République Algérienne Démocratique et Populaire
Ministre de l'enseignement supérieur et de la recherche scientifique

UNIVERSITE MOHAMED KHIDER BISKRA
FACULTE DES SCIENCES EXACES, DES SCIENCES DE LA NATURE ET DE LA VIE
DEPARTEMENT D'INFORMATIQUE

N° d'ordre :…..
Série :…..

THESE

Présentée pour obtenir le grade de
DOCTORAT EN SCIENCES EN INFORMATIQUE

Une approche à base d'agents mobiles pour la sécurité des systèmes d'informations sur le web

Présentée par :

Mme. BOUKHLOUF Djemaa

Dirigée par :

Pr. KAZAR Okba
Université Mohamed Khider Biskra

Soutenue le: //2016

Devant le jury:

Président :	**Dr. Bennoui Hamadi**	**(MCA)**	**Université de Biskra**
Rapporteur :	**Pr. Kazar Okba**	**(professeur)**	**Université de Biskra**
Examinateurs :	**Pr. Khababa Abdallah**	**(professeur)**	**Université de Sétif 1**
	Pr. Touahria Mohamed	**(professeur)**	**Université de Sétif 1**
	Dr. Kabachi Nadia	**(MCA)**	**Université Claude Bernard Lyon 1**
	Dr. Benharzallah Saber	**(MCA)**	**Université de Biskra**
Invité :	**Dr. Kahloul Laid**	**(MCA)**	**Université de Biskra**

Remerciements

Louange à Dieu le tout puissant, qui m'a donné la volonté, le pouvoir, la santé et la patience pour terminer ce travail.

Je tiens à exprimer ma profonde gratitude à monsieur Okba Kazar, Professeur à l'université Mohamed Khider Biskra, qui a dirigé cette thèse, qui m'a amicalement conseillé tout au long de ce travail et, qui m'a proposé un cadre de travail très favorable. Je le remercie d'avoir consacré beaucoup de son temps pour les nombreuses relectures de mon document et les articles publiés. J'ai beaucoup profité de sa grande expérience. Ses remarques pertinentes et constructives sur mon travail m'ont aidé à en améliorer la qualité.

Je tiens à remercier particulièrement monsieur Laid Kahloul pour toutes nos discussions, ses conseils et son aide.

Je remercie monsieur Bennoui Hamadi, maitre de conférences à l'université de Biskra, qui me fait l'honneur de présider mon jury.

Mes sincères remerciements aux membres du jury qui ont accepté de juger mon travail et dont les remarques et les suggestions auront le plus grand impact pour l'amélioration et le raffinement de ce travail. Ainsi, je remercie :

Pr. Khababa Abdallah professeur à l'université de Sétif 1,

Pr. Touahria Mohamed, professeur à l'université de Sétif 1,

Dr. Kabachi Nadia maitre de conférences à l'université Claude Bernard Lyon 1,

Dr. Benharzallah Saber maitre de conférences à l'université de Biskra.

La liste n'est pas exhaustive: Je remercie toutes les personnes, des amies aux inconnus, qui, même au cours d'une rencontre fugace, m'ont fait découvrir le plaisir d'apprendre.

Dédicace

A ma source de courage, à ceux que j'ai de plus cher:
Mon Père et Ma maman
A mon mari et à toute ma famille :Djamel, Messaoud ,Dalila, Ouarda, Hanane, Maroua
Dédicace Spéciale à mes enfants : Hibet Allah , Seif Allah et souraya
A tous mes amies

ملخص

نظم المعلومات تفتح اليوم المزيد والمزيد على الانترنت. هذا الانفتاح المفيد بالطبع يثير مشكلة رئيسية: عدد متزايد من الهجمات. وينضم هذا العمل البحثى فى مجال تقنية امن المعلومات. ومن بين التدابير المضادة ضد الهجمات الذى يهمنا فى اطار هذا العمل هو نظام اكتشاف التطفل(IDS). وضع IDS بهدف الكشف عن الاعمال الغير عادية لنظم وشبكات المعلومات, مشيرا الى ان الاجراءات لا تتفق مع سياسة الامان يقودها واحد او عدة مستخدمين.

نظام اكتشاف التطفل المركزى يعانى الكثير من المساوئ عندما تستخدم فى الشبكات ذات النطاق الترددى العريض, ولا سيما عندما تواجه بتوزيع التدخلات. وفى هذا الصدد نقترح توزيع النهج القائم على عملاء المحمول للكشف عن التدخلات. ويستخدم العمل المقترح المنهاج Aglet لانشاء وتوزيع العملاء الى اربعة انواع : عملاء المحمول لجمع المعلومات من قبل وكلاء جمع و عملاء المحمول لتحليل المعلومات. الوكيل المدعم يجعل تحليل توزيع الاحداث لكشف التطفل الموزع. ان العامل المسؤول يقوم بلطلاق عملائه وادارة الرسائل المستلمة من العملاء الاخرين.

الكلمات المفتاحية : عملاء محمولة, المنهاج Aglets, نظام اكتشاف التطفل, الطريقة الهجينة.

Abstract

Information systems are more and more opened today on the Internet. This opening, beneficial prejudice raises nevertheless a major problem: it ensues from it an increasing number of attacks. This research work joins in the field of the IT security. One of the measures anti-attacks in which one we are interested within the framework of this work is the System of Detection of the Intrusions (IDS). The IDS was developed with the aim of detecting abnormal functionings of information systems and networks, indicating that actions not corresponding to the safety policy are led by one or several users.

The centralised IDS suffer considerable limits when they used in broadband networks, and especially when they face distributed intrusions. In this work we propose a distributed approach based on mobile agents for the detection of intrusions.

The proposed approach uses the plate-form Aglet for the creation and the distribution of four types of mobile agents to make the collection of information by the agents Collectors, the analysis of packages by the mobile agents Analysers. The agent Redirector makes the analysis of the events to detect distributed intrusions. The Generator agent is responsible for launch of these agents and for management of messages received from the latter.

Keywords: Mobile agents, Aglets, Intrusion detection system, hybrid approach.

Résumé

Les systèmes d'informations sont aujourd'hui de plus en plus ouverts sur Internet. Cette ouverture, a priori bénéfique, pose néanmoins un problème majeur : il en découle un nombre croissant d'attaques. Ce travail de recherche s'inscrit dans le domaine de la sécurité informatique. Une des mesures anti-attaque à laquelle nous nous intéressons dans le cadre de ce travail est le Système de Détection des Intrusions (IDS).

Les IDS ont été développés dans le but de détecter des fonctionnements anormaux des systèmes d'informations et des réseaux, indiquant que des actions non conformes à la politique de sécurité sont menées par un ou plusieurs utilisateurs. Les IDS centralisés souffrent des limites considérables quand ils utilisés dans des réseaux hauts débits, et surtout quand ils font face à des intrusions distribuées. Dans ce travail nous proposons une approche distribuée basée sur des agents mobiles pour la détection d'intrusions.

L'approche proposée utilise la plate forme Aglet pour la création et la distribution de quatre types d'agents mobiles pour effectuer la collecte d'informations par les agents collecteurs, l'analyse de paquets par les agents mobiles analyseurs. Les agents redirecteurs effectuent l'analyse des évènements afin de détecter des intrusions distribuées. L'agent générateur est responsable de lancement de ces agents et de gestion des messages reçus de ces derniers.

Mots-clés : Agent mobile, Aglets, système de détection d'intrusion, Approche hybride.

Table des matières

Introduction générale

Chapitre 1 : Sécurité informatique

Chapitre 2 : Systèmes de détection d'intrusions IDS

Chapitre 3 : Agents mobiles et les systèmes de détection d'intrusions

Chapitre 4 : Une approche hybride basée agents mobiles pour un système de détection d'intrusions distribué

Chapitre 5 : Mise en œuvre et tests

Table des figures

Chapitre 1 : Sécurité informatique

Chapitre 2 : Systèmes de détection d'intrusion

Chapitre 4 : Une approche hybride basée agents mobiles pour un système de détection d'intrusions distribué

Chapitre 5 : Mise en œuvre et tests

Liste des tableaux

Abréviation et Acronymes

AH	Authentication Header
Aglet	Agent aplet
AUML	Agent Unified Modeling Language
CID	Confidentialité, intégrité et disponibilité
CIDF	Common Intrusion Detection Framework
CRC	Cyclic Redundancy Check
DARPA	Defense Advanced Research Projects Agency
DMZ	Zone démilitarisée
DoS	Déni de service
DSA	Algorithme de signature numérique
ESP	Encapsulating Security Payload
HIDS	Système de détection d'intrusion hôte
IETF	Internet Engineering Task Force
ICMP	Protocole de message de contrôle Internet; protocole ICMP
IDEA	International Data Encryption Algorithm; algorithme IDEA
IDES	Intrusion Detection Expert System
MEF	Intrusion Detection Message Exchange Format
ICMP	Internet Control Message Protocol
IDS	Système de détection d'intrusions
IDWG	Intrusion Detection Working Group
IDXP	Intrusion Detection eXchange Protocol
IP	Protocole Internet; protocole IP
IPS	Système de prévention d'intrusion
IPSEC	Sécurité du protocole Internet; protocole IPSEC
IP Spoofing	Usurpation d'adresse IP.
ISAKMP	IP Security Association Key Management Protocol
KQML	Knowledge Query and Manipulation Language
MD5	Message Digest 5; algorithme MD5

NIDES	Next-generation IDES
NIDS	Système de détection d'intrusion réseau
NIPS	Network Intrusion Prevention System
PCT	Private Communication Technology
PIN	Personal Identification Number
RADIUS	Service d'usager commuté à authentification distante
RC4	Rivest Cipher 4; algorithme RC4
RFC	Request For Comments
RPC	Remote Procedure Calling
RSA	Rivest-Shamir-Adleman; algorithme RSA
SET	Transaction électronique sécurisée; protocole SET
SGC	Service de gestion des certificats
SGC	Système de génération des clés
SHA	Secure Hash Algorithm
SNMP	Protocole simple de gestion de réseau; protocole SNMP
SMTP	Protocole simple de transfert de courrier; protocole SMTP
SSH	Secure Shell
SSL	Secure Socket Layer
TCP/IP	Protocole de contrôle de transmission/protocole Internet; protocole TCP/IP
VPN	Réseau privé virtuel
XML	Extensible Markup Langage

Introduction générale

1. Problématique

Le partage des ressources et des informations sur un réseau gagne de plus en plus de popularité et d'importance. La complexité et les portées des transactions réalisables sur Internet qui demeure en expansion constante, sont les enjeux majeurs de l'informatique. En réalité, les systèmes informatiques qui tendent à être de plus en plus ouverts et distribués, découvrent par la même occasion qu'ils sont de plus en plus susceptibles d'être la cible de dérèglements divers tels que les congestions, les accès malveillants et les attaques. A cet effet, il devient inéluctable de munir ces systèmes d'outils et de mécanismes de sécurité capables d'inhiber ces dérèglements.

Cependant, la mise en place d'un système de sécurité doit, d'une part, garantir la détection des tentatives malveillantes qui visent à nuire au bon fonctionnement du réseau et d'autres part, fournir les moyens de les arrêter en temps opportun ou en cas de sinistre pourvoir ce qui est nécessaire (mesures et techniques) pour la reprise et la correction.

L'arsenal déployé pour la protection des ressources et de la communication au sein d'un réseau s'acharne pour défendre l'armature de la sécurité informatique formée de quatre buts différents mais complémentaires et indissociables, qui sont : la confidentialité, l'intégrité, l'authentification et la disponibilité [21].

Renforcer la sécurité des réseaux en même temps que leur performance nous apparaît, irréfutablement, comme un enjeu technologique et économique d'importance capitale. Concrètement, pour atteindre ce but il faut munir le réseau d'un système de sécurité qui puisse offrir un ensemble de services en s'appuyant sur un ensemble de techniques et d'outils appropriés.

Les menaces deviennent de plus en plus innombrables et diverses. La délimitation des scénarii d'attaques s'avère difficile. Fort heureusement, la recherche s'applique pour donner résultat à des méthodes et des techniques de défense. Parmi ces techniques nous citons : la cryptographie, les par feux, les VPN (Réseaux privés virtuels), les proxys et les IDS (les systèmes de détection d'intrusions) [19][20]. Ces derniers ont été développés dans le but de détecter des fonctionnements anormaux des systèmes d'informations et des réseaux, indiquant que des actions non conformes à la politique de sécurité sont menées par un ou plusieurs

utilisateurs. A ce stade, plusieurs domaines sont impliqués [36] : Data mining, Réseaux de neurones, Immunologie, Algorithmes génétiques et Agents mobiles. Des recherches actuelles sont concentrées sur les IDS surtout sur les IDS distribués à base d'agents mobiles et intelligents. Il existe une panoplie de raisons qui justifient le choix d'utiliser les agents mobiles sont [2][3]:

- Réduire la charge des réseaux,
- Surmonter le temps de latence sur le réseau,
- S'exécuter de manière asynchrone et autonome,
- S'adapter dynamiquement aux environnements d'exécution,
- Hétérogènes,
- Robustes et tolérants

Les tendances de la recherche vont de la machine au réseau, d'un système monolithique vers un système distribué, de l'unicité des méthodes vers les méthodes hybrides. Les proportions des découvertes dans le domaine de la détection des intrusions comme outil logiciel contemporain de sécurité ne cessent de s'étendre.

2. Objectifs et motivations

Une des mesures anti-attaque à laquelle nous nous intéressons dans le cadre de ce travail est le Système de Détection des Intrusions (IDS). Repérer des tentatives de nuire à un système informatique de par le matériel et le logiciel, est l'apanage des IDS. La détection des intrusions consiste à scruter le trafic réseau, collecter tous les événements, les analyser et générer des alarmes en cas d'identification de tentatives malveillantes. Toutefois, une contradiction s'impose: Comment est-ce que la plupart des IDS sont monolithiques et centralisés alors que la collecte des données sur le réseau est distribuée?

Le traitement des données d'une manière centralisée induit des vulnérabilités dans le système informatique, en l'occurrence, des réductions de performances en termes de scalabilité, configurabilité et de tolérence aux pannes. En effet, il suffit que l'engin central soit défectueux pour avoir une défection de tout le système de détection des intrusions. En outre, un grand flux d'événements peut entraîner un ralentissement du temps de traitement des données, une réaction tardive du système, une surcharge du réseau voire même des pertes de données rendant toute analyse biaisée. Notons aussi que la centralisation freine l'extensibilité du système et rend sa reconfiguration difficile.

Dans cet esprit, nous nous proposons de développer un modèle et un prototype quant à l'utilisation de la technologie des agents mobiles dans la distribution du système de détection des intrusions.

En fait, deux catégories de méthodes sont utilisées pour les IDS (Intrusion Detection System) : L'approche par scénarios et l'approche comportementale. La première consiste à détecter des signatures d'attaques connues dans les paquets circulant sur le réseau. La seconde, consiste quant à elle, à détecter une activité suspecte dans le comportement de l'utilisateur. Ces deux techniques, aussi différentes soient-elles, peuvent être combinées au sein d'un même système afin d'accroître la sécurité. Cette dernière approche est dite hybride, et elle est utilisée dans notre modèle pour l'analyse des informations.

L'approche que nous proposons a pour objectif de remplir les fonctions nécessaires à la détection des attaques de sécurité. Pour réaliser cela, notre solution doit fournir les points suivants :

- une approche hybride pour détecter les attaques qui se caractérisent par des signatures définies ou inconnues ;
- un modèle collaboratif permettant la détection des attaques de nature distribuée.
- un modèle utilisant la technologie d'agents mobiles pour distribuer la détection d'intrusions.

Afin d'atteindre ces objectifs, notre travail comporte deux contributions [43][44][45]. Une consiste à distribuer le système de détection d'intrusion en se basant sur la technologie d'agents mobiles et qui est facilité par l'utilisation de la plate forme Aglet Workbench d'IBM [14]. L'autre concerne l'utilisation d'une approche hybride d'analyse des paquets capturés par un sniffer.

3. Organisation de la thèse

Le travail présenté dans cette thèse est divisé en cinq chapitres. Les trois premiers chapitres constituent un état de l'art et décrivent les concepts nécessaires pour la bonne compréhension du notre travail. Les deux derniers chapitres sont dédiés pour la présentation de notre contribution.

Le premier chapitre présente les buts de la sécurité informatique, dresse un panel non exhaustif, mais cependant représentatif, des menaces et des attaques sur la sécurité et finit par définir ce qu'est une politique de sécurité, ses approches de conception et l'architecture réseau qui la supporte ainsi les protocoles de sécurité les plus courants.

Le deuxième chapitre on arbore les systèmes de détection des intrusions de par leur définition, leurs caractéristiques, leurs méthodes et outils. Ce chapitre en soulevant, à la fin, les limites des IDS existants, motive et argumente le choix du modèle proposé dans cette thèse.

Le troisième chapitre, est dédié à la présentation de quelques notions sur les agents mobiles ainsi l'utilisation de ces derniers dans le domaine de détection d'intrusions. Ainsi nous décrivons quelques travaux récents dans l'utilisation des agents mobiles pour la distribution du système de détection d'intrusion.

Le quatrième chapitre, présente notre modèle proposé pour mettre en place un IDS distribué à base d'agents mobiles. Nous décrirons les idées, les caractéristiques de l'approche, l'architecture générale, le schéma structurel et fonctionnel des agents du notre modèle, ainsi nous décrirons le type de communication entre ces agents.

Dans le chapitre cinq on décrit les détails de réalisation d'un prototype du modèle proposé pour prouver la faisabilité de concevoir et implémenter un IDS distribué. Ce chapitre présent les algorithmes nécessaires pour l'implémentation des agents mobiles il présente aussi quelques aspects d'implémentation et une étude de cas en utilisant la plateforme Aglet.

La thèse se termine par une conclusion qui établit un bilan de ce travail et dresse les perspectives.

Chapitre 1
Sécurité informatique

1.1 Introduction

L'évolution fulgurante des réseaux informatiques et d'Internet ces dernières années a modifié le paysage économique ainsi que les différents modes de communication et d'échanges. Attirée par une flexibilité de services et de nouvelles niches économiques, la majorité des organismes privés et gouvernementaux utilisent les ressources d'Internet aussi bien pour inter-connecter des sites distants que pour proposer des services en ligne.

La sécurité des systèmes d'information est l'ensemble des moyens techniques, organisationnels, juridiques et humains nécessaires et mis en place pour conserver, rétablir, et garantir la sécurité de l'information et du système d'information [21][34].

Ce chapitre, va présenter les définitions nécessaires pour comprendre ce qui est une sécurité informatique et quels sont ses objectifs et ses domaines. Puis il aborde la notion de gestion des risques ainsi présente les différentes attaques de sécurité. Par la suite, il va entamer les politiques de sécurité, les outils de prévention et quelques protocoles de sécurité.

1.2 Objectifs de la sécurité informatique

Avant de présenter les différents objectifs de la sécurité informatique nous commençons par quelques définitions importantes :

1.2.1 Systèmes d'information

L'information se présente sous trois formes : les données, les connaissances et les messages. On a l'habitude de désigner par « système d'information » l'ensemble des moyens techniques et humains qui permet de stocker, de traiter ou de transmettre l'information. De fait, on confond souvent, même si ce n'est pas très exact, la notion de « systèmes et réseaux informatiques » et celle de « systèmes d'information (SI) ». On dira donc qu'un système d'information est « *tout moyen dont le fonctionnement fait appel d'une façon ou d'une autre à*

l'électricité et qui est destiné à élaborer, traiter, stocker, acheminer, présenter ou détruire l'information » [34].

1.2.2 Sécurité d'un système d'information

Le concept de sécurité des systèmes d'information recouvre un ensemble de méthodes, techniques et outils chargés de protéger les ressources d'un système d'information.

La sécurité informatique tente de maintenir neuf caractéristiques principales [21][27]:

– **La confidentialité:** dans la littérature, la confidentialité semble être la qualité la plus importante d'un système sûr. Elle a été est la première préoccupation des militaires. Assurer la confidentialité des données consiste à faire en sorte que les informations restent secrètes et que seules les personnes autorisées y aient accès. Les utilisateurs du système ont besoin d'avoir la garantie que leurs informations ne risquent pas d'être divulguées à des personnes non autorisées.

– **L'authentification:** fournir deux preuves en parallèle, l'une est aussi importante et indispensable que l'autre: Prouver qu'un sujet (site, personne, système, processus, etc.) est celui qu'il prétend être et que les informations reçues sont conformes à celles fournies. Pratiquement, dans le cas d'un simple message, le service d'authentification assure que le message provient de l'endroit d'où il prétend venir. Dans le cas d'un échange bidirectionnel, deux aspects sont à vérifier. Il faut assurer que les deux entités communicantes sont bien ce qu'elles affirment être. De plus, le service d'authentification doit montrer que la connexion ne peut être brouillée par une troisième entité essayant de se faire passer pour un des deux correspondants.

– **L'intégrité:** outre la confidentialité des informations, l'intégrité évite la corruption, l'altération et la destruction des données dans le réseau de manière non autorisée. L'intégrité reste un domaine très large couvrant à la fois les modifications, les moyens de modification mais également l'après-modification et donc la cohérence des données.

– **La disponibilité:** assurer aux utilisateurs légitimes une continuelle disponibilité des informations, des services et des ressources dans le réseau (les temps de réponse, la tolérance aux fautes, le contrôle de concurrence, le partage équitable de ressources, etc.). En somme, veiller à la disponibilité d'un système, c'est prévenir contre les perturbations et les interruptions de son fonctionnement et aussi contre toute utilisation abusive des services et des ressources du système.

– **La non répudiation de l'origine:** la technique de non répudiation à l'origine vise à éliminer le risque qu'un émetteur puisse nier avoir envoyé un message alors que réellement tel a été le cas. A titre d'exemple, lors d'une transaction commerciale électronique, le service de la non répudiation de l'origine oblige le client à ne pas démentir le fait qu'il a adressé une requête d'achat au fournisseur.

– **La non répudiation de la délivrance:** le destinataire ne pourra jamais nier l'arrivée d'un message qu'il a réellement reçu. A titre d'exemple, lors d'une transaction commerciale électronique, le service du non répudiation de la délivrance oblige le fournisseur à ne pas démentir le fait que le client lui a adressé une requête d'achat.

– **Le contrôle d'accès:** en matière de sécurité, le contrôle d'accès est indispensable. Paradoxalement, son étude n'a pas suscité l'intérêt d'une large communauté de chercheurs. Lorsqu'un sujet souhaite effectuer une opération sur un objet, le système transforme l'opération en une requête qu'il fait passer au moniteur de référence. Ce dernier est responsable du contrôle d'accès aux ressources. Si le sujet est autorisé à accéder à l'objet et à réaliser le type d'opération demandé alors l'accès à l'objet est accordé et l'opération peut se dérouler sans aucune entrave. Dans le cas contraire, il retournera un refus catégorique à l'utilisateur. Les buts du contrôle d'accès rejoignent ceux de la disponibilité.

– **Le secret du flux:** l'intérêt est d'empêcher tout utilisateur non autorisé d'avoir la possibilité d'analyser les flux des données à travers le réseau. Tout accès illégal, même en lecture, à un flux de données permet à l'utilisateur de déduire des informations utiles et qui peuvent, ultérieurement, servir ses intentions malveillantes. La taille des messages échangés, leurs sources et leurs destinations, ainsi que la fréquence des communications entre les utilisateurs sont des exemples de données à préserver pour prévenir le secret des flux dans le réseau et le rendre plus sûr.

– **Le confinement:** ce principe est complémentaire à la confidentialité s'inscrivant dans le même but du bon usage des informations. Le confinement garantit qu'un sujet n'arrive pas à divulguer volontairement le contenu des objets auxquels il a accès à quelqu'un qui n'a pas le droit d'y accéder.

Il convient de noter, pour clore cette section, qu'il y a certains principes de sécurité, parmi ces neuf, qui se chevauchent. Ils peuvent, aussi, être mutuellement exclusifs, à titre d'exemple une confidentialité trop forte entraîne une perte de disponibilité.

1.3 Domaines de la sécurité

Toutes les sphères d'activité de l'informatique, tous les membres d'une entreprise, sont concernés par la sécurité des systèmes d'information. En fonction de leur domaine d'application, les moyens de la sécurité se classifient en [18] [26] :

- Sécurité physique
- Sécurité de l'exploitation
- Sécurité logique
- Sécurité applicative
- Sécurité des télécommunications

1.3.1 Sécurité physique

La sécurité physique concerne tous les aspects liés à la maitrise des systèmes (matériel, composants, câble, etc…) et de l'environnement dans lesquels ils se situent (locaux, alimentation énergétique, climatisation, etc). Sans vouloir être exhaustif, nous retiendrons que la sécurité physique repose essentiellement sur [18]:

- Des normes de sécurité ;
- La protection des sources énergétiques, de l'environnement et des accès (protection physique des équipements, locaux de répartition, tableaux de connexion, etc) ;
- La traçabilité à des entrées des locaux (dissuasion) ;
- Une gestion rigoureuse des clés d'accès aux locaux ;
- La sureté de fonctionnement et la fiabilité des matériels ;
- La redondance physique ;
- Un marquage des matériels (protection par effet dissuasif) ;
- Un plan de maintenance préventive (test, etc) et corrective (pièces de rechange, etc).

Ce dernier point fait également partie de la sécurité d'exploitation.

1.3.2 Sécurité de l'exploitation

On entend par sécurité de l'exploitation tout ce qui touche au bon fonctionnement des systèmes. Cela comprend la mise en place d'un ensemble d'outils et de procédures relatifs aux méthodologies de diagnostic, et de tests pour une maintenance préventive régulière et de réparation, voire de remplacement des matériels défectueux. La sécurité de l'exploitation s'appuie généralement sur un plan de secours (plan de sauvegarde et de continuité), une

infrastructure redondante, sur des procédures de gestion, de contrôle, de validation, de reprise mais aussi sur des personnes compétentes.

Les points clés de la sécurité de l'exploitation sont les suivants [26] :

- Plan de sauvegarde ;
- Plan de continuité ;
- Plan de tests ;
- Inventaires réguliers, pérennants, dynamiques (temps réel) ;
- Gestion du parc informatique ;
- Gestion des configurations, des mises à jour ;
- Gestion des incidents et suivis jusqu'à leur résolution ;
- Automatisation, contrôle et suivi de l'exploitation ;
- Analyse des fichiers de journalisation et de compatibilité ;
- Gestion des contrats de maintenance ;
- Séparation des environnements de développement, d'industrialisation et de production des applicatifs.

Le domaine de la sécurité de l'exploitation peut dans une certaine mesure rejoindre celui des télécommunications, si l'on considère que c'est au niveau des procédures d'exploitation que l'on fixe les paramètres servant à la facturation de l'utilisation des ressources informatique ou de télécommunication. Toutefois, cela touche plus spécifiquement des problèmes de compatibilité et de maîtrise du risque financier. C'est au niveau d'exploitation des ressources que l'on vérifie l'adéquation du niveau de service offert par rapport au contrat de service et à sa facturation.

1.3.3 Sécurité logique

Par opposition, à ce qui précède, la sécurité logique fait référence à la gestion du contrôle d'accès logique, laquelle repose sur un triple service d'identification, d'authentification et d'autorisation (figure 1.1).

La sécurité logique comprend également toute les mesures de prévention de l'infection des données et des programmes par contamination de virus. La mise en place de logiciels antivirus, une gestion efficace des mots de passe et des procédures d'authentification, la maîtrise de l'emploi de micro-ordinateurs portables, des réseaux, la sauvegarde des informations sensibles sur des disques durs extractibles et sauvegardés dans des lieux sécurisés contribueront à la réalisation de la sécurité logique.

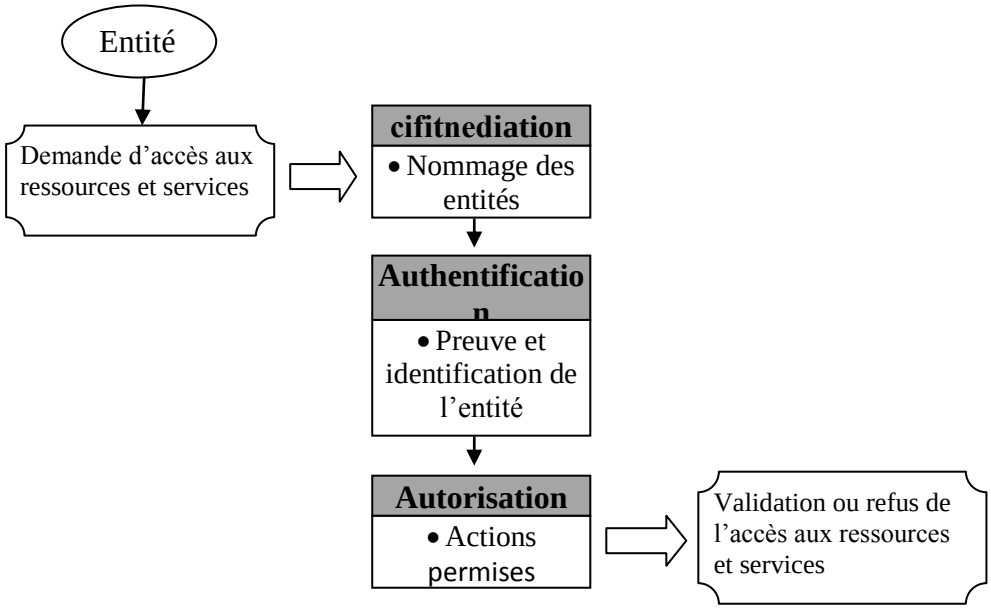


Figure 1.1. Trois points clés du contrôle d'accès [18].

Pour protéger l'information, il faut par exemple comprendre son importance stratégique, la dépendance de la prise de décision à son égard, etc. Afin de déterminer le niveau de protection nécessaire des informations manipulées, on établit au préalable une classification des données. Celles-ci qualifie leur degré de sensibilité (normale, confidentielle, etc). Ainsi à partir d'un tableau mettant en relation le type de données et leur degré de sensibilité, on peut identifier la nature et le nombre de verrous logiques à y affecter. La démarche évoquée ici, s'intègre dans la phase de détermination d'une stratégie sécuritaire.

La sécurité logique vient d'être rappelée sous l'angle de la gestion des accès aux ressources ; il faut y ajouter la dimension liée au développement et aux cycles de vie des applications et introduire la notion de sécurité applicative [18].

1.3.4 Sécurité applicative

La sécurité applicative repose sur l'ensemble des facteurs suivants [18]:

- Une méthodologie de développement ;
- La robustesse des applications (intégration de la sécurité, d'outils d'administration, de contrôle de qualité, de traçabilité, etc) ;
- Des contrôles programmés ;

- Des jeux de tests ;
- Des procédures de recettes ;
- La sécurité des progiciels (choix des fournisseurs, interface sécurité, etc) ;
- L'élaboration et la gestion des contrats, les relations avec des sous traitants éventuels (clauses d'engagement de responsabilité) ;
- Un plan de migration des applications critiques ;
- Un plan d'assurance sécurité ;
- La validation et l'audit des programmes ;
- La qualité et la pertinence des données.

Pour illustrer l'intérêt de la sécurité applicative et insister sur l'importance de la conception et du développement d'application, voici un exemple de problème appartenant au domaine de la sécurité applicative.

Historiquement, pour des raisons de gain de place mémoire, la date ne figurait que sur deux chiffres : seule la partie année est codée. Constructeurs informatiques et éditeurs de logiciels se sont conformés à cette représentation, comme d'ailleurs les développeurs d'application. Actuellement, même les compilateurs suivent cette règle. Tous les matériels, logiciels système, progiciels, applications, toutes entreprises, quel que soit leur domaine d'activité, sont concernés par ce point liés au développement de logiciel et à l'exploitation. Le problème évoqué est trivial mais pour générer des risques sécuritaires très graves pouvant conduire à l'arrêt total des activités d'une entreprise.

Il convient donc de pouvoir mémoriser la date dans son intégralité et d'étendre le format de stockage de celle-ci, ou encore de trouver un nouveau format de donnée et il faudrait pouvoir le faire même pour les archives. En revanche changer de format ne résout pas du tout le problème lié à l'exploitation des données, c'est-à-dire aux traitements portant sur la date. Cela concerne alors la structure des logiciels d'application.

1.3.5 Sécurité des télécommunications

Les architectures de communication doivent être évolutives, réactives, performantes et sécurisées afin de ne pas constituer le maillon faible du système d'information. La mise en commun et le partage de ressources matérielles et logicielles, l'échange d'informations supporté par divers acteurs liés à leur transport et à leur traitement, doit s'intégrer dans une politique stratégique cohérente de conception, d'évolution, et de gestion des

télécommunications. En aucun cas cela ne doit introduire une quelconque vulnérabilité dans le système d'information.

La sécurité des communications [18] consiste à offrir à l'utilisateur final, c'est-à-dire aux applications communicantes, une connectivité fiable, de qualité, de bout en bout. Pour cela, il faut être capable de mettre en œuvre un canal de communication sûr entre les correspondants, e soient le nombre et la nature des éléments intermédiaires (système ou réseau) nécessaires à l'acheminement des données. Ceci implique la réalisation d'une infrastructure réseau sécurisée au niveau des accès, des protocoles de communication, des systèmes d'exploitation, et des équipements.

Par ailleurs, il est également impératif de sécuriser l'infrastructure applicative dans laquelle s'exécutent les applications sur les systèmes d'extrémité au niveau de l'environnement de travail de l'utilisateur et des applications (procédures d'estampillage du temps, de confirmation de l'origine, de non répudiation, de confidentialité). De plus, la sécurité applicative doit permettre d'assurer le respect de la sphère privée de l'utilisateur. La sécurité des télécommunications est peu différente de celle que l'on doit mettre en œuvre pour protéger les ordinateurs. Bien que vulnérable, les réseaux ne le sont pas plus que les systèmes d'extrémité.

Se doter d'un environnement de communication sécurisé passe par la sécurisation de tous les éléments de la chaine informatique. Aussi, la sécurité de télécommunications ne peut s'envisager sans une analyse de risque, spécifique à chaque entreprise, en fonction de son infrastructure environnementale, humaine, organisationnelle et informatique.

1.4 Les risques informatiques

Avant de traiter ce qu'est un risque informatique, il convient de présenter quelques définitions [20].

1.4.1 Définitions

L'actif

C'est la partie d'un bien qui compose le patrimoine et présentant de la valeur pour l'entreprise. Il peut représenter : les équipements, les matériels, les logiciels, les brevets, les processus et activités métier…

Les vulnérabilités

La vulnérabilité est une faille dans les actifs, les contrôles de sécurité technique ou les procédures d'exploitation ou d'administration utilisées dans l'entreprise. Elle consiste, en

général, en une faiblesse dans la protection du système, sous la forme d'une menace qui peut être exploitée pour intervenir sur l'ensemble du système ou d'un intrus qui s'attaque aux actifs.

Les menaces

Une menace est quelqu'un ou quelque chose qui peut exploiter une vulnérabilité pour obtenir, modifier ou empêcher l'accès à un actif ou encore le compromettre. Elle existe en corrélation avec des vulnérabilités. Il peut y avoir aussi plusieurs menaces pour chaque vulnérabilité. La connaissance des différents types de menaces peut aider dans la détermination de leur dangerosité et des contrôles adaptés permettant de réduire leur impact potentiel.

La menace est une source effective d'incidents pouvant entraîner des effets indésirables et graves sur un actif ou un ensemble d'actifs, l'entreprise par elle-même.

Les menaces peuvent être classées par :

- Origine ou sources,
- Type,
- Motivation, action

Elles peuvent être

-délibérées (vol, fraude, virus, hacking, incendie, attentat, sabotage, interception, divulgation ou altération de données…),

-naturelles ou environnementales (tremblement de terre, éruption volcanique, inondation, coupure de courant, incendie…),

-accidentelles (erreurs d'utilisation, omissions…),

-dues à des pannes techniques : mauvais fonctionnement d'un équipement, d'un logiciel.

Le risque

Le risque est la possibilité qu'une chose critique apparaisse. Son évaluation permet d'établir des actions pour réduire et maintenir la menace à un niveau raisonnable et acceptable. Les risques peuvent être qualifiés selon leurs origines (externes ou internes).

➢ Les risques externes :

- Les attaques non ciblées. Toute entreprise est concernée par l'agression de virus ou d'attaques globales sur le réseau (déni de service).
- Les attaques ciblées. Les risques physiques (vol ou destruction de matériel) ou logique (accès d'intrus).

- Les risques internes :
  - Ils sont plus difficiles à appréhender car ils concernent des ressources internes à l'entreprise.

Il existe des facteurs aggravants de risque liés au métier de l'entreprise :

-Les postes nomades : ordinateurs portables, assistants numériques de poche, téléphones évolués portables.

-Des infrastructures, services et applications mal protégés.

-Un plan de sauvegarde ou secours informatique inexistant ou non opérationnel.

L'impact du risque

Il peut être exprimé par les conséquences ou les préjudices affectant un actif : atteinte à l'intégralité, perte de disponibilité, atteinte à l'image de marque, perte de chiffre d'affaires.

Les impacts peuvent être évalués selon les critères suivants :

-financier (frais de remise en état ou de restauration, pertes d'exploitation…),

-juridique et légal,

-réputation et image de l'entreprise (par rapport à l'extérieur et au personnel),

-expertise et savoir faire reconnus de l'entreprise.

1.4.2 Gestion du risque informatique

Un risque est généralement caractérisé par [20] :

-sa source ou son origine : employé malveillant, intrus, programmes illégaux…

-une menace : divulgation d'information confidentielle, coupure de l'électricité…

-la probabilité d'occurrence ou potentialité : durée et lieu, probabilité d'occurrence,

-la vulnérabilité ayant permis ce risque : erreur de conception, erreur humaine, manque de suivi des évènements suspects…

-l'impact, conséquence ou préjudice : indisponibilité du service, perte de marché ou d'image pour l'entreprise…

-les mesures de sécurité ou protections ou contre-mesures pour s'en protéger,

-les contrôles d'accès, la politique de sécurité, la sensibilisation du personnel…

La démarche idéale consiste à en :

-choisir la méthode d'évaluation adaptée au contexte de l'entreprise,

-définir les critères d'identification des risques.

Pour avoir une idée plus précise sur la méthode à utiliser, il est utile de se référer aux méthodologies les plus connues d'analyse et d'évaluation de risques [20]:

-MEHARI (MEthodologie Harmonisée d'Analyse de Risques)

-EBIOS (Expression des Besoins et Identification des Objectifs de Sécurité)
-OCTAVE (Operationnally Critical Threat, Asset, and Vulnerability Evaluation)

1.4.3 Méthode d'analyse des risques

Elle est définie par les actions suivantes :

- Identification des actifs (une première bonne pratique est de connaître la liste des matériels et logiciels) utilisés par les services informatiques.
- Identification des menaces et leurs impacts possibles sur la confidentialité, la disponibilité et l'intégrité de ces actifs.
- Pour chacune des menaces possibles, identification des vulnérabilités.
- Détermination de la probabilité d'occurrence des menaces et des niveaux de vulnérabilité qui pourraient impacter les services.

1.4.4 L'évaluation du risque

Le processus d'évaluation consiste à effectuer l'analyse et l'interprétation du risque et comprends trois activités de base :

- Détermination de la méthodologie et le périmètre d'évaluation.
- Collecte et analyse des données.
- Interprétation des résultats d'analyse de risques.

A partir de ces éléments, les personnels responsables de l'infrastructure informés du contexte établissent les protections à mettre en place.

Etablissement de la valeur des actifs

Cette démarche comprend l'inventaire des actifs concernés susceptibles d'être affectés par les menaces [20] (les matériels, les logiciels et support associés, les licences, …).

- Les matériels : serveurs, stations de travail, ordinateurs personnels, imprimantes, unités de stockage, lignes de communication, routeurs, pare-feu, passerelles de réseau…
- Les logiciels : systèmes d'exploitation, programmes applicatifs et de diagnostic, les logiciels de pare-feu, les outils logiciels…
- Les données : stockées en ligne ou archivées, les sauvegardes, les bases de données, les journaux d'erreurs et les rapports de supervision, en transit sur les supports de communication, toutes celles qui sont en cours de traitement à tout instant dans l'entreprise.
- Les programmes et procédures.
- Les supports d'installation de logiciels, les supports magnétiques…

Basé sur cette liste, cet inventaire va enregistrer un minimum d éléments pour chaque actif. Cela peut être :

- Le propriétaire de l'information,
- Son emplacement physique ou logique,
- Un numéro d'identification.
- L'estimation du coût d'un actif consiste en sa valeur intrinsèque, les impacts à court terme et les conséquences à long terme de sa vulnérabilité.

Identification des menaces

Une menace est une entité ou un évènement qui perturbe le système d'information. Elle inclut les erreurs volontaires ou involontaires, les fraudes, les actions possibles des employés mécontents, les incendies et autres causes naturelles, les hackers, les programmes néfastes ou virus.

Une fois l'inventaire des actifs requérant une protection effectué, il est nécessaire d'identifier les menaces en lien avec chacun d'eux et les risques de perte (d'information, financière…). Ce qui permettra l'élaboration de protections adaptés.

Identification et évaluation des impacts

Après avoir identifier les actifs et les menaces, l'impact de l'attaque est évalué et des mesures appropriées de protection de sécurit é doivent être mises en place, sous les formes suivantes :

- Identification des vulnérabilités du système, des protections mises en place,
- Analyse des probabilités d'occurrence de menaces qui pourraient exploiter ces vulnérabilités,
- Evaluation des conséquences possibles pour chaque menace,
- Sélection des outils ou mécanismes de sécurité adéquats.

Cette évaluation estime le degré de perte pour les actifs. Plus les conséquences d'une menace sont graves, plus le risque pour le système d'information ou l'entreprise est important.

Les impacts peuvent être estimés selon le contexte et les pertes possibles. Les conséquences peuvent être directes (financières) ou indirectes (perte de confiance).

Analyse des vulnérabilités

Une vulnérabilité est une faiblesse de procédures de sécurité, de contrôles techniques ou physiques. Il peut s'agir de même absence de protections qui peuvent être exploités par une menace.

Les interrelations entre vulnérabilités, menaces et actifs sont déterminantes pour l'analyse de risques.

Analyse des protections

Une protection consiste en toute action, procédure, outil qui limite la vulnérabilité d'un système par rapport à une menace.

Evaluation de probabilité de survenance

Cette probabilité, appelée aussi probabilité d'occurrence, est une estimation de la possibilité d'apparition d'une menace ou d'un évènement grave pouvant mettre en péril un système d'information. Dans l'estimation, outre les possibilités d'incendies ou d'inondation, l'historique du système et l'expérience des personnels peuvent donner une indication valable.

Les résultats des analyses de risque et mise en place des protections

L'interprétation des résultats des analyses de risques peut amener à sécuriser en priorité certains éléments du système d'information jugés critiques. Naturellement la suppression des vulnérabilités de l'ensemble du système d'information est toujours l'objectif essentiel.

1.5 Les attaques sur la sécurité des réseaux informatiques

Une attaque peut être définie comme toute action ou ensemble malveillant d'actions qui peut porter atteinte à la sécurité des informations d'un système ou réseau informatique ou bien les services qu'il fournit [1] [22]. Dans ce qui suit, nous allons dans un premier temps analyser ce que nous appellerons « l'anatomie d'une attaque », puis dans un second temps, nous caractériserons ces attaques et observerons leur déroulement.

1.5.1 Anatomie d'une attaque

Fréquemment appelés « les 5 P » dans la littérature, ces cinq verbes anglophones constituent le squelette de toute attaque informatique : Probe, Penetrate, Persist, Propagate, Paralyze.

Observons le détail de chacune de ces étapes [2][35] :

- **Probe** : consiste en la collecte d'informations par le biais d'outils comme whois, Arin, DNS lookup. La collecte d'informations sur le système cible peut s'effectuer de plusieurs manières, comme par exemple un scan de ports grâce au programme Nmap pour déterminer la version des logiciels utilisés, ou encore un scan de vulnérabilités à l'aide du programme

Nessus. Pour les serveurs web, il existe un outil nommé Nikto qui permet de rechercher les failles connues ou les problèmes de sécurité. Des outils comme firewalk, hping ou SNMP Walk permettent quant à eux de découvrir la nature d'un réseau.

- **Penetrate** : utilisation des informations récoltées pour pénétrer un réseau. Des techniques comme le brute force ou les attaques par dictionnaires peuvent être utilisées pour outrepasser les protections par mot de passe. Une autre alternative pour s'infiltrer dans un système est d'utiliser des failles applicatives que nous verrons ci-après.

- **Persist** : création d'un compte avec des droits de super utilisateur pour pouvoir se réinfiltrer ultérieurement. Une autre technique consiste à installer une application de contrôle à distance capable de résister à un reboot (ex : un cheval de Troie).

- **Propagate** : cette étape consiste à observer ce qui est accessible et disponible sur le réseau local.

- **Paralyze** : cette étape peut consister en plusieurs actions. Le pirate peut utiliser le serveur pour mener une attaque sur une autre machine, détruire des données ou encore endommager le système d'exploitation dans le but de planter le serveur.

Après ces cinq étapes, le pirate peut éventuellement tenter d'effacer ses traces, bien que cela ne soit rarement utile. En effet, les administrateurs réseaux sont souvent surchargés de logs à analyser.

1.5.2 Les différents types d'attaques

Les attaques sont regroupées en trois catégories [22][23][28][35]:

1.5.2.1 Les attaques réseaux

Ce type d'attaque se base principalement sur des failles liées aux protocoles ou à leur implémentation. Les RFC (Request For Comment) ne sont parfois pas assez spécifiques, et un choix particulier d'implémentation dans les différents services ou clients peut entraîner un problème de sécurité.

Observons quelques attaques bien connues.

- ***Les techniques de scan***

Les scans de ports ne sont pas des attaques à proprement parler. Le but des scans est de déterminer quels sont les ports ouverts, et donc en déduire les services qui sont exécutés sur la machine cible (ex : port 80/TCP pour un service HTTP). Par conséquent, la plupart des attaques sont précédées par un scan de ports lors de la phase Probe qui est comme nous l'avons vu, la première phase des 5P's dans le déroulement d'une attaque. Il existe un nombre important de techniques de scan. Idéalement, la meilleure technique de scan est celle qui est la plus furtive afin de ne pas alerter les soupçons de la future victime. Voici une description des techniques de scan les plus répandues :

- Le scan simple : aussi appelé le scan connect(), il consiste à établir une connexion TCP complète sur une suite de ports. S'il arrive à se connecter, le port est ouvert ; sinon, il est fermé. Cette méthode de scan est très facilement détectable.
- Le scan furtif : aussi appelé scan SYN, il s'agit d'une amélioration du scan simple. Ce scan essaie également de se connecter sur des ports donnés, mais il n'établit pas complètement la connexion : pas de commande ACK (acquittement) après avoir reçu l'accord de se connecter. Grâce à ceci, la méthode est bien plus furtive que le scan normal.
- Les scans XMAS, NULL et FIN : se basent sur des détails de la RFC du protocole TCP pour déterminer si un port est fermé ou non en fonction de la réaction à certaines requêtes. Ces scans sont moins fiables que le scan SYN mais ils sont un peu plus furtifs. La différence entre ces trois types de scan se situe au niveau des flags TCP utilisés lors de la requête.
- Le scan à l'aveugle : s'effectue via une machine intermédiaire et avec du spoofing. Le système attaqué pense que le scan est réalisé par la machine intermédiaire et non par le pirate.
- Le scan passif : est la méthode la plus furtive. Consiste à analyser les champs d'en-tête des paquets (TTL, ToS, MSS, …) et les comparer avec une base de signatures qui pourra déterminer les applications qui ont envoyé ces paquets. L'utilitaire incontournable pour réaliser des scans de ports se nomme Nmap.

- ***IP Spoofing***

But : usurper l'adresse IP d'une autre machine.

Finalité : se faire passer pour une autre machine en truquant les paquets IP. Cette technique peut être utile dans le cas d'authentifications basées sur une adresse IP (services tels que rlogin ou ssh par exemple).

Déroulement : il existe des utilitaires qui permettent de modifier les paquets IP ou de créer ses propres paquets (ex : hping). Grâce à ces utilitaires, il est possible de spécifier une

adresse IP différente de celle que l'on possède, et ainsi se faire passer pour une autre « machine ».

Cependant, ceci pose un problème : en spécifiant une adresse IP différente de notre machine nous ne recevrons pas les réponses de la machine distante, puisque celle-ci répondra à l'adresse spoofée. Il existe toutefois deux méthodes permettant de récupérer les réponses :

- Source routing : technique consistant à placer le chemin de routage directement dans le paquet IP. Cette technique ne fonctionne plus de nos jours, les routeurs rejetant cette option.
- Reroutage : cette technique consiste à envoyer des paquets RIP aux routeurs afin de modifier les tables de routage. Les paquets avec l'adresse spoofée seront ainsi envoyés aux routeurs contrôlés par le pirate et les réponses pourront être également reçues par celui-ci.

- ***ARP Spoofing (ou ARP Redirect)***

But : rediriger le trafic d'une machine vers une autre.

Finalité : grâce à cette redirection, une personne mal intentionnée ne peut se faire passer pour une autre. De plus, le pirate peut rerouter les paquets qu'il reçoit vers le véritable destinataire, ainsi l'utilisateur usurpé ne se rendra compte de rien. La finalité est la même que l'IP spoofing mais on travaille ici au niveau de la couche liaison de données.

Déroulement : pour effectuer cette usurpation, il faut corrompre le cache ARP de la victime. Ce qui signifie qu'il faut lui envoyer des trames ARP en lui indiquant que l'adresse IP d'une autre machine est la sienne. Les caches ARP étant régulièrement vidés, il faudra veiller à maintenir l'usurpation.

- ***DNS Spoofing***

But : fournir de fausses réponses aux requêtes DNS, c'est-à-dire indiquer une fausse adresse IP pour un nom de domaine.

Finalité : rediriger, à leur insu, des Internautes vers des sites pirates. Grâce à cette fausse redirection, l'utilisateur peut envoyer ses identifiants en toute confiance par exemple.

Déroulement : il existe deux techniques pour effectuer cette attaque.

- DNS Cache Poisoning : les serveurs DNS possèdent un cache permettant de garder pendant un certain temps la correspondance entre un nom de machine et son adresse IP. Le DNS Cache Poisoning consiste à corrompre ce cache avec de fausses informations. Ces fausses informations sont envoyées lors d'une réponse d'un serveur DNS contrôlé par le pirate à un autre serveur DNS, lors de la demande de l'adresse IP d'un domaine (ex : www.ledomaine.com). Le cache du serveur ayant demandé les informations est alors corrompu.

- DNS ID Spoofing : pour communiquer avec une machine, il faut son dresse IP. On peut toutefois avoir son nom, et grâce au protocole DNS, nous pouvons obtenir son adresse IP. Lors d'une requête pour obtenir l'adresse IP à partir d'un nom, un numéro d'identification est placé dans la trame afin que le client et le serveur puissent identifier la requête. L'attaque consiste ici à récupérer ce numéro d'identification (en sniffant le réseau) lors de la communication entre un client et un serveur DNS, puis, envoyer des réponses falsifiées au client avant que le serveur DNS lui réponde.

Remarque : Une attaque que nous allons voir ci-après, le Déni de Service, peut aider à ralentir le trafic du serveur DNS et ainsi permettre de répondre avant lui.

- ***Fragments attacks***

But : le but de cette attaque est de passer outre les protections des équipements de filtrage IP.

Finalité : en passant outre les protections, un pirate peut par exemple s'infiltrer dans un réseau pour effectuer des attaques ou récupérer des informations confidentielles.

Déroulement : deux types d'attaque sur les fragments IP peuvent être distingués.

Fragments overlapping : quand un message est émis sur un réseau, il est fragmenté en plusieurs paquets IP. Afin de pouvoir reconstruire le message, chaque paquet possède un offset. Le but de l'attaque est de réaliser une demande de connexion et de faire chevaucher des paquets en spécifiant des offsets incorrects. La plupart des filtres analysant les paquets indépendamment, ils ne détectent pas l'attaque. Cependant, lors de la défragmentation, la demande de connexion est bien valide et l'attaque a lieu.

Tiny fragments : le but de l'attaque est de fragmenter une demande de connexion sur deux paquets IP : le premier paquet de taille minimum (68 octets selon la RFC du protocole IP) ne contient que l'adresse et le port de destination. Le deuxième paquet contient la demande effective de connexion TCP. Le premier paquet est accepté par les filtres puisqu'il ne contient rien de suspect. Quand le deuxième paquet arrive, certains filtres ne le vérifient pas pensant que si le premier paquet est inoffensif, le deuxième l'est aussi. Mais lors de la défragmentation sur le système d'exploitation, la connexion s'établit !

De nos jours, une grande majorité des firewalls sont capables de détecter et stopper ce type d'attaques.

- ***TCP Session Hijacking***

But : le but de cette attaque est de rediriger un flux TCP afin de pouvoir outrepasser une protection par mot de passe.

Finalité : le contrôle d'authentification s'effectuant uniquement à l'ouverture de la session, un pirate réussissant cette attaque parvient à prendre possession de la connexion pendant toute la durée de la session.

Déroulement : dans un premier temps, le pirate doit écouter le réseau, puis lorsqu'il estime que l'authentification a pu se produire (délai de n secondes par exemple), il désynchronise la session entre l'utilisateur et le serveur. Pour ce faire, il construit un paquet avec, comme adresse IP source, celle de la machine de l'utilisateur et le numéro d'acquittement TCP attendu par le serveur. En plus de désynchroniser la connexion TCP, ce paquet permet au pirate d'injecter une commande via la session préalablement établie.

1.5.2.2 Les attaques applicatives

Les attaques applicatives se basent sur des failles dans les programmes utilisés, ou encore des erreurs de configuration. Toutefois, comme précédemment, il est possible de classifier ces attaques selon leur provenance.

- **Les problèmes de configuration**

Il est très rare que les administrateurs réseaux configurent correctement un programme. En général, ils se contentent d'utiliser les configurations par défaut. Celles-ci sont souvent non sécurisées afin de faciliter l'exploitation du logiciel.

De plus, des erreurs peuvent apparaître lors de la configuration d'un logiciel. Une mauvaise configuration d'un serveur peut entraîner l'accès à des fichiers importants, ou mettant en jeu l'intégrité du système d'exploitation. C'est pourquoi il est important de bien lire les documentations fournies par les développeurs afin de ne pas créer de failles.

- **Les bugs**

Liés à un problème dans le code source, ils peuvent amener à l'exploitation de failles. Il n'est pas rare de voir l'exploitation d'une machine suite à une simple erreur de programmation. On ne peut toutefois rien faire contre ce type de problèmes, si ce n'est attendre un correctif de la part du développeur.

- **Les buffers overflows**

Les buffers overflows, ou dépassement de la pile, sont une catégorie de bug particulière. Issus d'une erreur de programmation, ils permettent l'exploitation d'un shellcode à distance. Ce shellcode permettra à une personne mal intentionnée d'exécuter des commandes sur le système distant, pouvant aller jusqu'à sa destruction. L'erreur de programmation est souvent la même : la taille d'une entrée n'est pas vérifiée et l'entée est

directement copiée dans un buffer dont la taille est inférieure à la taille de l'entrée. On se retrouve donc en situation de débordement, et l'exploitant peut ainsi accéder à la mémoire.

- **Les scripts**

Principalement web (ex : Perl, PHP, ASP), ils s'exécutent sur un serveur et renvoie un résultat au client. Cependant, lorsqu'ils sont dynamiques (i.e. qu'ils utilisent des entrées saisies par un utilisateur), des failles peuvent apparaître si les entrées ne sont pas correctement contrôlées.

L'exemple classique est l'exploitation de fichier à distance, tel que l'affichage du fichier mot de passe du système en remontant l'arborescence depuis le répertoire web.

- **Les injections SQL**

Tout comme les attaques de scripts, les injections SQL profitent de paramètres d'entrée non vérifiés. Comme leur nom l'indique, le but des injections SQL est d'injecter du code SQL dans une requête de base de données. Ainsi, il est possible de récupérer des informations se trouvant dans la base (exemple : des mots de passe) ou encore de détruire des données.

- **Man in the middle**

Moins connue, mais tout aussi efficace, cette attaque permet de détourner le trafic entre deux stations. Imaginons un client C communiquant avec un serveur S. Un pirate peut détourner le trafic du client en faisant passer les requêtes de C vers S par sa machine P, puis transmettre les requêtes de P vers S. Et inversement pour les réponses de S vers C.

Totalement transparente pour le client, la machine P joue le rôle de proxy. Il accédera ainsi à toutes les communications et pourra en obtenir les informations sans que l'utilisateur s'en rende compte.

1.5.2.3 Le Déni de service

Evoqué précédemment, le déni de service est une attaque visant à rendre indisponible un service. Ceci peut s'effectuer de plusieurs manières : par le biais d'une surcharge réseau, rendant ainsi la machine totalement injoignable ; ou bien de manière applicative en crashant l'application à distance. L'utilisation d'un buffer overflow peut permettre de planter l'application à distance. Grâce à quelques instructions malicieuses et suite à une erreur de programmation, une personne mal intentionnée peut rendre indisponible un service (serveur web, serveur de messagerie, … etc) voire un système complet.

Voici quelques attaques réseaux connues permettant de rendre indisponible un service :

- **SYN Flooding** : exploite la connexion en 3 phases de TCP (Three Way Handshake : SYN / SYN-ACK / ACK). Le principe est de laisser un grand nombre de connexions TCP en attente. Le pirate envoie de nombreuses demandes de connexion (SYN), reçoit les SYN-ACK mais ne répond jamais avec ACK. Les connexions en cours occupent des ressources mémoire, ce qui va entraîner une saturation et l'effondrement du système.
- **UDP Flooding** : le trafic UDP est prioritaire sur TCP. Le but est donc d'envoyer un grand nombre de paquets UDP, ce qui va occuper toute la bande passante et ainsi rendre indisponible toutes les connexions TCP.
- **Packet Fragment** : utilise une mauvaise gestion de la défragmentation au niveau ICMP. Exemple : ping of death. La quantité de données est supérieure à la taille maximum d'un paquet IP.
- **Smurfling** : le pirate fait des requêtes ICMP ECHO à des adresses de broadcast en spoofant l'adresse source (en indiquant l'adresse de la machine cible). Cette machine cible va recevoir un nombre énorme de réponses, car toutes les machines vont lui répondre, et ainsi utiliser toute sa bande passante.
- **Déni de service distribué** : le but est ici de reproduire une attaque normale à grande échelle. Pour ce faire, le pirate va tenter de se rendre maître d'un nombre important de machines. Grâce à des failles (buffer overflows, failles RPC (Remote Procedure Call), … etc) il va pouvoir prendre le contrôle de machines à distance et ainsi pouvoir les commander à sa guise. Une fois ceci effectué, il ne reste plus qu'à donner l'ordre d'attaquer à toutes les machines en même temps, de manière à ce que l'attaque soit reproduite à des milliers d'exemplaires. Ainsi, une simple attaque comme un SYN Flooding pourra rendre une machine ou un réseau totalement inaccessible [35].

1.6 Politique de sécurité

Les premiers à employer ce terme "politique de sécurité" étaient les militaires. En 1973, le département de la défense américain (DoD), par la directive 5200.28 et le manuel DoD 5200.2866-M, établit une politique de sécurité uniforme désignant un ensemble de règles de sécurité à respecter par tous appelée politique multi-niveaux [20] [21].

1.6.1 Définition

"La politique de sécurité d'un système est l'ensemble des lois, règles et pratiques qui régissent la façon dont l'information sensible et les autres ressources sont gérées, protégées et distribuées à l'intérieur d'un système spécifique" [3][37].

La politique de sécurité décrit les objectifs de sécurité du système et identifie les menaces auxquelles le système devra faire face. Les politiques de sécurité définissent les autorités et les ressources, spécifient les droits et les règles d'usage (voir la figure 1.2).
Donc, le terme "politique de sécurité" est associé à l'ensemble des propriétés de sécurité que l'on désire appliquer et déployer dans un système ainsi que les dispositifs pour les assurer.

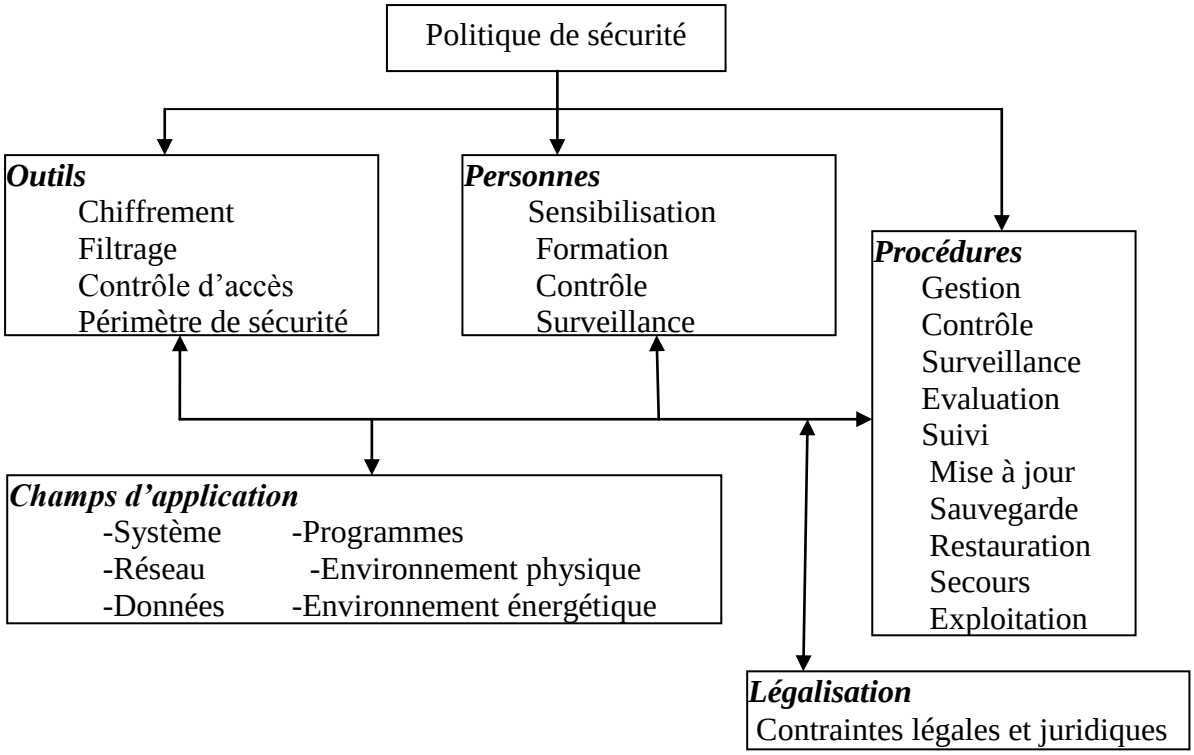


Figure 1.2. Stratégie et politique de sécurité [20]

1.6.2 Les trois politiques de sécurité

Une politique de sécurité est composée de trois sous-politiques : la politique de sécurité administrative qui contrôle l'administration du système et les personnes qui l'utilisent, la politique de sécurité physique qui contrôle le droit d'accès physique aux ressources, et la politique de sécurité logique qui contrôle les droits d'accès aux données [3][19][21].

– politique de sécurité administrative

Cette politique gère administrativement le système et les gens qui l'utilisent. Elle précise les responsabilités de chacun, notamment le nom du responsable de la sécurité et le

nom du responsable des ressources. Elle détermine les personnes de confiance et leurs accorde et confie l'exécution de certaines opérations.

Enfin, cette politique réglemente la réalisation des tâches ordinaires telles que la manipulation des entrées-sorties (imprimantes, sauvegardes, etc.), l'installation et maintenance des logiciels, l'installation des nouvelles machines sur le réseau.

– ***politique de sécurité physique :*** ces politiques veillent à garder intact le matériel qui constitue le réseau. En particulier, elles définissent les mesures contre le cambriolage, les incendies, les catastrophes naturelles, et les coupures d'électricité, etc.

– ***politique de sécurité logique :*** la politique de sécurité logique se devise en deux sous-politiques distinctes: la politique d'authentification et la politique d'autorisation.

– La politique d'authentification réglemente les méthodes, les techniques et les mécanismes d'authentification des utilisateurs (et programmes) auprès du système et la vérification de cette authentification.

– La politique d'autorisation fixe les droits que les sujets ont sur les objets et spécifie la manière dont ces droits peuvent être modifiés. On distingue deux types de politiques d'autorisation : les politiques discrétionnaires et les politiques mandataires.

– Les politiques discrétionnaires stipulent que les droits d'accès peuvent être manipulés librement par les sujets eux mêmes. L'abus de ce principe peut amener le système dans un état d'insécurité. D'ailleurs, les systèmes dont le contrôle d'accès est basé sur cette politique, cette large confiance entre les sujets sont particulièrement vulnérables aux attaques comme le cheval de Troie. Paradoxalement, les politiques discrétionnaires sont très répandues dans les systèmes d'utilisation civile telle qu'Unix et Windows NT.

– Les politiques mandataires sont inspirées du politique multi-niveau appliqué par les militaires. Elles spécifient des règles incontournables pour régenter les interactions entre sujets et objets. Ces règles fixent les droits d'accès qu'un sujet particulier peut posséder sur n'importe quel objet en imposant une hiérarchie pour les sujets et pour les objets. La confiance dans ce cadre est remplacée par le contrôle.

Dans une telle politique, les informations sont classées selon leur sensibilité et les utilisateurs sont habilités à accéder à l'information jusqu'à un certain niveau de classification de sécurité.

1.6.3 Architecture réseau supportant une politique de sécurité

Une architecture réseau supportant une politique de sécurité est une architecture sécurisée [38]. La finalité de cette architecture est de déterminer les emplacements adéquats pour la mise en place des mécanismes de sécurité requis par la politique.

Envisager de construire une nouvelle architecture plus adaptée aux exigences de sécurité définies par la politique ou garder les architectures actuelles, telle est la question. La première alternative est la plus appropriée, toutefois, il est plus judicieux d'admettre que les architectures existantes [39][40] sont très largement répandues et qu'on ne peut pas arrêter subitement leur utilisation [41][17].

Il est donc préférable d'opter pour la deuxième alternative et de considérer la sécurité comme un complément qui apporte de nouveaux services à ceux déjà offerts sur le réseau. Dans cette étude, on se place dans le cadre de l'architecture OSI et on présentera dans ce qui suit deux visions possibles d'un réseau sécurisé:

– *Interconnexion de systèmes centralisés sécurisés:* adoptée par les normes IEEE 802.10 [42] et ISO 7498-2 [40]. Le réseau est considéré comme un ensemble de systèmes gérés de différentes manières par des autorités différentes.

– *Noyau de sécurité réparti:* contrairement à la vision précédente, le réseau est considéré comme une entité indissociable dirigée par une seule autorité.

1.7 Outils de sécurité

Dans ce qui suit nous allons présenter un ensemble non exhaustif d'outils de sécurité :

1.7.1 Antivirus

Les antivirus [20] sont des logiciels conçus pour identifier, neutraliser et éliminer des logiciels malveillants (dont les virus ne sont qu'un exemple). Ceux-ci peuvent se baser sur l'exploitation de failles de sécurité, mais il peut également s'agir de programmes modifiant ou supprimant des fichiers, que ce soit des documents de l'utilisateur de l'ordinateur infecté, ou des fichiers nécessaires au bon fonctionnement de l'ordinateur.

Un antivirus vérifie les fichiers et courriers électroniques, les secteurs de boot (pour détecter les virus de boot), mais aussi la mémoire vive de l'ordinateur, les médias amovibles (clefs USB, CD, DVD, etc.), les données qui transitent sur les éventuels réseaux (dont internet), etc.

Les méthodes utilisées par les antivirus sont différentes, nous pouvons les résumer dans les point suivants:

- Les principaux antivirus du marché se concentrent sur des fichiers de *signatures* et comparent alors la *signature virale* du virus aux codes à vérifier.
- La *méthode heuristique* est la méthode la plus puissante, tendant à découvrir un code malveillant par son comportement. Elle essaie de le détecter en analysant le code d'un programme inconnu. Parfois de fausses alertes peuvent être provoquées.
- L'*analyse de forme* repose sur du filtrage basé entre des règles regexp ou autres, mises dans un fichier junk. Cette dernière méthode peut être très efficace pour les serveurs de courriels supportant les regexp type postfix puisqu'elle ne repose pas sur un fichier de signatures.

Les antivirus peuvent balayer le contenu d'un disque dur, mais également la mémoire de l'ordinateur. Pour les plus modernes, ils agissent en amont de la machine en scrutant les échanges de fichiers avec l'extérieur, aussi bien en flux montant que descendant. Ainsi, les courriels sont examinés, mais aussi les fichiers copiés sur ou à partir de supports amovibles tels que cédéroms, disquettes, connexions réseau, clefs USB…

1.7.2 Pare-feu (Firewall)

Un ***pare-feu*** (Firewall en anglais), est un système physique (*appliance*) ou logique (*logiciel*) servant d'interface entre un ou plusieurs réseaux afin de contrôler et éventuellement bloquer la circulation des paquets de données. Pour cela, les pare-feux analysent les informations contenues dans les couches 3, 4 et 7 du modèle OSI*I* et contrôlent ainsi les informations en transit [20][21].

Pour une appliance, il s'agit d'une machine physique comportant au minimum deux interfaces réseau : une interface pour le réseau à protéger (réseau interne) et une interface pour le réseau externe (domaine dangereux : réseau extérieur ou Internet).

Dans le cas d'un ordinateur équipé d'un logiciel pare-feu, celui-ci contrôle tous les accès du système à l'interface réseau externe (on parle de Firewall personnel pour un poste client ou basé hôte pour un serveur). On retrouve les mêmes domaines réseaux dont le réseau interne est réduit au système d'exploitation de l'ordinateur.

Il y a en réalité deux principaux types de *Firewall*:

• *Firewall IP Filter* ou *Chokes* : ce type de *Firewall* travail au niveau des paquets transmis sur les réseaux, il permet de contrôler le flux de paquets en fonction de l'origine, de la

destination, des ports et du type d'information contenue dans chacun d'eux. C'est un système relativement facile à mettre en place en instaurant un certains nombre de règles permettant de contrôler les paquets entrants ou sortants du réseau privé. Il s'agit soit d'un ordinateur soit d'un périphérique de communication permettant de restreindre le flux de paquets entre les réseaux.

• *Gates* : il s'agit d'un programme, d'un périphérique ou d'un ordinateur qui reçoit les connections des réseaux externes et les retransmet dans le réseau privé. Aucun utilisateur n'est autorisé à accéder à ce *Gate* pour des raisons de sécurité (seule une connexion sur la console par l'administrateur est autorisée). Sur ce *Gate*, un ou plusieurs des programmes suivants doivent être utilisés:

– *Network Client Software* : les utilisateurs doivent, dans cette configuration, avoir un compte sur le *Gate* pour pouvoir utiliser des applications telles que ftp, telnet, netscape. . . Les utilisateurs doivent donc se connecter sur le *Gate* afin d'accéder à l'extérieur. C'est un système facile à mettre en place mais très peu sécurisé du fait de l'accès d'utilisateurs sur l'élément protégeant la partie privée de la partie publique.

– *Proxy Server* : un *Proxy* est un programme qui transmet les données qu'il reçoit à un autre programme situé sur une autre machine du réseau. Dans le cas d'un *Firewall*, un *Proxy* sert à "forwarder" une requête (Web, FTP,. . .) à travers le *Firewall* du réseau interne au réseau externe.

– *Network Server* : il est également possible de lancer des serveurs (*daemon*) comme par exemple Sendmail sur le *Gate* afin de fournir d'autres services aux utilisateurs.

Par contre, il n'est pas conseillé d'installer un serveur Web comme Apache sur un *Gate* pour des raisons de sécurité.

1.7.3 Cryptographie

La cryptographie [25][19] est l'étude des méthodes permettant de transmettre des données de manière confidentielle. Afin de protéger un message, on lui applique une transformation qui le rend incompréhensible ; c'est ce qu'on appelle le chiffrement, qui, à partir d'un texte en clair, donne un texte chiffré. Inversement, le déchiffrement est l'action qui permet de reconstruire le texte en clair à partir du texte chiffré en utilisant une clé particulière et un algorithme de déchiffrement. On distingue deux catégories de systèmes cryptographiques:

– Des systèmes de cryptographie symétrique: Outre l'authenticité, ces systèmes doivent garantir la sécurité des clés. Le principe de fonctionnement du système d'authentification à clés symétriques est le suivant:

Soit 'M' un ensemble de messages vérifiant une propriété particulière (définie à l'avance par les deux interlocuteurs) et soit 'm', appartenant à 'M', le message à échanger. L'émetteur chiffre 'm' avec la clé secrète 's' et envoie le résultat Es(m) au destinataire, ce dernier déchiffre le message Ds(Es(m)) = m et vérifie si le message déchiffré appartient à 'M', si c'est le cas alors l'authenticité du message est prouvée, sinon le message est rejeté.
Parmi les algorithmes de cryptographie symétrique les plus connus on cite [19] [25]:
– DES (Data Encryption Standard) : Crée en 1977. Cet algorithme consiste à découper le texte en clair en bloc de 64 bits qui sont ensuite cryptés un par un en utilisant une clé de 56 bits.
– IDEA (International Data Encryption Algorithm) : Cet algorithme a été conçu dans les années 90. Il consiste en un chiffrement par blocs avec itération qui utilise une clé de 128 bits et comporte 8 rondes.
– Des systèmes de cryptographie asymétrique: Ces systèmes ont été proposés par Whitfield Diffie et Martin Hellman en 1976 marquant ainsi la naissance de la cryptographie moderne. La cryptographie asymétrique (appelée aussi cryptoraphie à clé publique) utilise deux clés : une clé publique et une clé privée. Les clés publique et privée sont mathématiquement liées par l'algorithme de cryptage de telle manière qu'un message crypté avec une clé publique ne puisse être décrypté qu'avec la clé privée correspondante. Mais Ces clés ont la propriété que si l'on connaît la clé publique, il est impossible de trouver par calcul la clé privée. Dans un système cryptographique à clé publique, chaque partie communicante possède sa propre paire de clés publique/privée. Tout le monde peut connaître la clé publique; par contre, la clé privée demeure secrète.

Voici comment un tel système peut assurer la confidentialité des communications : deux sujets, Alice et Bob, désirent échanger des informations qu'ils tiennent à garder secrètes et intègres. Alice va chiffrer l'information avec la clé publique de Bob. La confidentialité de l'information est maintenue, car seul Bob peut la déchiffrer au moyen de sa clé privée.
L'algorithme de cryptographie asymétrique le plus connu est **RSA (Rivest, Shamir, Adleman)**. Inventé par Rivest, Shamir et Adleman en 1978, RSA repose sur la difficulté de factoriser des grands nombres.

1.7.4 Réseau privé virtuel : VPN

La connexion à distance sur un réseau interne d'une PME impose d'utiliser un réseau privé virtuel d'entreprise ou VPN. Cette fonctionnalité chiffre le trafic réseau sensible et requiert une authentification forte, fournissant un accès à distance sécurisé (voir la figure 1.3). En effet, le principe de sécurité est la caractéristique principale des VPN [20] :

Intégrité : les données reçues par le site principal sont identiques à celles envoyées par le site externe ou le poste nomade,

Confidentialité : la propriété privée des données est complètement assurée,

Authentification : le récepteur des données sur le site de l'entreprise doit être sûr que les données ont bien été émises par le bon utilisateur.

L'échange de données confidentielles entre les personnels nomades et l'entreprise ou entre différentes entités implique la mise en œuvre de liaisons sécurisées, virtuelles (VPN) en liaison avec un parefeu ou physiques (lignes louées spécialisées).

Trois types de solutions de VPN existent associées avec un parefeu :

- intégrée comme service du parefeu,
- systèmes autonomes placés devant le parefeu,
- systèmes autonomes placés derrière le parefeu (solutions logicielles).

L'échange de données entre sites distants de la même entreprise peut aussi être effectué en utilisant une liaison spécialisée ou dédiée utilisant les services d'un opérateur de télécommunication. Cette solution permet de s'affranchir de toutes les menaces liées à l'utilisation du réseau Internet, mais elle représente un coût plus important.

Le serveur VPN c'est la machine qui fait office de point de contrôle dans l'entreprise. Elle doit être située sur le réseau, elle peut servir aussi de parefeu ou de routeur de frontière. Autonome, ce dispositif doit être placé dans la DMZ (zone démilitarisée est un sous-réseau isolé par un pare-feu) dédiée. La technologie VPN utilisée dans l'environnement de DMZ permet d'être protégé par les règles de parefeu et autoriser uniquement le trafic du VPN. L'algorithme de chiffrement employé dans le VPN doit être un algorithme fort et reconnu. Le système d'authentification doit être un système à 2 facteurs : nom d'utilisateur et mot de passe. Une autre solution serait de combiner les cartes à puce avec un numéro d'identification personnel (PIN : *Personal Identification Number*).

Le positionnement du serveur VPN, en général, le meilleur endroit pour l'implémenter est en protection derrière le parefeu. Ce qui va impliquer que le trafic VPN passe sous forme chiffré en sortie à travers le parefeu. Celui-ci est alors incapable d'inspecter le trafic en entrée et en

sortie, d'effectuer des contrôles d'accès, de journalisation ou de vérification de virus. Le trafic VPN est chiffré pour garantir la confidentialité des transferts de données pendant la durée de la connexion. Le principe du VPN consiste à établir un chemin virtuel unique avec l'identification de l'émetteur et du récepteur. Ce chemin est aussi appelé **tunnel**.

De plus, le trafic est chiffré pour garantir la confidentialité des transferts de données pendant la durée de la connexion. Le principe du VPN est aussi conçu pour traiter des protocoles différents. Le VPN établit une connexion sécurisée vers un réseau à distance en utilisant la technique de **tunnelling** à travers Internet. Cette technique consiste en l'encapsulation, la transmission, la désencapsulation d'un paquet de données sous forme de message à l'intérieur d'un paquet IP (*Internet Protocol*). Le transfert de ces données est réalisé à travers Internet.

De plus, l'authentification de la connexion VPN est accomplie au niveau de l'utilisateur externe et des données transmises (intégrité). Il existe trois protocoles pour le **tunneling** à travers Internet : *Internet Protocol Security* (IPSec), *Layer 2 Tunneling Protocol* (L2TP) et *Point to Point Tunneling Protocol* (PPTP). Les réseaux virtuels d'entreprise sont utilisés pour fournir des liens réseaux sécurisés à travers des réseaux qui n'ont pas de relation d'approbation, par exemple via Internet ou pour créer des réseaux sécurisés entre entreprises.

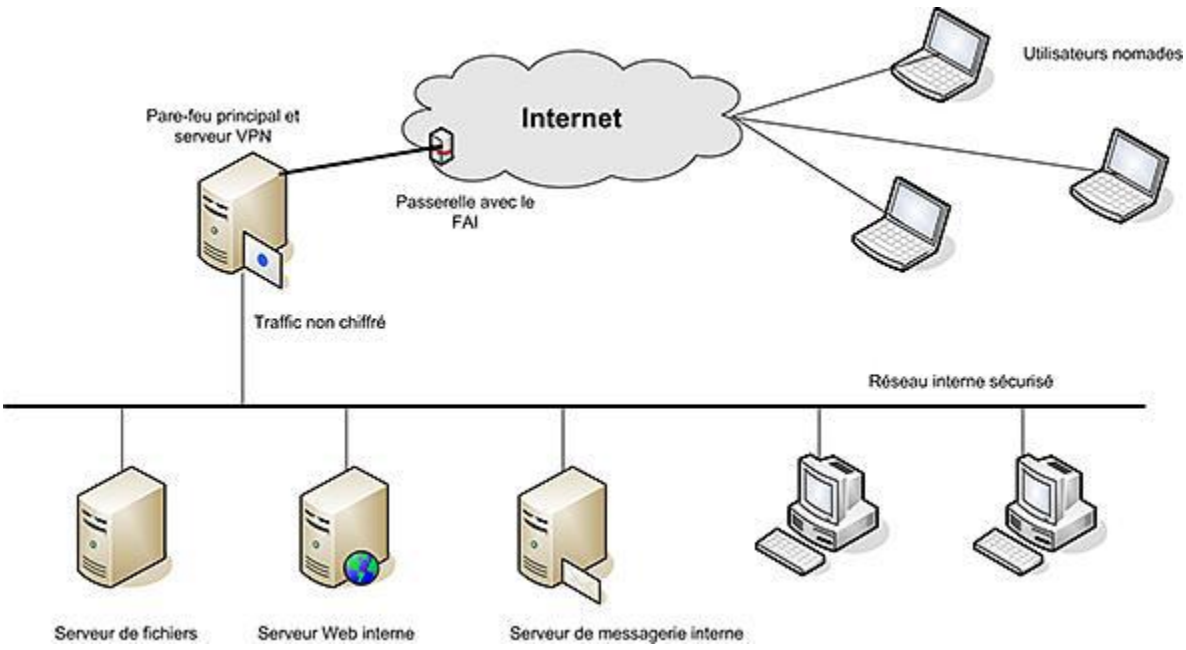


Figure1.3 Exemple d'environnement de VPN [20]

1.7.5 Système de détection d'intrusion : IDS

En sécurité informatique, la détection d'intrusion est l'acte de détecter les actions qui essaient de compromettre la confidentialité, l'intégrité ou la disponibilité d'une ressource. Un système de détection d'intrusion est un mécanisme destiné à repérer des activités anormales ou suspectes sur la cible analysée (réseau, hôte) [3][35].

La détection d'intrusion peut être effectuée manuellement ou automatiquement. Dans le processus de détection d'intrusion manuelle, un analyste humain procède à l'examen de fichiers de logs à la recherche de tout signe suspect pouvant indiquer une intrusion. Un système qui effectue une détection d'intrusion automatisée est appelé système de détection d'intrusion (IDS). Lorsqu'une intrusion est découverte par un IDS, les actions typiques qu'il peut entreprendre sont par exemple d'enregistrer l'information pertinente dans un fichier ou une base de données, de générer une alerte par e-mail ou un message sur un pager ou un téléphone mobile. Déterminer quelle est réellement l'intrusion détectée et entreprendre certaines actions pour y mettre fin ou l'empêcher de se reproduire, ne font généralement pas partie du domaine de la détection d'intrusion. Cependant, quelques formes de réaction automatique peuvent être implémentées par l'interaction de l'IDS et de systèmes de contrôle d'accès tels que les pare-feu.

Deux techniques de détection d'intrusion sont généralement mises en œuvre par les IDS courants.

La détection d'abus (misuse detection). Dans la détection d'abus (aussi appelée détection de mauvaise utilisation), l'IDS analyse l'information recueillie et la compare (*pattern matching*, approche par scénarii) avec une base de données de signatures (motifs définis, caractéristiques explicites) d'attaques connues (i.e., qui ont déjà été documentées), et toute activité correspondante est considérée comme une attaque (avec différents niveaux de sévérité).

La détection d'anomalie (anomaly detection). La détection d'anomalie de comportement est une technique assez ancienne. L'idée principale est de modéliser durant une période d'apprentissage le comportement "normal" d'un système/programme/utilisateur en définissant une ligne de conduite (*dite baseline ou profil : Par exemple, profil de connexion : fréquence de login (combien de fois par jour/semaine l'utilisateur se connecte-t-il ?), lieu de login (statistiques sur la connexion distante/locale), etc. Un profil est donné par une métrique et un modèle statistique*), et de considérer ensuite (en phase de détection) comme suspect tout comportement inhabituel ; les déviations significatives par rapport au modèle de comportement "normal". Les modèles de détection d'anomalie incluent fréquemment des modèles statistiques.

Les concepts liés aux systèmes de détections d'intrusions ainsi d'autres aspects sont détaillés dans le chapitre suivant (Voir Chapitre 02).

1.7.6 Les systèmes de prévention d'intrusions (IPS)

Un IPS (Intrusion Prevention System) [35] est un ensemble de composants logiciels et matériels dont la fonction principale est d'empêcher toute activité suspecte détectée au sein d'un système. Contrairement aux IDS simples, les IPS sont des outils aux fonctions « actives », qui en plus de détecter une intrusion, tentent de la bloquer. Cependant, les IPS ne sont pas la solution parfaite comme on pourrait le penser. Plusieurs stratégies de prévention d'intrusions existent :

-host-based memory and process protection: surveille l'exécution des processus et les tue s'ils ont l'air dangereux (buffer overflow).

-session interception / session sniping : termine une session TCP avec la commande TCP Reset : « RST ». Ceci est utilisé dans les NIPS (Network Intrusion Prevention System).

-gateway intrusion detection : si un système NIPS est placé en tant que routeur, il bloque le trafic ; sinon il envoie des messages à d'autres routeurs pour modifier leur liste d'accès.

Un IPS possède de nombreux inconvénients. Le premier est qu'il bloque toute activité qui lui semble suspecte. Or, il est impossible d'assurer une fiabilité à 100% dans l'identification des attaques. Un IPS peut donc malencontreusement bloquer du trafic inoffensif ! Par exemple, un IPS peut détecter une tentative de déni de service alors qu'il s'agit simplement d'une période chargée en trafic. Les faux positifs sont donc très dangereux pour les IPS.

Le deuxième inconvénient est qu'un pirate peut utiliser sa fonctionnalité de blocage pour mettre hors service un système. Prenons l'exemple d'un individu mal intentionné qui attaque un système protégé par un IPS, tout en spoofant son adresse IP. Si l'adresse IP spoofée est celle d'un noeud important du réseau (routeur, service Web, ...), les conséquences seront catastrophiques. Pour palier ce problème, de nombreux IPS disposent des « white lists », c'est-à-dire des listes d'adresses réseaux qu'il ne faut en aucun cas bloquer.

Et enfin, le troisième inconvénient et non le moindre : un IPS est peu discret. En effet, à chaque blocage d'attaque, il montre sa présence. Cela peut paraître anodin, mais si un pirate remarque la présence d'un IPS, il tentera de trouver une faille dans celui-ci afin de réintégrer son attaque... mais cette fois en passant inaperçu.

Voilà pourquoi les IDS passifs sont souvent préférés aux IPS. Cependant, il est intéressant de noter que plusieurs IDS (Ex : Snort, RealSecure, Dragon, ...) ont été dotés d'une fonctionnalité de réaction automatique à certains types d'attaques.

1.8 Protocoles de sécurité

Dans cette section on va présenter quelques protocoles de sécurité [24][28]:

1.8.1 SSH (Secure SHell)

Développé par le Finlandais Tatu Ylonen en 1995, ce protocole permet de se connecter en ligne de commande sur une machine distante en toute sécurité. Il se base principalement les algorithmes RSA, IDEA et DEA. En fait, SSH utilise un système de cryptage hybride à clé publique (RSA) et à clé secrète. En outre SSH se sert des possibilités d'authentification de RSA pour s'assurer que les sujets sont bien ceux qu'ils prétendent être. A cet effet, SSH n'est pas cassable comme rsh/rshd par les attaques d'usurpation d'adresse IP. SSH est également capable d'établir des connexions encryptées entre une machine cliente et une machine serveur pour n'importe quel port TCP, par transmission de port (tunneling).

1.8.2 SSL (Secure Socket Layer)

SSL est le protocole de sécurité standard sur Internet qui crypte les données qui sont transmises sur Internet assurant ainsi le transfert des informations de façon sécuritaire sur Internet. En plus, SSL permet à un serveur de s'authentifier auprès d'un client et inversement pour garanti ainsi une communication sécurisée entre ces deux machines. Le fonctionnement de base du protocole SSL consiste en ces huit étapes:

1- Le client et le serveur obtiennent un certificat d'identité auprès d'un tiers de confiance.

2- Le client se sert de son certificat pour s'identifier auprès du serveur.

3- Le serveur envoie un message au client et lui demande de le crypter avec sa clé privée.

4- Le client crypte ce message à l'aide de sa clé privée et envoie le message crypté au serveur.

5- Le serveur décrypte le message du client en utilisant la clé publique du client, et le compare avec le message original. S'ils sont identiques, c'est la preuve que le client est bien la personne qu'il prétend être.

6- Le serveur se connecte au tiers de confiance qui a délivré le certificat au client, lui envoie un message et lui demande de le chiffrer.

7- Le tiers de confiance chiffre le message avec sa clé privée et le renvoie au serveur.

8- Le serveur décrypte le message reçu en utilisant la clé publique du tiers de confiance et le compare avec l'original, s'ils sont identiques le tiers de confiance est aussi le tiers de confiance qu'il prétend être. La communication pourra s'effectuer une fois que le client et le serveur se sont mis d'accord sur le choix de l'algorithme de cryptage à utiliser pour une transmission de données la plus sécurisée.

1.8.3 IPSec (IP Security)

Ce protocole, entièrement dédié aux communications privées et sécurisées à travers Internet, est un ensemble de protocoles développés par L'IETF [3]. IPSec fournit des mécanismes qui protègent les informations contenues dans les paquets IP. Ces mécanismes se résument en trois protocoles:

– *ESP (Encapsulating Security Payload):* Il garantit l'intégrité et la confidentialité des données dans le message original en combinant une fonction de hachage et un chiffrement. Cette sécurité, qui permet de contrer l'attaque appelée " Man in the Middle ", utilise généralement l'algorithme DES.

– *AH (Authentication Header):* C'est un procédé qui garantit l'authenticité des trames IP en ajoutant un champ utilisé pour vérifier l'authenticité des données renfermées dans le datagramme.

– *ISAKMP (IP Security Association Key Management Protocol):* Il régit l'échange de clés, en s'assurant qu'elles sont délivrées de façon sûre. ISAKMP vérifie l'identité des deux parties communicantes. Ceci est réalisé soit à travers des mots de passe appelés secrets pré-partagés, soit à laide de certificats. Une fois que l'identité de chaque partie est connue, ISAKMP échange une série de messages UDP afin de mettre en place les clés partagées.

1.8.4 PCT (Private Communication Technology)

PCT est un mécanisme qui garantit la protection des accès et des communications sur Internet. Il inclut les fonctions de confidentialité, d'authentification et d'identification mutuelle. Il repose sur la dernière évolution du standard SSL qu'il améliore en séparant l'authentification et le cryptage pour optimiser leur sécurité. PCT permet aux applications d'utiliser le cryptage à l'aide de clés sur 128 bits aux Etats-Unis, et de clés sur 40 bits dans les autres pays.

Enfin, pour terminer il convient de dire que le niveau de sécurité offert par ces protocoles dépend largement des paramètres qu'on leur assigne. D'autant plus que leurs utilisations sont contraintes par diverses conditions. Par exemple, l'apparition

d'infrastructures à clés publiques fonctionnelles est reconnue indispensable pour une utilisation pratique et répandue d'IPSec; SSL n'est d'usage libre que si la clé utilisée est inférieure à 40 bits, quant à l'acquisition et l'usage de SSH, ils sont soumis, généralement, à des conditions prévues par la législation en matière de cryptographie, il relève donc d'une autorisation préalable.

1.9 Conclusion

Les réseaux informatiques sont aujourd'hui indispensables à quasiment toutes les activités humaines. Leur étendue et leur complexité d'architecture ne cesse de s'accroître faisant naître de nouvelles exigences en matière de sécurité.

L'arsenal déployé pour la protection des ressources et de la communication au sein d'un réseau s'acharne pour défendre l'armature de la sécurité informatique formée de quatre buts différents mais complémentaires et indissociables, qui sont : la confidentialité, l'intégrité, l'authentification et la disponibilité.

Mais malgré toutes ces techniques utilisées pour décliner les menaces et les attaques sur le réseau, un système n'est jamais totalement sûr. La cryptographie a ses faiblesses : une clé de chiffrement peut être cassée. A cet effet, la recherche, inexorable, innove et trouve les solutions.

Nous nous intéressons dans le cadre de ce travail à l'un des outils les plus populaires qui est le système de détection d'intrusions. Pour cela, on a réservé le chapitre deux aux aspects et détails concernant ce type de système.

Chapitre 2
Systèmes de détection d'intrusion

2.1 Introduction

Les intrusions électroniques constituent un phénomène inévitable de l'informatique contemporaine, faisant de sorte que même à des milliers de kilomètres quelqu'un peut accéder à vos ressources et données, à votre insu, et faire ce que bon lui semble.

Face à ce danger, l'approche préventive, qui consiste à définir une politique de sécurité et de déployer les outils pour l'appliquer, n'est plus, désormais, suffisante. La nécessité de détecter toute tentative de violation de la politique de sécurité s'impose. Autrement dit le recours à un outil de détection automatique des intrusions (IDS, Intrusion Detection System) s'impose.

Dans le cadre de ce travail, nous nous intéressons à cet outil de sécurité. Pour cela ce chapitre est consacré à la présentation des détails concernant les systèmes de détection d'intrusion.

2.2 Définitions

Le concept de système de détection d'intrusions a été introduit en 1980 par JAMES ANDERSON [8]. Mais le réel départ du domaine était marqué par la publication d'un modèle de détection des intrusions par Denning en 1987 [9].

Au préalable, il est utile d'expliquer certains concepts pour pouvoir définir à la fin ce qu'est un système de détection d'intrusions [3][36] :

Intrusion : nous appellerons intrusion toute utilisation d'un système informatique à des fins autres que celles prévues, généralement dues à l'acquisition de privilèges de façon illégitime. L'intrus est généralement vu comme une personne étrangère au système informatique qui a réussi à en prendre le contrôle, mais les statistiques montrent que les utilisations abusives (du

détournement de ressources à l'espionnage industriel) proviennent le plus fréquemment de personnes internes ayant déjà un accès au système.

– **Mécanisme d'audit de sécurité :** C'est l'enregistrement des actions effectuées par les utilisateurs sur un système informatique, pour en faire une analyse ultérieure permettant de démasquer les auteurs d'intrusions.
– **Journal d'audit :** C'est l'ensemble des informations générées par les mécanismes d'audit.
– **Flux d'audit élémentaire :** C'est une suite temporelle reportant les événements se produisant sur une même partie du système et traduisant son comportement.
– **Flux d'audit :** nous appellerons flux d'audit une suite temporelle d'événements, pouvant être le mélange de plusieurs flux élémentaires.
– **détection des intrusions** consiste, alors, à analyser les informations collectées par les mécanismes d'audit de sécurité, à la recherche d'éventuelles attaques.
Les systèmes de détection des intrusions peuvent revêtir plusieurs formes:
– *Outils de diagnostic:* Ces produits scrutent le réseau à la recherche des points de vulnérabilités connus, de part les ordinateurs et tous les périphériques servant à acheminer les données (routeurs, concentrateurs et commutateurs). Ces outils peuvent également tester le pare-feu et générer un rapport complet sur l'état du réseau et suggérer les actions à entreprendre pour colmater les brèches recensées.
– *Anti- renifleurs:* Ces outils permettent d'évaluer les ordinateurs individuels d'un réseau afin de dépister ceux qui sont susceptibles d'être dans une situation de " promiscuité électronique ", c'est-à-dire pouvant être manipulés par un logiciel renifleur pour épier l'ensemble des activités du réseau.
– *Détecteurs d'attaques:* Ces logiciels servent à sonner l'alarme dès que survient une attaque électronique connue.
– *Les leurres:* Ces produits simulent un serveur présentant des failles connues. Ces serveurs factices attirent ainsi l'attention des véritables serveurs critiques [36].

2.3 Taxonomie des systèmes de détection d'intrusion

La détection d'intrusion [2][3][36] a longtemps été vue comme le moyen le plus prometteur de combattre les intrusions. Pourtant elle fait partie d'un ensemble plus grand de techniques anti-intrusions qui ont aussi leur place. [47] propose une taxonomie de toutes ces techniques, que l'on peut résumer en le schéma de la figure 2.1, et que l'on détaillera ci-après.

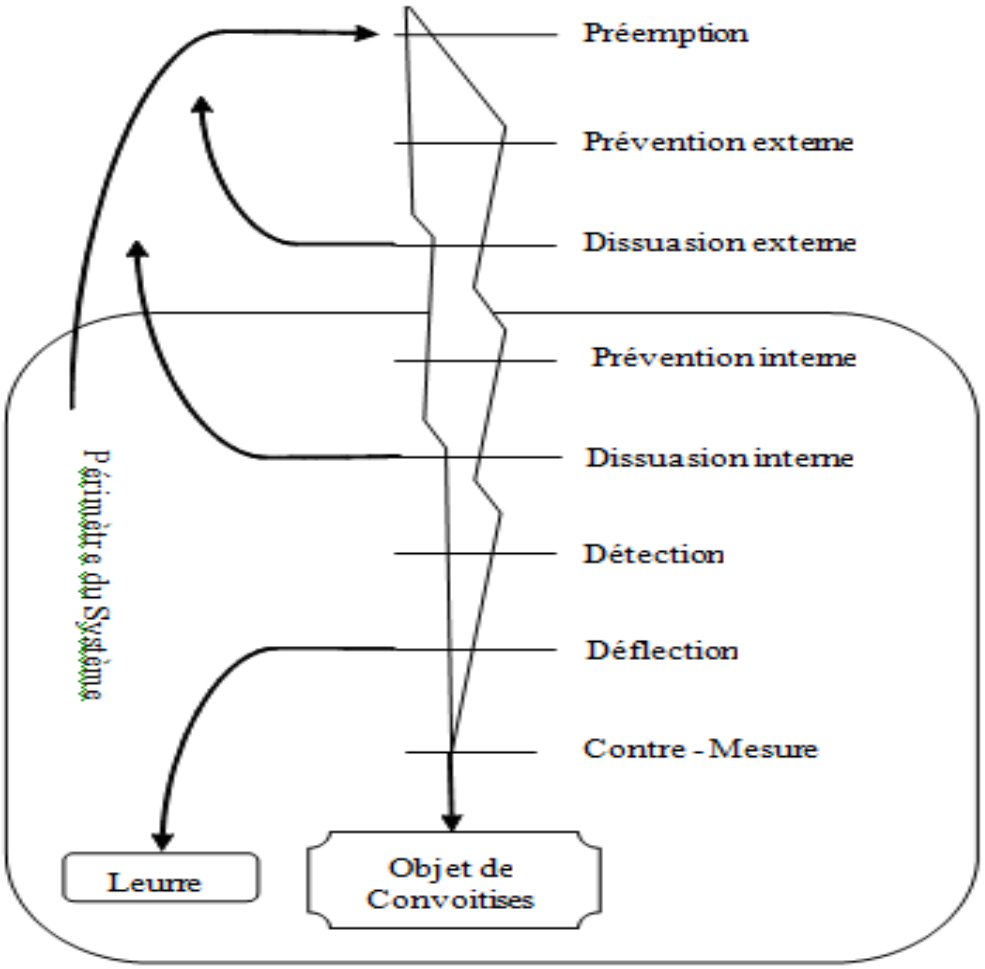


Figure 2. 1. Résumé des différentes techniques anti-intrusions [47]

– <u>Préemption</u>

Les techniques de préemption d'intrusion prennent place avant d'éventuelles tentatives d'intrusion pour les rendre moins susceptibles de se dérouler effectivement.

– Mise en œuvre :

– **Veille active :** le principe est de tenter d'empêcher de futures intrusions en cherchant les signes préliminaires d'une activité non désirée, en remarquant par exemple les étapes exploratoires d'une intrusion, et en prenant sans attendre des mesures contre les utilisateurs concernés.

– **Infiltration :** Une infiltration insidieuse pourrait être d'inonder les réseaux pirates d'informations fausses pour embrouiller ou dissuader ceux-ci.

– <u>Prévention</u>

Il s'agit de concevoir, implémenter et configurer le système assez correctement pour que les intrusions ne puissent avoir lieu, ou au moins soient sévèrement corrompues.

– *Mise en oeuvre :*

– **Conception et implémentation correcte :** Ceci se résume en l'utilisation des techniques classiques d'identification, d'authentification, de contrôle d'accès physique et logique.

– **Outils de recherche de failles :** Ce sont des outils qui recherchent un grand éventail d'irrégularité : mots de passe faibles, logiciels non autorisés, permissions d'accès ou propriété invalides, nœuds fantômes sur le réseau, etc.

– **Pare-feux :** ils examinent le flux d'information entre deux réseaux et les protègent l'un de l'autre en bloquant une partie du trafic qui semble non-conforme aux règles de sécurité entreprises par le système.

– Dissuasion

C'est la dévaluation du système par camouflage et l'augmentation du risque perçu par l'affichage de messages de mises en garde, ou qui sur-estiment la surveillance du système.

– *Mise en œuvre :*

– **Camouflage :** on peut cacher ou dévaluer les cibles du système et adopter comme politique de faire le moins de publicité possible sur les systèmes et leur contenu. Camoufler le système le rends moins utilisable et intuitif.

– **Mises en garde :** elles informent les utilisateurs que la sécurité du système est prise sérieusement en compte et exagèrent les pénalités encourues si une activité illégale est observée. Toutefois, Cette méthode permet à l'intrus de savoir à quelles signatures ou quels comportements le système réagit.

– **Paranoïa :** on essaye de donner l'impression à l'utilisateur qu'il est sous surveillance constante. La technique de la fausse alarme de voiture est le mécanisme le plus simple pour donner cette impression, notamment la technique de la caméra de surveillance.

– **Obstacles :** Le but est d'augmenter la quantité de temps et d'efforts qu'un intrus doit dépenser pour arriver à ses fins. On peut par exemple rallonger les temps d'exécution des commandes ou simuler une saturation des ressources pour l'exaspérer, sans pour autant lui donner l'impression d'être détecté. Toutes ces techniques peuvent cependant gêner les utilisateurs légitimes.

– Détection

Les données d'audit du système sont parcourues à la recherche de signatures connues d'intrusion ou de comportements anormaux, dans ces cas une alerte sera déclenchée au personnel qualifié tentera de remédier à l'intrusion.

On détaillera la mise en œuvre de la détection des intrusions dans la section des méthodes de détection d'intrusions.

– <u>Déflection</u>

La déflection d'intrusion fait croire à un attaquant qu'il a réussi à accéder aux ressources système alors qu'il a été dirigé dans un environnement préparé et contrôlé.

– *Mise en œuvre :*

– **Faux systèmes en quarantaine :** ils sont conçus pour faire croire aux intrus qu'ils sont sur le système cible. Cette déflection est faite par un frontal réseau tel qu'un routeur ou un firewall. Un faux système efficace encourage l'intrus à y rester assez longtemps pour que son identité et ses intentions soient découverts. Seulement, dédier une machine et des ressources pour l'entretenir coûte cher.

– **Faux comptes :** ils sont conçus pour faire croire à l'intrus qu'ils utilisent un compte normal corrompu alors qu'ils sont bloqués dans un compte spécial aux accès limités. Dans ce cas, les contrôles de déflection sont inclus directement dans le système d'exploitation et les commandes du système.

– <u>Contre-mesures</u>

Les contre-mesures donnent au système la capacité à réagir aux tentatives d'intrusions.

– *Mise en œuvre :*

– **Les ICE (Intrusion Countermeasure Equipment) :** Ces équipements permettent à un système de répondre en temps réel à une tentative d'intrusion, et ce de façon autonome. Les réactions peuvent être les suivantes:

– Alerter la personne chargée de la sécurité avec une alarme locale,

– Déconnecter tout le système du réseau,

– Annuler les paquets concernés,

– Journaliser les événements sur le système pour pouvoir revenir à un état considéré comme sûr,

– Ralentir le système ou rajouter des obstacles,

– Bloquer l'accès au réseau.

Pour présenter les caractéristiques des systèmes de détection d'intrusions nous retenons la classification proposée par L. Mé et C.Michel [17]. Celle-ci, représentée sur la figure 2.2, utilise les critères de classification suivants :

− Le principe de détection des intrusions,

− Le comportement après détection,

− La source des données,

− La fréquence d'utilisation.

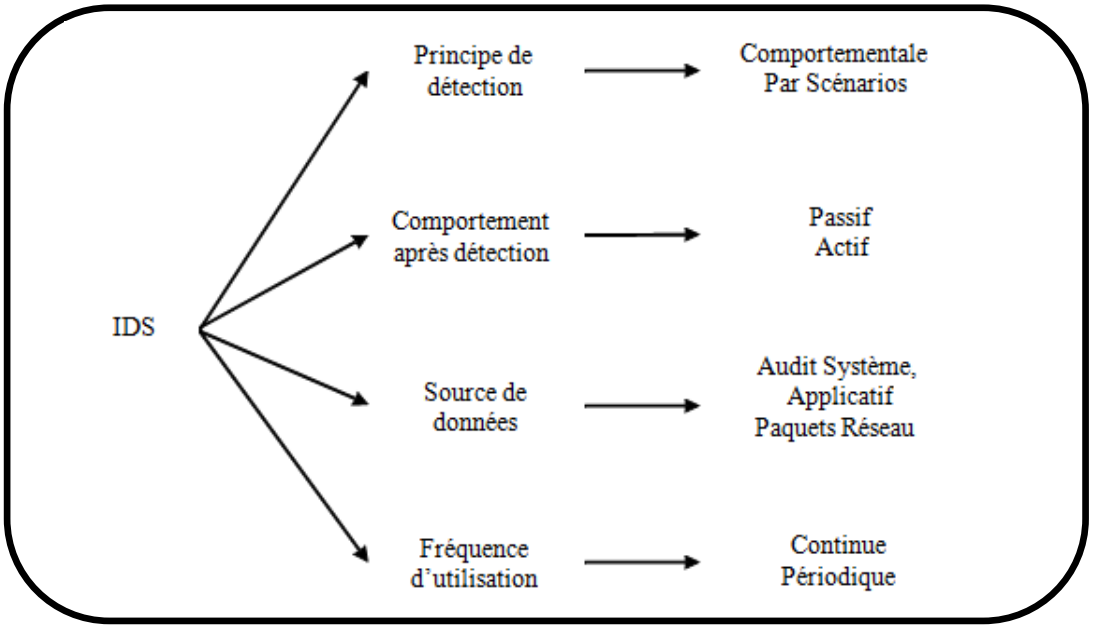


Figure. 2.2. Taxonomie des systèmes de détection d'intrusions [17]

Le principe de détection des intrusions. Les deux approches utilisées à ce jour sont l'approche comportementale et l'approche par scénarios. La première se base sur l'observation du comportement de l'utilisateur et sur la détection d'un comportement déviant par rapport à ses habitudes. Cette modification du comportement peut traduire une tentative d'intrusion, due soit à une usurpation de l'identité de l'utilisateur, soit à l'exécution par celui-ci de commandes non autorisées. La seconde approche consiste à identifier chaque attaque par une signature propre et ensuite à rechercher dans les fichiers d'audits du système les traces de ces signatures. On peut noter que cette approche par scénarios nécessite de connaître la signature d'une attaque avant de pouvoir la détecter. L'approche comportementale permet de détecter des attaques inconnues, mais la définition du comportement normal d'utilisateur demeure la principale difficulté.

Le comportement après détection. La nature de la réponse apportée par les systèmes après détection d'une intrusion peut aussi être utilisée pour classifier les IDS. Deux types de réponses sont implémentés à ce jour, soit une réponse passive, par exemple l'émission d'une alarme à l'administrateur, soit une réponse active, comme la mise en place de nouvelles règles de filtrage sur un pare-feu.

La source des données. Les IDS peuvent être classés en fonction de la provenance de leurs données d'audit, selon qu'elles viennent du système, des applications, des paquets du réseau ou encore d'un autre IDS.

La fréquence d'utilisation. La fréquence d'analyse des données d'audit est aussi un élément distinctif des systèmes de détection d'intrusions. Certains IDS peuvent surveiller en permanence le système d'information tandis que d'autres se limitent à une analyse périodique.

2.4 L'audit de sécurité

J. P. ANDERSON [8] a été le premier à montrer le rôle crucial que joue l'audit de sécurité dans un processus de sécurisation d'un système informatique. La reconstitution des opérations entreprises par certains utilisateurs spécifiques du système, dits utilisateurs audités, faite à partir de l'étude des séquences d'événements contenues dans le journal d'audit doit permettre de répondre aux six questions suivantes :

– Quelle opération a été faite?

– Qui a fait l'opération ? Cette question peut être posée autrement: Quelle est la paire identifiant et label du processus et de l'utilisateur pour le compte duquel il agit?

– Quelles ressources du système ont-t-elles été affectées par l'opération?

– Quand l'opération a-t-elle été réalisée?

– Où l'opération est-elle arrivée? Par exemple, dans le cas où l'opération a conduit à un enregistrement sur un serveur distant, on enregistre l'identifiant de ce serveur.

– Quelles sont les raisons qui ont fait qu'une opération a échoué?

La réponse à ces questions nécessite la spécification, préalable, des utilisateurs et des activités système à auditer, la garantie de la collecte des événements dans le fichier d'audit et l'analyse régulière de ce fichier. Enfin il est nécessaire de prévoir la réparation des dégâts éventuellement détectés.

2.4.1 Spécification des activités système à auditer

Les activités systèmes sont diverses et n'ont pas toutes le même niveau d'importance en matière de sécurité. Au vue du niveau de sécurité souhaité, on peut avoir cette liste d'informations pertinentes à auditer:

– **Informations sur les accès au système :**

L'identifiant de l'utilisateur ou du processus, l'horodatage de l'accès au système, l'identifiant du terminal, l'adresse du site cible de l'opération et le mode d'entrée (interactif, batch local, connexion distante) constituent des informations utiles dans le sens où elles peuvent nous renseigner sur le début d'un processus de violation éventuelle de la sécurité.

– **Informations sur l'usage fait du système :**

Ce sont des informations à propos des ressources systèmes qui ont été utilisées. Elles décrivent aussi comment ces ressources ont été exploitées.

– **Informations sur l'usage fait des fichiers :**

Ces informations relèveront des détails tels que l'horodatage de l'accès, source de l'accès (utilisateur, terminal ou application), type de l'accès (ouverture, fermeture, lecture, modification, purge du fichier, etc), volume d'informations échangées lors de l'accès.

– **Informations relatives à chaque application :**

On s'intéresse aux événements produits par toute application et qui peuvent avoir une influence sur la sécurité du système. On pourra enregistrer les événements suivants :

– Les commandes exécutées et leurs résultats,

– Les lancements et arrêts d'application,

– Les modules réellement exécutés,

– Les données entrées,

– Les sorties produites.

– **Informations sur les violations éventuelles de la sécurité :**

C'est des informations relatives à tous les événements pour lesquels il y a eu une tentative d'accès illégale par rapport à la politique de sécurité en vigueur. Parmi ces événements, on peut citer :

– Le changement des droits d'accès à des fichiers sensibles,

– La tentative d'exécution de certaines commandes du système réservées à des utilisateurs privilégiés,

– La tentative d'accès à un fichier non autorisée ou la fourniture d'un mot de passe erroné pour cet accès,

– La tentative d'exécution d'une application dans un mode privilégié (par exemple la modification des droits d'accès sous Unix),

– L'accès au système à des moments ou depuis des lieux inhabituels.

– **Informations statistiques sur le système :**

Les remarques de tout excès ou absence d'un événement d'une manière inhabituelle (le niveau anormalement élevé des refus d'accès au système, le niveau anormalement élevé ou bas de l'usage de certaines commandes du système) sont utiles pour pouvoir tirer des conclusions en matière de sécurité.

2.4.2 Collecte des événements

La plupart des systèmes d'exploitation disposent d'un sous-système d'audit capable de générer certains types d'événement. Le noyau du système assure alors la génération et la collecte de ces événements.

2.4.3 Analyse du journal d'audit

Analyser le journal d'audit revient à identifier toute violation des règles de la politique de sécurité. Explicitement l'analyse permet de déterminer les responsables, identifier les dégâts éventuels, tenter de les réparer et proposer des plans d'action qui assurent la sécurité du système [36]. L'analyse du journal d'audit nous permet de détecter des actions malveillantes portant atteinte à la confidentialité (vol de données, inférence illégitime, etc) ou à l'intégrité (modification non autorisée des fichiers de données) ou à la disponibilité (utilisation illégitime ou abusive des ressources, destruction illicite ou abusive de fichier, réduction illégale des droits des utilisateurs, etc).

Les actions qui peuvent être entreprises à l'encontre d'une intrusion en cours, sont la déconnexion de l'intrus ou son confinement dans des répertoires particuliers en vue de réduire les dégâts possibles.

2.4.4 Fréquence de l'analyse des traces d'audits

Il n'est plus suffisant aujourd'hui de faire des analyses journalières. Pratiquement, il est d'usage d'envisager un écart de quelques minutes et même de quelques secondes entre deux analyses successives. En fait, les analyses doivent être fréquentes afin que le minimum de prévarications reste indétecté.

2.4.5 Protection du journal d'audit

Le journal d'audit constitue la base et le point de départ de l'analyse faite en vue de détecter tout comportement déviant des règles de sécurité. Pour cette raison, veiller à la confidentialité, l'intégrité et la disponibilité de ce journal est une pierre angulaire si on veut tirer des conclusions utiles et cohérentes à partir de son contenu.

2.4.6 L'audit dans le cas des réseaux

Lorsqu'on est amené à faire de l'audit sur un système distribué, il est impératif de disposer d'une base de temps commune (ou une fenêtre de temps [46]) qui permettra de gérer

les événements survenant sur toutes les machines connectées. En outre, il faut être capable de diffuser les types d'événements à auditer et de rassembler les journaux d'audit en provenance des différentes machines du réseau, de manière à constituer un journal d'audit global. Des considérations de sécurités doivent être prises pour ces transferts de données.

2.6 Classification des IDS

On peut classifier les systèmes de détection d'intrusions en se basant sur leurs emplacements dans le système informatique, et leurs sources de données en deux grandes familles distinctes : les N-IDS et les H-IDS qui seront décrits dans les sous sections suivantes [1][2].

2.5.1 Les N-IDS (Network Based IDS)

Les NIDS *(Network Based Intrusion Detection Systems)* sont probablement les systèmes les plus connus (voir la figure 2.3). Ces systèmes pouvant être assimilés à un sniffer, capturent et décodent toutes les trames qui transitent par le segment sur lequel il est connecté. Toutefois, contrairement à un sniffer, cette sonde analyse les paquets IP dans leur intégralité afin de repérer des signatures d'attaque déjà connues ou des anomalies dans les entêtes des paquets. Internet, cette interconnexion des systèmes informatiques de plus en plus étendue et complexe fait apparaître continuellement de nouvelles attaques. Les NIDS s'avèrent d'un grand secours pour le fait qu'ils ne se suffisent pas aux sources d'audit locales. Les NIDS utilisent des dispositifs matériels et logiciels développés en vue de capturer le trafic réseau en temps réel.

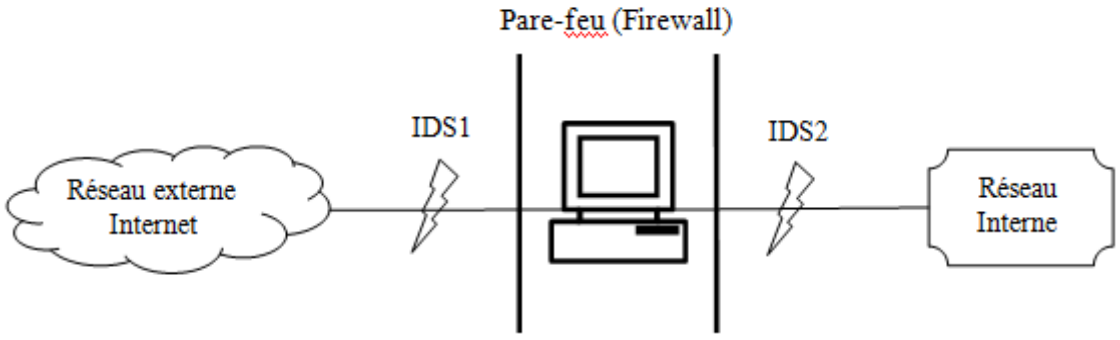


Figure 2.3 Installation des N-IDS [2]

2.5.2 Les H-IDS (Host Based IDS)

Les HIDS (Host Based Intrusion Detection Systems) représentent le complément naturel des NIDS. Installer des HIDS sur les machines à protéger revient à les pourvoir d'agents logiciels qui offrent trois services:

– Détection d'attaques contre des applications installées sur le système protégé,

– Vérification de l'intégrité des fichiers sensibles,

– Corrélation des fichiers journaux en provenance d'applications ou d'équipements tiers tels les routeurs, firewalls, commutateurs.

Les HIDS utilisent des informations d'audit collectées, provenant essentiellement de la machine à protéger à partir de diverses sources (traces d'audit système, historique des commandes exécutées, etc).

2.5.3 Les systèmes de détection d'intrusions « hybrides »

Généralement utilisés dans un environnement décentralisé, ils permettent de réunir les informations de diverses sondes placées sur le réseau. Leur appellation « hybride » provient du fait qu'ils sont capables de réunir aussi bien des informations provenant d'un système HIDS qu'un NIDS.

2.6 Les méthodes de détection d'intrusions

Deux familles de méthodes ont été proposées, à ce jour [36]. La première consiste à observer le trafic sur le réseau, généralement en observant le contenu des paquets de données qui circulent d'une machine à l'autre, afin de détecter un " profil "d'attaque (ou signature) connu. La seconde approche consiste à définir un comportement normal du système et à rechercher tout ce qui dévie de ce comportement.

2.6.1 Approche comportementale (Anomaly Detection)

Cette approche, proposée en 1980 par J.P. Anderson [8] puis reprise et étendue par D.E. Denning [9], consiste à utiliser des méthodes basées sur l'hypothèse selon laquelle l'exploitation d'une vulnérabilité du système implique un usage anormal de celui-ci. Une intrusion est donc identifiable en tant que déviation par rapport au comportement habituel d'un utilisateur.

Il est, donc, primordial pour le fonctionnement de cette approche de pouvoir dresser un modèle de comportement. Il s'agit de choisir une méthode de description du comportement normal d'un utilisateur [36].

La mise en œuvre peut être réalisée par :

Observation de seuils : on fixe le comportement normal d'un utilisateur par la donnée de seuils à certaines mesures (par exemple, le nombre maximum de mots de passe erronés). On a ainsi une définition claire et simple des comportements non acceptés. Il est cependant difficile de caractériser un comportement intrusif en termes de seuils, et on risque beaucoup de fausses alarmes ou beaucoup d'intrusions non détectées sur une population d'usagers non uniforme.

Profilage d'utilisateurs : on crée et on maintient des profils individuels du travail des usagers, auxquels ils sont censés adhérer ensuite. Au fur et à mesure que l'utilisateur change ses activités, son profil de travail attendu se met à jour. Certains systèmes tentent de concilier l'utilisation de profils à court terme et de profils à long terme. Il reste cependant difficile de profiler un utilisateur irrégulier ou très dynamique. De plus, un utilisateur peut arriver à habituer lentement le système à un comportement intrusif.

Profilage de groupes : on place chaque utilisateur dans un groupe de travail qui montre une façon de travailler commune. Un profil de groupe est calculé en fonction de l'historique des activités du groupe entier. On vérifie que les individus du groupe travaillent de la manière que le groupe entier a définie par son profil. Cette méthode réduit drastiquement le nombre de profils à maintenir. De plus, un utilisateur peut beaucoup plus difficilement élargir le comportement accepté comme dans un profil individuel. Mais il est parfois dur de trouver le groupe le plus approprié à une personne : deux individus ayant le même poste peuvent avoir des habitudes de travail très différentes. Il est de plus parfois nécessaire de créer un groupe pour un seul individu.

Profilage d'utilisation de ressources : on observe l'utilisation de certains ressources comme les comptes, les applications, les mémoires de masse, la mémoire vive, les processeurs, les ports de communication sur de longues périodes, et on s'attend à ce qu'une utilisation normale n'induise pas de changement sur cette utilisation par rapport à ce qui a été observé par le passé. On peut aussi observer les changements dans l'utilisation des protocoles réseau, rechercher les ports qui voient leur trafic augmenter anormalement. Ces profils ne dépendant pas des utilisateurs et peuvent permettre la détection de plusieurs intrus qui collaboreraient.

Les écarts par rapport au profil sont cependant très durs à interpréter.

Profilage de programmes exécutables : on observe l'utilisation des ressources du système par les programmes exécutables. Les virus, chevaux de Troie, vers, bombes logiques et autres

programmes du même goût se voient démasqués en profilant la façon dont les objets du système comme les fichiers ou les imprimantes sont utilisés. Le profilage peut se faire par type d'exécutable. On peut par exemple détecter le fait qu'un daemon d'impression se mette à attendre des connexions sur des ports autres que celui qu'il utilise d'habitude.

Profilage statique c'est un profilage où la mise à jour du profil ne se fait pas en permanence mais seulement de temps en temps, et avec la bénédiction de la personne chargée de la sécurité. Dans le cas de profils utilisateurs, cela empêche l'usager d'élargir petit à petit son champ d'action. Cependant, les mises à jour peuvent être souvent nécessaires.

Profilage adaptatif : Le profil est mis à jour en permanence pour refléter les changements de comportement de l'utilisateur, du groupe ou du système. La personne en charge de la sécurité précise si la nouvelle activité est

– intrusive et des mesures doivent être prises,

– non intrusive, et peut être ajoutée au profil

– non intrusive, mais est une aberration dont la prochaine occurrence sera intéressante à connaître.

Profilage adaptatif à base de règles : cette technique diffère des autres profilages en ce que l'historique de l'utilisation est connu sous formes de règles. Contrairement à la détection de malveillances à base de règles, on n'a pas besoin des connaissances d'un expert. Les règles d'utilisation normale son générées automatiquement pendant la période d'apprentissage. Pour être efficace, beaucoup de règles sont nécessaires, et s'accompagnent d'autant de problèmes de performance.

Réseaux neuronaux : les réseaux neuronaux offrent une alternative à la maintenance d'un modèle de comportement normal d'un utilisateur. Ils peuvent offrir un modèle plus efficace et moins complexe que les moyennes et les déviations standard. L'utilisation de réseaux neuronaux doit encore faire ses preuves, et même s'ils peuvent s'avérer moins gourmands en ressources, une longue et minutieuse phase d'apprentissage est requise.

Approche immunologique : L'approche immunologique tente de calquer le comportement du système immunologique pour faire la différence entre ce qui est normal (le soi) et ce qui ne l'est pas (le non-soi). Le système immunologique montre en effet beaucoup d'aspects intéressants comme son mode d'opération distribué (il n'y a pas de système de contrôle central) qui lui permet de continuer à fonctionner même après des pertes, sa capacité à apprendre automatiquement de nouvelles attaques pour mieux réagir les prochaines fois qu'elles se présentent, sa capacité à détecter des attaques inconnues, etc. On peut voir l'approche immunologique de la détection d'anomalies comme une méthode de détection

d'anomalie où l'on utilise les techniques de détection des malveillances. En effet, les techniques de détection d'anomalie connaissent ce qui est bien et vérifient en permanence que l'activité du système est normale, alors que les techniques de détection de malveillance connaissent ce qui est mal et sont à sa recherche. L'approche immunologique propose de rechercher ce qui est mal en connaissant ce qui est bien. On devrait même dire que l'approche propose de rechercher ce qui n'est pas bien, et l'on peut se permettre la comparaison avec la notion de tiers exclus en logique : on n'obtient pas ce qui est mal en prenant la négation de ce qui est bien. Cependant, les résultats obtenus sont satisfaisants.

Inconvénients :

– Choix délicat des différents paramètres du modèle statistique.

– Hypothèse d'une distribution normale des différentes mesures non prouvée.

– Choix des mesures à retenir pour un système cible donné délicat.

– Difficulté à dire si les observations faites pour un utilisateur particulier correspondent à des activités que l'on voudrait prohiber.

– Pour un utilisateur au comportement erratique, toute activité est normale.

– Pas de prise en compte des tentatives de collusion entre utilisateurs.

– En cas de profonde modification de l'environnement du système cible, déclenchement d'un flot ininterrompu d'alarmes.

– Utilisateur pouvant changer lentement de comportement dans le but d'habituer le système à un comportement intrusif.

2.6.2 Approche par scénarios (misuse detection)

L'approche par scénarios [3][36] est basée sur la recherche d'activités abusives, par comparaison avec des descriptions abstraites de ce qui est considéré comme malveillant. Cette approche dresse les règles qui décrivent les tentatives d'attaque, en s'appuyant sur des intrusions passées ou des faiblesses théoriques connues. L'efficacité de cette détection dépend de l'acuité et la couverture de tous les abus possibles par les règles. Une règle peut être relative à un seul événement malveillant une séquence d'événements représentant un scénario d'intrusion. En somme, la détection des intrusions par recherche de scenarii repose sur une base de signatures d'intrusions et guette ces signatures dans le journal d'audits.

La mise en œuvre peut être réalisée par :

Systèmes experts : ils peuvent être utilisés pour coder les signatures de malveillance avec des règles d'implication *si . . .alors*. Les signatures décrivent un aspect d'une attaque ou d'une classe d'attaque. Il est possible de rentrer de nouvelles règles pour détecter de nouvelles attaques. Les règles deviennent généralement très spécifiques au système cible et donc sont peu portables.

Raisonnement sur des modèles : on essaye de modéliser les malveillances à un niveau élevé et intuitif d'abstraction en termes de séquences d'événements qui définissent l'intrusion. Cette technique peut être utile pour l'identification d'intrusions qui sont proches mais différentes. Elle permet aussi de cibler les données surs lesquelles une analyse approfondie doit être faite. Mais en tant qu'approche recherchant des signatures, elle ne peut que trouver des attaques déjà connues.

Analyse des transitions d'états : on crée un modèle tel qu'à l'état initial le système ne soit pas compromis. L'intrus accède au système. Il exécute une série d'actions qui provoquent des transitions sur les états du modèle, qui peuvent être des états ou l'on considère le système compromis. Cette approche de haut niveau peut reconnaître des variations d'attaques qui passeraient inaperçues avec des approches de plus bas niveau.

Réseaux neuronaux : la flexibilité apportée par les réseaux neuronaux peut permettre d'analyser des données même si elles sont incomplètes ou déformées. Ils peuvent de plus permettre une analyse non-linéaire de ces données. Leur rapidité permet l'analyse d'importants flux d'audit en temps réel. On peut utiliser les réseaux neuronaux pour filtrer et sélectionner les informations suspectes pour permettre une analyse détaillée par un système expert. On peut aussi les utiliser directement pour la détection de malveillances. Mais leur apprentissage est extrêmement délicat, et il est difficile de savoir quand un réseau est prêt pour l'utilisation. On peut également lui reprocher son côté boite noire (on ne peut pas interpréter les coefficients).

Algorithmes génétiques : on définit chaque scénario d'attaque comme un ensemble pas forcément ordonné d'événements. Lorsqu'on veut tenir compte de tous les entremêlements possibles entre ces ensembles, l'explosion combinatoire qui en résulte interdit l'usage d'algorithmes de recherche traditionnels, et les algorithmes génétiques sont d'un grand secours.

La détection d'intrusion par recherche de scenarii repose sur une base de signatures d'intrusions et recherche ces signatures dans le journal d'audits. On peut rapprocher les méthodes utilisées à celles que l'on peut rencontrer dans le domaine des antivirus, où on

recherche la signature de certains programmes dans les fichiers du système informatique, ou encore dans le domaine de la génomique où l'on recherche une séquence d'ADN dans un brin, ou, plus généralement, tout ce qui s'apparente à l'appariement de séquences.

Inconvénients :

– Base des signatures difficile à construire.

– Pas de détection d'attaques non connues.

2.6.3 Systèmes hybrides

Pour tenter de compenser quelques inconvénients de chacune des techniques, certains systèmes utilisent une combinaison de la détection d'anomalies et de la détection de malveillances [36]. Par exemple, un compte d'administrateur aura un profil qui lui permet d'accéder à certains fichiers sensibles, mais il peut être utile de vérifier que des attaques connues ne sont pas utilisées contre ces fichiers. À l'inverse, utiliser des fichiers comportant le mot "nucléaire" ne caractérise aucune signature d'attaque, mais il peut être intéressant de savoir que cela est arrivé si ce n'était pas dans les habitudes de l'utilisateur.

2.8 Domaines impliqués dans la détection d'intrusions

Dans cette section, on va présenter quelques domaines utilisés pour la détection d'intrusions.

2.7.1 Data mining

Le but est d'extraire des modèles descriptifs des énormes volumes de données d'audit. Plusieurs algorithmes du domaine du data mining peuvent être utiles. :

Classification : la classification associe chaque élément de donnée à une plusieurs catégories prédéfinies. Ces algorithmes de classification génèrent des classifieurs, sous forme d'arbres de décision ou de règles. Une application dans la détection d'intrusion serait d'appliquer ces algorithmes à une quantité suffisante de données d'audit normales ou anormales pour générer un classifieur capable d'étiqueter comme appartenant à la catégorie normale ou anormale de nouvelles données d'audit [85].

Analyse de relations : on établit des relations entre différents champs d'éléments d'audit, comme par exemple la corrélation entre la commande et l'argument dans l'historique des commandes de l'interpréteur de commandes, pour construire des profils d'usage normal. Un

programmeur, par exemple pourrait avoir une forte relation entre *emacs* et des fichier *C*. On définit ainsi des règles d'association.

Analyse de séquences : Ces algorithmes tentent de découvrir quelles séquences temporelles d'événements se produisent souvent en même temps. On peut noter que [48] utilise le Common Intrusion Detection Framework (CIDF). Le CIDF est un effort pour développer des protocoles et des APIs pour permettre aux projets de recherche sur la détection d'intrusion de partager les informations et les ressources, et pour que les composants de détection d'intrusion puissent être réutilisés. Un Internet Engineering Task Force (IETF) working group a été créé et nommé Intrusion Detection Working Group (IDWG).

2.7.2 Agents

Plusieurs domaines de recherche dans les Mobile Agents Intrusions Detection System (MAIDS) sont ouverts par [49]. Certains sont mis en oeuvre dans [50], [51], [52] ou [53].

Détection multi-point : La détection multi-point est faite en analysant les flux d'audit de plusieurs hôtes pour détecter des attaques distribuées ou autres stratégies d'attaques d'un réseau dans sa globalité. Il est rarement possible de transporter tous les flux d'audit à un IDS central, et même dans ce cas, la détection est difficile.

Les agents mobiles peuvent apporter le calcul distribué, le fait qu'on transporte l'analyseur au flux d'audit et non le flux d'audit à l'analyseur. Les agents pourraient détecter ces attaques, les corréler, et découvrir les stratégies d'attaques distribuées.

Architecture résistante aux attaques : Une architecture hiérarchique est souvent utilisée pour des raisons de performances et de centralisation du contrôle. Cette conception a souvent plusieurs lignes de communication non-redondantes. Un intrus peut couper une branche de l'IDS, voire le désactiver totalement en le décapitant.

Les agents mobiles peuvent apporter quelques solutions à ce problème :

– une architecture complètement distribuée

– une architecture hiérarchique standard on un agent peut remplacer chaque nœud et ramener une fonctionnalité perdue

– des agents mobiles qui se déplacent quand une activité suspecte est détectée.

Partage de connaissances : Souvent, plusieurs systèmes de détection d'intrusion différents fonctionnent sur un site. Idéalement, ils partageraient les informations sur les attaques récentes pour améliorer la détection d'attaques futures.

Même si ce n'est pas un domaine de recherche réservé aux MAIDS, ceux-ci permettent une approche plus naturelle de ce genre de choses.

Agents errants : L'échantillonnage aléatoire est utilisé avec succès depuis de longues années dans le domaine du contrôle de qualité, et les mathématiques sous-jacentes sont bien comprises et les paramètres peuvent être calculés.

Chaque agent effectue un test spécifique et peut errer aléatoirement d'hôte en hôte. Quand des tests indiquent la possibilité d'une intrusion, des tests plus poussés peuvent être effectués sur l'hôte suspect.

Imprévisibilité : Un intrus peut pénétrer dans un système sans être immédiatement détecté par l'IDS. Cela peut lui laisser le temps d'effacer ses traces ou de neutraliser l'IDS.

Un MAIDS reste vulnérable à cela mais se trouve néanmoins pourvu de quelques avantages. Lorsqu'un agent arrive sur un hôte, il transporte du code non encore altéré, et pourrait par exemple tester l'intégrité de la plateforme IDS locale. L'arrivée des agents et leur manière de fonctionner peuvent être imprévisibles, et il peut être très dur de rester inaperçu.

Diversité génétique : Les IDS à base d'agents mobiles peuvent être vus comme un ensemble d'entités autonomes. Chaque agent peut être différent des autres par son code ou par les données sur lesquelles il travaille. Cependant, s'ils ne diffèrent que par leurs données, leurs tests peuvent devenir prévisibles.

Si chaque agent d'une même classe avait une façon différente de détecter la même chose, il serait autrement plus difficile de prévoir quoi que ce soit. Une manière de faire les choses serait de décrire ce qu'il faut détecter dans un langage standard et de laisser chaque instance de la classe trouver une manière de le détecter. Les agents avec un faible taux de détection pourraient essayer de muter en introduisant de légères modifications dans leur façon de détecter l'attaque.

2.7.3 Réseaux de neurones

Les réseaux neuronaux sont utilisés pour leur rapidité de traitement et leur relative résistance aux informations incomplètes ou déformées. [54] les utilise de deux manières différentes. Ils sont d'abord utilisés comme filtres pour filtrer et sélectionner les parties suspectes dans les données d'audit. Celles-ci sont ensuite analysées par un système expert. On peut ainsi réduire les fausses alarmes, et on peut même augmenter la sensibilité du système expert car il ne travaille que sur des données suspectes. Puis ils sont utilisés de façon à prendre seuls la décision de classer une séquence d'événements comme malveillante.

2.7.4 Immunologie

Le système immunitaire biologique est précisément conçu pour détecter et éliminer les infections. Il se montre capable de plusieurs propriétés que l'on aimerait voir figurer dans des systèmes artificiels.

Il est tout d'abord robuste, du fait de sa diversité, de sa distribution, de son dynamisme et de sa tolérance aux fautes. La diversité améliore la robustesse, deux éléments peuvent ne pas être vulnérables aux mêmes infections. Le fait que le système immunitaire soit distribué, que plusieurs composants interagissent localement pour obtenir une protection globale, permet de ne pas avoir de centre de contrôle, de point faible. Sa dynamique, le fait que ses agents sont continuellement créés, détruits et circulent augmentent la diversité temporelle et spatiale. Elle est tolérante aux fautes car l'effet d'un individu est faible, et quelques erreurs ne sont pas catastrophiques. Ensuite, le système immunitaire est adaptatif, c'est-à-dire qu'il est capable d'apprendre et de reconnaître de nouvelles infections.

Enfin, il est autonome. Il ne nécessite pas de contrôle extérieur. De plus, comme il fait partie du corps, il se protège lui-même. L'algorithme utilisé par le système immunitaire pour détecter les intrusions est appelé algorithme de sélection négative. Les lymphocytes sont appelés détecteurs négatifs parce qu'ils sont conçus pour se lier au non-soi, c'est-à-dire que lorsqu'un lymphocyte est activé, le système immunitaire répond à une intrusion. Les lymphocytes sont créés avec des récepteurs générés aléatoirement. Le récepteur décide à quoi s'accrochera le lymphocyte. Comme ils sont générés aléatoirement, ils peuvent détecter aussi bien le soi que le non-soi. Aussi y a-t-il une période pendant laquelle le lymphocyte n'est pas mature et meurt s'il s'accroche à quelque chose. Les lymphocytes qui arrivent à maturité sont donc sensés détecter le non-soi. Lorsque le système immunitaire rencontre pour la première fois un certain type d'agents pathogènes, il produit une réponse primaire, qui peut prendre plusieurs semaines pour éliminer l'infection. Pendant cette réponse, il apprend à reconnaître ces agents et si une deuxième infection de ce type se déclare, il déclenchera une réponse secondaire en général assez efficace pour que l'infection passe inaperçue. La réponse primaire est lente parce qu'il peut n'y avoir qu'un petit nombre de lymphocytes qui s'attachent à eux. Pour accroître leur efficacité, chaque lymphocyte activé se clone, et on a ainsi une croissance exponentielle de la population qui peut détecter ces agents. Une fois l'infection éliminée, le système immunitaire garde cette population de lymphocytes qui avait une grande affinité avec ces agents pathogènes, alors que les autres lymphocytes ont une durée de vie de quelques jours. Les lymphocytes T sont créés et maturent uniquement dans le thymus. Bien que l'on

rencontre en ce lieu la grande majorité des protéines du corps, il se peut que certains lymphocytes arrivent à maturité avec un détecteur qui s'attache à des cellules saines. Pour éviter cela, pour devenir actif, un lymphocyte nécessite une costimulation, c'est-à-dire que s'il s'attache sur une cellule, il lui faut aussi être stimulé par un second signal, qui est généralement un signal chimique généré quand le corps est agressé. Ce signal provient du système immunitaire lui-même ou d'autres cellules.
[55] tentent d'appliquer vaguement quelques un de ces principes. Par contre, ARTIS (ARTificial Immune System) [56] calque parfaitement tous ces comportements, plus finement détaillés pour arriver à un résultat très encourageant.

2.7.5 Algorithmes génétiques

Comme expliqué dans [57], les algorithmes génétiques ont été proposés par John Holland dans les années 70 [58]. Ils s'inspirent de l'évolution génétique des espèces, et plus précisément du principe de sélection naturelle. Il s'agit d'algorithmes de recherche d'optimums. On ne fait aucune hypothèse sur la fonction dont on recherche l'optimum. En particulier, elle n'a pas à être dérivable, ce qui est un avantage considérable sur toutes les méthodes classiques de recherche d'optimum. Un algorithme génétique manipule une population de taille constante d'individus représentés par une chaîne de caractères d'un alphabet. Chaque individu code chacun une solution potentielle au problème. La population est créée aléatoirement, puis elle évolue, génération par génération. À chaque génération, de nouvelles créatures sont créées en utilisant les plus fortes, tandis que les plus faibles sont éliminées, les adjectifs *faible* et *fort* étant bien sur déterminés par une fonction sélective, dépendant fortement de la fonction dont on cherche l'optimum. Des mutations peuvent aussi se produire, pour éviter à une population de perdre irrémédiablement une information sur la solution. On a donc trois transformations de la population : sélection, recombinaison et mutation. Cela a inspiré [59], article n'est pas resté sans suite puisque le logiciel GASSATA [60] a ensuite été créé.

2.8 Analyse temps réel et analyse en mode batch

L'analyse des données peut se faire soit en temps réel, soit plus tard [1] (i.e. en mode "batch"). Dans les systèmes de détection d'intrusions fonctionnant en mode "batch"(référé), l'analyse des données ne se fait qu'après que les données aient été collectées. Le principal avantage d'une telle analyse est qu'elle peut se faire à un moment où le taux de CPU utilisé est

au plus bas et/ou sur un système totalement différent [61]. Son désavantage est que certaines attaques ne seraient détectées qu'une fois que des dommages considérables auraient été causés. Les systèmes de détection d'intrusions, fonctionnant en mode "batch", peuvent être très utiles dans des environnements où les résumés périodiques des utilisations suspicieuses sont suffisants [62].

Quant aux systèmes de détection d'intrusions fonctionnant en temps réel, les données d'audit doivent être analysées dès qu'elles sont créées. Ce mode d'analyse peut se révéler crucial pour des systèmes soucieux d'identifier les comportements utilisateurs suspicieux pendant qu'ils se produisent et de détecter de façon imminente toute violation de sécurité. Cependant, l'analyse en temps réel, peut engendrer un problème de performance matérielle et logicielle. En effet, une haute fiabilité, une grande capacité de stockage et un matériel à très haut débit, deviennent des problèmes clés [62]. De plus le temps entre le moment où une transaction d'audit se produit et le moment où elle est écrite sur le disque, doit être réduit au maximum.

2.9 Réponses des systèmes de détection d'intrusions

L'utilité d'un IDS est de réagir après, l'analyse des activités, auprès de l'hôte et du réseau. La nature de la réponse apportée par ces systèmes après détection d'une intrusion peut aussi être utilisée pour les classifier. Deux types de réponses sont implémentés à ce jour, soit une réponse passive, par exemple l'émission d'une alarme à l'administrateur, soit une réponse active, comme la mise en place de nouvelles règles de filtrage sur un pare-feu. Dans ce qui va suivre on va donner un aperçu sur ces deux méthodes [2].

2.9.1 Réponse Active

Une Réponse active se traduit par une réaction automatique si l'IDS détecte une attaque ou un essai d'attaque. En fonction de la sévérité de l'attaque, la plupart des IDS offrent plusieurs options de réponses :

Contre mesures : Agir contre l'intrus potentiel peut inclure une variété d'étapes comme bloquer l'accès de la personne ou déclencher des attaques contre elle. Cette dernière mesure est souvent facile et attractive mais reste illégale, surtout avec la présence de l'effet « Tierce Partie ». Cet effet signifie que l'intrus attaque d'abord une personne ou un réseau innocent, ensuite il utilise ce réseau pour attaquer d'autres réseaux. Le problème ici est qu'on va considérer le réseau innocent comme attaquant et la contre attaque sera menée contre

l'innocent tout en négligeant le vrai intrus. Comme résultat de cette contre-attaque, des dommages indéterminés peuvent être causés à l'innocent et on sera responsable de toutes ces atteintes, y compris celles de l'intrus, surtout si ce dernier est assez intelligent pour cacher ses traces.

Collecte des informations complémentaires : Ces données concernent l'intrus, son attaque, et ses implications, de cette manière, des profils d'attaquants seront créés, avec lesquels on aura l'avantage de pouvoir analyser les journaux (log) en détail, fermer les échappatoires possibles et cela va permettre aussi de prendre des mesures légales. Par exemple si un IDS détermine qu'un utilisateur spécifique a étendu avec succès ses privilèges, l'observation de cet utilisateur peut être étalée, par exemple, en enregistrant ses commandes, les endroits où cet utilisateur s'introduit, combien de temps il reste, quand s'introduit-il, essaie-t-il de transférer des données spécifiques, etc.

Changement de la configuration : Ces changements concernent le système, le pare-feu, etc. Pour cela on peut, par exemple, bloquer et noter les demandes d'accès d'origines suspectes, si un attaquant utilise des adresses IP spécifiques, on peut l'empêcher de se connecter au réseau avec ces adresses. Dans certains cas spécifiques, il serait utile de bloquer simplement tout accès au réseau lui-même ou toute requête à des ports ou à des interfaces réseaux spécifiques.

Une autre possibilité de Réponse Active est d'arrêter une connexion TCP, connue aussi sous le nom de « TCP kill ». Pour déconnecter un autre ordinateur, on peut envoyer un RST (Drapeau Reset). Pour MS - Windows NT, il existe un utilitaire portant le même nom.

Comme on peut le constater, il y a une multitude de possibilités complètes de réaction actives qui peuvent remplacer les contre attaques qui ne devraient pas, en fait, être effectuées.

2.9.2 Réponse Passive

La réponse passive consiste à noter des alertes qui doivent être examinées par l'administrateur de sécurité. Ces alertes représentent n'importe quelle sorte d'avis pour l'utilisateur de l'IDS d'une activité d'intrus. Les alertes peuvent prendre la forme d'avertissements, de conseils, de notations ou bien elles seront représentées par la génération de rapports ou de comptes- rendu de surveillance des systèmes pour une période bien définie. Si, par exemple, on essaie de démarrer des services spécifiques ou d'effacer des fichiers système importants, un avertissement annonçant l'incident peut être généré, indiquant également qui a mené cette opération et à quel moment.

Les alertes lancées sont généralement stockées dans des fichiers (journaux) ou dans des bases

de données où elles peuvent être examinées plus tard par des experts en matière de sécurité. Généralement un IDS peut produire des alertes sous beaucoup de formes et peut également envoyer la même alerte à des destinations multiples. Presque tous les IDS assurent des réponses passives et ils sont capables de générer des avertissements ou d'envoyer des indications à l'utilisateur, ainsi le statut d'un système peut demeurer surveillé pendant une longue période.

2.10 Installation des systèmes de détection d'intrusions

Pour pouvoir configurer un IDS d'une manière efficace, il faut bien comprendre son fonctionnement interne [2]. Toute erreur lors de son installation pourra le rendre inefficace ou inutilisable. Lors du déploiement d'un IDS, il faut le configurer correctement en fonction de l'OS, des applications et du matériel utilisés, et il faut avoir à l'esprit qu'il est impensable de vouloir analyser tout le trafic d'un réseau. Il faut donc donner la priorité aux systèmes à risque. De plus, afin de réduire la charge du senseur (une sonde, placée sur le réseau, dont le rôle est d'attraper le trafic circulant sur celui-ci) de l' IDS, Il est souvent préférable de le placer après le pare-feu du côté interne. Ainsi, seul les flux acceptés par le pare-feu seront analysés.

Le choix matériel a également une grande importance. Puisqu'une sonde d'un IDS doit pouvoir analyser le trafic réseau quelque soit sa destination, elle devra recevoir tous les paquets. Mais le matériel réseau pose parfois des problèmes :

L'utilisation des commutateurs simples (switchs) rend la conversation avec l'IDS impossible du fait que le switch commute les paquets directement au destinataire. Il est dès lors impossible d'installer une sonde qui analysera le trafic global. De plus, Malgré que l'utilisation des concentrateurs (hubs) rende la conversation avec l'IDS possible car les concentrateurs répètent les paquets en les émettant à toutes les machines connectées, ces équipements sont peu fiables et sont donc à éviter.

La solution sera donc l'utilisation des switches professionnels qui copie le trafic et l'envoie sur un port spécifié (où sera placé le N-IDS) : SPANport (Switch Port Analyzer). Un autre problème doit être pris en considération : il est impossible pour un IDS de décrypter les flux lorsque ces derniers sont cryptés (ex : par SSL, qui est le cas pour les VPN). Il faut donc utiliser un proxy SSL pour traiter ce problème.

Un autre point ne doit pas être oublié qui est la sécurisation du senseur et des logs d'alertes. En effet, un pirate pourrait très bien rendre l'IDS complètement inefficace, en visant ces deux

composants. Pour sécuriser les sondes et les fichiers d'alertes, il est par exemple possible de mettre en place un réseau de management très contrôlé, avec son propre pare-feu et plusieurs mesures devront être prises pour assurer son fonctionnement tels la mise à jour régulière du système d'exploitation du senseur, la mis en place d'un système d'authentification robuste pour renforcer la sécurité, le changement régulier des mots de passe.

2.11 Normalisation des systèmes de détection d'intrusions

Pour décrire l'architecture globale d'un IDS, un comité du DARPA a défini quatre briques fonctionnelles qui vont composer de tels systèmes [35] :

Générateur d'événements (boîte E) : envoie des événements à la boîte A Analyseur d'événements (boîte A) : produit des alertes Base de données événementielle (boîte D) Système de réponse (boîte R) : réponse en temps réel face aux attaques Ce comité a aussi défini un langage de description des intrusions (CISL – Common Intrusion Specification Language) qui utilise des expressions verbales («ouvrir session», «effacer objet», etc.).

Cependant, ce langage n'a jamais été utilisé mais il a inspiré d'autres Comités, comme L'IDWG (Intrusion Detection Working Group) qui a effectué la plupart des travaux dans le domaine de la standardisation des IDS :

Norme IDMEF (Intrusion Detection Message Exchange Format) : définit le format des messages échangés dans un IDS. La norme IDMEF préconise une représentation en XML des messages. Dans chaque message, on retrouve l'ID de l'analyseur, le type d'alerte, l'emplacement (le nœud réseau), le jour e t l'heure, l'adresse, la classification de l'attaque, etc.

Protocole IDXP (Intrusion Detection eXchange Protocol) : procédures de transfert des messages entre les entités de l'IDS. l'IDWG a définit aussi un IDS comme étant un ensemble de plusieurs senseurs, analyseurs et managers. Cependant ce schéma reste théorique, et les IDS actuels sont rarement implémentés de cette façon (figure 2.4).

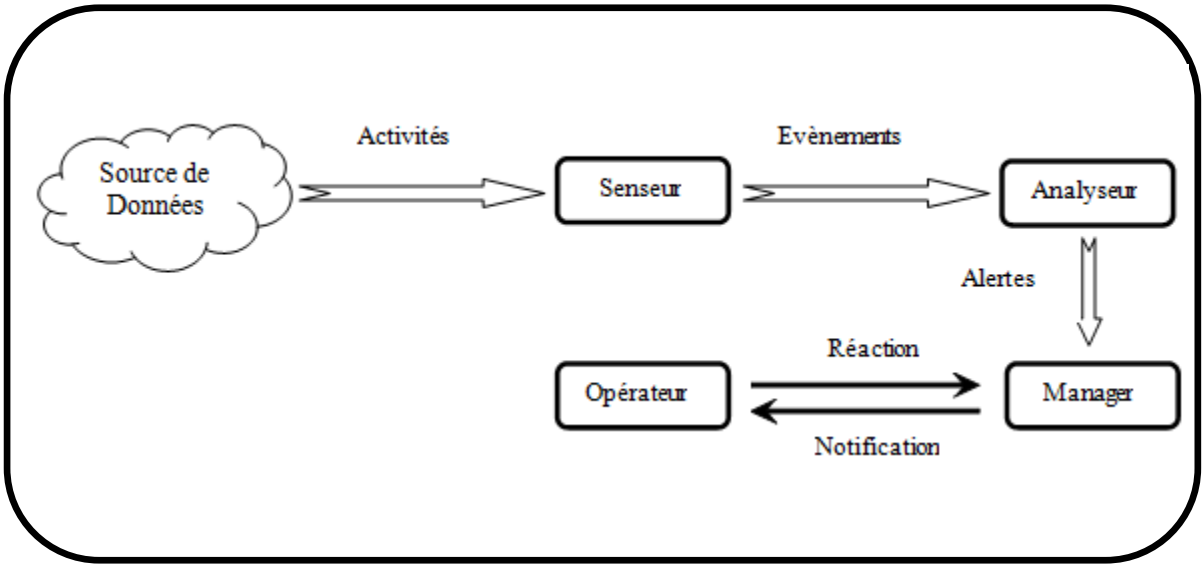


Figure 2.4 Schéma Standard d'un IDS Proposé par l'IDWG [35]

2.12 Testabilité des systèmes de détection d'intrusions

Malgré le fait que les systèmes de détection d'intrusions sont devenus les outils de défense omniprésents dans les systèmes informatiques d'aujourd'hui, jusqu'à présent on n'a aucune méthodologie complète et scientifiquement rigoureuse pour examiner l'efficacité de ces systèmes [63]. Dans cette section on va donner un ensemble de mesures partielles qui peuvent être utilisées pour tester le rendement des IDS. On va se concentré sur des mesures quantitatives relier à l'exactitude de la détection.

2.12.1 Couverture

Cette mesure détermine quelles attaques l'IDS peut détecter dans des conditions idéales. Pour les systèmes qui utilisent l'approche par scénarios, cela consiste simplement à compter le nombre de signatures et de les arranger sous un schéma d'appellation standard. Pour les autres systèmes, on doit déterminer quelles attaques hors de l'ensemble de toutes les attaques connues peuvent être détectées par une méthodologie particulière. Les dimensions qui composent chaque attaque rendent cette mesure difficile. En effet, chaque attaque à un but particulier, vise un logiciel particulier fonctionnant sur des versions particulières des systèmes d'exploitation ou contre un protocole particulier, et laisse des traces dans des endroits différents. Les attaques peuvent également dépendre du matériel du système objet. Dans ce qui va suivre on va donner quelques concepts qui rentre dans le cadre de cette mesure [64] [63].

1.12.2 Pertinence (Taux de Faux Positifs)

Cette notion concerne principalement le taux des faux positifs produits par un IDS dans un environnement donné pendant une période de temps particulière. Un faux positif, ou fausse alarme, est une alerte provoquée par le trafic en absence de l'attaque. Quelques causes pour les N-IDS incluent les signatures faibles; la détection des violations communes du protocole TCP. Elles peuvent également être provoquées par la surveillance et la maintenance du trafic générée par des outils de gestion du réseau. Il est difficile de mesurer les faux positifs, parce qu'un IDS peut avoir différents taux de faux positifs dans chaque environnement réseau, sachant qu'il n'existe pas de réseau « standard » . En outre, il est difficile de déterminer les aspects du trafic réseau ou les activités de l'hôte qui causeront de telles alertes. En conséquence, il est difficile de garantir qu'on peut produire le même nombre et type de fausses alarmes dans les tests des IDS comme dans le cas des vrais réseaux. Pour cela, il faux jouer sur la multitude des méthodes de configuration des IDS afin de réduire le taux des faux positifs. Sachant que cela rend difficile de déterminer quelle configuration d'IDS doit être utilisée pour un test particulier de faux positifs.

2.12.3 Complétude (Taux de Faux Négatifs)

Cette notion concerne le taux d'attaques détectées correctement par un IDS dans un environnement donné pendant une période particulière. En parle aussi des faux négatifs, c'est-à-dire l'occurrence d'attaque en l'absence d'alerte. La difficulté dans la mesure des taux de détection est que le succès d'un IDS dépend en grande partie de l'ensemble d'attaques utilisées pendant le test. En outre, la probabilité de la détection change avec le taux des faux positifs, et un IDS peut être configuré soit pour favoriser la capacité à détecter les attaques ou bien pour réduire au minimum les faux positifs. La figure 2.5 illustre ce problème d'adéquation par rapport à l'approche utilisée.

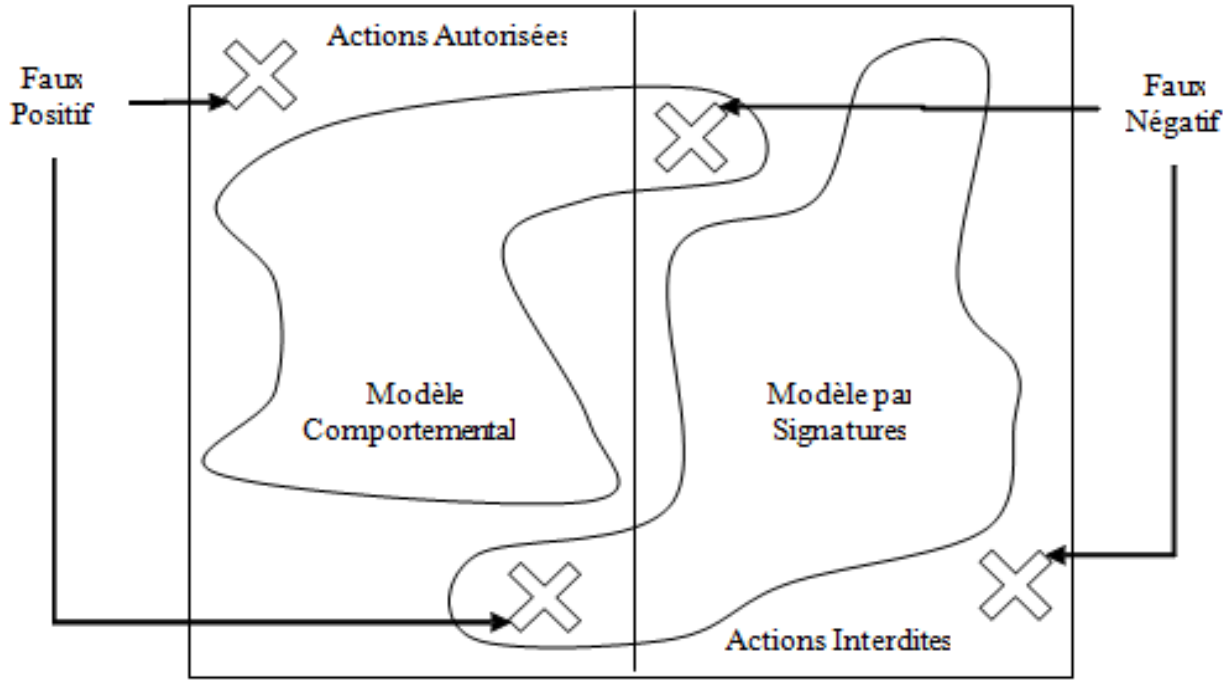


Figure 2.5. Adéquation des faux positifs/ négatifs par rapport à l'approche utilisée [2].

Une mesure qui rentre dans le cadre de la complétude d'un système est sa capacité à identifier une attaque, cette mesure montre à quel point un IDS peut identifier l'attaque qu'il a détecté en l'affectant à une catégorie, et en la marquant par un nom commun (de vulnérabilité) avec les autres attaques identiques.

Une autre mesure est la capacité de détecter les nouvelles attaques qui montre à quel point un IDS peut détecter les attaques qui n'ont jamais été produites avant. Pour les systèmes commerciaux, il n'est généralement pas utile de considérer cette mesure puisque leurs technologies basées généralement sur les signatures peuvent seulement détecter des attaques qui s'étaient produites précédemment (à quelques exceptions). Cependant, les systèmes basés sur la détection d'anomalie peuvent convenir à ce type de mesures. Généralement les systèmes qui permettent de détecter les nouvelles attaques produisent des faux positifs plus que ceux qui n'ont pas ce dispositif.

2.12.4 Résistance aux attaques

Cette mesure montre à quel point un IDS peut résister à la tentative d'un attaquant à perturber son fonctionnement correct. Parmi les formes des attaques contre un IDS on peut citer [2] :

L'envoie d'un grand volume trafic qui excède les capacités du traitement d'un IDS. Avec un

tel trafic à traiter, l'IDS peut planter et ne puisse pas détecter les attaques. L'envoie aux IDS des paquets dont l'objectif est de déclencher beaucoup de signatures, accablant de ce fait l'opérateur humain des IDS avec des faux positifs brisant ainsi les traitements d'alertes.
L'envoie à un IDS un grand nombre de paquets d'attaques d'identifications prévues pour distraire l'opérateur humain des IDS tandis que l'attaquant incite une vraie attaque cachée en dessous de "l'écran de fumées" créé par la multitude de ces attaques. L'envoie à un IDS des paquets contenant des données qui exploitent une vulnérabilité dans les algorithmes du traitement des IDS. De telles attaques seront réussies seulement si les IDS contiennent une erreur de programmation connue qui peut être exploitée par un attaquant malin.
Parmi les mesures qu'un IDS doit entreprendre pour remédier à ces attaques, le fait de pouvoir corréler les événements d'une attaque. Ces événements peuvent être recueillis d'IDS eux mêmes, des routeurs, des pare-feux, des applications logs, ou d'une large variété d'autres dispositifs. Cette corrélation permet aussi d'identifier les attaques de pénétration par étapes. Actuellement, les IDS ont seulement des aptitudes limitées dans ce domaine. Une autre mesure est la capacité de déterminer le succès d'une attaque qui est essentielle pour l'analyse de la corrélation et du scénario de l'attaque. Cela va également simplifier considérablement le travail d'un analyste en distinguant les attaques réussites qui sont les plus importantes et les attaques échouées habituellement moins préjudiciables.

2.12.5 Capacité de manipuler le trafic réseau

Cette mesure montre à quel point un IDS fonctionnera en présence d'un grand volume du trafic. La plupart des N-IDS commenceront à ignorer les paquets arrivés à mesure que le volume du trafic augmente, entraînant ces IDS à manquer certain pourcentage des attaques. Cette mesure est presque identique à « la mesure de résistance aux dénis de service » quand l'attaquant envoie un grand volume de trafic aux IDS. La seule différence est que cette mesure calcule l'aptitude de l'IDS à manipuler les volumes particuliers par rapport au fond normal du trafic.

Il est à noter aussi qu'un N-IDS peut être éludé par des versions furtives des attaques. Les techniques utilisées pour rendre les attaques furtives incluent la fragmentation des paquets, l'utilisation de divers types de codage des données en utilisant des drapeaux TCP inhabituels des paquets d'attaques cryptés, l'extension des attaques à travers des sessions multiples du réseau, et le lancement des attaques à partir des sources multiples [65]. Pour remédier à ces attaques, un N-IDS exige des capacités d'inspection dans des niveaux plus profonds des paquets du réseau. Par conséquent, il est important de mesurer les capacités d'un

N-IDS à capturer et à traiter, avec le même niveau d'exactitude, sous une charge réseau donnée, comme pour le cas d'un réseau passif. Pour cela, Par exemple, Hall [66] ont proposé une méthodologie de test et des métriques concernant le trafic pour normaliser les tests d'efficacité des N-IDS.

2.12.6 Autres Mesures

Il y a d'autres mesures de test de l'efficacité des IDS, telles que la facilité d'utilisation, la facilité de maintenance, les issues de déploiements, les besoins de ressources, la disponibilité et la qualité des supports, etc. Ces mesures ne sont pas directement liées aux performances de l'IDS mais peuvent être plus significatives dans beaucoup de situations commerciales.

2.13 Outils de Détection d'Intrusions

Depuis le milieu des années 80 plusieurs IDS, mettant en œuvre de nombreux travaux s'inspirant du modèle d'Anderson, ont été commercialisés. IDES (Intrusion Detection Expert System) était le premier IDS hybride regroupant l'approche comportementale et celle par scénarios, utilisant à la fois les méthodes statistiques et les systèmes experts [67]. Le prototype a été ensuite amélioré pour aboutir à NIDES (Next-generation IDES) [68]. NIDES et d'autres outils développés durant les mêmes années ont montré la possibilité d'utiliser les différents comportements d'utilisateurs sur une machine UNIX, pour détecter un éventuel essai d'intrusion. Cependant, si le comportement de l'utilisateur est trop riche, sa modélisation devient difficile et l'approche comportementale trouve sa limite.

A partir du milieu des années 90, la plupart des IDS commerciaux ont commencé à implémenter principalement l'approche par scénario, vu sa simplicité à être déployée, configurée et maintenue, sa pertinente théorique, avec la possibilité d'ajout, et de retrait des signatures. Historiquement, les premiers IDS utilisés étaient les H-IDS, destinés aux serveurs de calcul basés sur une architecture UNIX avec des utilisateurs essentiellement locaux. Les informations d'audit collectées, provenaient surtout de la machine elle-même. L'utilisation des N-IDS a vu le jour avec l'interconnexion des systèmes informatiques via Internet et l'insuffisance des sources d'audit locales pour détecter les nouvelles attaques réseau. Aujourd'hui, la plupart des IDS commerciaux sont des N-IDS.

Le tableau 2.1 montre quelques outils commerciaux et libres, ainsi que les approches de détection qu'ils utilisent. Dans ce qui va suivre, afin de bien illustrer les aspects qu'on a

montré précédemment, on va essayer de donner un aperçu sur le principe de fonctionnement de trois outils de détection d'intrusions.

	Système	Approches
H-IDS	Asax (Audit UNIX) Gassata (Audit UNIX) Sutekh	Par Scénarios
N-IDS	BlackICE BRO Diams Dragon ETrust (ex Session Wall) NetProwler NetRanger NFR Shadow Snort	Par Scénarios
	GrIDS	Comportementale
	Emerald	Comportementale+ Par Scénarios
IDS hybrides	Centrax (Audit NT) CyberCop Monitor (UNIX / NT) Intruder Alert (Audit UNIX) Real Secure (Audit NT)	Par Scénarios

Tableau 2.1 Quelques outils d'IDS commerciaux et libres

2.13.1 Sutekh

Sutekh est un H-IDS par signatures qui permet d'analyser les appels systèmes [69]. Son principe de fonctionnement repose sur un langage déclaratif de scénarios qui permet de décrire ce qui doit être détecté sans se préoccuper de la manière de le faire. Pour cela il utilise des filtres sur les événements à l'aide des connecteurs logico temporels entre ces derniers.

Soit A, B deux évènements, parmi les connecteurs logico temporels entre ces deux évènements : La séquence A then B, le choix A or B, l'ordre partiel A and B, la négation A if not B, la condition A such that B. Sachant que ce langage permet aussi une corrélation d'événements par unification de variables.

La compilation d'un scénario s'effectuera ensuite sous la forme d'un automate à états fini non déterministe qui va représenter la sémantique opérationnelle de l'IDS. Les états de cet automate représentent les étapes de reconnaissance d'une signature Sutekh et les arcs orientés sont étiquetés par des filtres sur les événements. La transition d'un état à un autre dépend de la satisfaction de l'événement courant, des contraintes exprimées par le filtre (valeurs des variables) et par les prédicats (connecteurs logico temporels) Cependant comme la plupart des H-IDS, Sutekh influe sur les performances des hôtes en surchargeant la CPU, la mémoire, ainsi que sur le système lui même a cause des collectes des appels. Sutekh présente aussi des problèmes de fonctionnement, comme la possibilité de l'existence d'instances multiples d'une signature dans une trace. Ensuite il y a le problème des classes d'équivalences entre les signatures [69] (deux signatures sont équivalentes si leurs variables ont les mêmes valeurs). D'où la nécessite d'une vigilance particulière pour éviter d'engendrer un grand nombre d'instances au cours de la reconnaissance.

2.13.2 GrIDS

GrIDS [70] est un système de détection d'intrusions utilisé pour les réseaux dont la taille est importante. Son principe de fonctionnement est basé sur l'utilisation des graphes. Pour cela une méthode pour construire des graphes d'activité du réseau a été proposée par les auteurs de ce système. Les nœuds d u graphe représentent les hôtes du réseau et les arêtes constituent les connexions entre ces hôtes. L'activité entre ces hôtes est constituée d'un ensemble de règles qui permet de décider quel trafic entre les hôtes va être représenté par l'agent de sécurité. Ces règles permettent aussi de calculer des attributs, comme les heures de connexion, qui vont être affectées aux arêtes du graphe.

Le paradigme organisationnel d'une hiérarchie des départements est la base de construction de l'activité du réseau. Un département est formé de plusieurs hôtes, les données d'audit sont combinées et centralisées pour générer le graphe d'un département, en se référant à l'ensemble de règles spécifiées. Les graphes de deux départements peuvent être combinés, selon les règles spécifiées, si des événements réseau d'un hôte du premier département impliquent un hôte du deuxième. Le graphe résultat sera constitué donc d'une nouvelle arête

qui spécifie le trafic entre ces deux départements. Une hiérarchie des départements sera construite en répétant ce processus.

Les règles jouent un rôle très important dans le système GrIDS, car c'est elles qui permettent d'abord de décider de combiner deux graphes en un graphe de niveau supérieur et de la manière de procéder et les actions à prendre quand les deux graphes seront combinés, ensuite définir la manière de calculer les attributs du graphe résultat. Comme les règles peuvent être très compliquées, GrIDS utilise aussi un langage qui permet de spécifier les politiques du comportement acceptable et inacceptable du réseau.

2.13.3 Snort

Snort est le N-IDS le plus populaire de nos jours, son principe de fonctionnement repose sur l'approche par scénarios. Snort est un produit Open Source (http://www.snort.org) ce qui lui permet d'avoir une communauté très importante d'utilisateurs et de contributeurs. Snort est logiquement divisé à des composants multiples. Ces composants fonctionnent ensemble pour détecter des attaques particulières et pour générer un rendu dans un format requis par le système de détection. La figure 2.6 montre comment ces composants sont arrangés.

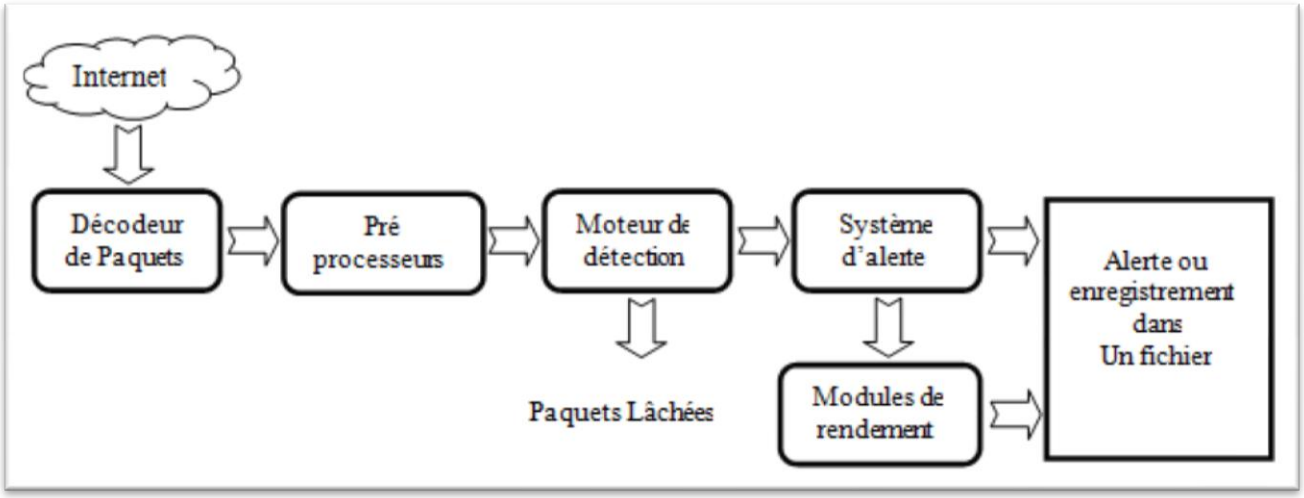


Figure 2.6 Composants de l'IDS Snort [2].

Dans ce qui va suivre on va montrer le rôle de chaque module :

Décodeur De Paquets : Il prend les paquets des différentes interfaces réseau et les prépare pour être prétraités ou pour être envoyés au moteur de détection. Les interfaces peuvent être : Ethernet, SLIP, PPP, etc.

Pré-processeurs : Sont des composants de connexion qui peuvent être employés avec Snort pour arranger les paquets de données avant de les envoyer vers le moteur de détection,

pour découvrir si ces paquets sont utilisés par un intrus. Quelques pré-processeurs effectuent également la détection en cherchant les anomalies dans les en-têtes des paquets et en générant des alertes si c'est le cas. Les pré-processeurs sont utilisés pour se protéger aussi contre les différentes techniques employées par les intrus pour duper les IDS, puisqu'ils permettent de défragmenter les paquets, décoder les URI HTTP, rassembler les flux TCP et ainsi de suite.
Ces fonctions sont une partie très importante dans un bon système de détection d'intrusions.
Moteur de détection : C'est la partie la plus importante de Snort. Son rôle est de détecter l'existence de n'importe quelle activité d'intrusion dans un paquet. Le moteur de détection utilise les règles Snort à cette fin. Les règles sont lues à partir de structures ou de chaînes de données internes où elles sont comparées avec tous les paquets. Si un paquet ressemble à n'importe quelle règle, une action appropriée est pris; autrement le paquet est lâché. Ces actions consistent à stocker le paquet sous le format approprié ou produire des alertes. Le moteur de détection est la partie critique en temps de Snort, dépendant de la puissance de la machine, la vitesse des bus internes est du nombre des règles définies, il peut prendre des durées différentes pour répondre aux différents paquets. Si le trafic réseau est important, Snort peux laisser tomber quelques paquets et ne peux pas obtenir de véritable réponse en temps réel.
Système d'alerte et d'enregistrement : Dépendant de ce que le moteur de détection trouve à l'intérieur d'un paquet, le paquet peut être employé pour enregistrer l'activité ou pour produire une alerte. Les enregistrements sont maintenus dans les fichiers textes simples, des fichiers de style tcpdump ou sous une autre forme. Les modules de rendement : Ces modules peuvent faire différentes opérations selon la façon dont l'utilisateur veut traiter le rendement produit par le système d'alerte et d'enregistrement de Snort. Selon la configuration établie, les modules de rendement peuvent :
-Envoyer des trappes SNMP ou des messages au syslog;
-Stocker des comptes rendus dans une base de données comme MySQL, Oracle ou bien simplement les stockés dans des fichiers avec une extension donnée.
-Produire des fichiers Extensible Markup Langage (XML).
-Modifier la configuration des routeurs et des pare-feux.
-Envoyer des messages bloquants du serveur (SMB) vers les machines fonctionnant sous Microsoft Windows. Sachant que d'autres outils peuvent également être utilisés pour envoyer les alertes sous d'autres formats tels que les E-mails ou la visualisation d'alertes en utilisant une interface web.

2.14 Les systèmes de détection d'intrusions distribuées

Les systèmes de détection d'intrusions distribuées [71] implémentent les SDIs traditionnels comme des senseurs à travers tout le réseau pour assurer la surveillance de tous les segments de ce réseau, chercher et déceler les attaques distribuées ainsi que les relations qui peuvent exister entre toutes les attaques. L'architecture distribuée peut être centralisée c'est-à-dire les rapports des différents SDIs sont remontés vers une seule unité d'analyse et de décision centrale qui assure le rôle de traitement des décisions élémentaires et de la décision finale sur la présence éventuelle d'une attaque distribuée. Elle peut être aussi décentralisée et dans ce cas le traitement et la décision sont assurés par plusieurs nœuds de décision. L'architecture distribuée offre une sécurité plus meilleure car si un nœud de décision tombe en panne la décision pourra être prise. La figure 2.7 illustre la structure d'un SDI distribué.

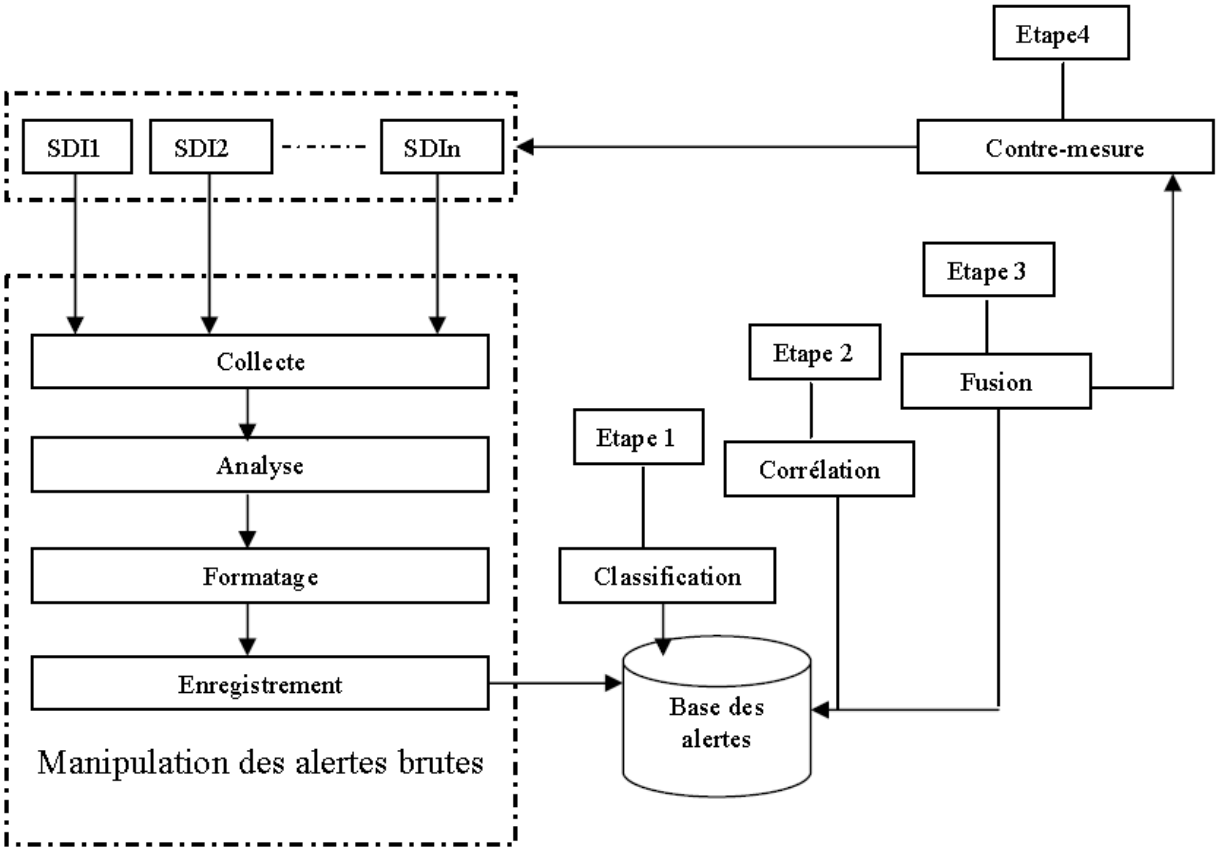


Figure 2.7. IDS distribué [71]

2.15 Conclusion

Dans ce chapitre, nous avons donné une vue détaillée sur les aspects concernant les IDS. On a vue que les IDS distribués offrent des avantages pour la détection des attaques distribuées, ainsi l'utilisation d'une méthode d'analyse comportementale avec celle par scénario permet de diminuer les faux positifs/négatifs. Pour cela, nous proposons une approche hybride basée sur les agents mobiles pour la distribution de la détection d'intrusions. Le chapitre suivant présente les éléments nécessaires à la conception d'une telle approche.

Chapitre 3
Agents mobiles et les systèmes de détection d'intrusions

3.1 Introduction

Dans le domaine des systèmes d'exploitation distribués, la question de la mobilité du code a été étudiée dans le contexte de la migration de processus pour traiter le problème de déplacement de processus d'un système d'une machine à un autre système. Ceci a été réalisé dans le but d'améliorer l'efficience du système en fournissant des capacités de chargement, et aussi pour permettre une reconfiguration dynamique du système afin qu'un processus puisse se déplacer vers une autre machine dans le cas où l'hôte courant va être arrêté. Une approche plus adaptée à ce genre de problème est étudiée dans le contexte de migration d'objets. Cette approche assure la mobilité des objets qui peuvent être réalloués à partir d'un espace d'adressage à un autre. Cette réallocation est utilisée pour une meilleure efficience, mais aussi permet d'avoir d'autres avantages tel que la migration des paramètres d'objets vers un site éloigné qui vont servir durant l'invocation de l'objet.
Un des obstacles majeurs pour l'utilisation réelles du code mobile est l'hétérogénéité des environnements d'exécution (matériel, système d'exploitation, logiciel installé etc.). Ceci a amené au développement de langages Script conçus pour résoudre ces problèmes tels que Perl, Tcl, Python etc. Ces langages Script sont exécutés dans un contexte qui interprète les commandes du langage Script. Ce qui les rend portables.
La multiplicité et la complexité des problèmes physiquement distribués et le besoin pressant de les résoudre en temps opportun et d'une manière efficace placent les systèmes agents mobiles parmi les solutions les plus perspicaces.

Les Systèmes agents mobiles sont placés sous la houlette de l'Intelligence Artificielle Distribuée (IAD) qui stipule que l'intelligence et l'expertise sont distribuées dans un groupe

d'entités autonomes, en interactions, qui travaillent synergiquement pour résoudre un problème.

L'idée est de répartir la charge d'un domaine initial sur des entités. La mobilité, la communication, la coopération, la négociation, la coordination et l'organisation entre ces entités permettent d'améliorer le degré de contribution de chacune d'elle dans la résolution du problème globale.

3.2 Définition d'un agent mobile

La mobilité est une propriété non obligatoire des agents. En effet, tous les agents ne sont pas mobiles. Un agent peut communiquer avec son environnement selon des moyens conventionnels comme par exemple RPC (Remote Procedure Calling). Ce type d'agents est dit stationnaire ou statique.

Un agent statique est un agent qui s'exécute seulement dans le système où il commence son exécution. S'il a besoin d'information non disponible dans le système ou a besoin d'interagir avec un agent dans un système différent, il utilise un mécanisme de communication tel que RPC [7].

Par contre, un agent mobile n'est pas lié au système dans lequel il débute son exécution. L'agent mobile est capable de se déplacer d'un hôte à un autre dans le réseau. Il peut transporter son état et son code d'un environnement vers un autre dans le réseau où il poursuit son exécution [3].

– **Les agents sont capables d'agir :** l'action repose, dans un système Multi-Agent, sur le fait que les agents accomplissent des tâches qui vont modifier l'environnement et par la suite modifier leur prise de décision.

– **Les agents sont autonomes :** les agents sont indépendants et ne subissent pas d'ordre. Ils sont conduits par leurs objectifs individuels qu'ils cherchent à satisfaire. Chaque agent a la liberté d'agir et de répondre aux requêtes des autres agents. Pour cela, il a besoin d'un certain nombre de ressources ou d'une mémoire locale propre à lui qui le rend indépendant de son environnement et des autres agents.

– **Représentation partielle de l'environnement :** les agents n'ont pas une vision globale de tout le système. D'ailleurs ils ne sont pas tous en relation. Un agent ne connaît en général que quelques agents qui forment ses connaissances appelés aussi accointances. C'est avec ce groupe d'agents qu'il communique et échange des informations.

– **Comportement d'un agent :** le comportement d'un agent consiste à communiquer et à agir afin de satisfaire ses besoins et ses objectifs à partir de tous les éléments dont il dispose (perceptions, représentation, actions, communications et ressources).

3.3 Les composants d'un modèle à agents mobiles

Les deux concepts fondamentaux d'un modèle à agents mobiles sont l'agent et son environnement appelé système d'agent [11].

3.3.1 L'agent

Un agent mobile est une entité qui possède cinq attributs : son état, son implémentation, son interface, son identifiant et son autorité. Quand un agent se déplace à travers le réseau, il transporte ses attributs.

L'état : Quand un agent voyage, il transporte avec lui son état, ceci lui permet de reprendre on exécution quand il arrive à destination. L'état d'un agent peut être considéré comme une photo instantanée de son exécution. Dans la plupart des langages de programmation, on peut partitionner cet état en un état d'exécution (qui inclut ses programmes et ses interfaces) et un état d'objet (qui contient les valeurs des variables d'instance dans un objet).

L'implémentation : Comme n'importe quel autre programme, l'agent mobile a besoin d'un code pour pouvoir s'exécuter. Quand il se déplace à travers le réseau, l'agent peut soit emporter son code, soit, aller à destination, voir quel code est disponible sur la machine distante et récupérer le code manquant à partir du réseau (c'est la technique du "cod-on-demand").

L'implémentation d'agent doit être à la fois exécutable et sans risque pour l'hôte de destination. Les langages interprétés offrent une indépendance par rapport aux plates formes ainsi qu'un environnement d'exécution contrôlé qui caractérise le mécanisme de sécurité puisqu'il limite l'accès aux ressources privées de l'hôte.

L'interface : Un agent fournit une interface qui permet aux autres agents et aux autres systèmes d'interagir avec lui. Cette interface peut être un ensemble de méthodes qui permet aux autres agents et applications d'accéder aux méthodes de l'agent par un système de messagerie.

Le langage KQML (Knowledge Query and Manipulation Language) [3] est un langage et protocole de communication entre les agents. KQML offre une variété de types de messages, qui exprime une attitude concernant le contenu de l'échange. Ce langage permet aussi d'assister les agents dans le traitement de leurs requêtes.

L'identifiant : Chaque agent possède un identifiant unique durant son cycle de vie, qui lui permet d'être identifié et localisé. Puisque l'identifiant est unique, il peut être utilisé comme clé dans les opérations qui exigent un moyen pour référencer une instance particulière d'agents.

L'autorité : Une autorité est une entité dont l'identité peut être authentifiée par n'importe quel système auquel elle essaye d'accéder. Une autorité peut être soit une personne privée, soit une organisation. L'identité est constituée d'un nom et d'autres attributs. Pour les agents, il existe principalement deux types d'autorité :

– Le fabriquant (manufacturer) qui est le fournisseur du code d'implémentation de l'agent,

– Le propriétaire (owner) qui a la responsabilité du comportement de l'agent.

3.3.2 Le système d'agent

Un système d'agent (aussi appelé serveur d'agent) est un environnement qui est capable de créer, interpréter, exécuter, transférer et arrêter un agent. De la même manière qu'un agent, un système d'agent est associé à une autorité qui identifie la personne ou l'organisation pour laquelle il agit. Un système d'agent est identifié par son nom et son adresse. Une machine hôte peut contenir plusieurs systèmes d'agent. Cinq concepts jouent un rôle important dans le système d'agent :

La place : Un agent mobile se déplace d'un environnement d'exécution à un autre. Cet environnement est appelé " place ". Une place est un contexte au sein d'un système d'agent, dans lequel un agent s'exécute. Ce contexte peut fournir un ensemble de services uniformes sur lesquels l'agent peut compter indépendamment de sa localisation spécifique.

La place de départ et la place de destination peuvent être situées au sein d'un même système d'agent ou sur des systèmes d'agents différents. Un système d'agent peut contenir une ou plusieurs places.

Le type d'un système d'agent : Le type d'un système d'agent permet de définir le profil d'un agent. Par exemple, si le type d'un système d'agent est " AGLET ", alors le système d'agent est implémenté par IBM et supporte Java comme langage de programmation.

Les ressources : Le système d'agent et la place fournissent un accès contrôlé aux ressources locales et aux services (base de données, processeurs, mémoire, disques).

Localisation : La localisation est un concept important pour les agents mobiles. On définit la localisation d'un agent comme étant la combinaison de la place dans laquelle il s'exécute et l'adresse réseau du système d'agent où réside la place. Concrètement, elle est définie par l'adresse IP et le port d'écoute du système d'agent avec le nom de la place comme attribut.

Région : Une région est un ensemble de systèmes d'agent qui appartiennent à la même autorité mais qui ne sont pas nécessairement de même type. Le concept de région permet à une autorité d'être représentée par plusieurs systèmes d'agent.

3.3.3 La création et la mort des agents mobiles

Un agent est créé dans une place. La création peut être initiée soit par un autre agent résidant dans la même place ou par un agent ou un système non_agent en dehors de la place. Le créateur doit s'authentifier dans la place et établir les autorisations et les références que va posséder l'agent. Le créateur peut aussi définir des arguments d'initialisation pour l'agent.
La classe de définition nécessaire pour instancier l'agent peut résider dans l'hôte local ou sur une machine distante ou fournie par le créateur. La création se fait en trois étapes :

- **L'instanciation et l'attribution de l'identificateur :** Le code de la classe agent est chargé puis exécuté. L'objet agent est instancié. La place assigne un identificateur unique à l'agent.
- **L'initialisation :** L'agent peut s'initialiser en utilisant les arguments d'initialisation fournis par son créateur. Quand l'initialisation se termine, l'agent est complètement installé dans la place.
- **L'exécution autonome :** Après l'installation, l'agent peut commencer son exécution. Il est maintenant capable de s'exécuter indépendamment des autres agents dans la même place.

Dans la plupart du temps, l'agent est tué lorsqu'il quitte sa place. Ce processus peut être initié par l'agent lui même, ou par un autre agent ou système non-agent résidant ou non dans la même place. L'agent peut aussi être renvoyé de sa place par le système pour l'une des raisons suivantes :

- La fin de son " temps de vie ",
- L'agent n'est pas utilisé ou référencé,
- Violation des règles de sécurité,
- Le système est arrêté.
- Le processus de destruction est amorcé en deux étapes :
- La préparation : une chance est donnée à l'agent de finir sa tâche courante avant d'être renvoyé.
- Suspension de l'exécution : la place suspend l'exécution de l'agent.

3.3.4 Le transfert des agents mobiles

Le processus de transfert peut être initié par l'agent lui même, par un autre agent résidant dans la même place ou par un agent ou système non agent résidant dans une autre

place. L'agent est ensuite transféré de sa place courante, pour être reçu par la place spécifiée (destination).

La place d'origine et la place destination gèrent le processus de transfert. Quand la place d'origine contacte la place de destination, celle-ci peut soit réaliser le transfert, soit envoyer un message d'échec à la place d'origine. Si la place d'origine ne peut pas contacter la place destination, elle doit retourner un message d'échec à l'agent.

Le processus de transfert va être décrit dans ce qui suit [7] :

– **L'envoi de l'agent :** Quand l'agent mobile se prépare pour son voyage, il doit être capable d'identifier sa destination. Si la place est non spécifiée, l'agent va s'exécuter dans la place par défaut sélectionnée par le système d'agent de destination. Une fois la localisation de la destination est établie, l'agent mobile informe le système d'agent local qu'il veut se transférer au système d'agent de destination. Ce message est retransmis via un API interne entre l'agent et le système d'agent.

Quand le système d'agent reçoit la requête de transfert, il doit :

– Suspendre l'agent : l'agent est d'abord prévenu à propos de son transfert, et un temps lui est alloué pour terminer sa tâche courante. Une fois ce temps écoulé, son thread d'exécution est arrêté.

– Sérialiser l'agent : l'agent (composé de son état et de la classe agent) est sérialisé par un moteur. La sérialisation est le processus de création d'une représentation persistante de l'objet agent qui peut être transportée à travers le réseau. La sérialisation de l'agent peut inclure son état d'exécution.

– Encoder l'agent sérialisé : le moteur encode l'agent sérialisé pour le protocole de transport choisi.

– Transférer l'agent : le moteur établit une connexion réseau à l'hôte spécifié et transfère l'agent sérialisé et encodé.

– **Réception de l'agent :** Avant que le transfert puisse avoir lieu, l'origine de l'agent doit s'authentifier auprès du moteur de réception. Lorsque l'authentification est réussie, le processus de transfert peut alors commencer :

– Réception de l'agent : l'agent encodé est reçu.

– Décoder l'agent.

– Désérialiser l'agent : la représentation persistante de l'agent est désérialisée.

La classe agent est instancié et l'état de l'agent transféré est rétabli.

– Reprendre l'exécution de l'agent : l'agent "recrée" est notifié dans sa place destination. Un nouveau thread d'exécution lui est assigné.

– **Le transfert de la classe d'agent :** L'agent ne peut pas reprendre son exécution sans que ses classes ne soient présentes. Il y a plusieurs manières de mettre ces classes à la disposition du moteur de destination :

– *La classe existe à destination :*

Si la classe est déjà disponible dans l'hôte éloigné -soit dans la mémoire cache ou dans un fichier système local, alors il n'est pas nécessaire de la transférer. L'agent transféré n'a besoin que des informations requises pour identifier la classe, tel que le nom complet de la classe. Il peut aussi contenir des informations supplémentaires qui décrivent la localisation de la classe et sa définition.

– *La classe existe à l'origine :*

Si la classe est localisée à l'origine, comme il est souvent le cas, elle peut être facilement transportée avec l'état de l'agent au moteur destination.

– *Le code à la demande :*

Dans ce cas, la classe est disponible à partir d'un serveur et le moteur de destination peut la retirer en appliquant le principe du code à la demande.

Cependant, notons que le moteur de destination doit, dans ce cas, réaliser une connexion réseau supplémentaire pour retirer la classe.

Après l'instanciation de l'agent, celui-ci peut créer d'autres objets. Il est évident que les classes de ces objets sont nécessaires pour leur instanciation et pour la continuation de l'exécution. Si l'une de ces classes, n'est pas disponible dans le moteur de destination, elle doit être transférée soit à partir de l'origine où à partir du moteur qui a lancé l'agent.

3.3.5 La communication entre les agents mobiles

Les agents peuvent communiquer avec d'autres agents résidents dans la même place ou avec des agents résidents dans d'autres places. Un agent peut évoquer une méthode d'un autre agent comme il peut lui envoyer des messages s'il est autorisé à le faire. La communication inter-agents peut suivre trois schémas différents :

– Now-type messaging : c'est le type de messagerie le plus utilisée. C'est un type synchrone. Il bloque l'exécution de l'émetteur du message jusqu'à ce que le receveur aura complètement téléchargé le message et aura envoyé sa réponse.

– Future-type messaging : c'est un type de messagerie asynchrone qui ne bloque pas l'exécution. L'expéditeur retient une variable qui peut être utilisée pour obtenir le résultat. Ce type de messagerie est utile en particulier quand plusieurs agents communiquent ensemble.

– One-way-type messaging : c'est un type asynchrone qui ne bloque pas l'exécution courante. L'expéditeur ne va pas retenir une variable pour ce message et le receveur ne va jamais répondre. Ce type est utile quand deux agents engagent une conversation où l'agent expéditeur n'a pas besoin de la réponse de l'agent receveur. Ce type peut aussi être appelé fire-and-forget.

3.4 Approches basées agents mobiles pour la détection d'intrusions

Plusieurs domaines de recherche dans les Mobile Agents Intrusions Detection System (MAIDS) ont été inaugurés par [49]. Certains sont mis en œuvre dans [6], [50], [51], [52] ou [53]. Dans ce qui suit une réponse à la question: En quoi est-ce que les agents mobiles peuvent être utiles dans le domaine de la détection d'intrusions? :

– **Architecture résistante aux attaques:** C'est souvent pour un gain en performance qu'on a recours à une architecture hiérarchique. Cette conception a souvent plusieurs lignes de communication non redondantes. Toutefois, un intrus peut désactiver une branche de l'IDS. Les agents mobiles peuvent solutionner ce problème :

– Une architecture entièrement distribuée

– Les agents mobiles ont la capacité de se déplacer quand une activité suspecte est détectée,

– Un agent peut remplacer chaque nœud dans une architecture hiérarchique et remédier à une perte de fonctionnalité.

Par ailleurs, un IDS peut ne pas détecter immédiatement la présence d'un intrus. Ce dernier peut profiter de cette situation pour effacer ses traces et neutraliser l'IDS. Un système de détection des intrusions à base d'agents mobile reste vulnérable à cela mais se trouve néanmoins pourvu de quelques avantages. En effet, l'agent transporte du code non encore altéré, une fois arrivé sur un hôte il pourrait par exemple tester l'intégrité de la plateforme IDS locale.

– **Détection multi-point:** Il s'agit d'analyser les flux d'audit en provenance de plusieurs hôtes pour détecter des attaques distribuées ou autres stratégies d'attaques d'un réseau dans sa globalité. Il est quasiment impossible de faire supporter tous les flux d'audit à un IDS central, et même dans ce cas, la détection est difficile.

Les agents mobiles nous permettent de faire le calcul distribué. L'idée est de faire transporter l'analyseur vers les flux d'audit et non les flux d'audit vers l'analyseur.

Les agents pourraient dépister les attaques, les corréler, et découvrir les plans d'attaques distribuées.

– **Agents errants:** Chaque agent effectue un test spécifique et peut se déplacer aléatoirement d'hôte en hôte. Quand des tests indiquent la possibilité d'une intrusion, l'agent se déplace vers le site suspect pour effectuer des tests plus précis.

– **Partage de connaissances:** Le partage des connaissances est l'une des principales caractéristiques des agents mobiles. Ceci peut être utile dans plusieurs systèmes de détection des intrusions différents fonctionnent sur un site et que pour des raisons de pertinence ils partageraient les informations sur les attaques récentes pour améliorer la détection d'attaques futures.

3.5 Présentation de quelques travaux récents dans les IDS distribués

Les agents mobiles ont été proposés comme une technologie pour la distribution de la détection d'intrusions récemment par plusieurs travaux [75] [76][77][78][79]. Les raisons ont été discutées dans la section précédente. Dans ce qui suit, on va présenter quelques travaux récents dans l'utilisation des agents mobiles pour la distribution du système de détection d'intrusions.

3.5.1 AAFID : Autonomous Agent for Intrusion Detection

Le système AAFID traite le problème de la détection d'intrusions sous un angle selon [81]. Au lieu de concevoir un système de détection d'intrusions monolithique, il propose une architecture distribuée, où plusieurs petits processus indépendants opèrent de manière coopérative pour assurer la surveillance du système cible. Ses principaux avantages sont : son efficacité, sa tolérance aux fautes, sa résistance aux dégradations et son extensibilité.

L'architecture d'AAFID est composée de trois entités essentielles :

- les ***agents***,
- les ***"transceivers"***,
- les ***"monitors"***.

Un ***agent*** est une entité d'exécution indépendante qui surveille certains aspects d'un host et reporte les comportements anormaux ou suspicieux au *"transceiver"* approprié. Sur un host, il peut y avoir un ou plusieurs agents. Les agents n'ont aucune autorité pour générer les alarmes et ne peuvent pas communiquer entre eux.

Les ***"transceivers"*** représentent l'interface de communication externe d'un host. Il y en a un pour chaque host surveillé. Un *"transceiver"* a deux rôles importants :

- un rôle de *contrôle* : il doit lancer et stopper les agents s'exécutant sur son host, garder une trace de ces agents et répondre aux commandes envoyées par ses *"monitors"* ;
- un rôle de *traitement de données* : il doit recevoir et analyser les rapports générés par les agents s'exécutant sur son host, puis envoyer les résultats de son analyse à un ou plusieurs *"monitors"*.

Quant aux ***"monitors"***, ils représentent l'entité de plus haut niveau dans l'architecture de AAFID. Ils ont eux aussi un rôle de contrôle et un rôle de traitement de données similaires à ceux des *"transceivers"*. La différence entre les *"monitors"* et les *"transceivers"* est qu'un *"monitor"* peut contrôler des entités qui s'exécutent sur différents hosts alors qu'un *"transceiver"* ne peut contrôler que les agents locaux. Dans leur rôle de contrôle, les *"monitors"* peuvent :

- recevoir des instructions d'autres *"monitors"*,
- contrôler des *"transceivers"* et d'autres *"monitors"*.

Par contre, pour ce qui est de la tâche de traitement de données, les *"monitors"* font une corrélation des informations reçues des *"transceivers"* qu'ils contrôlent. Ils détectent ainsi des événements qui impliquent différents hosts. De plus, ils peuvent communiquer avec une interface utilisateur et fournir ainsi un point d'accès au système AAFID.

3.5.2 *MSAIDS – Multi-Level and Secured Agent-based Intrusion Detection System*

L'architecture *MSAIDS* [80] fournit une méthodologie où l'intrusion est faite à deux niveaux. Le premier est le niveau de détection inférieur (LLD) qui a les agents de données et les agents de traitement. Les agents de données se déplacent entre les nœuds dans le réseau pour rassembler les informations associées. Les deux agents de traitement aussi connus comme agents nœud où agent Nœud 1 est responsable de la construction de la base de données du premier niveau depuis les informations rassemblées et les données de classification et de formation. L'agent nœud 2 est responsable du *data mining* et en première détection de l'intrusion et communique la possibilité d'intrusions à l'agent de l'interface à travers l'agent de l'alarme.

Le niveau de détection supérieure (ULD) aussi connu comme niveau de la confirmation est impliqué dans la séparation du processus de la détection d'intrusion. l'ULD, les agents du niveau inférieur assemblent le données des agents du données et informent le Contrôleur et dispositif de protection (CP) qui actes comme l'agent de l'Animateur au sujet de la nature des données assemblées. Le CP assure aussi la communication adéquate et la distribution de

service entre les agents. Les données assemblées sont utilisés pour mettre à jour la base de données ULD alors; l'ULD ne vérifie pas pour l'intrusion s'il n'y a aucun signal du LLD.

3.5.3 DSCIDS: Distributed Soft Computing for Intrusion Detection System.

Ce travail [4] est basé sur une architecture hiérarchique avec analyseur central et Contrôleur (CAC) comme un cœur du DIDS (Distributed Intrusion Detection System). Le CAC consiste en une base de données et webserver qui permet une interrogation interactive par l'administrateur du réseau pour habituellement des renseignements / analyse de l'attaque et qui initier des mesures préventives. CAC exécute aussi l'agrégation de l'attaque, construction des statistiques, identifier les modèles d'attaque et permettre l'analyse de l'incident rudimentaire.
Les auteurs ont testé le modèle en utilisant des différentes techniques entre autres : le réseau neuronal, le système de déduction flou, raisonnement approximatif et les techniques de l'optimisation libres dérivatives sur un ensemble de données KDD. Les expériences ont trois phases à savoir: réduction de données, phase de formation et phase de teste. Dans la phase de la réduction de données, les variables importantes pour la découverte de l'intrusion en temps réel sont sélectionnées par sélection du trait. Dans la phase de la formation, les différentes techniques de l'informatique sont construites en utilisant les données de la formation. Les données de teste sont utilisées à partir du modèle de formation pour détecter des intrusions dans la phase de teste.

3.5.4 IMA-IDS- Intelligent and Mobile Agent for Intrusion Detection System

L'architecture [3][13] implique quatre agents qui sont multipliés par clones à savoir: Agent collecteur, agent correlateur, agent analyseur et l'agent directeur ont chargé des taches suivantes:

Le Manager : L'Agent Manager est au centre de ce système distribué de détection d'intrusions. Il a à sa charge la création, l'activation et la distribution à travers le réseau des agents collecteurs (Agent Collector), analyseurs (Agent Analyser) et corrélateurs (Agent Correlator).

Outre la gestion des agents qui composent l'architecture IMA-IDS, l'Agent Manager supervise leur bon fonctionnement, décide de leur création et de leur arrêt, leur communique des messages selon le besoin et supervise leur itinéraire.

L'Agent Collector : L'Agent Collector est capable de se mettre à l'écoute du réseau, de rapporter toutes les occurrences d'événements et de les transmettre périodiquement à l'entité

centre de notre modèle qui est l'Agent Manager. L'Agent Collector assigne des priorités aux événements qu'il collecte et transmet les plus prioritaires à l'Agent Correlator.

L'Agent Correlator : Le rôle de ce dernier est de dépêcher les événements importants susceptibles de faire partie d'un scénario d'intrusion (événements critiques) et de les transmettre en temps opportun à l'entité Agent Analyser appropriée. Notons que chaque type d'événement critique est spécifique à un Agent Analyser particulier.

L'Agent Analyser : Les agents analyseurs traitent les informations qu'ils reçoivent, appliquent des corrélations et des analyses de haut niveau (analyse par signature, analyse comportementale, analyse des protocoles de sécurité basée sur les interprétations abstraites) pour détecter les événements intrusifs qu'ils signaleront à l'agent Manager.

3.5.5 MAD-IDS: Novel Intrusion Detection System using Mobile Agents and Data Mining Approaches

Le système [15] intègre les algorithmes de *data mining* et la technologie de l'agent mobile dans le but de détection d'attaques connues et inconnues dans le réseau. Ce système emploie la technologie d'agent mobile pour le traitement des informations à partir de chaque hôte. En effet, la détection basé signature par agent mobile permet la détection des attaques connues. De l'autre côté, l'intégration des deux méthodologies d'agent mobile et la technique *data mining*, la détection comportementale par un agent fournit l'avantage de détecter de nouvelles attaques inconnues. Le modèle résultant est plus efficace depuis que c'est capable à détecter des genres connus et nouveaux attaques dans un environnement distribué. Par conséquent, l'intégration de la technologie de l'agent mobile et les techniques *data mining* font un IDS plus autonome et effectif.

Le tableau 3.1 donne un résumé sur les IDS distribués présentés ci-dessus.

	Approche	Source de données	Sécurité d'agents mobiles	Technique
AAFID (1998)	Scénarios	audit Unix	N'est pas discutée	Algorithmes génétiques
MSAIDS(2006)	Détection par scénarios	Basé réseau	Se base sur la plate forme Aglet	Algorithmes modifiés préalablement
DSCIDS(2007)	Hybride	Basé réseau	N'est pas discutée	Paradigme de calcul logiciel
IMA-IDS (2003)	comportementale	Hybride	Se base sur la plate forme Aglet	Fonctions comportementales et statistiques
MAD-IDS(2010)	Comportementale et par scénarios	Basé réseau	N'est pas discutée	Data mining

Tableau 3.1 Résumé des IDS distribués

3.6 Les limites des systèmes de détections existants

Mis à part les inconvénients déjà évoqués et qui concernent les approches de détection d'intrusion, beaucoup d'IDS existants (NIDS et HIDS) sont faits d'un seul bloc ou module qui se charge de toute l'analyse. Ils réalisent leur collecte de données ainsi que leur analyse en utilisant une architecture centralisée. Ces systèmes monolithiques exigent beaucoup de données d'audit, ce qui consomme beaucoup de ressources de la machine à protéger, pose des problèmes de mise à jour et font de ce point central d'analyse une cible d'attaque propice (Si un intrus parvient à le faire compromettre alors la totalité du réseau se retrouve sans protection).

En somme les majeurs problèmes issus de cette architecture sont [3] :

– Il est généralement difficile de mettre à jour des profils ou de signatures d'attaques. De plus, les systèmes de détection des intrusions demandent de plus en plus de compétence à celui qui administre le système de sécurité.

– La flexibilité du réseau est limitée: Le calcul de toutes les informations sur une seule station implique des limites sur la taille du réseau à observé. Au-delà de cette limite, l'analyseur central devient incapable de gérer le flot d'informations.

La collecte de données peut également engendrer des problèmes lors de trafics excessifs sur le réseau.

– Il est difficile de mettre à jour ces IDS: Les changements et l'ajout de possibilités sont habituellement effectués en éditant un fichier de configuration et cela en ajoutant une entrée dans une table ou en installant un nouveau module. L'IDS nécessite habituellement d'être redémarré afin de prendre en compte ces changements.

– L'analyse des données du réseau peut être imparfaite [65]: réaliser la collecte de données d'un réseau ailleurs que sur la station destinée à les recevoir peut offrir à des intrus la possibilité d'attaques dites d'insertion ou d'évasion.

Ceux-ci se servent des failles dans les piles de protocoles du réseau de différents centres serveurs pour dissimuler des attaques ou des dénis-de-service.

Afin de résoudre la plupart des problèmes mentionnés ci-dessus, d'autres types d'IDS ont été conçus. Les tendances vont vers la détection des intrusions distribuées.

Un système hiérarchique est décrit dans [82]. [10] décrit un mécanisme sans autorité centrale. Le point culminant s'est constitué en 1993 avec le projet NADIR [5] qui propose des audits de sécurité via une collecte d'informations distribuée qui sera analysée par un système expert.

3.7 Conclusion

Il existe une panoplie de raisons qui justifient le choix d'utiliser les agents mobiles sont d'après [11], nous en retiendrons les suivantes:

– Réduire la charge des réseaux,

– Surmonter le temps de latence sur le réseau,

– S'exécuter de manière asynchrone et autonome,

– S'adapter dynamiquement aux environnements d'exécution,

– Hétérogènes,

– Robustes et tolérants.

D'où découle la finalité de ce travail de recherche: pouvoir impliquer les agents mobiles dans le domaine de distribution de la détection d'intrusions.

Dans le chapitre suivant, nous allons décrire les détails de l'approche proposée.

Chapitre 4
Une approche hybride basée agents mobiles pour un système de détection d'intrusions distribué

4.1 Introduction

La gestion de la sécurité d'un réseau ou d'un système implique :

1. la mise en place des mécanismes de sécurité préventifs pour protéger les données et ressources du système ou réseau contre tout accès non autorisé ou abusif ;
2. le déploiement des outils de sécurité pour détecter les attaques qui réussiraient à porter atteinte à la sécurité du réseau ou système malgré les mesures préventives ;
3. la mise en place des mécanismes de réponse aux attaques détectées.

Prévenir de toutes les violations de sécurité apparaît quelque peu irréel. En effet, il est pratiquement impossible d'avoir un réseau complètement sûr et de le protéger contre toutes les attaques possibles. Malgré la mise en place de politiques préventives de sécurité, les réseaux et systèmes restent sujets à des attaques potentielles entreprises par des personnes qui réussissent à contourner ces mesures préventives par des comportements frauduleux.

Pour évaluer et assurer les besoins de sécurité d'une entreprise, il faut considérer trois aspects:

- les services de sécurité,
- les attaques de sécurité,
- les mécanismes de sécurité pour la prévention, la détection des attaques et la réponse à ces attaques.

Dans le cadre de cette thèse, nous nous intéressons au troisième aspect : les mécanismes de détection d'intrusions qui représentent les outils de sécurité les plus récents.

Nous allons donc dans ce chapitre, commencer par motiver l'approche proposée, et donner ses objectifs. Puis nous donnons l'architecture générale du modèle proposé tout en présentant

le schéma structurel et fonctionnel du système. Ensuite nous expliquons la méthode d'analyse utilisée et le type de communication entre les différents agents du système.

4.2 Motivations

L'évolution des réseaux, en termes de nombre d'utilisateurs et de services, les rend toujours plus complexes et par conséquent vulnérables à de nouveaux types d'attaques. Les systèmes de détection d'intrusions existants ne sont pas facilement adaptables à cette complexité croissante des interactions et des comportements utilisateurs. Un autre facteur important est que le traitement des données d'une manière centralisée induit des vulnérabilités dans le système informatique, en l'occurrence, des réductions de performances en termes de scalabilité, configurabilité et de tolérance aux pannes. En effet, il suffit que l'engin central soit défectueux pour avoir une défection de tout le système de détection des intrusions. En outre, un grand flux d'événements peut entraîner un ralentissement du temps de traitement des données, une réaction tardive du système, une surcharge du réseau voire même des pertes de données rendant toute analyse biaisée. Notons aussi que la centralisation freine l'extensibilité du système et rend sa reconfiguration difficile.

L'utilisation des agents mobiles pour la détection d'intrusion dans un système distribué offre plusieurs avantages (voir le paragraphe 3.4 du chapitre 03).

Ainsi, il existe une panoplie de raisons qui justifient le choix d'utiliser les agents mobiles sont [2][3]:

- Réduire la charge des réseaux,
- Surmonter le temps de latence sur le réseau,
- S'exécuter de manière asynchrone et autonome,
- S'adapter dynamiquement aux environnements d'exécution,
- Hétérogènes,
- Robustes et tolérants

L'un des avantages des agents mobile est la détection multi-point: Il s'agit d'analyser les flux d'audit en provenance de plusieurs hôtes pour détecter des attaques distribuées ou autres stratégies d'attaques d'un réseau dans sa globalité. Il est quasiment impossible de faire supporter tous les flux d'audit à un IDS central, et même dans ce cas, la détection est difficile. Les agents mobiles nous permettent de faire le calcul distribué. L'idée est de faire transporter

l'analyseur vers les flux d'audit et non les flux d'audit vers l'analyseur. Les agents pourraient dépister les attaques, les corréler, et découvrir les plans d'attaques distribuées.

Dans cet esprit, nous nous proposons de développer un modèle et un prototype quant à l'utilisation de la technologie des agents mobiles dans la détection des intrusions.

4.3 Objectifs de l'approche

L'approche que nous proposons a pour objectif de remplir les fonctions nécessaires à la détection des attaques de sécurité. Pour réaliser cela, notre solution doit fournir les points suivants :

• une approche hybride pour détecter les attaques qui se caractérisent par des signatures connues ou inconnues ;
• un modèle collaboratif permettant la détection des attaques de nature distribuée par la corrélation d'alertes;
• un modèle utilisant la technologie d'agents mobiles pour distribuer la détection d'intrusions.
Dans ce qui suit on va présenter les détails de cette approche.

4.4 Le modèle proposé : une approche hybride à base d'agents mobiles pour un IDS distribué

Dans cette section nous allons présenter notre modèle de détection d'intrusion. Premièrement nous décrivons l'architecture générale (Figure 4.1), et nous détaillons les entités composantes de notre système de détection d'intrusion. Puis nous donnons quelques informations de nos agents et les objets manipulés par ces mêmes agents. Nous donnons aussi leurs diagrammes de classe selon le model AUML [12][73].

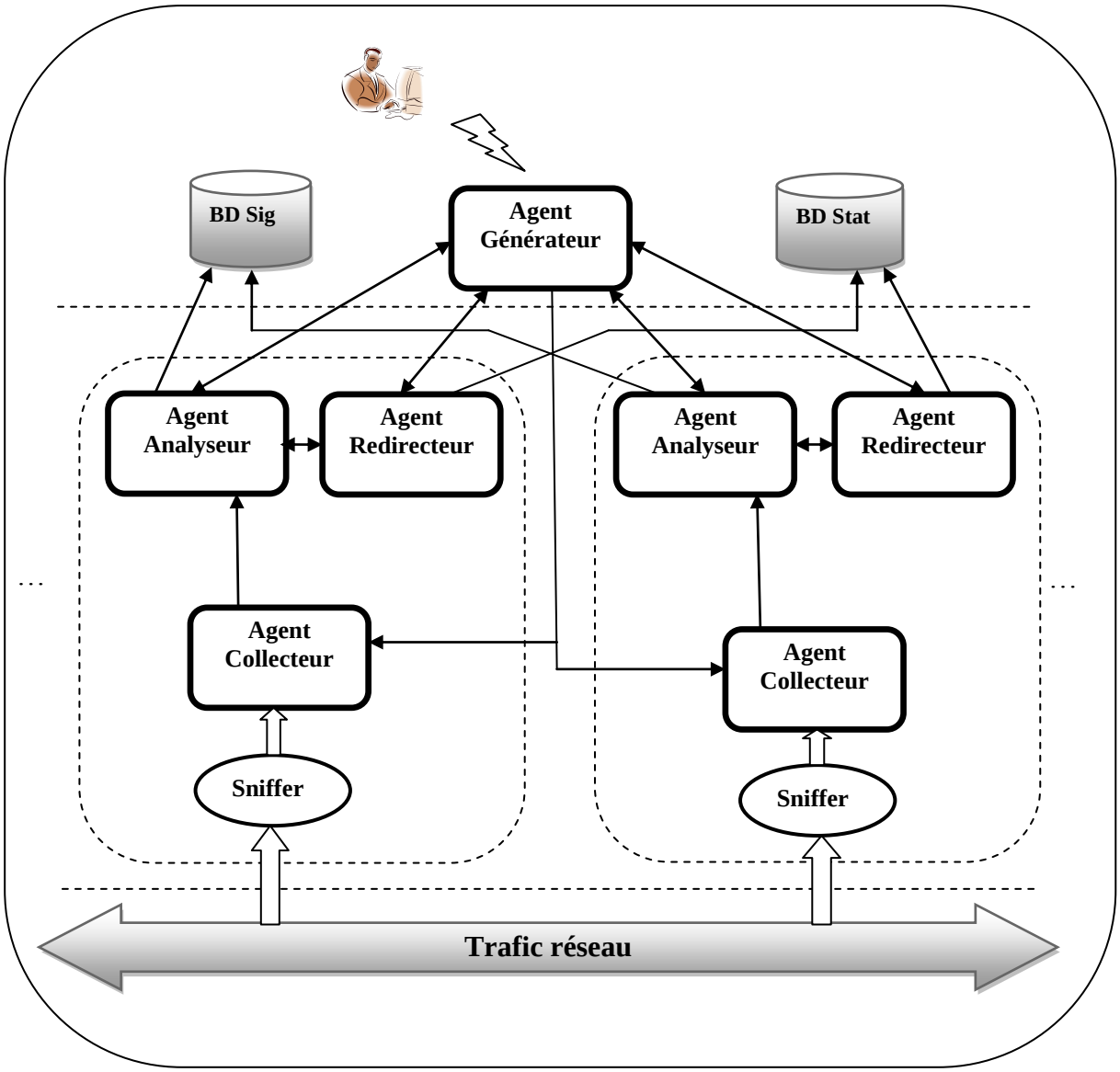


Figure 4.1 Architecture générale du modèle proposé.

Notre architecture se compose de quatre types d'agents : agent générateur, agent collecteur, agent redirecteur et agent analyseur. Ces entités sont capables de se déplacer d'une machine à une autre tout en étant collaboratives, coopérantes et communicantes.

La figure 4.1 illustre les différents canaux de communication qui s'établissent entre les agents du notre modèle. L'agent générateur est responsable de la création et la distribution des trois autres types d'agents, à savoir l'agent collecteur, redirecteur et analyseur. Il enregistre une liste des identificateurs de ces dernières afin de l'utiliser ultérieurement pour la communication entre ces différents agents (pour récupérer les références de l'agent à qui ils veulent envoyer des messages). L'agent collecteur transfert les informations capturées (par un sniffer) et filtrées à l'agent analyseur afin de les analyser (en utilisant la base de signature). Si

une intrusion est détectée, l'agent analyseur va envoyer une alerte à l'agent générateur indiquant les détails sur cette intrusion. Ainsi, l'agent analyseur doit transférer les paquets filtrés et les évènements critiques à l'agent redirecteur afin de compléter l'analyse comportementale (statistique : en utilisant la base de données stat). L'agent générateur peut corréler les alertes qui arrivent des différents agents analyseurs pour arriver à des conclusions.

4.4.1 Schéma structurel de l'approche

Il est essentiel de préciser les propriétés importantes que doivent vérifier les agents qui composent notre modèle:

– Les informations collectées et filtrées : Les agents doivent garantir la bonne qualité des informations qu'ils collectent. Pour évaluer la qualité des informations nous considérons que ces dernières peuvent être classées comme suit :

- L'information pertinente : Est-ce l'information collectée par les agents est importante, utile et correcte relativement au processus de détection d'intrusions.
- information / événement d'indication : Les agents doivent être capables de rapporter aussitôt que nécessaire tout occurrence d'événement utile pour la détection d'intrusion.

– Le niveau de confiance à l'information : Un des inconvénients majeurs de la plupart des IDS actuels, qu'ils soient basés sur l'analyse par scénarios ou l'analyse comportementale, est qu'ils génèrent des faux positifs (l'IDS détecte l'intrusion là où aucune intrusion réelle n'a été perpétrée) ou des faux négatifs (l'IDS ne détecte pas une intrusion ayant réussi en temps opportun). C'est pourquoi il est utile que les agents analyseurs aient les informations qui leurs sont appropriées avec tous les événements corrélés. Par ailleurs, en assignant un degré de confiance à chaque événement selon l'agent source, un agent analyseur peut décider s'il prend en compte ou pas l'information.

– Le protocole de communication agent : En plus du modèle classique de communication agent (authentification, canaux privés ou publiques), nous avons besoin de définir un schéma de protocole spécifique pour faire transiter l'information utile (Chiffrement des messages).

4.4.1.1 Agent Générateur

C'est une entité stationnaire qui constitue le cœur du système. Parmi ses rôles nous pouvons citer : Création et terminaison des différents agents, Gestion des alarmes, Interaction avec les différents agents, mise à jour des bases de données, et détection des connexions.
Le diagramme de classe correspondant à cet agent est présenté dans la figure 4.2.

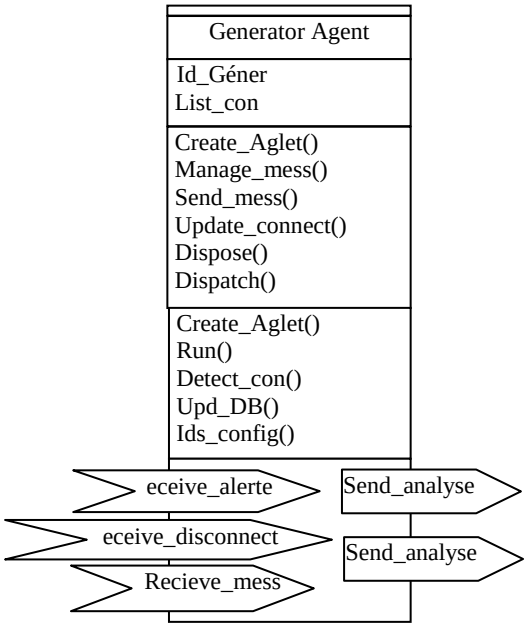


Figure 4.2 Diagramme de classe de l'Agent Générateur

4.4.1.2 Agent Collecteur

Son rôle est la collecte d'informations (paquets) et la préparation (filtrage) de ces derniers. Un sniffer est utilisé pour l'écoute du trafic réseau.
Le diagramme de classe de cet agent est montré dans la figure 4.3.

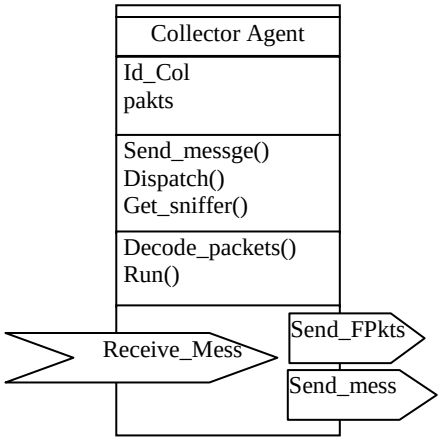


Figure 4.3 Diagramme de classe de l'Agent Collecteur

4.4.1.3 Agent Analyseur

C'est un agent mobile qui peut se déplacer entre les postes. Pour l'analyse de paquets reçus de l'agent Collecteur. Cette analyse est basée sur l'utilisation de la base de données des signatures B.D Sig (analyse par pattern matching). Si l'agent analyseur détecte une attaque il peut envoyer les messages d'alertes à l'agent générateur Si les paquets ne contiennent pas une signature d'intrus, il envoie les paquets à l'agent redirecteur.

Le diagramme de classe d'agent Analyseur est présenté dans la figure 4.4.

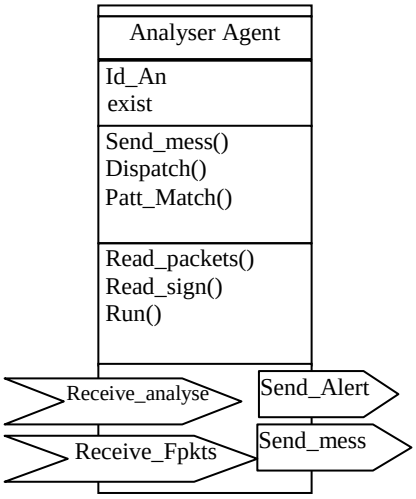


Figure 4.4 Diagramme de classe de l'Agent Analyseur

4.4.1.4 Agent Redirecteur

C'est agent mobile son rôle est l'analyse des paquets reçus en utilisant la base de données B.D Stat (analyse comportementale). En se basant sur des statistiques prédéfinies. En cas de détection de malveillances il envoie les messages d'alertes au générateur.

La figure 4.5 représente le diagramme de classe de cet agent.

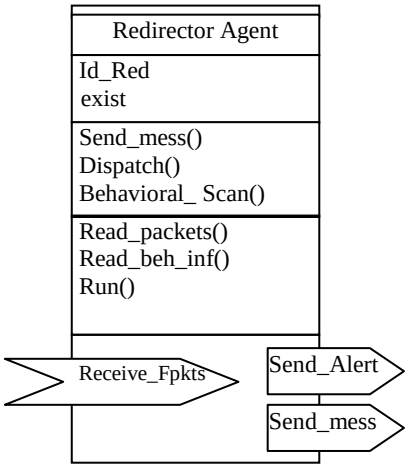


Figure 4.5 Diagramme de classe de l'Agent Redirecteur

4.4.1.5 La classe de Gestionnaire

Cette classe est ajoutée pour l'identification des utilisateurs lors de leurs connexions. La figure 4.6 montre le diagramme de cette classe.

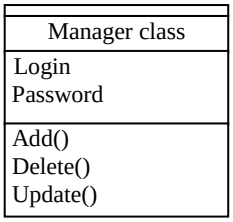


Figure 4.6 Diagramme de classe Gestionnaire

La figure 4.7 présente un diagramme de classes pour le niveau conceptuel de notre système, en montrant les différentes classes de notre IDS.

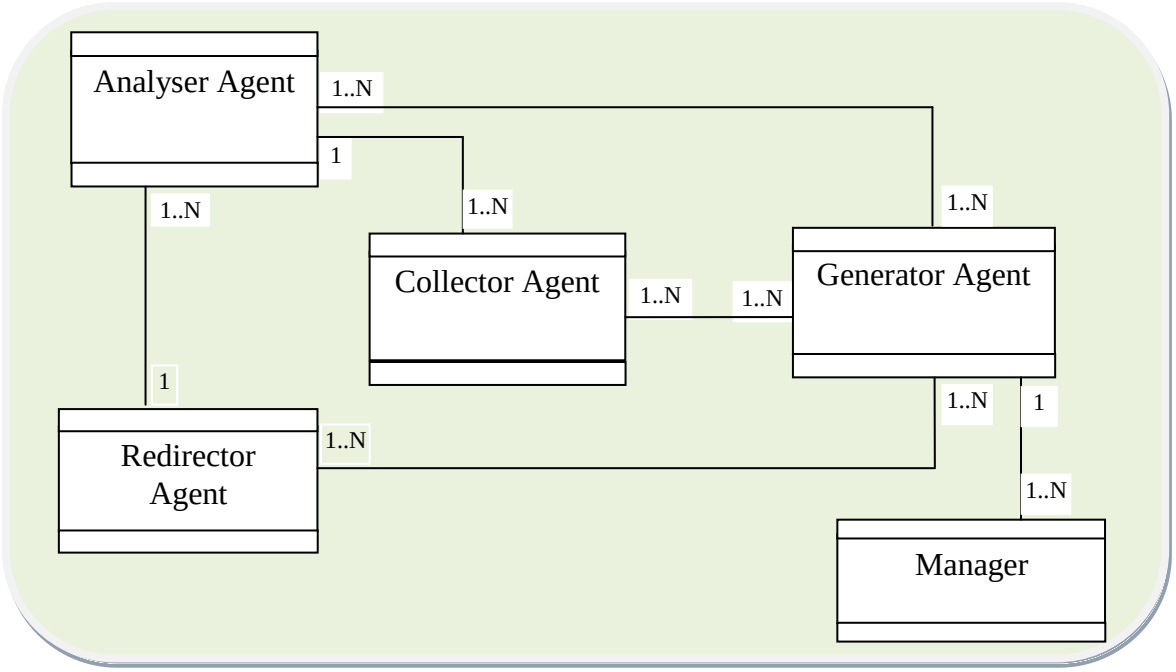


Figure 4.7 Diagramme de classes de l'IDS proposé

4.4.2 Schéma fonctionnel de l'approche

Ici nous décrivons les activités principales de l'IDS illustré par des diagrammes de séquence.

4.4.2.1 Collecte d'informations

Cette activité est illustrée par le diagramme de séquence de la figure 4.8 Où :

Les opérations sont :

-Detect_con() : Obtenir l'adresse de la station détectée.

-MAJ_connect() : mise à jour de la liste des Stations connectées.

-Lire_sniffer() : lire le résultat du sniffer.

-Decoder_paquets() : prépare les paquets.

Et les événements sont:

De l'agent Générateur à l'agent collecteur

-DEM_Collecte : demande à l'agent collecteur de commencer la collecte d'informations de la station détectée.

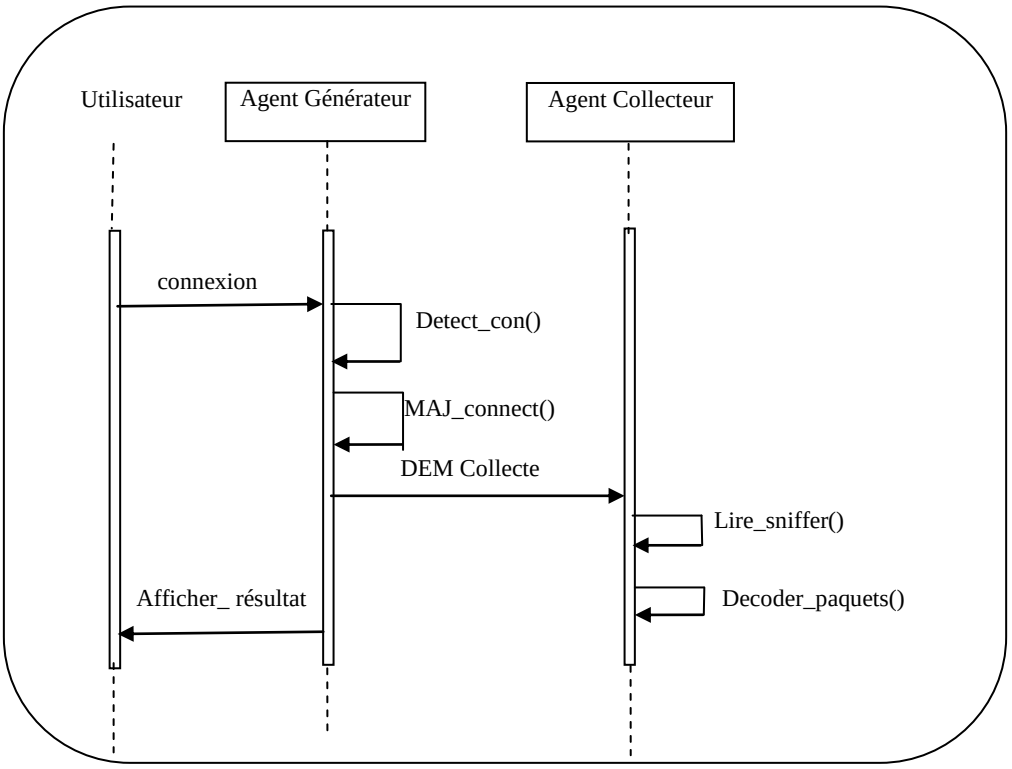


Figure 4.8 Diagramme de séquence de l'activité collecte d'informations

4.4.2.2 Analyse et détection d'intrusions (Pattern matching)

Cette activité est illustrée par la figure 4.9 Où :

Les opérations sont :

-Lire_paquets() : lire les paquets.

-Lire_sign() : lire la liste des signatures d'attaques.

-Patt_Match() :analyse par Pattern Matching.

-MAJ_BD() : mise à jour de la base de données des signatures

Les événements sont:

de l'agent générateur à l'agent Analyseur

-Dem_Analyse : demander à l'agent analyseur de démarrer l'analyse de la station.

De l'agent Analyseur à l'agent Générateur

-Envoyer_Alerte : envoie un message concernant l'attaque détectée

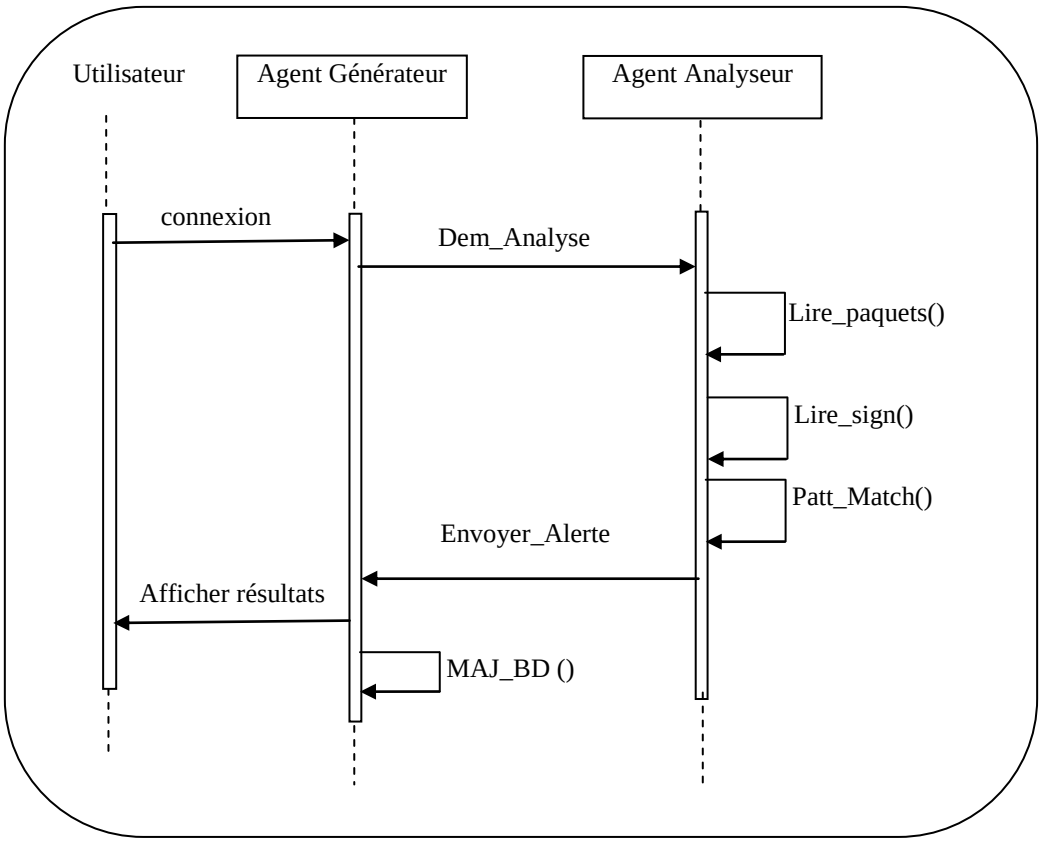


Figure 4.9 Diagramme de séquence de l'activité analyse et détection d'intrusions (Pattern Matching)

4.4.2.3 Analyse et détection d'intrusions (analyse statistique)

Le diagramme de séquence de cette activité est présenté dans la figure 4.10 où :

Les opérations sont :

-Lire_paquets() : lire les paquets.

-Lire_stat() : lire les statistiques prédéfinies.

- scan_Comp () : analyse comportementale.

Les événements:

De l'agent générateur à l'agent redirecteur:

-Lire_Analyse : Demande d'analyse.

De l'agent redirecteur à l'agent Générateur :

-Envoyer_Alerte : envoie de message concernant l'attaque détectée.

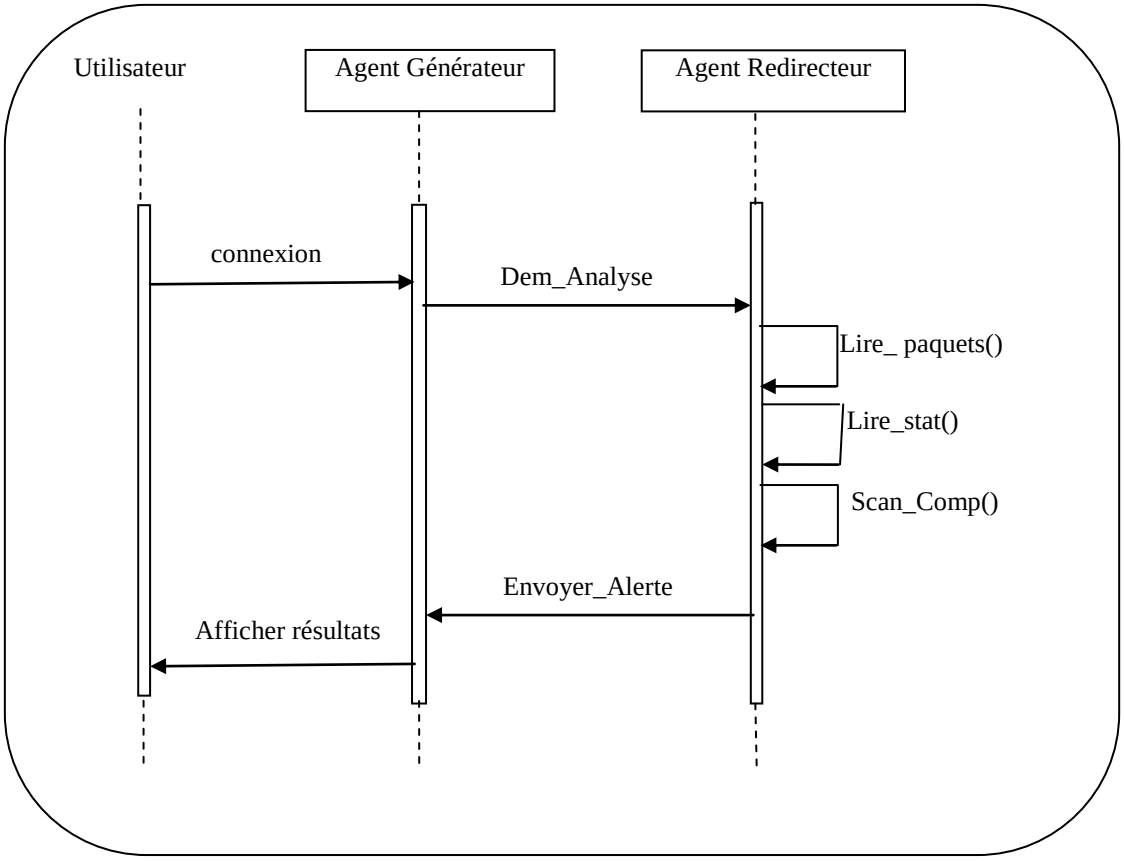


Figure 4.10 Diagramme de séquence de l'activité analyse et détection d'intrusions (analyse statistique)

4.5 Schéma d'une signature d'Attaque

Nous avons défini quatre schémas d'attaque. Ces schémas ont été définis pour détecter les attaques suivantes :

- "*Doorknob Rattling*" : tentatives répétées de login,
- "*Ping Sweep*" : envoi de requêtes "*Ping*" répétées pour identifier les machines connectées au réseau d'entreprise,
- "*Port Scanning*" : balayage des différents ports ouverts sur une machine,
- "*ICMP flooding*" : envoi de requêtes "*ICMP*" répétées pour surcharger le réseau d'entreprise.

Les schémas d'attaques sont représentés par les tableaux symboliques suivants :

a) Doorknob Rattling

Point Observation	Type	Nom	Source	Destination	Résultat
trafic réseau	*connexion*	*Telnet*	*même*	*n'importe quel*	*échec*

b) Ping Sweep

Point Observation	Type	Nom	Source	Destination	Résultat
Trafic Réseau	*connexion*	*Telnet*	*même*	*Différent*	

c) Port Scanning

Point Observation	Type	Nom	Source	Destination	Résultat
trafic réseau	*connexion*	*Telnet*	*même*	*même*	

d) ICMP flooding

Point Observation	Type	Nom	Source	Destination	Résultat
trafic réseau	*Réseau*	*ICMP*	*même*	*différent*	

4.6 Filtrage et préparation des informations

Les informations collectées par le sniffer doivent être filtrées pour les préparer à l'analyse. Le filtrage consiste à classifier les paquets selon un ensemble de critères entre autres : le protocole utilisé, le temps de réception…

Ainsi chaque paquet capturé doit être subdivisé selon les champs à analyser par la suite par l'agent Analyseur soit :

Par type de protocole : {TCP, UDP, ICMP}

Par type de service : {FTP, HTTP, TELNET, etc.}

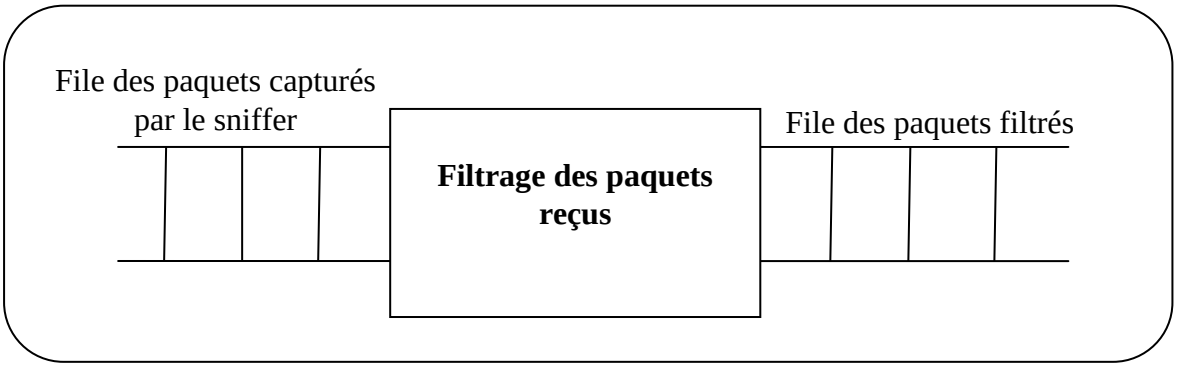


Figure 4.11 Filtrage des paquets réseau

4.7 Choix de l'algorithme Pattern matching

Il existe plusieurs techniques pour réaliser l'approche par scénarios (voir la section 2.6.2), nous nous concentrons sur la méthode de recherche de motifs ou « pattern matching » (voir la figure 4.12).

Le sniffer a pour rôle de capturer les informations circulées dans le réseau, ces informations sont utilisées par le pattern matching pour les comparées avec celles de la base de données des intrusions afin de détecter s'il y'a ou non d'éventuelles attaques. S'il y a une ressemblance entre les paquets filtrés et les signatures des attaques alors une alerte devra être déclenchée.

Plusieurs algorithmes de pattern matching sont définis dans la littérature [35]. Deux catégories de pattern matching peuvent être distinguées [35]:

· avec motifs fixes et texte variant : c'est le cas des NIDS.

· avec texte fixe et motifs variants : correspond à une recherche dans un dictionnaire.

Dans tous les cas, l'efficacité d'un algorithme de pattern matching dépend du nombre de comparaisons de caractères nécessaires pour rechercher le motif. Lors du choix et de

l'implémentation d'un algorithme de pattern matching, plusieurs considérations doivent être prises en compte :

a) ***Le type de pattern matching*** : recherche multi ou simple motif, motifs statiques ou dynamiques, …

b) ***Sensibilité à la casse*** : dans le cas d'une recherche insensible à la casse, les caractères du motif et du texte seront par exemple convertis en majuscules.

c) ***La taille des motifs*** : certains algorithmes ont des performances réduites pour les motifs de grande taille.

d) ***La taille de l'alphabet*** : la plupart des algorithmes fonctionnent facilement si les caractères sont stockés sur un octet, mais dans le cas du codage Unicode, certaines adaptations devront être apportées.

e) ***Le risque d'attaques de l'algorithme*** : des pirates peuvent éventuellement utiliser les propriétés et le comportement de l'algorithme pour réduire les performances du programme. Il est donc nécessaire, surtout dans le cas des IDS, d'évaluer les conséquences de telles attaques.

f) ***La fréquence des recherches et la taille des textes de recherche*** : ces deux aspects peuvent influencer les performances, et l'algorithme devra être adapté aux types de recherches réalisés.

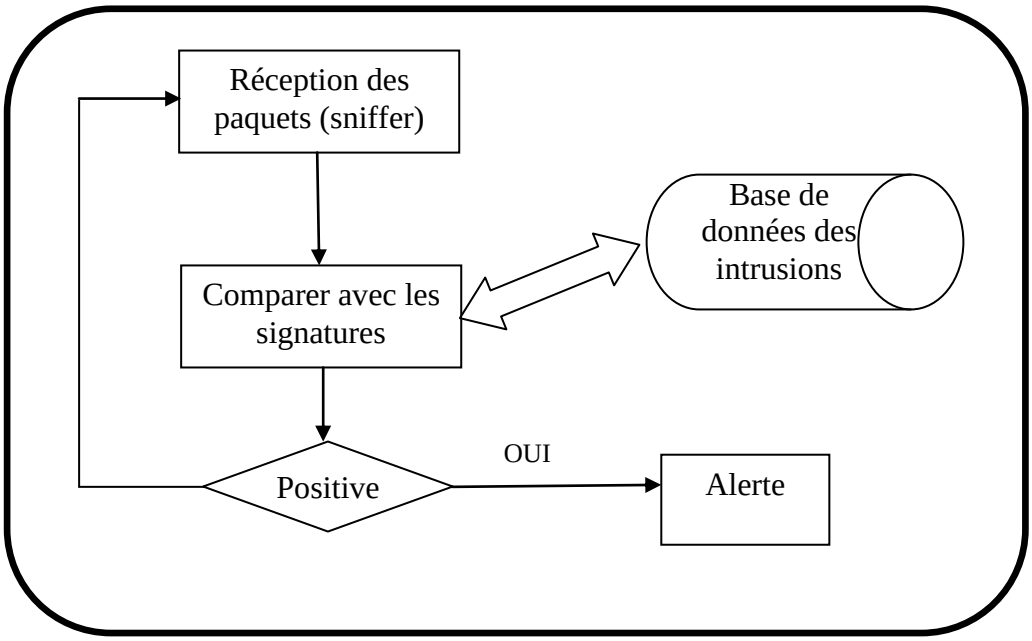


Figure 4.12 Organigramme de Pattern matching

Dans notre méthode on a choisi *l'algorithme naïf* (Brute Force Algorithm) pour le pattern matching. Ce choix est justifié par la simplicité de l'algorithme, ainsi puisque nous voulons tester l'IDS avec quelques types d'attaques.

Dans ce qui suit les notations suivantes seront utilisées :

· x = x0x1…xm-1 : le motif à rechercher, de longueur m

· y = y0y1…yn-1: le texte sur lequel aura lieu la recherche, de longueur n

· x[i] et y[i] : le ième caractère de x ou y, équivalent à xi et yi

La plupart des algorithmes de pattern matching utilisent une fenêtre glissante de taille m. Cette fenêtre représente la position de l'éventuelle occurrence du motif. En d'autres termes, elle englobe les caractères du texte qui vont être comparés avec le motif pour la tentative en cours. Cette fenêtre est décalée à chaque non correspondance (mismatch) et la procédure de comparaison reprend au début de la fenêtre (voir figure 4.13).

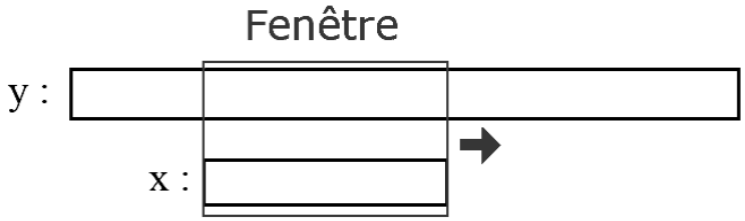


Figure 4.13 Fenêtre utilisée dans les algorithmes de recherche de motifs

L'algorithme naif consiste à vérifier pour chaque position dans le texte si une occurrence du motif commence là ou non. Il compare donc le motif x à chaque facteur du texte y de longueur m. Si une occurrence est rencontrée, on le signale ; sinon, on recommence avec le facteur suivant de y en décalant la fenêtre d'exactement une position vers la droite.

```
Procédure Recherche_Naïve (x, y):
  i := 0
  j := 0
  tantque i < m et j < n faire
     si y[j] = x[i] alors // Caractère identique :
          i := i +1 // on continue dans la même fenêtre
          j := j+1 //
     sinon
          j := j - i +1 // Décalage de la fenêtre
          i := 0 // Le motif est reparcouru
     finsi
  fintantque
  si i = m alors
    Occurrence de x trouvée à la position j – m
  sinon
    Pas d'occurrence
  finsi
```

Figure 4.14 l'algorithme naif

4.8 Analyse comportementale

Dans notre modèle nous nous intéressons à une approche hybride utilisant les deux méthodes de détection d'intrusions : la méthode comportementale et celle par scénarios.

La technique utilisée pour la modélisation du comportement du notre système est basée sur un *Modèle statistique* dit *: Détection basée sur la notion de seuil* (Voir la section 2.6.1 du chapitre 02).

La détection de seuil est l'une des formes de détection les plus utilisée. L'idée de cette approche est d'enregistrer chaque occurrence d'un événement spécifique et lorsque le nombre d'occurrences devient trop élevé durant une certaine période de temps et dépasse un certain seuil, la présence d'un intrus est détectée. Ce seuil ainsi que l'intervalle de temps d'analyse sont définis au préalable par l'officier de sécurité.

L'identification et la précision de ces deux paramètres repose entièrement sur l'expérience de l'officier de sécurité. Cette tâche n'est pas très simple car il faut définir des valeurs de sorte à minimiser le nombre de fausses alarmes et à maximiser le nombre de détections d'intrusions. En effet, définir une valeur de seuil trop faible provoquerait un taux élevé de fausses alarmes

et à l'inverse une valeur de seuil trop élevée peut laisser passer des intrusions sans qu'elles ne soient détectées.

Pour notre cas, nous utilisons des mesures tel que :

- le nombre de connexions durant une certaine période de temps.
- Le nombre de login et logout, connexion et déconnexion.
- le nombre de mots de passe erronés durant un intervalle de temps, …etc.

4.9 Communication entre les agents mobiles

Les quatre agents qui composent notre modèle s'inscrivent dans un but global et commun qui est la détection des intrusions en temps réel. A cet effet la communication entre les agents de ce modèle revêt un rôle important et crucial pour que ces agents puissent coopérer, se partager les tâches de la détection d'intrusions, comparer les résultats obtenus et collaborer afin de prévenir en temps opportun contre les tentatives malveillantes de nuire à la sécurité et au bon fonctionnement du réseau informatique.

Au sein d'une communauté agents mobiles la communication est réalisée soit par échange de messages classiques soit par la migration lorsque l'agent revient sur le site du client (émetteur de la requête d'informations). La communication dans notre modèle est basée essentiellement sur l'envoi de messages.

Les agents peuvent communiquer avec d'autres agents résidents dans la même place ou avec des agents résidents dans d'autres places. Un agent peut évoquer une méthode d'un autre agent comme il peut lui envoyer des messages s'il est autorisé à le faire. La communication inter-agents peut suivre trois schémas différents [3] :

– Now-type messaging : c'est le type de messagerie le plus utilisée. C'est un type synchrone. Il bloque l'exécution de l'émetteur du message jusqu'à ce que le receveur aura complètement téléchargé le message et aura envoyé sa réponse.

– Future-type messaging : c'est un type de messagerie asynchrone qui ne bloque pas l'exécution. L'expéditeur retient une variable qui peut être utilisée pour obtenir le résultat. Ce type de messagerie est utile en particulier quand plusieurs agents communiquent ensemble.

– One-way-type messaging : c'est un type asynchrone qui ne bloque pas l'exécution courante. L'expéditeur ne va pas retenir une variable pour ce message et le receveur ne va jamais répondre. Ce type est utile quand deux agents engagent une conversation où l'agent expéditeur n'a pas besoin de la réponse de l'agent receveur. Ce type peut aussi être appelé fire-and-forget.

Pour encrypter le transfert des agents et des messages qu'ils échangent, nous proposons d'utiliser un des algorithmes de chiffrement/déchiffrement asymétriques dit RSA.

Ce système est proposé par Rivest, Shamir et Adleman en 1978, et aujourd'hui le plus connus et le plus utilisé de par sa simplicité [19][25].

Dans le système RSA, un utilisateur crée son couple (clé publique, clé privée) en utilisant la procédure présentée dans la figure 4.15.

- Choisir au hasard deux grands nombres premiers p et q. Il faut que p et q contiennent au moins 100 chiffres décimaux chacun ;
- Calculer n = pq ;
- Choisir un petit entier e qui est premier avec Ø(n)=(p-1)(q-1) ;
- Calculer d, l'inverse par la multiplication de e modulo Ø(n) ;
- Publier la paire K_e= (e,n) comme sa clé publique RSA.

On a alors :

- Chiffrement RSA : $E_{Ke}(M)= M^e \bmod n$;
- Déchiffrement RSA : $D_{Kd}(M)= M^d \bmod n$.

Figure 4.15 Algorithme RSA [25]

4.10 Conclusion

Nous avons proposé un modèle de distribution du système de détection des intrusions basé sur des entités mobiles, collaboratives et communicantes en vue d'implémenter la collecte et l'analyse distribuée et en temps réel des événements survenus sur le réseau.

L'approche décrite dans ce chapitre est basé sur une méthode d'analyse hybride qui donne la possibilité de la détection des intrusions connues ou inconnues et de diminuer les fausses alarmes et les alarmes négatives.

Pour cela, notre solution doit fournir les points suivants :

- une approche hybride pour détecter les attaques qui se caractérisent par des signatures définies ou inconnues ;
- un modèle collaboratif permettant la détection des attaques de nature distribuée par la corrélation d'alertes;
- un modèle utilisant la technologie d'agents mobiles pour distribuer la détection d'intrusions.

Au cours du chapitre suivant nous allons montrer la faisabilité du modèle proposé, son implémentation et son adaptation à une plate-forme agents mobiles.

Chapitre 5
Mise en œuvre et tests

5.1 Introduction

Nous avons présenté, dans le cadre de ce travail, un modèle utilisant une approche hybride à base d'ag

ents mobiles pour la sécurité d'un réseau, plus précisément pour la distribution de détection d'intrusions.

Dans ce chapitre, nous présentons l'implémentation de notre modèle. Dans un premier temps, nous décrivons Les aspects techniques de l'implémentation. Puis, nous présentons l'environnement de développement des agents : Aglet. Nous décrivons ensuite l'implémentation des différents types d'agents du modèle. Enfin nous finissons par un exemple de scénario comme une étude de cas.

5.2 Aspects techniques de l'implémentation

Afin d'implémenter un prototype d'une manière rapide et structurelle, il faut faire recours aux outils, plateformes et bibliothèques près existantes qui facilite la tâche d'implémentation.

5.2.1 Développement des agents

Au vue de la liste des critères et de l'étude comparative des plates formes agents mobiles citée dans [72] [3], la plate-forme élue pour l'implémentation des agents mobiles est Aglets Workbench D'IBM.

En effet, ce choix est justifié par les faits suivants:

– Le contexte, qu'est un composant essentiel de l'architecture de la plate-forme Aglet, représente l'espace de travail de l'agent Aglet. Le plus important c'est que cet espace de travail est un objet stationnaire qui fournit les moyens pour maintenir et gérer l'activité de l'agent Aglet et ce dans un environnement d'exécution uniforme où le système hôte est

sécurisé contre les Aglets malicieux. D'autant plus qu'il sert de bouclier pour protéger l'agent Aglet contre les accès directes à ses méthodes publiques.

– Le proxy, qu'est un autre composant de l'architecture de la plate-forme Aglet, fournit une transparence de localité pour l'agent Aglet et ainsi sa localité réelle. La communication avec un Aglet s'effectue par le biais de son Proxy.

– La communication au sein de l'Aglet se fait par échange de message entre les Aglets et pas par appel de méthodes. Sachant que l'échange de message offre une plus grande flexibilité d'interaction et d'échange de connaissances entre les systèmes.

– L'Aglet Transfert Protocol (ATP), modélisé à base du protocole HTTP, et l'Aglet API qui représentent une partie intégrante de la plate-forme Aglet, définissent les règles de transport des Aglets et de leurs interfaçages.

– Le protocole de transport, par défaut, des Aglets est atp: (pour Aglet Transfer Protocol), mais d'autres protocoles (ftp, http) peuvent être supportés par le serveur cible.

– L'API permet d'écrire en une seule fois les applications, qui s'exécuteront partout avec les avantages liés à Java. C'est-à-dire, que les Aglets fonctionneront sur chaque machine qui supporte l'API. On n'aura plus à se soucier de la couche matériel ou du système d'exploitation, ni de la manière dont le code de l'API a été programmé.

– Parmi les composants de gestion inclus dans l'API de l'Aglet, on cite le Security Manager, responsable de la protection des hôtes et des Aglets contre toutes entités malveillantes, et le Persistence Manager, responsable de la sauvegarde et de la restauration des Aglets désactivés.

– L'API de communication dans l'environnement Aglet est dérivée du standard MASIF d'OMG, qui offre une interopérabilité entre une variété de systèmes agents.

– Un Aglet doit s'exécuter dans un contexte restreint, notamment afin de ne pas pouvoir accéder aux ressources de l'hôte, sauf autorisation explicite.

– Un Aglet peut restreindre les droits d'accès aux fonctions influant sur son cycle de vie (méthodes dispatch, dispose).

– Aglet offre un contrôle de l'accès en lecture et écriture au système de fichier local.

– L'agent Aglet est autonome et réactif aux messages qu'il reçoit. En plus, Aglet est un agent mobile Java, ce langage qualifié de simple, orienté objet, distribué, interprété, robuste, sûr, portable, dynamique et multithread.

– Le modèle de programmation des Aglets permet au programmeur de configurer des listeners personnalisés dans un Aglet. Ces listeners captent les évènements particuliers du cycle de vie d'un Aglet et permettent au programmeur d'agir lors de ces évènements.

– Les méthodes d'envoi de messages dans Aglet ont l'avantage d'abstraire l'emplacement de l'aglet.

– A chaque Aglet est attaché un objet information, AgletInfo, qui contient les informations suivantes: l'identité de l'Aglet, sa place d'origine, sa date de création, sa place courante, le nom de sa classe et son code de base.

– Il existe différentes manières de localiser un Aglet, on peut citer le fait d'envoyer un agent Aglet à sa recherche ou envoyer un message multicast et attendre la réponse de l'Aglet recherché, sachant qu'à chaque fois qu'un agent Aglet quitte son hôte, il laisse une empreinte électronique faisant référence à sa nouvelle localité, aussi les Aglets sauvegardent leurs positions courantes dans une base de données qui, en général gardera la dernière information à propos de la localité de l'aglet.

– La plate-forme Aglet offre une exécution parallèle, par exemple, si un Aglet doit chercher une information au niveau de dix hôtes, il peut créer lui même 10 Aglets (travailleurs) et les dispatcher en parallèle au niveau des dix hôtes, ces travailleurs, après qu'il aient terminé leurs recherches, retournent les résultats trouvés à leur Aglet parent (créateur) et ils s'auto-détruisent après avoir terminer leur mission.

– Chaque aglet dispose d'un gestionnaire de priorité des messages qui sérialise l'arrivée des messages, établit une priorité pour un type donné de message.

– Le gestionnaire de messages dont dispose Aglet permet de gérer les messages parallèles. En effet, à travers la methode exitMonitor, un Aglet est capable de continuer le traitement du message courant et la réception simultanée d'autres messages et leurs traitements en parallèle.

– Ce gestionnaire de message permet aussi la gestion des messages multicast.

– Les Aglets d'IBM fournissent un modèle d'interface pour une communication plus flexible, et plus extensible que l'invocation normale de méthodes.

– L'aglet supporte un environnement de communication qui est :

– Extensible,

– Indépendant de la localisation,

– Synchrone,

– Asynchrone.

Les mesures de sécurité au niveau des plate-formes Grasshopper, Concordia, Voyager ne sont offerts que dans les versions commerciales.

5.2.2 Description de la plate-forme Aglet

Nous avons choisi la plate forme Aglet pour l'implémentation des agents de notre modèle, ce choix est justifié par les points indiqués ci-dessus. Dans ce qui suit, nous présentons les concepts de base de cette plate forme.

5.2.2.1 Environnement d'Aglet

Le système d'IBM Aglet Workbench [31] propose un environnement de programmation d'agents mobiles dans le langage Java qui est issu du modèle d'agents itinérants de Chess [32]. Aglet est compatible avec la configuration UNIX, X86/Solaris et aussi avec celle de Windows NT/95.

Les aglets (**Ag**ents App**lets)** sont des objets Java mobiles qui peuvent se déplacer d'une machine à une autre. Ainsi, un aglet qui s'exécute sur un hôte peut stopper son exécution, se déporter vers un hôte distant et continuer cette exécution dans son nouvel environnement. [30]

Dans le contexte de l'API Aglets (voir figure 5.1), un *Aglet* est un objet représentant un agent mobile, et dérivé de la classe *com.ibm.aglet.Aglet*. Le *Contexte* est une de *com.ibm.aglet.AgletContext*.

Chaque Aglet possède un identifiant unique sur l'ensemble du réseau, défini par *com.ibm.aglet.AgletID*.

Le *Proxy* (com.ibm.aglet.AgletProxy) permet de conserver une référence sur l'Aglet sans avoir à se soucier de sa localisation réelle. Il permet également de rendre inaccessible l'instance de l'Aglet aux autres clients, notamment pour des raisons de sécurité.

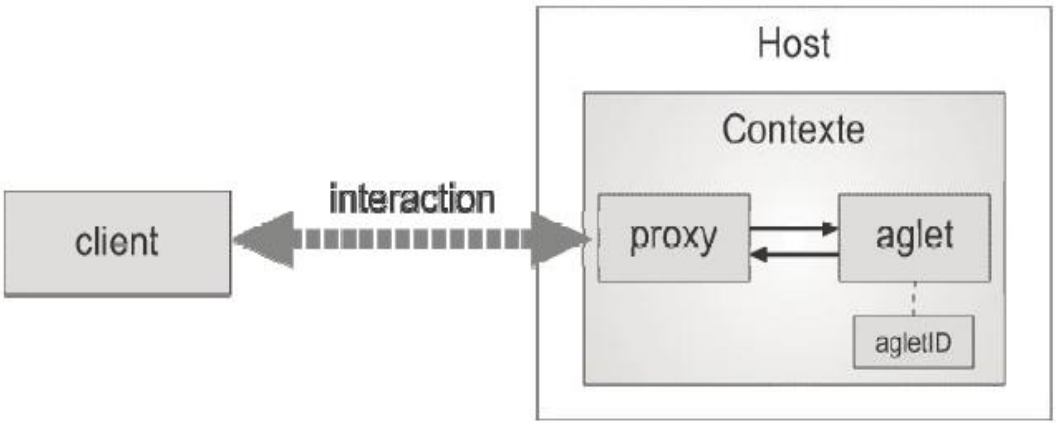


Figure 5.1 l'environnement d'un Aglet

5.2.2.2 Cycle de Vie

Les types de comportement des Aglets ont été implémentés de manière à répondre aux principaux besoins des agents mobiles. Les principales opérations affectant la vie d'un aglet sont [33]:

- ✓ **Création**

Se fait dans un Contexte. Un Identifiant unique est assigné. L'initialisation et l'exécution de l'aglet commence immédiatement.

- ✓ **Clonage**

Création d'un clone dans le même contexte que l'original. Un Identifiant différent est alors attribué. A noter que les processus (thread) ne peuvent pas être clonés.

- ✓ **Déportation** (Dispatching)

Transfert d'un aglet d'un contexte à un autre. On dit que l'aglet à été *poussé* vers son nouveau context.

- ✓ **Récupération** (Retractation)

L'aglet déporté est récupéré (*tiré*) dans son contexte d'origine.

- ✓ **Activation et Désactivation**

La désactivation d'un aglet une interruption temporaire de son exécution et stockage de son état dans un support secondaire de stockage.

- ✓ **Libération ou destruction**

Fin de vie de l'aglet et son retrait du context.

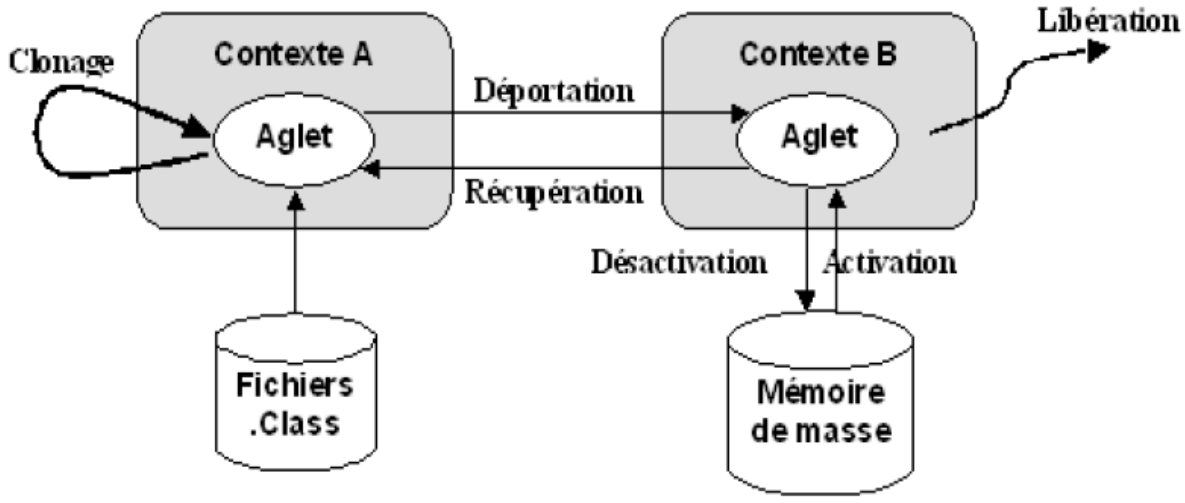


Figure 5.2 Modèle du Cycle de Vie d'un Aglet

5.2.2.3 Configuration de la plate-forme Aglets

La plate-forme dans sa version 2.0.2 intègre un contexte générique d'exécution et d'utilisation des agents. Il s'agit du TAHITI Server représenté par la figure 5.3.

Avant de pouvoir utiliser Aglets 2.0.2 il faut effectuer les tâches préalables suivantes, pour plus de détails voir [30]:

– Un JDK 1.4 : Installer et configurer correctement la JVM [74] avec l'ensemble des variables d'environnement Java à savoir :

– JAVA_HOME

– CLASSPATH

– PATH

– Un environnement de développement Java : C'est pour substituer le mode commande et avoir un environnement de programmation convivial tel que *NetBeans IDE (version 6.5.1)*.

– Aglet 2.0.2 : La version 2.0.2 est disponible sur le Web à l'adresse [14]:

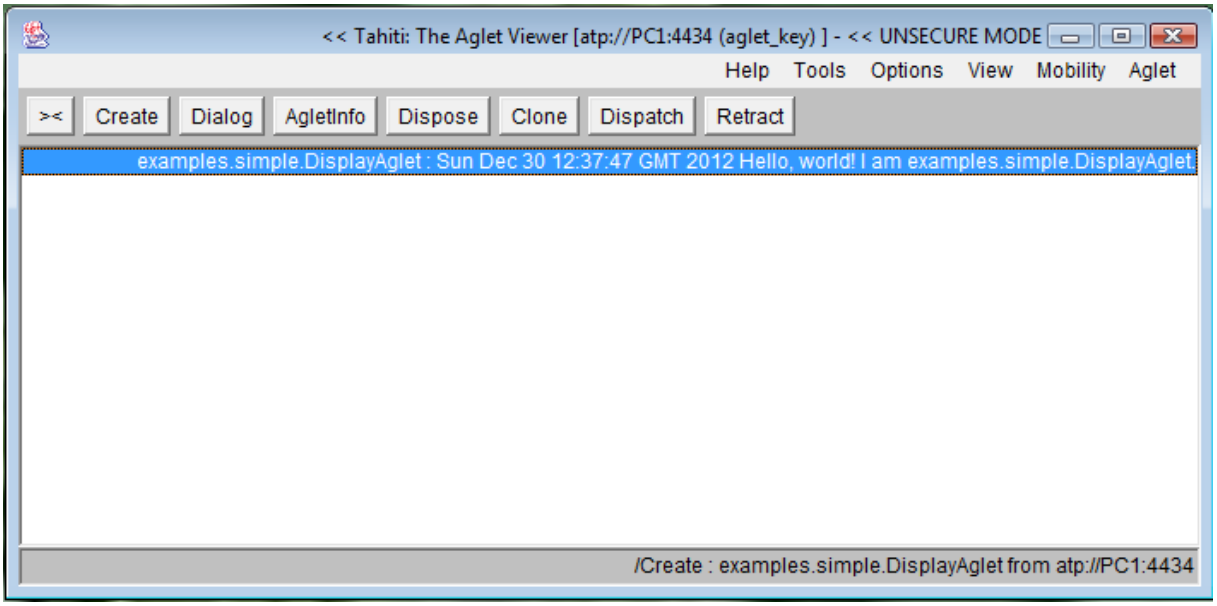


Figure 5.3 L'interface du serveur Tahiti

5.2.3 API Jpcap-0.6

- Jpcap a été évalué sur Microsoft Windows (98/2000/XP/Vista), Linux (Fedora, Mandriva, Ubuntu), le Mac OS X (Darwin), FreeBSD et Solaris.

- Jpcap est une bibliothèque du Java permet de capturer et envoyer des paquets dans le réseau.
- Jpcap est basé sur libpcap/winpcap. Donc, Jpcap devrait travailler sur n'importe quel système d'exploitation qui supporte libpcap/winpcap et JDK.
- Jpcap peut capturer Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP et des paquets ICMPV4.

La version qu'on a utilisée est :

http://ipamss.googlecode.com/files/WinPcap_4_0_2.exe

http://netresearch.ics.uci.edu/kfujii/jpcap/jpcap-0.6.zip.

5.3 Réalisation et Prototypage

Dans cette section, on va décrire le package et les classes Aglet nécessaire pour l'implémentation des agents du notre modèle [14][30].

5.3.1 Le package Aglet

Outre la présentation générique de la plate-forme Aglets faite dans la section précédente, on va détailler dans ce qui suit les classes et interfaces de l'API Aglet qu'on va utiliser pour créer et gérer les Aglets de notre modèle.

Le package Aglet contient les méthodes d'initialisation des Aglets et de traitement des messages. L'API Aglet [3][72] est un package Java composé de classes et interfaces dont on retient les suivantes pour la suite de notre implémentation :

– La classe Aglet,

– L'interface AgletProxy,

– L'interface AgletContext,

– La classe Message,

– La classe AgletID,

– La classe AgletInfo.

5.3.1.1 La classe Aglet

La classe Aglet est la classe clée de l'API Aglets. On l'utilise comme classe de base pour créer d'autres Aglets. Explicitement pour définir un nouveau agent aglet nommé MonPremierIDSAglet qui hérite de classe Aglet:

```
import com.ibm.aglet.*;
public class MonPremierIDSAglet extends Aglet {
//les méthodes des Aglets
```

}

Par la suite, si on veut que MonPremierIDSAglet fasse des initialisations quand il est crée, on utilise la méthode **onCreation()** comme suit:

```
public void onCreation(Object init) {
//initialisation
}
```

Si on a besoin que MonPremierIDSAglet fasse des initialisations à chaque fois qu'il est activé ou qu'il arrive à un nouveau contexte alors on utilise la méthode **run()** comme suit:

```
public void run() {
//initialisation
}
```

5.3.1.2 L'interface AgletProxy

L'interface AgletProxy joue le rôle d'un identificateur de l'aglet à travers lequel on peut accéder aux méthodes de l'aglet. Il est important de noter que la classe aglet contient des méthodes publiques qui ne peuvent pas être accédées directement par d'autres Aglets et ce pour des raisons de sécurité. Un *aglet1*, pour communiquer avec un *aglet2*, doit d'abord obtenir l'accès au proxy de l'*aglet2* pour ensuite pouvoir intéragir avec l'interface AgletProxy de ce dernier. Rappelons à ce propos que l'AgletProxy, quand il est invoqué, consulte le *security Manager* pour déterminer si le contexte courant a les droits d'accès nécessaires.

La création d'un aglet retourne un proxy. Le proxy peut être obtenu de diverses manières :

– *Aglet.getProxy()*

– *AgletContext.getAgletProxy*

– On peut obtenir le proxy d'un aglet par passage de message. En effet, un objet AgletProxy peut être inclu en tant qu'argument dans un objet message qui sera envoyé à un aglet local ou distant.

– On peut inclure l'objet AgletProxy dans les propriétés du contexte en utilisant la méthode *AgletContext.setProperty*.

5.3.1.3 L'interface AgletContext

La notion de contexte pour un aglet est équivalente à la notion de place. Les Aglets utilisent l'interface AgletContext pour avoir des informations a propos de leur environnement d'exécution et pour envoyer des messages aux Aglets actifs se trouvant dans cet environnement. Cet interface fournit les moyens de gérer et maintenir l'exécution des Aglets dans un environnement où le système hôte est sécurisé contre les Aglets malveillants. La classe Aglet fournit des méthodes qui permettent l'accès au contexte courant :

context = getAgletContext();

Ainsi, à partir de la variable *context*, l'aglet peut créer d'autres Aglets :

context.createAglet(getCodeBase(), "nom_du_nouveau_aglet", null);

Les arguments de la méthode de création d'un aglet (**createAglet()**) sont :

– Le CodeBase : c'est une URL qui représente la base de code de l'aglet, en d'autres termes, l'argument getCodeBase doit contenir l'adresse où se trouve le code de l'aglet à créer. En fait, la méthode **getCodeBase()** retourne le Code-Base courant. Si on veut spécifier une autre base de code on remplace la dernière instruction par :

URL codeBase = new URL("C:\LePrototypeIDS\MesAglets");

context.createAglet(codeBase, "nom_du_nouveau_aglet", null);

– Le second argument représente le nom du nouveau aglet.

– Ce dernier argument désigne l'objet qui peut être affecté à l'aglet lors de sa création, c'est un argument d'initialisation.

5.3.1.4 La classe Message

La méthode de traitement des messages reçus par les aglets s'appelle **handleMessage()**.

Les méthodes **sendMessage()** ou **sendAsyncMessage** sont utilisées pour envoyer directement un message synchrone ou asynchrone à un aglet. Ces deux méthodes ont l'avantage d'abstraire l'emplacement de l'aglet. Ce qui signifie que peu importe où se trouve l'aglet, l'interface sendMessage reste identique. L'envoie de message se fait selon la syntaxe suivante:

Message msg = new Message("Le message");

Proxy.sendMessage(msg);

Il est important de noter qu'un aglet ne peut envoyer de message à un autre aglet qu'à travers le proxy de ce dernier. Afin de pouvoir gérer les messages reçus chez le destinataire, on utilise la méthode **handleMessage()** :

```
public boolean handleMessage(Message msg)
{
traitements ;
return true;
}
```

La communication synchrone entre aglet est assurée par la méthode **getReply()**. Après avoir envoyé un message, l'aglet attend la réponse du destinataire par la méthode **getReply()**. Cette méthode bloque l'exécution de l'aglet émetteur du message jusqu'à ce que la réponse lui soit retournée. La syntaxe correspondante à l'envoi de message synchrone est la suivante :

```
Message msg = new Message("objet du message");
FutureReply future = proxy.sendAsyncMessage(msg);
replytype reply = ((replytype)future.getReply());
```

La transmission non bloquante ou asynchrone de messages permet à l'aglet qui a envoyé le message de poursuivre son exécution jusqu'à l'arrivée de la réponse. La classe FutureReply comprend un ensemble de méthodes qui concernent la recherche de la réponse d'un message. Une première méthode **isAvailable()** permet de tester si la réponse est disponible. Une deuxième méthode **waitForReply()** permet d'attendre l'arrivée de la réponse pour continuer l'exécution :

```
FutureReply future = proxy.sendAsyncMessage(msg);
while ( !future.isAvailable())
traitements ;
replytype reply = (replytype)future.getReply();
```

Chaque aglet a un gestionnaire de messages qui lui permet de déterminer l'ordre de prise en compte des messages, en les insérant dans une file d'attente. L'aglet expédie ces messages un par un au gestionnaire de messages d'aglet, dans l'ordre d'arrivée.

Le gestionnaire de messages s'assure également que le prochain message ne sera pas expédié tant que le message actuel ne soit encore traité. Concrètement, le gestionnaire de messages sérialise l'arrivée des messages, établit une priorité élevée pour un type donné de messages et assure qu'il sera placé dans la file d'attente avant les messages de priorité inférieure. Une priorité est affectée à un message en utilisant la méthode **etPriority()** de la classe MessageManager :

getMessageManager().setPriority(msg, priorité);

Précédemment, nous avons traité les messages de type direct, appelé encore monocast, où l'agent émetteur du message a besoin d'avoir le proxy et l'identifiant de l'agent receveur pour pouvoir lui envoyer le message. Ce besoin constitue une limite quand on prévoit de concevoir une coordination d'activité entre des groupes d'Aglets par envoi de message. Dans ce cas le multicast comme type de messagerie offert par la plate-forme Aglets présente la solution. La messagerie multicast stipule qu'on a un ensemble d'Aglets originaires de différentes sources qui se rencontre dans un contexte spécifique (un lieu de meeting). Ainsi, un aglet peut envoyer un message à tout ce groupe d'agents en même temps. Pratiquement, la messagerie multicast nécessite que les Aglets s'inscrivent à un ou plusieurs messages multicast en invoquant la méthode **subscribeMessage()** et traitent la réception des messages.

5.3.1.5 La classe AgletID

A chaque aglet crée est assignée une identité unique tout le long de sa durée de vie. La classe AgletID est l'abstraction de cette identité. On peut récupérer l'identité d'un aglet à travers son proxy :

AgletID aid = proxy.getAgletID();

Ayant l'identifiant de l'aglet et son contexte on peut récupérer l'aglet comme suit :

proxy = context.getAgletProxy(aid);

5.3.1.6 La classe AgletInfo

La classe AgletInfo contient toutes les informations concernant l'aglet actif :

– Son identifiant,

– L'adresse où il réside,

– Son nom,

– L'adresse de sa base de code,

– L'adresse où il a été instancié,

– L'heure de sa création,

– Le nom de son autorité de certification.

5.3.2 Mise en œuvre des agents du modèle

Dans cette section, On va présenter les différents algorithmes des agents du modèle.

5.3.2.1 Algorithme Agent Générateur

Voici l'algorithme correspondant à la mise en œuvre de l'agent Générateur :

AgletID idGen; // idetificateur d'agent Générateur.
String ippc ; // adresse IP de PC Generateur.
AgletProxy tableproxy[ ][ ] ;// tableau contient les proxys d'agents qu' il a crée.
String listePcsEnReseau;// liste des PCs connectés.

Début

idGen ← getAgletID();
ippc ← adresse PC Generateur ;

1. Donner le nom utilisateur et mot de passe // Authentification
2. Si (le nom d'utilisateur et mot de passe sont correcte) Alors

2.1 Si (voulez changer le mot de passe) Alors
Mot de passe ←nouveau mot de passe ;
2.2 FinSi.
2.3.1 Initialiser listePcsEnReseau par les IPs des Pcs Connectés
2.3.2 Pour (String ip : listeDesPcsEnReseaux)
2.3.2.1 Si (ip = ippc) Alors
// Créer les agents.

- tableproxy[0][index_ip]←getAgletContext().createAglet(getCodeBase( ), "code.Analyseur",null);//créer agent Analyseur.
- tableproxy[1][index_ip]←getAgletContext().createAglet(getCodeBase( ),"Redirecteur",null);// ceer agent Redirecteur.
- tableproxy[2][index_ip]←getAgletContext().createAglet(getCoeBase(), "Collecteur",null);// ceer agent Collecteur.

// envoyer idGen pour chaque agent à travers le proxy.

- tableproxy[0][index_ip].sendMessage(new Message("ID",idGen));
- tableproxy[1][index_ip].sendMessage(new Message("ID",idGen));
- tableproxy[2][index_ip].sendMessage(new Message("ID",idGen));

// envoyer ippc de chaque agent.

- tableproxy[0][index_ip].sendMessage(newMessage("IP",ippc));
- tableproxy[1][index_ip]. sendMessage(new Message("IP",ippc));
- tableproxy[2][index_ip]. sendMessage(new Message("IP",ippc));

// dispatcher les agents.

- tableproxy[0][index_ip].dispatch(new URL(new String("atp:// ip"))) ;
- tableproxy[1][index_ip].dispatch(new URL(new String("atp:// ip"))) ;
- tableproxy[2][index_ip].dispatch(new URL(new String("atp:// ip"))) ;

2.3.2.2 FinSi.
2.3.3 FinPour.
// Traitement de reception des messages
2.3.3.1 Si (msg= ("attack_network")) Alors
ecrire("attaque network dans le PC :",détail_attack()) ;
Correler_attack() ;
2.3.3.2 Sinon (msg= ("MAJ")) Alors
//Faire le mis a jour de la base de signature d'intrusions.

2.3.3.3 FinSi.
2.3.4 FinSi.

3. Sinon
   Ecrire (‘ erreur nom d’utilisateur ou mot de passe ‘) ;

4. FinSi.
Fin.

Figure 5.4 Algorithme d’Agent Generateur

5.3.2.2 Algorithme Agent Collecteur

- ***Mode promiscuous***

Les cartes réseaux peuvent fonctionner sous un mode spécial : *promiscuous mode*, ce qui permet à la carte de récupérer tous les paquets passant sur un support partagé comme un bus Ethernet. Par défaut une carte réseau ne va récupérer que les paquets lui étant destinés. En mode promiscuous, la carte va littéralement avaler tous les paquets qui circulent sur le réseau. Les paquets capturés sont alors transmis du pilote de périphérique vers le SDI pour analyse.
Nous avons utilisé les bibliothèques WinpCap et JpCap pour implémenter l’Agent Collecteur. Pour cela il faut importer :
import jpcap.*;

- ***Le sniffer***

Le code source utilisé pour capturer les paquets (le sniffer) circulant dans le réseau est comme suit :

```
import jpcap.*;

class JSniffer implements JpcapHandler
{public void handlePacket(Packet packet)
{
System.out.println(packet);
}
public static void main(String[] args) throws java.io.IOException
{
String[] lists=Jpcap.getDeviceDescription();
System.out.println("\n\t\t***My Simple Network Sniffer***\n");
System.out.println("Start capturing on "+lists[0]);
Jpcap jpcap=Jpcap.openDevice(Jpcap.getDeviceList()[0],1000,false,20);
jpcap.loopPacket(-1,new JSniffer());
}
}
```

Figure 5.5 Code source du sniffer

L'algorithme de l'agent Collecteur est comme suit :

Début

1. Reception message (ID) ;
2. Reception message (IP) ;
3. Choisir l'interface reseau local.
4. Tanque(true)

   4.1 Capter le paquet / /utilisant le sniffer

   4.2 Filtrer_Packet();

   4.3 Envoyer_Packet() ;
5. FinTanque

Figure 5.6 Algorithme d'Agent Collecteur

5.3.2.3 Algorithme Agent Analyseur

```
AgletProxy pGen ;// proxy d'agent Generateur.
String   ippc ;  // adresse IP de PC où il existe.

 Début
    1.  Reception message (ID) ;
    2.  Reception message (IP) ;
    3.  pGen←getAgletContext().getAgletProxy(new URL("atp:/IP",ID)) ; //ceer proxy
Generateur.
    4.  Tanque(true)
        4.1 Capter le paquet / /utilisant le sniffer
        4.2 Test←Pattern_matching();
        4.3  Si (Test=detecter attaque) Alors
          4.3.1 pGen. sendMessage(new Message("attack_network", ippc));
        4.4  FinSi.
       FinTanque.
 Fin.
```

Figure 5.7 Algorithme d'Agent Analyseur

5.3.2.4 Algorithme Agent Redirecteur

```
AgletProxy pGen ;// proxy d'agent Generateur.
String   ippc ;  // adresse IP de PC où il existe.

 Début
    1.  Reception message (ID) ;
    2.  Reception message (IP) ;
    3.  pGen←getAgletContext().getAgletProxy(new URL("atp:/IP",ID)) ; //ceer proxy
Generateur.
    4.  Tanque(true)
        4.1 Lire le paquet
        4.2 Test←Analyse_Comp();
        4.3  Si (Test=detecter attaque) Alors
          4.3.1 pGen. sendMessage(new Message("attack_network", ippc));
        4.4  FinSi.
    5. FinTanque.
 Fin.
```

5.3.3 Connexio **Figure 5.8** Algorithme d'Agent Redirecteur

La base de données des signatures d'attaques et la base de données des paramètres statistiques du système sont implantées sous le SGBDR Access et le système d'exploitation Windows XP. L'API Java JDBC est l'interface java pour accéder aux bases de données relationnelles. Chaque base doit proposer son driver JDBC qui permet d'y accéder. Quelque soit la base, le driver JDBC se manipule de la même manière, cette interface fournit un ensemble de classes permettant l'utilisation d'un ou plusieurs SGBD relationnels à partir des programmes Java [29].

5.4 Etude de cas : Exemple de scénario

Pour tester le modèle proposé, nous avons utilisé trois ordinateurs reliés entre eux via le réseau indiqué dans la figure 5.9. Le système est configuré comme suit:

- Dans les trois machines on a installé la plate forme Aglet.
- Dans la machine PC1, on a configuré notre IDS (installation des API décrit dans la section 5.2.3, ainsi les classes des Aglets du modèle)

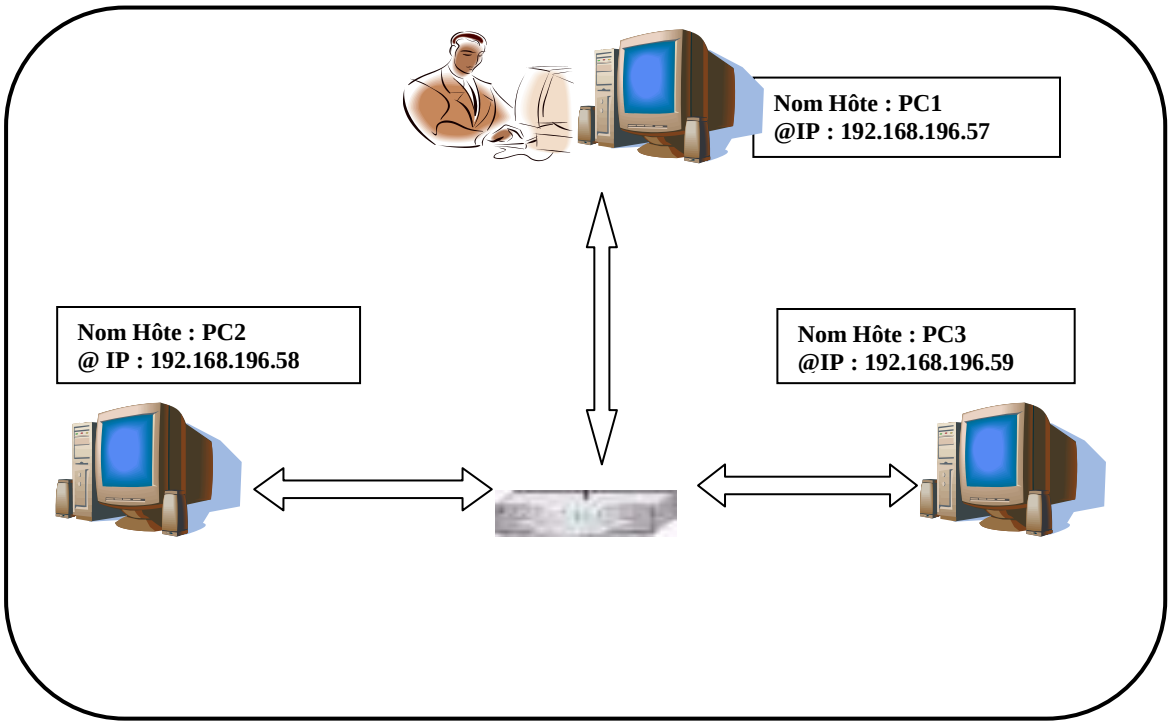


Figure 5.9 Exemple de scénario

Lors du démarrage du système, une fenêtre initiale (voir la figure 5.10) s'affiche à l'utilisateur demandant l'authentification.

Figure 5.10 Fenêtre Identification/authentification

Après la saisie de l'identificateur et mot de passe l'interface principale de notre IDS s'affiche (voir la figure 5.11). Après la demande de scan de réseau le système démarre son analyse.

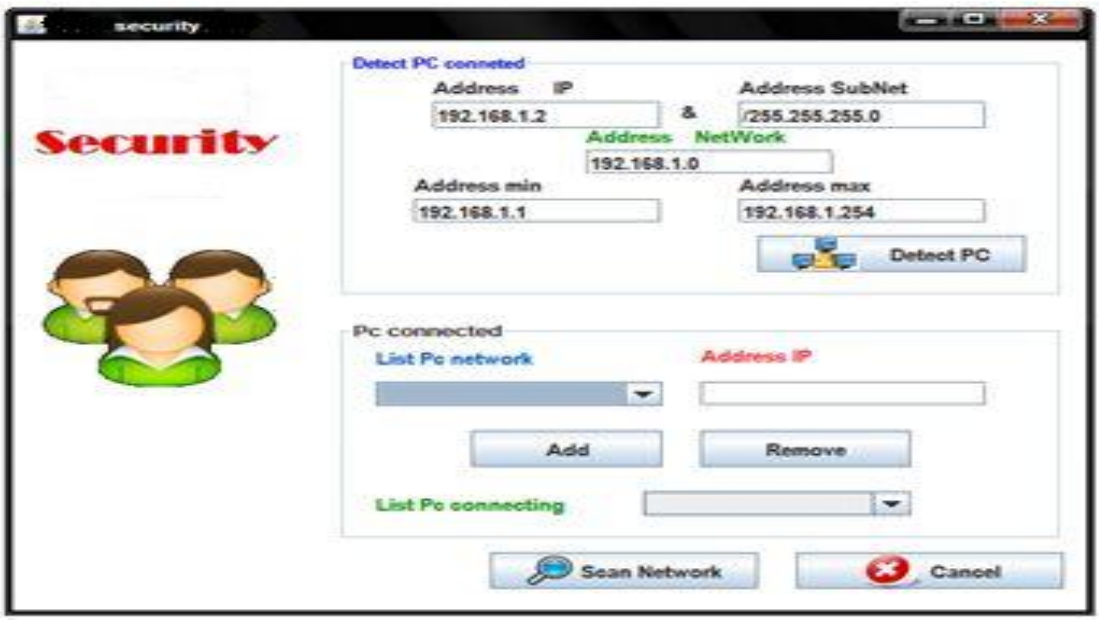


Figure 5.11 Interface principale du système

L'agent Generateur va détecter les adresses IP des deux machines, puis il va créer les trois types d'agents à savoir: l'agent analyseur, l'agent collecteur et l'agent redirecteur. Il va par la suite dispatcher les trois agents vers les deux autres machines. A l'arrivé des agents aux postes PC1 et PC2, ils démarrent leurs activités :

- l'agent collecteur va capturer les paquets circulant dans le réseau en utilisant le sniffer (décrit en 5.3.2.2). Puis il filtre ces paquets afin de les préparer à l'analyse, et les envoie à l'agent analyseur.
- L'agent analyseur, après la réception d'un paquet, il utilise d'une part, la base de données des signatures d'intrusions et d'autre part les informations sur le paquet filtré afin d'appliquer le pattern matching et tester s'il y a ou non une trace d'intrusion. S'il détecte une intrusion il envoie une alerte à l'agent générateur indiquant les détails sur celui-ci. Il envoie ainsi ce paquet à l'agent redirecteur.
- L'agent redirecteur après la réception du paquet à ça part, il applique l'analyse statistique en utilisant les informations du paquet filtré ainsi la base de données des

statistiques sur les intrusions. S'il détecte un évènement anormal il envoie une alerte à l'agent générateur.

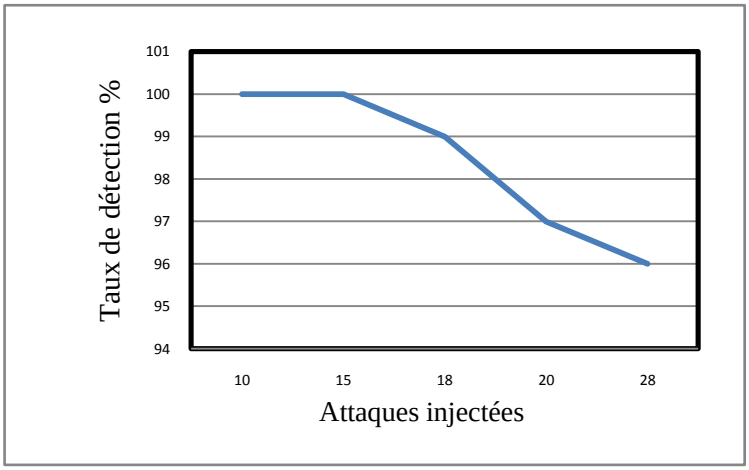
L'agent générateur après la réception des alertes, peut corréler ces derniers et génère une alerte indiquant une intrusion réseau.

Pour tester notre système, nous avons simulé les attaques décrites dans la section 4.5 du chapitre 04. Pour cela on a utilisé l'outil HPING [16] qui est capable d'envoyer des paquets TCP/IP aux différents hôtes du réseau.

Le système détecte ces attaques et les alertes des différents postes sont envoyées à l'agent Générateur. L'interface d'une alerte est comme indiquée dans la figure 5.12.

Figure 5.12 Alerte indiquant les détails sur l'intrusion détectée

Nous avons teste le système sur le réseau reel. La mise en œuvre nous donne les résultats attendus du terme de taux de détection. La figure 5.13 montre le taux de détection des attaques injectées simulées dans le réseau.



En fait, pour évaluer la performance d'un IDS, deux métriques intéressantes sont d'habitude d'utilisation [87] : le Taux de Détection (Detection Rate :DR) et le Taux de Faux Positif (False positive Rate :FR).

1. Le DR est égale au nombre d'intrusions correctement détectées divisées par le nombre total d'intrusions dans l'ensemble de données;
2. Le FR égale le nombre total des cas normaux que l'on a inexactement considérés comme des attaques divisées par le nombre total de cas normaux dans l'ensemble de données. On cherche une valeur du DR d'être aussi grand que possible, tandis que l'on cherche une valeur du FR d'être aussi petit que possible.

Un taux de détection au-dessus de 96 % est abouti par l'application de notre approche. Des résultats expérimentaux montrent que l'approche mise en œuvre démontre l'efficacité du système proposé.

5.5 Etude comparative

Pour évaluer la performance de notre approche, nous avons comparé nos résultats avec les résultats d'autres modèles existants. La table 5.1 montre que notre approche est efficace dans plusieurs aspects comparée aux autres schémas. Le taux de détection d'intrusions est élevé tant pour les intrusions connus que pour les intrusions inconnus, ceci, est justifié par l'utilisation de l'analyse hybride de l'information. Ainsi, l'approche proposée performe la détection des intrusions distribuées.

Nom du modèle	Intrusion Detection of Packet Dropping Attacks in Mobile Ad Hoc Networks (2006) [84]	Agent based efficient anomaly intrusion detection system (2010) [86]	Efficient intrusion detection system using random tree (2015)[83]	Approche proposée
Auteur	A.Mitroktsa, R.Mavropod & C. Douligeris	R.Nakeeran	Deepa, A.J. and Kavitha, V.	D. Boukhlouf
Algorithme	détection distribuée basée sur le réseau neuronal	Agent coopératif et système distribué	algorithme d'arbre aléatoire comme un algorithme de classification d'exploration de données	Agent Mobile basé coopérative et distributive
méthode de détection d'Intrusion	Approche comportementale	Approche comportementale	Approche basé signature et comportementale	Hybride
Taux de détection	moyen	élevé	élevé	élevé
Avantage	Identifier la source de l'attaque packet dropping -Capable pour identifier de nouvelles attaques	Faible taux de fausses alarmes et les performances sont meilleures que d'autres IDS	Arbre de décision fonctionne bien avec d'énormes ensembles de données. - Haute précision de détection.	Détecte des attaques connues et inconnues. - la détection des attaques distribuées.
inconvénient	Les classes de données formées doivent être définies manuellement. U-matrice eSOM mise à jour en permanence.	Pas de description sur les aspects de sécurité des agents mobiles.	Construire un arbre de décision est une tâche de calcul intensif.	Plus de calcul et de ressources sont necessaries.

Tableau 5.1 Comparaison des performances de l'approche proposée avec d'autres modèles existants

5.6 Conclusion

On a proposé une approche hybride basée sur les agents mobiles pour la détection d'intrusions. L'approche est fondée sur la plate forme Aglets pour la création et la mobilité d'agents. Donc, ces derniers peuvent se déplacer de poste en poste pour analyser les paquets récoltés par les agents Collecteurs. L'analyse hybride est assurée par les agents analyseurs et redirecteurs de telle sorte qu'ils détectent d'éventuelles attaques.

L'approche proposée a les particularités suivantes :

- ✓ L'application des agents mobiles pour la détection d'intrusion facilite la distribution de l'IDS (au contraire des systèmes centralisés)
- ✓ L'utilisation d'une approche hybride profite des avantages des deux méthodes de détection d'intrusion (scénario et comportementale)
- ✓ La plateforme Aglets donne la possibilité d'utiliser des méthodes pour la gestion des agents mobiles et pour la distributivité.

Les résultats obtenus montrent l'efficacité de l'approche proposée.

Conclusion générale et perspectives

Avec ses nombreux avantages, l'Internet a également créé plusieurs manières pour compromettre la sécurité et la stabilité des différents systèmes qui le constituent. Pour cela, des politiques et des outils ont été développés pour fournir des mécanismes de défense de plus en plus efficaces. Les solutions de défense statiques, qui sont analogues aux barrières autour des propriétés, peuvent fournir un niveau de sécurité raisonnable. Ils sont prévus pour empêcher les attaques de se produire. Parmi les exemples de ces solutions le maintient des logiciels tels les systèmes d'exploitation à jour, et le déploiement des pare-feu aux points d'accès. Aucun système n'est parfait. Les pirates informatiques sont toujours en avance pour trouver les failles de sécurité. Pour cela des mécanismes de défense dynamiques tels que les Systèmes de Détection d'Intrusions (IDS) devraient être combinés avec les solutions préventives.

Notre point de départ a été de faire une analyse des systèmes de détection d'intrusions existants, puis d'étudier les technologies agents pour voir comment elles pouvaient apporter une solution optimale au problème de détection d'intrusions.

Nous avons analysé les systèmes de détection d'intrusions existants afin :

• d'identifier les faiblesses des systèmes existants par rapport à l'évolution des besoins de sécurité ;

• de déterminer les caractéristiques nécessaires pour une détection d'intrusions efficace et plus flexible.

Le modèle, ainsi développé, réalise l'objectif qu'on s'est fixé de prouver la faisabilité d'utiliser les agents mobiles dans l'IDS pour pallier à certaines limites des IDS centralisés et bénéficier des avantages d'une approche agents mobiles. Le prototype du modèle met en œuvre une intégration claire de la technologie agents mobiles coopératifs dans le processus de détection d'intrusions.

La majeure part de notre intérêt était portée sur le choix judicieux de la plate-forme agents mobiles dans laquelle on a implémenté notre prototype. Les critères de sélection étaient orientés vers les mesures de sécurité mises en œuvre par la plate-forme pour protéger les agents et les sites hôtes des attaques.

L'approche proposée est une approche hybride basée sur les agents mobiles pour la distribution de la détection d'intrusions. L'approche est fondée sur la plate forme Aglets pour la création et la mobilité d'agents. Donc, ces derniers peuvent se déplacer de poste en poste pour analyser les paquets récoltés par les agents Collecteurs. L'analyse hybride est assurée par les agents analyseurs et redirecteurs de telle sorte qu'ils détectent d'éventuelles attaques dans les paquets capturés..

Nos contributions se situe dans les points suivants [43][44][45][88] :

- ✓ L'application des agents mobiles pour la détection d'intrusion facilite la distribution de l'IDS (au contraire des systèmes centralisés)
- ✓ L'utilisation d'une approche hybride profite des avantages des deux méthodes de détection d'intrusion (scénario et comportementale)
- ✓ La plateforme Aglets donne la possibilité d'utiliser des méthodes pour la gestion des agents mobiles et pour la distributivité.

Néanmoins, le travail présenté dans cette thèse a besoin des extensions. Donc, des améliorations et des perspectives de recherches se dessinent et les points suivants décrivent les principales idées qui vont guider nos travaux futurs.

- ➢ étendre la méthode de représentation des schémas de signatures par l'utilisation des techniques de représentation telle que : Datamining, réseaux bayésiens…
- ➢ étendre l'agent Générateur pour corréler les alarmes afin de déduire tous les attaques distribuées.
- ➢ Ajout d'un agent Gestionnaire pour gérer les alarmes entre les intranets.
- ➢ Orienter cette architecture vers le cloud computing et vers internet des objets.

Références

[1] K. Boudaoud. *Détection d'intrusions, Une nouvelle approche par systèmes multi-agents*. thèse de doctorat, Lausanne, EPFL 2002.

[2] AH. Boudjelida, *Réseaux Bayésiens Naïfs Augmentés TAN pour les Systèmes de Détection d'Intrusions*. Mémoire de Magistère ISI, 2008.

[3] F.A.M Barika, *Vers un IDS Intelligent à base d'Agents Mobiles*. Mémoire de DEA Université de Tunis 2003.

[4] Abraham, A., Jain, R., Thomas, J., & Han, S. Y, *D-SCIDS: Distributed soft computing intrusion detection system*. Journal of Network and Computer Application, 30, 81-98. 2007.

[5] J. Hochberg; K. Jackson; C. Stallings; J. F. McClary; D. DuBois; J. Ford, *Nadir: An automated system for detecting network intrusion and misuse*. Computers and Security, 12(3):235-248, 1993.

[6] D. Boughaci, K. Ider and S. Yahiaoui, *Design and Implementation of a Misused Intrusion Detection System Using Autonomous and Mobile Agents.* LRIA/ USTHBBP El-Alia, Beb-Ezzoaur, Algiers, 16111, Alegria. 2007.

[7] Jeffrey M. Bradshaw, *An introduction to soft-ware agents*. In Jeffrey M. Bradshaw, editor, Software Agents, chapter 1. AAAI Press/The MIT Press, 1997.

[8] J. Anderson, *Computer security threat monitoring and surveillance.* Technical report, Pennsylvania, April 1980.

[9] D. E. Denning, *An intrusion detection model*. IEEE Transactions on software engeneering, SE-13 :222232, 1987.

[10] G. B. White; E. A. Fisch; U. W. Pooch, *Cooperating security managers: A peer based intrusion detection system*. IEEE Network, pages pp. 20–23, Janvier-Février 1996.

[11] DB. Lange; M. Oshima, *Programming and Deploying Java Mobile Agents with Aglets*. Massachusetts, Addison Wesley, seconde edition, isbn 0-201-32582-9 edition, 225 p. 1998.

[12] B. Bauer, H. Van Dyke Parunak, J. Odell, *Extending UML for Agents.* 2001.

[13] F. A. Barika & N. El Kadhi & K. Ghedira, *Agent IDS based on Misuse Approach*.Journal of Mathematical Sciences 2009.

[14] Aglets Web: http://www.trl.ibm.com/aglets/index_e.htm

[15] I. Brahmi, S. Ben Yahia, and P. Poncelet, *MAD-IDS: Novel Intrusion Detection System using Mobile Agents and Data Mining Approaches*. Lecture Notes in Computer Science Volume 6122, pp 73-76, 2010.

[16] HPING. http://www.hping.org. Accès Mars 2016.

[17] L. Mé et C. Michel, *La détection d'intrusions : bref aperçu et derniers développements*. Actes du congrès EUROSEC'99, 1999.

[18] S.Ghernaouti-Héli, *Stratégie et ingénierie de la sécurité des réseaux.* Inter Editions, 1998.

[19] L.Bloch, C.Wolfhugel, *Sécurité informatique Principe et méthode*. Eyrolles,2007.

[20] J.F Carpentier, *La sécurité informatique dans la petite entreprise Etat de l'art et Bonnes pratiques.* Editions ENI, 2009.

[21] E. Maiwald, *Sécurité des réseaux.* Campus Press, 2001.

[22] D.J.Stang, S.Moon, *Sécurité des réseaux.* Dunod, 1996.

[23] E.Filiol, *Les virus informatiques : théorie, pratique et applications.* Springer-Verlag France, 2004.

[24] B.Farve, P.A Goupille, *Guide pratique de la sécurité informatique.* Dunod, 2005.

[25] T.Ebrahimi,F.Leprévost, B.Warusfel, *Cryptographie et sécurité des systèmes et réseaux*. Lavoisier, 2006.

[26] S.Ghernaouti-Héli, *Sécurité Internet Stratégies et technologies*. Dunod, 2000.

[27] S.Ghernaouti-Héli, *Sécurité informatique et réseaux.* Dunod, 2005.

[28] C. Liorens, L. Levier, D.Valois, *Tableau de bord de la sécurité réseau.* Eyrolles, 2006.

[29] Gregory D. Speegle, *JDBC: Practical Guide for Java Programmers*. The Morgan Kaufmann Practical Guides Series Editor Michael J. Donahoo, ISBN: 1-55860-736-6.

[30] L. Ferrari. *The Aglets 2.0.2 User's manual.* 2004.

[31] D. LANGE, *Programming Mobile Agent in Java*. Aglet White Paper, IBM Corp., 1996. Disponible sur: http://www.ibm.co.jp/trl/aglets/whitepaper.htm.

[32] D. CHESS, *Itinerant agents for mobile computing*. IEEE Personal communications, Octobre 1995. Disponible sur: http://www.research. ibm.com/massdist/rc20010.ps.

[33] B. Aoued, *Les Aglets d'IBM.* Cours IFT6802, Université de Montréal, 2003.

[34] R.Longeon JL.Archimbaud, *Guide de la sécurité des systèmes d'information*. Centre National de la Recherche Scientifique CNRS Paris, 1999.

[35] D. Burgermeister, J. Krier, *Les systèmes de détection d'intrusions.* Article disponible sur http://dbprog.developpez.com, 2006.

[36] P. Biondi, *Architecture expérimentale pour la détection d'intrusions dans un système informatique.* Thèse de doctorat, 2001.

[37] European Comunities, *Information technology security evaluation criteria.* Juin 1991.

[38] International Standards Organisation (ISO), *OSI Reference Model – Security Architecture*. International Standards Organization Publication, 1989.

[39] M. Abadi; R. Needham, *Prudent Engineering Practice for Cryptographic Protocols*. IEEE Transaction on Software Engineering, Janvier 1996.

[40] International Standards Organisation (ISO), *Information Processing Systems - OSI Reference Model*. International Standards Organization Publication, Octobre 1984.

[41] L. Mé; J. Vazquez; P. Rolin, *Sécurité des systèmes informatiques, des systèmes centralisés aux réseaux*. Réseaux et informatique répartie, Vol.4, Numéro 1, Janvier 1994.

[42] Security Working Group, *Standard of Interoperable Local Area Network (LAN) Security (SILS)*. IEEE, Décembre 1989.

[43] D. Boukhlouf, O. Kazar, *Hybrid Approach based Mobile Agent for Intrusion Detection System: HAMA-IDS.* Journal of Information Security Research, ISSN : 0976-4143 Volume 3, Page: 30-41, March 2012.

[44] D. Boukhlouf, O. Kazar, *Intrusion Detection System: Hybrid Approach based Mobile Agent.* ICEELI'2012 : International Conference on Education and E-Learning Innovations July 1-3 Sousse,Tunisia, 2012.

[45] D. Boukhlouf, O. Kazar, *Hybrid Approach based Mobile Agent for Distributed Intrusion Detection System.* In Proceeding ICDSD'2012 : International Conference on Distributed Systems and Decision, ISSN 2335-1012,Oran Algeria 2012.

[46] J. Kimmins, *Developing a Network Security Architecture : Concepts and Issues*. Proceedings of the SECURICOM91 Conference, Mars 1991.

[47] L. Halme and R. Bauer, *Aint misbehaving - a taxonomy of antiintrusion techniques.* 1995.

[48] Wenke Lee, Rahul A. Nimbalkar, Kam K. Yee, Sunil B. Patil, Pragneshkumar H. Desai, Thuan T. Tran, and Salvatore J. Stolfo, *A data mining and CIDF based approach for detecting novel and distributed intrusions*. In Recent Advances in Intrusion Detection, pages 49–65, 2000.

[49] W. Jansen, P. Mell, T. Karygiannis, and D. Marks, Applying mobile agents to intrusion detection and response, 1999.

[50] JS. Balasubramaniyan, Jose Omar Farcia-Fernandez, David Isacoff, Eugene Spafford, and Diego Zamboni, *An architecture for intrusion detection using autonomous agents*. Technical report, November 1998.

[51]G. Helmer, J. Wong, V. Honavar, and L. Miller, Intelligent agents for intrusion detection, 1998.

[52] G. Helmer, J. Wong, V. Honavar, and L. Miller, *Lightweight agents for intrusion detection.* 2000.

[53] W. Jansen, P. Mell, T. Karygiannis, and D. Marks, *Mobile agents in intrusion detection and response.* Proceedings of the 12th annual Canadian information technology security symposium, Ottawa, Canada; June 2000.

[54] J. Cannady, *Artificial neural networks for misuse detection.* 1998.

[55] Stephane Forrest, Steven A. Hofmeyr, and Anil Somayaji, Computer immunology. Communications of the ACM, 40(10) :88–96, 1997.

[56] SA. Hofmeyr and S. Forrest, *Architecture for an artificial immune system*. Evolutionary Computation, 8(4) :443–473, 2000.

[57] L. Mé and V. Alanou, *Détection d'intrusion dans un système informatique: méthodes et outils*. In TSI, volume 4, pages 429–450. 1996.

[58] JH. Holland, *Adaptation in Natural and Artificial Systems*. PhD thesis, University of Michigan, Ann Arbor, 1975.

[59] L. Mé, *Genetic algorithms, a biologically inspired approach for security audit trails analysis*, 1996.

[60] L. Mé. Gassata, *a genetic algorithm as an alternative tool for security audit trails analysis*. In Proceedings of the First International Workshop on the Recent Advances in Intrusion Detection, Louvain-la-Neuve, Belgium, 1998.

[61] K. Ilgun, *USTAT: A Real-time Intrusion Detection System for UNIX*. Master thesis, University of California, 1992.

[62] P. A. Porras, *State Transition Analysis : A Rule-Based Intrusion Detection Approach*. IEEE Transactions on Software Engineering, pp. 181-199, 1995.

[63] P. Mell, V. Hu, R. Lippmann, J. Haines, M. Zissman, *An Overview of Issues in Testing Intrusion Detection Systems.* National Institute of Standards and Technology ITL, Massachusetts Institute of Technology Lincoln Laboratory, 2003.

[64] J. McHugh, A. Christie, & J. Allen, *Defending yourself: The Role of Intrusion Detection Systems.* Software Engineering Institute, CERT Coordination Center, IEEE Software 2000.

[65] T. H. Ptacek and Newsham T. N. *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection.* Secure Networks Inc., 1998.

[66] M. Hall, and K. Wiley, *Capacity Verification for High Speed Network Intrusion Detection Systems.* Fifth International Symposium on Recent Advances in Intrusion Detection (RAID 2002), Zurich, Switzerland, October 16-18, 2002.

[67] T. F. Lunt, *IDES: An Intelligent System for Detecting Intruders.* In Proceedings of the symposium : Computer Security, Threat and Countermeasures, Rome, Italy, 1990.

[68] D. Anderson, T. Frivold, & A. Valdes, *Next-generation intrusion detection expert system.* 1995.

[69] J.P. Pouzol & M. Ducassé, *Formal Specification of Intrusion Signatures and Detection Rules.* IEEE Computer Security Foundations Workshop (CSFW), 2002.

[70] S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, J. Rowe, S. Staniford-Chen, R. Yip, and D. Zerkle, *The design of grids : A graph-based intrusion detection system.* 1999.

[71] F. Yousfi, *Contribution à la conception d'un système de détection d'intrusions tolérant aux intrusions*. Ecole supérieur des communications de Tunis, 2005.

[72] B. Aoued, *Les Aglets d'IBM.* Université de Montréal Cours IFT6802, 2003.

[73] J. Odell, H.V.D Parunak, B. Bauer. *Extension de UML pour des système basés agents*. 2006.

[74] Java Web site: http://java.sun.com/

[75] D. Dasgupta, F. Gonzalez, K. Yallapu, J. Gomez, R. Yarramsettii, *CIDS: An agent-based intrusion detection system.* Computers & Security, Elsevier, 2005.

[76] A. Vahid Dastjerdi, K. Abu Bakar, *A Novel Hybrid Mobile Agent Based Distributed Intrusion Detection System.* World Academy of Science, Engineering and Technology, 2008.

[77] SA Onashoga , AD. Akinde, AS. Sodiya, *A Strategic Review of Existing Mobile Agent-Based Intrusion Detection Systems.* Proceeding of Issues in Informing Science and Information Technology Volume 6, 2009.

[78] M. Benattou, and K. Tamine, *Intelligent Agents for Distributed Intrusion Detection System*. PROCEEDINGS OF WORLD ACADEMY OF SCIENCE, ENGINEERING AND TECHNOLOGY VOLUME 6 ISSN 1307-6884. JUNE 2005.

[79] B.Trivedi, J.Rajput, C. Dwivedi and P. Jobanputra. *Distributed Intrusion Detection System using Mobile Agents*. International Symposium on Computing, Communication, and Control. ISCCC 2009. Proc .of CSIT vol.1, 2011.

[80] Sodiya, A. S, *Multi-level and secured agent-based intrusion detection system*. *Journal of Computing and Information Technology*, *14*(3), 217-223. 2006.

[81] S. Balasubraniyan, GFJ. Omar, I. David, S. Eugene, Z.Diego, *AAFID - Autonomous Agents For Intrusion Detection.* Technical report 98/05, COAST Laboratory Purdue University, June 1998.

[82] L. Wall; T. Christiansen; R. L. Schwartz. *Programming perl.* 1996.

[83] Deepa, A.J. and Kavitha, V, Efficient intrusion detection system using random tree, Int. J. Enterprise Network Management, Vol. 6, No. 4, pp.275–285, 2015.

[84] A. Mitrokotsa, R. Mavropodi, C. Douligeris, Intrusion Detection of Packet Dropping Attacks in Mobile Ad Hoc Network, TAyia Napa, Cyprus, July 6-7, 2006.

[85] Ahmim, A. and Ghoualmi-Zine, N, A new adaptive intrusion detection system based on the intersection of two different classifiers, *Int. J. Security and Networks*, Vol. 9, No. 3, pp.125–132, 2014.

[86] R. Nakkeeran, T. Aruldoss Albert and R.Ezumalai, Agent Based Efficient Anomaly Intrusion Detection System in Adhoc networks, IACSIT International Journal of Engineering and Technology Vol. 2, No.1. ISSN: 1793-8236, 2010.

[87] L. Portnoy, E. Eskin, and W. S. J. Stolfo, Intrusion Detection with Unlabeled Data using Clustering. In *Proceedings of ACM CSS Workshop on Data Mining Applied to Security(DMSA-2001), Philadelphia, PA*, 2001.

[88] *D.Boukhlouf, O. Kazar, L. Kahloul,* Network Security: Distributed Intrusion Detection System using Mobile Agent Technology. International Journal of Communication Networks and Distributed Systems IJCNDS, volume 16, 2016 (à apparaître).