiellement distribuée, ou Entièrement distribuée.

II.15.1 Centralisée

La stratégie centralisée consiste à commander à partir d'une console centrale la surveillance, la détection et l'audit. Dans cette stratégie, il y a une seule console de commande. A partir de laquelle on communique avec les différents systèmes de surveillance: réseau (NIDSs), hôte (HIDSs) et applications, et s'il y a des indices d'intrusion, on lance des commandes pour changer les règles de sécurité au niveau des firewalls et routeurs.

II.15.2 Partiellement distribuée

La surveillance et la détection sont commandées à partir d'un nœud local de commande, avec un système hiérarchique d'audit. Dans ce cas, on trouve dans chaque sous réseau une console qui fournit des rapports à la console de niveau supérieur (entreprise *IDS consol*). Les actions sont prises à partir de cette dernière. Si l'entreprise comporte des réseaux géographiquement séparés, un système indépendant est mis en œuvre pour la surveillance, la détection et la réaction. La console principale a comme but de commander les différents IDSs de l'entreprise.

II.15.3 Entièrement distribuée

La surveillance et la détection sont réalisées en utilisant une approche basée sur des agents, où les décisions de réponse sont prises au lieu où l'analyse s'effectue (IDSs autonomes).

II.16 Conclusion

Ces dernières années, les systèmes de détection d'intrusion ont gagné une place importante dans la conception de la sécurité des systèmes d'information. Ils sont largement déployés dans les entreprises pour diverses raisons telles que: la documentation des attaques, l'évaluation de la sécurité, et plus généralement la surveillance des systèmes d'information pour arrêter, voir empêcher les attaques afin de limiter les dégâts. Les systèmes de détection d'intrusion se caractérisent principalement par:

- La méthode de détection, on distingue deux principales méthodes: détection par scénario et détection d'anomalie. Ces deux méthodes présentent des avantages et des inconvénients;
- Les sources d'information, qui peuvent être: le réseau, le hôte et les applications;
- Le comportement de la détection, qui peut être passif ou actif;

- La synchronisation entre les sources et le détecteur, qui peut être périodique ou continue pour les IDS temps réel.

Enfin, notons que le système de détection d'intrusion n'est pas une solution complète, il possède beaucoup de limites et problèmes liés à plusieurs facteurs tels que: l'observation des événements, le facteur humain et les attaques par déni de service. Mais le problème majeur de ce mécanisme de sécurité est le taux très important de faux positifs, par la suite nous avons détaillé les différents axes de recherche qui traitent de ce problème.

Chapitre 3

LA DÉTECTION DE FAUX POSITIFS

Chapitre 3

La détection de faux positifs

III.1 Introduction

Nous avons noté que les systèmes de détection d'intrusion (*IDS*) sont parmi les outils de sécurité les plus récents. On peut les classer en différents types selon leurs caractéristiques: techniques de détection, architecture et portée de détection.

Malheureusement, malgré leur utilité, en pratique la plupart des IDS souffrent plus ou moins de deux problèmes: le nombre important de faux positifs et de faux négatifs. Les faux positifs (les fausses alertes) sont générés lorsque l'IDS identifie des activités normales comme des intrusions. Le nombre de faux positifs peut atteindre jusqu'à mille alertes par jour, alors que les faux négatifs correspondent aux attaques ou intrusions qui ne sont pas détectées (aucune alerte n'est générée).

III.2 Les raisons de faux positifs

Parmi les raisons essentielles qui font que, les IDSs ne peuvent pas analyser de manière infaillible les activités d'un réseau on trouve :

- Dans certains cas, la détermination d'une base de signature équilibrée est difficile et peut identifier des actions légitimes comme intrusions.
- Les actions qui sont normales dans certains environnements peuvent être malveillantes dans d'autres. L'*IDS* utilise une configuration standard qui peut identifier éventuellement beaucoup d'activités normales comme malveillantes.

III.3 La détection de faux positifs

A cet effet beaucoup de travail a été effectué pour aider l'analyste à détecter les faux positifs. Les méthodes actuelles peuvent être divisées en trois catégories, qui sont décrites ci-après [Xiao et al, 08].

III.3.1 Des méthodes basées sur la classification

Le principe de cette méthode consiste à construire un classifieur d'alerte qui détermine les faux positifs et les vrais positifs. Tadeusz Pietraszek a présenté un système dans [Pietraszek 04], qui génère d'une part une base d'apprentissage basée sur le *feedback* de l'analyste. Ensuite, ces données sont utilisées par le processus d'apprentissage automatique pour construire et mettre à jour le classifieur. Enfin, ce dernier est utilisé pour traiter les nouvelles alertes. Cette méthode peut classifier les alertes automatiquement, mais la difficulté se situe au niveau du processus d'apprentissage. En effet, la question est celle de comment construire une base de connaissance fiable?

L'approche de machine learning est l'une des méthodes basée sur la classification et détaillée ci-dessous.

Approche de machine Learning ALAC

La machine Learning établit un classifieur d'alerte *ALAC (Adaptive Learner for Alert Classification)* qui indique les faux positifs (voir Figure III.1). Les alertes sont classifiées en faux et vrais positifs, aussi la classification peut être étendue pour indiquer le type d'une attaque [Pietraszek et al, 05].

Les alertes sont classifiées par un classifieur d'alertes, ces dernières peuvent être classifiées automatiquement en utilisant une technique de *machine learning* ou manuellement par un expert [Pietraszek 06].

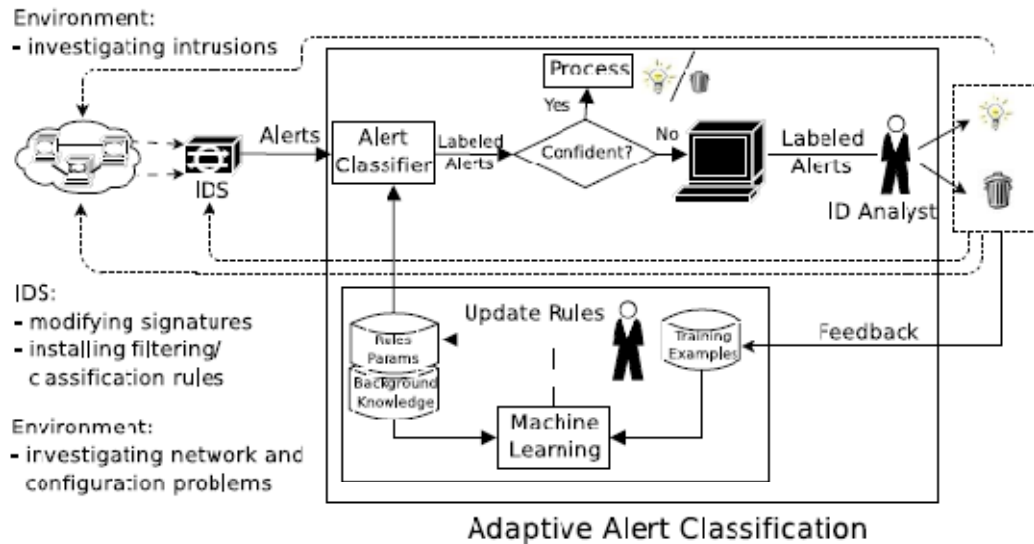


Figure III.1. Machine learning (ALAC) [Pietraszek et al, 05]

Le système utilise un classifieur d'alerte pour assigner des étiquettes aux alertes et les passe à l'analyste. Ce dernier étudie les étiquettes et les corrige au besoin, actuellement, il existe un modèle simple d'interaction homme-machine, dans lequel l'analyste classe séquentiellement les alertes en faux et vrais positifs, et les transmet à la base de connaissance (*training example*).

Formellement, il y a un analyste humain O de détection d'intrusion révisant un ordre des alertes $(A1, A2, \dots, AI, \dots)$ dans un fichier log L . La revue est faite en assignant un ensemble prédéfini de classes $\{C1, C2, \dots, CN\}$ (qui peuvent être en particulier: vrais positifs ou faux positifs $\{\ll + \gg, \ll - \gg\}$) pour chaque alerte. La revue est typiquement faite séquentiellement et en temps réel [Pietraszek 06].

Donner

- Une séquence d'alertes : $(A1, A2, \dots, AI, \dots)$ dans un log L ,
- Un ensemble de classes $C = \{C1, C2, \dots, NC\}$,
- Un analyste de détection d'intrusion O en temps réel,
- Une fonction U réduisant le coût de fausse classification,

Trouver

- Le classifieur classe les alertes, maximise la fonction U .

ALAC fonctionne en deux modes:

➤ **Mode recommander**

Le système ne traite aucune alerte automatiquement, mais prévoit seulement les étiquettes et les envoie à la console afin de les vérifier par l'analyste (Figure III.2). Ceci signifie que l'analyste doit réviser toutes les alertes [Pietraszek et al, 05].

L'utilisation de la base de connaissance a pour objectif d'apprendre des règles améliorées, Ces règles sont alors employées par (*ALAC*) pour classifier les alertes.

Grâce à l'utilisation des classifieurs interprétables, l'analyste peut examiner les règles pour s'assurer qu'elles sont correctes.

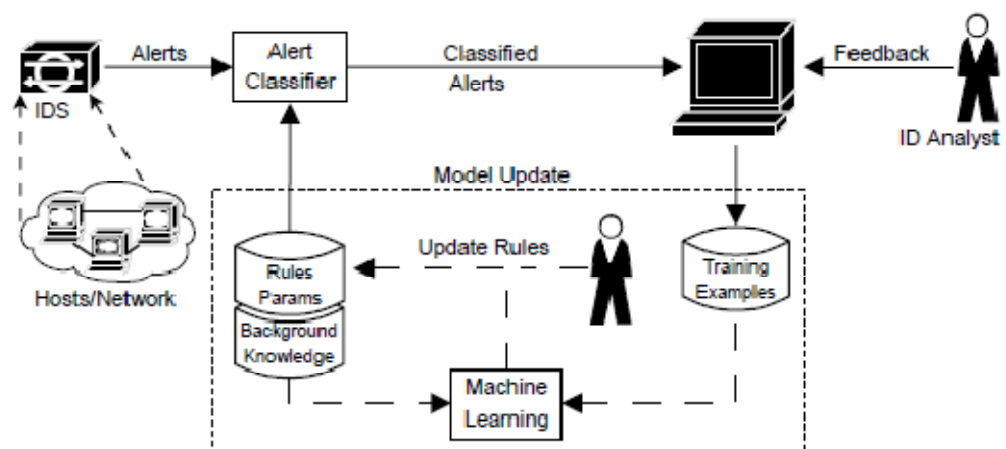


Figure III.2. Architecture ALAC en mode recommander [Pietraszek 06]

L'algorithme (ALAC) en mode *recommender* est présenté ci-dessous (Figure III.3) :

Algorithme (ALAC) en Mode recommender:

Entrée : une sequence d'alertes $(A_1, A_2, \dots, A_n)$

Sortie : une sequence d'alertes classifiées $((A_1, C_{A_1}), (A_n, C_{A_n}), \dots, (A_n, C_{A_n}))$

```

1  Initialisation ;
   /*les alertes utilisées pour l'apprentissage initial */
2   $x \leftarrow x_0$ ;
3  Tq  $(x < n)$  faire
4       $S_i \leftarrow \text{sous sequence}(A_1, \dots, A_x)$ ;
5       $C_i \leftarrow \text{Notification de mise à jour du classifieur}(C_{i-1}, S_i)$ ;
6      Tq bonne performance de  $\text{classification}(WA_{th})$  faire
7           $C_x \leftarrow \text{classifie}(C_i, A_x)$ ;
8           $C_{A_x} \leftarrow \text{demander à l'analyste de vérifier la classification}(A_x, C_x)$ ;
9           $\text{mettre à jour la performance de la classification}(C_x, C_{A_x})$ ;
10          $x \leftarrow x + 1$ ;
11     Fin
12      $i \leftarrow i + 1$ ;
13 Fin
  
```

Figure III.3. Pseudo-code de l'algorithme (ALAC) en Mode recommender [Pietraszek 06]

➤ **Mode agent:**

Le système évalue la confiance de la classification, c'est-à-dire si elle est au-dessus d'un seuil prédéfini alors le système traite les alertes automatiquement. Dans le cas des faux positifs, le système les rejette. Dans le cas des vrais positifs le système peut enregistrer l'événement pour l'administrateur ou lancer une réponse automatisée (Figure III.4).

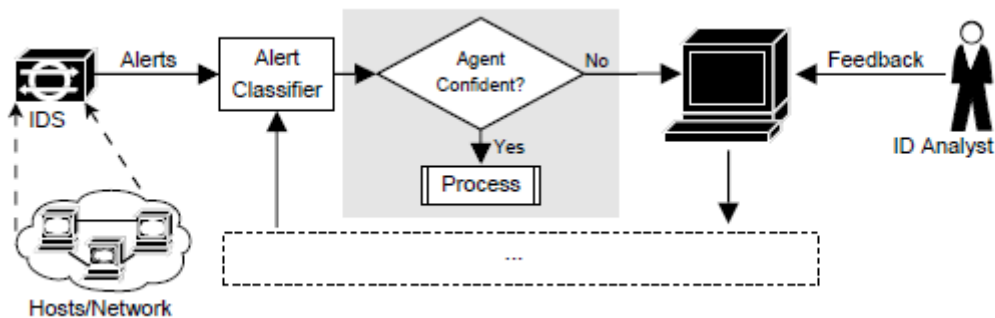


Figure III.4. Architecture ALAC en mode agent [Pietraszek 06]

L'algorithme (ALAC) en mode *agent* est détaillé sous dessous (Figure III.5):

```

Entrée : une sequence d'alertes  $(A_1, A_2, \dots, A_n)$ 
Sortie : une sequence d'alertes classifiées  $((A_1, C_{A_1}), (A_n, C_{A_n}), \dots, (A_n, C_{A_n}))$ 

1  Initialisation ;
   /* les alertes utilisées pour l'apprentissage initial */
2   $x \leftarrow x_0$ ;
3  Tq  $x \leq n$  faire
4       $S_i \leftarrow \text{sous sequence}(A_1, \dots, A_x)$ ;
5       $C_i \leftarrow \text{Notification de mise à jour du classifieur}(C_{i-1}, S_i)$ ;
6      Tq bonne performance de  $\text{classification}(WA_{th})$  faire
7           $C_x \leftarrow \text{classifie}(C_i, A_x)$ ;
8          Si  $(\text{nonconfident}(C_x, c_{th})) \vee (C_x == \text{VRAIE\_ALERTE}) \vee (\text{prélèvements}(s))$ 
9              alors
10                  $C_{A_x} \leftarrow \text{demander à l'analyste vérifier la classification}(A_x, C_x)$ ;
11             Fin
12             Sinon
13                 /* e.g. , éliminer les faux positif */
14                 traitement automatique() ;
15                  $C_{A_x} \leftarrow \emptyset$ ;
16             Fin
17             Mettre à jour la performance de la classification( $C_x, C_{A_x}$ );
18              $x \leftarrow x + 1$ ;
19         Fin
20      $i \leftarrow i + 1$ ;
21 Fin

```

Figure III.5. Pseudo-code de l'algorithme (ALAC) en Mode agent [Pietraszek 06]

L'inconvénient de ce mode est que les alertes classifiées et traitées automatiquement ne peuvent pas être ajoutées à la liste de la base de connaissance (*training example*) car l'analyste ne les a pas révisées.

III.3.2 Des méthodes basées sur les causes d'origine

Cette méthode essaye de découvrir tout d'abord les causes des alertes générées par l'IDS. Ensuite, en fonction de ces causes, les alertes déclenchées par les attaques peuvent être distinguées à partir de celles déclenchées par des événements bénins. Dans [Julisch 02], [Julisch 01] et [Julisch 03], Klaus Julisch a proposé un modèle qui repose sur *clustering*. Il a généralisé les alertes suivant chaque attribut de l'alerte en utilisant la généralisation hiérarchique. Ensuite les alertes généralisées sont présentées à l'analyste pour l'aider à découvrir les causes d'origine.

Le principal inconvénient de cette méthode est la généralisation hiérarchique qui n'est pas facile à construire. Et le résultat dépend de l'expérience des experts. De plus ce genre de méthodes fonctionne en temps différé.

L'approche proposée par Julisch est l'une des méthodes basée sur les causes d'origine détaillées ci-dessous :

Approche de data mining CLARAty

Dans [Julisch 01] [Julisch 02], Julisch propose d'adapter une méthode de fouille de données sous le nom CLARAty (*Clustering Alerts for Root cause Analysis*) (voir Figure III.6), elle est basée sur la méthode d'AOI (*attribute-oriented induction*) pour grouper les alertes et identifier le phénomène à l'origine des groupes d'alertes (voir Figure III.8).

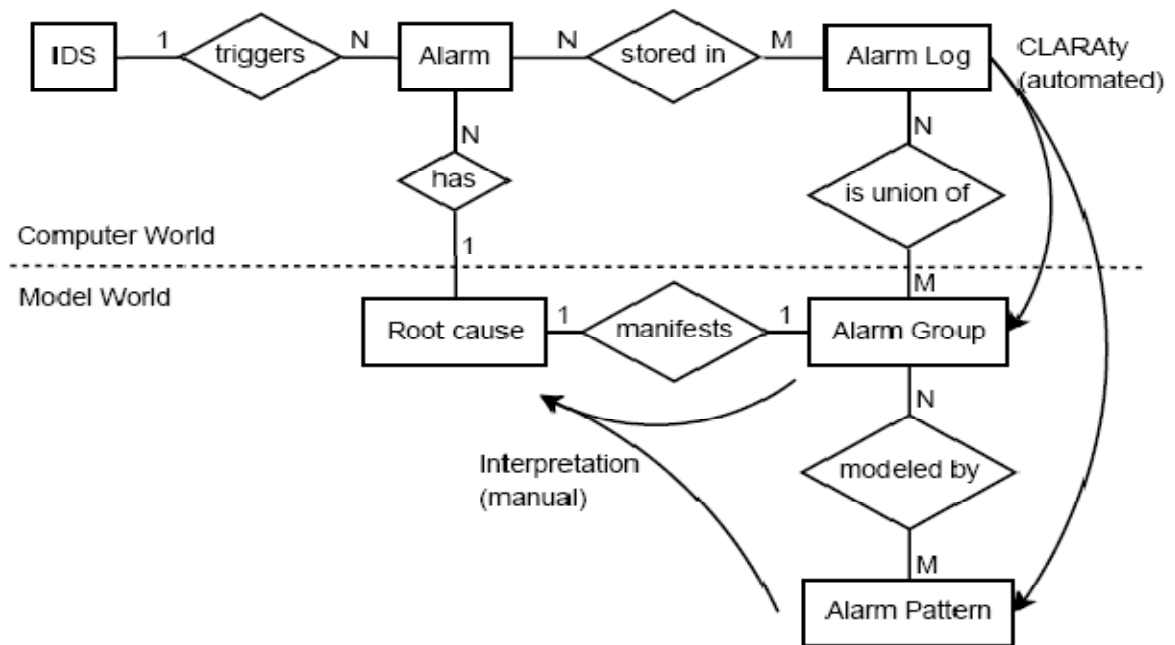


Figure III.6. Diagramme entité-association utilisé par CLARAty [Pietraszek 06]

De manière générale, l'AOI consiste à fusionner des données représentées par des n -uplets d'attributs en fonction de *hiérarchies de concepts* (ou *taxonomies*), liées à chaque attribut.

Formellement, une hiérarchie de concepts est un ensemble fini constitué de l'ensemble des valeurs que peut prendre un attribut, muni d'un ordre partiel. Le niveau d'abstraction des valeurs d'attributs (ou nœuds) des diagrammes va croissant des feuilles au sommet de l'arborescence. Les attributs des données de base en l'occurrence les alertes issues des sondes appartiennent aux feuilles des arborescences.

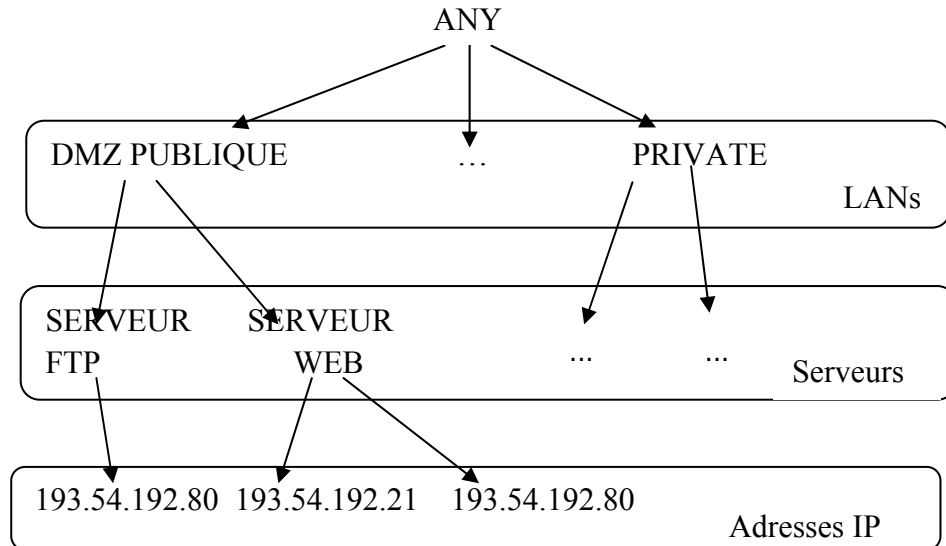


Figure III.7. Hiérarchie pour les cibles des attaques [Debar et al, 04]

La Figure III.7 représente la hiérarchie de l'attribut correspondant à la cible des attaques. Le niveau d'abstraction le plus bas est composé d'adresses IP (contenues dans les alertes), suivi de types d'hôtes particuliers, puis de domaines du réseau surveillé. Dans l'approche de Julisch, les alertes sont des quadruplets (*ident*, *source*, *cible*, *t*) où *ident* est l'identifiant de l'attaque fourni par l'IDS, *source* est l'adresse IP source de l'attaque, *cible* l'adresse IP destination et *t* la date d'occurrence.

Le mécanisme d'AOI consiste à abstraire progressivement les valeurs des attributs des données en suivant la hiérarchie de concepts associée à chaque type d'attribut et à fusionner les données rendues égales par l'abstraction. De cette manière, le nombre de tuples est diminué et les valeurs des attributs sont plus abstraites. Les alertes fournies à l'opérateur sont moins nombreuses et moins spécifiques. L'un des avantages de cette approche est d'intégrer des connaissances environnementales dans les arbres de concepts (la topologie, par exemple) et de les exploiter pour *qualifier* les groupes d'alertes : les attributs des alertes de haut niveau fournies à l'opérateur ne sont pas l'union des attributs des alertes du groupe.

L'algorithme d'AOI conventionnel consiste à abstraire les valeurs des attributs jusqu'à ce que le nombre total de données devienne inférieur à un seuil fixé. Ce critère d'arrêt est mal adapté à la corrélation d'alertes car le nombre souhaitable d'alertes issues du processus d'abstraction n'est pas connu au préalable. De plus, dans le processus d'AOI conventionnel, une abstraction d'alerte selon un attribut est faite sur l'ensemble des données, même si cela n'est pas pertinent. Cette abstraction sauvage conduit à une sur généralisation des données.

Pour ces raisons, Julisch propose un algorithme modifié d'AOI. A chaque itération de l'algorithme, un attribut est sélectionné selon une heuristique pour être généralisé. Les groupes d'alertes qui se généralisent en une même alerte et contenant plus d'alertes qu'un seuil préalablement fixé sont retirés de l'ensemble des alertes à généraliser et sont fournis à l'opérateur. L'abstraction effectuée sur les autres alertes est annulée et un autre attribut est choisi pour être généralisé. De cette manière, les alertes ne sont pas sur généralisées.

Cette approche permet par exemple de constituer des alertes du type « un **serveur proxy** génère des **scans de ports** vers l'**extérieur** » (les termes en gras sont les attributs des alertes générées par le mécanisme de corrélation). **Serveur proxy** est la source issue de l'abstraction de l'adresse IP d'un serveur proxy ; **scan de port** est un identifiant d'attaque n'ayant subi aucune abstraction ; **extérieur** est la destination de l'attaque, issu de plusieurs abstractions de la cible.

L'approche de Julisch n'a pas pour objectif de construire des scénarios d'attaques, mais plutôt d'effectuer des regroupements d'alertes correspondant à des *tendances* remarquables dans une base d'alertes. L'opérateur peut traiter les alertes par lots et donc se concentrer sur les alertes éventuellement plus sévères.

L'approche de Julisch a toutefois quelques inconvénients. Tout d'abord, les hiérarchies des structures d'attributs sont des arbres, par conséquent leur pouvoir expressif est limité. Des graphes acycliques seraient préférables. Ensuite, le système d'AOI n'est pas incrémental : il doit être ré exécuté à chaque fois que l'opérateur souhaite avoir un condensé des alertes. Enfin, l'ordre de généralisation des attributs est basé sur des heuristiques qui sont discutables. Par contre cette approche a un avantage en temps d'exécution, elle est très efficace sur un PC modeste (en moyenne quelques minutes pour grouper 100.000 alertes) (voir Tableau III.1).

Type d'alerte	Port source	Adresse source	Port destination	Adresse destination	Temps	Taille
WWW IIS view source attack	Non Privileged	Internet	80	Ip5	Any Time	54310
FTP SYST command attempt	Non Privileged	Firewall	21	Internet	Any Time	6439
IP fragment attack	n/a	ip6	n/a	Firewall	Workday	4581
TCP SYN host sweep	Non Privileged	Firewall	25	AnyIP	Any Time	761

Tableau III.1. La généralisation d'un fichier log d'IDS [Pietraszek et al, 05]

Formellement, cette méthode consiste : à donner un grand nombre d'alertes L avec les attributs $(T_1, \dots, T_n)$, la généralisation hiérarchie G_i pour chaque attribut d'une alerte, et un paramètre k_{min} . CLARAty produise une séquence de cluster $\{P_1, P_2, \dots, P_n\}$ [Pietraszek et al, 05] [Pietraszek 06].

Donner

- un grand nombre d'alertes L avec les attributs $(T_1, \dots, T_n, \dots)$,
- la généralisation hiérarchie G_i pour chaque attribut d'une alerte,
- les constants k_{min} , MIN_SIZE .

Trouver

- une séquence de cluster décrit les propriétés d'alertes $\{P_1, P_2, \dots, P_n\}$.

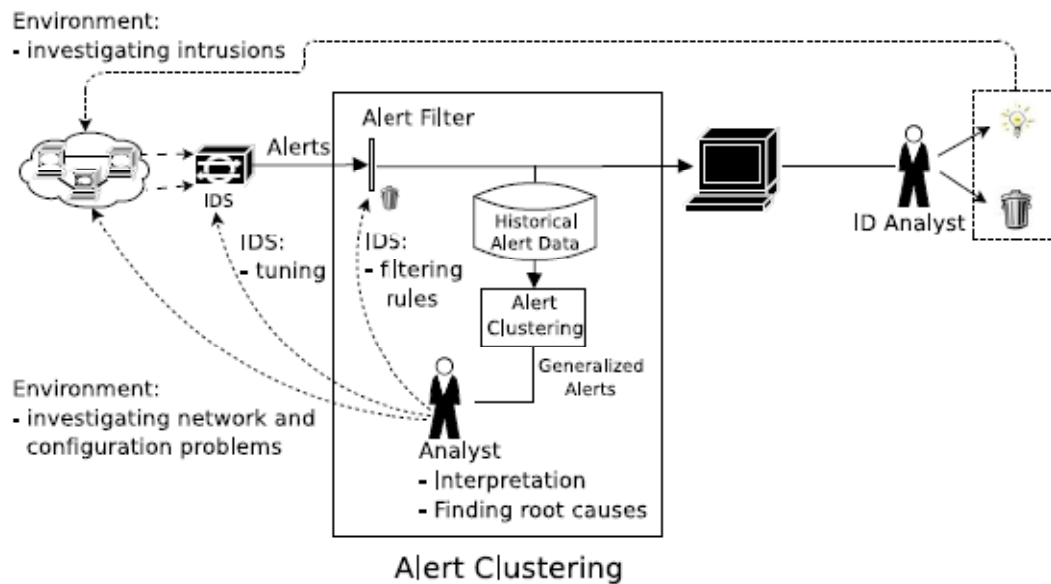


Figure III.8. Data-mining (CLARAty) [Pietraszek et al, 05]

L'algorithme CLARAty est détaillé ci-dessous (Figure III.9):

```

Entrée: ensemble d'alertes  $L$ , généralisation hiérarchique arborescente  $G_i, K_{min}, MIN\_SIZE$ 
Sortie: une séquence de clusters (alertes généralisées)  $P$ .
 $\forall A \in L : Acount = 1;$ 
 $min\_size \leftarrow K \min \cdot \sum_{A \in L} Acount;$ 
 $C \leftarrow \phi;$ 
Tq ( $min\_size \geq MIN\_SIZE$ ) faire
    Sélection de l'attribut  $T_i$  suivant une heuristique a fin de le généraliser;
    Généraliser  $T_i$  (remplacer la valeur de l'attribut  $T_i$  avec celle de son père dans l'
    arborescence  $G_i$  dans l'ensemble des alertes  $L$ );
    Groupe des alertes identiques  $A, A'$ :  $Acount \leftarrow Acount + A'count$ , supprimer  $A'$ ;
    Si  $\exists A \in L : Acount \geq min\_size$  Alors
        /*instaurer le cluster                                     */
         $P \leftarrow P \cup \{A\};$ 
        /*supprimer les alertes de l'ensemble  $L$                  */
         $L \leftarrow L / \{A\};$ 
        /*Annuler la généralisation dans  $L$                        */
         $min\_size \leftarrow K \min \cdot \sum_{A \in L} Acount;$ 
    Fin
Fin

```

Figure III.9. Pseudo-code de l'algorithme CLARAty [Pietraszek 06]

III.3.3 Des méthodes basées sur l'hypothèse "Fréquence d'alertes"

Plusieurs méthodes de réduction d'alertes appartiennent à cette catégorie. Par exemple, dans [Clifton et al, 00] des exploitations fréquentes sont appliquées sur les alertes générées par l'IDS. Lors de la détection des séquences d'alertes qui ont la même destination, elles seront présentées à l'administrateur afin de les classer (c'est-à-dire faux positif ou non). Certaines approches [Alharby et al, 05] et [Manganaris et al, 00] ont le même principe que celle de [Clifton et al, 00], toutefois dans [Viinikka et al, 06] le filtrage d'alertes se fait en fonction du temps d'apparition.

Modèle de séries chronologiques

Le filtrage d'alertes se fait en fonction du temps d'apparition, dans [Viinikka et al, 06] Viinikka et Debar ont introduit un modèle de séries chronologiques pour la détection de faux positif; Ils ont analysés les caractéristiques de temps d'alertes générées par l'IDS: constants,

périodiques et aléatoires; Alors l'hypothèse de base est que les alertes dont les caractéristiques de temps constants et périodiques sont des faux positifs.

Ils ont identifié quatre profils de flux pour les alertes générées par l'IDS, les profils sont définis selon la capacité d'expliquer que le comportement normale ou anormal:

- **Classe A:** La génération d'alerte est presque constante, Le flux contient un peu d'anomalie (comportement normal).
- **Classe B:** Le flux d'alerte contient la périodicité clairement visible, le flux contient quelques anomalies (comportement normal).
- **Classe C:** Le flux d'alerte est moins stable que dans la classe B, il a plus d'anomalies visibles (comportement anormal).
- **Classe D:** Le flux d'alerte est plus ou moins aléatoire (comportement anormal).

III.4 Étude comparative entre les trois méthodes

Le tableau suivant présente une étude comparative entre les trois méthodes (voir Tableau III.2):

	Data mining CLARAty	Machine learning ALAC	Fréquence d'alertes
Type	Temps différé	Temps réel	Temps différé
Fournir	Les alertes non classifiées	Les alertes classifiées	Les alertes non classifiées
Taille des données	Grand	Petit-moyen	Petit-moyen
Base de connaissance	Généralisation hiérarchique	Attribut-valeur	Attribut-valeur en précisant l'attribut temps
Interprétation du résultat	La description du cluster	Les règles de classification	La description des classes
Objectif	Faux positif avec l'identification des causes d'origine	Faux positif et vrai positif	Faux positif

Tableau III.2. Étude comparative entre les trois approches

III.5 Les avantages et les inconvénients de chaque méthode

Le tableau suivant présente les avantages et les inconvénients de chaque approche (voir Tableau III.3):

	Les avantages	Les inconvénients
Des méthodes basées sur les causes d'origine	• Très efficace sur un PC modeste.	• Le choix du seuil. • Les causes d'origine ne sont pas fixées et elles sont difficiles à identifier. • Cette méthode supporte le processus en temps différé • La généralisation hiérarchique qui n'est pas facile à construire
Des méthodes basées sur la classification	• Le système adopte une étude de règle rapide et pertinente. • Différencier les faux positifs et les vrais positifs avec une confiance plus élevée.	• La taille de la base d'apprentissage se développe infiniment.
Des méthodes basées sur l'hypothèse "Fréquence d'alerte"	• Hypothèse facile à implémenter. • Cette approche peut soulager les opérateurs à 90% ou plus.	• Méthode statistique traditionnelle. • Le taux de faux positifs est relativement grand. • Ne fournit pas un diagnostic d'alerte.

Tableau III.3. Avantages et inconvénients de chaque méthode

III.6 Conclusion

Dans ce chapitre nous avons introduit le problème posé par les systèmes de détection d'intrusion en précisant le faux positif. Ensuite, nous avons dressé un état de l'art des différentes approches et algorithmes existants dans la littérature traitant le problème de faux positif.

Par la suite nous avons présenté une étude comparative entre les trois approches afin de présenter les avantages et les inconvénients de chacune d'eux.

Chapitre 4

LES BENCHMARKS

Chapitre 4

Les benchmarks

IV.1 Introduction

Dans ce chapitre, nous aborderons les ensembles de données qui peuvent être utilisées et leurs caractéristiques. Cela donnera une meilleure compréhension de la base de données disponible que nous avons utilisée pour évaluer l'approche proposée. L'évaluation des systèmes de détection d'intrusion est devenue une tâche très difficile, elle exige notamment des connaissances techniques profondes relevant de domaines différents, comme la détection d'intrusion, les réseaux et systèmes, les techniques d'attaques, les techniques de test et d'évaluation, etc.

Une autre difficulté, c'est que les *IDS_s* doivent être évalués en conditions réelles, et surtout en environnement malveillant, donc dans des plateformes de test bien configurées et qui doivent être similaires à la réalité en tenant compte notamment de modes d'utilisation inattendus et parfois même inconnus. Toutes ces considérations rendent la tâche de la construction de données représentatives pour l'évaluation très difficile.

IV.2 Les benchmarks

Depuis plusieurs années, des groupes de recherche ont créé des benchmarks pour évaluer les *IDS_s*. Ces collections permettent de procurer des données d'apprentissage et de tests. De plus, ils offrent la possibilité de comparer la performance de plusieurs *IDS_s* sur une même collection de données. Ces données sont obtenues soit par des simulateurs, soit par des systèmes réels. Néanmoins, dans ce dernier cas il est beaucoup plus complexe de les obtenir, puisqu'elles peuvent représenter des informations privées.

IV.2.1 L'évaluation par test

Plusieurs travaux antérieurs se sont intéressés à cette méthode d'évaluation. Parmi les travaux les plus importants dans ce domaine on trouve [Gadelrab 08].

IV.2.1.1 Le travail réalisé par Puketza

L'un des plus anciens, ce travail utilise un ensemble de scripts pour simuler des cas de test (pour des sessions normales et intrusives) en se basant sur la politique de sécurité de l'organisation [Puketza 96] [Puketza 97]. Il met en œuvre, non seulement des intrusions séquentielles provenant d'une seule session d'attaque, mais aussi des intrusions simultanées provenant de plusieurs sessions. La procédure de test suivie vise trois objectifs majeurs:

- Identification des intrusions;
- Utilisation des ressources;
- Test aux limites ou « *stress testing* ».

IV.2.1.2 Le travail réalisé par IBM Zurich

Ce travail a été réalisé par IBM *Zurich* en vue d'une évaluation comparative de plusieurs *IDS*_s. Pour cela, une plateforme de test a été créée en utilisant plusieurs postes de travail (clients et serveurs) contrôlés par une seule station. Le trafic normal sur le réseau et les événements non intrusifs du système sont générés en utilisant des suites de tests construits par les développeurs du système d'exploitation, tandis que les attaques sont sélectionnées dans une base de données propre à IBM. Le rapport publié indique seulement que quatre *IDS*_s intégrés sur hôte (*HIDS*) ont été testés contre des attaques FTP, mais ne dit rien sur les métriques utilisées ni sur les détails des résultats obtenus.

IV.2.1.3 Le projet sponsorisé par DARPA

DARPA (*Defense Advanced Research Projects Agency*) est une agence du département de la Défense des États-Unis qui est chargé de la recherche et du développement des nouvelles technologies, chargée aussi des projets de recherche militaire. La *DARPA* a été à l'origine spécialisée dans le développement de nombreuses technologies qui ont eu des conséquences considérables actuellement dans le monde entier, en particulier dans les réseaux informatiques (notamment *l'ARPANET* qui a fini par devenir Internet) et le *NLS* qui a été à la

fois le premier système hypertexte et un précurseur important des interfaces graphiques devenues omniprésentes de nos jours [Carlson 01].

La *DARPA* et l'*AFRL* (*Air Force Research Laboratory*) en collaboration avec l'*IST* (*The Information Systems Technology Group*) de laboratoire Lincoln du *MIT*, ont collecté et distribué la première norme standard des données de test pour l'évaluation des *IDS*. Ce travail a débuté en 1998. Le but était de fournir un ensemble significatif de données de test, comprenant un trafic de fond et des activités intrusives. Le trafic de fond était déduit des données statistiques collectées sur le réseau des bases de l'Air Force alors que les attaques étaient générées par des scripts créés spécialement, mais aussi par des scripts collectés à travers des sites spécialisés et des listes de diffusion, ces évaluations mesurent la probabilité de détection et de la probabilité de faux positif pour chaque système sous test.

IV.2.2 L'évaluation analytique

L'évaluation analytique porte sur la définition de méthodes permettant de maîtriser le modèle. Parmi les travaux les plus importants dans ce domaine, on cite celui d'Alessandri [Alessandri 04]. Il a proposé un modèle descriptif des *IDS* et de fournir une documentation aux concepteurs. Cette évaluation est résumée comme suit [Gadelrab 08]:

- Classifier les attaques selon leurs caractéristiques observables par l'*IDS*;
- Décrire le comportement de l'*IDS*, et notamment la manière de collecter et d'analyser les informations;
- Déterminer si un certain type d'attaques sera détectable par l'*IDS* ou pas.

IV.3 L'évaluation du DARPA 98

L'évaluation de *DARPA* 1998 a été conçue pour trouver les points forts et les points faibles des approches existantes afin d'améliorer les performances et de construire une évaluation valable aux systèmes de détection d'intrusion. Le concept était de générer un ensemble d'attaques réalistes, et les intégrer dans des trafics normaux, à fin d'évaluer le taux de faux positif générés par le système de détection d'intrusion, puis améliorer la base de signature ou le comportement des systèmes en corrigeant les faiblesses constatées. L'avantage de l'évaluation est de construire et d'améliorer les performances de l'*IDS*, mais le problème majeur est que c'est une tâche difficile parce que il n'y a pas une mesure standard pour la comparaison (Figure IV.1).

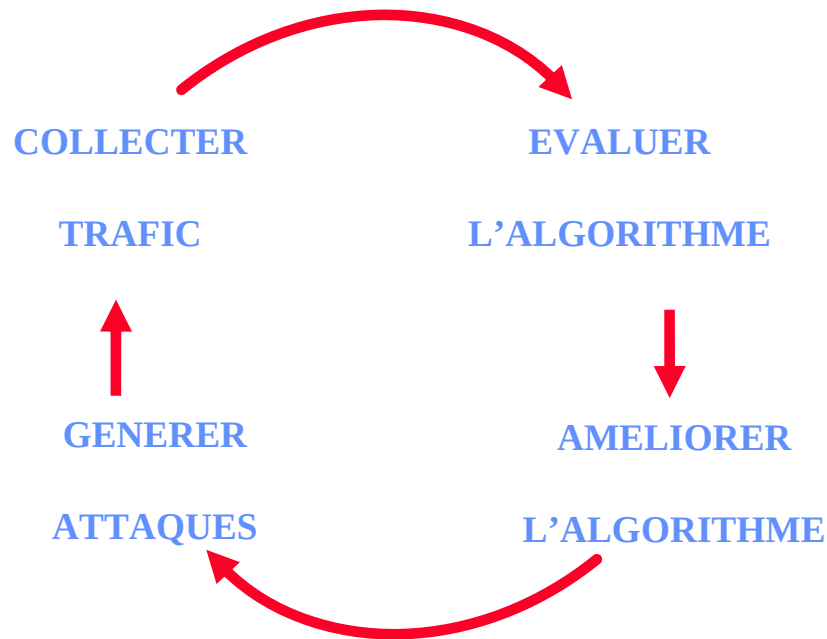


Figure IV.1. Le processus d'évaluation d'IDS [Richard et al, 98]

Il existe deux aspects de *DARPA* 1998: une évaluation en temps différé et une évaluation en temps réel. Les *IDS*_s sont testés avec l'évaluation en temps différé en utilisant le trafic réseau et l'audit des logs sur un réseau de simulation. Ces données ont été mises à la disposition des chercheurs en Février 1998. Le jeu de test *DARPA* 98 utilise environ 300 attaques (classées en 38 types d'attaques). Ces types peuvent être classifiés dans 4 catégories: *Denial of Service* (DOS), *Remote to Local* (R2L), *User to Root* (U2R), *Surveillance or Probe* [Kendall 99].

- **Déni de Service - « Denial-Of-Service (DOS) »:** Ces attaques sont à but purement « destructeur » et sont souvent très simples à mettre en place et donnent une sensation de puissance à l'attaquant, ce qui explique leur fréquence. Un exemple d'attaques *DOS* est le *Smurf* qui provoque un déni de service via des requêtes *d'écho ICMP* manipulées à une adresse de diffusion d'un réseau.
- **Les attaques de type « Remote to Local access » (R2L):** ce type d'attaque essaie d'exploiter la vulnérabilité du système afin de contrôler la machine distante. Comme exemple d'attaque *R2L*, il y a celle qui vise des failles des protocoles *IMAP* (*Internet Message Access Protocol*). Ces protocoles permettent à des utilisateurs d'accéder à leurs comptes de courrier depuis des réseaux internes ou externes.

- **Les attaques de type « User to Root attacks » (U2R):** où l'attaquant essaie d'avoir les droits d'accès à partir d'un poste afin d'accéder au système. Un exemple d'attaques *U2R* est *Rootkit*, qui après avoir obtenu un accès root pour l'intrus, remplace les commandes systèmes afin qu'il puisse revenir quand il le souhaite en tant que root.
- **Reconnaissance - Probing:** ces actions ne sont pas vraiment des attaques puisqu'elles ne sont pas « destructrices » au sens où elles n'empêchent pas une entité de fonctionner correctement, mais permettent d'acquérir des informations parfois cruciales pour mener une attaque de plus grande envergure plus tard. Un exemple d'outils de reconnaissances Probing est *Satan (Security Administrator Tool for Analyzing Networks)*, qui est un analyseur de ports TCP/IP qui recherche sur des hôtes distants les failles de sécurité et les défauts de configuration courants.

Le tableau suivant (Tableau IV.1) montre toutes les attaques disponibles aux données *DARPA 98*, les attaques montrées en gras sont des attaques utilisées sauf pour les données de tests (2 semaines):

	Solaris	SunOS	Linux	Cisco routeur
Denial Of Service (11 types)	Apache2 Back Mailbomb Neptune Ping Of Death Process Table Smurf Syslogd UDP Storm	Apache2 Back Land Mailbomb Neptune Ping of death Process Table Smurf UDP Storm	Apache2 Back Mailbomb Neptune Ping of death Process Table Smurf Teardrop UDP Storm	snmpgetattack
Remote to User (14 types)	dictionary ftp-write guest httptunnel phf xlock xsnoop	dictionary ftp-write guest httptunnel phf xlock xsnoop	dictionary ftp-write guest httptunnel imap named phf sendmail xlock xsnoop	
User to Superuser (7 types)	eject ffbconfig fdformat ps	loadmodule ps	perl xterm	
Surveillance/ Probing (6 types)	ip sweep mscan nmap saint satan	ip sweep mscan nmap saint satan	ip sweep mscan nmap saint satan	ip sweep mscan nmap saint satan

Tableau IV.1. Les attaques disponibles aux données DARPA 98 [Kendall 99]

IV.4 Les éléments principaux de DARPA 1998

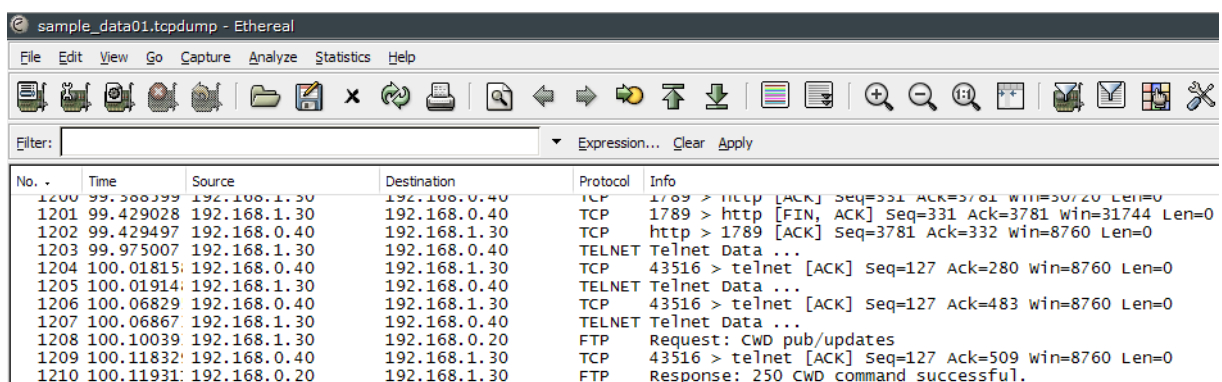
- Toutes les instructions et la description d'attaques de la base d'apprentissage sont disponibles sur le site web du *Lab Lincoln* (ideval.ll.mit.edu).
- Un échantillon de données, Février: 10 Minutes
 - ✓ L'objectif : pour illustrer le format de donnée, le type du trafic utilisé pour l'évaluation, les attaques, et en fin l'étiquetage.
- Plus d'échantillon de données, moi de Mai: quatre heures de donnée d'apprentissage.

- Les données d'apprentissage (du 6 Juillet – 17 Septembre): la base d'apprentissage de Sept semaines
 - ✓ Les attaques étiquetées sont utilisées pour l'apprentissage et le développement.
- Les données de tests, 26 Octobre: une base de tests de deux semaines
 - ✓ Les attaques ne sont pas étiquetées.
- Vingt CD-ROM, soit environ 10 Go.

IV.5 L'ensemble des données utilisées

La première étape consiste de préparer les données lors la phase d'apprentissage, ces données permettent de connaître en entrée les différentes d'attaques et des trafics normaux. Néanmoins, la question est de savoir s'il existe une mesure de la quantité de données nécessaire pour avoir un modèle correct, c'est-à-dire qui respecte un certain taux d'erreur accepté, cette question est encore ouverte à ce jour.

Dans cette partie, nous aborderons les ensembles de données qui peuvent être utilisées dans l'évaluation de l'approche proposée, dans ce cas nous avons utilisé comme une base d'apprentissage les jours de la septième semaine (lundi, mardi, mercredi, jeudi), le trafic (*dump*) est documenté dans un fichier (*tcpdump.list*), ce *dump* est utilisé par *Snort* en temps différé pour générer les alertes, pour faciliter la lecture du trafic nous avons utilisé le filtre de *l'Ethereal*. *Ethereal* est un analyseur de protocole gratuit pour Windows, Unix et ses dérivés. Il permet d'examiner des trames à partir d'un fichier (*dump*) ou directement en les capturant sur le réseau. En outre, le logiciel possède des fonctionnalités très utiles comme les filtres de capture et d'affichage et la reconstitution du flux d'une session TCP. De plus, le nombre de protocoles reconnus par l'analyseur est très élevé (Figure IV.2).



No.	Time	Source	Destination	Protocol	Info
1200	99.586399	192.168.1.30	192.168.0.40	TCP	1789 > telnet [ACK] Seq=331 Ack=3781 win=8760 Len=0
1201	99.429028	192.168.1.30	192.168.0.40	TCP	1789 > http [FIN, ACK] Seq=331 Ack=3781 win=31744 Len=0
1202	99.429497	192.168.1.30	192.168.0.40	TCP	http > 1789 [ACK] Seq=3781 Ack=332 win=8760 Len=0
1203	99.975007	192.168.1.30	192.168.0.40	TELNET	Telnet Data ...
1204	100.01815	192.168.1.30	192.168.1.30	TCP	43516 > telnet [ACK] Seq=127 Ack=280 win=8760 Len=0
1205	100.01914	192.168.1.30	192.168.0.40	TELNET	Telnet Data ...
1206	100.06829	192.168.1.30	192.168.1.30	TCP	43516 > telnet [ACK] Seq=127 Ack=483 win=8760 Len=0
1207	100.06867	192.168.1.30	192.168.0.40	TELNET	Telnet Data ...
1208	100.10039	192.168.1.30	192.168.0.20	FTP	Request: CWD pub/updates
1209	100.11832	192.168.1.30	192.168.1.30	TCP	43516 > telnet [ACK] Seq=127 Ack=509 win=8760 Len=0
1210	100.11931	192.168.0.20	192.168.1.30	FTP	Response: 250 CWD command successful.

Figure IV.2. La capture des trames sur Ethereal

IV.5.1 Le système de détection d'intrusion utilisé

Nous nous baserons sur le *NIDS (Network-Based IDS)* appelé *Snort* [Snort 03]. Ce choix a été dicté principalement par le fait que *Snort* est un logiciel open-source ; vu que la plupart des NIDS commerciaux, tels que *Symantec NetProwler*, *Cisco Secure IDS*, *NFR NIDS*, et *Niksun NetDetector* sont très chers. Par ailleurs, *Snort* est déjà reconnu comme étant un bon produit ; il est de plus en plus utilisé dans les entreprises.

Snort peut être configuré pour fonctionner dans l'un des trois modes suivants :

- *Sniffeur* : Dans ce mode, Snort lit en continu les paquets circulant sur le réseau, puis affiche leur contenu sur l'écran;
- *Logueur ou enregistreur de paquets* : dans ce mode, Snort enregistre les paquets lus dans des fichiers de logs;
- *IDS ou Détecteur d'intrusion* : Dans ce mode, Snort applique un ensemble de règles sur les paquets lus pour y découvrir des signes d'attaques intrusives.

Dans notre cas nous avons utilisé le mode *IDS* ou Détecteur d'intrusion, ce dernier est utilisé en temps différé pour générer les alertes.

Snort utilise un langage de description de règles très flexible qui combine les bienfaits des méthodes d'inspection basées sur la signature, le protocole et/ou les anomalies pour détecter les intrusions. Une règle Snort est composée de deux parties distinctes : *le header* et *les options*. Le header fournit, d'un côté, les champs de base relatifs aux paquets concernés par la règle (le protocole ainsi que les adresses IP et les ports sources et destination des paquets) et spécifie, d'un autre côté, l'action que doit entreprendre Snort lorsque il rencontre ces paquets.

Les trois actions principales supportées par Snort.

- *Alert* : génère un message d'alerte puis enregistre le paquet dans un fichier de logs.
- *log* : journalise le paquet dans un fichier de logs.
- *pass* : ignore le paquet.

Dans notre cas nous avons traité les alertes enregistrées dans le fichier de *log*.

Les options sont spécifiées entre parenthèses. Elles permettent de préciser le message d'alerte et d'affiner l'analyse, en précisant les valeurs observées dans certains champs d'entêtes ou dans les données des paquets traités. Voici un exemple d'une règle Snort (la Figure IV.3) :

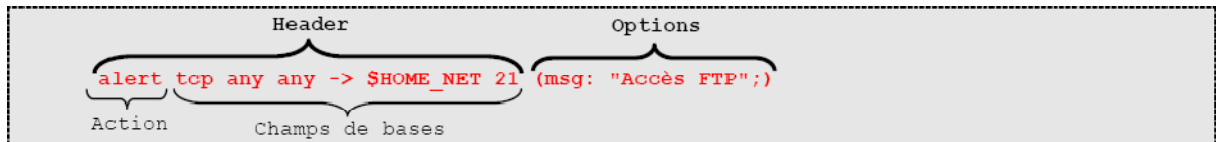


Figure IV.3. Une règle de Snort

Cette règle permet de générer le message d'alerte « *accès FTP* » lorsqu'un flux *TCP* à destination du port 21 d'une machine du réseau local est détecté. Pour tester cette règle, nous l'incluons dans le fichier « *\Snort\rules\local.rules* », puis nous lançons Snort avec la commande « *Snort -c c:\snort\etc\snort.conf -l c:\snort\log* ». Voici l'alerte obtenue par Snort lors de l'établissement d'une connexion *FTP* à l'*IDS* depuis le poste de travail.

```
[**] [1:0:0] Accès FTP [**]
[Priority: 0]
07/11-04:07:39.534561 192.168.10.11:56059 -> 192.168.10.2:21
TCP TTL:64 TOS:0x10 ID:65454 IpLen:20 DgmLen:65 DF
***AP*** Seq: 0xE275625F Ack: 0x10BC9A5 Win: 0x5B4 TcpLen: 32
TCP Options (3) => NOP NOP TS: 1613301 3940089
```

Figure IV.4. Un message d'alerte généré par Snort

Cette sortie d'écran indique que *Snort* a produit un message d'alerte (consignés dans le fichier *\Snort\log\alerte.ids*) (voir la Figure IV.4). Ce message fournit le détail des champs du paquet à l'origine de l'alerte. Nous remarquons que ce paquet comporte une signature identique (protocole, adresse IP et port destination) à celle de la règle *Snort* définie préalablement.

Notons que lorsque *Snort* est lancé en tant que détecteur d'intrusion, tous les messages d'alertes qu'il produit sont consignés dans un fichier nommé *c:\snort\log*. Les paquets à l'origine de ces alertes sont également consignés dans des fichiers de logs nommés *c:\snort\log\snort.log.x* (x est un numéro d'ordre qui distingue les fichiers de logs entre eux).

IV.5.2 La représentation des alertes

Les alertes $\{A_1, \dots, A_n\}$ générées par *Snort* sont stockées dans un fichier de *log*, chaque alerte peut être facilement représentée comme des attributs $(T_1, T_2, \dots, T_n)$. Par exemple l'adresse d'un attaquant, le type d'alerte et la date, dans notre cas nous avons reconfiguré *Snort* pour générer les messages d'alertes en format *CSV* (*Comma Separated Value*) (voir la Figure IV.5), le module *CSV* transmet sa sortie dans un fichier dont le

délimiteur est la virgule. Le fichier *alerte.csv* peut être facilement importé dans d'autres bases de données et feuilles de calcul, les principaux attributs d'une alerte sont :

Timestamp est en général un attribut numérique (mois, jours, heures, minutes, secondes, millisecondes) exemple : *Timestamp* = 01/23-23:55:59.279341.

Message d'alerte ou la signature est un attribut catégorique pour déterminer le type d'attaque détecté par l'*IDS*, par exemple, *IIS exploit*, *PORTSCAN*. Généralement les signatures peuvent être regroupées en groupes selon le type d'attaque similaire par exemple *exploite un serveur web*.

Protocole est un attribut catégorique par exemple la détermination du protocole, *TCP*, *UDP* ou *ICMP*.

Adresse de la source est un attribut catégorique décrivant l'attaquant, L'adresse source est généralement une adresse *IP* de 32 bits, mais dans le cas des protocoles couramment utilisés comme *TCP* et *UDP*, l'adresse source est une paire (*Ip*, *s-port*), où *IP* est une adresse *IP* de 32 bits et *s-port* est le port source 16 bits, l'adresse est généralement représentée comme deux attributs distincts, par exemple : (adresse source, port source) = 192.168.1.30,1784.

Adresse de destination est un attribut catégorique décrivant la victime, tandis que le port de destination détermine le service de l'ordinateur (par exemple, 22 est un port *SSH*, 80 est un port *HTTP*), par exemple : (@destination, port destination) =192.168.0.40,80.

Payload (charge utile) est un attribut de texte libre contenant des données et qui permet à l'analyste de mieux déterminer la nature du problème, en inspectant le fragment de trafic réel, ce qui a déclenché l'alerte.

Dans notre cas nous avons utilisés tous les attributs cités ci-dessus en général 27 attributs.

IV.5.3 L'étiquetage des alertes

L'approche proposée suppose que les alertes sont marquées par l'analyste. Rappelons que, dans un premier temps, nous avons généré les alertes par *Snort* en temps différé. Dans la deuxième étape, nous avons étiqueté automatiquement les alertes en utilisant le fichier *tcpdump.list* (le trafic documenté du *DARPA 98*). Pour faire l'étiquetage, nous avons utilisé une approche automatique, c'est-à-dire marquer automatiquement les alertes à l'aide d'un fichier *tcpdump.list*, ce dernier considère que toutes les alertes réunissent les critères suivants

relatifs à une attaque : (i) correspond à l'adresse IP source, (ii) correspond à l'adresse IP destination et (iii) la durée de temps dans lequel l'attaque a eu lieu. Pendant l'examen manuel des alertes nous avons constaté que dans de nombreux cas, la classification est ambiguë (par exemple, une alerte *PING* bénigne peut être ainsi classée comme malveillante si elle est envoyée à l'hôte attaqué), cela peut introduire une erreur dans la classification et par conséquent réduire la précision du classifieur.

La Figure IV.5 ci-dessous présente deux alertes étiquetées en format *CSV* (**V** : vrais positif ou **F** : faux positif) :

```
01/23-23:55:59.279341 ,1,366,7,ICMP PING
*NIX,ICMP,192.168.1.30,,192.168.0.11,,0:60:97:79:AF:BE,0:0:C:4:41:
BC,0x62,,,,,64,0,11196,84,20,8,0,26376,0,F

01/23-23:56:45.930298 ,1,1762,5,WEB-CGI phf arbitrary
command execution attempt,TCP,192.168.1.30,1784,192.168.0.40,80,
0:60:97:79:AF:BE,0:0:C:4:41:BC,0x6A,***AP***,0x770813F8,0xC1F76CC8
,,0x7C00,64,0,13009,92,20,,,,,V
```

Figure IV.5. Représentation d'une alerte en format CSV

IV.6 Conclusion

Dans ce chapitre nous avons introduit les ensembles de données qui peuvent être utilisées pour évaluer les systèmes de détection d'intrusion de façon générale en précisant les données *DARPA*, ainsi nous avons dressé en détail les caractéristiques de *DARPA 98*. Ensuite, nous avons présenté en bref l'IDS (*Snort*) utilisé pour évaluer l'approche proposée. Par la suite, nous avons présenté le format d'alerte ainsi que l'étiquetage utilisé pour l'évaluation.

Chapitre 5

L'APPROCHE PROPOSÉE POUR LA DÉTECTION DE FAUX POSITIF

Chapitre 5

L'approche proposée pour la détection de faux positif

V.1 Introduction

Les règles de prédiction sont les modèles de la forme:

SI < conjonction de conditions > **ALORS** < conclusion >.

Les conditions concernent la valeur des attributs individuels, tandis que la conclusion donne une prédiction de la valeur de la classe cible (variable). Les règles de prédiction peuvent être ordonnées ou non ordonnées. Les règles non ordonnées sont considérées de façon indépendante et plusieurs d'entre elles peuvent vérifier une nouvelle instance que nous avons besoin de classer. Un conflit se produit si par exemple deux règles vérifient une même instance. Par contre les règles ordonnées sont sous forme d'une liste appelée '*decision list*'. Les règles sont considérées de haut en bas de la liste. Le principe de *decision list* est que les règles sont évaluées en commençant par la première et en parcourant les règles séquentiellement jusqu'à ce qu'un antécédent soit vérifié. Si aucune règle n'est vérifiée, on trouve généralement une règle par défaut qui attribue une classe aux instances restantes. Dans ce qui suit, nous donnerons la définition d'une *decision list*.

V.2 Définition

Formellement, *decision list* est un moyen de fournir une représentation booléenne. Chaque fonction booléenne peut être représentée par une formule *DNF* (*Disjunctive Normal Form*), et toutes les fonctions booléennes peuvent également être représentées par une *decision list*.

Supposons K tout ensemble de fonctions booléennes sur $\{0,1\}^n$, avec n fixé. Étant donné une instance y , nous avons d'abord évalué $f_1(y)$, $f_1 \in K$. Si $f_1(y)=1$, nous attribuons une valeur fixe c_1 (soit 0 ou 1) à $f(y)$, sinon nous évaluons $f_2(y)$, $f_2 \in K$, et si $f_2(y)=1$ alors $f(y)=c_2$, sinon nous évaluons $f_3(y)$, et ainsi de suite. Si y ne vérifie aucune f_i , alors $f(y)$ prend la valeur 0 par défaut [Anthony 05].

L'évaluation d'une *decision list* peut donc être considérée comme une séquence de 'si alors sinon' :

Si $f_1(y)=1$ alors $f(y)=c_1$

Sinon si $f_2(y)=1$ alors $f(y)=c_2$

...

...

Sinon si $f_r(y)=1$ alors $f(y)=c_r$

Else $f(y)=0$.

V.2.1 Concepts fondamentaux

Dans le contexte de la classification, la connaissance extraite ou *decision list* se présente généralement sous la forme d'un ensemble de règles. Ces règles ont généralement la forme [Parpinelli et al, 02]:

- o **SI** *antécédent* **ALORS** *conséquent*

L'antécédent est constitué de conditions reliées par l'opérateur logique ET. Chaque condition appelée un terme est formée par un triplet < attribut, opérateur, valeur >.

- o Exemple de terme : *protocole = ICMP*

Où "*protocole*" est un attribut de la table alerte (voir Tableau V.1), le signe "=" est l'opérateur de comparaison et "*ICMP*" est la valeur de comparaison.

- o Exemple d'antécédent : *proto = ICMP ET id > 3910*

Le conséquent est constitué de la classe prédite par la règle qui peut être dans notre cas, les classes *fausse* ou *vrai*.

- o Exemple de règle : **SI** *proto = ICMP ET id > 3910* **ALORS** *Fausse*

Cette règle prédit que si une alerte vérifie chaque condition (terme) de l'antécédent, alors il est probable que cette alerte soit *fausse*.

L'ensemble des règles forme un modèle de classification ou simplement une *decision list*.

- o Exemple de *decision list* :

SI *sig_generator <= 1 ET icmpid = ?* **ALORS** *Fausse*

SI proto = ICMP **ET** id > 3910 **ALORS** Fausse

SI tos > 0 **ALORS** Fausse

SI dgmlen <= 107 **ALORS** Vrai

SINON Fausse

On associe généralement à chaque règle plusieurs mesures de qualité [Witten et al, 05]. On retrouve la notion de couverture qui est le cardinal de l'ensemble des instances qui vérifient l'antécédent. On retrouve aussi la notion de précision qui est le cardinal des instances qui vérifient l'antécédent mais aussi pour lesquelles l'attribut de classe vérifie le conséquent. En effet plus une règle est complexe et moins elle est compréhensible. Ces mesures s'appliquent aussi à un ensemble de règles tel qu'un modèle.

V.2.2 La présentation des alertes

Le tableau suivant montre l'importation du module *CSV* dans une base de données, les alertes sont organisées en tableau (voir Tableau V.1) qui contiennent un ensemble d'instances.

	attribut N° 1	attribut N° 2	attribut de classe		
i n s t a n c e	sig_gen	msg	proto	.	Classe
	1	Accès FTP	TCP	.	<i>F</i>
	1	ICMP Echo...	ICMP	.	<i>V</i>
	125	(ftp_telnet)...	TCP	.	<i>F</i>
	...	...		.	
	1	WEB-CGI...	TCP	.	<i>F</i>
	1	INFO web bug..	TCP	.	<i>F</i>

Tableau V.1. La présentation d'un ensemble d'alertes

Chaque ligne de la matrice correspond à une instance qui est associée à un cas expérimental à analyser. Chaque colonne de la matrice représente toutes les valeurs possibles associées à un attribut. Par exemple, un attribut : soit *sig_generator*, le message d'alerte (*msg*), protocole (*proto*) etc... Un attribut peut être numérique ou nominal. Un attribut nominal prend ses valeurs à partir d'un ensemble de choix discret. Enfin un attribut particulier appelé l'attribut de classe, est utilisé comme attribut de classification. L'ensemble des valeurs

de l'attribut de classe représente l'ensemble des catégories que l'on attribue à chaque instance. Dans notre cas, cela peut être par exemple des classes telles que : fausse alerte ou vrai alerte.

V.3 L'architecture de l'approche proposée

Dans le contexte de la classification, la connaissance extraite ou *decision list* se présente généralement sous la forme d'un ensemble de règles. La classification est définie comme un processus d'extraction de connaissances à partir d'une base de données (des alertes classifiées) [Hornick et al, 07]. Le but de la classification est d'assigner à chaque instance (alerte), une classe choisie à partir d'un ensemble de classes prédéfinies, en fonction de la valeur de certains attributs [Parpinelli et al, 02].

On parle de données d'apprentissage (des alertes classifiées) si toutes les données contiennent des valeurs pour l'attribut de classe (voir Tableau V.1). On parle de données de test (des alertes non classifiées) si ces dernières doivent être classées et donc ne possèdent pas de classe ou en d'autres termes possèdent comme valeur d'attribut de classe, la valeur indéterminée (voir Figure V.1 et Figure V.2) dans notre cas nous avons étiqueté les données de tests pour évaluer la précision et le taux de faux positif et négatif.

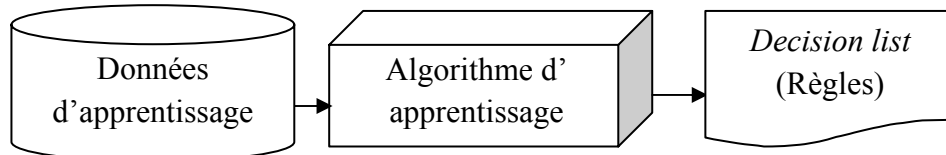


Figure V.1. Illustration du processus d'apprentissage

Toutefois les règles extraites ne sont utiles que si l'ensemble de règles n'est pas trop complexe et si la précision de leurs prédictions est suffisamment élevée.

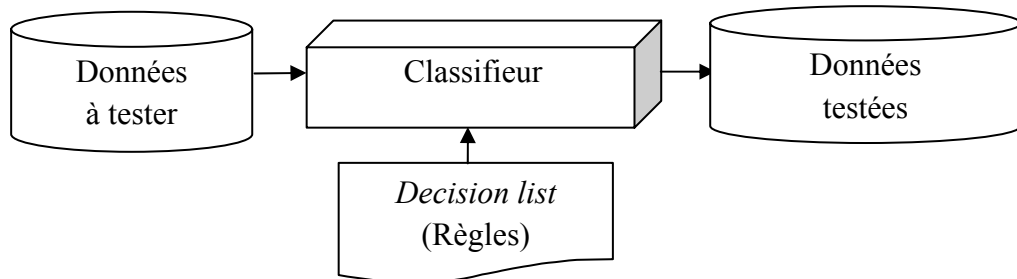


Figure V.2. Illustration du processus de classification

Donc l'approche proposée est basée sur deux phases, la première consiste à extraire des règles généralisées à partir d'une base d'apprentissage, la deuxième est un processus de classification basée sur la notion de *decision list*. Ce processus utilise les règles extraites dans la première phase pour classifier des alertes brutes (voir Figure V.3).

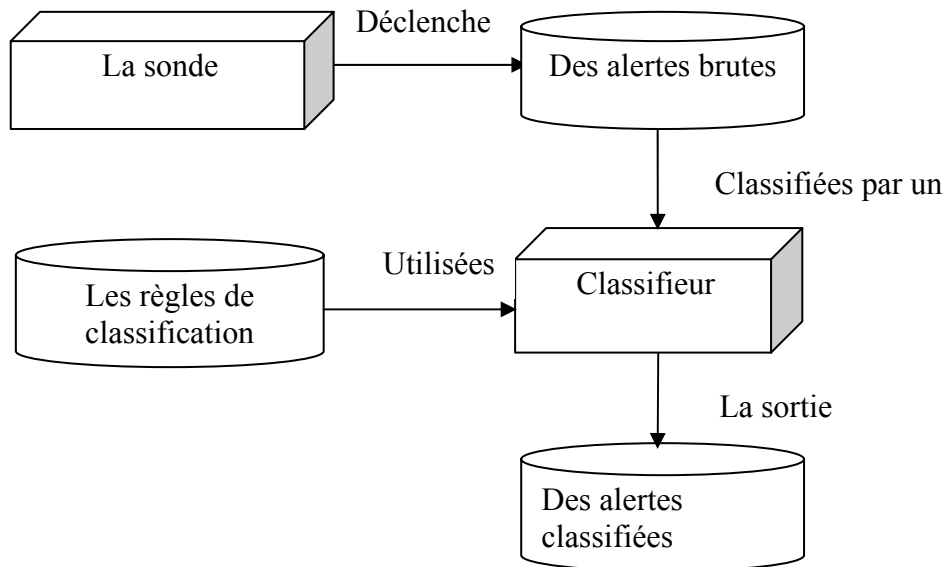


Figure V.3. Le schéma du système

V.3.1 La technique d'apprentissage utilisée

Jusqu'à présent, nous avons mis l'accent sur l'architecture générale de l'approche proposée. Dans cette partie, nous mettons l'accent sur l'outil d'apprentissage utilisé. On peut formuler les exigences suivantes pour la technique d'apprentissage :

- Apprendre à partir d'une base d'apprentissage de la septième semaine comme nous avons déjà dit dans le chapitre précédent (lundi, mardi, mercredi, jeudi).
- Générer les alertes par un système de détection d'intrusion (*Snort*) [Snort 03].
- L'étiquetage des alertes.
- Intégrer le classifieur *PART*, ce dernier est un système de production de règles de classification à partir d'une base de données. Nous avons choisi le classifieur *PART* parce que c'est un moyen rapide de production de règle. De plus, les règles *PART* satisfont trois conditions très souhaitables : une précision et compréhension élevée, aussi les règles générées par *PART* basées sur la technique de *decision list*.
- Être en mesure d'évaluer la précision de la classification.

V.3.2 Le codage de règle

Une règle est représentée sous la forme d'un tableau. Chaque attribut est codé sur trois positions (voir la Figure V.4). Le premier indique si cet attribut apparaît dans la règle ou pas. A cet effet il est représenté par un attribut booléen qui peut prendre les valeurs vraie ou faux. Le deuxième attribut identifie l'opérateur de comparaison utilisé dans la condition. Le troisième contient la valeur à laquelle l'attribut est comparé. Le type d'attribut peut être binaire, catégorique ou numérique.

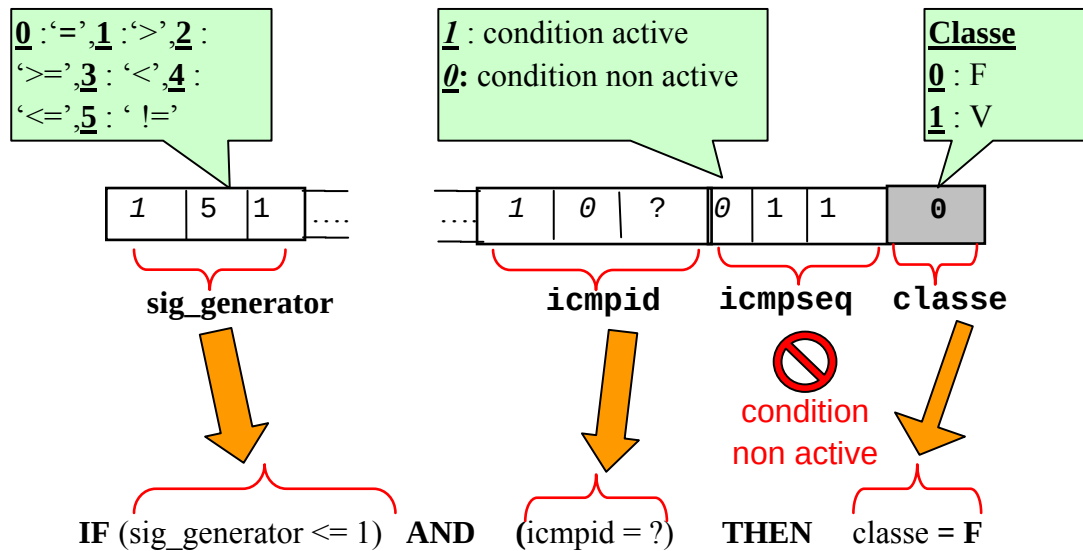


Figure V.4. Exemple de codage de règles

La précision est mesurée en comparant le résultat de la classification obtenu en utilisant la règle courante sur les données d'apprentissage. La précision est égale au nombre d'instances correctement classées divisé par le nombre total d'instances couvertes par cette règle.

V.3.3 L'algorithme de l'approche

Pour chaque instance, une règle est choisie. Pour ce faire, une fonction de score est utilisée pour choisir la règle ayant le score maximum. Généralement les règles obtenues par l'extracteur de règle *PART* ont un score décroissant, en d'autre terme la première règle possède un score maximum ensuite la deuxième règle et ainsi de suite jusqu'à la dernière qui a un score minimum. Si aucune règle n'est vérifiée, on trouve une règle par défaut qui attribue une classe aux instances restantes. Le processus se répète jusqu'à ce que toutes les instances soient classifiées (voir la Figure V.5).

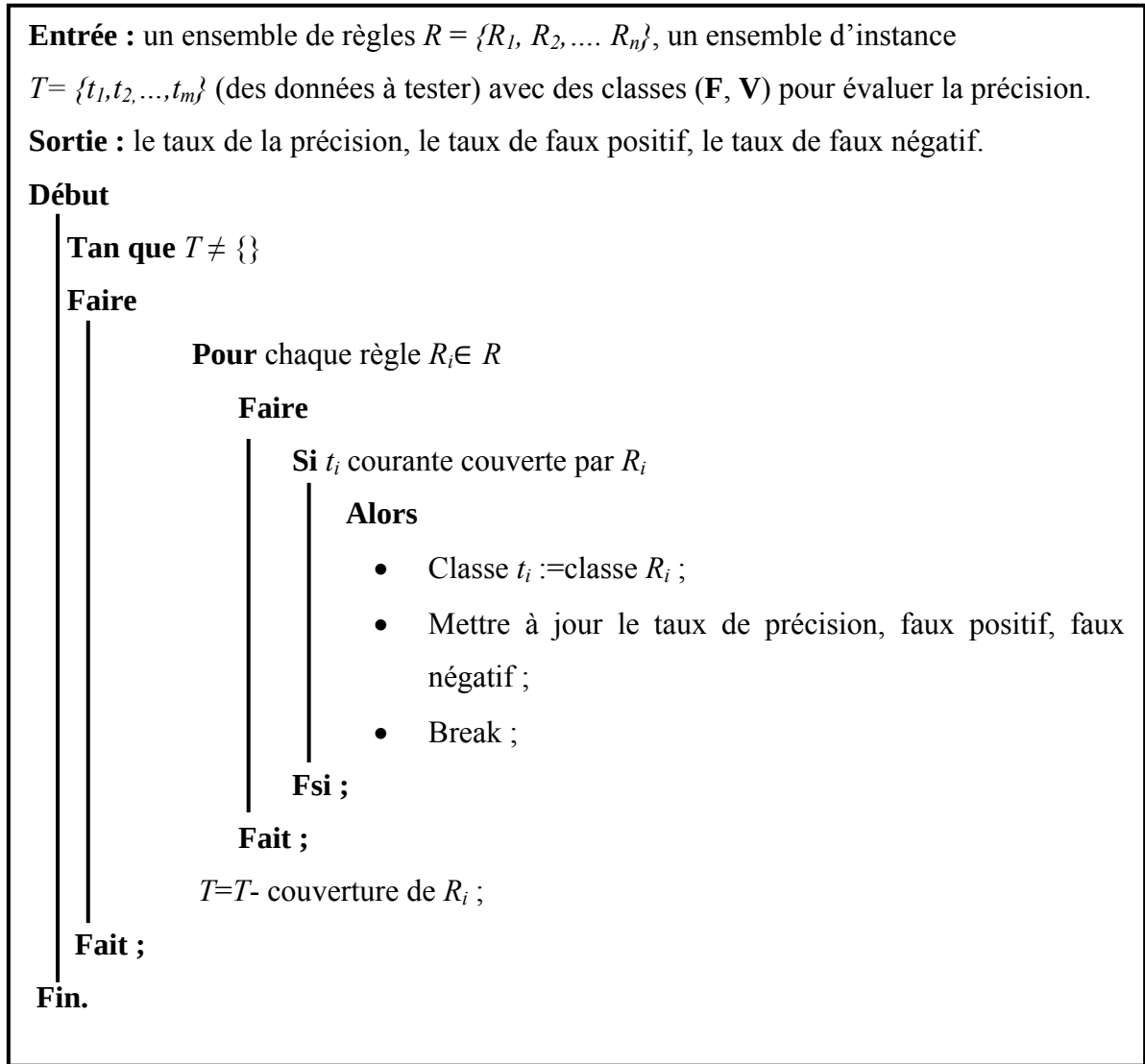


Figure V.5. L'algorithme de l'approche

Pour toutes les règles R et toutes instances T , nous définissons les notations suivantes:

$Inst_R$: Toutes les instances couvertes par R ,

$$Inst_R = \{T | T \text{ couvertes par } R\}$$

$Inst_R^+$: Toutes les instances correctement classées par R ,

$$Inst_R^+ = \{T | T \text{ correctement classées par } R\}$$

$Inst_R^-$: Toutes les instances non correctement classées par R ,

$$Inst_R^- = \{T | T \text{ incorrectement classées par } R\}$$

Précision (RT) : La précision est une valeur numérique rattachée à la classification, ce qui représente la façon dont elle est susceptible d'être correcte.

$$Précision(RT) = \frac{Inst_R^+}{Inst_R} = \frac{Inst_R^+}{Inst_R^+ + Inst_R^-}$$

V.4 Conclusion

Dans ce chapitre, nous avons détaillé la notion de *decision list* dont le principe est que les règles sont évaluées en commençant par la première et en parcourant les règles séquentiellement jusqu'à ce qu'un antécédent soit vérifié. Si aucune règle n'est vérifiée, on trouve généralement une règle par défaut qui attribue une classe aux instances restantes, ensuite nous avons présenté les données qui peuvent être utilisées au niveau de l'apprentissage.

Par la suite, nous avons présenté l'architecture de l'approche proposée et la technique d'apprentissage utilisée, ensuite nous avons montré le codage des règles utilisées par le classifieur.

Chapitre 6

LES TESTS

Chapitre 6

Les tests

VI.1 Introduction

Dans ce chapitre, nous présentons les simulations menées, afin d'évaluer la proposition décrite précédemment. Il décrit aussi les résultats expérimentaux réalisés. Nous commençons par présenter l'implémentation dans laquelle nous décrivons l'environnement de développement et l'outil utilisé. Puis, nous présentons les résultats obtenus ainsi que l'analyse faite.

VI.2 Outils utilisés

Nous avons utilisé un langage de programmation, des bibliothèques externes et des bases de données.

VI.2.1 Le langage de programmation utilisé

Nous avons utilisé le langage de programmation JAVA (Eclipse version 1.0) qui est un langage orienté objet, il a été développé par Sun Microsystems 1991, conçu sur le modèle du langage C++, simple, concis et portable sur toutes les plateformes et systèmes d'exploitation.

VI.2.2 La bibliothèque externe

Nous avons utilisé *Weka* (**W**aikato **E**nvironment for **K**nowledge **A**nalysis) [Witten et al, 05] qui est un logiciel écrit en java développé à l'Université de Waikato en Nouvelle Zélande. Weka est un ensemble d'outils permettant de manipuler et d'analyser des fichiers de données, implémentant la plupart des algorithmes d'intelligence artificielle, entre autres, les arbres de décision et les réseaux de neurones. Il se compose principalement :

- ✓ De classes Java permettant de charger et de manipuler les données, dans laquelle nous avons chargé le fichier *alerte.csv* et de l'importer dans la base de données de Weka.
- ✓ De classes qui implémentent pour les principaux algorithmes de classification supervisée ou non supervisée tels que *J48*, *Id3*, *Table de décision*, *OneR*, *PART*, dans notre cas nous avons utilisé ce dernier.
- ✓ D'outils de sélection d'attributs, de statistiques sur ces attributs.
- ✓ De classes permettant de visualiser les résultats, dans notre cas les règles de *decision list* générées par *PART*.

VI.3 Les benchmarks DARPA 98

Les données *DARPA* que nous avons utilisées contiennent un grand nombre de données correspondant à l'activité d'un réseau réel sélectionnées par l'agence du département de la Défense des États-Unis en collaboration avec *l'IST (The Information Systems Technology Group)* du laboratoire Lincoln (*MIT*). Les données sont mises à la disposition de la communauté des chercheurs en sécurité réseau pour évaluer les systèmes de détection d'intrusion. Les bases de données *DARPA* sont largement utilisées dans le monde et considérées comme une référence, d'où l'importance de les utiliser pour construire et améliorer les performances d'un système de protection informatique [Richard et al, 98].

VI.3.1 Les données d'apprentissage

Pour la base d'apprentissage que nous avons sélectionné, nous avons utilisé quatre jours de la septième semaine (lundi, mardi, mercredi, jeudi) afin de les assembler en une seule base d'apprentissage.

- Cette base de données a été utilisée par le système de détection d'intrusion (*Snort*) afin de générer les alertes en temps différé : 36632 alertes générées.
- L'étiquetage d'alertes en deux classes, les alertes étiquetées en **V** =36302, les alertes étiquetées en **F** =330).
- Le nombre d'attributs=28.

Le Tableau VI.1 résume la base d'apprentissage utilisée.

Les données d'apprentissage	Nombre d'instances	Instances F	Instances V
Lundi	9822	9802	20
Mardi	9356	9350	6
Mercredi	7964	7660	304
Jeudi	9490	9490	0
BD utilisée	36632	36302	330

Tableau VI.1. Résumé sur la base de données utilisée

VI.3.2 Le choix du classifieur *PART*

Nous avons pris la base de données illustrée précédemment puis nous avons exécuté le classifieur pour chaque taille (voir Tableau VI.2).

Nous avons choisi le classifieur *PART* parce que c'est un moyen rapide de production de règles de classification. Les règles *PART* satisfont trois conditions très souhaitables: une précision et une compréhension élevées, aussi les règles générées par *PART* basées sur la technique de *decision list*.

La base d'apprentissage	La précision
Lundi	99,88%
Mardi	99,98%
Mercredi	99,92%
BD utilisée	99,95%

Tableau VI.2. Evaluer la précision pour chaque jour

Voici ci-dessous l'ensemble de règles obtenues selon la base d'apprentissage utilisée (voir Figure VI.1).

<i>SI sig_generator <= 1 ET icmpid = ? ALORS F</i>	<i>couverture : 35065 instances</i>
<i>SI proto = ICMP ET id > 3910 ALORS F</i>	<i>couverture : 762 instances</i>
<i>SI tos > 0 ALORS F</i>	<i>couverture : 461 instances</i>
<i>SI dgmlen <= 107 ALORS V</i>	<i>couverture : 315 instances</i>
<i>SINON F</i>	<i>couverture : 17 instances</i>

Figure VI.1. Les règles générées par la base de données utilisée

VI.3.3 Les mesures utilisées pour évaluer la qualité de l'approche

Dans nos tests, nous avons évalué la qualité de l'approche proposée construite en utilisant les mesures suivantes:

- une mesure appelée précision qui nous donne le pourcentage des instances bien classées.
- Une mesure appelée faux positif. Toute alerte doit correspondre à une intrusion effective. Lorsqu'un système de détection d'intrusions génère une alerte qui n'a pas lieu d'être, cette alerte est qualifiée de faux positif. La pertinence (ou crédibilité) d'un Analyseur est liée à son taux de faux positifs qui représente alors le pourcentage de faux positif.
- Une mesure appelée faux négatif. Idéalement, toute intrusion doit donner lieu à une alerte. Une intrusion non détectée, c'est-à-dire n'ayant pas généré d'alerte, constitue alors un faux négatif. La fiabilité (ou couverture) d'un Analyseur est liée à son taux de faux négatifs qui représente alors le pourcentage d'intrusions non détectées, ce taux devant être le plus bas possible.

VI.4 Résultats de l'expérimentation

Dans cette section, nous allons présenter les résultats de l'expérimentation faite pour valider l'approche proposée. L'objectif de l'expérimentation est de tester l'efficacité de la méthode de détection de faux positif basée sur la notion de *decision list*, l'expérimentation a été faite sur les données de test *DARPA 1998*.

Dans cette expérimentation, nous avons rejoué les données de test dans une station de test surveillée par le système de détection *Snort* [Snort 03]. Le choix de ce détecteur a été fait, parce qu'il est le plus populaire dans le monde et ouvert, et aussi pour la disponibilité de la documentation. Nous avons changé la configuration par défaut de Snort pour qu'il reporte les

messages d'alerte en format *CSV*. Nous avons conservé les messages d'alertes donnés par *Snort* pour étiqueter les alertes dans le but d'évaluer les trois mesures définies précédemment.

Le premier but de cette expérimentation est de tester l'efficacité de la technique de *decision list* utilisée ; pour tester l'efficacité de ce dernier il faut évaluer la précision de l'approche. Les tableaux (Tableau VI.3 et Tableau VI.4) montrent un taux de précision performant qui peut atteindre 86,10%.

Le deuxième but est d'évaluer le taux de réduction du faux positif. Nous remarquons d'après les deux graphes ci-dessous (Figure VI.3 et Figure VI.5) que l'approche proposée donne un taux de faux positif minimal par rapport à celui du *IDS (Snort)*, le taux de faux positif obtenu peut atteindre 12,43%.

Le troisième but est d'évaluer le taux de faux négatif. Au début nous considérons que *Snort* détecte toutes les attaques malgré que ce n'est pas le cas. Dans ce cas le taux de faux négatif est égal à 0%. Nous remarquons d'après les deux tableaux ci-dessous (Tableau VI.3 et Tableau VI.4) que l'approche proposée donne un taux de faux négatif minimal en moyenne 0,80%.

	Précision	Faux positif	Faux négatif
1000 alertes	81,40%	18,60%	0%
2000 alertes	84,80%	14,25%	0,95%
3000 alertes	86,10%	12,43%	1,46%
Moyenne	84,10%	15,09%	0,80%

Tableau VI.3. Résultats obtenus par un ensemble alertes de la base de tests jeudi

Ce tableau peut être schématisé dans la Figure VI.2. Le graphe suivant montre la variation de précision, faux positif et faux négatif suivant le nombre d'alertes générées par la base de test de jour jeudi.

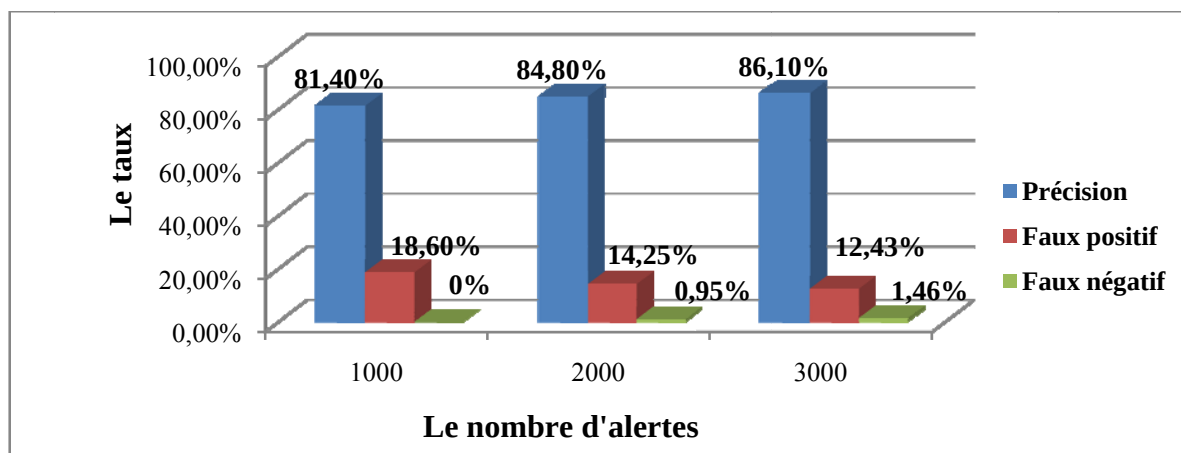


Figure VI.2. Variation de précision, FP et FN suivant le nombre d'alerte

Par la suite nous avons comparé les résultats obtenus avec ceux de l'original en fonction du nombre d'alertes (Figure VI.3 et Figure VI.5).

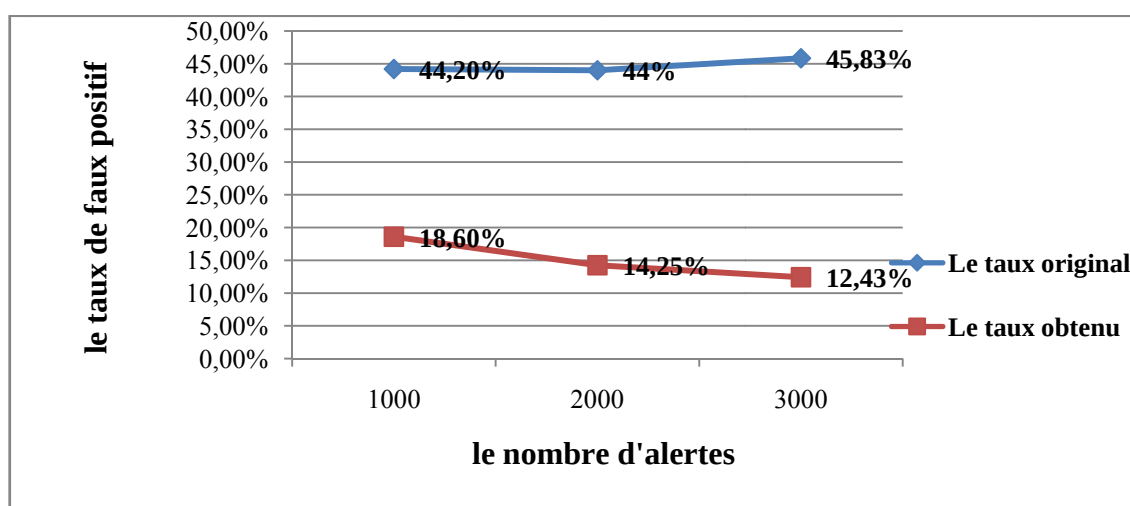


Figure VI.3. Comparaison entre le taux de FP original et le taux de FP obtenu

Le Tableau VI.4 suivant montre les résultats obtenus en utilisant toutes les alertes générées par la base de tests de jour jeudi de la deuxième semaine.

	<i>Précision</i>	<i>Faux positif</i>	<i>Faux négatif</i>
3000 alertes	86,10%	12,43%	1,46%
6000 alertes	76,66%	22,60%	0,73%
9000 alertes	82,63%	16,87%	0,40%
12494 alertes	85,85%	13,51%	0,60%
Moyenne	82,81%	16,35%	0,80%

Tableau VI.4. Résultats obtenus par toutes les alertes de la base de tests jeudi

Ce tableau peut être schématisé dans la Figure VI.4. Le graphe suivant montre la variation de précision, faux positif et faux négatif suivant le nombre d'alertes générées.

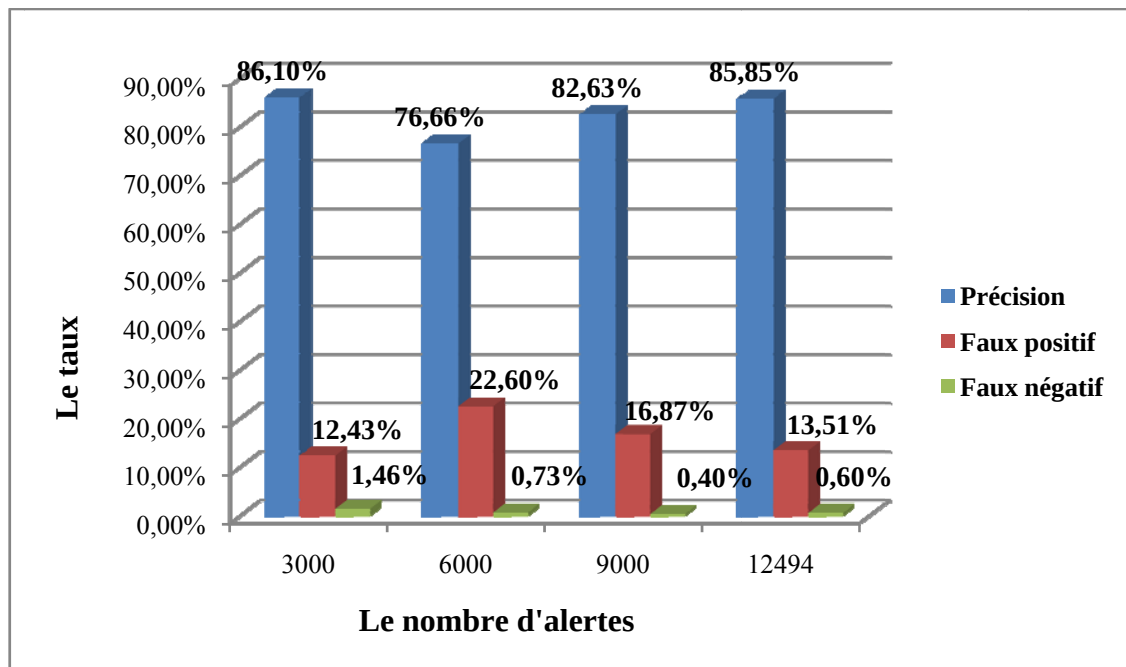


Figure VI.4. Variation de précision, FP et FN suivant le nombre d'alertes

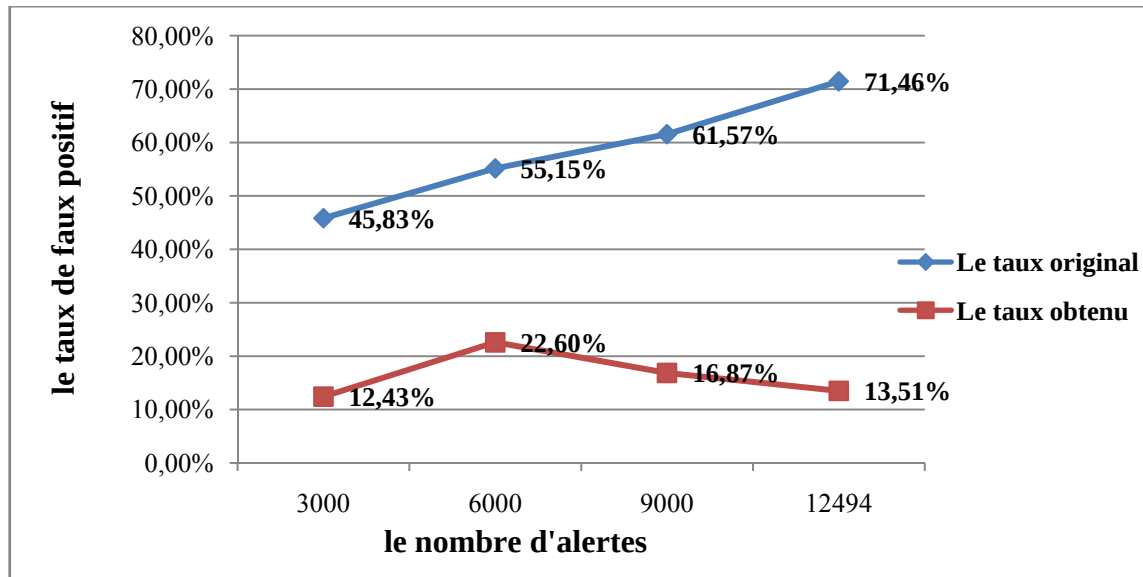


Figure VI.5. Comparaison entre le taux de FP original et le taux de FP obtenu

VI.5 Conclusion

Nous avons présenté dans ce chapitre les tests d'un prototype de système de détection de faux positif. Ce prototype inclut les principales fonctions, notamment l'acquisition des alertes à partir d'un fichier *log* et en temps différé, l'extraction des règles en utilisant un algorithme général *PART* permettant d'exploiter les règles générées pour classer des nouvelles alertes.

Le prototype développé a été validé par une expérimentation sur les bases de données de test *DARPA* 1998. Nous avons obtenu un classifieur de qualité avec une précision performante peut atteindre jusqu'à 86,10%. Ainsi, nous avons montré que l'approche proposée donne un meilleur résultat en terme de taux de faux positif pouvant atteindre jusqu'à 12,43% par rapport à celui de *Snort* qui génère un taux de faux positif de 71,46%.

Dans ce travail, nous n'avons pas comparé le prototype avec ceux de la littérature, car aucun d'eux ne prend en considération le taux faux positif et le taux de faux négatif sous contraintes de précision.

Cependant, il faut signaler que le classifieur (*PART*) utilisé n'est pas parfait car les résultats ont été obtenus en utilisant un algorithme général. Ceci nous amène à penser qu'il est possible d'améliorer les résultats avec de nouveaux développements, tels qu'un algorithme spécifique pour la classification.

Il faut signaler aussi que l'analyse des alertes a été faite en temps différé, ce qui rend l'analyse plus simple.

Conclusion générale et perspectives

Conclusion générale et perspectives

Ce mémoire a pour objectif l'étude du problème de faux positif déclenché par les systèmes de détection d'intrusion (IDS_s).

Pour cette étude, nous avons consacré une partie de notre travail à l'état de l'art, dans laquelle nous avons commencé par donner un bref aperçu sur la sécurité des systèmes informatique avec une présentation des IDS_s . Nous avons présenté le problème posé par les IDS_s . Malgré leur utilité, en pratique les IDS_s souffrent d'au moins de deux problèmes : le nombre de faux positif et de faux négatif. Les faux positif (les fausses alertes) sont générés lorsque l' IDS identifie des activités normales comme des intrusions, alors que les faux négatifs correspondent aux attaques ou intrusions qui ne sont pas détectées.

Ensuite, nous avons présenté une classification sur les approches existantes qui traitent le problème de faux positif. Nous avons vu qu'il existe trois classes qui traitent le problème de faux positif : des méthodes basées sur la classification, des méthodes basées sur les causes d'origine et des méthodes basées sur l'hypothèse de la fréquence d'apparition d'alertes. Nous avons montré aussi les données qui peuvent être utilisées pour simuler notre approche. Les données *DARPA* utilisées contiennent un grand nombre d'instances mises à la disposition de la communauté des chercheurs en sécurité réseau en particulier pour évaluer les systèmes de détection d'intrusion. Par la suite, nous avons présenté le système de détection d'intrusion utilisé (*Snort*) et la méthode de l'étiquetage des alertes générées par ce dernier.

Dans ce mémoire, nous avons proposé une approche de détection de faux positif basée sur la notion de *decision list*. Les règles de *decision list* sont évaluées en commençant par la première et en parcourant les règles séquentiellement jusqu'à ce qu'un antécédent soit vérifié. Si aucune règle n'est vérifiée, on trouve généralement une règle par défaut qui attribue une classe aux instances restantes. Ensuite, nous avons utilisé l'extracteur de connaissance *PART*. Ce dernier extrait des règles à partir d'une base d'apprentissage. Ensuite ces règles sont exploitées pour classer de nouvelles alertes. Nous avons choisi le classifieur *PART* parce

que c'est un moyen rapide et efficace de production de règles qui sont à la fois précises et compréhensibles.

Dans le but de montrer l'efficacité de notre approche, nous avons mené des tests qui consistent à montrer le taux de faux positif et de faux négatif en fonction de la précision sur la base de tests de *DARPA 98*. D'après les résultats obtenus, nous avons obtenu un classifieur de qualité avec une précision acceptable qui peut atteindre 86,10%. Ainsi, nous avons montré que l'approche proposée donne un meilleur résultat en terme de taux de faux positif qui peut atteindre jusqu'à 12,43% par rapport à celui de *Snort* qui génère un taux de faux positif allant jusqu'à 71,46%. Nous avons remarqué aussi que l'approche donne un taux de faux négatif minimal, en moyenne 0,80%.

Dans ce travail, nous n'avons pas comparé notre approche avec celles de la littérature, car aucune ne prend en considération le taux de faux positif et le taux de faux négatif.

Perspectives

Le travail réalisé peut être étendu et enrichi, notamment :

- Premièrement, dans le prototype réalisé nous avons utilisé un système de détection de faux positif simple. Pour améliorer les résultats du système, nous pouvons utiliser un système de classification basé sur un système expert ;
- Deuxièmement, nous pouvons renforcer amplement la capacité d'extraction de règles de classification des alertes avec un algorithme spécifique ;
- Enfin, nous souhaitons compléter ce travail par le développement d'un module d'acquisition des alertes généré par *l'IDS* et les stocker dans une base de données en temps réel. Ceci permettra de faire des tests de notre système en temps réel.

En développant les modules cités précédemment, notre système pourra être efficacement déployé.

Bibliographie

Références bibliographiques

- [Alessandri 04] Dominique Alessandri, "Attack-Class-Based Analysis of Intrusion Detection Systems", Ph.D. Thesis, University of Newcastle upon Tyne, School of Computing Science, Newcastle upon Tyne, UK, 2004.
- [Alharby et al, 05] A. Alharby, H. Imai, "IDS False Alarm Reduction Using Continuous and Discontinuous Patterns". Proceedings of ACNS 2005, Springer, Heidelberg, 2005, pp. 192-205.
- [Anthony 05] Martin Anthony "Decision lists, chapter in *Boolean Methods and Models*", CDAM Research Report LSE-CDAM, 2005.
- [Bace 00] Rebecca Bace and Peter Mell. Intrusion Detection Systems. NIST Special Publication on Intrusion Detection Systems, 2000.
- [Baudru 09] Nicolas Baudru, Sécurité des systèmes informatiques, esil, 2009.
- [Berthomier 05] Eric Berthomier, "Formation Sécurité des Réseaux, Version 1.0.1", Support Instructeur, 2005.
- [Bishop 03] Bishop, M. Computer Security Art and Science. Numéro ISBN 0201440997. Addison-Wesley Professional, 2003.
- [Briffaut 08] Jérémy BRIFFAUT, "Formalisation et garantie de propriétés de sécurité système : application à la détection d'intrusions", thèse de doctorat, université d'ORLEANS 2008.
- [Bsi 00] International Standards Organization. Information Processing Systems - OSI – Basic Reference Model - Part 2: Security Architecture. ISO 7498-2, February 1989.
- [Carlson 01] D. CARLSON, Modélisation d'application XML avec UMP, Edition EYROLLES, Paris, 2001.
- [Clifton et al, 00] C. Clifton, G. Gengo, "Developing Custom Intrusion Detection Filters Using Data Mining", Proceedings of MILCOM 2000, IEEE Press, New York, pp. 440-443, 2000.
- [Debar et al, 00] H. Debar, M. Dacier and A. Wespi. A revised taxonomy for intrusion detection systems. Annales des télécommunications. July–August 2000.
- [Debar et al, 04] Hervé Debar, Benjamin Morin, Frédéric Cuppens, Fabien Autrel, Ludovic Mé, Bernard Vivinis, Salem Benferhat, Mireille Ducassé, Rodolphe Ortalo,

- “Sécurité informatique”, RSTI - TSI – 23/2004. Sécurité informatique, pages 359 à 390.
- [Florin et al, 99] G. FLORIN, S. NATKIN, La sécurité, CNAM-Cédric, 1999.
- [Gadelrab 08] M.S. GADELRAH, Évaluation des Systèmes de Détection d’Intrusion, Thèse de doctorat, Université de Toulouse 2008.
- [Hornick et al, 07] M.F Hornick, E. Marcadé and S. Venkayala, Java Data mining: strategy, standard and practice, Morgan Kaufmann publishers, 2007.
- [ISO 05] ISO/IEC. « Information Security Management Systems – Requirements ». (27001), 2005.
- [Julisch 01] Klaus Julisch. “Mining Alarm Clusters to Improve Alarm Handling Efficiency”, Computer Security Applications Conference, 2001. ACSAC 2001. Proceedings 17th Annual 10-14 Dec. 2001 Page(s):12 - 21. IEEE 2001.
- [Julisch 02] K. Julisch, M. Dacier, “Mining Intrusion Detection Alarms for Actionable Knowledge”, Proceedings of KDD’02, ACM Press, New York, 2002, pp. 366-375.
- [Julisch 03] Klaus Julisch. “Using Root Cause Analysis to Handle Intrusion Detection Alarms”, PhD thesis, University of Dortmund, Germany, IEEE 2003.
- [Kendall 99] K. Kendall, "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems", Master Thesis, MIT Department of Electrical Engineering and Computer Science, USA, 1999.
- [Laurent et al, 09] Laurent Bloch, Christophe Wolfhugue, Sécurité informatique, Principes et méthode à l’usage des DSI, RSSI et administrateurs, 2ème édition, EYROLLES, Paris, 2009.
- [Leopld et al, 99] E. LEOPLD, S. LHOSTE, La sécurité informatique, PUF, 1999.
- [Manganaris et al, 00] S. Manganaris, M. Christensen, D. Zerkle, et al, “A Data Mining Analysis of RTID Alarms”, Computer Networks, 34(4), pp. 571-577, 2000.
- [Marcus 09] Marcus J. Ranum. « The Six Dumbest Ideas in Computer Security ». <http://www.certifiedsecuritypro.com/>, Mars 2009.
- [Nassih 06] Mohamed Nassih, SYSTÈME DE DÉTECTION D’INTRUSION, université de Casablanca, 2006.
- [Papini 05] Odile PAPINI, MASTER SIS PRO: logique et sécurité, Détection d’intrusionLSIS. Université de Toulon et du Var, 2005.

- [Parpinelli et al, 02] R.S Parpinelli, H.S Lopes and A.A Freitas "Data mining with an Ant colony optimization algorithm", IEEE trans. on Evolutionary Computation, vol (6), pp 321-332, 2002.
- [Phillip et al, 98] A. Phillip, Porras and Alfonso Valdes. Live traffic analysis of tcp/ip gateways. Proc. ISOC Symposium on Network and Distributed System Security (NDSS98). (San Diego, CA, March 98), Internet Society, 1998.
- [Pietraszek 04] T. Pietraszek, "Using Adaptive Alert Classification to Reduce False Positives in Intrusion Detection", Proceedings of RAID'04, Springer, Heidelberg, pp. 102-124, 2004.
- [Pietraszek et al, 05] Tadeusz Pietraszeka and Axel Tannera "Data Mining and Machine Learning Towards Reducing False Positives in Intrusion Detection", Appearing in Information Security Technical Report, 10(3), 2005.
- [Pietraszek 06] Tadeusz Pietraszek. "ALERT CLASSIFICATION TO REDUCE FALSE POSITIVES IN INTRUSION DETECTION", Thesis for the acquisition of the doctor degree. University of Germany, July 2006.
- [Puketza 96] N.J. Puketza, K. Zhang, M. Chung, B. Mukherjee, R. A. Olsson, "A Methodology for Testing Intrusion Detection Systems", IEEE Transactions on Software Engineering, Vol. 22, Number 10, pp. 719-729, 1996.
- [Puketza 97] Nicholas Puketza, Mandy Chung, Ronald A. Olsson and Biswanath Mukherjee, "A Software Platform for Testing Intrusion Detection Systems", Journal of IEEE Software, Vol. 14, Number: 5, pp. 43-51, 1997.
- [Richard et al, 98] Richard P.Lippmann, "MIT Lincoln Laboratory Offline Component of DARPA 1998 Intrusion Detection Evaluation", MIT Lincoln Laboratory, PI Meeting, 14 Décembre 1998.
- [Snort 03] M. Roesch, "SNORT. The Open Source Network Intrusion System," Web page at <http://www.snort.org> (1998–2003).
- [SOUBRIER 98] C. SOUBRIER, Sécurité Optimale, Simon & Schuster Macmillan, France, 1998.
- [Tabia 08] Karim Tabia, Modèles graphiques et approches comportementales pour la détection d'intrusions, thèse présentée A L'UNIVERSITE d'Artois pour obtenir le grade de DOCTEUR DE L'UNIVERSITE d'Artois, Novembre 2008.

- [Viinikka et al, 06] J. Viinikka, H. Debar, L. Mé, et al, “Time Series Modeling for IDS Alert Management”, Proceedings of Asia CCS’06, ACM Press New York, pp 102-113, 2006.
- [Witten et al, 05] I. H. Witten, E. Frank, Data Mining: Practical Machine Learning tools and Techniques with JAVA Implementations, Morgan Kauffman Publishing, 2005.
- [Xiao et al, 08] Fu Xiao, Xie Li, “Using Outlier Detection to Reduce False Positives in Intrusion Detection”, IFIP International Conference on Network and Parallel Computing2008, China, IEEE2008.
- [Yves 09] M. Yves Deswarte, Évaluation des Systèmes de Détection d'Intrusion, thèse présentée A UNIVERSITE TOULOUSE III – PAUL SABATIER pour obtenir le grade de DOCTORAT DE L'UNIVERSITE DE TOULOUSE, mars 2009.