Titre: Title:	Détection d'intrus dans les réseaux à l'aide d'agents mobiles
Auteur: Author:	Simon Trudeau
Date:	2006
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Trudeau, S. (2006). Détection d'intrus dans les réseaux à l'aide d'agents mobiles [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/2384/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/2384/
Directeurs de recherche: Advisors:	Jose Manuel Fernandez, & Samuel Pierre
Programme: Program:	Génie informatique

UNIVERSITÉ DE MONTRÉAL

DÉTECTION D'INTRUS DANS LES RÉSEAUX À L'AIDE D'AGENTS MOBILES

TRUDEAU SIMON
DÉPARTEMENT DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION DU DIPLÔME DE
MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
AOÛT 2006

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

DÉTECTION D'INTRUS DANS LES RÉSEAUX À L'AIDE D'AGENTS MOBILES

présenté par : TRUDEAU Simon
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury constitué de :

M. MERLO Ettore, Ph.D., président
M. FERNANDEZ José M., Ph.D., membre et directeur de recherche
M. PIERRE Samuel, Ph.D., membre et codirecteur de recherche
M. BOYER François-Raymond, Ph.D., membre

Ce mémoire est dédié à tous ces étudiants qui, armés de rêves, veulent changer le monde, qui échouent en essayant et se relèvent pour recommencer, pour un jour, peut-être, réussir.
Ce mémoire est aussi dédié à tous ceux pour qui le chemin dans la vie n'est pas balisé et qui doivent, à tâton, frayer leur chemin dans le noir, pour un jour voir la lumière éclairer leurs pas.
Finalement je dédie ce mémoire à tous les bums de bonnes familles qui mettent à jour et surmontent LEURS limites et à tous ceux qui se battent pour la justice et la vérité.

Remerciements

Je tiens à remercier l'équipe du *LAboratoire de Recherche en Réseautique et Informatique Mobile* (LARIM) et l'École Polytechnique, principalement mes directeurs Monsieur José Fernandez et Monsieur Samuel Pierre pour leur soutient logistique, financier et surtout leurs bons conseils sans lesquels il m'aurait été bien difficile de mener à terme ce projet de recherche.

Je tiens tout spécialement à remercier Eva-Maria Garcia qui m'a supporté à travers mes moments difficiles et m'a inspiré par sa propre détermination dans son projet de recherche. Merci Èva !

Je tiens à remercier mes parents pour m'avoir encouragé durant ma maîtrise et pour m'avoir apporté leur soutien et leur compréhension.

Finalement une bonne partie de mon questionnement mathématique serait resté sans réponse sans l'aide d'Étienne Chassé-St-Laurent qui a éclairé ma lanterne et m'a fait apprécier et comprendre les probabilités et statistiques.

Résumé

Aujourd'hui, il est difficile pour les grandes organisations de faire fit des défis que posent la sécurité informatique. Les infrastructures informatiques sont plus que jamais interconnectées entre elles et sont exposées autant aux menaces provenant de l'intérieur que de l'extérieur de l'organisation. Afin de faire face aux menaces, parmi les moyens mis en œuvre pour défendre le réseau contre les attaques on retrouve les systèmes de détection d'intrus (IDS). Ces systèmes, qui peuvent être de nature logicielle ou matérielle, sont de plus en plus répandus et peuvent représenter une part importante du budget du département informatique d'une organisation. Malheureusement ces équipements, bien que forts utiles, présentent des faiblesses non négligeables. Le nombre de faux positifs, soit des alarmes non fondées, peut rapidement dépasser la capacité de traitement des analystes en charge de la sécurité.

Afin d'adresser ce problème de performances, plusieurs solutions furent avancées. Ces dernières années nous avons vue faire leurs apparitions différentes architectures hiérarchiques, par agents et par agents mobiles. Plusieurs stratégies de détection ont été mises en œuvre pour tenter d'améliorer la situation tel que la détection par apprentissage statistique, par règles comme dans le cas du populaire IDS SNORT et par réseau de pétri, par exemple.

Malheureusement force fut de constater qu'il est bien difficile d'évaluer ces solutions pour en connaître leur efficacité réelle. À l'heure actuelle il nous est difficile de faire le lien entre les métriques de performance, les caractéristiques des IDS et l'environnement qui influence grandement les résultats.

Pour évaluer les liens entre l'environnement, les caractéristiques des IDS et les métriques de performance, nous avons décidé dans ce mémoire de mettre sur pied un cadre structurant pour l'évaluation des performances des IDS. Pour ce faire, nous avons élaboré une taxonomie de l'environnement afin de pouvoir le caractériser de façon objective et subjective. Nous avons aussi bâti une taxonomie des caractéristiques de détection des IDS susceptibles d'influencer les performances et qui se distingue en tenant compte des propriétés des agents mobiles. De plus, nous avons sélectionné un ensemble de métriques sujet à être de bons indicateurs de performance.

Dans le but de porter des jugements sur un environnement difficilement cernable

dans sa totalité, nous avons développé un modèle mathématique simplifié auquel nos conclusions s'appliquent. À l'intérieur de ce modèle il nous est permis de comprendre les relations qui existent entre l'environnement, les IDS à l'étude et leurs impacts sur les performances.

Pour valider de façon expérimentale notre cadre, nous avons conçu un IDS par agents mobiles qui fait usage de corrélation temporelle et spatiale afin d'en tester les performances versus une autre solution par agents mobiles et une architecture plus standard.

Afin de mener nos expériences à bien, nous avons conçu une plate-forme de test capable de générer des faux positifs et des attaques. Les événements générant des faux positifs ont été modélisés d'après un scénario d'interruption accidentelle de connexions TCP pouvant s'apparenter à une attaque de type *SYN Flood*. Les attaques, quant à elles, ont été modélisées pour s'apparenter à une attaque de type *Fast Scan*. Nous définirons un *Fast Scan* comme un balayage des ports d'un ou plusieurs hôtes durant un très court lapse de temps.

Lors de la réalisation de nos tests, nous avons choisi comme métrique le taux de faux positifs, c'est-à-dire le nombre de fausses alarmes, et le taux de faux négatifs, c'est-à-dire le nombre d'attaques qui n'ont pas déclenché d'alarmes. Nous avons comparé entre eux les taux de faux positifs et de faux négatifs des algorithmes pour différents paramètres environnementaux et de l'IDS. Nous avons constaté que les résultats obtenus lors de nos tests étaient fidèles aux résultats obtenus de façon purement théorique. Nous avons observé que chaque paramètre environnemental et de l'IDS influence les taux de faux positifs et de faux négatifs que nous obtenons.

Nos expériences ont permis de valider nos hypothèses et d'établir le lien qui existe entre les variables de l'environnement, les caractéristiques de chaque IDS et les performances de détection. Par ailleurs, il nous a été possible d'observer une relation fonctionnelle entre différentes métriques. Lors de nos expériences il fut impossible d'optimiser les taux de faux positifs sans observer une dégradation des taux de faux négatifs. L'usage de corrélation sur des dimensions de l'environnement nous a permis d'améliorer les résultats en permettant de mieux discriminer l'attaque.

Abstract

In today's world, it is very difficult for big organizations to overlook the challenges posed by information security. More and more, computer infrastructures are interconnected with each others and are exposed to threats coming from inside and outside of the organization. To face those threats, among the means to defend the network, we have the intrusion detection system (IDS). Those systems, which can be software or hardware in nature, have an increasing widespread use and can represent an important part of an Information Technology department's budget. Unfortunately, those equipments, even though very useful, have non negligible flaws. The number of false positive, being the false alarm, can rapidly overwhelm the capability of the security analysts to analyse them.

To address this performance issue, many solutions have been put forward. In recent years, we have seen different IDS architectures such as hierarchical, by agents and by mobile agents, for example, making their appearance. Many detection strategies were put forward to try to improve on the situation. We have seen strategies such as intrusion detection by statistical learning, by rules like popular IDS SNORT and by Petri Network just to name a few.

Unfortunately, we found out that assessing the real efficiency of those solutions was very difficult. Right now, it is very difficult to make the connection between performance metrics, IDS's characteristics and the environment which has a huge impact on results.

To assess the links between the environment, IDS's characteristics and performance metrics, we decided in this master thesis to put forward an IDS performance evaluation framework. To do so, we have elaborated a taxonomy to allow the characterization of the environment in an objective and subjective manner. We also have built a taxonomy of intrusion detection system's characteristics which can influence performances and take into account mobile agents' properties. Furthermore, we have selected a set of metrics which can be good performance indicators.

To allow us to make hypothesis on an environment too complex to be fully apprehended, we came up with a simplified mathematical model to which our conclusions will apply. Inside this model we can understand the relations between the environ-

ment, IDS of interest and their impacts on performance.

To allow for the experimental validation of our framework, we built an IDS which makes use of mobile agents to perform spatial and temporal information correlation. We compared the performances of our solution against another IDS making use of mobile agents and a standard architecture.

To carry on our experiments, we built a test platform able to generate false positive and attack events. False positive events were modeled after half opened TCP connections which can look like a *SYN Flood* attack. Attacks were modeled to look like a *Fast Scan* attack. We shall describe a *Fast Scan* as a port scan of one or more designated hosts over a very short period of time.

During our test sessions, we chose as a metric, the false positive rate, which is the number of false alarms, and the false negative rate, which is the number of attack that passed without triggering an alarm. We compared between each other the false positive and false negative rate of our algorithms for different IDS and environmental parameters. We found that our experimental results where coherent with our theoretical results. We found out that each environmental parameter and each IDS parameter have an influence on the false positive and false negative rate that we obtain.

Our experiment allowed us to validate our hypothesis and establish a link between the environment variables, the characteristics of each IDS and the intrusion detection performances. Also, were able to observe a functional relation between different metrics. Through out our experiments, we were unable to optimize the false positive rates without noticing a degradation of the false negative rates. Using correlation on dimensions of the environment allowed us to improve our detection results by making more precise threat recognition.

Table des matières

Dédicace iv

Remerciements v

Résumé vi

Abstract viii

Table des matières x

Liste des figures xiii

Liste des sigles et abréviations xv

Chapitre 1 INTRODUCTION 1
1.1 Les systèmes de détection d'intrus 2
1.2 Objectifs de recherche 4
1.3 Organisation du mémoire 6

Chapitre 2 SURVOL DES MÉTHODES DE DÉTECTION D'INTRUS DANS LES RÉSEAUX 7
2.1 Vulnérabilité dans les réseaux 7
2.2 Les vulnérabilités propres aux réseaux ad hoc 8
2.3 La détection d'intrus dans les réseaux 10
2.3.1 Les IDS 10
2.3.2 Les méthodes de détection d'attaques 11
2.4 Les limites des IDS 13
2.5 Les architectures des IDS 15
2.5.1 Systèmes Hiérarchiques 16
2.5.2 Systèmes *Peer-to-Peer* 17
2.5.3 Systèmes par Agents Mobiles 18
2.6 Travaux antérieurs sur les agents mobiles pour la détection d'intrusions 19
2.6.1 Les architectures dans le domaine filaire 20

2.6.2 Les architectures de IDS pour les réseaux ad hoc 24
2.7 Analyse des solutions d'IDS par agents mobiles 25
2.8 Objectifs de recherche 26

Chapitre 3 SYSTÈME DE DÉTECTION D'INTRUS PROPOSÉ 28
3.1 Énoncé du cadre théorique 29
3.2 Caractéristiques environnementales 31
3.2.1 La réalité objective 33
3.2.2 La réalité subjective 34
3.2.3 IDS (Méta) 35
3.3 Caractéristiques des systèmes de détection d'intrus à l'aide d'agents mobiles 35
3.4 Métriques de performances 38
3.4.1 Métriques de performance au niveau réseau 38
3.4.2 Métriques de performance au niveau de la mémoire 40
3.4.3 Métriques de performance au niveau des performances de calcul 40
3.5 IDS à l'étude et hypothèses de performance 41
3.5.1 Stratégie de résolution DSSL 41
3.5.2 Stratégie de résolution PSSG 42
3.5.3 Stratégie de résolution PSAR 43
3.5.4 Hypothèses de performance 44
3.5.5 Description du processus d'intrusion et de détection 46
3.6 Modélisation mathématique de l'environnement 47
3.6.1 Description de l'expérience 47
3.6.2 Variables 47
3.6.3 Quelques précisions 49
3.6.4 Probabilité d'alarme 50
3.6.5 Probabilité de Faux Positifs 53
3.6.6 Probabilité de Faux Négatifs 55
3.7 Analyse de l'impact des paramètres environnementaux et de l'IDS sur les taux de faux positifs et de faux négatifs 56
3.7.1 Probabilité de faux positifs en fonction du taux de trafic gris et de trafic noir 57

3.7.2 Probabilité de faux négatifs en fonction du taux de trafic gris et de trafic noir 58
3.7.3 Variation du nombre d'hôtes 60
3.7.4 Variation du seuil de détection 65
3.8 Analyse des taux de faux positifs en fonction des taux de faux négatifs, en fonction des paramètres expérimentaux 68
3.9 Vérification théorique des hypothèses 69

Chapitre 4 ÉVALUATION DE PERFORMANCE ET RÉSULTATS EXPÉRIMENTAUX 70
4.1 Environnement matériel et logiciel d'implantation 70
4.2 Description de l'environnement de test 72
4.2.1 Caractéristiques de l'environnement 73
4.3 Plan d'expériences 74
4.3.1 Choix des variables 74
4.3.2 Choix des métriques 74
4.3.3 Conception des sessions de test 76
4.3.4 Déroulement des tests 76
4.4 Analyse des résultats 78
4.4.1 Observation du taux de faux positifs et de faux négatifs en fonction du taux de trafic gris 78
4.4.2 Observation du taux de faux positifs et de faux négatifs en fonction du taux de trafic noir 82
4.5 Différences entre les résultats théoriques et expérimentaux 87
4.6 Vérification expérimentale des hypothèses 88

Chapitre 5 CONCLUSIONS 90
5.1 Synthèse des contributions 91
5.2 Recherches futures 92

Bibliographie 94

Liste des figures

FIGURE 2.1	Architecture hiérarchique	16
FIGURE 2.2	Déplacement d'agents mobiles avec SPARTA	21
FIGURE 3.1	Filtrage de la réalité	31
FIGURE 3.2	Taxonomie des caractéristiques de l'environnement	32
FIGURE 3.3	Taxonomie des caractéristiques de l'IDS	36
FIGURE 3.4	Probabilité d'alarme en fonction du nombre d'hôtes étant donné $t = 1$, $s_i = 10$, $\gamma_i = 10$, $\lambda = 1$.	53
FIGURE 3.5	Taux de faux positifs(β) en fonction du trafic noir (λ) et du trafic gris(γ_i)	57
FIGURE 3.6	Taux de faux positifs(β) en fonction de la quantité de trafic noir (λ) et la quantité de trafic gris (γ_i) par unité de temps (t)	58
FIGURE 3.7	Taux de faux négatifs(α) en fonction du trafic noir (λ) et du trafic gris(γ_i)	59
FIGURE 3.8	Taux de faux négatifs(α) en fonction de la quantité de trafic noir (λ) et la quantité de trafic gris (γ_i) par unité de temps (t)	60
FIGURE 3.9	Taux de faux positifs(β) en fonction du nombre d'hôtes (m)	61
FIGURE 3.10	Taux de faux négatifs(α) en fonction du nombre d'hôtes (m)	62
FIGURE 3.11	Taux de faux positifs(β) en fonction du ratio signal sur bruit (λ/γ_i) et du nombre d'hôtes (m)	63
FIGURE 3.12	Taux de faux négatifs(α) en fonction du ratio signal sur bruit (λ/γ_i) et du nombre d'hôtes (m)	64
FIGURE 3.13	Taux de faux positifs(β) en fonction du seuil (s_i)	66
FIGURE 3.14	Taux de faux négatifs(α) en fonction du seuil (s_i)	67
FIGURE 3.15	Taux de faux positifs en fonction des taux de faux négatifs en fonction du seuil.	68
FIGURE 4.1	Environnement de test	75
FIGURE 4.2	Algorithmes de détection	77
FIGURE 4.3	Taux de faux positifs(β) en fonction du taux de trafic gris(γ_i)	80
FIGURE 4.4	Taux de faux négatifs(α) en fonction du taux de trafic gris(γ_i)	81
FIGURE 4.5	Taux de faux positifs(β) en fonction du taux de trafic noir(λ)	83

FIGURE 4.6 Taux de faux positifs(β) en fonction du taux de trafic noir(λ) 84
FIGURE 4.7 Taux de faux négatifs(α) en fonction du taux de trafic noir(λ) 85
FIGURE 4.8 Taux de faux négatifs(α) en fonction du taux de trafic noir(λ) 86

Liste des sigles et abréviations

APHIDS	Agent Programmable Hybrid Intrustion Detection System
arachNIDS	Advanced Reference Archive of Current Heuristics for Network Intrusion Detection Systems
CPN	Colored Petri Net
DoS	Denial of Service
DMZ	Demilitarized Zone
DSSL	Détection Standard avec Seuil Local
HIDS	Host Intrusion Detection System
IDA	Intrusion Detection Agent system
IDMEF	Intrusion Detection Message Exchange Format
IDS	Intrusion Detection System
IDXP	Intrusion Detection Exchange Protocol
IPS	Intrusion Prevention System
LIDS	Local Intrusion Detection System
MAIDS	Mobile Agent Intrusion Detection Systems
MANET	Mobile Ad-hoc Network
MIB	Management Information Base
MLSI	Marks Left by Suspected Intruder
NIDS	Network Intrusion Detection System
P2P	Peer-to-Peer
PMAID	Polytechnique Mobile Agent Intrusion Detection
PSAR	Patrouille de Surveillance avec Analyse Raffinée
PSSG	Patrouille de Surveillance avec Seuil Global
ROC	Receiver Operating Characteristic
SFT	Software Fault Tree
SPARTA	Security Policy Adaptation Reinforced Through Agents
WEP	Wired Equivalent Privacy

CHAPITRE 1

INTRODUCTION

La sécurité de l'information est, plus que jamais, une préoccupation importante pour nos sociétés. En effet, Internet a contribué largement à démocratiser les outils servant à pénétrer les systèmes informatiques, les rendant accessibles aux script kiddies de ce monde. Par ailleurs, un besoin d'échange d'information toujours plus grand a mené à une intégration et une interconnectivité accrue des systèmes, les rendant par le fait même plus complexes et donc plus susceptibles de contenir des failles menaçant la sécurité.

Pour faire face à la menace, les grandes organisations de ce monde ont investi massivement dans l'achat de systèmes de détection d'intrus (Intrusion Detection System (IDS)). Ces systèmes ont pour but de reconnaître les différentes attaques en analysant différentes sources de données, principalement le trafic sur le réseau et les journaux d'enregistrement d'événements systèmes. Malheureusement leur efficacité demeure mitigée compte tenu du nombre généralement important de fausses alarmes obtenues.

Dans le but d'adresser ce problème, la communauté scientifique doit développer des architectures de systèmes de détection d'intrus novatrices et doit posséder les moyens de les évaluer pour savoir si elles répondent au problème. Ce mémoire se penchera sur l'évaluation des performances des différentes architectures d'IDS. Nous concentrerons nos efforts sur la mise au point d'un cadre visant à évaluer les performances des systèmes.

Dans ce chapitre, nous exposerons d'abord les définitions et concepts de base, nécessaires à la compréhension du mémoire. Ensuite, nous présenterons les éléments de la problématique qui justifient ce mémoire, suivra ensuite la présentation de nos objectifs de recherche. Pour terminer nous ferons un bref tour d'horizon du plan des chapitres qui suivent.

1.1 Les systèmes de détection d'intrus

Pour faire face aux menaces qui planent sur les réseaux informatiques, nous retrouvons dans la boîte à outil des spécialistes en sécurités informatiques, les systèmes de détection d'intrus - IDS (Intrusion Detection Systems). Ce moyen de défense est très répandu dans les grandes entreprises et ces dernières n'hésitent pas à investir des sommes considérables pour des solutions de détection d'intrus matérielles et logicielles pour protéger leurs infrastructures informatiques. Les IDS se divisent habituellement en deux grandes catégories soient les IDS de réseau (Network Intrusion Detection System (NIDS)) qui visent à assurer la protection du réseau et les systèmes visant à assurer la protection des hôtes (Host Intrusion Detection System (HIDS)).

Dans le cas des NIDS, ils assurent généralement la protection du réseau en écoutant de manière furtive le trafic sur le réseau dans le but de repérer des anomalies ou des comportements suspects. Ces derniers peuvent entre autre analyser les piles protocolaires (TCP, UDP, IP, ICMP, etc.) ou applicatives (NetBIOS) afin d'y déceler des violations qui seraient les signes d'une attaque tel que le *Ping-of-death*, *WinNuke* et *TCP Stealth Scanning* par exemple. La reconnaissance des attaques se fait dans la plus part des cas par la reconnaissance de patron d'attaques. Un patron d'attaque correspond généralement à une séquence de caractères que l'IDS associe à une attaque.

Dans le cas des HIDS le système se concentre particulièrement sur l'analyse des journaux (*syslogs, lastlogs, messages, wtmp,* etc.) mais aussi sur l'analyse du trafic réseau entrant et sortant d'un hôte afin d'y déceler des signes d'intrusions ou d'abus de privilèges tel que les Déni de Services, tentatives d'accès non autorisés, attaques par débordement de tampons, Backdoors, chevaux de troie, exécution de codes malicieux, etc.

Malheureusement, à l'heure actuelle ces systèmes sont loin de pouvoir identifier parfaitement tous les types d'attaques. Pour une grande partie des attaques, ces dernières prennent l'apparence d'une condition exceptionnelle du système, par exemple un paquet manquant ou tronqué, qui peut avoir des causes naturelles comme une défaillance au niveau électrique par exemple, pour se dissimuler. Elles ne sont bien souvent reconnaissables que si l'on a une description très précise de l'attaque, toutefois si la description de cette dernière est trop précise alors le moindre changement au patron d'attaque et cette dernière passera inaperçue. Aussi ces systèmes souffrent-ils d'un haut taux de faux positifs étant donné le nombre élevé de fausses alarmes qui

sont générées par des événements bénins. Dans le cas de réseaux au volume de trafic important, ce nombre d'alarmes peut rapidement dépasser la capacité de traitement des analystes chargés de déceler les vraies attaques. Malheureusement afin d'ajuster le nombre d'alarmes à la capacité de traitement, il arrive que la sensibilité de l'IDS doit être diminuée ce qui entraîne une augmentation du taux de faux négatifs, c'est-à-dire qui permet à des attaques de passer inaperçues. Ce moyen de défense, dans bien des cas, se révèle donc bien peu pratique à opérer.

Le problème des taux de faux positifs et de faux négatifs élevés intéressent la communauté scientifique et industrielle depuis longtemps. Plusieurs architectures ont été mises de l'avant afin d'améliorer la vitesse de traitement des IDS, qui peuvent entre autres avoir des quantités importantes de données à analyser et améliorer leurs capacités de détection et leur précision. Parmi les architectures avancées, nous retrouvons celles qui comportent une hiérarchie d'IDS permettant ainsi une distribution de l'analyse sur plusieurs systèmes et un mécanisme d'agrégation des données, celles basées sur les règles, l'apprentissage de patrons d'attaques, etc. Plusieurs de ces architectures seront détaillées au chapitre 2 . Parmi les solutions avancées nous retrouvons les architectures basées sur les agents mobiles. Les agents mobiles ont la propriété de déplacer leur code d'exécution et parfois même leurs états sur différents systèmes pour effectuer, dans le cadre des IDS, de la détection d'intrus. Ils disposent souvent de mécanismes de communication inter agents ce qui fait d'eux d'excellents candidats pour corréler de l'information.

Toutefois, face au grand nombre de systèmes et d'architectures disponibles, nous sommes en droit de nous interroger sur leur efficacité. Pour ce faire, nous devons nous entendre sur des critères de performance et sur un ensemble de métriques pour les évaluer. Les recherches ont montré que l'environnement auquel étaient soumis les systèmes de détection d'intrus influençait grandement leur capacité de détection. Aussi les métriques doivent-elles tenir compte de l'environnement dans lequel les IDS seront déployés. Ce n'est pas une mince affaire quand on sait que certains environnements sont hautement dynamiques avec des profils d'usagers et de trafic très changeants. Aussi l'évaluation des performances des IDS est-elle une question complexe auquelle la communauté scientifique a répondu avec des résultats mitigés.

1.2 Objectifs de recherche

Afin d'adresser l'épineux problème des performances des IDS et ultimement en arriver à élaborer des solutions qui pourront adéquatement solutionner le problème du nombre élevé de faux positifs et de faux négatifs, nous nous proposons dans ce mémoire d'élaborer un cadre structurant. Ce cadre a pour but de mettre en relation l'environnement, les caractéristiques des IDS et les métriques de performance pour qu'ainsi nous puissions comparer différentes stratégies de détection. Pour ce faire, nous élaborerons une taxonomie de l'environnement de façon à pouvoir la circonscrire et la caractériser. De la même manière une taxonomie des systèmes de détection d'intrus tenant entre autre compte des nouvelles réalités telles que les agents mobiles sera élaborée pour pouvoir les caractériser. Nous proposerons aussi un ensemble de métriques de performances pour pouvoir évaluer les différents IDS.

L'Environnement est une réalité complexe, objective et subjective qui est très difficile à circonscrire, voir même impossible. Néanmoins, afin de l'évaluer et ainsi pouvoir porter un jugement sur les performances des IDS, nous l'encadrerons à l'intérieur d'un modèle mathématique dont l'expressivité sera une limite à la précision des jugements que nous pourront porter.

Afin de développer une modélisation de l'environnement, nous nous baserons sur un cas simple inspiré de la vie réelle. Il arrive fréquemment qu'une connexion TCP soit interrompue suite à une défaillance électrique du réseau ou à la terminaison de la connexion. Un paquet *SYN* est envoyé mais le *SYN-ACK* qui devait suivre est perdu sur le réseau. Cette condition normale s'apparente à une attaque de type *SYN-FLOOD* qui exploite cette particularité du protocole pour épuiser les ressources d'un serveur qui en alloue pour chaque paquet *SYN* en attente d'un paquet *ACK*. Un IDS qui surveille ce patron générera un grand nombre de fausses alarmes (faux positifs). Dans le modèle que nous proposons les événements pouvant donner lieu à un faux positif (« trafic gris ») sont de nature locale à chaque machine. Ces évènements sont indépendants dans l'espace, c'est-à-dire que ce qui se produit sur une machine n'est en rien relié à ce qui se produit sur une autre. De plus, ces événements sont indépendants dans le temps, c'est-à-dire que ceux-ci ne sont pas reliés entre eux, et se produisent selon une loi de Poisson.

Afin de modéliser un patron d'attaque, nous nous inspirerons d'une attaque de type *fast scan* où un grand nombre de paquets est envoyé sur le réseau en peu de temps vers

différentes cibles afin de les vérifier toutes en même temps. Les attaques considérées sont des évènements globaux, dits générant du trafic « noir », contenant des signatures détectables sur toutes les plate-formes. Ils se produisent à l'intérieur d'une période de temps déterminée, en principe assez courte, en comparaison avec la fréquence des évènements donnant lieu à des faux positifs. Dans notre expérimentation, ces événements visent à simuler un balayage horizontal rapide des machines d'un réseau LAN.

Afin de valider notre cadre structurant, nous proposons d'évaluer deux systèmes de détection d'intrus à l'aide d'agents mobiles et une solution de détection standard ne faisant pas appel aux caractéristiques des agents mobiles. Pour les besoins de cette recherche nous avons élaboré un système de détection d'intrus que nous appelerons Patrouille de Surveillance avec Analyse Raffinée (PSAR) basée sur une solution précédente appelée Patrouille de Surveillance avec Seuil Global (PSSG) produite à l'École Polytechnique de Montréal que nous évaluerons aussi. Ces deux solutions par agent mobiles seront contrastées par une solution standard appelée Détection Standard avec Seuil Local (DSSL).

Grâce à ces IDS, nous tenterons de valider les hypothèses suivantes :

- Qu'autant pour le paradigme de détection d'intrus de base DSSL que pour les systèmes de détection d'intrus par agent mobile (tel que l'algorithme PSSG), il existe une relation directe entre les taux de faux positifs / faux négatifs et les taux de génération de trafic gris et de trafic noir.
- Dans des conditions égales (taux de trafic gris et noir égaux), l'algorithme PSSG performe mieux que DSSL en termes de faux positifs, mais l'algorithme DSSL performe mieux en termes de faux négatifs.
- La performance en termes de faux positifs/faux négatifs de la stratégie PSAR est supérieure à celle de PSSG dans le même modèle d'environnement et avec les mêmes valeurs de taux de trafic gris et noir.

La validation de ces hypothèses devrait nous permettre de faire le lien avec notre cadre structurant en établissant :

- Qu'il existe une relation entre les critères de performance, les paramètres environnementaux et les caractéristiques des IDS.
- Qu'il peut exister une relation fonctionnelle entre différentes métriques, soit dans notre cas, entre les taux de faux positifs et les taux de faux négatifs.

Pour valider nos hypothèses nous nous proposons de concevoir un modèle mathé-

matique de la performance afin de pouvoir obtenir des résultats de façon théorique. Afin de valider de façon expérimentale notre modèle mathématique, nous construirons des prototypes de systèmes de détection d'intrus tel que mentionné plus haut, implémentant les différentes stratégies de détection d'intrus. Nous construirons un environnement de test permettant de reecréer l'environnement modélisé pour ensuite conduire une batterie de tests en variant les paramètres de l'environnement et des systèmes de détection d'intrus par agent mobile afin d'obtenir des mesures de métriques de performance. Nous procéderons ensuite à analyser les résultats de ces tests et les comparer aux résultats obtenus avec le modèle mathématique afin d'en tirer des conclusions quant à la validité des hypothèses émises.

1.3 Organisation du mémoire

Ce mémoire est organisé en cinq chapitres. Le chapitre 2 se veut un compte rendu de l'état de l'art sur les systèmes de détection d'intrus à l'aide d'agents mobiles dans les domaines filaires et sans-fil. Il est question des architectures marquantes qui ont précédé l'introduction du paradigme par agents mobiles et des avantages et inconvénients de ces dernières. Le paradigme par agent mobile pour la détection d'intrus est introduit et les recherches et architectures marquantes sont présentées. Le chapitre 3 présente le cadre théorique de tests de performance pour agents mobiles que nous proposons. Le chapitre 4 fait état des tests effectués afin de valider les hypothèses visant à valider notre cadre théorique. Pour terminer, le chapitre 5 présente les conclusions tirées de cette recherche et les mets en liens avec l'état de l'art. Ce chapitre met en lumière d'autres avenues de recherches qui pourront être poursuivies dans le futur.

CHAPITRE 2

SURVOL DES MÉTHODES DE DÉTECTION D'INTRUS DANS LES RÉSEAUX

Dans ce chapitre, nous tenterons de mettre en relief les menaces auxquelles doivent faire face les réseaux. Nous nous attarderons aux menaces spécifiques aux réseaux ad hoc et nous explorerons la question de la détection d'intrusion dans les réseaux. Nous nous questionnerons particulièrement sur la pertinence d'une approche distribuée pour la détection d'intrus. Nous regarderons les menaces qui ne peuvent être détectées que par une approche distribuée et nous tenterons de dresser un portrait des solutions existantes qui s'attaquent à ce problème. Nous tenterons de mettre en lumière les forces et les faiblesses des solutions proposées à ce jour et nous proposerons notre modèle afin d'adresser certains de ces problèmes.

2.1 Vulnérabilité dans les réseaux

Il est relativement difficile de définir de façon claire et exhaustive les menaces qui pèsent sur les réseaux informatiques contemporains. Il y a les menaces que nous connaissons et celles que nous ne connaissons pas. Il y a celles qui viennent de l'extérieur du réseau local et celles qui peuvent provenir de l'intérieur même de ce que certains pourraient considérer comme la zone démilitarisée (Demilitarized Zone (DMZ)). En effet, pendant longtemps la sécurité des réseaux a été comparée à celle d'un château fort moyenâgeux, avec ses remparts et ses tours. Malheureusement, cette vision est appelée à changer avec la venue des équipements sans fil dans les réseaux qui font « tomber les murs » du château fort d'antan.

Attaques venant de l'extérieur. Dans les réseaux, Internet ou autres, au-delà du réseau local, il existe plusieurs menaces à la sécurité des données. De façon générale, les menaces prennent la forme d'actions qui peuvent compromettre la **C**onfidentialité, l'**I**ntégrité et la disponibilité (**A**vailability) des données (**CIA**). Par exemple, un attaquant peut tenter d'écouter le réseau pour mettre la main sur des conversations confidentielles et il peut tenter de falsifier les données sur le réseau ou même sur la machine d'un utilisateur. Aussi il peut même tenter de priver un usager légitime de l'usage de ses ressources informatiques en saturant de trafic l'accès au réseau ou en compromettant sa machine et en la forçant à effectuer du travail inutile de façon à empêcher un usage légitime.

La plupart des moyens à la disposition des attaquants exploitent une faille du système ou des protocoles de la machine de la victime pour en prendre le contrôle. Parfois même, ils peuvent exploiter la crédulité et l'ignorance d'une victime (*social engineering*) pour prendre le contrôle à son insu.

Les attaques venant de l'intérieur. En plus des menaces venant de l'extérieur sous la forme de vers, de trojans, de balayage de ports, d'attaques de déni de service, etc... un système doit faire face aux menaces venant souvent de son entourage immédiat. On compte dans cette catégorie les employés qui veulent mettre la main sur des documents corporatifs confidentiels, le sabotage industriel, l'espionnage industriel, l'abus de ressources informatiques, etc. On comprend que dans le cas de menaces internes, les chances de succèes d'un attaquant sont plus élevées, compte tenu du fait qu'il est souvent considéré par le système comme un usager de confiance et qu'il a un accès privilégié au-delà des premiers niveaux de défense.

2.2 Les vulnérabilités propres aux réseaux ad hoc

Nous adopterons la définition suivante pour les réseaux ad hoc : « Un réseau ad hoc est un réseau sans fil capable de rendre transparentes aux utilisateurs mobiles les modifications de topologie qu'il subit » (Percher et Jouga, 2002). Les réseaux ad hoc se caractérisent par une infrastructure composée d'un ensemble de *nœuds mobiles* qui sont capables de communiquer entre eux et d'établir et de maintenir des connexions. Ces réseaux possèdent une topologie changeante et opèrent de façon décentralisée. Il est précisé dans Mishra *et al.* (2004) que les nœuds mobiles peuvent agir de façon

collaborative entre eux en échangeant des ressources (bande passante, information de routage, etc.) pour permettre de communiquer entre deux points éloignés dans le réseau ad hoc. Ces réseaux n'ont pas de limites physiques clairement définies comme les réseaux filaires et n'ont donc pas de point d'entrée spécifique. Par ailleurs, ces réseaux sans fil comportent souvent des limitations de bande passante, de capacité des liens et d'autonomie énergétique Ping *et al.* (2004). La nature même de ces réseaux les rend donc vulnérable à un ensemble d'attaques. Dans cette section, nous présenterons de façon générale trois catégories d'attaques contre lesquelles les réseaux ad hoc sont particulièrement vulnérables :

- Attaques contre le protocole de routage ;
- Attaques de déni de service (Denial of Service (DoS) attack) ;
- Interception de trafic.

Attaques contre le protocole de routage. Parmi les problèmes inhérents à la sécurité des réseaux mobiles ad hoc (Mobile Ad-hoc Network (MANET)), on retrouve la difficulté d'établir si un nœud est légitime ou non. En effet, les MANET, par définition, ne possèdent pas de point central pour agir à titre d'autorité d'authentification. Une des attaques contre les MANET très courantes dans la littérature est celle visant à compromettre un nœud ou à insérer un nœud malicieux dans le réseau afin de compromettre le protocole de routage. Dans ce type d'attaque, le nœud malicieux peut s'attaquer au processus de découverte de chemin :

- en modifiant un message de retour ;
- en envoyant de fausses routes aux autres nœuds ;
- en refusant de participer au processus de découverte de chemin.

Le nœud malicieux peut aussi s'attaquer au mécanisme de routage soit en modifiant le contenu des paquets qui lui sont envoyés soit en mettant de côté des paquets.

Attaques de déni de service. Les MANETs sont très vulnérables aux attaques de déni de service. Puisque le médium n'est pas protégé, il est possible de perturber les ondes pour empêcher un point d'accès de fonctionner. Aussi est-il possible, de façon plus traditionnelle, d'envoyer un grand nombre de demandes de route au nœud mobile en forgeant des paquets avec une mauvaise adresse de retour, forçant ainsi le nœud à épuiser ses ressources (calcul, bande passante, etc. . .).

Interception de trafic. Bien que des techniques de cryptage peuvent répondre à plusieurs besoins en terme de sécurité tel que protéger le médium contre l'écoute et permettre l'authentification des usagers du réseau, les solutions actuelles sont loin d'être sans faille. Un exemple célèbre est l'encryption de type Wired Equivalent Privacy (WEP) pour le 802.11. Il a été démontré que ce protocole d'encryption est faillible et qu'avec les outils existants actuellement il est facile pour un attaquant de trouver la clef WEP et ainsi d'écouter sur le canal. Si un attaquant a accès au canal, il lui est possible d'avoir un accès aux ressources du système ou de lancer des attaques plus sophistiquées comme *Man-in-the-Middle*, *Session High-Jacking* ou *Replay Attack* par exemple. Le fait que le médium soit exposé présente donc une vulnérabilité accrue au MANETs.

En conclusion, les MANETs sont vulnérables parce qu'ils opèrent sur un médium de communication ouvert (les ondes), que la topologie du réseau est dynamique, les algorithmes sont coopératifs, qu'ils ne possèdent pas de point centralisé de surveillance et de gestion, et parce qu'ils ne possèdent pas de défenses bien définies.

2.3 La détection d'intrus dans les réseaux

Dans la section précédente, nous avons identifié certaines des menaces auxquelles les systèmes informatiques peuvent être confrontés. Afin d'y faire face, quelques outils sont à la disposition du gestionnaire de sécurité, comme par exemple, le pare-feux. Un pare-feux n'est, malheureusement, pas suffisant pour nous protéger de toutes les menaces. Un IDS est un outil indispensable pour protéger les infrastructures informatiques, car il permet, à l'instar des pare-feux, de pouvoir détecter la fraude et les abus commis par des utilisateurs légitimes d'un système. Par ailleurs, il permet d'identifier des schémas complexes d'attaques dont les actions peuvent s'échelonner sur de longues périodes dans le temps. De plus, comme nous le verrons en détail, certaines actions malicieuses distribuées dans l'espace (ayant lieu sur plusieurs hôtes différents) et le temps ne peuvent être repérées que par un IDS distribué.

2.3.1 Les IDS

Depuis l'introduction du concept de détection d'intrus en 1980 par Anderson (1980) et raffiné par Denning et Neumann (1985) et Denning (1987), deux princi-

paux types de systèmes de détection d'intrusion ont fait leur apparition : les systèmes de détection au niveau des hôtes et les systèmes de détection au niveau du réseau.

Détection au niveau de l'hôte. Les systèmes de détection d'intrus locaux (HIDS) collectent de l'information spécifique à un hôte en particulier, habituellement au niveau du système opératif. Cette façon de faire a l'avantage de permettre de collecter de l'information de grande qualité, directement à la source. Malheureusement certaines attaques ne peuvent être détectées à un endroit unique dans un réseau comme les balayages de ports par exemple. Certaines intrusions distribuées peuvent ne laisser que peu de traces individuellement à chaque hôte mais lorsque l'information provenant de plusieurs sources est corrélée, il est possible d'identifier une intrusion. Par ailleurs, dans le cas de vers ou de *telnet chain*, il est plus facile de les repérer quand on prend en considération l'information provenant de plusieurs hôtes.

Détection au niveau du réseau. Les systèmes de détection d'intrus situés au niveau du réseau (NIDS) surveillent particulièrement les paquets qui circulent sur le réseau grâce à des cartes configurées en mode « espion » seulement. Ils ont la possibilité de surveiller ainsi un ensemble d'hôtes mais éprouvent de la difficulté lorsque les communications sont chiffrées. Par ailleurs, dépendamment de l'architecture du système, ils peuvent être victimes de problèmes d'extensibilité lorsqu'il y a un nombre important d'hôtes ou que la charge sur le réseau est élevée.

2.3.2 Les méthodes de détection d'attaques

Les techniques employées par les IDS pour la détection d'intrus dans les réseaux sont classées principalement en deux catégories : la détection d'abus et la détection d'anomalies.

Détection d'abus.

Dans le cas de la détection d'abus, les comportements malicieux sont caractérisés et exprimés sous forme de règles, dans une forme compréhensible de la machine et l'IDS vérifie l'existence d'un tel comportement en faisant usage d'une méthode déterministe. Le travail de spécifier les comportements malicieux est laissé à la discrétion des développeurs de IDS et des utilisateurs du système.

Un exemple d'un tel système est SNORT (Roesch, 1999). SNORT est configuré à l'aide d'une base de données de signatures qui caractérisent les paquets potentiellement malicieux. Grâce à cette base de données, SNORT surveille un réseau et enregistre toutes les apparitions de paquets qui correspondent à une signature.

Le plus grand problème de ce type de détection est qu'il n'est pas possible de détecter des attaques dont le système ne possède pas la signature.

Exemples de systèmes de détection d'abus

Dans les paragraphes qui suivent, nous présentons d'autres façon de faire de la détection d'abus. Ce court résumé est tiré de (Kruegel et Toth, 2000).

Système Expert. Le système expert enregistre des règles de type `if-then-else` dans une base de donnée. Le côté gauche (`if`) de la règle définit les pré-conditions requises pour une attaque. Quand toutes les conditions sont remplies, la règle déclenche les actions à prendre qui apparaissent du côté droit de la règle (`then`). Ceci peut entraîner le déclenchement d'autres règles ou l'identification d'une intrusion. Ce type de système a l'avantage de séparer la logique de contrôle du problème. Toutefois, avec cette façon de faire il est difficile de détecter des marques d'intrusions distribuées dans le temps.

Système de raisonnement basé sur un modèle. Cette approche se base sur une base de données de scénarios d'attaques (enregistrés comme une séquence de comportements ou d'activités). À n'importe quel moment, le système considère un sous-ensemble de ces attaques comme ayant lieu en ce moment et tente de vérifier ses hypothèses à l'aide de piste de vérification. Quand des preuves pour les attaques sont trouvées, les probabilités pour ces attaques augmentent, sinon elles décroissent. Quand les probabilités atteignent un certain niveau, le système conclut à une attaque et donne l'alarme. Ce modèle repose sur les théories mathématiques de raisonnement en cas d'incertitude. Malheureusement de tels modèles sont difficiles à construire.

Analyse des états de transition. L'analyse des états de transition demande la construction d'une machine à états finis. Les états de cet automate représentent différent état du système avec les transitions qui décrivent certains évènements qui entraînent des changements d'état dans le système. Un état du système peut représenter

l'état de la pile du protocole réseau, la validité et l'intégrité de certains fichiers ou d'un processus. Quand un intrus effectue des actions sur le système qui entraînent l'automate à atteindre un état correspondant à un danger pour la sécurité, une intrusion est rapportée.

Analyse des touches entrées par l'utilisateur. Cette technique enregistre les touches entrées par un utilisateur pour déterminer la possibilité d'intrusion. Ici, il est simplement question d'observer certaines séquences de clefs afin de détecter une attaque. Malheureusement, cette approche peut souffrir du fait qu'il peut exister la possibilité qu'une même activité peut être exprimée avec différentes séquences de clefs ou l'utilisation de macros non couverts par le patron de détection.

Détection d'anomalies

La détection d'anomalies fait référence à une approche où le système est entraîné à apprendre le « comportement normal » d'un réseau. Une alarme est déclenchée quand une déviation par rapport au comportement normal appris est observée. Théoriquement ce type de système est en mesure de détecter des attaques inconnues, contrairement aux systèmes basés uniquement sur la détection d'abus. Puisqu'une alarme est déclenchée à cause d'une déviation de comportement, il est parfois difficile de retracer la source de la déviation. Cette façon de faire repose en grande partie sur une analyse statistique du comportement du système, des heuristiques de détection, ainsi que sur des techniques d'intelligence artificielle telles que les réseaux neuronaux.

La détection reposant sur cette technique uniquement est généralement reconnue comme générant davantage de faux positifs / négatifs et d'être moins précise que les techniques de détection d'abus.

2.4 Les limites des IDS

Les solutions présentement disponibles pour la détection d'intrus dans les réseaux présentent plusieurs limites. Dans cette section nous détaillerons les limites les plus évidentes.

Les faux positifs. Un faux positif survient lorsqu'une alarme est déclenchée par un IDS à la suite d'un stimulus qui est en fait bénin. Ceci revient à dire que l'IDS

a fait une erreur. Un bon exemple est lorsqu'un IDS déclenche une alarme pour un *SYN flood* parce qu'un grand nombre de paquets `SYN` sont envoyés à un serveur web auquel un grand nombre de requêtes légitimes sont envoyées.

Les faux négatifs. Un faux négatif a lieu lorsqu'une condition qui mériterait le signalement d'une attaque survient mais qu'aucune n'est déclenchée. On peut retrouver un faux négatif lorsqu'un IDS ne réussit pas à détecter le parcours du répertoire tampon d'un serveur web parce que l'attaquant a réussi à développer une méthode jusqu'à présent inconnue pour obscurcir le nom du fichier qui est demandé.

Un autre exemple est lorsqu'un IDS ne réussit pas à capturer tous les paquets nécessaires pour rassembler toutes les actions d'un attaquant à cause d'une charge trop importante sur le réseau ou des changements dans la topologie de routage.

Les faux négatifs représentent un problème grave pour les IDS et ces derniers se doivent de minimiser le taux de faux positifs au maximum. Généralement un IDS se doit d'obtenir un très bas taux de faux négatifs pour les attaques qu'il est censé pouvoir détecter.

Fondamentalement, tous les IDS représentent un coup d'option entre être trop sensibles et générer trop de faux positifs ou de bruit et manquer de sensibilité et générer des faux négatifs. Le taux de faux positifs tombe généralement rapidement lorsque davantage d'information est ajouté pour décrire une attaque comme des états par exemple. Quand à lui, le taux de faux négatifs est inconnu, mais nous nous attendons à le voir augmenter plus les règles de détection deviennent précises et donc s'accommodent moins bien des variations dans les attaques.

Le Bruit. Selon (Ranum, 2003), est considéré comme bruit un IDS qui génère une alarme pour une condition qui n'est pas dangereuse ou non applicable au système sous surveillance mais qui est correctement diagnostiquée. L'IDS n'a pas fait d'erreur mais l'alarme a, dans bien des cas, une valeur douteuse. Comme exemple on retrouve lorsqu'un IDS identifie un débordement de tampon pour une machine Windows/Intel, sur un système Solaris/SPARC qui ne sera donc pas affecté par l'attaque !

Un autre exemple est lorsqu'un IDS à l'extérieur d'un pare-feux identifie correctement des activités de balayage hostiles dont l'administrateur sait pertinemment qu'elles ne pénètreront pas le pare-feux.

Bien que ce genre d'événements n'aient pas d'incidence sur le système à protéger,

ils ne devraient pas être ignorés pour autant. Un attaquant qui utilise une « mauvaise attaque » n'est pas moins dangereux pour autant et devrait être surveillé au cas où il trouverait un moyen de percer les défenses. Malheureusement ce genre de surveillance nécessite beaucoup de moyens. Il est possible pour un attaquant de lancer un nombre important de « mauvaises attaques » avec pour seule intention de faire diversion pour permettre à ses « vraies attaques » de réussir sans être détectées.

Zero-day attack. La pire des menaces est toujours celle que nous ne connaissons pas, les *zero-day attack* sont les attaques qui exploitent une faille inconnu des concepteurs du système. Il existe quelque part une menace à un système sous surveillance qui n'a pas encore été identifiée et qui se confond avec le fonctionnement typique du système. Ce genre d'attaque, représentent celles du futur contre lesquelles, un IDS ne peut rien. Ceci correspond à l'ultime limite des systèmes de détection.

Capacité de traitement versus le trafic sur le lien. La capacité de traitement de l'IDS représente aussi une limite. En effet, lorsque le trafic sur le lien dépasse la capacité de traitement de l'IDS, ce dernier laisse passer des paquets potentiellement dangereux qui ne seront pas analysés.

Attaques avec états versus sans état. Dans plusieurs cas, la capacité de détection d'un IDS est limitée par sa mémoire, autrement dit, sa capacité à se souvenir dans le temps d'événements ayant eu lieu sur plusieurs hôtes pour pouvoir les corréler entre eux (Levchenko *et al.*, 2004). Bien que la détection de certaines attaques ne nécessite pas que le système se « souvienne » de plusieurs événements, certaines nécessitent que le système conserve en mémoire des états précédents pour la détection de certaines intrusions. Dans cette catégorie d'attaque on retrouve les *balayages de ports* et le *TCP SYN Flooding*. Cette mémoire nécessaire constitue donc une limite au système.

2.5 Les architectures des IDS

Dans cette section nous décrirons les architectures de IDS présentées dans les travaux de recherche.

2.5.1 Systèmes Hiérarchiques

Une architecture hiérarchique distribuée représentée à la Figure 2.1 peut être décrite comme étant structurée à la manière d'un arbre avec un système de gestion et de contrôle à son sommet, des unités d'agrégation d'information comme nœuds internes et des unités d'opérations comme nœud à ses extrémités. Les unités d'opérations peuvent être des senseurs de réseaux, des IDS locaux, des détecteurs de virus ou tout systèmes de réponses aux attaques.

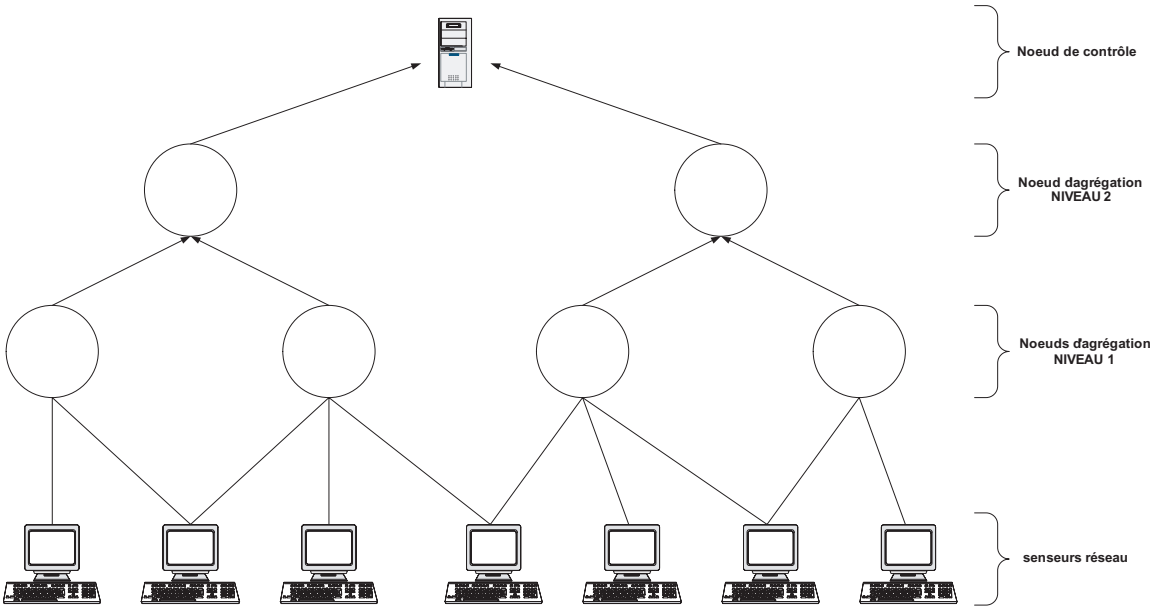


FIGURE 2.1 Architecture hiérarchique

Dans ce type d'architecture, la détection des attaques et la collecte d'information a lieu aux extrémités de l'arbre. L'information est envoyée à un nœud interne qui rassemble l'information provenant de plusieurs nœuds. L'information est ainsi agrégée, abstraite et réduite à des niveaux supérieurs dans la hiérarchie pour éventuellement atteindre le nœud racine de l'arbre. Le nœud racine effectue l'analyse de l'information et prend des décisions quant à la nature de la menace et prépare une réponse appropriée. Habituellement l'unité de gestion et de contrôle avise un opérateur humain de la situation pour que ce dernier puisse manuellement prendre action.

Avantages. La principale force de ce système est de posséder un degré acceptable d'expansion avec un point central d'administration.

Exemple – Dans cette catégorie nous retrouvons entre autre GrIDS (Staniford-Chen *et al.*, 1996) et AAFID (Spafford *et al.*, 2000). AAFID peut parraître un choix surprenant étant donné qu'il soit conçu selon le paradigme par agents mais il se trouve que les agents en questions sont autonomes et non mobiles et communique à l'intérieur d'une hiérarchie.

Inconvénients. Malheureusement, plusieurs critiques peuvent être adressées à ce type de système. Puisque l'information doit être acheminée vers le nœud racine, beaucoup de bande passante est utilisée pour transmettre les messages vers l'unité centrale. Cette dernière peut aussi manquer de ressources pour traiter l'information lorsque la charge sur le réseau est élevée et « manquer » des évènements, (Zhicai *et al.*, 2004). Si l'unité centrale de prise de décisions, le nœud racine de l'arbre, en vient à être compromis ou mis hors service, alors tout le système est compromis. Il est proposé dans Mell et McLarnon (1999) de remédier au problème en encapsulant les composantes du système de détection à l'intérieur d'agents mobiles qui se déplacent de façon aléatoire à l'intérieur du réseau sous surveillance afin d'être plus difficiles à cibler par une attaque. Par ailleurs les agents mobiles, si compromis, peuvent être détruits et ressuscités pour reprendre leur place dans le système. Marks *et al.* (2004) font remarquer que le modèle hiérarchique a toutefois des limites et chaque niveau d'agrégation impose un coût supplémentaire sur le système et ralentit ainsi la prise de décision, ce problème devient évident pour les systèmes de plusieurs milliers d'hôtes à surveiller.

2.5.2 Systèmes *Peer-to-Peer*

Afin d'adresser les manques des systèmes hiérarchiques, la solution « peer-to-peer » (P2P) a été envisagée dans la littérature. Chaque hôte opère un IDS local et un gestionnaire de sécurité qui peut échanger des informations à propos des menaces grâce à un système d'échange de messages avec les autres hôtes. Ramachandran et Hart (2004) ainsi que Janakiraman *et al.* (2003) présentent des prototypes de systèmes de détection d'intrus basés sur des agents autonomes ou mobiles mais où le partage de l'information entre les hôtes s'appuie sur les mécanismes des réseaux P2P. Dans les architectures proposées, les agents mobiles servent à vérifier l'état des entités voisines pour en vérifier l'intégrité à l'aide, entre autre, de « checksum » . Quand un

agent détecte qu'une entité a été compromise, il retourne vers son entité d'origine et un vote est déclenché. Dans ce genre de système, chaque entité est responsable de sa propre sécurité et surveille ses voisins. Le réseau P2P sert de réseau fiable de transport, pour les messages ou les agents entre les entités et sert à la découverte de service.

Avantages. Un tel système se démarque des architectures hiérarchiques parce qu'il n'a pas de centre de coordination central. Par ailleurs, Ramachandran et Hart (2004) font remarquer qu'un tel système est facile à mettre à jour. Dans Janakiraman *et al.* (2003) ils avancent l'idée de pouvoir distribuer la charge de la détection dans un tel système sans entrer dans les détails.

Inconvénients. Malheureusement, dans les modèles proposés par Ramachandran et Hart (2004) et Janakiraman *et al.* (2003), les systèmes proposés ne sont qu'à l'état embryonnaire et aucun vrai résultat ne fut publié en rapport avec la validité de leur modèles. Dans les systèmes présentés, il n'est aucunement question de mécanismes de corrélation de l'information.

2.5.3 Systèmes par Agents Mobiles

Un agent mobile peut être défini comme un programme qui peut exercer l'autorité d'un individu ou d'une organisation, peut travailler de façon autonome pour atteindre un but et peut rencontrer et interagir avec d'autres agents. Un agent mobile comprend le code et les informations d'états nécessaires pour exécuter un travail et nécessite une plate-forme pour lui fournir un environnement dans lequel il peut opérer. Les agents mobiles ont la capacité d'interrompre leur travail sur une plate-forme et de se déplacer vers une autre où ils peuvent continuer leurs tâches.

Tel que précédemment mentionné, les architectures hiérarchiques et par agents autonomes possèdent plusieurs problèmes. Afin de leur apporter une solution, la communauté scientifique s'est tournée récemment vers le paradigme par agents mobiles pour la détection d'intrus dans les réseaux.

Avantages. Les agents mobiles offrent les avantages suivants (Zhicai *et al.*, 2004) :

- les fonctionnalités et la sécurité des IDS sont décentralisées de façon à éviter les goulots d'étranglements au niveau du calcul et pour éviter la création d'un

point unique d'échec ;
- les modules d'un IDS sont déployés sur les nœuds formant le réseau de façon à réduire le transfert de données et ainsi réduire la charge sur le réseau ;
- ils permettent de détecter les intrusions au lieu où elles se produisent ;
- ils permettent une grande flexibilité et adaptation aux changements d'environnement. Faire face à une nouvelle menace signifie, dans bien des cas, avec des agents mobiles, de n'écrire qu'un nouvel agent.

Inconvénients. Il va sans dire que les agents mobiles présentent quelques vulnérabilités, entre autres :
- les communications entre agents doivent être surveillées et authentifiées de façon à s'assurer de l'intégrité et de la confidentialité des communications entre agents ;
- une plate-forme d'agents compatible doit être présente sur les hôtes pour recevoir les agents ;
- la plate-forme et les agents mobiles peuvent être victimes d'attaques ; aussi des mécanismes doivent-ils être mis en place pour déterminer si la plate-forme a été modifiée et le cas échéant, des mécanismes de restauration doivent être mis en place ;
- Déplacer du code sur le réseau peut s'avérer moins performant que des solutions statiques qui n'ont pas besoin d'être déployées lorsque l'application roule.

2.6 Travaux antérieurs sur les agents mobiles pour la détection d'intrusions

Ces dernières années, nous avons assisté à une augmentation des travaux sur la détection d'intrus dans les réseaux à l'aide d'agents mobiles. Plusieurs auteurs ont suggéré des architectures, autant dans le domaine filaire que dans le monde sans fil. Certaines de ces solutions peuvent être adaptées pour les deux domaines. Toutefois, les recherches pour le domaine sans fil se sont davantage consacrées aux attaques contre les protocoles de routage et les attaques de dénis de service. Dans les travaux présentés, peu d'auteurs donnent des résultats quant à l'efficacité de leur plate-forme et, jusqu'à présent, aucune comparaison n'a été faite entre les plate-formes d'agents

mobiles proposées et des IDS commerciaux existants sur le marché, de façon à vérifier de façon expérimentale le principe des agents mobiles comme façon valable de détecter des intrusions. Par ailleurs, dans les recherches présentées, la méthodologie de test n'est que bien peu élaborée lorsqu'il en est question.

La section qui suit vise à présenter certaines des plate-formes d'agents récemment proposées et à présenter certains détails lorsque cela est possible.

2.6.1 Les architectures dans le domaine filaire

La plupart des plate-formes d'agents se divisent principalent en deux grandes catégories nommées dans Yanxin *et al.* (2004) comme celles à *déplacement radial* et à *déplacement horizontal*. Dans les architectures à déplacement radial tel que APHIDS (Agent Programmable Hybrid Intrustion Detection System) et MAIDS (Mobile Agent Intrusion Detection Systems), les agents mobiles sont utilisés comme mécanisme de transmission de requêtes. C'est-à-dire qu'un agent mobile est envoyé de l'hôte A vers l'hôte B pour y effectuer une certaine tâche (dans la plupart des cas de la collecte d'information) et revient vers son hôte d'origine avec les résultats de la tâche. Dans le cas d'un déplacement horizontal comme pour MAIDS, les agents mobiles se déplacent d'hôte en hôte sans devoir rapporter leurs informations à un hôte d'origine.

Dans le cas d'architectures à déplacement radial, on retrouve souvent des composantes de détection locale (HIDS) qui, lorsqu'elles détectent des anomalies au niveau local, décident d'envoyer des agents mobiles « enquêter » sur d'autres hôtes pour en apprendre davantage sur l'événement. Cette façon de faire a comme désavantage de rendre la surveillance dépendante de la notion locale et pas nécessairement uniforme entre les hôtes de ce qu'est une anomalie. Autrement dit, l'hôte avec les niveaux de tolérance aux « anomalies » les plus élevés devient le maillon faible du réseau, puisqu'il déclenche le processus d'enquête après un plus grand nombre d'incidents.

Parmi les architectures présentées on retrouve des architectures hybrides telles que SPARTA (Security Policy Adaptation Reinforced Through Agents) illustrée à la Figure 2.2 qui utilisent un déplacement horizontal pour assurer la surveillance du réseau et un déplacement radial (à l'aide d'agents de support) lorsqu'une anomalie est détectée sur un hôte en particulier et qu'une enquête doit être déclenchée.

APHIDS. L'architecture APHIDS (Deeter *et al.*, 2004) est une variation des plate-formes de détections basées sur des agents mobiles et présente des similarités avec

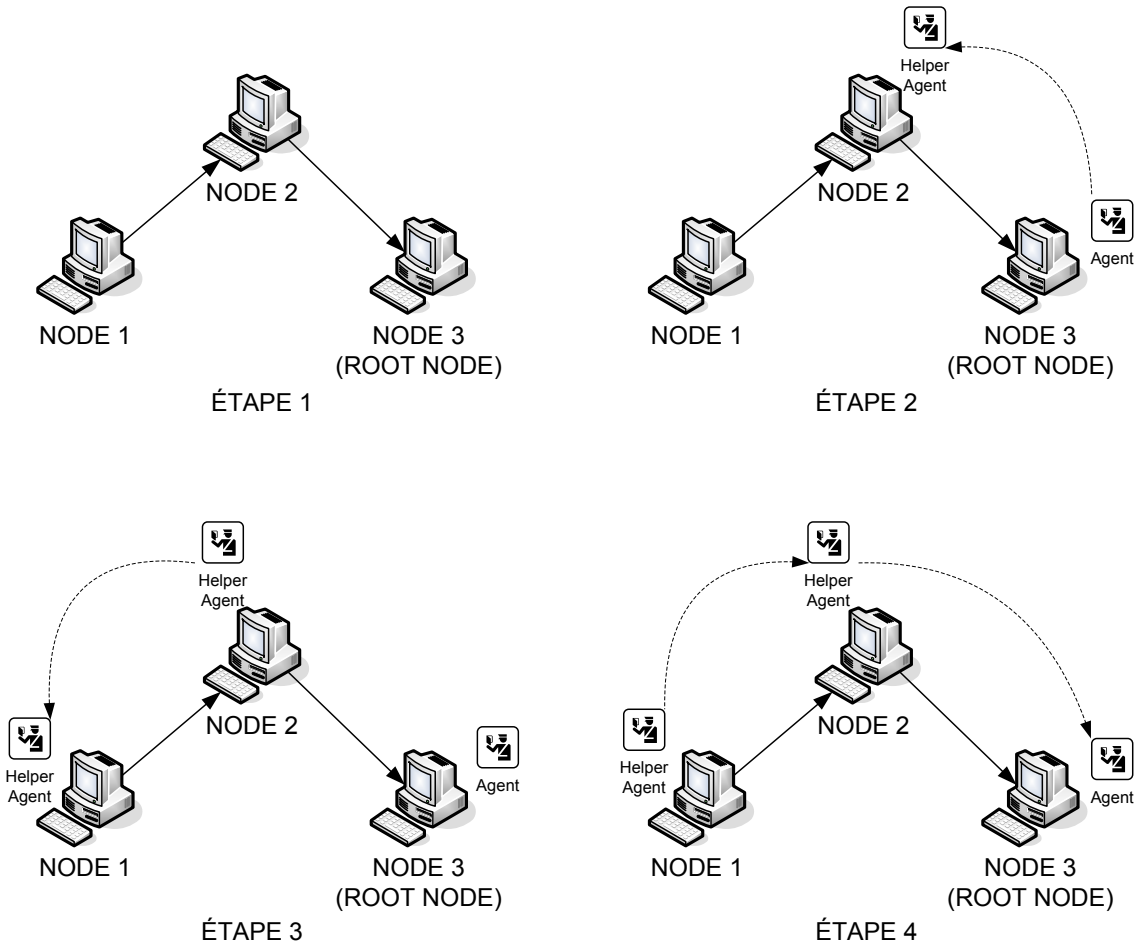


FIGURE 2.2 Déplacement d'agents mobiles avec SPARTA

SPARTA et MAIDS. APHIDS partage avec ces plate-formes le but commun d'exploiter la mobilité des agents pour effectuer de la corrélation d'événements de façon distribuée. APHIDS défini un langage de script pour spécifier les interactions entre les agents de surveillance et les agents d'analyse. Elle a pour but d'automatiser la collecte de preuves qu'un administrateur de système devrait faire manuellement dans d'autres circonstances.

Les agents mobiles sont utilisés pour inspecter les résultats contenus dans des fichiers de log, les états du système ou obtenus par d'autres IDS. Les agents mobiles utilisent le déplacement radial.

L'architecture APHIDS n'existe pour l'instant qu'à l'état de prototype et aucun

résultat n'a été publié.

SPARTA. SPARTA (Kruegel et Toth, 2001) fait usage des agents mobiles pour fournir au système des capacités d'interrogation pour reconstruire des patrons d'événements associées, distribués à travers plusieurs hôtes. Malheureusement, selon Percher et Jouga (2002), SPARTA nécessite la présence d'un nœud central dont le rôle est de maintenir une base de connaissance des nœuds présents dans le réseau. Cette caractéristique ne permet donc pas à SPARTA d'être déployé dans les réseaux sans fil ad hoc.

SPARTA, se base sur la plate-forme d'agents GYPSY et fait usage de son propre langage de spécification d'intrusions qui permet de générer automatiquement un filtre à partir d'une définition formelle du arachNIDS (Advanced Reference Archive of Current Heuristics for Network Intrusion Detection Systems). Par ailleurs SPARTA intègre un mécanisme de synchronisation des horloges des hôtes afin de pouvoir effectuer de façon précise la corrélation d'évènements. SPARTA fut testé entre autre contre l'attaque *telnet chain*. Dans Kruegel et Toth (2002) il est possible de constater que les résultats produits sont très peu nombreux et le nombre d'attaques employé pour les tests est très limité.

MAIDS. Les travaux de (Yanxin *et al.*, 2004) présentent une plate-forme d'agents mobiles qui utilise pour la détection d'activités d'intrusion l'analyse d'arbres logiciels de fautes (Software Fault Tree (SFT)) et de réseaux de Pétri colorés (Colored Petri Net (CPN)). Les travaux de cette équipe se concentrent présentement sur la génération automatique d'agents pour la détection à l'aide de SFT. Leur recherche propose de définir sous forme de SFT des patrons d'attaques qui seront automatiquement converti en CPN pour ensuite être compilé en un ensemble d'agents mobiles collaboratifs. Dans cette recherche il est ici question d'une hiérarchie d'agents qui collaborent entre eux pour la détection. Leur architecture s'appuie sur la plate-forme d'agent Voyager de Helmer *et al.* (2003) et les agents se déplacent de façon horizontale. Leur recherche présente des résultats pour l'attaque *FTP bounce*, et le *Trinoo DDoS*.

Autres plate-formes dignes de mention

Parmi les autres plate-formes digne d'être mentionnées, on retrouve :

MA-IDS Dans l'architecture MA-IDS, il est question d'une architecture par agent mobile avec déplacement radial des agents. Dans cette architecture, les agents mobiles ne servent qu'à la collecte des informations, ils ne contiennent pas de logique de détection, celle-ci étant contenue dans un « *MANAGER* » auxquelles les agents mobiles se rapportent. Le prototype présenté fait usage de la plate-forme d'agents GYPSY et les tests ont été effectués avec quatre stations connectées entre elles. (Li *et al.*, 2004)

Micael. Micael (de Queiroz *et al.*, 1999) est une plate-forme d'agents mobile à déplacement horizontal où chaque agent collecte de l'information et réagit aux signes d'intrusions. Toutefois, les agents mobiles ne sont déployés que lorsqu'une anomalie est détectée au niveau local.

JAM. JAM (Stolfo *et al.*, 1997) se distingue car il fait appel à des notions d'intelligence artificielle afin de concevoir un modèle évolutif de détection d'anomalies et de comportements intrusifs. Ici, il est question d'agents mobiles qui échangent de l'information entre eux et apprennent. Dans le cas de JAM, des agents locaux apprennent comment détecter la fraude.

IDA. IDA (Intrusion Detection Agent system) (Asaka *et al.*, 1999) fonctionne en se concentrant sur des événements spécifiques qui peuvent être des signes d'intrusions appelés *Marks Left by Suspected Intruder* (MLSI). Le système possède une structure hiérarchique avec une unité de contrôle central à son sommet et une variété d'agents à sa base. Lorsqu'un agent local détecte un MLSI, il en avertit le contrôle central qui envoie des agents de recherche jusqu'à l'hôte en question. Les agents de recherche envoient à leur tour des agents de collecte d'information pour collecter de l'information sur d'autres hôtes. L'information est par la suite renvoyée à l'unité centrale pour fin d'analyse.

D'autres architectures ont été aussi proposées dans la littérature mais sont peu référées par d'autres auteurs et témoignent de travaux moins matures. Parmi ceux-ci on retrouve Zhicai *et al.* (2004), (Dasgupta et Brian, 2001) et (Bernardes et dos Santos Moreira, 2000). Dans Guangchun *et al.* (2003) on introduit le concept de distribution de la charge pour l'analyse de détection d'intrus. Dans Li *et al.* (2004) on met l'emphase sur la vérification de l'intégrité des fichiers et sur le balayage de ports pour trouver des trojans.

2.6.2 Les architectures de IDS pour les réseaux ad hoc

Le domaine sans fil comporte les mêmes vulnérabilités que le domaine filaire et même davantage comme il a été précédemment expliqué à la section 2.6.1. Quelques recherches ont été effectuées sur la détection d'intrus dans les réseaux ad hoc à l'aide d'agents mobiles. La plupart des publications sur le sujet proposent des architectures semblables à celles présentées pour le domaine filaire mais bien souvent avec un ensemble de capacités réduites. L'ensemble des recherches s'attardent sur les attaques contre les protocoles de routage et s'intéressent particulièrement à la notion de nœuds compromis, autrement dit à un nœud dont le comportement est anormal et qui compromet ainsi le fonctionnement du réseau. Très peu de recherches couvrent d'autres attaques. Ceci peu s'expliquer souvent par la non utilisation d'une base de données de signatures d'attaques par exemple, dans les architectures proposées, ou par le besoin de réduire les composantes du système au maximum pour réduire les besoins en ressources. Dans un réseau ad hoc, une base de données d'attaques devrait être instanciée sur chacun des nœuds du réseau, ce qui représente un coût important dans un contexte ad hoc. Par ailleurs, les contraintes inhérentes aux réseaux ad hoc empêchent d'importer les architectures du domaine filaire qui comportent des unités centrales de traitement ou de coordination, ou des répertoires de services. (Mishra *et al.*, 2004)

Dans cette section nous tenterons de présenter quelques architectures faisant usage d'agents mobiles présentes dans la littérature.

Local Intrusion Detection System (LIDS)

Dans Albers *et al.* (2002), on propose un système de détection d'intrus reposant sur des IDS locaux à chaque nœud mobile (LIDS). Les LIDS collaborent entre eux à l'aide d'agents mobiles pour échanger de l'information lors d'intrusions ou pour enquêter sur des tentatives d'intrusion. Ici, les agents mobiles sont utilisés pour lancer des requêtes sur d'autres nœuds et ramener la réponse vers le nœud d'origine. Le LIDS accède aux données du réseau via SNMP et la MIB des nœuds. L'interopérabilité entre les LIDSs est assurée par l'usage de format de données d'alertes standard IDMEF (Intrusion Detection Message Exchange Format) (Curry *et al.*, 2006) et du protocole de transport IDXP (Intrusion Detection Exchange Protocol) (Feinstein *et al.*, 2003).

Détection d'intrus basé sur une base de donnée statique et stationnaire

Dans Smith (2001) on propose une architecture comprenant une surveillance de chaque nœud au niveau local par un agent et l'usage d'une base de donnée sécuritaire comme répertoire contenant les signatures d'attaques disponibles à tous les nœuds grâce à des agents mobiles. Cette architecture s'appuie sur la prémisse que la base de donnée ne sera pas compromise, ce qui est difficile à démontrer.

Détection d'intrusions à l'aide d'agents mobiles

Kachirski et Guha (2002) ont proposé une architecture pour la détection d'intrusions de manière distribuée à l'aide d'agents mobiles. Ils se démarquent par une approche qui ménage la bande passante et qui entraîne un faible coût sur le système. Dans cette architecture, chaque nœud mobile est connecté à un senseur pour la surveillance locale et le nombre de nœuds affectés à la surveillance du réseau est restreint à un nombre minimum, dynamiquement élus. Leurs agents sont utilisés pour les tâches de détection. Ici les agents mobiles sont utilisés pour relocaliser les composantes du système de détection de manière dynamique. La prise de décision quant à une éventuelle intrusion se fait par vote entre les nœuds d'une même grappe.

2.7 Analyse des solutions d'IDS par agents mobiles

Dans les travaux ci-haut mentionnés on y présente, dans l'ensemble, différentes architectures de détection d'intrus dans les réseaux à l'aide d'agents mobiles. Ces travaux, souvent des preuves de validation de la conception, tentent d'ajouter des fonctionnalités et des stratégies à l'aide d'agent mobiles pour permettre une meilleure détection. Malheureusement, aucune de ces recherches, quand elles présentent des résultats, ne s'attarde à vraiment considérer la performance dans son cadre le plus large. En effet, dans les recherches auxquelles nous avons fait référence précédemment, il est bien difficile de savoir à quoi attribuer les résultats obtenus. À une architecture révolutionnaire ou tout simplement à des conditions expérimentales favorables à la plate-forme présentée ? À partir des articles consultés il est souvent difficile de le savoir et donc, d'accorder un mérite important à la recherche au delà d'introduire le lecteur à une nouvelle façon d'aborder la détection d'intrus. Aucun des travaux

présentés n'arrive à montrer clairement qu'il permet de réduire les taux de faux positifs et parvient en même temps à détecter efficacement les tentatives d'attaques. Par ailleurs, les travaux consultés ne discuttent pas de la supériorité ou de l'infériorité d'une stratégie de développement d'IDS sur une autre.

Maxion et Tan (2000) et Lee et Xiang (2000) posent des bases importantes pour la détection expérimentale d'intrus en établissant, entre autres, que l'environnement réseau dans lequel se déroule l'expérience a un lien direct avec les résultats obtenus. Leurs études démontrent que des variations de l'environnement entraînent des variations dans les résultats obtenus. Ils soutiennent qu'on ne peut s'attendre à ce qu'un détecteur d'anomalie, évalué sur les bases de ses performances sur un ensemble de données d'une certaine régularité, performe de façon similaire sur un ensemble de données de régularité différente.

Dans les recherches présentées plus haut, il n'est jamais question de l'environnement réseau dans lequel se déroulent les expériences. Aussi est-il, à toute fin pratique, impossible de répéter exactement les expériences présentées dans les articles. Par ailleurs, aucune recherche ne s'attarde à faire varier la régularité de l'environnement pour que le lecteur puisse constater ses effets sur les plate-formes présentées.

2.8 Objectifs de recherche

Les recherches visant à réduire le taux de fausses alarmes représentent une avenue de recherche fondamentale pour la détection d'intrus dans les réseaux. En effet, Axelsson (voir Axelsson (1999)) tend à démontrer dans son article que le facteur limitant les performances de détection n'est pas, comme on pourrait être tenté de le croire, l'habileté à reconnaître les comportements intrusifs, mais bien l'habileté du système à supprimer les fausses alarmes. Les recherches présentées précédemment tendent à montrer que l'usage d'agents mobiles pourrait résoudre un grand nombre de problèmes auxquel font face les IDS conventionnels. Toutefois, il est crucial de démontrer l'efficacité des plate-formes qui seront développés et particulièrement des facteurs qui font qu'une solution est meilleure et sous quelles conditions.

Les recherches ont démontré que l'environnement réseau dans lequel l'IDS fonctionne a une influence déterminante sur les résultats sans toutefois être en mesure d'identifier des facteurs environnementaux et leurs influences sur les résultats de détection.

Dans cette recherche, nous nous proposons de formuler un cadre expérimental à l'intérieur duquel l'évaluation des performances d'IDS par agents mobiles devrait être réalisée. Ce cadre devrait permettre de qualifier l'environnement dans lequel l'expérience se déroule ainsi que les caractéristiques des agents mobiles mis en cause. Par ailleurs, ce cadre devrait permettre de formuler des hypothèses quantitatives à propos des performances d'un système de détection d'intrus et de les confirmer sur des modèles environnementaux et comportementaux simplifiés.

Grâce à ce cadre, il sera possible ultérieurement de faire de la recherche pour tenter de découvrir les relations existantes entre certaines métriques de performance pour la détection d'intrus à l'aide d'agents mobiles et les facteurs environnementaux, ainsi que les caractéristiques des agents mobiles.

Au chapitre suivant, nous introduirons notre cadre théorique qui nous servira à modéliser l'environnement pour pouvoir évaluer les performances de différents systèmes de détection d'intrus.

CHAPITRE 3

SYSTÈME DE DÉTECTION D'INTRUS PROPOSÉ

Les recherches antérieures dans le domaine des agents mobiles appliqués à la détection d'intrus dans les réseaux ont permis, par des démonstrations de bien-fondé de la conception, de démontrer que ce paradigme de programmation est applicable au domaine de la sécurité informatique. Il peut répondre à plusieurs limites propres à la détection d'intrus dans les réseaux tel que les contraintes de bande passante ou de puissance de calcul.

Malheureusement, jusqu'à présent, il n'a pas été démontré de façon empirique que ce genre d'architecture est supérieure en terme de capacité de détection, aux architectures (hiérarchiques ou autres) déjà existantes. Autrement dit, nous n'avons pas encore de réponse quant à savoir si le paradigme par agent mobile permet d'abaisser le nombre de faux positifs ou d'abaisser le temps de réponse à une intrusion. Les agents mobiles peuvent-ils faire mieux et plus précis que SNORT, par exemple, et si oui, sous quelles conditions ? L'état de la recherche n'est pas encore en mesure de nous apporter une réponse. Néanmoins cette question est fondamentale et se doit d'être adressée le plus rapidement possible.

Dans ce chapitre, nous élaborons un cadre théorique à l'intérieur duquel une évaluation des performances d'un système de détection d'intrus à l'aide d'agents mobiles devrait se dérouler.

Nous présenterons d'abord notre théorie concernant la formulation d'hypothèses de performance et suivra ensuite une explication détaillée des facteurs à prendre en compte lors de la formulation d'hypothèses, soient les caractéristiques du système de détection d'intrus, les métriques de performances et les caractéristiques environnementales. Par la suite, grâce au modèle que nous aurons élaboré, nous formulerons des hypothèses de performance que nous tenterons de prouver. Ces hypothèses viseront à comparer trois IDS entre eux au niveau de leurs performances de détection.

Pour ce faire, nous modéliserons le comportement des différents algorithmes et de l'environnement de façon mathématique.

3.1 Énoncé du cadre théorique

Dans le chapitre précédent, nous avons présenté plusieurs travaux sur la détection d'intrus dans les réseaux à l'aide d'agents mobiles. Malheureusement, la plupart de ces travaux ont le défaut commun de présenter leurs résultats de test en terme absolu. C'est-à-dire qu'il est question, par exemple, du nombre d'intrusions détectées mais bien peu des conditions environnementales régnant au moment du test. Ceci a pour effet de rendre incomparables entre eux les résultats des différentes expériences puisque chacune de ces expériences s'effectue dans un environnement distinct.

Afin de palier à ce genre de problème, nous proposons que l'expression de résultat de performance s'effectue de la façon suivante :

> Une métrique de performance s'exprime comme un rapport entre la valeur de deux mesures effectuées où, seuls les critères des systèmes de détection d'intrus (voir équation 3.1) ou les critères de l'environnement (voir équation 3.2) vont varier.

Par ailleurs, il peut exister une relation entre la performance selon une métrique et une autre métrique (voir équation 3.3). Exemple : La qualité de détection peut dépendre de la taille de la mémoire allouée au programme.

Principe. De façon formelle, nous proposons de formuler la thèse de la façon suivante : Pour les systèmes de détection d'intrus à l'aide d'agents mobiles autonomes, lors de l'évaluation de la performance, les critères suivants, suffisants et nécessaires, doivent être considérés :

- les caractéristiques de l'environnement ;
- les caractéristiques du système de détection d'intrus ;
- les métriques de performances.

Soient :

m	Le nombre d'IDS
I	L'ensemble des IDS $I_1, I_2, ..., I_m$
n	Le nombre de caractéristiques des IDS
C	L'ensemble des caractéristiques d'un IDS $C_1, C_2, ..., C_n$
x_r^i	Une valeur pour une caractéristique C_r d'un IDS i
$\mathcal{P}(C)$	L'ensemble des sous-ensembles de C
$p_i \in \mathcal{P}(C)$	un ensemble de caractéristiques pour un IDS I_i
p	Le nombre de caractéristiques de l'environnement
E	L'ensemble des caractéristiques environnementales $E_1, E_2, ..., E_p$
y_t	Une valeur pour une caractéristique environnementale E_t
$\mathcal{P}(E)$	L'ensemble des sous-ensembles de E
$e_j \in \mathcal{P}(E)$	un ensemble de caractéristiques environnementales
q	Le nombre de métriques de performance
M	L'ensemble des métriques de performance $M_1, M_2, ..., M_q$
$m_l(x_r^i, y_t)$	la valeur d'une mesure d'une métrique de performance M_l faites sur un IDS i et un environnement t

Thèse :

$$m_l(x_r^i, y_t) \leq m_{l'}(x_{r'}^{i'}, y_t) \tag{3.1}$$

C'est-à-dire, un IDS, différent d'un autre par une caractéristique choisi, performe mieux qu'un autre pour une métrique donnée.

$$m_l(x_r^i, y_t) \leq m_{l'}(x_r^i, y_{t'}) \tag{3.2}$$

C'est-à-dire, pour une métrique donnée, une variation de l'environnement peut entraîner une variation des performances pour un même IDS.

$$m_l(x_r^i, y_t) = f\Big(m_{l'}(x_r^i, y_t)\Big) \tag{3.3}$$

C'est-à-dire, il peut existe un lien de corrélation entre une métrique et une autre.

Un système de détection d'intrus, au niveau conceptuel, se veut de façon générale, une application servant à poser des jugements sur un ensemble de stimulus dont il est à l'écoute, un peu à l'image du cerveau humain. Les jugements, les questions auxquelles peut répondre le système, dépendent des capacités de ce dernier. De la même façon qu'un aveugle ne peut se prononcer sur la couleur d'une fleur, un IDS est dépendant de ses fonctionnalités pour obtenir de l'information à propos de son environnement. Les « sens » de l'IDS limitent l'information qu'il peut obtenir de l'environnement et agissent à la manière de filtres, déformant la réalité absolue de l'environnement. Cette réalité absolue de l'environnement existe mais n'est percevable qu'à travers des outils et donc n'est jamais appréhendée complètement par l'humain et encore moins par l'IDS. Par ailleurs, le modèle cognitif que l'IDS se fait imposer, par des règles ou des modèles statistiques préétablis, ou qu'il se construit, dans le cas de modèles évolutionnaires réactifs, agit aussi comme filtre sur les réalités de l'environnement comme illustré à la Figure 3.1. Aussi devons nous toujours garder en tête que l'IDS travaille avec un portrait partiel de la réalité sous observation et que la richesse du modèle qu'il se construit limite ses performances.

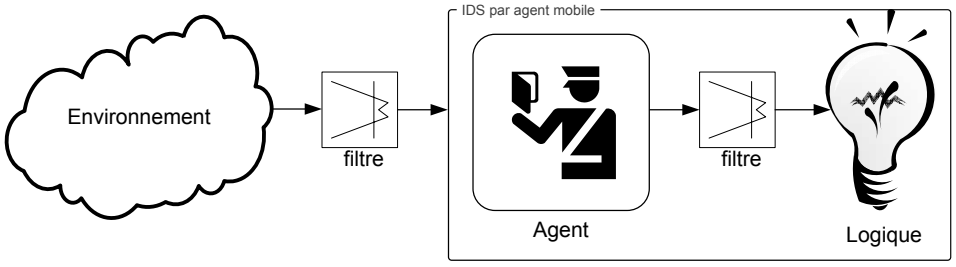


FIGURE 3.1 Filtrage de la réalité

Afin d'évaluer les performances d'un système de détection d'intrus, il est important d'être en mesure de pouvoir caractériser l'environnement dans lequel ce dernier opère ainsi que les capacités dont il dispose pour l'évaluer.

3.2 Caractéristiques environnementales

Il a été établi dans des recherches antérieures de Maxion et Tan (2000) et de Lee et Xiang (2000) que l'environnement dans lequel un système de détection d'intrus est appelé à opérer a un impact sur les performances de détection de l'IDS. Par

ailleurs, selon Maxion et Tan (2000), cet environnement est appelé à changer de façon dynamique au cours d'une même session et les résultats peuvent être aussi appelés à changer dynamiquement. Une modification de l'environnement de test par l'introduction d'un nouveau service Web sur le réseau, par exemple, perturbe le trafic observé par l'IDS et peut faire varier les résultats en induisant des faux positifs, par exemple, et en exposant le réseau à d'autres types d'attaques. Il est dont nécessaire, lors de l'expression de résultats de performances, de tenir compte des conditions environnementales durant l'expérimentation.

Il n'existe actuellement, à notre connaissance, aucune classification objective de l'environnement à des fins d'évaluation d'IDS. Aussi nous proposons une taxonomie qui a pour but de caractériser de façon objective l'environnement dans lequel est appelé à travailler un système de détection d'intrus.

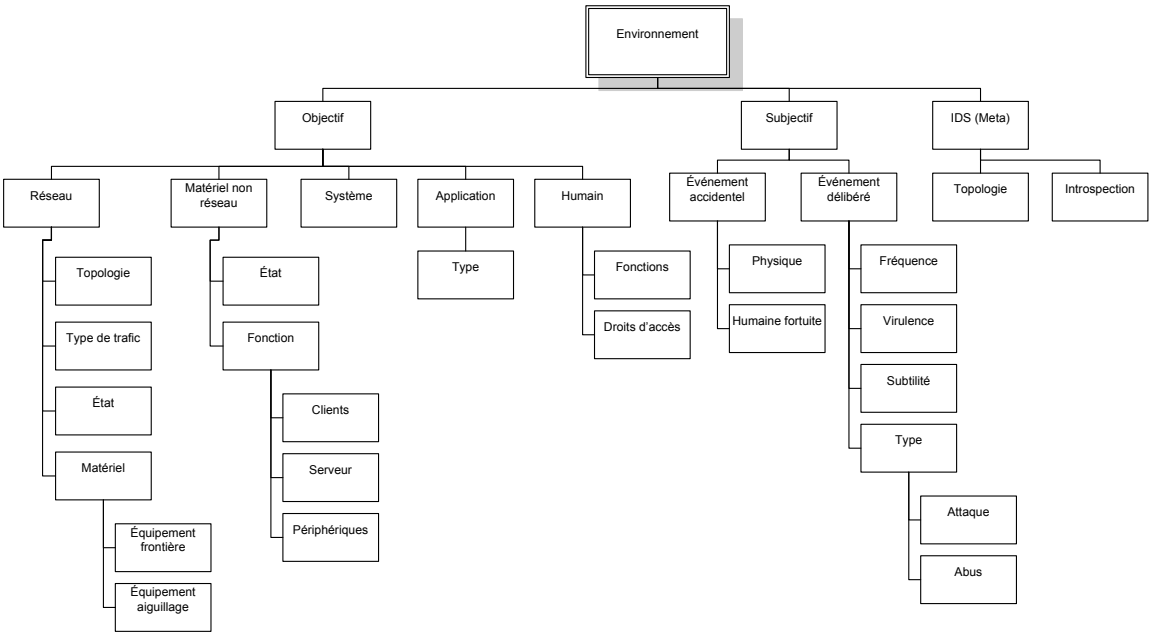


FIGURE 3.2 Taxonomie des caractéristiques de l'environnement

Nous avançons que la réalité environnementale peut être divisée en trois domaines distincts :

- la *réalité objective* ;
- la *réalité subjective* ;
- le domaine des *caractéristiques de l'environnement influençant l'IDS.*

3.2.1 La réalité objective

Nous définirons la *réalité objective*, comme celle qui peut être évaluée, qualifiée, quantifiée par un spécialiste connaissant bien le réseau. Par exemple un administrateur réseau. C'est à cette catégorie qu'appartient entre autre le trafic légitime.

Réseau. Ici il est question de la topologie du réseau sous surveillance, du matériel participant à son fonctionnement tel que les équipements frontières comme les pare-feux et les passerelles ainsi que les équipements d'aiguillage comme les routeurs, par exemple. Parmi les caractéristiques objectives de l'environnement nous retrouvons le type de trafic et de services circulant sur le réseau, TCP/IP, FTP, VoIP, etc... Aussi nous nous intéressons à l'état de l'appareil. Ce dernier est-il en fonction ou non ? Est-il occupé, en état de veille, etc... ? Par ailleurs, il est d'intérêt de connaître la consommation en ressources CPU, mémoire, bande passante, des différents équipements. Parfois, dans le cas de réseaux de senseurs par exemple, ces informations sont les seules dont nous disposons pour détecter les abus comme le déni de service par exemple. Afin de conduire une évaluation objective de l'environnement, il importe de connaître le médium à travers lequel l'information circule afin de pouvoir extraire des informations concernant les erreurs de retransmission, la latence, les pertes de paquets, etc... qui sont d'une grande importance pour évaluer le trafic gris par exemple.

Matériel Non Réseau. Le matériel non réseau est constitué de l'ensemble des machines sous surveillance présentes sur le réseau, autant PC qu'imprimantes, par exemple. Notre taxonomie s'intéresse particulièrement à la distinction qu'il est possible de faire au niveau fonctionnel entre les différentes composantes matérielles. La machine agit-elle à titre de client ou de serveur par exemple et si oui, à quel type de requête répond-t-elle et en provenance de qui ? Il est aussi question de l'état de l'appareil de la même façon que pour le matériel réseau.

Système. Ici il est question du système d'exploitation en particulier. La majeure partie des tentatives d'intrusions exploitent une faille spécifique à un système opératif en particulier. Il est donc important de connaître le système opératif, la version, les rustines appliquées, les services disponibles, etc...

Application. Les applications quant à elles peuvent fournir un point d'entrée, direct ou indirect, vers le système d'exploitation de la machine. Elles peuvent être exploitées pour permettre à un attaquant d'augmenter son niveau de privilège

ou de faire fonctionner du code malicieux. Aussi, au même titre que pour le système opératif, est-il important de connaître la version, le type et les rustines des différentes applications installées sur les différents systèmes. Par ailleurs, connaître les détails concernant les applications permet de caractériser le trafic qu'elles génèrent.

Humain. L'humain demeure sans doute la composante la plus variable d'un système. Ce dernier peut se caractériser entre autres par sa fonction à l'intérieur d'une organisation ainsi que par ses droits d'accès ou autres politiques administratives qui pourraient encadrer l'usage qu'il peut faire des ressources informatiques. En effet, les comportements de l'usager influencent beaucoup le trafic sur le réseau.

3.2.2 La réalité subjective

Nous définirons la *réalité subjective*, comme celle du monde de l'inconnu, dont font parties les menaces accidentelles et délibérées. L'IDS vise entre autre à discerner le trafic gris, du trafic noir et ainsi à faire passer le plus de trafic gris possible du côté de la réalité objective. En effet, l'environnement a tendance à confondre l'IDS. Un événement de trafic gris, lorsque comparé un à un, avec un événement de trafic noir, est identique à celui-ci. Toutefois, l'événement de trafic gris lorsque pris dans son context, peut être distingué de l'événement de trafic noir qui lui est malicieux et fait parti d'une attaque. Les IDS sont continuellement confrontés à un manque d'informations contextuelles sur les événements qu'ils doivent traiter, aussi doit-on tenir compte de la notion de trafic gris et de trafic noir.

Événement Accidentel. Une menace accidentelle peut être de nature physique comme une panne de courant, un court-circuit, un bris matériel, une erreur de transmission, de la latence sur le réseau ou elle peut être de nature humaine fortuite comme une mauvaise manipulation de la part d'un usager. Par ailleurs, le profil des événements peut être appelé à changer en fonction de l'usage qui est fait du réseau. Le taux d'utilisation du réseau, les périodes de pointes, de crises, les taux de retransmission, d'erreurs, etc. . . peuvent influencer les perceptions qu'a l'IDS de l'environnement. Ces événements influencent habituellement les taux de fausses alarmes signalées par un IDS.

Évènement Délibéré. Ici il est question de comportement étrange, suspect, attribuable seulement à un comportement malicieux et qui se caractérise souvent par

l'absence de confirmation quant à sa nature. Toutefois, il est souvent possible de qualifier la fréquence, la virulence, la subtilité ou l'effort mis pour dissimuler l'événement, ainsi que son type tel que attaque ou abus. Ces évènements font partie du trafic malicieux.

3.2.3 IDS (Méta)

Le troisième domaine objectif, celui des *caractéristiques de l'environnement influençant l'IDS*, en est un en soit, puisqu'il se surimpose à l'environnement tout en en faisant partie et possède ses caractéristiques propres.

Topologie L'environnement peut posséder la caractéristique d'accueillir un IDS. Dans le cas des IDS par agents mobiles, cela signifie que des plates-formes peuvent être installées sur des équipements du réseau sous surveillance et qu'elles peuvent accueillir des agents mobiles. La disponibilité des plates-formes et l'interconnexion entre celles-ci déterminent la topologie (disposition des senseurs) du réseau de surveillance.

Capacité d'introspection Cette caractéristique représente la capacité de l'environnement de fournir de l'information sur lui-même à l'IDS. Cette caractéristique conditionne le type d'information qu'il est possible d'obtenir d'un réseau sous surveillance.

3.3 Caractéristiques des systèmes de détection d'intrus à l'aide d'agents mobiles

Bien qu'il existe différentes taxonomies de systèmes de détection d'intrus (Debar *et al.* (1999), et Besson (2003)), il n'existe présentement pas de taxonomie qui tienne compte des agents mobiles impliqués dans la détection d'intrusions. Afin de différencier les différents IDS par agents mobiles et identifier les caractéristiques désirables ou non pour atteindre certains objectifs de performances, nous proposons la taxonomie présentée à la Figure 3.3. Cette taxonomie s'inspire librement de Debar *et al.* (1999) et de Besson (2003) et ajoute des éléments de mobilité et de modèle cognitif.

Migration. Cette dimension concerne particulièrement les mécanismes de migration disponibles de l'agent mobile. L'agent fait-il une copie de lui-même ou se déplace-t-il ? Est-ce que ce déplacement ou copie se fait avec conservation des états

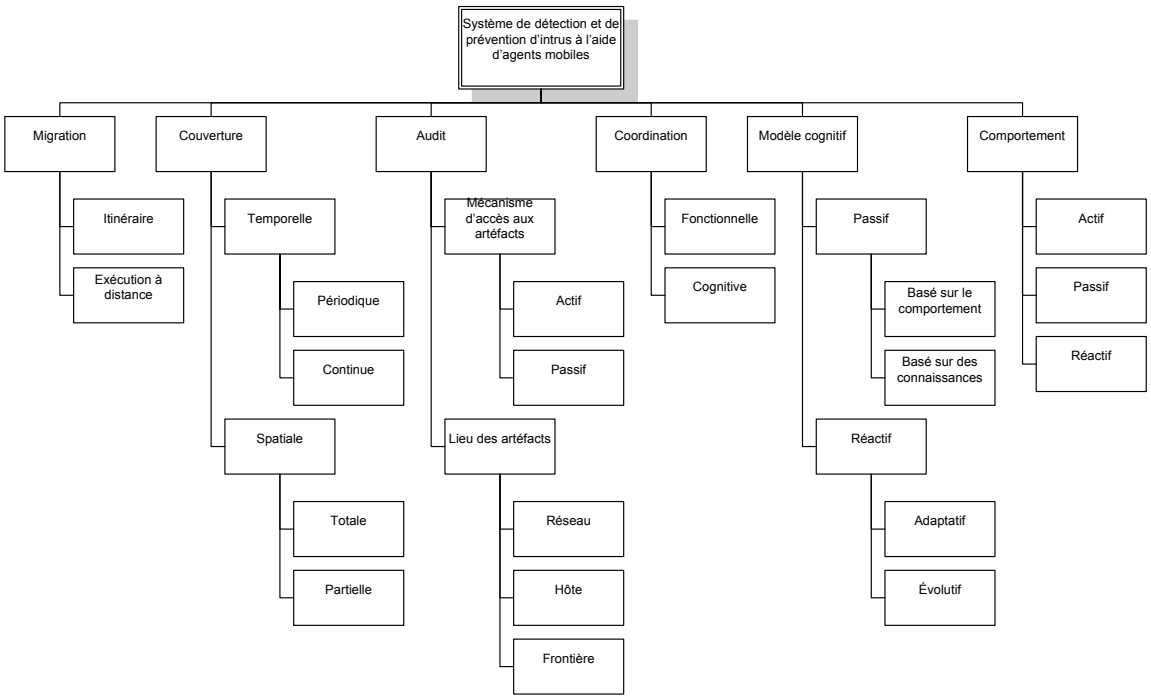


FIGURE 3.3 Taxonomie des caractéristiques de l'IDS

ou non ? Quel itinéraire est utilisé lors du déplacement de l'agent ? Comment procède-t-il à la découverte de son voisinage ?

Couverture. La couverture du système sous surveillance par l'IDS et ses agents mobiles peut être périodique ou continue dans le temps et peut être totale ou partielle dans l'espace. Différentes stratégies peuvent amener à faire varier la couverture, pour économiser des ressources ou de la bande passante, par exemple.

Audit. L'audit concerne les différents lieux où les agents mobiles peuvent obtenir de l'information pour enrichir le modèle cognitif ainsi que les différents mécanismes dont ils disposent pour les obtenir. Un agent mobile peut obtenir des informations différentes sur l'environnement en écoutant le réseau en tant que tel, en interrogeant les hôtes résidant sur le réseau et particulièrement en questionnant les équipements frontières. Cette collecte d'information peut se faire de façon passive ou active. **Passive** signifie que l'agent n'a pas à faire de requête pour obtenir l'information, cette dernière est directement disponible sous la forme de

journal (*log*), par exemple, sur le système où il se trouve. **Active** signifie que l'agent doit faire une requête et attendre une réponse. Ce dernier peut aussi être appelé à manipuler les données, les décrypter par exemple, avant de pouvoir les consommer.

Coordination. Cette dimension s'intéresse aux différents mécanismes disponibles à l'agent pour effectuer une synchronisation fonctionnelle. C'est-à-dire, les mécanismes servant à coordonner les agents entre eux, à s'assurer de leur bon fonctionnement et à assurer le respect des stratégies implantées. Il est aussi question de coordination cognitive, c'est-à-dire les mécanismes utilisés par les agents pour échanger de l'information servant à enrichir leur compréhension du système sous surveillance. Par exemple la coordination directe, orienté meeting, orienté tableau noir ou de type Linda (voir Cabri *et al.* (2000)).

Modèle cognitif. Le modèle cognitif, c'est la représentation interne de l'environnement par l'IDS et c'est sur ce modèle que ce dernier est appelé à porter un jugement. La logique qui anime l'IDS peut être passive ou réactive. Lorsqu'elle est passive, cette dernière est basée sur un ensemble de règles ou un modèle statistique immuable, par exemple. Dans le cas d'un modèle réactif, ce dernier est appelé à changer en fonction de l'environnement dans lequel il se trouve et des moyens à sa disposition pour accomplir sa mission. Un tel système peut démontrer des capacités d'apprentissage mimétique s'il y a transformation suite à l'acquisition d'expérience ou ce dernier peut faire preuve de capacité d'apprentissage génétique s'il y a sélection de comportements adéquats.

Comportement. L'aspect comportement d'un IDS s'intéresse particulièrement aux actions que peut poser un IDS. Un comportement passif signifie que l'IDS ne prend pas d'action suite à l'observation de tentatives d'intrusion. Au mieux, il se contente de faire rapport, à l'administrateur du réseau par exemple, de ses soupçons. Un comportement actif signifie que l'IDS tente d'aller plus loin et peut entreprendre des manœuvres pour découvrir la source d'une intrusion ou savoir l'étendue des systèmes affectés. Ce dernier ne prend toutefois pas d'action sur ses observations sauf celle d'enquêter davantage. Dans le cas d'un comportement réactif, l'IDS devient un système de prévention d'intrusions (Intrusion Prevention System (IPS)) et prend position face aux stimuli qu'il reçoit et peut décider de modifier son comportement pour mieux s'adapter à une menace, il peut reconfigurer des équipements sur le réseau tels que les pare-feux pour réagir

à la menace, il peut déplacer les agents mobiles sur les plates-formes, etc...

3.4 Métriques de performances

Les métriques de performance ont un rôle très important à jouer dans l'évaluation d'un système de détection d'intrus. Il peut exister un rapport entre les différentes métriques et l'optimisation du système. L'amélioration des résultats pour une métrique peut entraîner un impact négatif sur une autre. Par exemple, une économie d'espace mémoire d'un programme peut réduire la capacité de ce dernier à corréler des événements dans le temps ce qui peut entraîner une augmentation des taux de faux négatifs. Il importe d'évaluer un ensemble de métriques pour connaître les coûts réels associés aux modifications faites à un IDS.

Afin d'évaluer un système de détection d'intrus, nous proposons de l'observer sous trois angles, soit l'usage que fait ce dernier des ressources réseau, des ressources mémoire et des ressources de calcul. Nous proposons quelques métriques inspirées de Fink *et al.* (2002) afin de qualifier ces trois domaines.

1. **Réseau**
   - capacité de traitement en temps réel
   - trafic induit par l'IDS sur le réseau
   - capacité de traitement maximal
   - taux de faux positifs
   - taux de faux négatifs
   - discrimination
2. **Mémoire**
   - consommation de ressources mémoire
   - consommation de ressources disque
3. **Calcul**
   - usage de la puissance de calcul par l'IDS
   - réponse aux intrusions

3.4.1 Métriques de performance au niveau réseau

Dans cette section nous présentons des métriques afin d'évaluer les performances d'un IDS au niveau de son utilisation des ressources du réseau.

Capacité de traitement en temps réel. Si l'on considère une analyse, par l'IDS, qui ne tient pas compte des liens entre les paquets, la capacité de traitement en temps réel se définit comme le débit maximal d'acquisition (paquets/seconde) pouvant être analysé par l'IDS sans entraîner de retard par rapport à l'acquisition des paquets.

La capacité de traitement en temps réel mesure l'efficacité de l'IDS à analyser des paquets sans se soucier de leurs relations entre eux.

Trafic induit par l'IDS sur le réseau. Le trafic induit par l'IDS sur le réseau (moyen|maximal) correspond à un nombre de paquets par seconde introduit par l'IDS entre deux points du réseau. Ce trafic est causé entre autre par des activités de coordination inter agents, de déplacements des agents mobiles ou de requêtes d'information.

Le trafic induit par l'IDS sur le réseau mesure l'impact qu'entraîne la présence de l'IDS sur le réseau.

Capacité de traitement maximal. La capacité de traitement maximal se définit comme le débit de trafic, en paquets par seconde, qui entraîne l'arrêt ou le mauvais fonctionnement de l'IDS.

La capacité de traitement maximal définit la limite de fonctionnement de l'IDS avant que ce dernier ne puisse remplir adéquatement son rôle.

Taux de faux positifs. Le taux de faux positifs représente le rapport entre le nombre de fausses alarmes et le nombre total d'alarmes. Un IDS qui ne donnerait jamais l'alarme ou un IDS parfait qui détecte seulement les vraies intrusions obtiendrait un taux de 0%.

Taux de faux négatifs. Le taux de faux négatifs correspond au rapport entre le nombre de tentatives d'intrusions non détectées et le nombre total de tentatives d'intrusions.

Le taux de faux positifs et le taux de faux négatifs mesurent la précision de l'IDS et le degré de couverture qu'offre celui-ci.

Taux de détection. Le taux de détection correspond au rapport entre le nombre de tentatives d'intrusions détectées et le nombre total de tentatives d'intrusions.

Discrimination. La discrimination, illustrée habituellement par une courbe ROC (*Receiver Operating Characteristic*) exprime la relation entre le taux de détection et le taux de faux positifs. Cette courbe permet d'illustrer le coût en terme

de fausses alarmes que représente une amélioration des performances de détection. En effet, même si un haut taux d'intrusions est déterminé, il peut être difficile, voir impossible de les distinguer des fausses alarmes si le taux de ces dernières est trop élevé.

3.4.2 Métriques de performance au niveau de la mémoire

Dans cette section nous présentons des métriques afin d'évaluer les performances d'un IDS au niveau de son utilisation de la mémoire.

Consommation de ressources mémoire. Cette métrique décrit la quantité (moyenne|maximale) de mémoire consommée en un endroit par un IDS sur une période donnée.

La consommation de ressources mémoire mesure l'usage que fait l'IDS de la mémoire vive présente sur le système.

Consommation de ressources disque. Cette métrique décrit la quantité (moyenne|maximale) de mémoire disque consommée en un endroit sur une période donnée.

La consommation de mémoire disque est un indicateur entre autres de la capacité du système à faire des corrélations d'évènements dans le temps. Un débit important de trafic peut avoir pour effet de remplir rapidement la mémoire allouée à l'IDS sans pour autant permettre de reculer longtemps dans le temps. À l'inverse un faible débit peut permettre d'explorer le trafic sur une plus longue période de temps.

3.4.3 Métriques de performance au niveau des performances de calcul

Dans cette section nous présentons des métriques afin d'évaluer les performances d'un IDS au niveau de sa puissance de calcul pour analyser les intrusions potentielles.

Usage de la puissance de calcul par l'IDS Cette métrique représente l'usage (moyen|maximal) que l'IDS et/ou ses composantes font du processeur. Elle s'exprime en nombre de cycles utilisés sur le nombre total de cycle que peut fournir l'unité de calcul à un endroit donné.

Réponse aux intrusions Temps (moyen|maximal) entre la présence d'une intrusion dans le système sous surveillance et son signalement par l'IDS.

Le temps de réponse aux intrusions mesure le temps dont dispose un attaquant pour causer des dommages avant d'être détecté par l'IDS. Il est une mesure de l'efficacité du système de détection d'intrus.

3.5 IDS à l'étude et hypothèses de performance

Grâce à notre cadre théorique exprimé précédemment, nous sommes maintenant en mesure de formuler des hypothèses de performance en identifiant clairement les IDS et leurs caractéristiques concernées ainsi que l'environnement pour lequel les résultats s'appliquent.

Dans cette section nous présenterons trois IDS (appelés aussi stratégies ou algorithmes de détection) que nous décrirons selon la taxonomie proposée à la section 3.3. Nous formulerons des hypothèses de performances concernant les trois stratégies de détection à l'étude, puis nous décrirons l'environnement pour lequel nos hypothèses s'appliquent.

3.5.1 Stratégie de résolution DSSL

La stratégie de détection DSSL (Détection Standard avec Seuil Local), telle qu'exposée dans Northcutt et Novak (2002), peut s'exprimer de la façon suivante :

x événements de type y à l'intérieur d'une période de temps z en un lieu ω.

Afin de tester cette stratégie, nous utilisons un senseur logiciel installé sur chaque hôte qui, suivant le patron de l'observateur, surveille la base de donnée locale pour toute nouvelle entrée d'événements malicieux. Lors de l'ajout d'un nouvel événement, le programme compte les événements par type et adresse source. Si le nombre d'événements excède un certain seuil à l'intérieur d'une certaine période de temps, il y a intrusion et tous les événements du même type pour la même adresse IP à l'intérieur de la période de temps sont considérés comme des intrusions. Lorsque nous globalisons cet algorithme comme dans le cas de la surveillance d'un réseau, nous considérons qu'il y a une alarme sur le réseau lorsqu'une alarme survient sur un des hôtes sous surveillance.

3.5.2 Stratégie de résolution PSSG

La stratégie de détection PSSG (**P**atrouille de **S**urveillance avec **S**euil **G**lobal), décrite dans Fajardo (2005), introduit la notion d'agents mobiles et de surveillance globale de réseau.

Séquence de détection

Cette stratégie de détection fonctionne de la façon suivante :

1. L'agent Postman se déplace d'hôte en hôte suivant un ordre prédéterminé ;
2. Sur chaque hôte, l'agent communique avec la base de donnée locale et cherche des alertes correspondant à un type d'intrusion en particulier ;
3. Chaque événement rencontré dans la base de donnée est compté ;
4. Durant ses déplacements, si le compteur d'événements dépasse un certain seuil prédéterminé pour une adresse source donnée, l'agent signale la présence d'une intrusion à l'administrateur ;
5. Une fois le seuil atteint, chaque alerte ayant été enregistrée jusqu'à ce moment pour l'adresse IP en question est considérée comme étant malicieuse.

Caractéristiques de l'IDS

Selon la taxonomie des systèmes de détection d'intrus à l'aide d'agents mobile, ce système possède les caractéristiques suivantes :

Migration. Chaque agent mobile possède la capacité de se déplacer avec ses états vers un autre hôte. L'itinéraire de chaque agent mobiles est déterminé à l'avance et est fixe durant l'exécution.

Couverture. Au niveau de la couverture, l'agent mobile est présent de façon périodique sur chaque hôte et tous les hôtes sont visités de manière égale selon un algorithme dit de « round robin ». Son itinéraire est donc fixe, pré-déterminé et passe par tous les conteneurs d'agents mobiles.

Audit. L'agent mobile a accès aux données sur chaque hôte (les *log* d'alarmes) en se déplaçant sur l'hôte concerné et en effectuant une requête sur la base de donnée.

Coordination. La configuration actuelle n'implique qu'un seul agent, aussi ce dernier ne se coordonne qu'avec le service d'annuaire qui lui permet de localiser les autres plates-formes sur le réseau.

Comportement. L'agent a un comportement actif puisqu'il se déplace pour aller chercher son information, toutefois, suite à la découverte d'une tentative d'intrusion le comportement est relativement passif, l'agent ne se contente que d'alerter l'administrateur réseau.

Modèle Cognitif. Le modèle cognitif utilisé pour cet IDS en est un passif basé sur des connaissances, plus précisément des règles. En effet, l'agent (appelé Postman dans PSSG) fait usage d'une logique déterminée à l'avance et qui consiste à compter les événements suspects et à prendre une décision, après un certain temps, lorsque ce compte arrive à un niveau seuil.

3.5.3 Stratégie de résolution PSAR

Dans ce mémoire, nous proposons comme variante et amélioration aux deux stratégies précédente, la stratégie de détection PSAR (Patrouille de Surveillance avec Analyse Raffinée). Ce système de détection d'intrus est identique en tous points au système de détection d'intrus PSSG présenté précédemment à l'exception du modèle cognitif qui varie.

Modèle cognitif. En effet, la stratégie PSAR fait usage d'un modèle cognitif plus riche. Les attaques se trouvent à être corrélées dans le temps et dans l'espace alors que la stratégie PSSG ne propose que de corréler les informations dans l'espace.

Dans la stratégie PSAR, une intrusion se trouve à être décrite comme un ensemble d'événements suspects identiques ayant lieu sur plusieurs machines à l'intérieur d'un *interval de temps rapproché* et originant d'une même source.

Séquence de détection

1. L'agent appelé Sherlock se déplace d'hôte en hôte en suivant un ordre prédéterminé ;
2. Sur chaque hôte, l'agent communique avec la base de donnée locale et cherche des alertes correspondant à un type d'intrusion en particulier ;
3. Chaque événement rencontré dans la base de donnée d'un hôte est corrélé avec les événements survenus sur les autres hôtes à l'intérieur de la même période de temps ; Ici la période de temps de comparaision représente une fraction de

la période de temps de la patrouille. Le système peut tenter de corréler les événements ayant survenues à l'intérieur d'une période d'une seconde, par exemple entre la 3^{e} et la 4^{e} seconde de la patrouille de surveillance qui elle aura durée 30 secondes.

4. Si l'agent rencontre des événements survenus à l'intérieur de la même période de temps (petite) sur tous les hôtes, alors ces événements sont signalés comme intrusifs puisqu'ils correspondent à la signature recherchée.

3.5.4 Hypothèses de performance

Afin de valider le cadre théorique précédemment énoncé, nous testerons les hypothèses suivantes :

Hypothèse 1 : Nous émettons l'hypothèse que pour le paradigme de détection d'intrus PSSG, DSSL et PSAR, il existe une relation entre les taux de **faux positifs** et les taux de trafic gris et de trafic noir.

Hypothèse 2 : Nous émettons l'hypothèse que pour le paradigme de détection d'intrus PSSG, DSSL et PSAR, il existe une relation entre les taux de **faux négatifs** et les taux de trafic gris et de trafic noir.

Les hypothèses 1 et 2, visent à démontrer que, selon le cadre théorique énoncé à la page 30, l'environnement influence les performances d'un système de détection d'intrus. Pour un même IDS, une variation de l'environnement (taux de trafic gris et taux de trafic noir) entraîne une variation des performances. Ces hypothèses visent à vérifier, par exemple, que lors d'une grosse vente en ligne, l'achalandage d'un site web transactionnel augmente et le nombre de connexions à demi-ouverte de sources accidentelles augmente de la même manière. Aussi nous nous attendons à une augmentation des taux de faux positifs puisque l'IDS n'a pas pris en compte les changements au profil d'utilisation du site pour ajuster ses niveaux seuils. De la même manière, une attaque de type *Fast Scan* sur un site web où l'IDS est configuré pour un achalandage élevé peut possiblement passer sans être détectée.

Hypothèse 3 : Étant donné les hypothèses 1 et 2, nous émettons l'hypothèse que l'algorithme de découverte d'intrusion PSSG performe mieux que l'algorithme DSSL. C'est-à-dire que lorsqu'ils sont soumis à des conditions environnementales identiques et des seuils équivalents, le taux de faux positifs de PSSG est inférieur

au taux de faux positifs de DSSL (voir 3.4). Toutefois, nous soutenons que les taux de faux négatifs sont plus élevés avec l'algorithme PSSG qu'avec DSSL.

$$\beta_{\text{PSSG}} < \beta_{\text{DSSL}} \tag{3.4}$$

$$\alpha_{\text{PSSG}} > \alpha_{\text{DSSL}} \tag{3.5}$$

Hypothèse 4 : De la même façon que pour l'hypothèse 3, nous émettons l'hypothèse que, dans des conditions environnementales égales et avec des seuils équivalents, l'algorithme PSAR performe mieux que PSSG (voir 3.7 et 3.9) et il performe mieux que l'algorithme DSSL.

$$\beta_{\text{PSAR}} < \beta_{\text{PSSG}} \tag{3.6}$$

$$\beta_{\text{PSAR}} < \beta_{\text{DSSL}} \tag{3.7}$$

$$\alpha_{\text{PSAR}} < \alpha_{\text{PSSG}} \tag{3.8}$$

$$\alpha_{\text{PSAR}} < \alpha_{\text{DSSL}} \tag{3.9}$$

Les hypothèses 3 et 4, selon notre cadre théorique à la page 30, visent à démontrer que les caractéristiques d'un IDS ont une influence sur les performances telles qu'exprimées précédemment dans notre cadre théorique.

Afin de valider ces hypothèses, nous ferons varier l'environnement de façon à s'assurer que les hypothèses soient valides pour un ensemble de conditions environnementales. Précisément, nous soumettrons les trois algorithmes à différents taux de trafic gris et de trafic noir pour en observer l'influence relative sur les taux de faux positifs et de faux négatifs.

Hypothèse 5 : Nous émettons l'hypothèse qu'il existe un compromis entre les taux de faux négatifs et les taux de faux positifs pour les algorithmes PSSG (voir 3.10), DSSL (voir 3.11) et PSAR (voir 3.12) lorsque nous faisons varier le seuil de détection s.

$$\beta_{\mathrm{PSSG}}(s) = f\left(\alpha_{\mathrm{PSSG}}(s)\right) \quad (3.10)$$

$$\beta_{\mathrm{DSSL}}(s) = g\left(\alpha_{\mathrm{DSSL}}(s)\right) \quad (3.11)$$

$$\beta_{\mathrm{PSAR}}(s) = h\left(\alpha_{\mathrm{PSAR}}(s)\right) \quad (3.12)$$

Cette hypothèse vise à vérifier la notion de compromis, généralement bien acceptée, qui veut que lorsqu'on améliore les résultats en terme de faux positifs en faisant varier le niveau du seuil de détection, on assiste à une dégradation des performances par une augmentation des taux de faux négatifs et vice versa.

3.5.5 Description du processus d'intrusion et de détection

Un environnement réseau réel est sujet aux variations d'un très grand nombre de paramètres, aussi, dans le but de le modéliser et de vérifier nos hypothèses, nous en fixerons la majeure partie. Pour valider nos hypothèses, nous assumerons que l'environnement peut être décrit comme uniforme tant par sa composition que son comportement et se définit de la manière suivante pour les cinq hypothèses à l'étude.

L'environnement de test sera soumis durant l'expérimentation à deux types d'évènements :

Génération de faux positifs

Dans le premier cas, les bases de données de chaque victime, choisie à chaque fois de façon aléatoire, seront ensemencées d'événements dit de « trafic gris » selon un rythme prédéterminé au début de chaque expérimentation. Ces événements visent à simuler un faux positif.

Génération de faux négatifs

Le deuxième type d'événements, considérés comme les vraies intrusions, en tout points identiques aux faux positifs, seront générés sur la base de donnée de toutes les victimes selon un rythme appelé à varier durant l'expérimentation. Ces événements visent à simuler un balayage rapide d'un ensemble de machine sur un réseau.

3.6 Modélisation mathématique de l'environnement

Dans cette section, nous analyserons de façon mathématique les différents algorithmes à l'étude pour la démonstration de notre cadre théorique. Avec les résultats théoriques obtenus, nous tenterons de vérifier les hypothèses précédemment énoncées. Dans le chapitre suivant, nous tenterons de contraster les résultats obtenus théoriquement avec des valeurs obtenues de façon empirique.

3.6.1 Description de l'expérience

Nous considérons pour l'expérience un réseau avec une topologie en étoile dans lequel chaque hôte h_i est configuré identiquement et peut recevoir des agents mobiles ou posséder un IDS local dépendemment des algorithmes à l'étude. L'environnement réseau est constitué d'une quantité variable d'événements « gris » générés de façon aléatoire suivant une loi de Poisson avec un taux γ_i sur chaque hôte et vise à simuler des connexions inoffensives TCP à demi ouvertes, qui pourraient être interprétées comme précurseurs d'un *SYN Flood*. Cet environnement sera, par ailleurs, constitué de trafic dit « noir » représentant des attaques qui surviendront de façon aléatoire avec un taux λ durant la simulation. Dans la situation qui nous intéresse, une attaque est constituée d'un événement « noir » envoyé sur chaque hôte du réseau à l'intérieur d'un interval de temps très court et vise à simuler une attaque de type *Fast Scan*.

Afin de déterminer la présence d'intrusion ou non et signaler une alarme A, les différents IDS à l'étude étudiront le trafic durant une période de temps t après laquelle ils porteront un jugement quant à la nature du trafic qu'ils auront observé. Suite à cela, nous serons en mesure de déterminer les taux de faux positifs β (fausses alarmes) et de faux négatifs α (attaques non détectées) générés par les différents systèmes.

3.6.2 Variables

Afin d'exprimer de façon formelle le comportement probabiliste de chaque type d'événements à l'étude selon nos différents algorithmes, la notation suivante sera utilisée.

t	La taille de la fenêtre de temps sous observation .
t'	La taille de la fenêtre de temps sous observation représentant une fraction de t .

$H = \{h_1, h_2, \ldots, h_m\}$	L'ensemble des hôtes.
γ_i	Le taux de trafic gris sur h_i.
$\gamma = \sum_{i=1}^{m} \gamma_i$	Le taux de trafic gris sur le réseau.
λ	Le taux d'attaques sur l'ensemble des hôtes.
A_i	Une variable indicatrice $0-1$ prenant la valeur 1 s'il y a une alarme sur l'hôte h_i à l'intérieur de la fenêtre de temps t.
A	Une variable indicatrice $0-1$ prenant la valeur 1 s'il y a une alarme sur un réseau, qui peut être composé d'alarmes sur plusieurs hôtes.
$X_i \sim \text{Poi}(\gamma_i t)$	Une variable aléatoire suivant une distribution de Poisson et représentant le nombre d'événements de trafic gris sur h_i.
$X \sim \text{Poi}(\gamma t)$	Une variable aléatoire suivant une distribution de Poisson et représentant le nombre d'événements de trafic gris sur le réseau.
$Y_i \sim \text{Poi}(\lambda t)$	Une variable aléatoire suivant une distribution de Poisson et représentant le nombre d'éléments de trafic noir sur chaque h_i.
$Y \sim \text{Poi}(\lambda t)$	Une variable aléatoire suivant une distribution de Poisson et représentant le nombre d'attaques sur le réseau. Étant donné que le nombre d'attaques sur le réseau est égal au nombre d'attaques survenant sur chaque hôte, alors $Y_i = Y$ pour tous i.
s_i	Un seuil local sur h_i utilisé pour déterminer la présence ou non d'une alarme A_i sur h_i (voir équation (3.13)).
s	Un seuil global, appliqué à l'ensemble du réseau sous surveillance pour déterminer la présence ou non d'une alarme A sur le réseau (voir équation (3.14)).
β_i	Une variable indicatrice $0-1$ prenant la valeur 1 s'il y a un faux positif sur l'hôte h_i.
β	Une variable indicatrice $0-1$ prenant la valeur 1 s'il y a un faux positif sur l'ensemble du réseau comprenant 0 ou plus hôtes.

α_i	Une variable indicatrice $0-1$ prenant la valeur 1 s'il y a un faux négatif sur l'hôte h_i.
α	Une variable indicatrice $0-1$ prenant la valeur 1 s'il y a un faux négatif sur l'ensemble du réseau comprenant 0 ou plus hôtes.

3.6.3 Quelques précisions

Liens de corrélation Alors qu'il n'existe aucun lien de corrélation entre les événements de trafic gris X_i, il existe un lien de corrélation entre les événements de trafic noir Y sur tous les h_i. Lorsqu'un événement de trafic noir survient sur h_i, il survient nécessairement, à toute fin pratique au même moment, sur tous les éléments de H. Ce comportement du trafic noir vise à simuler un balayage rapide des ports d'un ordinateur sur le réseau.

Fenêtre de temps Dans cette expérience, la fenêtre de temps (t) sous observation (la durée d'une période de patrouille) est fixe pour toute la durée de l'expérience. Il n'est pas question ici d'une fenêtre de temps mobile mais bien d'une fenêtre de temps à intervalle régulier. Par ailleurs, chaque fenêtre de temps est indépendante, aussi les opérations de comptage à l'intérieur de chacune des fenêtres recommencent chaque fois à partir de 0.

Algorithme DSSL Tel que discuté à la section 3.5.1, il y aura une alarme sur le réseau dès qu'il y aura une alarme sur un des hôtes. Conséquemment, l'algorithme de détection d'intrus DSSL peut être décrit selon l'équation suivante :

$$A = 1 \Longleftrightarrow \exists i \mid A_i = 1 \Longleftrightarrow \exists i \mid X_i + Y_i \geq s_i \tag{3.13}$$

Algorithme PSSG L'algorithme de détection d'intrus PSSG peut être décrit selon l'équation suivante :

$$A = 1 \Longleftrightarrow \sum_{i=1}^{m} (X_i + Y_i) \geq s \tag{3.14}$$

L'algorithme PSAR L'algorithme PSAR, tel que mentionné précédemment est un algorithme basé sur la corrélation entre les événements de trafic. Nous pouvons l'associer en quelque sorte à un algorithme basé sur les signatures de comportement malicieux. Nous avons défini comme comportement malicieux le fait que survient pratiquement au même moment (dans les faits les événements ont lieu à l'intérieur d'un intervalle de temps très court) sur tous les hôtes un événement de trafic suspect. Cet algorithme peut être considéré comme une variante de l'algorithme PSSG où la durée t' de l'expérience est très courte et où le seuil s est égal au nombre d'hôtes m. Aussi pouvons nous le décrire de la même façon que l'algorithme PSSG avec $s = m$ et la fenêtre de temps t' est très petite. Ainsi, par substitution de variables, nous obtenons :

$$A = 1 \Longleftrightarrow \sum_{i=1}^{m} (X_i + Y_i) \geq m \tag{3.15}$$

Pour cet algorithme, $t^{'}$ est une fenêtre de temps à intervalle régulier et durant la durée de la patrouille, $\lfloor t/t^{'} \rfloor$ mesures seront effectuées. Chaque mesure vise à signaler la présence ou non d'une alarme. Entre chaque mesure le compte des événements suspects est remis à zéro.

3.6.4 Probabilité d'alarme

Dans cette section, nous énonçons la probabilité que survienne une alarme selon les différents algorithmes. Dans les équations qui suivent, afin d'adopter une notation claire, nous définirons la fonction de distribution de probabilité de la loi de Poisson comme :

$$p(x; \lambda t) \triangleq \frac{e^{-\lambda t}(\lambda t)^x}{x!}, \qquad x = 0, 1, 2, ...,$$

Où $e = 2,71828...$

La somme des probabilités de Poisson se définit quant à elle comme :

$$P(r; \lambda t) \triangleq \sum_{x=0}^{r} p(x; \lambda t)$$

DSSL

La probabilité d'alarme selon l'algorithme DSSL peut être exprimée comme suit :

$$\Pr(A=1) = 1 - \Pr(A=0) \tag{3.16}$$

$$= 1 - \sum_{y=0}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} \Pr(A_i = 0 | Y = y) \right) \Pr(Y = y) \right) \tag{3.17}$$

Compte tenue de l'équation (3.13) nous obtenons donc,

$$= 1 - \sum_{y=0}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} \Pr(X_i \leq s_i - y - 1) \right) \Pr(Y = y) \right) \tag{3.18}$$

$$= 1 - \sum_{y=0}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} P(s_i - y - 1; \gamma_i t) \right) p\left(y; \lambda t\right) \right) \tag{3.19}$$

$$= 1 - \sum_{y=0}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} \left(e^{-\gamma_i} \sum_{x_i=0}^{s_i-y-1} \frac{(\gamma_i t)^{x_i}}{x_i!} \right) \right) e^{-\lambda t} \frac{(\lambda t)^y}{y!} \right) \tag{3.20}$$

PSSG

La probabilité d'alarme selon l'algorithme PSSG s'exprime de la façon suivante :

$$\Pr(A=1) = 1 - \Pr(A=0) \tag{3.21}$$

$$= 1 - \sum_{y=0}^{\lfloor \frac{s-1}{m} \rfloor} \left(\Pr(A = 0 | mY = my) \Pr(mY = my) \right) \tag{3.22}$$

Compte tenu de l'équation (3.14) nous obtenons donc,

$$= 1 - \sum_{y=0}^{\lfloor \frac{s-1}{m} \rfloor} (\Pr(X \leq s - my - 1) \Pr(Y = y)) \tag{3.23}$$

$$= 1 - \sum_{y=0}^{\lfloor \frac{s-1}{m} \rfloor} (P(s - my - 1; \gamma t) p(y; \lambda t)) \tag{3.24}$$

$$= 1 - \sum_{y=0}^{\lfloor \frac{s-1}{m} \rfloor} \left(e^{-\gamma t} \left(\sum_{x=0}^{s-my-1} \frac{(\gamma t)^x}{x!} \right) e^{-\lambda t} \frac{(\lambda t)^y}{y!} \right) \tag{3.25}$$

Il est facile de constater que naturellement, lorsque $m = 1$, l'algorithme PSSG est le même que l'algorithme DSSL.

PSAR

L'équation représentant la probabilité d'alarme en fonction de l'algorithme PSAR est :

$$\Pr(A = 1) = 1 - \Pr(A = 0) \tag{3.26}$$

Si $Y > 0$ ou si $X \geq m$ une alarme est déclenchée alors

$$= 1 - \Pr(Y = 0 \wedge X < m) \tag{3.27}$$

$$= 1 - (\Pr(X < m) \Pr(Y = 0)) \tag{3.28}$$

$$= 1 - (\Pr(X \leq m - 1) \Pr(Y = 0)) \tag{3.29}$$

$$= 1 - (p(m - 1; \gamma t') p(0; \lambda t')) \tag{3.30}$$

$$= 1 - \left(e^{-\gamma t'} \left(\sum_{x=0}^{m-1} \frac{(\gamma t')^x}{x!} \right) e^{-\lambda t'} \right) \tag{3.31}$$

Analyse

Si nous réalisons le graphe de la probabilité d'alarme en fonction du nombre d'hôtes comme nous le montre la Figure 3.4, nous pouvons constater qu'il est beaucoup plus probable d'obtenir une alarme avec l'algorithme DSSL, qu'avec PSSG plus le réseau

sous surveillance compte d'hôtes. Par ailleurs, nous pouvons constater graphiquement que lorsque $m = 1$, tel que mentionné précédemment, l'algorithme PSSG est le même que l'algorithme DSSL.

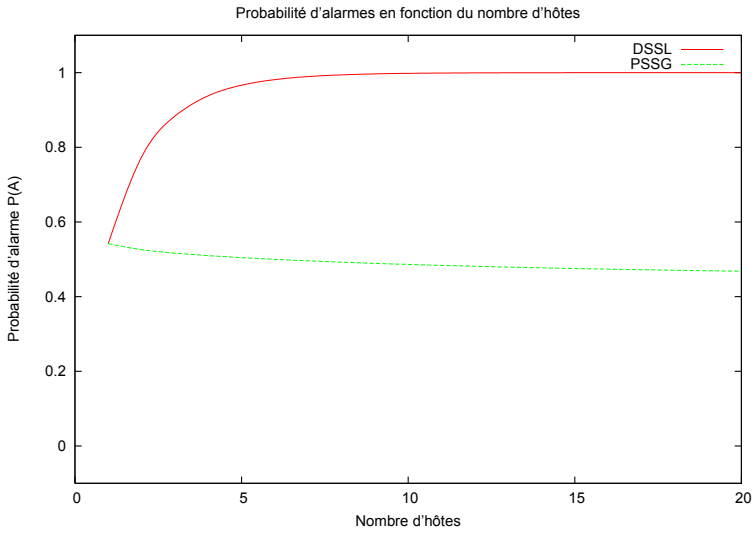


FIGURE 3.4 Probabilité d'alarme en fonction du nombre d'hôtes étant donné $t = 1$, $s_i = 10$, $\gamma_i = 10$, $\lambda = 1$.

3.6.5 Probabilité de Faux Positifs

Dans cette section nous énonçons la probabilité d'obtenir un faux positif selon les différents algorithmes.

Faux Positifs. Un faux positif se définit comme la probabilité d'obtenir une alarme **et** de n'avoir aucun événement **noir** à l'intérieur de la période sous observation.

$$\Pr(\beta = 1) = \Pr(A = 1 \wedge Y = 0) \tag{3.32}$$

DSSL

À l'équation 3.32, étant donné que $P(A)$ n'est pas indépendant de $P(Y)$, nous utilisons l'équation 3.17 afin d'obtenir la probabilité d'obtenir un faux positif.

$$\Pr(\beta = 1) = \Pr(A = 1 \wedge Y = 0) \tag{3.33}$$
$$= \Pr(A = 1 \mid Y = 0) \Pr(Y = 0) \tag{3.34}$$
$$= \left(1 - \left(\prod_{i=1}^{m} \Pr(A_i = 0 | Y = 0)\right)\right) \Pr(Y = 0) \tag{3.35}$$
$$= \left(1 - \left(\prod_{i=1}^{m} \Pr(X_i \leq s_i - 1)\right)\right) \Pr(Y = 0) \tag{3.36}$$
$$= \left(1 - \left(\prod_{i=1}^{m} P(s_i - 1; \gamma_i t)\right)\right) p\left(0; \lambda t\right) \tag{3.37}$$
$$= \left(1 - \left(\prod_{i=1}^{m} \left(e^{-\gamma_i} \sum_{x_i=0}^{s_i-1} \frac{(\gamma_i t)^{x_i}}{x_i!}\right)\right)\right) e^{-\lambda t} \tag{3.38}$$

PSSG

À l'équation 3.32, étant donné que $P(A)$ n'est pas indépendant de $P(Y)$, nous utilisons l'équation 3.22 afin d'obtenir la probabilité d'obtenir un faux positif.

$$\Pr(\beta = 1) = \Pr(A = 1 \wedge Y = 0) \tag{3.39}$$
$$= (1 - \Pr(A = 0 | mY = my = 0)) \Pr(mY = my = 0) \tag{3.40}$$
$$= (1 - \Pr(X \leq s - 1)) \Pr(Y = 0) \tag{3.41}$$
$$= (1 - P(s - 1; \gamma t)) \, p\left(0; \lambda t\right) \tag{3.42}$$
$$= \left(1 - e^{-\gamma t} \left(\sum_{x=0}^{s-1} \frac{(\gamma t)^x}{x!}\right)\right) e^{-\lambda t} \tag{3.43}$$

PSAR

À l'équation 3.32, étant donné que $P(A)$ n'est pas indépendant de $P(Y)$, nous utilisons l'équation 3.27 afin d'obtenir la probabilité d'obtenir un faux positif.

$$\Pr(\beta = 1) = \Pr(A = 1 \wedge Y = 0) \tag{3.44}$$
$$= (1 - \Pr(A = 0 | mY = my = 0)) \Pr(mY = my = 0) \tag{3.45}$$
$$= (1 - \Pr(X \leq m - 1)) \Pr(Y = 0) \tag{3.46}$$
$$= (1 - P(m - 1; \gamma t')) \, p(0; \lambda t') \tag{3.47}$$
$$= \left(1 - e^{-\gamma t'} \left(\sum_{x=0}^{m-1} \frac{(\gamma t')^x}{x!}\right)\right) e^{-\lambda t'} \tag{3.48}$$

3.6.6 Probabilité de Faux Négatifs

Dans cette section, nous énonçons la probabilité d'obtenir un faux négatif selon les différents algorithmes.

Faux Négatifs. Un faux négatif se définit comme la probabilité de ne pas obtenir d'alarme **et** d'avoir au moins un événement de trafic **noir** à l'intérieur de la période sous observation.

$$\Pr(\alpha = 1) = \Pr(A = 0 \wedge Y > 0) \tag{3.49}$$

DSSL

À l'équation 3.49, étant donné que $P(A)$ n'est pas indépendant de $P(Y)$, nous utilisons l'équation 3.17 afin d'obtenir la probabilité d'obtenir un faux négatif.

$$\Pr(\alpha = 1) = \sum_{y=1}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} \Pr(A_i = 0 | Y = y)\right) \Pr(Y = y)\right) \tag{3.50}$$
$$= \sum_{y=1}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} \Pr(X_i \leq s_i - y - 1)\right) \Pr(Y = y)\right) \tag{3.51}$$
$$= \sum_{y=1}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} P(s_i - y - 1; \gamma_i t)\right) p(y; \lambda t)\right) \tag{3.52}$$
$$= \sum_{y=1}^{\min(s_i)-1} \left(\left(\prod_{i=1}^{m} \left(e^{-\gamma_i t} \sum_{x_i=0}^{s_i-y-1} \frac{(\gamma_i t)^{x_i}}{x_i!}\right)\right) e^{-\lambda t} \frac{(\lambda t)^y}{y!}\right) \tag{3.53}$$

PSSG

À l'équation 3.49, étant donné que $P(A)$ n'est pas indépendant de $P(Y)$, nous utilisons l'équation 3.22 afin d'obtenir la probabilité d'obtenir un faux négatif.

$$\Pr(\alpha = 1) = \sum_{y=1}^{\lfloor \frac{s-1}{m} \rfloor} (\Pr(A = 0 | mY = my) \Pr(mY = my)) \tag{3.54}$$

$$= \sum_{y=1}^{\lfloor \frac{s-1}{m} \rfloor} (\Pr(X \leq s - my - 1) \Pr(mY = my)) \tag{3.55}$$

$$= \sum_{y=1}^{\lfloor \frac{s-1}{m} \rfloor} (P(s - my - 1; \gamma t) p(y; \lambda t)) \tag{3.56}$$

$$= \sum_{y=1}^{\lfloor \frac{s-1}{m} \rfloor} \left(e^{-\gamma t} \left(\sum_{x=0}^{s-my-1} \frac{(\gamma t)^x}{x!} \right) e^{-\lambda t} \frac{(\lambda t)^y}{y!} \right) \tag{3.57}$$

PSAR

$$\Pr(\alpha = 1) = \sum_{y=1}^{\lfloor \frac{m-1}{m} \rfloor} (\Pr(A = 0 | mY = my) \Pr(mY = my)) \tag{3.58}$$

$$= 0 \tag{3.59}$$

3.7 Analyse de l'impact des paramètres environnementaux et de l'IDS sur les taux de faux positifs et de faux négatifs

Suite à l'énoncé des probabilités de faux positifs et de faux négatifs, nous analyserons ici les algorithmes DSSL, PSSG et PSAR et l'impact que peut avoir une variation des paramètres environnementaux et de l'IDS. Ceci correspond à la vérification des hypothèses 1 à 4 énoncées précédemment.

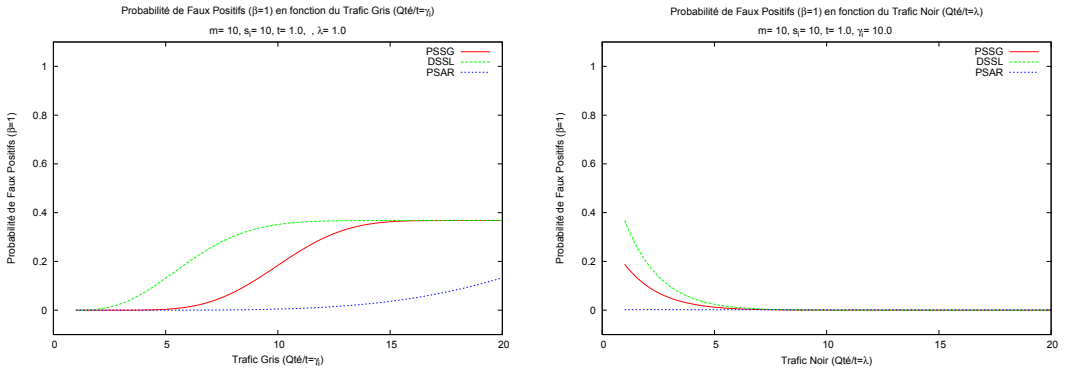
3.7.1 Probabilité de faux positifs en fonction du taux de trafic gris et de trafic noir

Dans cette expérience, nous avons fait varier les taux de trafic noir et de trafic gris auxquels est soumis notre système de détection d'intrus et avons observé les effets sur les taux de faux positifs. Lors de cette expérience nous avons fixé le nombre d'hôtes à 10 et un niveau seuil de détection à 10 événements par période sous observation.

Nous avons constaté à la Figure 3.6 que pour les trois algorithmes à l'étude, une augmentation des taux de trafic gris entraîne une augmentation des taux de faux positifs et une augmentation des taux de trafic noir entraîne une diminution des taux de faux positifs. Le graphique 3.5(a) montre plus facilement que ces relations tiennent pour l'algorithme PSAR et sont plus facilement observables pour des taux de trafic gris très élevés.

Nous observons par ailleurs que la probabilité d'obtenir un faux positif est limitée par la probabilité d'avoir un événement de trafic noir (ceci devient donc un cas de faux négatifs ou de vrais positifs) ainsi que par la probabilité d'obtenir un nombre d'événements de trafic gris inférieur au seuil, donc d'obtenir un vrai négatif.

Nous sommes en mesure de constater que l'algorithme DSSL performe généralement mieux que l'algorithme PSSG en terme de faux positifs et l'algorithme PSAR performe mieux que les deux autres.



(a) En fonction du trafic gris où $m = 10$, $s_i = 10$ et $\lambda = 1$

(b) En fonction du trafic noir où $m = 10$, $s_i = 10$ et $\gamma_i = 10$

FIGURE 3.5 Taux de faux positifs(β) en fonction du trafic noir (λ) et du trafic gris(γ_i)

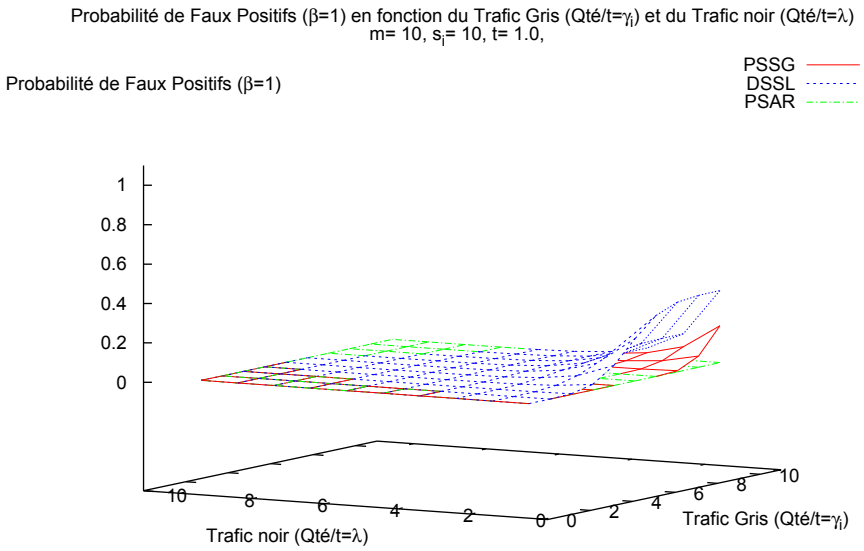


FIGURE 3.6 Taux de faux positifs(β) en fonction de la quantité de trafic noir (λ) et la quantité de trafic gris (γ_i) par unité de temps (t)

3.7.2 Probabilité de faux négatifs en fonction du taux de trafic gris et de trafic noir

Cette expérience, identique à la précédente, vise cette fois-ci à mesurer la probabilité de faux négatifs en fonction de différents taux de trafic gris et de trafic noir.

Lorsque nous observons la Figure 3.8, force est de constater que, compte tenu du modèle mathématique utilisé pour simuler l'algorithme PSAR, nous obtenons un taux de faux négatifs de zéro, et ce, indépendemment des valeurs de taux de trafic gris et de trafic noir employées.

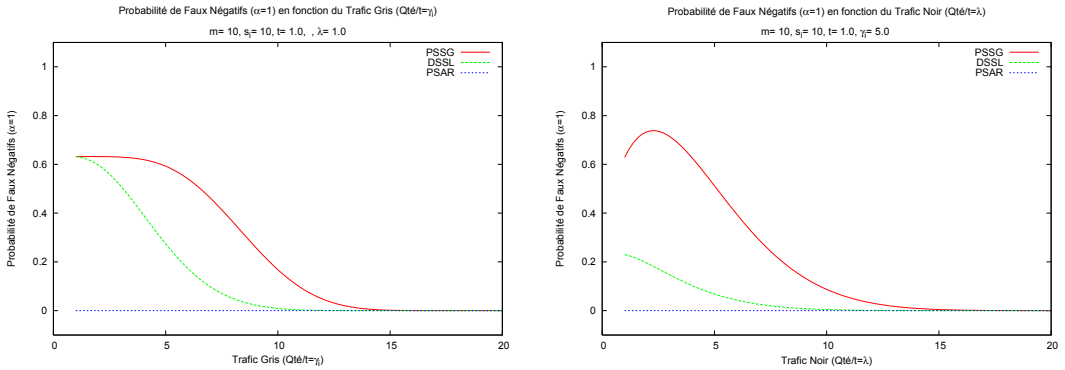
Nous pouvons cependant constater, comme à la Figure 3.7(a), qu'une augmentation des taux de trafic gris pour les algorithme PSSG et DSSL entraîne une diminution des taux de faux négatifs. En effet, une augmentation des taux de trafic gris augmente la probabilité de déclenchement d'alarme et diminue donc la probabilité de faux négatifs. Aussi, nous pouvons observer, comme par exemple à la Figure 3.7(b), qu'une augmentation des taux de trafic noir entraîne une augmentation des taux de faux négatifs pour une faible quantité de trafic alors que ces derniers baissent lorsque la

quantité de trafic devient plus importante. En effet, une augmentation de trafic noir entraîne une augmentation rapide de la probabilité d'alarme et donc une diminution des taux de faux négatifs.

La probabilité de faux négatifs est donc limitée par la probabilité d'alarme qui augmente à mesure que les taux de trafic (trafic gris + trafic noir) augmentent.

Il est important ici de souligner que les valeurs choisies pour le seuil de déclenchement d'une alarme sont importantes pour évaluer les taux de faux négatifs.

Ces observations nous permettent de conclure que la probabilité de faux négatifs de l'algorithme PSAR est inférieure à celle de l'algorithme PSSG qui est inférieur à celle de DSSL.



(a) En fonction du trafic gris où $m = 10$, $s_i = 10$ et $\lambda = 1$

(b) En fonction du trafic noir où $m = 10$, $s_i = 10$ et $\gamma_i = 5$

FIGURE 3.7 Taux de faux négatifs(α) en fonction du trafic noir (λ) et du trafic gris(γ_i)

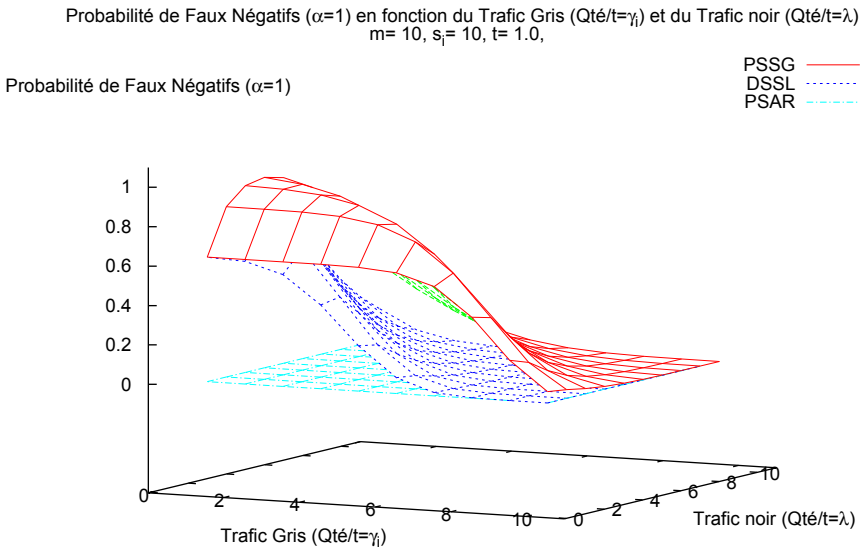
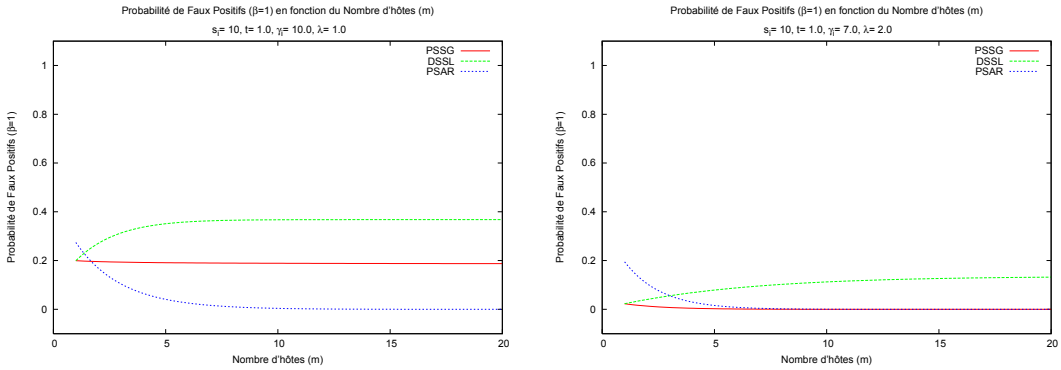


FIGURE 3.8 Taux de faux négatifs(α) en fonction de la quantité de trafic noir (λ) et la quantité de trafic gris (γ_i) par unité de temps (t)

3.7.3 Variation du nombre d'hôtes

Dans cette expérience, nous observons l'impact que peut avoir le nombre d'hôtes et le ratio signal sur bruit (c'est-à-dire le ratio du traux de trafic noir sur le taux de trafic gris) sur les taux de faux positifs et de faux négatifs. Nous avons fixé le seuil de détection à 10. Par ailleurs, nous observons aussi l'impact pour différents taux de saturation (q), c'est-à-dire pour différents volume de trafic (trafic gris + trafic noir)/niveau seuil.

Nous observons d'abord à la Figure 3.11 qu'une augmentation du nombre d'hôtes entraîne une augmentation des taux de faux positifs pour l'algorithme DSSL qui se stabilise à la borne supérieure à une valeur égale au taux permettant d'obtenir un événement de trafic noir. Nous constatons aussi que pour PSSG, une augmentation du nombre d'hôtes entraîne une diminution des taux de faux positifs. Puisque notre seuil est global pour l'ensemble des hôtes, il y a moins de chance qu'une alarme survienne. Nous constatons qu'il en est de même pour l'algorithme PSAR. Cet algorithme permet une meilleure distinction des événements quand il y a plus d'hôtes pour corréler les



(a) $m = 10$ $s_i = 10$ $\gamma_i = 10$ $\lambda = 1$ (b) $m = 10$ $s_i = 10$ $\gamma_i = 7$ $\lambda = 2$

FIGURE 3.9 Taux de faux positifs(β) en fonction du nombre d'hôtes (m)

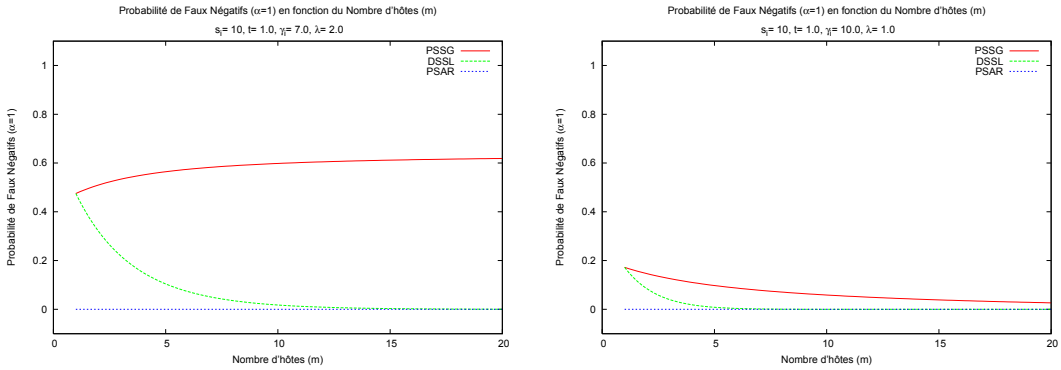
événements entre eux. Plus il y a d'hôtes, plus il est facile de reconnaître avec certitude le patron de détection. Il est d'ailleurs inférieur aux deux autres algorithmes pour un faible nombre d'hôtes ($m = 1$).

Si nous observons l'impact du ratio signal sur bruit, nous constatons qu'une augmentation du ratio entraîne dans tous les cas une diminution des probabilités de faux positifs, et ce, pour tous les algorithmes. Une augmentation du trafic noir a pour effet de diminuer le nombre de cas où il n'y a que des événements de trafic gris en nombre plus grand ou égal au seuil de détection.

À la Figure 3.10, nous pouvons constater qu'une augmentation du nombre d'hôtes entraîne une diminution des taux de faux négatifs pour l'algorithme DSSL étant donné que la probabilité d'alarme augmente à mesure que l'on augmente le nombre d'hôtes. Avec l'algorithme PSSG, nous observons une augmentation des taux de faux négatifs étant donné la diminution de la probabilité d'alarme. Toutefois lorsque le taux de saturation est supérieur au seuil et le ratio signal sur bruit est faible la probabilité de faux négatifs diminue à mesure que nous augmentons le nombre d'hôtes. Dans ce cas, la probabilité d'alarme augmente plus rapidement que la probabilité d'avoir du trafic noir.

Une variation du ratio signal sur bruit a pour effet d'augmenter les probabilités de faux négatifs pour DSSL et PSSG étant donné qu'un plus grand nombre d'événements de trafic noir va entraîner une plus grande probabilité de faux négatifs.

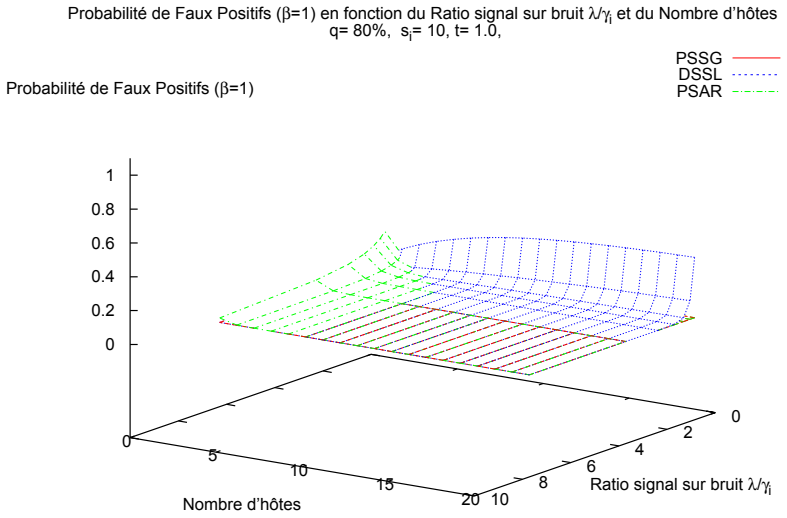
Nous pouvons donc constater que la plupart du temps, sauf pour un petit nombre



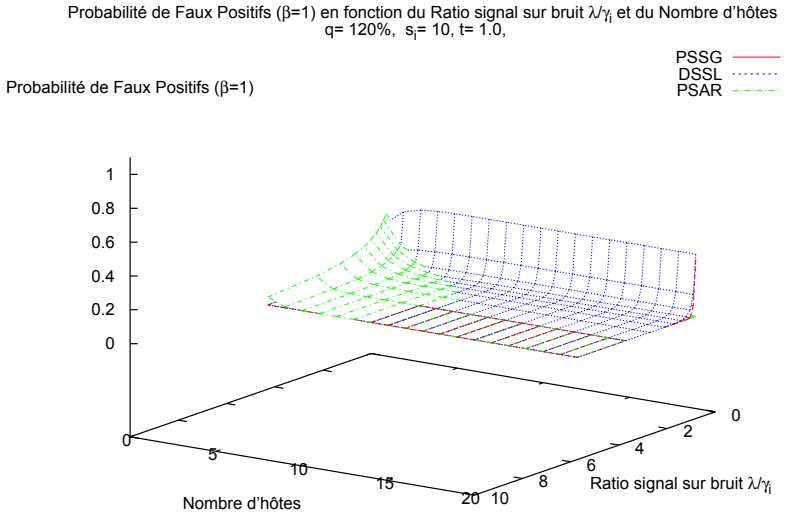
(a) $m = 10$ $s_i = 10$ $\gamma_i = 7$ $\lambda = 2$ (b) $m = 10$ $s_i = 10$ $\gamma_i = 10$ $\lambda = 1$

FIGURE 3.10 Taux de faux négatifs(α) en fonction du nombre d'hôtes (m)

d'hôtes ($m < 6$), l'algorithme PSAR performe mieux en terme de faux positifs que l'algorithme PSSG qui, lui, performe mieux que l'algorithme DSSL. En ce qui a trait aux faux négatifs, l'algorithme PSAR performe toujours mieux alors que cette fois-ci nous obtenons de meilleurs résultats avec l'algorithme DSSL qu'avec l'algorithme PSSG.

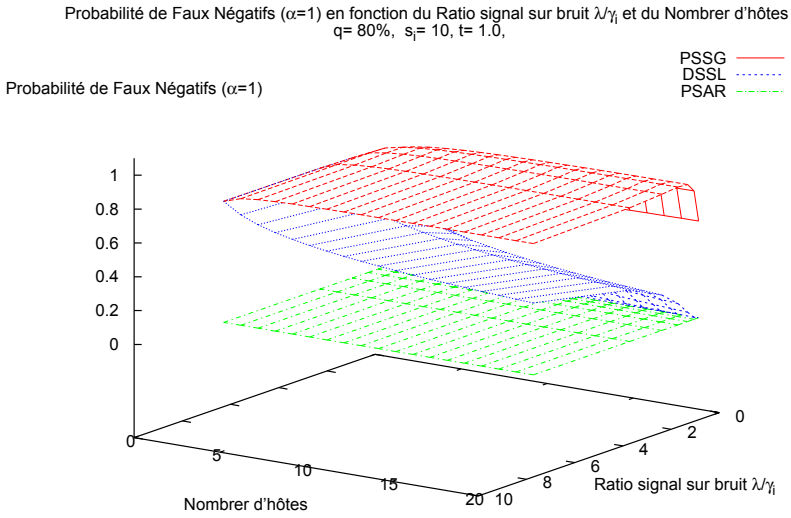


(a) $m = 10$ $s_i = 10$ $\gamma_i = 5$

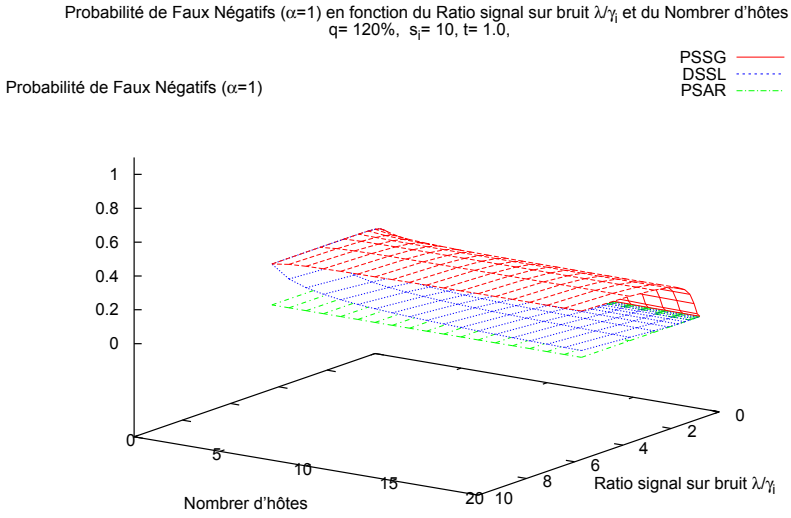


(b) $m = 10$ $s_i = 10$ $\gamma_i = 7$

FIGURE 3.11 Taux de faux positifs(β) en fonction du ratio signal sur bruit (λ/γ_i) et du nombre d'hôtes (m)



(a) $m = 10$ $s_i = 10$ $\gamma_i = 5$



(b) $m = 10$ $s_i = 10$ $\gamma_i = 7$

FIGURE 3.12 Taux de faux négatifs(α) en fonction du ratio signal sur bruit (λ/γ_i) et du nombre d'hôtes (m)

3.7.4 Variation du seuil de détection

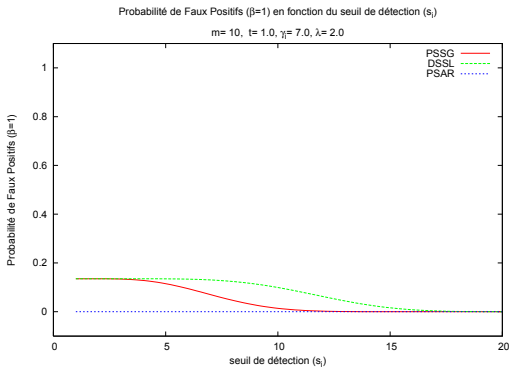
Dans cette expérience où nous avons fixé le nombre d'hôtes à 10, nous avons fait varier le niveau seuil pour en observer les effets sur les taux de faux positifs et de faux négatifs.

Nous avons observé comme le montre la Figure 3.13 qu'une augmentation du niveau seuil a pour effet de faire chuter les taux de faux positifs des algorithmes DSSL et PSSG. En effet, plus les seuils sont élevés, plus la possibilité que survienne un événement de trafic noir sans qu'il y ait eu d'alarmes augmente. D'ailleurs, nous observons qu'à volume de trafic égal, une augmentation des taux de trafic noir a pour effet de diminuer les taux de faux positifs. Par ailleurs, une augmentation des taux de trafic gris a pour effet de produire, pour un même seuil de détection, des taux de faux positifs plus élevés. Aussi, il est possible de remarquer que des taux de saturation élevés produisent davantage de faux positifs compte tenu d'une augmentation du nombre d'alarmes.

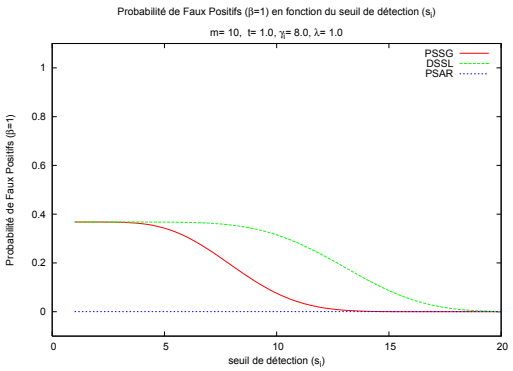
Nous pouvons encore une fois observer que l'algorithme DSSL performe mieux en terme de faux positifs que l'algorithme PSSG.

En ce qui a trait aux faux négatifs, nous observons que ceux-ci augmentent avec les seuils de détection. Par ailleurs, nous observons qu'une augmentation des taux de trafic noir augmente les taux de faux négatifs. De la même façon, nous observons qu'un ratio signal sur bruit plus bas entraîne une diminution des taux de faux négatifs. Nous pouvons aussi constater qu'une augmentation des taux de saturation a pour effet de diminuer les taux de faux négatifs, compte tenu de l'augmentation de la probabilité d'alarme.

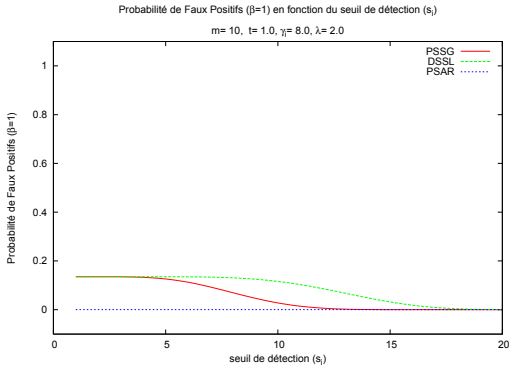
Les relations observées antérieurement sont maintenues, c'est-à-dire que les algorithmes PSAR et PSSG performent mieux que DSSL en terme de faux positifs alors que l'algorithme DSSL, performe mieux que PSSG en terme de faux négatifs.



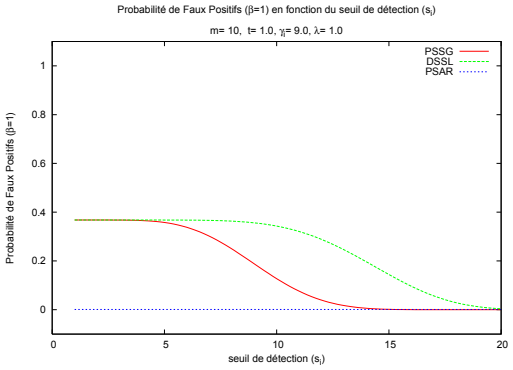
(a) $m = 10\ s_i = 10\ \gamma_i = 7\ \lambda = 2$



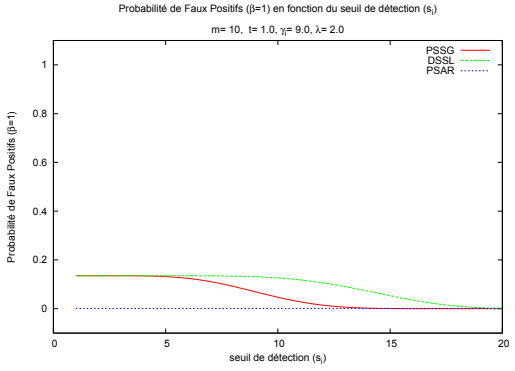
(b) $m = 10\ s_i = 10\ \gamma_i = 8\ \lambda = 1$



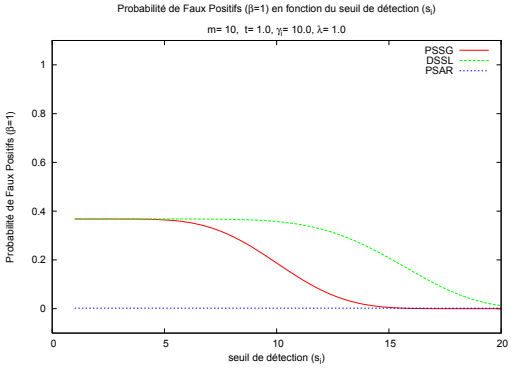
(c) $m = 10\ s_i = 10\ \gamma_i = 8\ \lambda = 2$



(d) $m = 10\ s_i = 10\ \gamma_i = 9\ \lambda = 1$

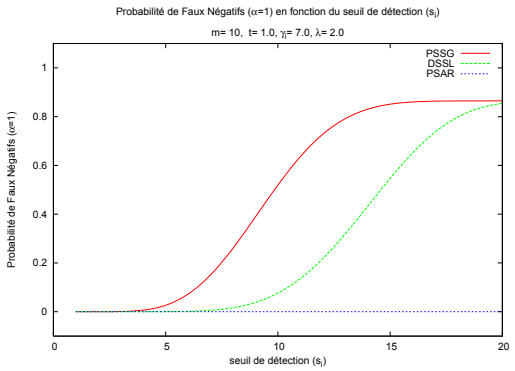


(e) $m = 10\ s_i = 10\ \gamma_i = 9\ \lambda = 2$

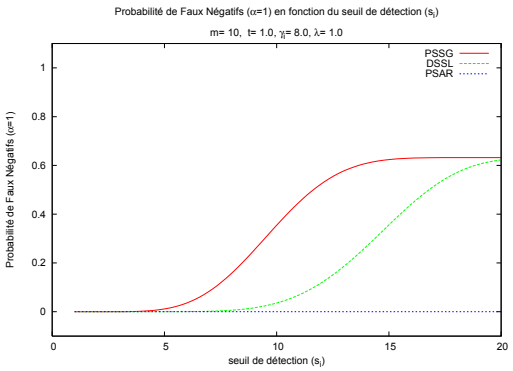


(f) $m = 10\ s_i = 10\ \gamma_i = 10\ \lambda = 1$

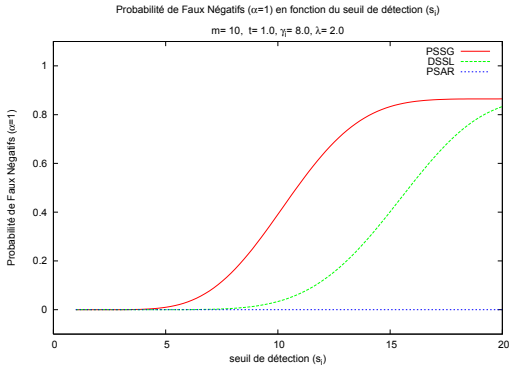
FIGURE 3.13 Taux de faux positifs(β) en fonction du seuil (s_i)



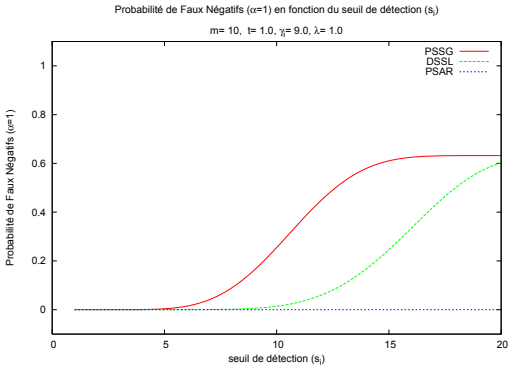
(a) $m = 10\ s_i = 10\ \gamma_i = 7\ \lambda = 2$



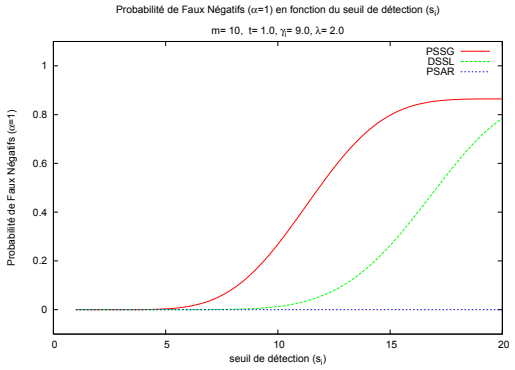
(b) $m = 10\ s_i = 10\ \gamma_i = 8\ \lambda = 1$



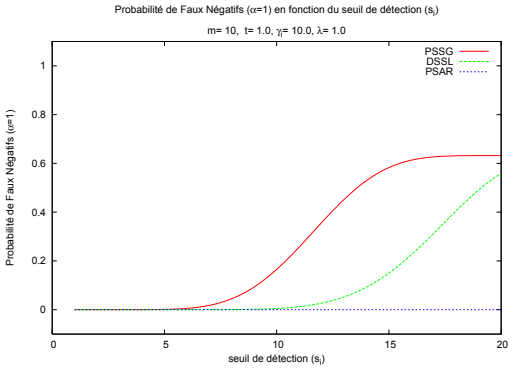
(c) $m = 10\ s_i = 10\ \gamma_i = 8\ \lambda = 2$



(d) $m = 10\ s_i = 10\ \gamma_i = 9\ \lambda = 1$



(e) $m = 10\ s_i = 10\ \gamma_i = 9\ \lambda = 2$



(f) $m = 10\ s_i = 10\ \gamma_i = 10\ \lambda = 1$

FIGURE 3.14 Taux de faux négatifs(α) en fonction du seuil (s_i)

3.8 Analyse des taux de faux positifs en fonction des taux de faux négatifs, en fonction des paramètres expérimentaux

Il est intéressant d'analyser la probabilité de faux positifs en fonction de la probabilité de faux négatifs en fonction du seuil. Ceci correspond à la vérification de l'hypothèse 5 énoncée précédemment. En effet, comme nous pouvons l'observer à la Figure 3.15, nous pouvons remarquer pour les algorithmes PSSG et DSSL qu'une augmentation des taux de faux positifs entraîne une diminution des taux de faux négatifs. La relation est donc inversement proportionnelle lorsque nous faisons varier le seuil de détection. Il n'y a pas de données pour l'algorithme PSAR, étant donné que ce dernier a toujours des taux de faux négatifs de 0.

Ces données nous montrent qu'il n'y a pas de niveaux optimaux entre les faux positifs et les faux négatifs comme nous pouvions nous y attendre. Selon ce graphique, nous pouvons constater que l'algorithme PSSG performe légèrement mieux que l'algorithme DSSL avec 10 hôtes. En effet, notre modèle mathématique montre que l'algorithme PSSG est supérieur en terme de faux positifs mais inférieur en terme de faux négatifs à l'algorithme DSSL.

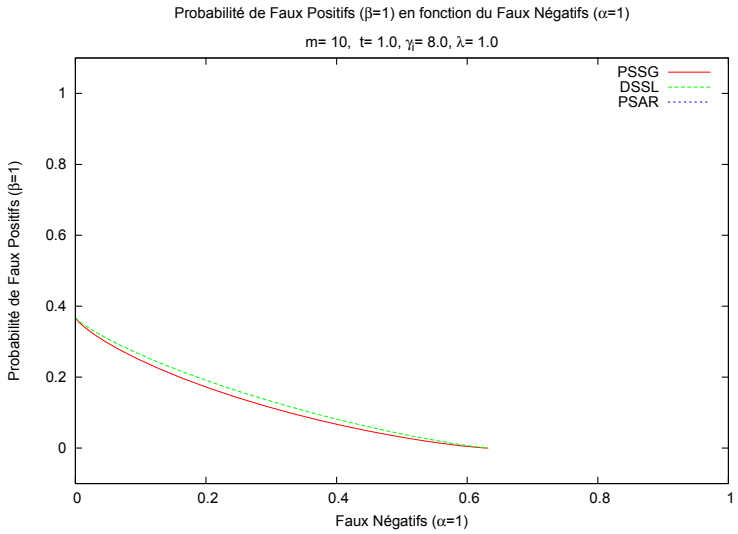


FIGURE 3.15 Taux de faux positifs en fonction des taux de faux négatifs en fonction du seuil.

3.9 Vérification théorique des hypothèses

Suite à l'analyse théorique effectuée précédemment, nous constatons que les hypothèses 1 et 2 énoncées au chapitre 3 sont vérifiées, à savoir qu'il existe une relation entre les taux de faux positifs et de faux négatifs, et les taux de trafic gris et de trafic noir. Nous constatons que l'augmentation du taux de trafic noir a pour effet de diminuer la probabilité d'obtenir un faux positif. De la même façon, les figures montrent bien que la probabilité, quant à elle, d'obtenir un faux négatif augmente. Inversement, nous observons qu'une augmentation du taux de trafic gris entraîne une diminution du taux de faux négatifs et une augmentation du taux de faux positifs.

Par ailleurs, nous constatons que les hypothèses 3 et 4 sont vérifiées. En effet, les taux de faux positifs de PSSG pour les valeurs observées sont inférieures aux taux obtenus par l'algorithme DSSL. Aussi, les taux de faux négatifs de PSSG sont toujours supérieurs à ceux obtenus avec DSSL. L'algorithme proposé dans ce mémoire, quant à lui, avec ses taux de faux positifs et de faux négatifs très bas, est meilleure que les deux autres dans la plupart des cas. Toutefois, par souci de précision, cet algorithme pourrait présenter des taux de faux positifs supérieurs aux deux autres algorithmes advenant une grande fenêtre t' ou un très haut taux de trafic gris ou un très petit nombre d'hôtes.

Il nous fut aussi possible de montrer qu'il existe une relation entre les taux de faux positifs et de faux négatifs pour les algorithmes DSSL et PSSG. En effet, en faisant varier le seuil de détection, nous pouvons montrer que les valeurs de faux positifs augmentent alors que les taux de faux négatifs diminue et vice versa pour nos algorithmes, ce qui valide l'hypothèse 5.

CHAPITRE 4

ÉVALUATION DE PERFORMANCE ET RÉSULTATS EXPÉRIMENTAUX

Dans cette section se retrouve l'ensemble des détails d'implantation entrant dans la réalisation des tests qui ont servi à valider le cadre théorique présenté précédemment. Nous traiterons d'abord de l'environnement matériel et logiciel, suivra par la suite une description de l'environnement de test comprenant les variables et métriques que nous avons fait varier pour chacune des sessions de tests. Pour terminer nous ferons l'analyse des résultats obtenus.

4.1 Environnement matériel et logiciel d'implantation

Cette section s'attarde sur les composantes matérielles et logicielles entrant dans la composition de la plate-forme de test.

L'environnement matériel entrant dans la composition de la plate-forme de test se décline de la façon suivante :

1 ordinateur de bureau

- processeur AMD64 3500+
- 1 Go. RAM DDR400
- 2 Disques durs de 120 Go. en configuration « Mirroring » pour une capacité totale de 120 Go.
- Connecteur réseau intégré Ethernet Gigabit LAN.

L'environnement logiciel, quant à lui, est beaucoup plus complexe et représente un compromis entre maintenir un degré d'isolement suffisant entre les différents hôtes

simulés à l'intérieur d'une seule machine et permettre un niveau de performance acceptable pour les besoins des tests.

L'environnement logiciel se décline de la façon suivante :

- Windows XP SP2
- MySQL 5.0 Community Edition
- Java 5 de Sun Microsystem
- plate-forme d'agent mobile JADE 3.3

JAVA 5. Les logiciels réalisés pour les tests l'ont été avec le langage JAVA 5 de Sun Microsystem.

Base de données. Une base de données MySQL 5.0 fut instanciée sur le PC afin de servir à recueillir le trafic généré pour l'expérience. Afin de rendre cette expérience plus fidèle à la réalité, nous avons utilisé comme schéma de donnée celui de l'IDS SNORT version 1.06. Afin de distinguer et d'isoler les données des différents hôtes sous observations, nous avons attribué à chaque hôte un schéma identique sur la base de données. Afin d'améliorer les performances, nous avons configuré la base de données pour posséder un nombre de connexions suffisantes pour soutenir le nombre important de requêtes concurrentes qu'elle pourrait subir. De plus, du côté de l'application, cette initiative est soutenue par l'usage de pool de connexions. Par ailleurs, nous avons rédigé les requêtes effectuées sur la base de données sous forme de procédures stockées afin d'améliorer la sécurité de l'ensemble et de permettre de meilleures performances. Les interactions avec la base de données se font par l'entremise du connecteur JAVA fourni par MySQL, soit Connector/J.

Générateur de trafic malicieux. Afin de simuler l'arrivée sur chaque hôte de trafic gris et de trafic noir, nous avons conçu et réalisé un générateur de trafic. Ce dernier fait usage de l'ordonnanceur en code libre Quartz ainsi que de la librairie Jakarta-Commons-Math pour assurer le lancement d'événements suivant une distribution exponentielle. La génération d'événements consiste à faire une requête sur le schéma de la base de données correspondant à l'hôte visé et à y inscrire l'événement désiré.

Polytechnique Mobile Agent Intrusion Detection (PMAID). Cette plate-forme logicielle de notre conception vise à détecter le trafic injecté et à faire la distinction entre le trafic gris et le trafic noir. C'est sur cette plate-forme, qui repose sur la plate-forme d'agent mobile JADE 3.4, que nous avons instancié les trois algorithmes de détection sous observation afin de pouvoir les comparer entre eux. Un conteneur JADE fut instancié par hôte simulé de façon à conserver un degré de séparation entre les différents hôtes et ainsi pouvoir simuler adéquatement les mécanismes de mobilité des agents mobiles.

GAUNTLET. GAUNTLET consiste en fait en une plate-forme unifiée pour fin de test comprenant le générateur de trafic malicieux ainsi que PMAID. GAUNTLET est organisé selon une architecture à « thread » d'exécution multiple. Ceci permet ainsi d'injecter du trafic et de permettre à la plate-forme d'agents mobiles d'opérer quasiment en parallèle. Par ailleurs, GAUNTLET réunit les différentes fonctionnalités permettant de produire des rapports et des graphiques.

4.2 Description de l'environnement de test

Compte tenu des moyens réduits dont nous disposons, les expériences seront toutes réalisées sur la même machine mentionnée précédemment. Les conteneurs JADE ainsi que les différents schémas sur la base de données visent à simuler des hôtes distincts sur la même machine. Pour les algorithmes PSSG et PSAR, les agents mobiles se déplacent de conteneur en conteneur en utilisant les mécanismes de sockets du PC, comme ils le feraient s'ils se déplaçaient sur un réseau LAN (voir Figure 4.1), par exemple. Compte tenu de l'environnement de test réduit dont nous disposons et pour des raisons de performance, les fonctionnalités de l'algorithme DSSL ont été fusionnées avec les agents implémentant les autres stratégies de détection. Pour cet algorithme en particulier, nous permettons toutefois de faire des requêtes sur tous les schémas de la base de données à intervalles réguliers de façon à simuler un agent stationnaire résidant sur chacun des hôtes.

4.2.1 Caractéristiques de l'environnement

L'environnement de test, en utilisant la taxonomie élaborée précédemment, peut être décrit comme uniforme tant par sa composition que son comportement et se définit de la manière suivante pour les cinq hypothèses à l'étude.

Réseau.

1. Simulation par isolation de ressources sur un seul PC d'une topologie en étoile.
2. Le trafic consiste en l'envoi de requêtes SQL vers le schéma de la base de données des différents conteneur d'agent mobile.
3. L'envoi de trafic se fait selon un rythme déterminé (poissonnien) par le générateur d'évènements.
4. Le partage d'information entre les différents conteneurs se fait par le déplacement d'agents mobiles.

Matériel non réseau. Le matériel non réseau consiste en un PC sur lequel réside un certain nombre de conteneurs d'agents mobiles isolés les uns des autres et identiquement configurés. C'est-à-dire qu'ils reçoivent dans leurs schémas de bases de données respectives des évènements réseau simulés par des requêtes SQL.

Système. Au niveau système, le PC est configuré avec Windows XP SP2.

Application. Au niveau applicatif, seules les plates-formes d'agents mobiles sont en opération ainsi qu'une base de données pour colliger les évènements réseaux reçus sur des schémas indépendants.

Évènements accidentels et délibérés. Au niveau des évènements accidentels et délibérés, ces derniers sont injectés directement dans les schémas de la base de données des hôtes victimes et sont identiques, ne variant entre l'un et l'autre que par leur distribution dans l'espace et le temps. (voir les sections 3.5.5 et 3.5.5 pour les détails).

IDS (Méta) Au niveau des caractéristiques de l'environnement pour supporter un IDS, le PC possède plusieurs plates-formes (que nous appellerons, de façon interchangeable dans ce mémoire, hôte) pouvant accueillir des agents mobiles. La topologie est donc identique à celle du réseau, soit en étoile avec un routeur virtuel en son centre. Chaque hôte a la capacité de divulguer aux agents mobiles le contenu de son schéma de base de donnée.

4.3 Plan d'expériences

Dans cette section, nous discuterons des variables que nous observerons durant les différentes sessions de tests et nous énoncerons les valeurs des indices utilisés pour les différents tests.

4.3.1 Choix des variables

Pour l'ensemble des simulations, les variables indépendantes choisies sont les mêmes que celles utilisées pour la démonstration théorique des différents algorithmes énoncés à la section 3.5.5. Ce sont les suivantes :

T	la durée de l'expérience
t	la taille de chaque fenêtre de temps sous observation durant l'expérience
t'	la taille de la fenêtre de temps sous observation représentant une fraction de t
m	le nombre d'hôtes
γ_i	le taux de trafic gris sur chaque hôte
$\gamma = m\gamma_i$	le taux de trafic gris sur le réseau
λ	le taux d'attaques sur l'ensemble des hôtes
s_i	un seuil local sur un hôte, utilisé pour déterminer la présence ou non d'une alarme
$s = ms_i$	un seuil global, appliqué à l'ensemble du réseau sous surveillance pour déterminer la présence ou non d'une alarme sur le réseau

4.3.2 Choix des métriques

Inspiré de notre cadre théorique, les métriques que nous comptons mesurer sont :
- le taux de faux positifs et
- le taux de faux négatifs.

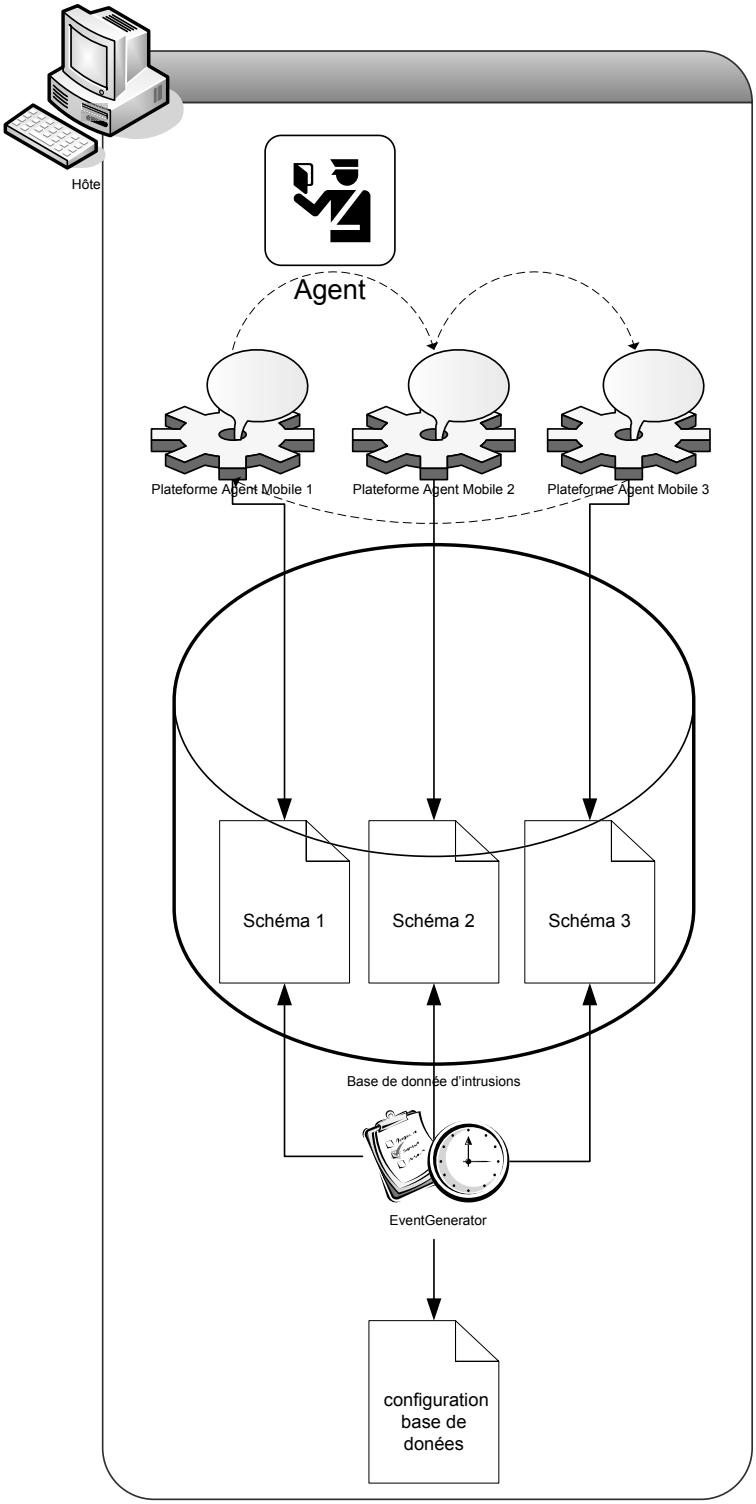


FIGURE 4.1 Environnement de test

Les valeurs utilisées lors de chacun des tests sont les suivantes :

$$\begin{aligned} m &= 10 \\ s_i &= 10 \\ t &= 30 \text{ secondes} \\ t' &= 1 \text{ seconde} \\ T &= 50 \text{ minutes} \end{aligned}$$

Nous serons appelés à faire varier les taux de trafic gris γ_i et de trafic noir λ après chaque temps T, ce qui sera précisé ultérieurement.

4.3.3 Conception des sessions de test

Les tests ont pour but de mesurer les taux de faux positifs et de faux négatifs obtenus par différents algorithmes soumis à des conditions environnementales identiques. Cet exercice a pour but de mesurer lequel des algorithmes sous observation se comporte le mieux.

L'objectif est d'évaluer les taux de faux positifs pour les algorithmes DSSL, PSSG et PSAR lorsque nous faisons varier les taux de trafic gris et de trafic noir. De la même façon nous voulons évaluer les taux de faux négatifs pour les mêmes algorithmes que précédemment mentionnés lorsque nous faisons varier les taux de trafic gris et de trafic noir.

4.3.4 Déroulement des tests

L'environnement dans lequel l'expérience se déroule, bien que uniforme, a les caractéristiques suivantes :

- Il y a injection de trafic gris sur chaque hôte de façon indépendante à un moment déterminé par une fonction aléatoire avec une distribution exponentielle ayant comme temps moyen entre les événements la valeur γ_i.
- Il y a injection de trafic gris sur tous les hôtes à un moment déterminé par une fonction aléatoire avec une distribution exponentielle ayant comme temps moyen entre les événements la valeur λ.

L'expérience possède une durée fixe de T secondes, durant laquelle à toutes les t secondes, diviseur de T, un jugement est porté par les différents algorithmes, à savoir

s'il y a eu intrusion ou non et s'il y a présence de faux positifs ou de faux négatifs.

L'expérience se déroule en deux temps. En premier lieu, nous faisons varier le taux de trafic gris et observons l'incidence sur les taux de faux positifs et de faux négatifs obtenus. Dans un deuxième temps, nous faisons varier le taux de trafic noir et observons les conséquences sur les taux de faux positifs et de faux négatifs.

L'expérience consiste à mesurer les éléments suivants :

- taux de faux positifs et de faux négatifs pour l'algorithme DSSL en fonction du taux de trafic gris et du taux de trafic noir ;
- taux de faux positifs et de faux négatifs pour l'algorithme PSSG en fonction du taux de trafic gris et du taux de trafic noir ;
- taux de faux positifs et de faux négatifs pour l'algorithme PSAR en fonction du taux de trafic gris et du taux de trafic noir.

Ce test complexe et complet consiste, pour chaque valeur de trafic gris, à essayer chaque valeur de trafic noir dans notre plage de test et à mesurer pour chaque couple de valeur gris-noir, le taux de faux positifs et de faux négatifs selon chacun des algorithmes à l'étude, i.e. :

```
A={DSSL,PSSG,PSAR}
G={1,2,...,12}  // Valeurs utilisées pour le taux de trafic gris
B={1,2,...,12}  // Valeurs utilisées pour le taux de trafic noir (le taux d'attaque)
for each g ∈ G {
        for each b ∈ B    {
                for each a ∈ A{
                        getValueOf α(g,b)
                        getValueOf β(g,b)
                }
        }
}
```

FIGURE 4.2 Algorithmes de détection

Comme vous pouvez le constater à la Figure 4.2, nous ferons varier les taux de trafic gris et de trafic noir de 1 événement par unité de temps (t) jusqu'à 12 événements par unité de temps afin de pouvoir comparer avec des valeurs obtenues avec notre modèle mathématique de la section 3.6.

Autrement dit, $g = 1$ signifie un taux de 1 événement gris par période de temps t, soit dans notre cas un taux γ_i de 1 événement gris par 30 secondes. Il en va de

même pour les taux de trafic noir λ. Une expérience consiste à déterminer les taux de faux positifs et de faux négatifs pour les trois algorithmes à l'étude, étant donné une combinaison de taux de trafic gris et de trafic noir. Dans le cadre de ce mémoire, les résultats obtenus l'ont été en menant 144 expériences d'une durée de 50 minutes chacune de façon à obtenir 100 échantillons par expérience dont nous calculerons la moyenne pour obtenir les taux de faux positifs et de faux négatifs.

Dans le cas de l'algorithme PSAR, étant donné la nature de l'algorithme et la précision du champ DATE de la base de données, à la seconde près, nous considérons les événements survenant durant la même seconde comme étant corrélés. Aussi pour chaque période de 30 secondes, nous évaluons donc 30 fenêtres de 1 seconde pour en évaluer le taux de faux positifs et de faux négatifs. Pour nos expériences, nous avons choisi de fixer le nombre d'hôtes à 10 et le seuil local (s_i) de détection d'intrusions à 10 aussi.

4.4 Analyse des résultats

Suite à la réalisation de nos expériences, nous procéderons maintenant à l'analyse des résultats obtenus. Nous comparerons les résultats expérimentaux avec les résultats théoriques exposés au chapitre précédent.

4.4.1 Observation du taux de faux positifs et de faux négatifs en fonction du taux de trafic gris

Dans cette section nous analysons l'impact que peut avoir une variation des taux de trafic gris sur les taux de faux positifs et de faux négatifs.

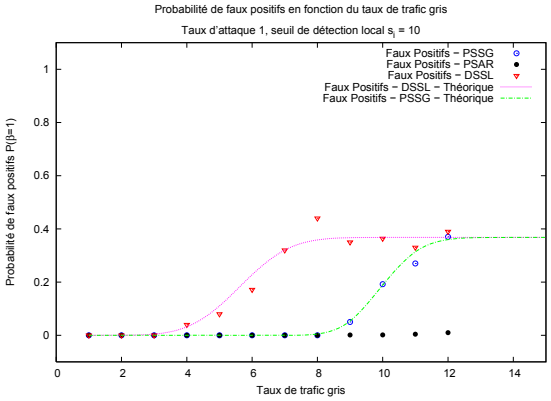
Taux de faux positifs

En observant la variation du taux de faux positifs en fonction du taux de trafic gris, nous observons tout d'abord que les relations observés lors de l'évaluation théorique tiennent. C'est-à-dire que les courbes de données obtenues pour les différents algorithmes ont les mêmes tendances que les valeurs théoriques obtenues par calcul à la section précédente. Nous constatons en effet que les taux de faux positifs avec l'algorithme PSSG ont des valeurs plus basses que ceux obtenus avec l'algorithme DSSL. Nous constatons par ailleurs que les résultats sont bornées en haut, comme pour les

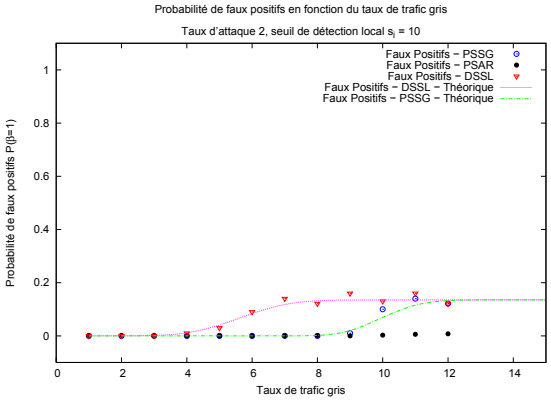
valeurs théoriques, à un moins la probabilité d'obtenir un événement de trafic noir. Plus nous augmentons le taux de trafic noir, plus la probabilité d'obtenir un faux positif diminue comme le montre la Figure 4.3. L'algorithme PSAR montre des taux de faux positifs constants, avoisinant zéro indépendamment de la variation des taux de trafic gris et de trafic noir.

Taux de faux négatifs

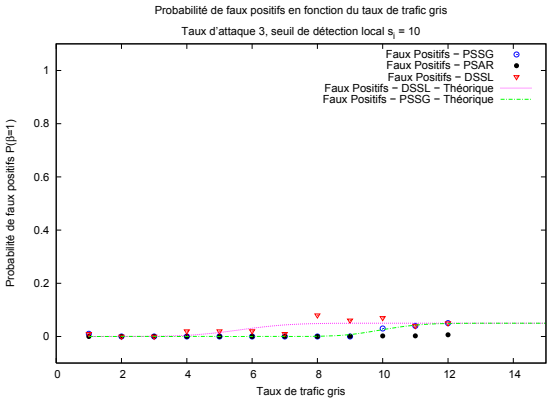
À la Figure 4.5, lorsque nous observons les taux de faux négatifs, cette fois-ci en fonction du taux de trafic gris, nous constatons que les résultats obtenus sont cohérents avec les données théoriques. Les taux de faux négatifs ont des valeurs plus élevées, plus le taux d'attaque augmente. Par ailleurs, il est facile de constater que ce taux tend vers zéro à mesure que la quantité de trafic s'approche de la valeur seuil pour le déclenchement d'une alarme. En effet, lorsque la quantité de trafic égale ou dépasse le seuil, alors une alarme est déclenchée et il n'y a donc plus de faux négatifs. Si l'on compare les algorithmes entre eux, comme le modèle mathématique l'a montré, on constate que l'algorithme PSAR fait mieux que DSSL et PSSG, tant au niveau des faux positifs que des faux négatifs, avec des taux de faux négatifs s'approchant de 0 indépendamment de la variation de trafic gris et de trafic noir. Par ailleurs, comme nous l'avions constaté précédemment, les taux de faux positifs pour PSSG sont inférieurs à ceux obtenus avec DSSL et nous observons le contraire avec les taux de faux négatifs.



(a) $\lambda = 1,\ s_i = 10$

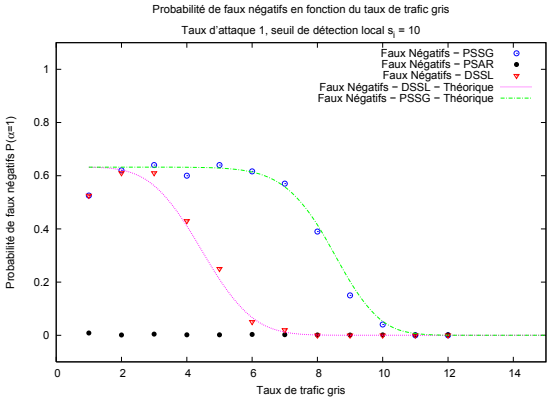


(b) $\lambda = 2,\ s_i = 10$

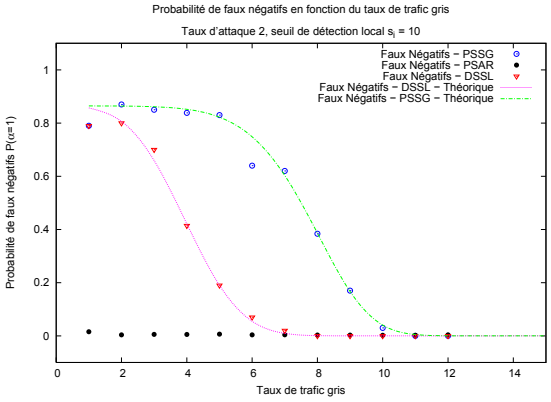


(c) $\lambda = 3,\ s_i = 10$

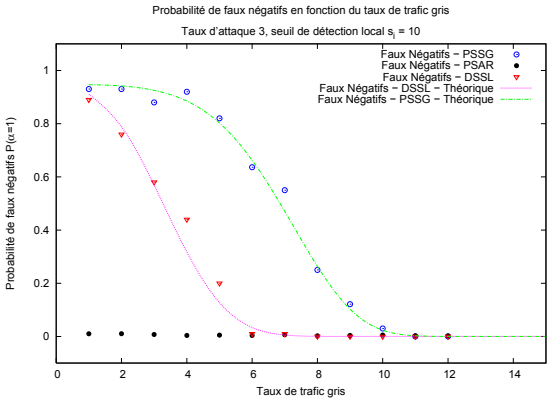
FIGURE 4.3 Taux de faux positifs(β) en fonction du taux de trafic gris(γ_i)



(a) $\lambda = 1,\ s_i = 10$



(b) $\lambda = 2,\ s_i = 10$



(c) $\lambda = 3,\ s_i = 10$

FIGURE 4.4 Taux de faux négatifs(α) en fonction du taux de trafic gris(γ_i)

4.4.2 Observation du taux de faux positifs et de faux négatifs en fonction du taux de trafic noir

Dans cette section nous analysons l'impact que peut avoir une variation des taux de trafic noir sur les taux de faux positifs et de faux négatifs.

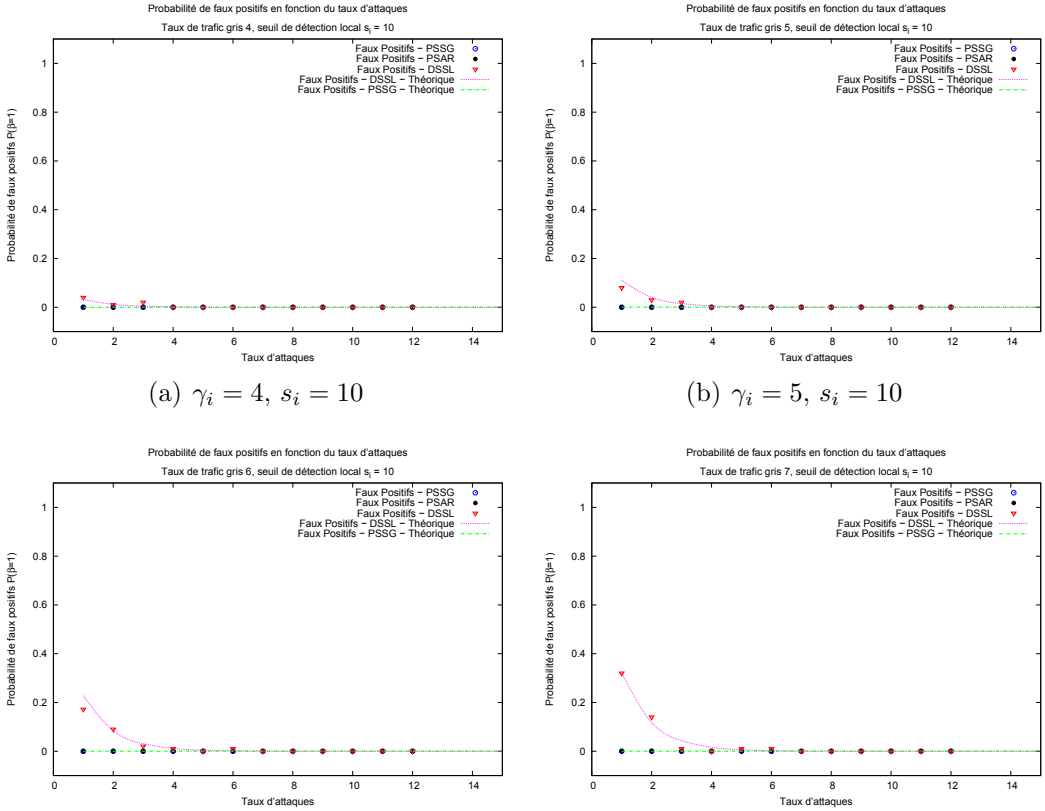
Taux de faux positifs

Lorsque nous observons la variation du taux de faux positifs en fonction du taux de trafic noir, nous constatons que, comme à la section précédente, les relations observées en théorie s'appliquent aux données expérimentales. En effet, si l'on observe les graphiques 4.5 et 4.6, nous constatons que les taux de faux positifs ont des valeurs plus élevées lorsque les taux de trafic gris augmentent. Nous constatons néanmoins que la croissance est négative compte tenu du fait que nous augmentons les taux de trafic noir, ce qui a pour effet de faire diminuer la probabilité de faux positifs. Nous sommes d'ailleurs en mesure d'observer des taux de faux positifs plus faibles pour l'algorithme PSSG que pour l'algorithme DSSL, tel que nous nous y attendions. Par ailleurs, on constate que l'algorithme PSAR se comporte beaucoup mieux que les deux autres maintenant des taux de faux positifs près de zéro indépendamment de la variation du taux d'attaque.

Taux de faux négatifs

Si nous considérons maintenant, aux figures 4.7 et 4.8, l'impact d'une variation du taux d'attaque compte tenue d'un certain taux de trafic gris sur les faux négatifs, nous constatons que les résultats obtenus demeurent cohérents avec les résultats théoriques. Nous observons d'abord que cette probabilité va en augmentant à mesure que le taux d'attaque augmente pour ensuite plafonner et diminuer tranquilement. En effet, la croissance du taux de faux négatifs est due à l'augmentation du taux d'attaque, alors que la quantité de trafic injectée demeure en deça du seuil d'alarme. Par la suite la quantité de trafic devient importante et la probabilité d'alarme augmente, entraînant ainsi une diminution de la probabilité de faux négatifs. Nous constatons par ailleurs que l'augmentation du taux de trafic gris a pour effet de diminuer la probabilité de faux négatifs. En effet, nous pouvons constater qu'un volume important de trafic a pour effet de faire déclencher l'alarme plus rapidement et donc à diminuer les taux de faux négatifs. Cette fois-ci, nous constatons que l'algorithme DSSL montre des taux

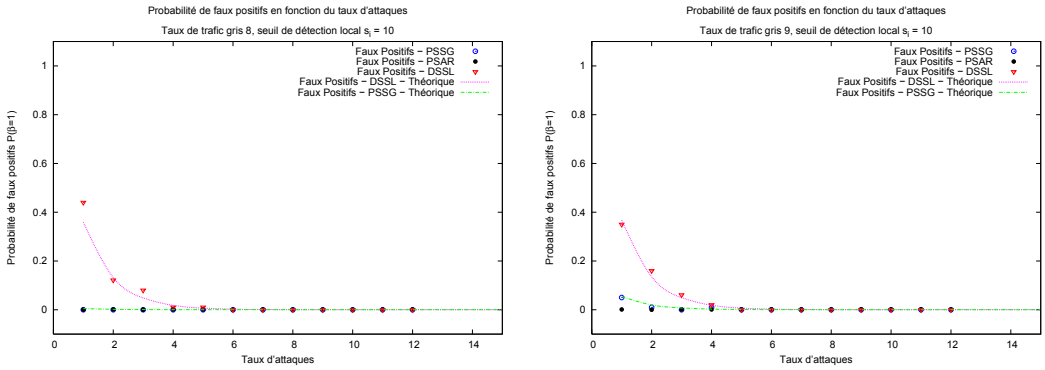
de faux négatifs inférieurs à ceux de l'algorithme PSSG. Tel qu'attendu, l'algorithme DSSL montre des taux de faux négatifs près de zéro, et ce, indépendamment des taux de trafic gris et de trafic noir.



(a) $\gamma_i = 4,\ s_i = 10$ (b) $\gamma_i = 5,\ s_i = 10$

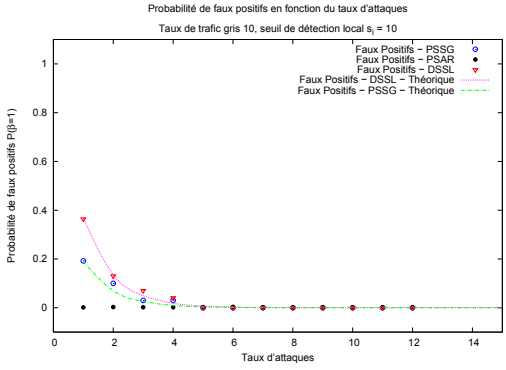
(c) $\gamma_i = 6,\ s_i = 10$ (d) $\gamma_i = 7,\ s_i = 10$

FIGURE 4.5 Taux de faux positifs(β) en fonction du taux de trafic noir(λ)

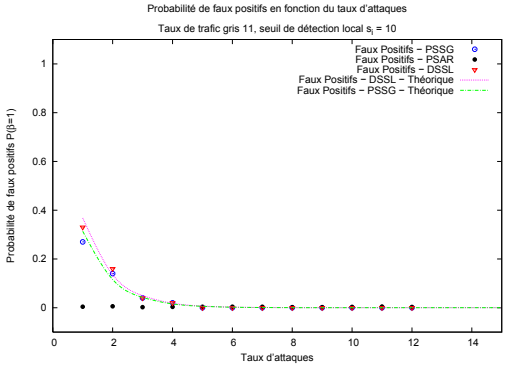


(a) $\gamma_i = 8,\ s_i = 10$

(b) $\gamma_i = 9,\ s_i = 10$

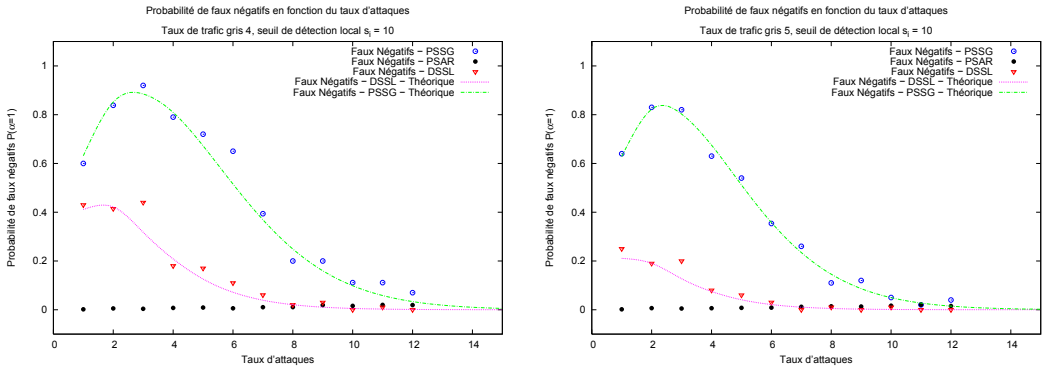


(c) $\gamma_i = 10,\ s_i = 10$



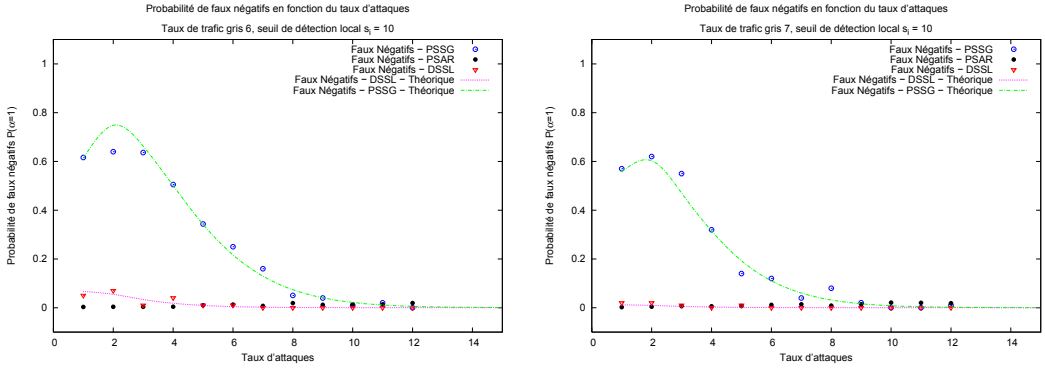
(d) $\gamma_i = 11,\ s_i = 10$

FIGURE 4.6 Taux de faux positifs(β) en fonction du taux de trafic noir(λ)



(a) $\gamma_i = 4,\ s_i = 10$

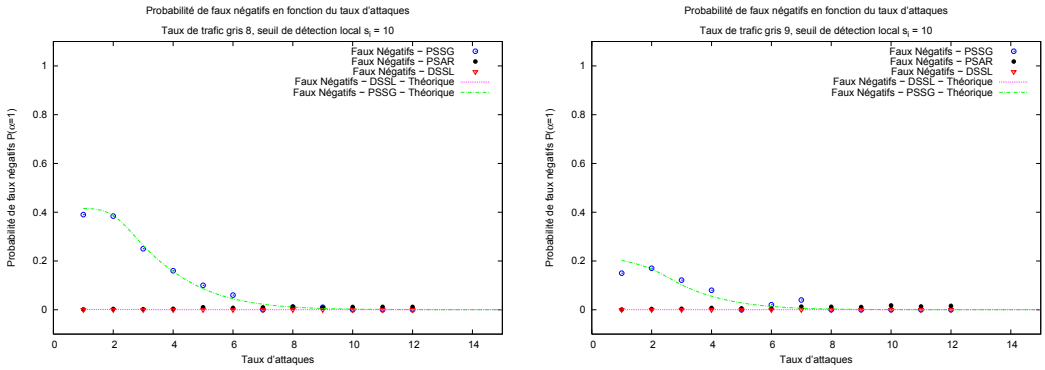
(b) $\gamma_i = 5,\ s_i = 10$



(c) $\gamma_i = 6,\ s_i = 10$

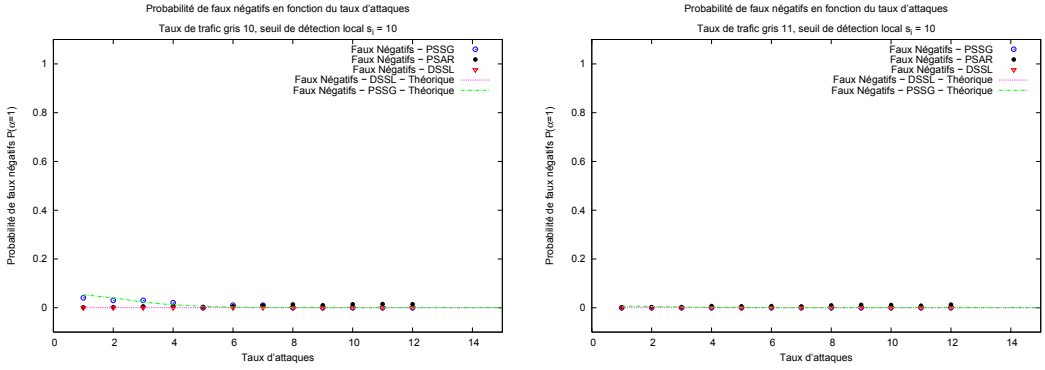
(d) $\gamma_i = 7,\ s_i = 10$

FIGURE 4.7 Taux de faux négatifs(α) en fonction du taux de trafic noir(λ)



(a) $\gamma_i = 8,\ s_i = 10$

(b) $\gamma_i = 9,\ s_i = 10$



(c) $\gamma_i = 10,\ s_i = 10$

(d) $\gamma_i = 11,\ s_i = 10$

FIGURE 4.8 Taux de faux négatifs(α) en fonction du taux de trafic noir(λ)

4.5 Différences entre les résultats théoriques et expérimentaux

Comme le montre les graphiques présentés précédemment, une certaine différence est observable entre les résultats théoriques attendus et les résultats expérimentaux obtenus. Nos résultats expérimentaux montrent une variance maximale de $\sim 25\%$ avec une déviation standard de $\sim 5\%$. La distance maximale observée entre une valeur expérimentale et un résultat théorique attendu ne fut pas supérieure à 12% avec une distance moyenne de $\sim 3,8\%$. Cette différence s'explique entre autres par des détails au niveau de l'implantation des algorithmes ainsi que l'environnement dans lequel se sont déroulés les tests. La comparaison entre les résultats théoriques et pratiques nous permet de mettre en lumière l'importance de l'environnement lors de l'exécution des tests.

Les expériences nécessitent entre autres un usage important de nombres aléatoires qui suivent une distribution exponentielle. La qualité de la source de nombres aléatoires, dont nous ne pouvons garantir la qualité, représente une source importante de variation des résultats. Lors de la réalisation de nos expériences nous nous fiés au générateur de nombres aléatoires *Jakarta Commons-Math*. En rétrospective, il aurait été souhaitable de tester la qualité des valeurs aléatoires retournées pour s'assurer de notre distribution de Poisson. Par ailleurs, compte tenu de notre environnement de test minimaliste étant donné les moyens à notre disposition, le déroulement déterministe dans le temps des différentes composantes logicielles roulant en parallèle ne peut être garanti. D'ailleurs, le fait que les tests ont été réalisés sur un ordinateur ne possédant qu'un seul processeur nuit à un vrai déroulement parallèle de l'application. Il existe une certaine gigue non négligeable entre le moment où les événements doivent survenir et l'instant réel auquel ils surviennent. Dans les faits, il est impossible de lancer simultanément les attaques sur tous les hôtes à la fois. Cette injection de trafic ne peut se faire, au mieux, que dans un intervalle de temps très restreint. Il en va de même pour l'insertion d'événements dans la base de données qui simule sur une seule machine un processus qui devrait avoir lieu sur plusieurs machines simultanément, introduisant ainsi des délais supplémentaires.

Une source de variabilité majeure des résultats provient de l'arrondi sur la date que force le format du champ DATE du schéma de la base de données MYSQL de SNORT. En effet, la précision de la date est limitée à la seconde près. Cette particularité a

pour effet de causer une certaine variabilité des résultats causés par la façon dont le système gère les arrondis. Particulièrement dans le cas de l'algorithme PSAR nous pouvons remarquer à l'occasion, par exemple, que pour 10 événements noirs devant survenir durant la même seconde, 7/10 se produiront dans la seconde alors que 3/10 surviendront durant la seconde suivante. Il se peut même que des événements, dit simultanés, puissent être inscris sur 3 secondes différentes ! Il va sans dire que ceci peut engendrer une certaine variation au niveau du nombre de faux positifs et de faux négatifs.

4.6 Vérification expérimentale des hypothèses

Suite à nos simulations, nous sommes en mesure d'affirmer que les hypothèses 1 et 2 énoncées au chapitre 3 sont vérifiées. En effet, il existe une relation entre les taux de faux positifs et les taux de trafic gris et de trafic noir, et de la même façon il en existe une entre les taux de faux négatifs et les taux de trafic gris et de trafic noir.

Nous constatons que l'augmentation du taux de trafic noir a pour effet de diminuer la probabilité d'obtenir un faux positif. De la même façon, les graphiques montrent bien que la probabilité d'obtenir un faux négatif augmente.

Inversement, nous observons qu'une augmentation du taux de trafic gris entraîne une diminution du taux de faux négatifs et une augmentation du taux de faux positifs.

Par ailleurs, nous constatons que les hypothèses 3 et 4 sont vérifiées. En effet, les taux de faux positifs de PSSG, pour nos paramètres choisis, sont toujours inférieurs aux taux obtenus par l'algorithme DSSL. Aussi, les taux de faux négatifs de PSSG sont toujours supérieurs à ceux obtenus avec DSSL. L'algorithme PSAR, quant à lui, avec ses taux de faux positifs et de faux négatifs très bas, montre qu'il est supérieur aux deux autres, sauf peut-être pour un petit nombre d'hôtes comme nous avons pu le constater au chapitre 3.

Par ailleurs, il nous a été permis de valider l'hypothèse 5 voulant qu'il puisse exister une relation entre différentes métriques. En effet, nous avons pu montrer que les taux de faux positifs croissent de façon inversement proportionnelle aux taux de faux négatifs.

Les hypothèses que nous avons vérifiées dans ce mémoire montrent l'influence que peut avoir l'environnement sur les résultats de détection d'intrus. Nous avons clairement montré que les taux de trafic gris et de trafic noir de l'environnement influencent

les valeurs de faux positifs et de faux négatifs que nous avons obtenus. Les performances ou contre-performances des différents algorithmes dépendent directement des variables environnementales dont font partie les taux de trafic gris et de trafic noir.

De plus, les performances du système de détection d'intrus, étant donné un environnement donné, dépendent entre autres des caractéristiques propres à l'IDS. En effet, l'usage d'un algorithme par comptage local comme DSSL versus un algorithme par comptage global tel que PSSG ou versus un algorithme basé sur les signatures comme PSAR influence les résultats obtenus. De la même façon, le nombre d'hôtes et les niveaux seuils ont aussi leur influence. En effet, un système de détection d'intrus, dépendamment de ses caractéristiques, peut se comporter plus ou moins bien en fonction de l'environnement auquel il est soumis.

CHAPITRE 5

CONCLUSIONS

Dans ce mémoire nous avons d'abord souligné les différentes contributions à la recherche des équipes et chercheurs qui ont développé des solutions originales de détection d'intrus et particulièrement ceux qui ont contribué au développement de solutions de détection d'intrus à l'aide d'agents mobiles.

Nous avons mis en lumière la problématique de la relativité des résultats des tests de performance des IDS. Afin d'encâdrer les tests de détection d'intrus, nous avons proposé une formulation des tests de performance en fonction des variables environnementales pouvant influencer les performances de détection et en fonction des caractéristiques propres aux systèmes de détection d'intrus.

Dans un deuxième temps, afin de traiter la problématique de la caractérisation de l'environnement et des IDS, nous avons proposé une taxonomie des caractéristiques de l'environnement propre à influencer les performances de détection. De la même façon, nous avons introduit une nouvelle taxonomie des caractéristiques des IDS en introduisant des conceptes de mobilité. Par ailleurs, nous avons choisi un ensemble de métriques adéquates pour caractériser les performances des systèmes de détection d'intrus.

Afin de valider notre cadre théorique, nous avons décidé d'énoncer cinq hypothèses selon notre cadre et de les prouver. Pour ce faire, nous avons construit un modèle mathématique qui nous a permis d'évaluer les performances en terme de taux de faux positifs et de faux négatifs pour trois algorithmes différents. Ce modèle se veut une représentation simplifiée de l'environnement comprenant un scénario de trafic et d'attaque simplifié. Dans ce modèle, le trafic est divisé entre le trafic « gris » qui est bénin mais d'apparence suspecte pour un IDS et le trafic « noir » qui lui représente le trafic généré par les attaques. Les algorithmes comparés sont : Détection Standard avec Seuil Local (DSSL), Patrouille de Surveillance avec Seuil Global (PSSG) et Patrouille de Surveillance avec Analyse Raffinée (PSAR). Pour chaque algorithme, nous avons fait varier les différentes variables environnementales et de l'IDS auxquelles étaient

soumis les IDS à l'étude. Ainsi, nous avons pu déterminer et comparer les performances de chacun des algorithmes à l'étude dans le cadre de notre environnement de test. Plus précisément, nous avons été en mesure de valider les hypothèses suivantes :

- Pour nos trois algorithmes, il existe une relation fonctionnelle entre les taux de faux positifs et les taux de trafic gris et de trafic noir ;
- De la même façon il existe une relation fonctionnelle entre les taux de faux négatifs et les taux de trafic gris et de trafic noir ;
- L'algorithme PSSG présente des taux de faux positifs inférieurs à l'algorithme DSSL mais il présente des taux de faux négatifs supérieur à DSSL ;
- L'algorithme PSAR présente des taux de faux positifs et de faux négatifs inférieurs aux deux autres algorithmes ;
- Finalement, lorsque nous faisons varier le seuil de détection, nous observons, pour les trois algorithmes, une amélioration des taux de faux positifs mais une dégradation des taux de faux négatifs et vice-versa.

Par la suite, nous avons construit une plate-forme de test afin de valider notre modèle mathématique. Cette plate-forme était constituée d'un module d'injection de trafic ainsi que d'une plate-forme de test capable d'évaluer les performances des différents algorithmes à l'étude. Force fut de constater que l'implantation de notre plate-forme produisait des résultats en ligne avec ce que nous nous attendions avec notre modèle mathématique. Ceci n'est pas surprenant puisque notre expérience fut bâtit selon notre modèle théorique. Le fonctionnement a permi de montrer que les algorithmes implantés correspondent au modèle mathématique.

5.1 Synthèse des contributions

Les travaux réalisés dans ce mémoire ont permis de mettre en lumière le lien entre l'environnement, les caractéristiques des IDS et les performances de détection de ces derniers et ainsi d'améliorer la théorie de performance des IDS avec agents mobiles. Nous avons ainsi contribué à une nouvelle taxonomie de l'environnement pour la sécurité informatique qui permet de mieux comprendre ce dernier. Aussi, nous enrichissons la taxonomie des systèmes de détection d'intrus en y rajoutant des éléments de mobilité pour prendre en compte les nouveaux systèmes de détection d'intrus basés sur les agents mobiles.

Dans ce mémoire, nous avons aussi enrichi le corpus de connaissances avec la

modélisation mathématique de deux algorithmes par comptage (DSSL et PSSG) et d'un algorithme (PSAR) que nous avons proposé qui utilise des règles de corrélation. Entre autres, grâce à ces modèles, il fut possible de montrer que, dans la majeure partie des cas, l'utilisation de règles de corrélation peut réduire de façon substantielle, le nombre de faux positifs et de faux négatifs en réduisant le nombre de dimensions du problème. Il nous a été aussi possible de montrer qu'un algorithme par comptage global tel que PSSG se comporte mieux en terme de faux positifs mais moins bien en terme de faux négatifs qu'un algorithme par comptage local comme DSSL.

5.2 Recherches futures

Au terme de notre projet, nous avons atteint les objectifs que nous nous étions fixés. En effet, nous avons réussi à démontrer les hypothèses que nous avions avancées. Nous avons été en mesure de modéliser mathématiquement les algorithmes que nous avions décidé d'évaluer et nous avons réussi à simuler le comportement de nos différents algorithmes. Nous avons montré l'impact de l'environnement et des caractéristiques des IDS sur les performances de détection dans le cadre du modèle théorique de l'environnement et des algorithmes simplifiés proposés.

Lors de nos recherches, nous avons volontairement réduit la complexité des scénarios de détection à évaluer afin de réduire le nombre de variables de l'environnement et de l'IDS à considérer pour notre analyse. Le but ici n'était pas de produire une solution commerciale mais bien d'évaluer le cadre théorique proposé. Par ailleurs, dans le but de faire simple mais non simpliste, nous avons choisis trois algorithmes de détection relativement génériques mais qui sont de bons représentants de leur catégorie.

Malheureusement, compte tenu des limites imposées à l'expérience précédemment nommées nous déplorons le fait que la simulation corresponde un peu trop bien à la modélisation. Il serait intéressant de pouvoir évaluer les différents algorithmes dans une situation de vie réelle avec du vrai trafic (Internet ou autre) et en observer le comportement. De plus, ce genre de simulation permettrait de qualifier un environnement plus riche. D'ailleurs, l'emploi de vrais algorithmes commerciaux de détection permetterait une évaluation plus fidèle de l'état de l'art en terme de performance de détection d'intrus.

Nos recherches ont montré de grandes opportunités pour la réduction de faux positifs et de faux négatifs grâce à l'emploi de techniques de corrélation d'informa-

tion. Compte tenu du fait que bien souvent, l'amélioration d'une métrique entraîne la détérioration d'une autre comme nous l'avons montré avec les faux positifs et les faux négatifs, nous croyons qu'il importe de réduire les dimensions du problème de détection d'intrus en corrélant les informations provenant des différents éléments de l'environnement. Par ailleurs, compte tenu que même une solution très optimisée pour certains cas précis peut se montrer inférieure à une solution plus générique dans d'autres cas, nous croyons qu'il serait intéressant d'investiguer un écosystème d'IDS, chacun optimisés pour certaines attaques et certains types d'environnement et qui corréleraient leurs informations entre eux. Nous sentons le besoin pour une solution de détection adaptative qui pourrait se modifier en fonction de variation dans son environnement.

Bien que, dans ce mémoire, nous n'avons pas réussi à optimiser les résultats des taux de faux positifs et de faux négatifs, il serait intéressant, dans des recherches futures, de prendre en compte des variables économiques comme le coût de traitement en dollard d'une alarme et l'impact de ne pas pouvoir traiter une vraie alarme (i.e. le coût d'un faux négatif). Ceci nous permettrait potentiellement d'optimiser en terme d'impact monétaire (coût de traitement vs. risque) le compromis entre taux faux négatif et faux positif qui est optimal étant donné l'environnement et les différents algorithmes de détection présentés ici. Ainsi nos résultats quantitatifs pourraient être d'utilisation immédiate par des responsables de sécurité informatique dans l'industrie.

Notre expérience avec les agents mobiles montre que ce paradigme de programmation est bien adapté à la corrélation spatiale et temporelle d'information et que cela présente un bon moyen pour analyser des données archivées. Toutefois, ils semblent moins bien adaptés pour une analyse temps réel en continu compte tenu de la charge supplémentaire induite par leur déplacements.

Bibliographie

ALBERS, P., CAMP, O., PERCHER, J.-M., JOUGA, B., MÉ, L. et PUTTINI, R. S. (2002). Security in ad hoc networks : A general intrusion detection architecture enhancing trust based approaches. Q. H. Mahmoud, éditeur, *Proceedings of the First International Workshop on Wireless Information Systems (WIS)*. ICEIS Press, 1–12.

ANDERSON, J. (1980). Computer security threat monitoring and surveillance. Rapport technique, Washington, PA, James P. Anderson Co.

ASAKA, M., TAGUCHI, A. et GOTO, S. (1999). The implementation of IDA : An intrusion detection agent system. Rapport technique, Waseda University.

AXELSSON, S. (1999). The base-rate fallacy and its implications for the difficulty of intrusion detection. *ACM Transactions on Information and System Security*. Singapore, 130–143.

BERNARDES, M. C. et DOS SANTOS MOREIRA, E. (2000). Implementation of an intrusion detection system based on mobile agents. *International Symposium on Software Engineering for Parallel and Distributed Systems*. Limerick, 158–164.

BESSON, J.-L. (2003). *Next Generation Intrusion Detection and Prevention for Complex Environments*. Master thesis in computer science, University of Zurich.

CABRI, G., LEONARDI, L. et ZAMBONELLI, F. (2000). Mobile-agent coordination models for internet applications. *Computer*, 33, 82–89.

CURRY, D., DEBAR, H. et FEINSTEIN, B. (2006). *The Intrusion Detection Message Exchange Format*. IETF. Status : DRAFT STANDARD.

DASGUPTA, D. et BRIAN, H. (2001). Mobile security agents for network traffic analysis. *DARPA Information Survivability Conference & Exposition II (DISCEX)*. Anaheim, CA, États-Unis, vol. 2, 332–340.

DE QUEIROZ, J. D., DA COSTA CARMO, L. F. R. et PIRMEZ, L. (1999). Micael : An autonomous mobile agent system to protect new generation networked applications. *2nd Annual Workshop on Recent Advances in Intrusion Detection*.

DEBAR, H., DACIER, M. et WESPI, A. (1999). Towards a taxonomy of intrusion-detection systems. *Comput. Networks*, 31, 805–822.

DEETER, K., SINGH, K., WILSON, S., FILIPOZZI, L. et VUONG, S. (2004). APHIDS : A mobile agent-based programmable hybrid intrusion detection system. *1st International Workshop on Mobility Aware Technologies and Applications (MATA)*. Florianopolis, Brazil, vol. 3284, 244–253.

DENNING, D. (1987). An Intrusion-Detection Model. *IEEE Transactions on Software Engineering*, 13, 222–232.

DENNING, D. et NEUMANN, P. (1985). *Requirements and Model for IDES-a Real-time Intrusion-detection Expert System : Final Report*. SRI International.

FAJARDO, J. V. (2005). Mobile agent-based architecture for intrusion detection. Projet de fin d'étude, École Polytechnique de Montréal.

FEINSTEIN, B., MATTHEWS, G. et WHITE, J. (2003). *The Intrusion Detection Exchange Protocol (IDXP)*. IETF. Status : DRAFT STANDARD.

FINK, G., CHAPPELL, B., TURNER, T. et O'DONOGHUE, K. (2002). A metrics-based approach to intrusion detection system evaluation for distributed real-time systems. *International Parallel and Distributed Processing Symposium (IPDPS '02 (IPPS & SPDP))*. IEEE, Washington - Brussels - Tokyo, 93–93.

GUANGCHUN, L., XIANLIANG, L., JIONG, L. et JUN, Z. (2003). MADIDS : a novel distributed IDS based on mobile agent. *ACM SIGOPS Operating Systems Review*, 37, 46–53.

HELMER, G., WONG, J. S., HONAVAR, V., MILLER, L. et WANG, Y. (2003). Lightweight agents for intrusion detection. *The Journal of Systems and Software*, 67, 109—122.

JANAKIRAMAN, R., WALDVOGEL, M. et ZHANG, Q. (2003). Indra : A peer-to-peer approach to network intrusion detection and prevention. *Proceedings of the*

Twelfth IEEE International Workshops on Enabling Technologies : Infrastructure for Collaborative Enterprises (WETICE). IEEE Computer Society, 226–231.

KACHIRSKI, O. et GUHA, R. (2002). Intrusion detection using mobile agents in wireless ad hoc networks. *IEEE Workshop on Knowledge Media Networking*. IEEE, Limerick, 153–158.

KRUEGEL, C. et TOTH, T. (2000). A survey on intrusion detection systems. Rapport technique, University of Technology, Vienna.

KRUEGEL, C. et TOTH, T. (2001). Applying mobile agent technology to intrusion detection. *In ICSE Workshop on Software Engineering and Mobility*.

KRUEGEL, C. et TOTH, T. (2002). Flexible, mobile agent based intrusion detection for dynamic networks. *Technical University of Vienna Information Systems Institute Distributed Systems Group*, 27, 12.

LEE, W. et XIANG, D. (2000). Information-theoretic measures for anomaly detection. *IEEE Symposium on Security and Privacy*. New York, 130–143.

LEVCHENKO, K., PATURI, R. et VARGHESE, G. (2004). On the difficulty of scalably detecting network attacks. *Proceedings of the 11th ACM conference on computer and communications security*. ACM Press, New York, NY, USA, 12–20.

LI, C., SONG, Q. et ZHANG, C. (2004). MA-IDS architecture for distributed intrusion detection using mobile agents. *Proceedings of the 2nd International Conference on Information Technology for Application (ICITA)*. 451–455.

MARKS, D. G., MELL, P. et STINSON, M. (2004). Optimizing the scalability of network intrusion detection systems using mobile agents. *Journal of Network and Systems Management*, 12, 95–110.

MAXION, R. A. et TAN, K. M. (2000). Benchmarking anomaly-based detection systems. *International Conference on Dependable Systems and Networks*. IEEE, New York, 623–630.

MELL, P. et MCLARNON, M. (1999). Mobile agent attack resistant distributed hierarchical intrusion detection systems. *Second International Workshop on the*

Recent Advances in Intrusion Detection. West Layfayette, Indiana, États-Unis, 7–9.

MISHRA, A., NADKARNI, K. et PATCHA, A. (2004). Intrusion detection in wireless ad hoc networks. *IEEE Wireless Communications*, 11, 48–60.

NORTHCUTT, S. et NOVAK, J. (2002). *Network Intrusion Detection*. David Dwyer, troisième édition.

PERCHER, J.-M. et JOUGA, B. (2002). Détection d'intrusions dans les réseaux ad hoc. Rapport technique, Ecole Supérieure d´Electronique de l'Ouest (ESEO).

PING, Y., YAN, Y., YAFEI, H., YIPING, Z. et SHIYONG, Z. (2004). Securing ad hoc networks through mobile agent. *InfoSecu '04 : Proceedings of the 3rd international conference on Information security*. ACM Press, New York, NY, USA, 125–129.

RAMACHANDRAN, G. et HART, D. (2004). A P2P intrusion detection system based on mobile agents. *ACM Proceedings of the 42nd annual Southeast regional conference*. ACM Press, Huntsville, Alabama, 185–190.

RANUM, M. J. (2003). False positives : A user's guide to making sense of IDS alarms. Rapport technique, ICSA Labs IDSC.

ROESCH, M. (1999). Snort-Lightweight Intrusion Detection for Networks. *Proceedings of USENIX LISA*, 99, 229–238.

SMITH, A. B. (2001). An Examination of an Intrusion Detection Architecture for Wireless Ad Hoc Networks. *National Colloquium for Information Systems Security Education 2001*, 1.

SPAFFORD, E. H., ZAMBONI, D. et KERSCHBAUM, F. (2000). Intrusion detection using autonomous agents. *Computer Networks*, 34, 547–570.

STANIFORD-CHEN, S., CHEUNG, S., CRAWFORD, R., DILGER, M., FRANK, J., HOAGLAND, J., LEVITT, K., WEE, C., YIP, R. et ZERKLE, D. (1996). GrIDS-a graph based intrusion detection system for large networks. *Proceedings of the 19th National Information Systems Security Conference*, 1, 361–370.

STOLFO, S., PRODROMIDIS, A. L., TSELEPIS, S., LEE, W., FAN, W. et CHAN, P. K. (1997). JAM : Java agents for meta-learning over distributed databases. Rapport technique, Colombia University.

YANXIN, W., BEHERA, S. R., WONG, J., HELMER, G., HONAVAR, V., MILLER, L., LUTZ, R. et SLAGELL, M. (2004). Towards the automatic generation of mobile agents for distributed intrusion detection system. *Journal of Systems and Software*, in press, 1–14.

ZHICAI, S., ZHENZHOU, J. et MINGZENG, H. (2004). A novel distributed intrusion detection model based on mobile agent. *Proceedings of the 3rd International Conference on Information security*. ACM Press, Shanghai, China, 155–159.