MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABDELHAMID IBN BADIS - MOSTAGANEM

Faculté des Sciences Exactes et d'Informatique

Département de Mathématiques et informatique

Filière : Informatique

RAPPORT DE FIN D'ÉTUDE

Option : **Réseaux des Systèmes d'Information**

THEME :

Proposition D'un Nouveau Système De Détection D'intrusion

Etudiant(e) : « **Aissat Abderrahmane Kaddour** »
Etudiant(e) : « **Ikkache Mohammad**»

Encadrant(e) : « **Filali Fatima Zohra.** »
Jury: « **Bentaouza C.** »
President: « **Bahness N.** »

Année Universitaire 2020-2021

Remerciement

En premier nous souhaiterons remercier **Dieu** avant tout de nous avoir aidé à terminer ce travail et notre chère professeur madame

« Filali Fatima Zohra » qui était toujours disponible pour nous guider et c'est grâce à elle que nous arrivons à faire tout ce travail et en pour finir sans oublier nos très chers **parents** qui nous ont était d'un aide psychologique et qui n'ont pas arrêté de prier pour notre réussite et **collègues** et **camarades** à qui nous devons aussi des remerciements.

Résumé

Avec l'augmentation du débit des réseaux et des menaces pour la sécurité, l'étude des systèmes de détection d'intrusion (IDS) a reçu beaucoup d'attention dans le domaine de l'informatique. La présence de la technologie de distribution a également facilité le traitement de nombreux problèmes. De nombreuses recherches ont été menées afin d'obtenir le prototype le plus fiable de DIDS (Distributed Intrusion detection System) et elles sont toujours en cours. Certaines ont été testées mais n'ont pas encore atteint la perfection sur les principaux problèmes connus dans la détection d'intrusion, à savoir : la détection en temps réel, la tolérance aux pannes et la gestion des DDoS. Avec l'étude des systèmes d'intrusion et des systèmes de distribution depuis les bases jusqu'au travail actuel, nous avons pu tester un prototype plus élaboré, dans de nombreux cas de détection accompagnés de tests et d'analyses de chaque étape. Ce prototype est expliqué dans ce document après quelques informations sur les systèmes de détection d'intrusion et les systèmes distribués. Enfin, ce prototype est encore en cours de développement pour une détection plus sophistiquée.

Mots-clé

Intrusion, Sécurité, IDS, Système, Distribution, Attaque, réseau, Détection DoS, Ray framework

Abstract

With the increasing amount of network throughput and security threat, the study of intrusion detection systems (IDSs) has received a lot of attention throughout the computer science field. Also with the presence of the distribution technology that made it much easier to handle alot of issues. Many research has been put to work in order to get the most reliable prototype of DIDS (Distributed Intrusion detection System) and they still on. some have been tested but still luck the perfection on the major known problems in the intrusion detection and that is : the real-time detection, fault tolerance and DDoS handling issue. with the study of both Intrusion Systems and Distribution Systems from the basics to the current work, we were able to test a more elaborated prototype, in many case of detection accompanied with tests and analysis of each step. This prototype is explained on this document after some information about intrusion detection systems and distributed systems. Finally this prototype is still under development for more sophisticated detection.

keywords

Intrusion, Security, IDS, System, Distribution, Attack, Network, DoS detection, Ray framework

Liste des figures

Liste des tableaux

Liste des Abréviations

IDS ..... Intrusion detection System
SD ..... Système Distribuée
HIDS ..... Host-based intrusion detection system
NIDS ..... Network Intrusion Detection System
CIDS ..... Collaborative intrusion detection system
DIDS ..... Distributed intrusion detection system
RPC ..... Remote procedure call
APD ..... Appel de procédure à distance
API ..... Application Programming Interface
API ..... Acquittement
TCP ..... Transmission Control Protocol
IP ..... Internet protocol
UDP ..... User Datagram Protocol
P2P ..... Peer to Peer
DOS ..... Denial of service
DDOS ..... Distributed denial of service
LAN ..... Local Area Network
WAN ..... Wide Area Network
VMs ..... Virtual machines
AI ..... Artificial intelligence
AIS ..... Artificial immune system
MAIS-IDS .... Multi artificial immune system-IDS
RFC ..... Request for Comments

Table des matières

Introduction Générale

Il est presque universellement admis que l'Internet joue un rôle de plus en plus important dans nos vies. D'autre part, l'escalade des vulnérabilités et des risques de l'Internet, combinée au problème des piratages, a conduit à un grave problème de sécurité de l'Internet pour les entreprises - en particulier celles qui entrent dans l'arène du commerce électronique.
Assurer la communication entre les utilisateurs, les dispositifs et les applications, ce que l'on appelle le rôle de réseau de l'entreprise, signifie que toute erreur dans le réseau peut exposer l'entreprise à des pertes importantes, en particulier l'aspect sécurité de l'entreprise, qui est presque à l'abri des attaques.

Il est évident qu'un réseau d'entreprise typique possède plusieurs points d'accès à d'autres réseaux, tant publics que privés, le défi consiste à garantir la sécurité de ces réseaux tout en assurant leur accessibilité pour les clients. Les attaques d'aujourd'hui sont si sophistiquées qu'elles sont capables de vaincre les meilleurs systèmes de sécurité, y compris ceux qui n'utilisent que le cryptage ou les pares-feux comme protection. Malheureusement, ces technologies ne suffisent pas à elles seules à contrecarrer les attaques actuelles.

Malgré ces problèmes de sécurité, l'une des solutions pour arrêter et détecter ces attaques est l'IDS. Les systèmes de détection des intrusions surveillent en permanence le réseau pour détecter les incidents potentiels, et enregistrent également les informations et les signalent aux administrateurs de la sécurité. En outre, certains réseaux utilisent les IDS pour identifier les problèmes liés aux politiques de sécurité et dissuader les individus de violer ces politiques. Les IDS sont devenus essentiels à l'infrastructure de sécurité de la plupart des entreprises précisément parce qu'ils sont capables de détecter les attaquants lorsqu'ils tentent de récupérer des informations sur le réseau. La détection des intrusions est définie comme le processus de surveillance et d'analyse des événements survenant sur le réseau afin d'identifier les incidents potentiels, les violations et les menaces imminentes aux politiques de sécurité.

Malheureusement, avec le développement d'attaques telles que le (Distributed Daniel of service), l'attaque par ver, l'attaque par routage, il est devenu difficile de les détecter

par un simple IDS, c'est pourquoi les progrès actuels tendent à développer leur solution en distribuant les dispositifs IDS : au lieu de détecter l'intrusion avec un seul hôte ; l'action sera distribuée sur plusieurs hôtes, ce type de structure permet une analyse plus précise et plus fiable des événements des systèmes à surveiller.

Pour mieux comprendre et approfondir les détails de ces technologies et savoir quels sont les avantages et les inconvénients des IDS, et des systèmes distribués, nous proposons dans ce mémoire quatre chapitres, un chapitre sur les IDS qui aborde une définition, les caractéristiques, la technique de détection, le second chapitre concerne les systèmes distribué, ainsi que les IDS distribués. Le 3ème chapitre aborde notre proposition d'un nouveau IDS basée sur les systèmes distribué, par une introduction, des problématique, la contribution de cette recherche, l'architecture et les algorithmes de cette proposition afin de mieux expliquer la base du système proposé et son fonctionnement. Et enfin un 4ème chapitre sur la réalisation et le test de cette proposition, commencent par une introduction, ensuite présentation de l'environnement de travail matériel et logiciel, puis une section sur l'implémentation, et enfin des tests et évaluations avec une discussion et une conclusion.

Chapitre 1

Les systèmes de détection d'intrusion

1.1 Introduction

Depuis les travaux fondateurs de Denning en 1981 [1], de nombreux prototypes de détection d'intrusion ont été créés. Sobirey tient une liste partielle de 59 d'entre eux [2]. Des systèmes de détection d'intrusion sont apparus dans le domaine de sécurité des ordinateurs en raison de la difficulté d'assurer qu'un système d'information sera exempt de failles de sécurité. En effet, une taxonomie des failles de sécurité de Landwehr et al. [3] montre que les systèmes informatiques souffrent de la sécurité quel que soit leur objectif, leur fabricant ou leur origine, et que cela est également difficile comme économiquement coûteux (à la fois en termes de construction et d'entretien d'un tel système) pour garantir que l'ordinateur les systèmes et les réseaux ne sont pas sensibles aux attaques. Dans ce chapitre, nous allons définir les systèmes de détection d'intrusion et leurs caractéristiques puis nous allons présenter les principaux types d'IDS et les techniques d'intrusion.

1.2 Définition

Un système de détection d'intrusion (IDS) est une application logicielle ou matérielle qui surveille le trafic circulant sur les réseaux et à travers les systèmes pour rechercher des activités suspectes et des menaces connues, en envoyant des alertes lorsqu'il trouve de tels éléments. « L'objectif général d'un IDS est d'informer le personnel informatique qu'une intrusion réseau peut avoir lieu. Les informations d'alerte comprendront généralement des informations sur l'adresse source de l'intrusion, l'adresse cible/victime et le type d'attaque suspectée » Brian Rexroad. [4]

1.3 Caractéristiques des IDS

Tout système de détection d'intrusion (comme illustré dans la figure 1.1), quel que soit son type et sa base de fonctionnement, doit présenter les caractéristiques suivantes [5] :

- Il devrait fonctionner en permanence sans surveillance humaine. Le système doit être suffisamment fiable pour fonctionner en arrière-plan dans le cadre du dispositif ou du réseau observé.
- Il doit être tolérant aux fautes en ce sens qu'il doit être capable de survivre à un crash du système.
- Il doit être résistant aux perturbations, en ce sens qu'il peut se surveiller lui-même pour s'assurer qu'il n'a pas été perturbé.
- Il devrait imposer des frais généraux minimaux au système. Un système qui consomme beaucoup de ressources informatiques ne devrait pas être utilisé.
- Il doit observer les écarts par rapport au comportement standard.
- Il devrait être facilement adaptable au système d'exploitation déjà installé, puisque chacun d'eux a un schéma de fonctionnement différent et que le mécanisme de défense devrait s'adapter facilement à ces schémas.
- Il doit faire face aux changements de comportement du système à mesure que de nouvelles applications sont ajoutées au système.
- Il doit aider à identifier la provenance des attaques et à recueillir des preuves qui peuvent être utilisées pour identifier les intrus.
- Il doit être "difficile à percer" et apporter aux spécialistes de la sécurité "une certaine tranquillité d'esprit".

1.3.1 Méthode de détection

Il y'a deux possibilités pour détecter les actions abusive,la première est de définir et modéliser les actions interdites, cette méthode est appelée les scénarios interdits. La deuxième consiste à définir et modéliser les actions de comportements autorisés.

1.3.2 Méthode de comportement

La notion de profil de comportement fonctionne sous le principe de tout ce qui n'est pas autorisé est interdit.

Avantage

- Approche générique.
- Pas besoin d'une base de signature.
- Permet donc théoriquement de détecter des attaques inconnues.

Inconvénients

- Faux positif : si profil non exhaustif.
- Faux négatifs si :
  — attaques pendant l'apprentissage.
  — modification brutale de l'environnement.
  — modification lente de comportement en cours de surveillance.

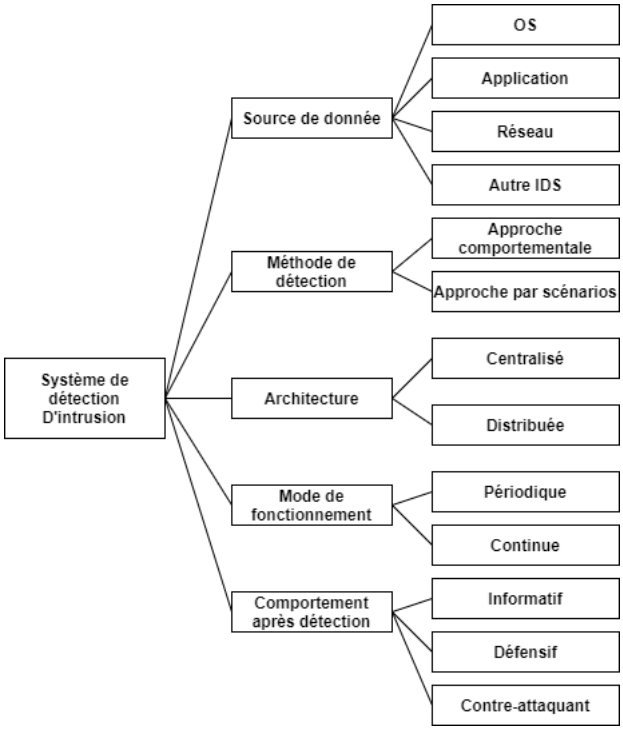

FIGURE 1.1 – Caractéristiques des IDS

1.4 Types des IDS

Il existes plusieurs types des systèmes de détection d'intrusion, catégorisés comme ce qui suit :

1.4.1 HIDS

Système Détection d'Intrusion basée sur l'Hôte (HIDS), est un système de détection d'intrusion installé en tant qu'agents sur un hôte, capable de surveillance et d'analyse des fichiers journaux du système et des applications pour détecter toute activité d'intrus.[6] Certains de ces systèmes sont réactifs, c'est-à-dire qu'ils informent uniquement lorsque

quelque chose est arrivée. Autres sont proactifs, ils peuvent sniffer le trafic réseau arrivant vers un hôte particulier sur lequel l'agent est installé et alerté en temps réel.

1.4.2 NIDS

Les NIDS sont des systèmes de détection d'intrusion qui collectent les données voyageant sur le médias réseau (câbles, sans fil) et les analysent grâce à une base de données de signatures. Si un paquet correspond à une signature d'intrus, une alerte est déterminée ou le paquet est enregistré dans un fichier ou une base de données. [6] Généralement un NIDS capture et analyse le trafic au sein d'un sous-réseau, d'un réseau ou entre un réseau et Internet.

1.4.3 CIDS

Les CIDS représentent des IDS collaboratives permettant d'avoir plusieurs IDS communiquant entre eux afin d'avoir plus de performances. On discute ce type d'IDS plus en détails dans le chapitre 2.

1.5 Technique de détection d'intrusion

Le principe de base de la détection d'intrusion repose sur l'hypothèse que les activités d'intrusion sont sensiblement différentes des activités normales et sont donc détectables [7]. De nombreuses approches de détection d'intrusion ont été suggérées dans la littérature depuis Les travaux d'Anderson [8]. Traditionnellement, ces approches sont classées en trois catégories : détection d'abus, détection des anomalies et détection basé sur les spécifications.

1.5.1 Détection d'abus

La détection d'abus est l'utilisation d'un système d'identification des intrusion grâce a des informations prédéfinie dans une base de donné [9]. Donc elle est pleinement efficace pour découvrir les attaques connues, mais elle est inutile face à des formes d'attaques inconnues ou nouvelles pour lesquelles les informations nécessaire ne sont pas encore disponibles. De plus, toute erreur dans la définition de ces signatures augmentera le taux de fausses alarmes et diminuera l'efficacité de la technique de détection.

Parmi les techniques couramment utilisées dans détection d'abus, on retrouve les techniques de correspondance des modèles, les techniques basées sur des règles, les techniques basées sur les états et les techniques basées sur le Data Mining.

Correspondance de motifs

La Correspondance de motifs (Signature-Based) est de faire correspondre les signatures disponibles dans sa base de données avec les données collectées à partir des activités pour identifier les intrusions [9]. Les approches de détection d'intrusion basées sur la correspondance de modèle sont couramment utilisées dans des systèmes de détection d'intrusion basés sur le réseau dans lesquels des schémas d'attaque sont modélisés, appariés et identifiés en fonction de la tête du paquet, du contenu du paquet ou des deux. Les attaques des modèles être établis dans des systèmes de détection d'intrusion basés sur la concaténation des mots représentant les appels système dans une piste d'audit système. Avec l'émergence continue de nouveaux types et de formes variées d'attaques le nombre des signatures ne cessent de croître, ce qui rend la correspondance de motifs plus chère en termes de coût de calcul.

Techniques basées sur les règles

La technique des règles est l'utilisation d'un ensemble de règles d'implication «si-alors» pour caractériser les attaques informatiques [10]. Les systèmes experts codent les scénarios d'intrusion sous la forme d'un ensemble de règles qui correspondent par rapport aux données d'audit ou de trafic réseau. Tout écart dans le processus de correspondance des règles est signalé comme une intrusion.

Techniques basées sur l'État

Dans cette approche, les IDS tentent d'identifier l'intrusion en utilisant une machine à états finis déduite du réseau. Une activité identifie une intrusion si les transitions d'état dans la machine à états finis du réseau se reflètent dans l'état séquentiel [9]. C'est-à-dire les modèles d'état simplifient la spécification des modèles pour attaques connues et peut être utilisé pour décrire des scénarios d'attaque plus facilement que celles basées sur les règles.

Techniques basées sur le Data Mining

Ces dernières années, des techniques de Data Mining ont été appliquées aux données des IDS réseau et pour construire des modèles de détection d'abus [11, 12, 13]. Dans ce cas, la détection est considérée comme un processus d'analyse de données, dans lequel les techniques d'exploration de données sont utilisés pour découvrir et modéliser automatiquement les fonctionnalités normales ou intrusives du comportement de l'utilisateur.

1.5.2 Détection d'une anomalie

La détection d'anomalie est l'analyse du comportement actuel et la distinction correct avec un comportement normal. C'est-à-dire cette méthode fonctionne en utilisant la définition « Les anomalies ne sont pas normales ». D'autre-parte contrairement à la détection d'abus, la détection des anomalies est dédiée à l'établissement de profils d'activité malveillants du système.

Les études de détection des anomalies commencent par former une opinion sur ce que sont les attributs normaux des objets observés, puis citer quels types d'activités devraient être signalés comme des intrusions. En raison des hypothèses sous-adjacentes aux mécanismes de détection des anomalies, les taux de leurs fausses alarmes sont en général très élevés.

Parmi les techniques utilisé couramment dans la détection d'anomalies on trouve les méthodes basées sur des statistiques, basées sur la distance, basées sur des règles, basées sur des profils et les méthodes basées sur des modèles.

Méthodes basées sur les statistiques

Les méthodes statistiques surveillent le comportement de l'utilisateur / du réseau en mesurant certaines statistiques de variables au fil du temps [9].

Méthodes basées sur la distance

Ces méthodes tentent de surmonter les limites de l'approche de détection statistique des valeurs aberrantes lorsque les données sont difficiles à estimer dans les distributions multidimensionnelles. [10]

Basé sur des règles

Dans les systèmes basés sur des règles, les IDS ont une connaissance du comportement normal de l'utilisateur / du réseau et identifie l'intrusion en comparant ce comportement normal prédéfini aux activités actuelles de l'utilisateur / du réseau.

Méthodes basées sur le profil

Cette méthode est similaire à la méthode basée sur des règles, mais dans ce type, le profil de comportement normal est construit pour différents types de trafic réseau, d'utilisateurs et de tous les périphériques et l'écart par rapport à ces profils signifie une intrusion.

Méthodes basées sur des modèles

D'autres approches basées sur la déviance des comportements normaux et anormaux les modélisent mais sans en créer plusieurs profils [10]. Dans les méthodes basées sur des modèles, les chercheurs tentent de modéliser les comportements normaux et / ou anormaux et l'écart par rapport à ce modèle signifie une intrusion.

1.5.3 Détection basée sur les spécifications

Les approches de détection basées sur les spécifications se concentrent sur la création de comportements spécifiés basé sur le principe le moins privilégié (ou le principe de refus par défaut). Une exécution séquentielle exécutée par le sujet qui enfreint la spécification des programmes sera considérée comme une attaque. Cette approche, en théorie, peut détecter des attaques invisibles qui peuvent être exploité à l'avenir [14]. Cependant, spécifier le comportement d'un grand nombre de programmes privilégiés s'exécutant dans des environnements d'exploitation réels est une tâche décourageante et difficile. Même si la programmation logique inductive est utilisée pour la synchronisation automatiquement. D'autre-part ces approche ne sont ni fondées sur la détection d'abus, ni sur la détection d'anomalies car ils utilisent des spécifications comportementales du système pour détecter les attaques et que toute exécution de système bien comportée doit être conforme à la spécification de son comportement prévu [15]. Au lieu d'apprendre les comportements du système, les systèmes experts déterminent les limites de fonctionnement (seuil) d'une approche de détection. Une fois que le comportement du système correct (ou autorisé) est spécifié, les événements déviant à partir de la spécification génère une alerte [14].

1.5.4 Détection hybride

Les premiers travaux de recherche sur les systèmes de détection d'intrusion suggéraient que les capacités de détection peuvent être améliorées grâce à une approche hybride consistant à la fois la détection de signature (mauvaise utilisation) et la détection d'anomalies [16]. Dans un tel système hybride, la technique de détection de signature détecte les attaques connues et la technique de détection des anomalies détecte les attaques nouvelles ou inconnues.

1.6 Conclusion

Dans ce chapitre, nous avons vu qu'il y'a plusieurs types d'IDS, chacun ayant ses avantages et inconvénients. De plus, chaque IDS est approprié pour un domaine spécifique. Cependant, on retrouve toujours le problème de la puissance pour les IDS et leur temps d'interaction avec les intrusions, ainsi que d'autres problèmes.... Ce qui nous mène à poser la question : est-il possible d'améliorer les IDS ? La distribution permetterait de régler quelques problèmes de ce genre. Ceci constituera le sujet de notre étude pour le chapitre suivant.

Chapitre 2

Systèmes Distribués

2.1 Introduction

Nous avons vu dans la section précédente que les systèmes de détection d'intrusions se déclinaient sous différentes formes, suivant leur placement et le système surveillé. Nous avons mis en avant l'existence des IDS distribués, qui correspondent à une composition de plusieurs IDS, potentiellement hétérogènes. Vasilomanolakis et al. Définissent dans [18] les IDS distribués et collaboratifs comme des systèmes composés de sondes, responsables de la collecte, et de nœuds d'analyse, qui se chargent de la détection d'intrusions. Ce type de structure permet d'analyser plus précisément et avec plus de fiabilité les événements des systèmes à surveiller. Les deux éléments principaux de cette structure sont les collecteurs et les analyseurs. Initialement, ces deux types de composants étaient clairement distingués, mais les dernières propositions de la littérature s'orientent maintenant vers des sondes capables à la fois de collecter et d'analyser les données.

Un système distribué permet d'atteindre des niveaux de performance et de disponibilité importants plus facilement et à « moindre coût » grâce à l'utilisation de plusieurs serveurs en comparaison avec les systèmes centralisés se basant sur l'utilisation d'un gros serveur unique ayant une capacité de traitement supérieure

Mais la construction de systèmes distribués [19] est un défi en soi et souvent, ce sont des solutions uniques sur mesure. Heureusement, les progrès technologiques ont considérablement réduit les défis de construction des systèmes distribués. Tels que la popularité croissante des conteneurs et des orchestrateurs de conteneurs. Ces blocs de construction conteneurisés sont à la base du développement de composants et de modèles réutilisables qui simplifient considérablement et rendent accessibles les pratiques de construction de systèmes distribués fiables.

Nous discutons dans ce chapitre les systèmes distribués, en décrivant la définition de

ces systèmes et leur architecture , ainsi que les avantages et les inconvénients, et en dernier nous tentons de parvenir à une relation entre la distribution et la détection d'intrusion.

2.2 Définition des systèmes distribués

Le Domaine de systèmes distribué est vaste et complexe, pour comprendre le concept de SD, nous allons présenter 2 définitions :

2.2.1 Définition de Tanenbaum

Un système distribué est un ensemble d'ordinateurs indépendants qui apparaît à un utilisateur comme un système unique et cohérent[2]

- Les machines sont autonomes
- Les utilisateurs ont l'impression d'utiliser un seul système.

2.2.2 Définition générale

Un système distribué est un ensemble d'entités autonomes de calcul (ordinateurs, PDA, processeurs, processus, processus léger etc.) interconnectées par un réseau et qui peuvent communiquer [20].

Exemples

- Matériels.
- Traitement parallèle : logiciel avec plusieurs processus sur une même machine.
- Base de données distribuée.

2.3 Architectures de systèmes distribués

Les systèmes distribués doivent avoir un réseau qui connecte tous les composants machines, matériel ou logiciel ensemble afin qu'ils puissent communiquer entre eux. Ils utilisent généralement l'un des quatre types d'architecture ci-dessous :

2.3.1 Le modèle appel de procédure à distance

Appel de procédure à distance (RPC en anglais) est un outil pour construire des applications client-serveur dans un langage de haut niveau, APD est un protocole de demande-réponse. Un APD est lancée par le client, qui envoie un message de demande à un serveur distant connu pour exécuter une procédure spécifiée avec les paramètres fournis. Le serveur distant envoie une réponse au client et l'application poursuit son processus.

Appel de procédure à distance

La figure 2.1 montre le comportement d'une APD. Les deux composantes concernées ont des comportements asymétriques. Lorsque l'invoquer est prêt (1 dans la figure 2.1), il génère une requête à l'invoqué qui détient le code à exécuter et attend une réponse. Une fois que l'invoqué reçoit la demande et parfois (il peut y avoir une liste de demandes en attente si l'invoqué en reçoit beaucoup), celle-ci est traitée (2 dans la figure 2.1) et une réponse est ensuite renvoyée à l'invoquer. Lorsque la réponse est reçue, l'auteur de la demande poursuit son exécution (3 dans la figure 2.1). [21]

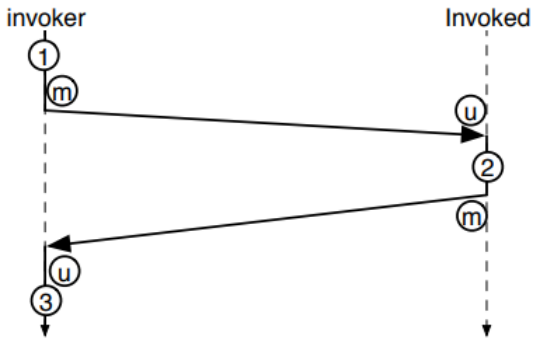

FIGURE 2.1 – l'appel de procédure à distance mécanisme [21]

Le premier message contient le paramètre requis par le code à exécuter. Le deuxième message contenant les valeurs de retour (ou les paramètres modifiés). Lorsqu'il n'y a pas de valeur de retour, un message vide est envoyé car l'APD est un mécanisme synchrone c'est à dire (l'invoqué ne doit pas poursuivre son exécution avant la fin de l'exécution de la procédure à distance).

L'APD repose sur deux principes : la transmission de messages et la communication point à point. C'est le cas pour tout mécanisme de communication de haut niveau qui nécessite l'échange de messages sur le réseau. Un mécanisme comme l'APD vise à structurer les interactions entre les éléments en parallèle à la création de l'APD.

Fonctionnement générale de L'APD

- le programme APD s'enregistre localement auprès de son portmap, en précisant son numéro de programme, sa version et ses fonctions
- le client voulant appeler une fonction de ce programme interroge le portmap pour récupérer le numéro de port TCP UDP

- le client communique avec ce port via des sockets pour demander l'appel de la fonction
- la fonction est appelée sur le programme
- le client reçoit en retour le résultat de la fonction

2.3.2 Le modèle client / serveur

Le modèle client-serveur (voir figure 2.2), ou architecture client-serveur, est un cadre d'application distribué répartissant les tâches entre les serveurs et les clients, qui résident dans le même système ou communiquent via un réseau informatique ou Internet. Le client s'appuie sur l'envoi d'une requête à un autre programme pour accéder à un service mis à disposition par un serveur. Le serveur exécute un ou plusieurs programmes qui partagent des ressources et distribuent le travail entre les clients.

C'est une importante évolution qui augmente la portabilité et permet de faire évoluer les applications distribuées. [21]

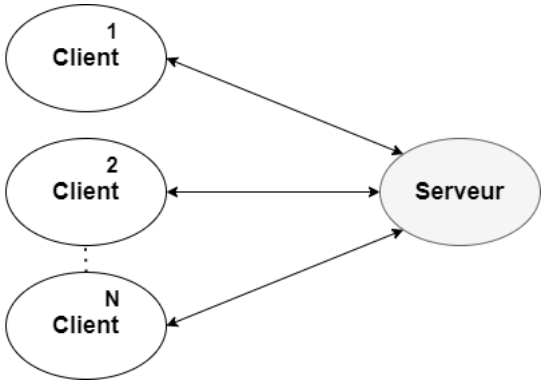

FIGURE 2.2 – L'architecture de modèle client-serveur [21]

Avantages

- **Des ressources centralisées** : étant donné que le serveur est au centre du réseau, il peut gérer des ressources communes à tous les utilisateurs, comme par exemple une base de données centralisée, afin d'éviter les problèmes de redondance et de contradiction
- **Une meilleure sécurité** : car le nombre de points d'entrée permettant l'accès aux données est moins important
- **Une administration au niveau serveur** : les clients ayant peu d'importance dans ce modèle, ils ont moins besoin d'être administrés

- **Un réseau évolutif** : grâce à cette architecture il est possible de supprimer ou rajouter des clients sans perturber le fonctionnement du réseau et sans modification majeure

Inconvénients

- **un coût élevé** dû à la technicité du serveur
- **un maillon faible** : le serveur est le seul maillon faible du réseau client/serveur, étant donné que tout le réseau est architecturé autour de lui ! Heureusement, le serveur a une grande tolérance aux pannes

2.3.3 Le modèle Master / slaves ou Master / Workers

Le modèle master/slaves (voir figure 2.3) est une variante du modèle client-serveur. Dans ce modèle, toutes les initiatives sont prises par le " master " qui fournit des emplois aux " slaves ou Workers".

Dans ce modèle, les " slaves " sont les réactifs et la communication utilise le modèle point à point. Le " master " gère le contexte de la demande et prend les décisions. Généralement, pour éviter les problèmes en cas de crash, le contexte et les données associées sont sauvegardés régulièrement. Si un problème survient, le contexte peut être rechargé et l'exécution continue à partir du dernier point de contrôle (backup time). [21]

Avantages

- La parallélisassions augmente l'efficacité.
- La présence du maître augmente la fiabilité du système.
- Le résultat est garanti. [22]

Inconvénient

- L'architecture est sensible à la fiabilité du maître.
- Il y a des solutions qui ont plusieurs maîtres (backup).
- Les esclaves sont isolés. Il y a un risque de redondance.
- L'architecture n'est pas utilisable sans une capacité suffisante de calcul.
- N'est pas applicable à des problèmes qui ne sont pas décomposables. [22]

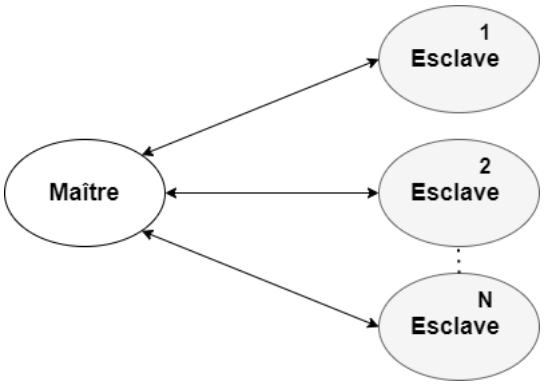

FIGURE 2.3 – L'architecture du modèle master/slaves

2.3.4 Le modèle pair-à-pair

Le pair-à-pair, peer-to-peer ou P2P, définit un modèle de réseau informatique d'égal à égal entre ordinateurs, qui distribuent et reçoivent des données ou des fichiers. Dans ce type de réseau, comparable au réseau client-serveur, chaque client devient lui-même un serveur. Le P2P facilite et accélère les échanges entre plusieurs ordinateurs au sein d'un réseau.

Pour en comprendre le concept, le modèle pair-à-pair n'est pas un moyen de télécharger des fichiers, mais un moyen de structurer des applications distribuées. Son architecture est présentée dans la (figure 2.4). Elle montre qu'aucun composant ne gère les autres. Chaque composant collabore pour atteindre un objectif commun. Pour ces raisons, chaque participant peut être en mesure de communiquer avec les autres. Bien entendu, aucune liaison physique de communication dédiée ne sera établie entre chaque processus d'exécution. [21]

Inconvénient de modèle pair-à-pair

- Ce système n'est pas du tout centralisé, ce qui le rend très difficile à administrer.
- La sécurité est très peu présente.
- Aucun maillon du système n'est fiable.

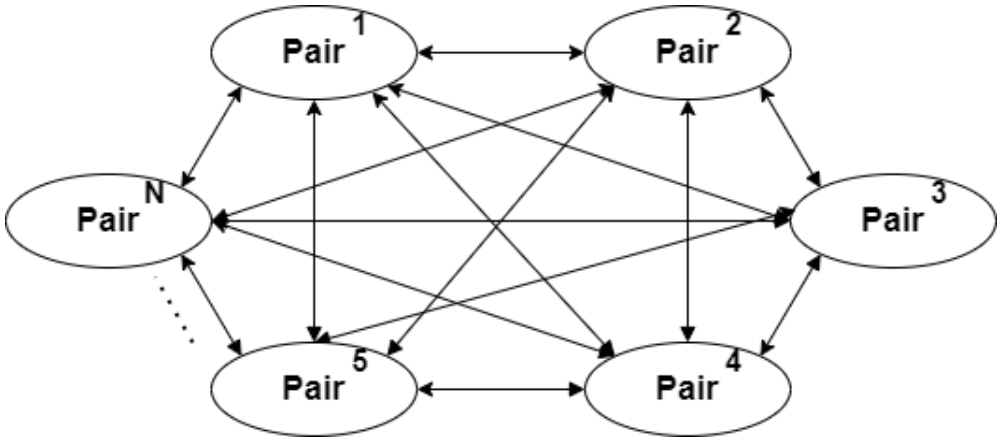

FIGURE 2.4 – L'architecture de modèle pair-à-pair [21]

Les mécanismes de communication sont généralement basés sur la diffusion ou le multicast pour permettre l'échange de données entre les composants. La liaison point à point peut être utilisée lorsqu'une grande quantité de données doit être transmise d'un composant à l'autre pour des raisons de performance (c'est plus efficace).

2.3.5 L'utilisation de ces modèles

Dans certains cas, les systèmes distribués ne suivent qu'un seul schéma d'architecture. C'est généralement le cas des applications client/serveur sur Internet où l'intelligence se trouve dans le serveur. Par contre, pour les applications plus complexes, plusieurs modèles d'architecture peuvent être utilisés à différents stages de l'exécution, voire simultanément lorsque le modèle choisi est plus efficace pour les actions à effectuer.

2.3.6 Choix de l'architecture

La couche application définit le rôle fonctionnel des composants dans un système distribué, qui peut être différent pour chaque composant. Il existe plusieurs architectures courantes utilisées par les systèmes distribués. Le choix de l'architecture peut influencer les considérations de conception décrites ci-dessous :

- **Réactivité** : à quelle vitesse le système répond-il aux demandes ?
- **Débit** : combien de demandes le système peut-il traiter (par seconde, par exemple) ?
- **Répartition de la charge** : les demandes sont-elles réparties uniformément entre les différents éléments du système ?
- **Tolérance aux pannes** : le système peut-il continuer à traiter les demandes en cas de défaillance d'un composant ?

- **Sécurité** : le système garantit-il que les ressources sensibles sont protégées contre les attaques ?

Actuellement, le modèle client-serveur est probablement le plus populaire. Le serveur est responsable de l'acceptation, du traitement et de la réponse aux demandes. Il est le producteur. Le client est simplement le consommateur. Il demande les services du serveur et accepte les résultats.

Mais, comme nous l'avons déjà présenté, il existe d'autres modèles et chacun de ces modèles présente des avantages et des inconvénients.

2.4 Avantages et inconvénients de SD

Comme tout autre système, les systèmes distribués présentent un ensemble d'avantage et inconvénients. L'avantage principal d'un système distribué est de permettre l'évolutivité, les performances et la haute disponibilité des applications on retrouve aussi les avantages suivant [23] :

- **Partage des ressources** : Partage des ressources matérielles et logicielles.
- **Ouverture** : Flexibilité dans l'utilisation du matériel et des logiciels de différents fournisseurs.
- **Concurrence** : Traitement simultané pour améliorer les performances.
- **Évolutivité** : Augmentation de la capacité du système par l'ajout de nouvelles ressources.
- **Tolérance aux pannes** : Possibilité de continuer à fonctionner après une panne

Cependant les systèmes distribués présentent également des inconvénients. La complexité étant son inconvénient principal. Dans un système distribué, il y a plus de machines, plus de message plus de données transmises entre les parties, ce qui entrent divers problèmes tel que [23] :

- **Complexité** : Ils sont plus complexes que les systèmes centralisés.
- **Sécurité** : Ils sont plus sensibles aux attaques extérieures.
- **Gestion** : Plus d'efforts requis pour la gestion du système.
- **Imprévisibilité** : Réponses imprévisibles en fonction de l'organisation du système et de la charge du réseau.

2.5 La distribution du système de détection d'intrusion

Comme les systèmes de détection d'intrusion ont été abordés dans le chapitre précédent, nous allons présenter dans cette partie le principe des IDS distribués, leur architec-

ture, leurs avantages et inconvénients ainsi que les différents travaux existants.

Après le chapitre 1, nous savons ce qu'est exactement l'IDS.

2.5.1 Définition d'un DIDS / CIDS

Un IDS distribué (DIDS) consiste en plusieurs systèmes de détection d'intrusion (IDS) sur un grand réseau, qui communiquent tous entre eux, ou avec un serveur central qui facilite la surveillance avancée du réseau, l'analyse des incidents et les données d'attaque instantanées. En répartissant ces agents coopératifs sur un réseau, les analystes d'incidents, les responsables de l'exploitation des réseaux et le personnel de sécurité peuvent avoir une vision plus large de ce qui se passe sur l'ensemble de leur réseau [24]

Les CIDS sont constitués de plusieurs moniteurs qui font office de capteurs et collectent des données. De plus, ils contiennent généralement une ou plusieurs unités d'analyse qui effectuent la détection effective d'intrusion sur les données obtenues des capteurs. En plus d'améliorer la précision de détection d'un réseau surveillé, les CIDS peuvent également réduire considérablement les tâches complexes des administrateurs de la sécurité. Selon les CIDS, les moniteurs et les unités d'analyse peuvent également être installés dans les mêmes locaux. Sauf indication contraire et pour le reste de cet article, nous supposons qu'un moniteur est une combinaison d'un capteur et d'une unité d'analyse. Les CIDS renforcent la coopération entre les différents contrôleurs et sont donc plus évolutifs que les IDS autonomes. Les CIDS peuvent être classés en fonction de leur architecture de communication, comme le montre la figure 2.5, en CIDS centralisés, décentralisés et distribués [25]

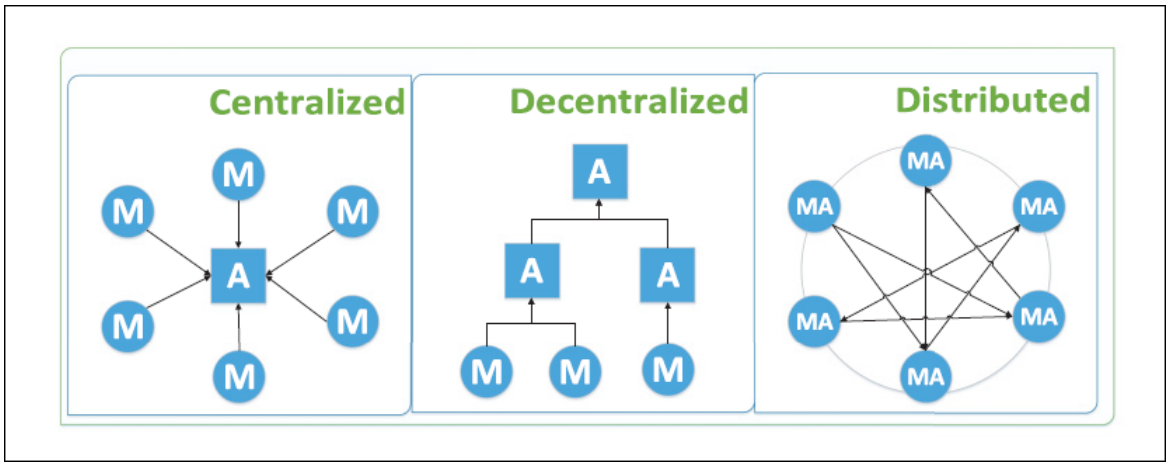

FIGURE 2.5 – L'architectures IDS centralisées, décentralisées et distribuées [25]

Les CIDS centralisés sont constitués de plusieurs moniteurs qui surveillent le comportement de leur hôte respectif ou du réseau trafic passant par là. Ces moniteurs partagent leurs données avec une unité centrale d'analyse. [25]

Les CIDS décentralisés utilisent généralement une structure hiérarchique de points de contrôle ou de multiples déploiements autonomes d'IDS. [25]

Les CIDS distribués partagent les tâches de l'unité centrale d'analyse de manière égale entre tous les moniteurs, de sorte que chaque moniteur est également une unité d'analyse. Ainsi, les CIDS distribués utilisent généralement une architecture Peer-to-Peer (P2P), dans laquelle les données surveillées sont corrélées, agrégées et analysées de manière totalement répartie entre les moniteurs. [25]

L'avantage de l'approche distribuée est sa résistance aux fautes : si un nœud tombe en panne ou est infecté, le système sera tout de même capable de continuer sa tâche de corrélation d'alertes. Toutefois, des avis négatifs sont exprimés sur ce type d'approche. Les critiques qui leurs sont faites portent principalement sur deux points : détection d'attaques imprécise ou répartition de la charge inégale

2.5.2 Les avantages et inconvénients d'un IDS distribué et centralisé

Le DIDS offre à la personne qui analyse l'incident de nombreux avantages par rapport aux autres systèmes IDS monomodes, l'avantage principal est la capacité de détecter des modèles d'attaque sur l'ensemble d'un réseau d'entreprise.

Le deuxième grand avantage est qu'une seule équipe d'analyse peut désormais faire ce qui nécessitait auparavant plusieurs équipes d'analyse d'incidents en raison de la distance physique. (Voir la table 2.1 [26])

IDS	Avantage	Inconvénients
IDS Distribué	- Flexibilité et évolutivité. - Détecte les attaques DOS pour le réseaux à haut débit. - Réduit les couts de calcul. - La surveillance, l'analyse et le traitement des données d'attaque sont plus faciles et plus rapides. - Rendre possible une détection précoce des intrusions. qui peuvent entrainer et blocage du trafic entrant dans les réseaux à haut débit.	- Le flux de données entre l'hote et l'agent peut produire des surcharges de trafic réseau élevées. - Les données dont le chemin est long de leur source à L'IDS peuvent etre interceptées ou modifiées, ce qui peut entrainer des interprétations erronées. - Peut générer des résultats divers à partir de différentes ID.
IDS Centraliser	- Le cout de maintenance et d'administration est inférieur par rapport au cas d'un système distribué. - Toutes les activités de l'IDS sont directement controlées par une console centrale.	- Impossible de détecter des événements malveillants se produisant à différents endroits e meme temps. - Un hacker peut désactiver les programmes fonctionnant sur un système, rendant l'IDS inutilisable ou peu fiable.

TABLE 2.1 – Avantages de l'IDS distribué et centralisé et inconvénients

2.5.3 Architecture de DIDS

L'une des solutions existant de DIDS actuelle est en utilisant un agent autonome, l'architecture DIDS combine la surveillance distribuée et la réduction des données avec l'analyse centralisée des données. [27]

Cette approche est unique parmi les IDS actuels. Les composants de DIDS sont le directeur DIDS, un seul moniteur par hôte et un seul moniteur LAN pour chaque segment de diffusion LAN dans le réseau surveillé. Le DIDS peut potentiellement gérer des hôtes sans moniteur puisque le moniteur LAN peut rendre compte des activités de ces hôtes sur le réseau (voir figure 2.6).

Chaque moniteur hôte et LAN dispose d'un agent de communication unique qui assure l'interface entre le moniteur et le directeur du DIDS. L'agent sert de tampon entre le moniteur local et le directeur du DIDS en traitant toutes les communications à destination et en provenance de l'hôte. Cette conception permet aux composants d'analyse du moniteur de se concentrer sur la détection des intrusions et de ne pas se préoccuper des besoins de communication.

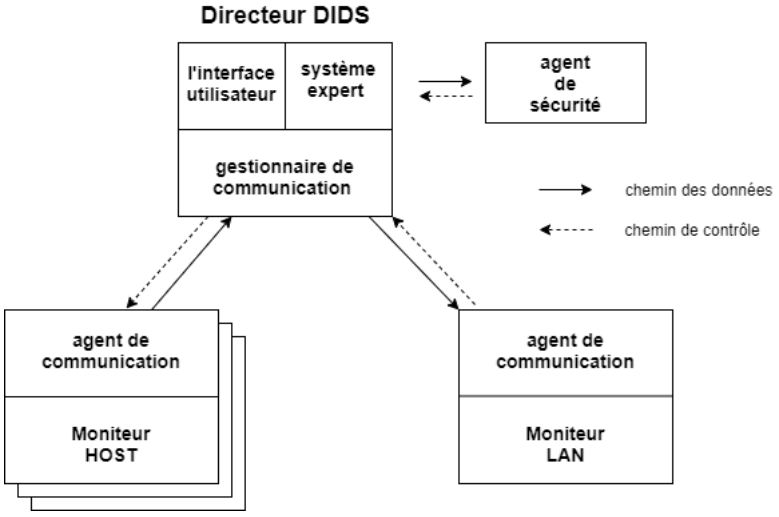

FIGURE 2.6 – L'architecture DIDS d'un agent autonome [27].

2.5.4 Travaux existent

La détection d'intrusion est définie comme le processus consistant à surveiller intelligemment les événements qui se produisent dans un système ou un réseau informatique, en l'analysant pour détecter les signes de violation de la politique de sécurité. L'objectif principal des systèmes de détection d'intrusion (IDS) est de protéger la disponibilité, la confidentialité et l'intégrité des systèmes d'information critiques en réseau [28].

DIDS centralisé

Snapp et ses collaborateurs [29] ont proposé un modèle de DIDS centralisé. Cette méthode fait du directeur du DIDS un point central de défaillance pour cette architecture.

Parmi les avantages de cette solution, on peut citer :

- Le coût de maintenance et d'administration est inférieur par rapport au cas d'un système distribué
- Toutes les activités de l'IDS sont directement contrôlées par une console centrale

Par contre cette solution présente aussi des inconvénients, qui sont :

- Impossible de détecter des événements malveillants se produisant à différents endroits en même temps.
- Un hacker peut désactiver les programmes fonctionnant sur un système, rendant l'IDS inutilisable ou peu fiable.

DIDS AIS distribuée

Hosseinpour et al, [30] ont proposé des DIDS basés sur l'AIS qui utilise un moteur central. Tous les IDS participants sont synchronisés avec ce moteur central qui sert également d'intermédiaire entre deux IDS ayant l'intention de partager entre eux les enregistrements des détecteurs.

Un système immunitaire artificiel (AIS) est une catégorie d'algorithme inspirée par les principes et le fonctionnement du système immunitaire naturel des vertébrés.

Avantage :

— Il est possible de détecter des attaques encore inconnues et d'en générer immédiatement une signature pour la détection.

Désavantage :

— Si la base de données des signatures n'est pas à jour ou si une nouvelle attaque est utilisée pour laquelle il n'existe pas encore de signature, l'IDS ne trouvera rien de suspect [31]

— Les taux de faux positifs et de faux négatifs de détection d'anomalies sont malheureusement beaucoup plus élevés qu'avec la détection d'abus.

DIDS AIS multi-agents

Afzali et Azmi [31] ont proposé une approche AIS multi-agents (MAIS-IDS). Le MAIS-IDS a une plus grande précision de reconnaissance grâce à la collaboration entre les machines virtuelles que les travaux individuels

Les avantages et inconvénients des agents MAIS-IDS

— La mobilité aide le système MAS à fonctionner efficacement avec un faible nombre d'agents. En d'autres termes, au lieu d'utiliser de nombreux agents à travers la région, quelques agents mobiles sont utilisés. Ce mouvement nécessite une coordination et une liaison de ressources, mais pour réduire l'utilisation des ressources de l'architecture MAS proposée, la surcharge de liaison de ressources est négligée. La mobilité des agents entre les machines virtuelles résulte de la migration de l'hôte. [31]

DIDS du réseau d'agents coopératifs

L'une des autres solutions proposées est le réseau d'agents coopératifs, qui est l'une des composantes les plus importantes du DIDS, Un agent est un logiciel qui signale les informations d'attaque au serveur central d'analyse. L'utilisation de plusieurs agents sur un réseau permet à l'équipe d'analyse des incidents d'avoir une vision plus large du réseau que celle obtenue avec un seul système IDS.

Dans l'idéal, ces agents seront situés sur des segments de réseau et des emplacements géographiques distincts (voir la figure 2.7). Les agents peuvent également être répartis sur plusieurs sites physiques, ce qui permet à une seule équipe d'analyse des incidents de visualiser les données d'attaque sur plusieurs sites de l'entreprise. [24]

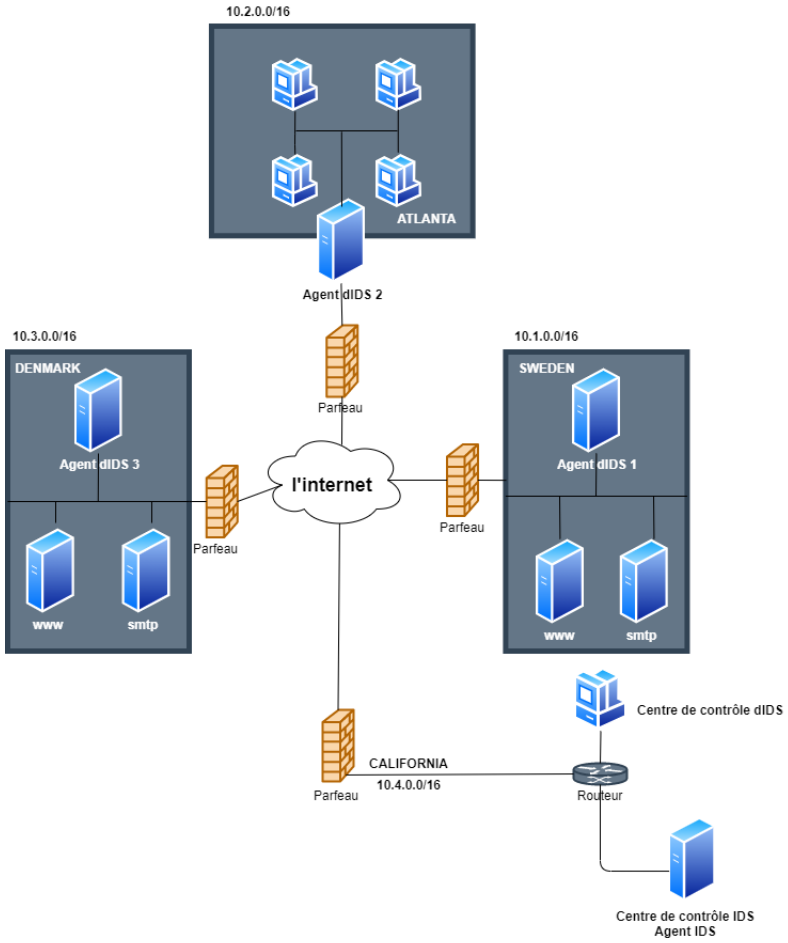

FIGURE 2.7 – L'architecture DIDS sur un réseau d'agents coopératifs.[24]

2.6 L'intérêt de la distribution d'IDS

Comme les attaquants et les méthodes d'attaque deviennent de plus en plus complexes, le besoin d'un système DIDS dans les grands réseaux d'entreprises et militaires augmente considérablement. Avec la complexité croissante de ces attaques, les analystes s'exposent aux problèmes de pannes de communication, où un analyste voit une seule attaque sur son segment et la rejette comme un rien. Alors que plusieurs autres segments reçoivent les mêmes attaques de manière coordonnée, leurs analystes peuvent rejeter la gravité de

l'attaque. Cependant, lorsque toutes les données relatives à l'attaque sont considérées ensemble, une perspective radicalement différente de l'attaque peut apparaître [24]

Donc les systèmes DIDS sont le prochain niveau logique vers lequel les systèmes IDS doivent passer. Ils peuvent être mis en place avec des architectures et des systèmes IDS préexistants, ce qui les rend encore plus efficaces en termes de coûts.

2.7 Analyse et discussion

Les systèmes modernes de détection d'intrusion distribués se concentrent sur la découverte des tentatives d'intrusion qui peuvent provenir de nombreux attaquants et affecter un grand nombre d'hôtes de l'entreprise. Cependant, notre travail se concentre sur le fonctionnement des systèmes inter-connectés et non sur l'efficacité du processus de détection des intrusions lui-même, par exemple l'existence de faux positifs, l'efficacité d'algorithmes de détection spécifiques, etc.

Notre idée pour résoudre ce problème diffère en ce sens qu'elle se base sur une approche NIDS entièrement distribuée où aucun contrôleur central n'est nécessaire et où les NIDS participants (chacun situé au point d'entrée du réseau dont il est membre) communiquent directement entre eux, dans lequel on va distribuer le fonctionnement de l'IDS sur plusieurs hôtes en cas de surcharge du serveur d'IDS principal.

Dans ce qui suit, nous allons tenter de concevoir et mettre en œuvre un prototype de système de détection d'intrusion distribué (DIDS) qui combine la surveillance distribuée et la réduction des données avec l'analyse centralisée des données pour surveiller un réseau d'ordinateurs.

2.8 Conclusion

Il est évident que les IDS ont un rôle très important dans la sécurité des réseaux, mais face au changement des méthodes de pénétration et face au développement des attaques ces systèmes restent toujours faible dans le point ou il faut détecter de nouvelles attaques manuellement et le suivre puis garder les informations nécessaire de ce dernier dans la base de l'IDS ce qui pose le réseaux dans un grave danger pour une période inconnue. Parvenu au terme de notre travail où il était question pour nous de comprendre la notion de système distribué et son rôle, nous avons ainsi entamé notre recherche par une présentation générale des systèmes distribués avec une introduction et une définition, les architectures de SD et ses avantages et inconvénients. Puis, nous avons discuté les IDS distribués : leur définition, avantages et inconvénients les travaux existants. Enfin, nous avons conclu notre travail par une discussion pour faire le point sur notre recherche puis une conclusion.

Chapitre 3

Proposition d'un nouveau IDS

Introduction

Les outils informatiques d'aujourd'hui occupent une place indispensable dans la réussite des entreprises. Cependant, quelle que soit la taille de la structure de sécurité, nous ne sommes pas à l'abri d'attaques. En effet, les attaquants ont adopté de nombreux plans pour réussir à exploiter les vulnérabilités, que ce soit en récupérant des informations sur les systèmes, en injectant une porte dérobée, en volant les informations critiques des utilisateurs, en divulguant les données de l'entreprise... Par conséquent, les administrateurs sont de plus en plus tenus de protéger leurs systèmes informatiques.

Dans les chapitre précédents, nous avons détaillé les systèmes de détection, la distribution, les modèles et les architectures. Tous ces paramètres peuvent influencer la quantité d'informations échangées entre les utilisateurs. De ce fait, nous constatons l'importance de la protection des utilisateurs et la résolution des problèmes de réseau contre les nouvelles attaques par exemple DDoS [32].

Le trafic réseau peut provenir de centaines de milliers d'hôtes indépendants, qui souvent n'informent même pas l'utilisateur que son ordinateur fait partie de l'attaque.

Les détails actuels sur les attaques et leurs protections ont toujours ses inconvénients, c'est pourquoi nous avons eu l'idée de proposer un nouveau système de détection. Dans ce chapitre, nous allons décrire les différentes notions de notre prototype, les algorithmes, l'architecture ainsi que les avantages et les inconvénients qui en découlent.

3.1 Problématique

Pour protéger les systèmes informatiques, des politiques de sécurité [33], doivent être mises en œuvre. Une stratégie de sécurité est un ensemble de règles qui spécifient comment gérer les ressources pour répondre aux exigences de sécurité et quelles opérations sont autorisées et/ou interdites. La politique de sécurité peut être définie par deux caractéristiques : le niveau d'intervention de la politique de sécurité et les outils utilisés pour assurer cette politique. Elle peut intervenir à plusieurs niveaux pour assurer que :

- les utilisateurs peuvent toujours accéder aux données ainsi qu'à la disponibilité des services.
- La confidentialité, pour garantir que les données ne doivent être visibles que par le personnel autorisé,
- L'intégrité qui garantit que les données ne sont pas modifiées par une personne non autorisée, etc.

En raison du développement rapide des attaques, et l'apparition des attaques collaboratives/distribuées faites par des bots automatisés, les systèmes de détection/prévention ont besoin d'une évolution de même niveau ou plus pour être capable d'obtenir les résultats attendus en temps réel et de stopper d'éventuelles nouvelles attaques. Ceci nous oblige à améliorer le fonctionnement des systèmes de sécurité actuels. Dans ce chapitre, nous nous intéressons au système de détection d'intrusions (IDS). Malgré son utilisation répandue, ce système présente plusieurs limites.

1er Problème

Le premier problème avec le système IDS concerne les alertes excessives, la granularité des alertes et la fragmentation des attaques [34]. En fait, IDS génère des milliers d'alertes chaque jour. La méthode d'analyse, l'architecture du système surveillé et les attributs de l'attaque détectée peuvent déterminer le nombre d'alertes générées. Pour certains IDS, lorsqu'un événement unique a plusieurs attributs suspects, une seule alerte n'est pas générée pour l'événement, mais plusieurs alertes isolées (une alerte par attribut suspect) sont générées pour chaque attribut suspect. Il s'agit du problème de granularité des alertes. IDS ne peut pas non plus synthétiser une seule alerte pour représenter le même comportement intrusif. En fait, il divise l'attaque et la considère comme non pas une seule mais plusieurs attaques, c'est le problème de la fragmentation d'attaques.

2éme Problème

Le deuxième problème du système IDS concerne les faux positifs et les faux négatifs [35]. Les fausses alertes négatives sont des alertes qui sont générées en l'absence d'attaque, et les fausse alertes positifs sont des alertes perdues en présence d'une attaque. Les sources de faux positifs et négatifs sont liées à des facteurs internes et externes.

3.2 Contribution

Les raisons qui nous ont amené à essayer d'améliorer les performances du système de détection d'intrusion sont nombreuses, en terme de performance globale du service, un système où les fonctions sont concentrées en un seul endroit, les performances globales du service sont meilleures. En répartissant la charge de calcul entre différents nœuds, la pression sur chaque entité peut être réduite. Cela permet à chaque nœud de fonctionner plus efficacement, améliorant ainsi les performances de l'ensemble du service, nous citons dans ce qui suit notre contribution quant à la problématique abordée.

3.2.1 Optimisation de la détection

Il est très utile de pouvoir ajouter des traitements complexes comme le filtrage ou la détection et la corrélation d'informations afin de réduire la masse de données. Pour cela le temps joue un rôle important dans le réseau informatique, de sorte que s'il y a une charge sur le système telle qu'une attaque par déni de service (DoS) ou autres attaques, il est primordial de prendre en considération chaque milliseconde afin de détecter ces attaques. En effet, la différence entre 2 ms ou 5 ms dans le réseau informatique aura une plus grande influence quant à la détection. C'est pourquoi, nous considérons cela comme l'une des priorités pour le développement de notre système de détection.

Pour éviter ce problème nous proposons d'utiliser ou d'ajouter la notion de distribution dans le système de détection de sorte à répartir la charge de calcul et la partager entre les différents agents, c'est-à-dire à un certain niveau si le serveur reçoit plusieurs requêtes en même temps alors le serveur lui-même va diviser où plutôt distribuer le travail sur les autres nœuds en même temps et c'est ce qu'on appelle la synchronisation, ainsi la charge va être réduite et sera plus efficace.

3.2.2 Extensibilité/évolutivité

La Extensibilité ou scalabilité en anglais est la capacité des équipements informatiques à s'adapter au rythme de la demande. Sachant que la capacité du matériels est fixe, le nombre de requêtes pouvant être traitées dans un temps donné est limité. Si cette limite est dépassée, les demandes seront multiplexées et la concurrence entre les demandes traitées en même temps entraînera un temps d'attente supplémentaire, qui deviendra un goulot d'étranglement[36]. (voir figure 3.1).

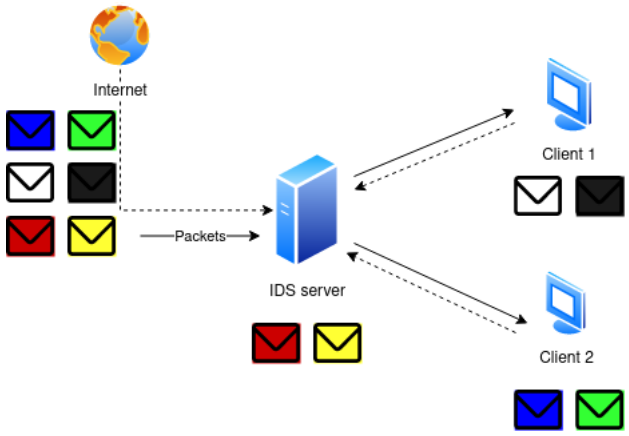

FIGURE 3.1 – Distribution des paquets sur les client

Pour notre cas, on a pas de demandes mais plutôt des attaques pouvant endommager le système. En général, il existe toujours des limites dans le nombre d'intrusions pouvant être détecter ou traiter, c'est pour ça que l'évolutivité est une caractéristique essentielle pour tous les services réseaux. Ainsi, à l'instant où le système ne peut plus détecter d'attaques, il faudrait ajouter des ressources afin de contourner le problème de surcharge.

3.3 Proposition de la solution

3.3.1 Description

Après diverses recherches, nous proposons dans ce qui suit notre solution qui est un système de détection distribué. Ce système traite les paquets un par un grâce au décryptage de chaque paquet, en séparant les entêtes de chaque couche du modèle TCP/IP Ensuite, il vérifie chaque champ s'il a été bien formulé et s'il n'a pas été modifié ou généré avec un outil de malveillance. Puis il vérifie les données existantes dans le paquet si elles contiennent une suite de bytes spécifiques ou non. D'autre part, il est possible d'intègre dans ce système l'apprentissage machine. De plus, notre proposition utilise la distribution pour offrir plusieurs avantages soit dans la détection, apprentissage machine. La figure (3.2) correspond à une proposition initiale de note solution.

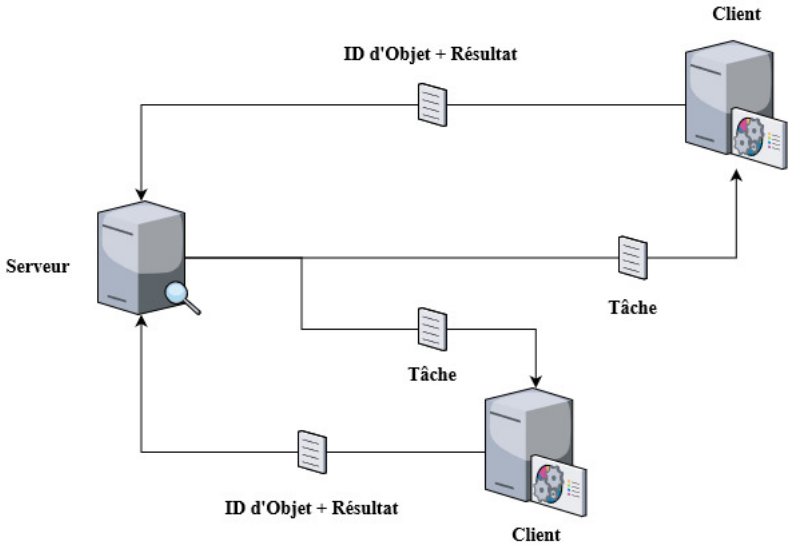

FIGURE 3.2 – Proposition initiale

3.3.2 Intégration de la distribution

Le système de détection d'intrusion basé sur l'hôte a été le premier IDS à apparaître. Lorsque le premier système de détection a été inventé, l'environnement cible était le système sur lequel tous les utilisateurs opéraient. L'interaction avec l'extérieur du système est très rare, ce qui facilite la tâche du système de détection d'intrusion. Le système de détection d'intrusion analyse les informations d'audit fournies par l'ordinateur local ou central sur un ordinateur séparé et signale tout événement suspect.

Avec l'avènement des réseaux informatiques, plusieurs prototypes de systèmes de détection d'intrusions ont été développés pour répondre à ces nouvelles exigences. notre essai dans ce domaine est de créer un IDS communicant à partir des IDSs basés système. Dans un environnement distribué, les utilisateurs passent d'un ordinateur à un autre, de sorte qu'ils lancent des attaques sur plusieurs systèmes. Par conséquent, le système de détection d'intrusion local doit échanger des informations avec ses égaux. Cet échange d'informations se produit à plusieurs niveaux, que ce soit l'échange d'enregistrement d'audit, l'émission d'alertes ou le partage de paquets.

La distribution a énormément influencé le domaine de l'informatique, et a servi dans la résolution de beaucoup de problèmes et l'amélioration des services. Même les Hackers ont profité de la distribution pour rendre leurs attaques plus puissantes et efficaces. Mais le domaine de détection d'intrusion est resté traditionnel et centralisé.

L'intégration de la distribution permet d'exploiter les ressources des serveurs/machines au profit d'optimisation de la sécurité. L'IDS distribué a pour objectif de minimiser le temps de détection et l'impacte des attaques. Plusieurs chercheurs actifs proposent des IDS distribués, cependant, une révolution majeure est nécessaire pour la sécurité.

Choix du Modèle de distribution

Il est important de connaître la configuration de base pour réaliser la solution. La distribution doit être liée à des modèles d'architecture et l'un d'entre eux est le modèle client /serveur, comme nous l'avons vu dans le premier chapitre le modèle client/serveur est une structure d'application distribuée dont la caractéristique décrit la relation des programmes coopérants dans une application ou même des services, le composant serveur fournit une fonction ou un service à un ou plusieurs clients et ce dernier renvoie le résultat au serveur[37]. (Voir figure 3.3).

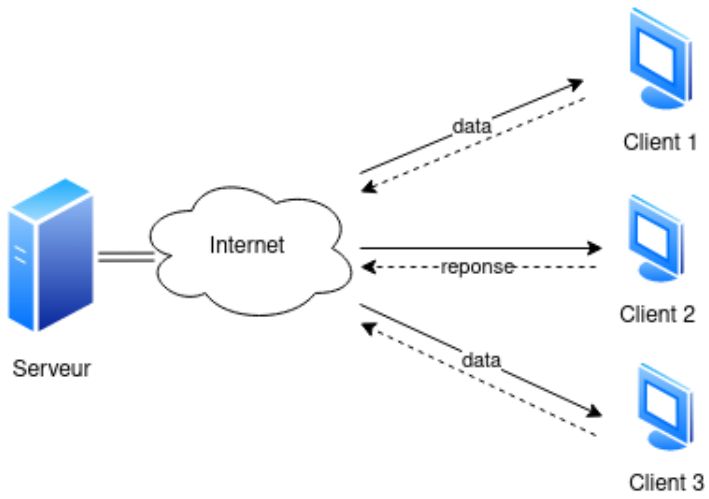

FIGURE 3.3 – Modele de base (Client/serveur)[37]

La notion de modèle client/serveur de l'informatique distribuée est très importante dans notre solution qui se concentre spécifiquement sur le partage des ressources et des informations entre les composants avec les avantages suivants :

- Il s'agit d'une architecture de système très ouverte qui permet d'ajouter de nouvelles ressources selon les besoins.
- Le système est flexible et évolutif.

D'autre part, nous nous concentrons sur le système qui peut distribuer les données et les travaux entre les clients, comme un intergiciel ou un cadre de travail (framework) qui permettra cette distribution.

3.4 Architectures

Pour comprendre l'idée de nos propositions sur les IDS, nous proposons une architecture qui peut faciliter une analyse approfondie.

3.4.1 Architecture globale de l'IDS

La figure 3.4 s'appuie sur un cadre architectural, qui se compose des éléments suivants :

- Le serveur central est responsable de la collecte des données (paquets), de leur contrôle et de leur distribution.
- Les workers sont responsables de la réception des données et de leur décryptage.
- WAN pour le réseau internet
- LAN pour le réseau local.

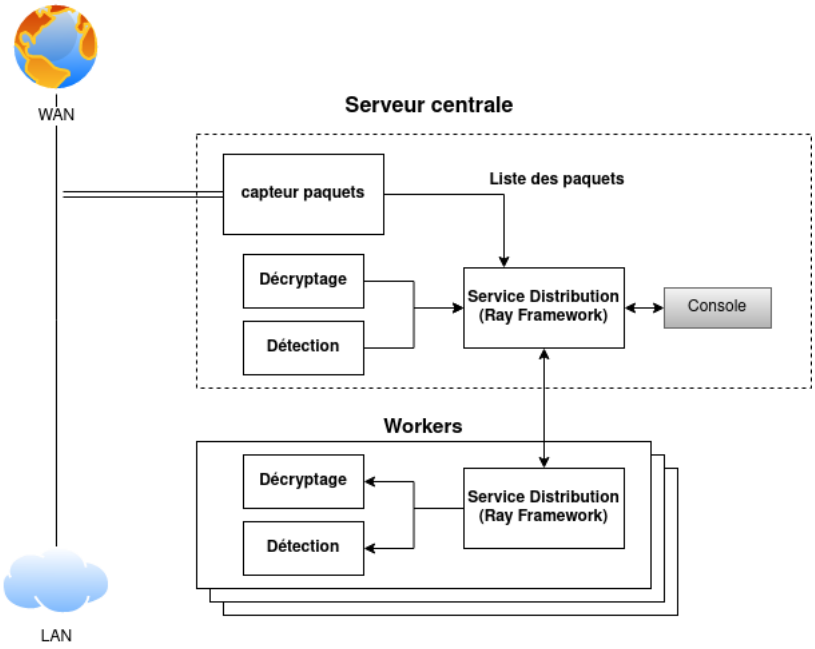

FIGURE 3.4 – Architecture globale sur la proposition de la distribution des IDS avec le framework Ray

L'architecture globale contient des services spéciaux qui sont essentiellement utilisés dans notre système, et sont les suivants :

Capture paquets

Le service de capture paquets consiste à capturer le trafic du réseau informatique c'est à dire pour capturer les paquets d'entrée venant d'internet sur un serveur qui est le serveur central.

Ray Framework

Les applications d'IA émergentes doivent être plus rapides, plus simples et plus efficaces, et pas seulement autonomes. Ces applications doivent interagir régulièrement avec leur environnement et apprendre des retours d'informations pour se développer en même temps. Par conséquent, dans un contexte de flexibilité et de haute performance, les systèmes distribués sont maintenant les piliers de l'IA émergente. Une telle plateforme d'exécution distribuée récemment développée est le framework Ray (Voir figure 3.5), parfois aussi appelé Ray.[38]

Ray est un moteur d'exécution distribuée basé sur Python. Le même code peut être exécuté sur une seule machine pour obtenir un multitraitement efficace, et il peut être utilisé sur un cluster pour les gros calculs.[38]

Ray sera utile pour construire un ensemble d'applications qui nécessitent une décision rapide, comme ce qui est requis pour notre système de détection. L'idée était donc d'utiliser le framework Ray avec le système de détection d'intrusion qui aidera à distribuer les tâches sur les différents nœuds, nous verrons plus de détails sur le Ray dans le chapitre IV.

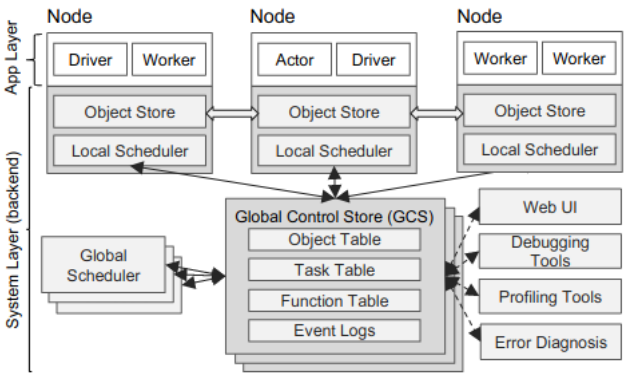

FIGURE 3.5 – Ray's architecture.[38]

Figure 3.5 se compose de deux parties : une couche d'application et une couche système. La couche applicative met en œuvre l'API, la couche système met en œuvre l'ordonnancement des tâches et la gestion des données pour satisfaire aux exigences de performance et de tolérance aux pannes[38].

Décryptage

Le décryptage est la transformation d'un cryptogramme en texte lisible, par l'utilisation d'un algorithme cryptographique et d'une clé. l'idée est de décrypter les paquets pour

que ce soit clair à analyser et à détecter.

Détection

Ce service consiste à détecter l'attaque d'intrusion par le serveur ou l'un des travailleurs (workers) qui est un message clair, ce message ne peut pas être détecté jusqu'à ce qu'il soit décrypté.

Fonctionnement

Le serveur commence à surveiller le réseau et regroupe les données en une liste de paquets. La deuxième étape qui est la plus importante est la distribution des données aux workers, par le framework Ray. D'autres part le worker reçoit les paquets pour les décrypter ou détecter une intrusion. A la fin, le worker retourne le résultat au serveur central. Nous pouvons également faire en sorte que le serveur central soit un worker en même temps. Pour bien comprendre, veuillez voir la figure 3.6. Le lien noir signifie qu'il y a une distribution entre les workers et le serveur.

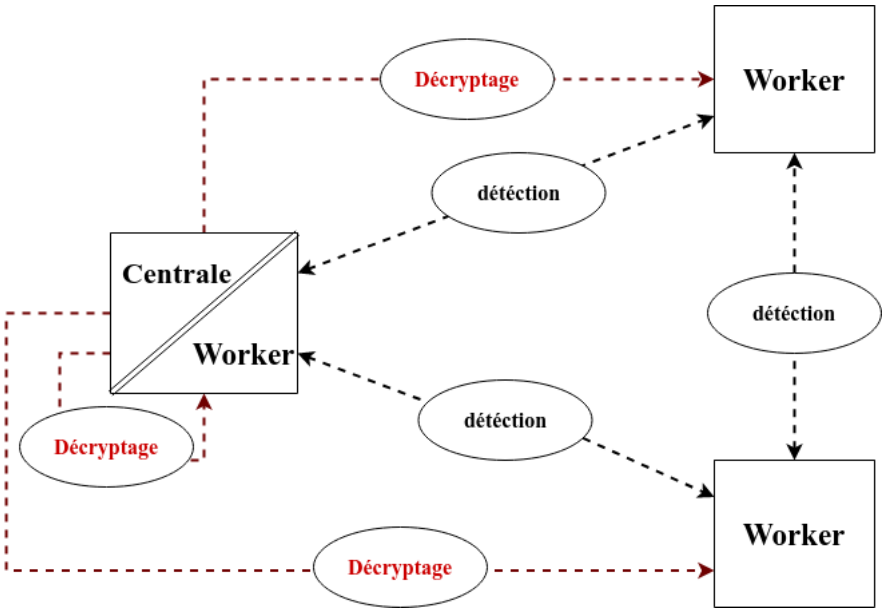

FIGURE 3.6 – Architecture globale sur la proposition de la distribution des IDS avec le framework Ray.

3.4.2 Architecture de distribution

Pour approfondir notre idée, nous présentons la deuxième partie sur l'architecture incluant le core du système proposé, la figure 3.7 montre une vue plus détaillée.

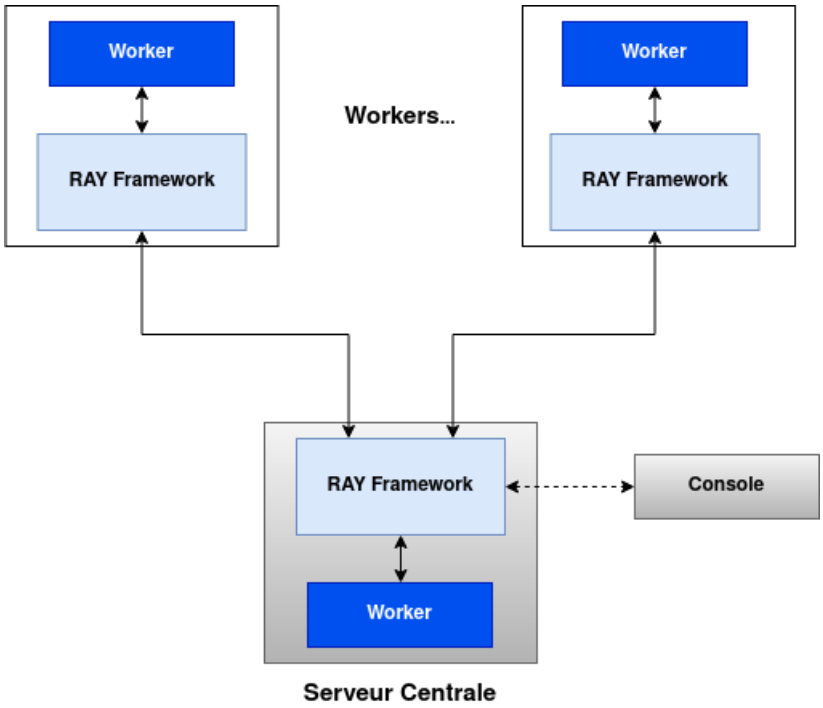

FIGURE 3.7 – Architecture globale sur le core de la distribution d'IDS avec le framework Ray.

Cette architecture représente une console centrale liée à deux ou trois workers utilisant le framework Ray. Chaque worker a le framework Ray dont le rôle est de distribuer les données entre ces workers, nous verrons plus de détails sur le framework Ray dans le chapitre IV.

3.5 Algorithme

Notre proposition se compose de 4 algorithmes : captures des paquets, décryptage des paquets, détection d'intrusion et distribution. Les détails de ces différents algorithmes sont comme suit :

3.5.1 Capture des paquets

D'abord le programme a besoin de la bibliothèque socket. grâce à cette bibliothèque on aurait l'accès à plusieurs fonctionnalités du système concernant le réseau. Le programme commence ensuite dans une boucle infinie avec une condition de vérification si un nouveau paquet vient d'arriver ou non. Enfin le paquet est stocké dans une liste des paquets pour le passer à la distribution afin de le décrypter et détecter l'intrusion. la figure 3.8 illustre cet algorithme.

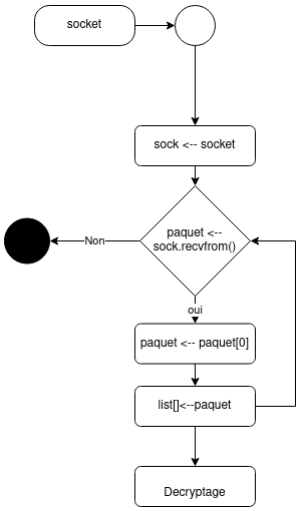

FIGURE 3.8 – Algorithme de capture des paquets

3.5.2 Décryptage des paquets

Pour décrypter les paquets il faut inclure les bibliothèques ”sys” et ”struct”, pour nous aider à avoir des privilèges administrateurs et déballer la suite de bits d'un paquet. Enfin il faut avoir la liste des paquets générés dans la capture pour décrypter les paquets un par un stockés dans la liste. Le fonctionnement commence par une boucle qui porte la condition (si i est inférieure à la taille de la liste des paquets). Ensuite, une variable paquet reçoit l'élément i de la liste, et passe cette variable à des fonctions de décapsulation des entêtes, puis incrémente i pour passer au paquet suivant de la liste. la décapsulation est basé sur le modèle TCP/IP. Enfin les données de chaque entête sont passées à la détection d'intrusion (Plus de détails dans ”3.5.5 Description du fonctionnement par un exemple”). La figure 3.9 illustre cet algorithme.

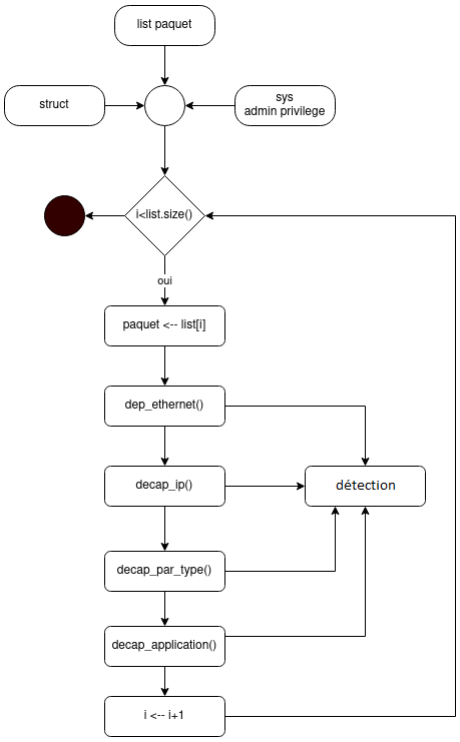

FIGURE 3.9 – Algorithme de décryptage des paquets

3.5.3 Détection d'intrusion

La figure 3.10 illustre la détection d'intrusion qui commence par l'inclusion des bibliothèques strict et socket, et la réception des entêtes d'un paquet à chaque fois. Puis l'entête reçu passe par des conditions de vérification du type de l'entête afin de contrôler la forme de l'entête dans la fonction d'analyse des données et confirmer que l'entête est généré à travers une demande de client ou un contact entre un client et un serveur et non pas un paquet généré par un outil (les standards RFC appliqués dans cette fonction). Enfin si l'entête reçu est inconnue, une alerte est générée aux experts pour investiguer le cas le plus vite possible. Sinon des notifications sont envoyées à partir de la fonction d'analyse qui peuvent être une alerte, un avertissement ou rien du tout.

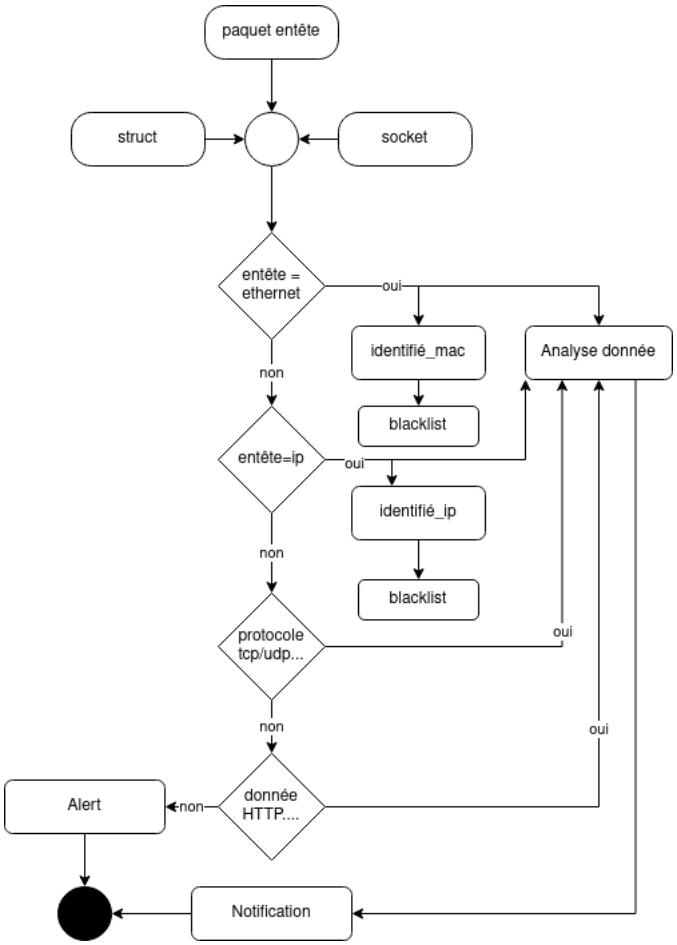

FIGURE 3.10 – Algorithme de détection d'intrusion

3.5.4 Distribution

Le framework Ray commence par recevoir les fonctions de détection et décryptage pour les distributions sur le réseau, ainsi que la liste des paquets pour le passer aux workers dans le moment de décryptage des paquets. Au début le service vérifie si le serveur est actif ou non, puis reçois la liste des workers disponibles pour distribuer les tâches. S'il n'y a pas de workers, le serveur réagi en tant que serveur et worker en même temps. S'il y a une liste de workers, le serveur commence par vérifier si les workers sont libre un par un, le worker libre (i) reçois un paquet avec la fonction de décryptage, après le décryptage du paquet le worker (i) vérifie s'il y a un autre worker (j) libre ou non. S'il y a un worker (j) libre, il le passe à la fonction de détection avec le résultat du décryptage. Ce dernier revoie la résultat final au serveur, sinon le worker lui-même applique la fonction de détection sur les résultats obtenus et renvoie le résultat final au serveur, comme illustré dans la figure 3.11.

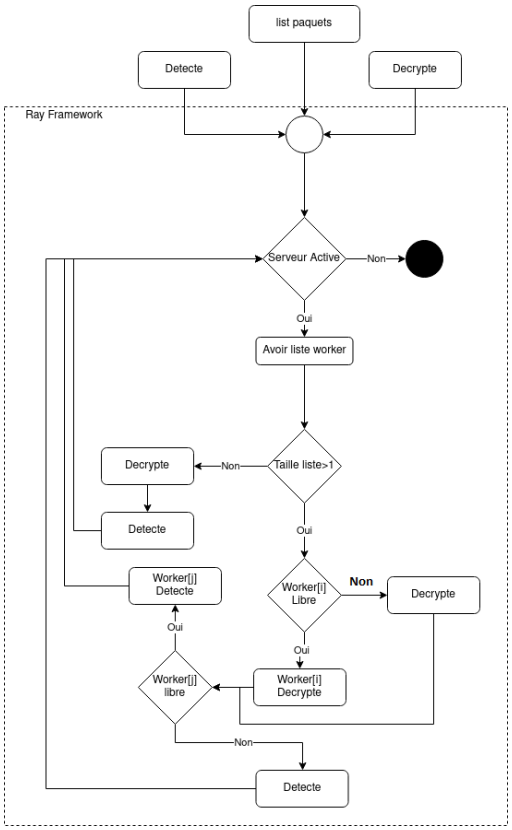

FIGURE 3.11 – Algorithme de distribution

3.5.5 Description du fonctionnement par un exemple

Pour expliquer notre solution plus en détails, nous avons choisit d'illustrer le fonctionnement de notre système par un exemple. A cet effet, nous avons choisit d'utiliser l'en-tête du protocole TCP. La figure 3.12 montre la forme de l'entête du protocole TCP. Le décryptage ici sépare chaque élément du protocole à part et stocke ce dernier dans une liste pour être traité par la fonction d'analyse. Ensuite dans la détection la fonction d'analyse vérifie chaque élément de l'entête ajouté dans la liste reçu. Par exemple, suivant le standard RFC l'élément (URG... FIN). "Le TCP doit récupérer des données endommagées, perdues, dupliquées ou livrées hors-service par le système de communication internet. Ceci est réalisé en attribuant un numéro de séquence à chaque octet transmis, et nécessite un accusé de réception positif (ACK) de réception TCP" [RFC 793]. En se basant sur cette fonctionnalité la fonction d'analyse vérifie les données perdues endommagées, et confirme que la communication entre le serveur et le client dans le paquet est valide à travers les numéros de séquence. Si une donnée est ambiguës, l'analyse retourne une alerte de détection d'un paquet contenant des données ambiguës dans l'entête TCP. Enfin le fonctionnement se base toujours sur les standards RFC et le modèle TCP/IP.

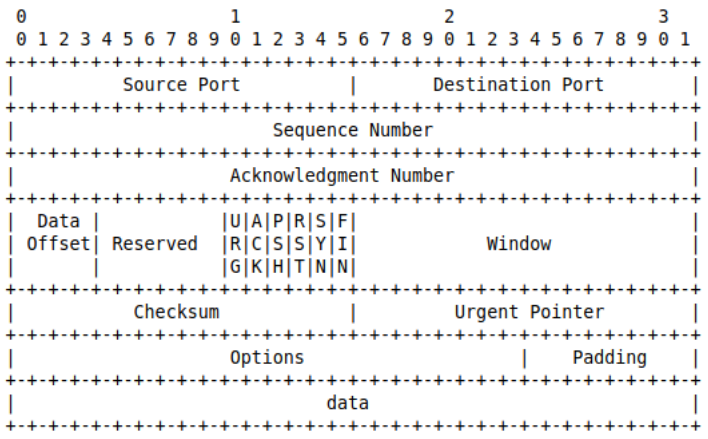

FIGURE 3.12 – Entête TCP (RFC 793) [39]

3.6 Conclusion

En conclusion, nous nous sommes concentrés sur les principaux problèmes du système de détection les plus utilisés tels que les alertes excessives et les fausses alertes. Ce genre de problèmes pose actuellement plusieurs inconvénients sur le système plus précisément sur les performances du service en ajoutant également la vitesse de détection et la tolérance aux pannes, et pour cela, nous avons fait une proposition préliminaire du fonctionnement de détection similaire à l'IDS pour améliorer la performance totale et d'après nos recherches nous avons pensé à la notion de distribution, dont nous croyons que ça rendra l'IDS plus efficace et plus puissant, exempt de certains défauts, et fournira une haute disponibilité et une meilleure performance de service.

Dans ce chapitre nous avons commencé par une introduction suivie de la problématique. Nous nous sommes concentrés ensuite sur la description de notre proposition suivie par les architectures et les algorithmes utilisés, et dans le chapitre suivant nous finaliserons le travail par une petite démonstration sur l'implémentation de notre proposition en ajoutant une description détaillée des différentes parties.

Chapitre 4

Réalisation et test

4.1 Introduction

Cette partie a pour objet d'analyser les données, de livrer les principaux résultats du mémoire et de fournir une discussion portant sur la contribution du travail. Il s'agit d'apporter une véritable réflexion sur le sujet par la confrontation de la pratique à la théorie. Et pour cela ce chapitre consiste à présenter, analyser et interpréter les résultats de notre proposition, y compris l'environnement de travail et une explication complète de tous les concepts, en terminant par des futures perspectives.

4.2 Environnement de travail

Pour mener à bien une recherche ou un prototype de recherche, il est toujours nécessaire de disposer d'un laboratoire qui fournit les tests et les analyses nécessaires à la recherche, de sorte que le résultat final soit satisfaisant.

4.2.1 Matériel

Dans le cadre de notre recherche, afin de fournir un prototype, d'effectuer des diagnostics et de surveiller les systèmes, nous avons utilisé du matériel (ordinateurs personnels), et nous avons également utilisé la technique de virtualisation afin de pouvoir disposer de travailleurs pour notre proposition qui nécessite plusieurs clients (Workers) pour faciliter la distribution des tests et obtenir un résultat de détection d'intrusion distribuée. On a utilisé deux ordinateurs personnels avec les capacités suivantes :

HP Pavillion 250	HP Pavillion 230
CPU :I5 4600U 4 Core RAM : 16Go OS : Ubuntu Desktop 20.04LTS amd64	Pentium N3530 2 Core 4Go Ubuntu Desktop 20.04LTS amd64

Table 4.1 – Matériel Personnel

Vmware Machine 1	Vmware Machine 2
CPU :I5 4600U 2 Core RAM : 4Go OS : Ubuntu Serveur 20.04LTS amd64	Pentium N3530 2 Core 4Go Ubuntu Serveur 20.04LTS amd64

Table 4.2 – Machines Virtuel

4.2.2 Logiciel

Développement

- **Geany** :
  Geany est un éditeur de text dédié au programmeur. Il offre des fonctionnalités satisfaisantes qui facilitent la programmation, et il dans le flux de travail. Geany fonctionne sous tous les system d'exploitation (Distribution Linux, Windows, MacOS) ainsi qu'il support plus de 50 langages de programmation. URL : `https://www.geany.org/`

- **Ubuntu Desktop :**
  Ubuntu Desktop est un système d'exploitation GNU Linux basé sur Debian. Il est développé, commercialisé et maintenu pour les ordinateurs individuels par la société Canonical.
  Enfin Il est possible de télécharger Ubuntu Desktop à partir du lien suivant :`https://ubuntu.com/download/desktop`

- **Ubuntu Serveur :**
  La version Ubuntu Serveur est différente que la version desktop, la différence est l'absence d'environnement graphique sur la version serveur, ainsi le processus d'installation, les options différentes du noyau (Entré et Sortie, Mode Préemptif, fréquence d'interuption).
  URL : `https://doc.ubuntu-fr.org/ubuntu_server`.
  Finalement Il est possible de télécharger Ubuntu Serveur à partir du lien suivant :`https://ubuntu.com/download/server`

- **Vmware Workstation :**
  Vmware est un outil de virtualisation des postes de travail créé par la société VMware, utilisé généralement comme laboratoire afin de tester les nouveaux logiciels développés, ou pour tester les architectures complexes avant une installation réelle sur une machine physique. Vmware Workstation est une version commerciale payante gratuite pour une duré de 30 jours.

- **Ray Framework** :
  Ray est un moteur d'exécution distribué basé sur Python. Le même code peut être exécuté sur une seule machine pour obtenir un multitraitement efficace, et il peut être utilisé sur un cluster pour les gros calculs. Il offre aussi plusieurs avantages pour le développement des applications de l'apprentissage machine. [38] Pour installé Ray Framework il faut suivre les étape suivantes : Commencé par installer le Installeur-pip du python grâce a la commande (Linux) :

  Terminal : *sudo apt-get install python-pip*

  Puis, il suffit simplement d'exécuter la commande suivante sur le terminal pour installer Ray :
  Terminal : *pip install ray*

  Remarque : *Ray doit être installé dans toutes les machines pour que la distribution fonctionne correctement et doit être de la mème version*

Langage de programmation

Cette section présente le langage de programmation que nous avons utilisé pour créer un code simple afin de tester notre recherche.

- **Python :**
  "Python a été créé au début des années 1990 par Guido van Rossum au Stichting Mathematisch Centrum (CWI, voir `https://www.cwi.nl/`*) aux Pays-Bas, comme successeur d'un langage appelé ABC. Guido reste l'auteur principal de Python, bien que le programme comprend de nombreuses contributions d'autres personnes".*
  `https://docs.python.org/3/license.html`

4.3 L'implémentation

4.3.1 L'architecture utilisée

La Figure 4.1 présente une architecture du travail pour simplifier et comprendre ce que nous avons fait dans notre implémentation.

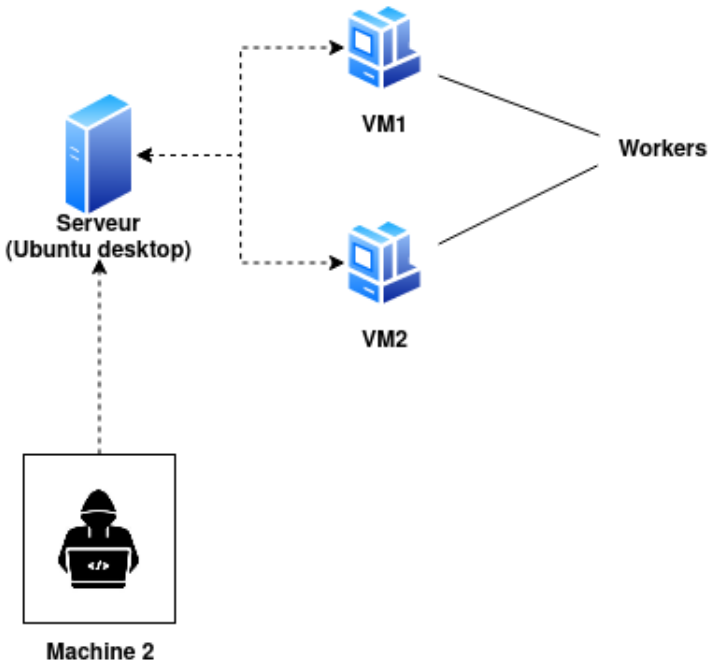

FIGURE 4.1 – Architecture de Test

Au début on a deux machines personnelles, la 1ére machine opère comme un serveur de distribution (Ubuntu desktop), cette dernière contient deux machines virtuelles (Ubuntu serveur). Ces machines virtuelles jouent le rôle d'un Worker. Toutes ces machines sont connectées entre elles dans un sous réseau local. D'autre part la deuxième machine personnelle agit comme un outil d'attaque, afin de tester la détection et la distribution.

Tout d'abord, nous commençons par présenter la configuration de la VMs (Voir la figure 4.2) :

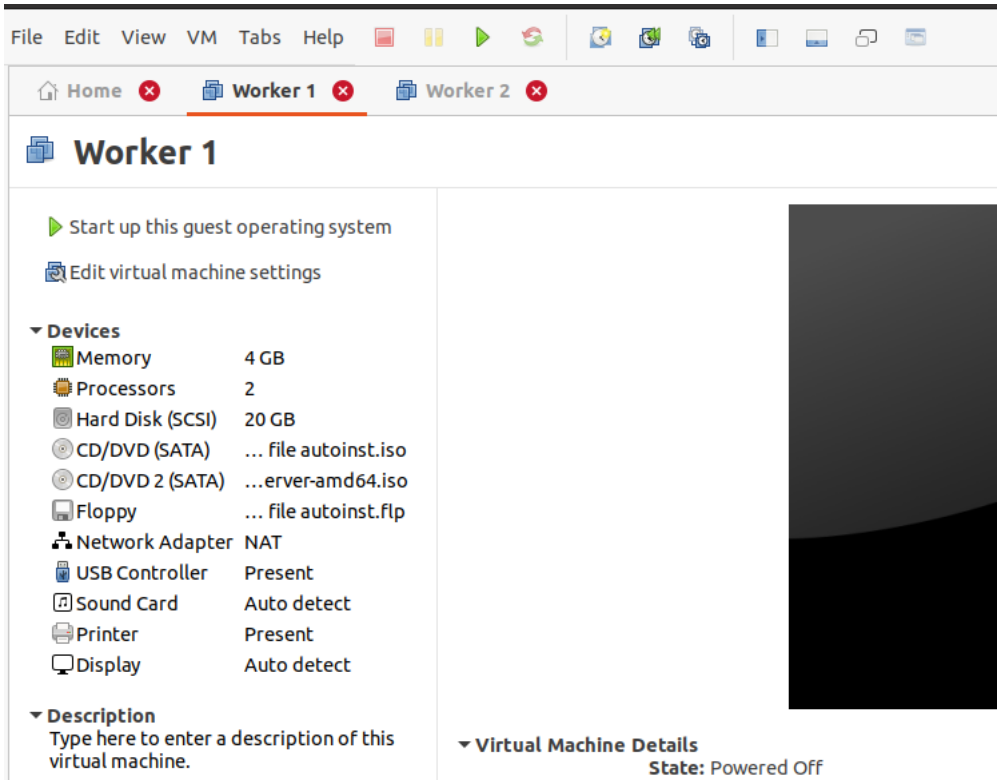

FIGURE 4.2 – Les machines virtuelles

4.4 Les tests et évaluations

4.4.1 Test

Dans cette section nous allons présenter le test de l'implémentation en commençant par le serveur. La première étape consiste à lancer le framework Ray sur la machine serveur (Ubuntu desktop) et les deux machines virtuelles (Workers), sachant que nous devons avoir la même version de Ray sur chacune des machines.

Sur le Serveur

Dans cette section, nous allons ouvrir le terminal Ubuntu et démarrer le framework Ray et le résultat est le suivant :

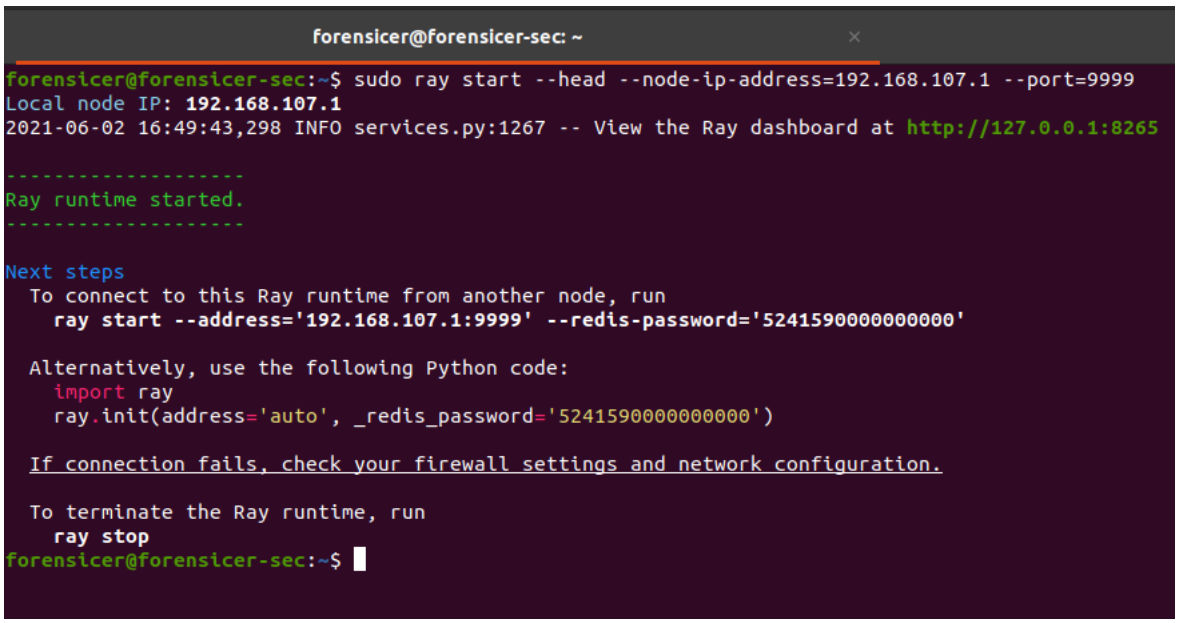

FIGURE 4.3 – Lancement du Ray sur le serveur

Sur le VM

Dans cette section, tout d'abord, nous allons ouvrir le terminal Ubuntu sur chaque machine virtuelle, sachant que toutes les machines sont connectées au sous-réseau, la deuxième étape consiste à connecter ces VMs (travailleurs) avec le serveur en utilisant la connexion Ray et le résultat est le suivant :

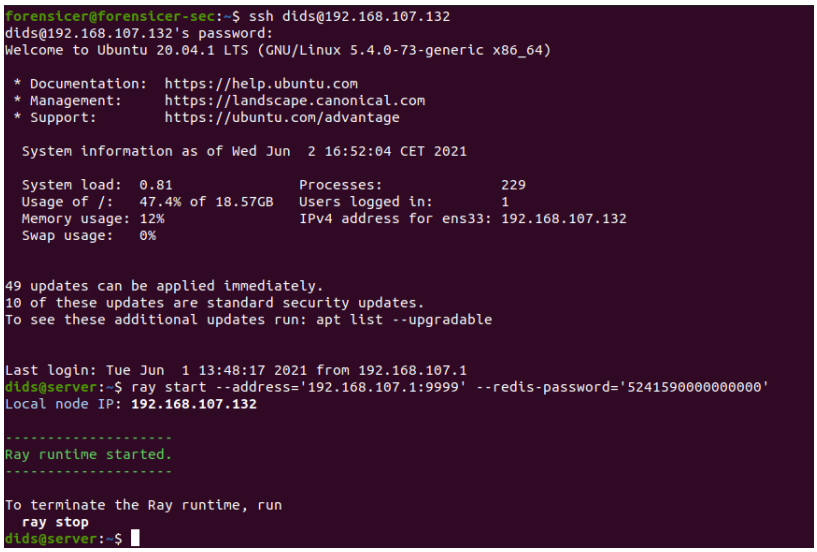

FIGURE 4.4 – Lancement du Ray sur les Workers

Lancement du système

Dans cette section, nous allons présenter le travail du serveur, dans la machine serveur nous allons exécuter le programme python dans notre terminal pour lancer le système qui distribuera le traitement aux autres travailleurs (Workers), le résultat est le suivant : Une

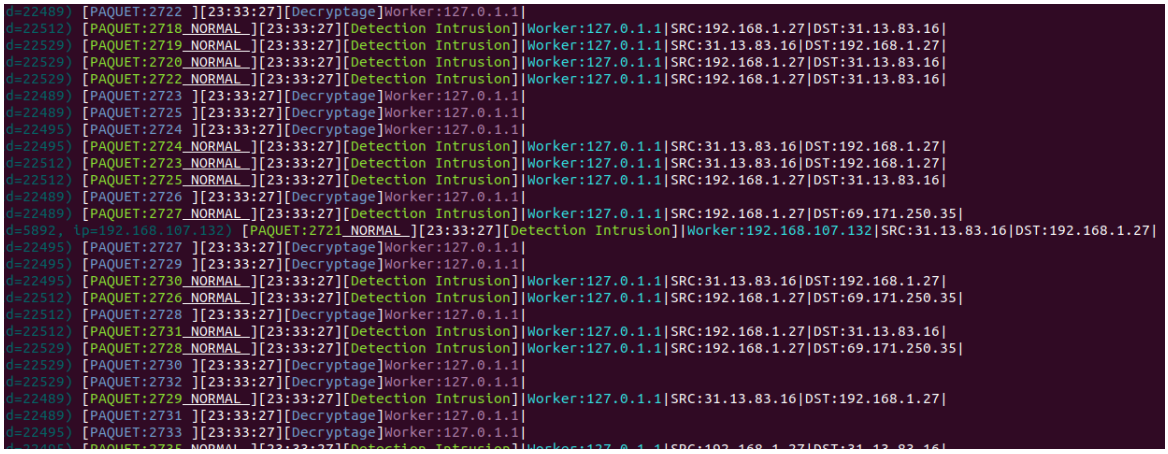

FIGURE 4.5 – Lancement du traitement

fois que le programme est lancé, le système commence à capturer/décrypter les paquets et à les distribuer aux autres travailleurs comme vous le voyez dans la figure 4.5 sachant que l'adresse Ip :*"192.168.107.132"* est un travailleur.

4.4.2 Évaluation

Dans cette section, nous allons discuter l'évaluation de notre prototype par la contribution de deux parties :

Optimisation de temps

Dans cette section, nous allons présenter les différentes comparaisons possibles, et pour ce faire, nous allons commencer le test avec un seul serveur (sans distribution) et voir combien de temps faudrait-il pour terminer le traitement.

Traitement 100K paquets sans la distribution

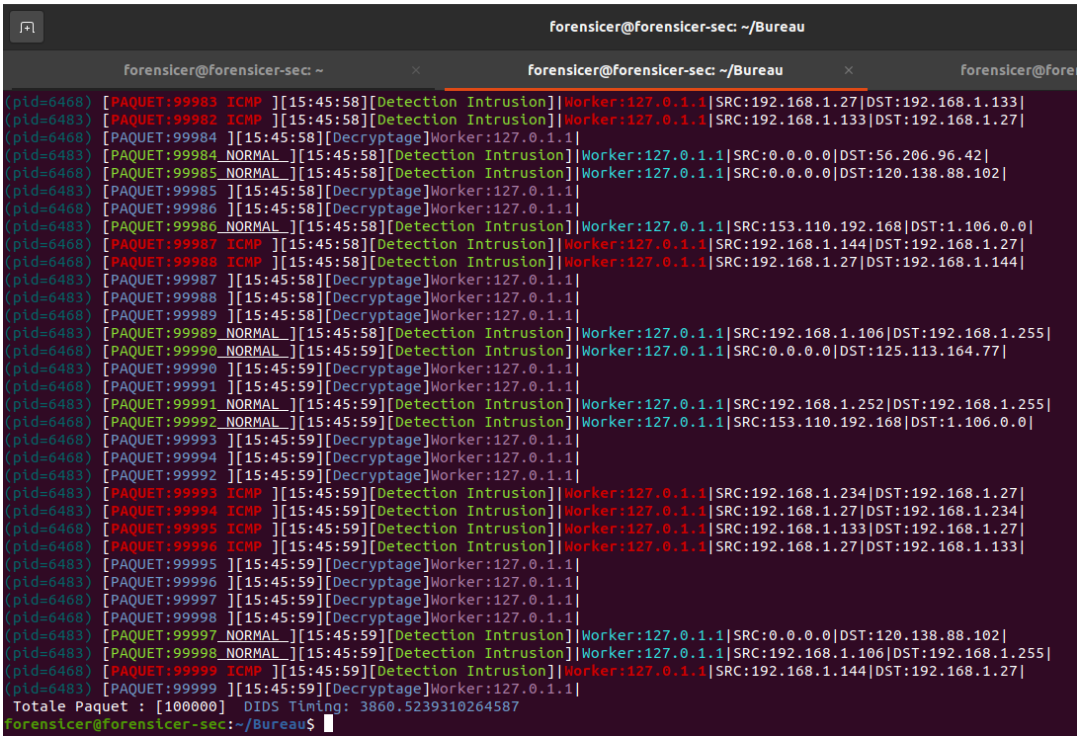

FIGURE 4.6 – Traitement 100K paquets sans la distribution

Traitement 100K paquets avec la distribution

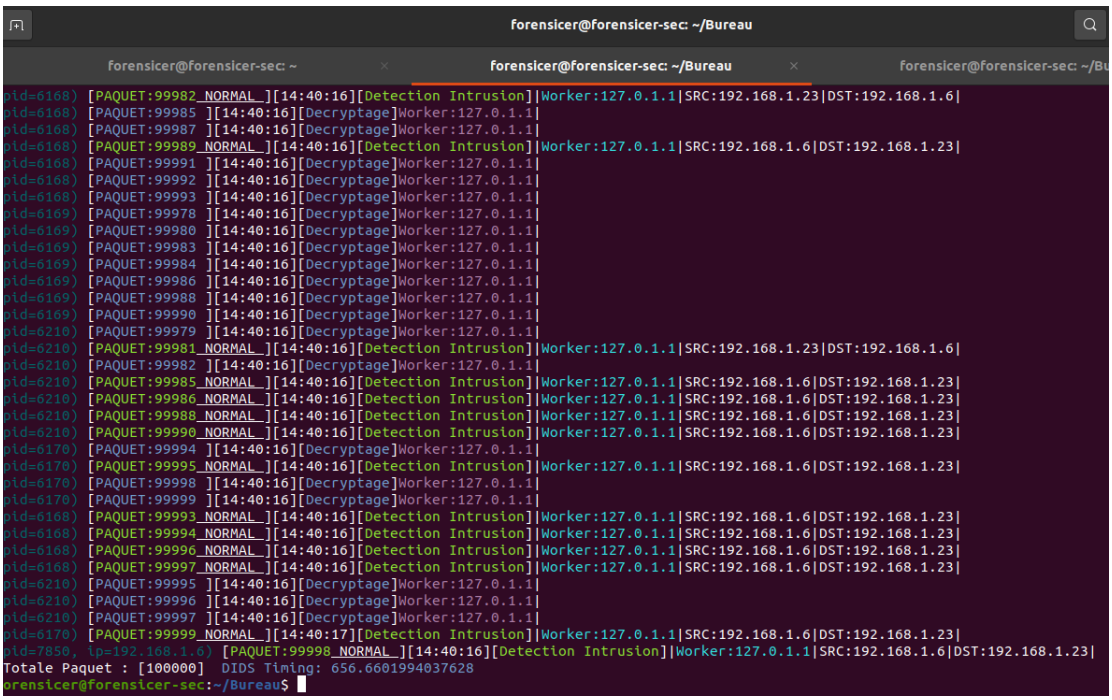

FIGURE 4.7 – Traitement 100K paquets avec la distribution

Discussion

Comme le montre la dernière figure, la durée du traitement varie, ce qui nous amène à conclure que la relation entre les travailleurs et la durée du traitement est une relation inverse, c'est-à-dire que plus il y a de travailleurs (Workers), plus la durée du traitement est courte.

Tolérance aux panne

Dans cette section, nous allons présenter le test de la tolérance aux pannes, en se basant sur l'avantage du dashboard de Framework Ray qui nous fournis la possibilité de voir l'état des workers et surveiller les log.

Dashboard Ray

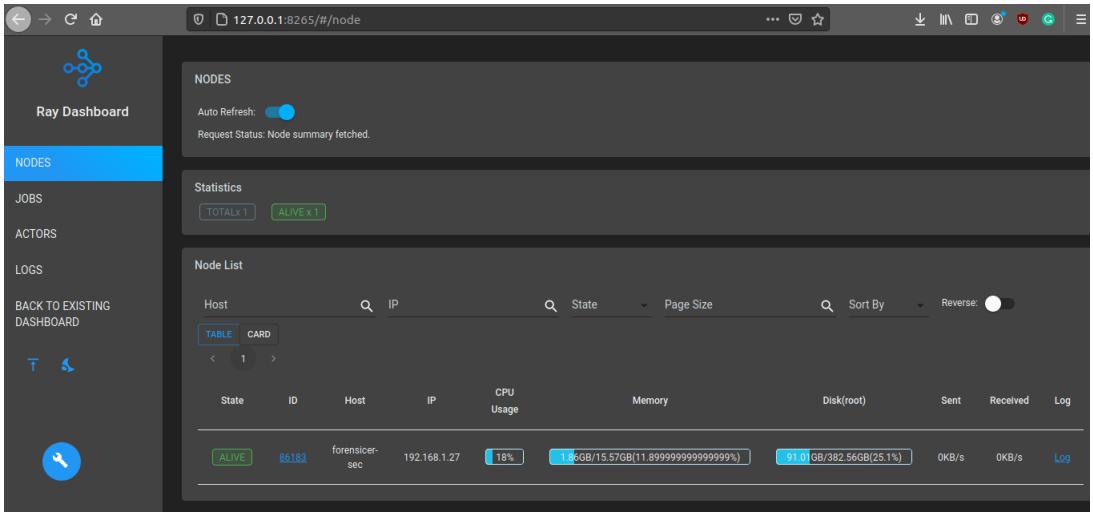

FIGURE 4.8 – le tableau de bord du Framework Ray

L'état normal des workers

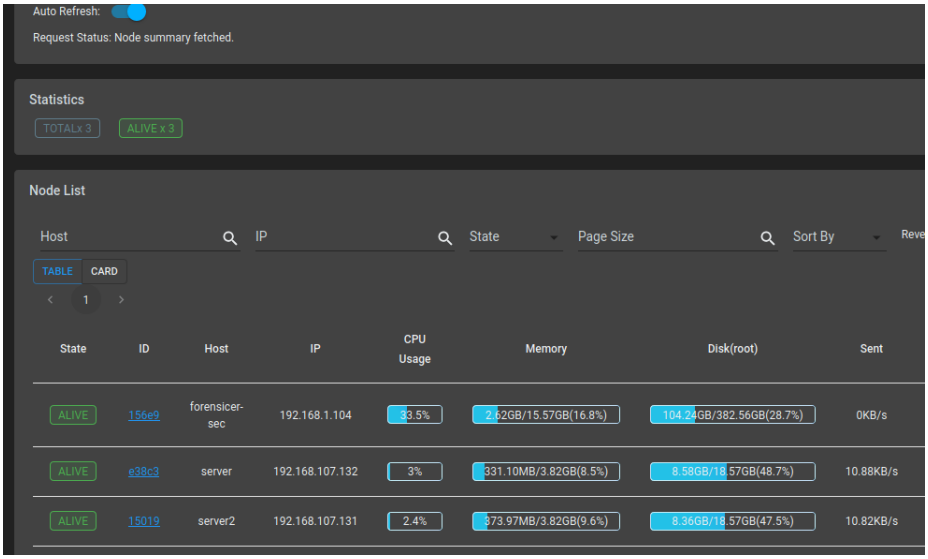

FIGURE 4.9 – L'état normal des workers

L'état des workers avec la panne

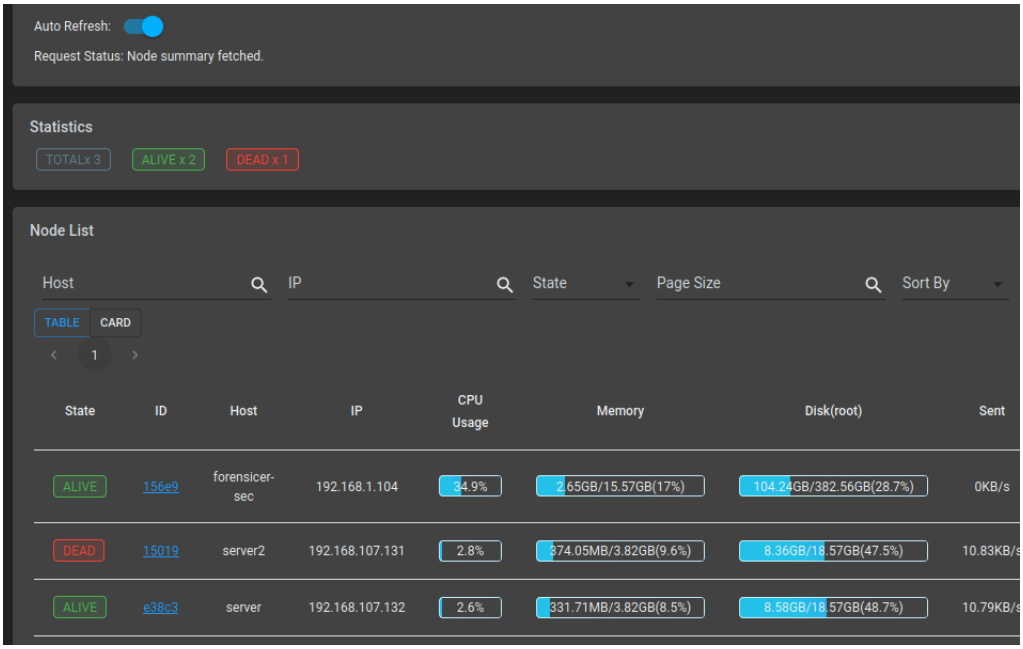

FIGURE 4.10 – L'état des workers avec la panne

Logs serveur

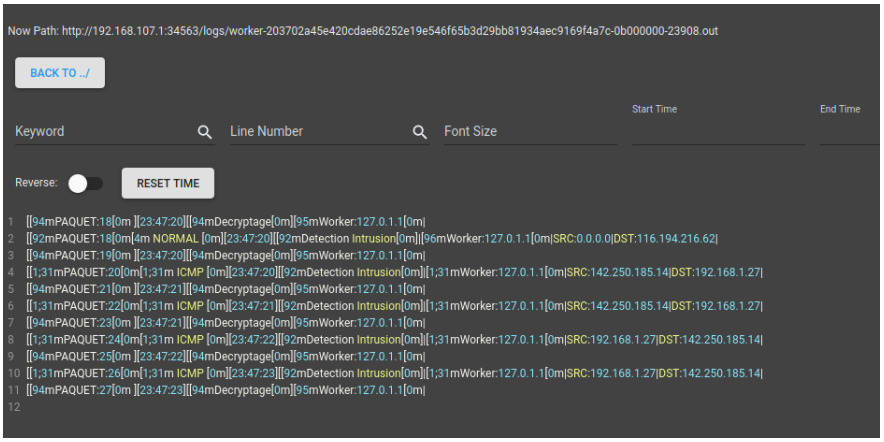

FIGURE 4.11 – Logs serveur

Logs client (workers)

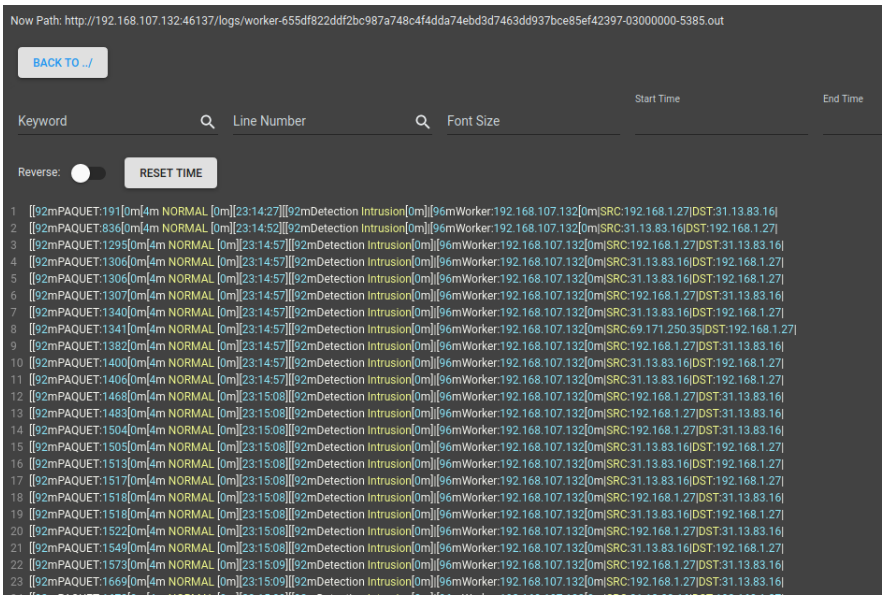

FIGURE 4.12 – Logs client (workers)

4.5 Discussion

Dans cette section, nous allons présenter les principaux avantages et inconvénients de notre proposition :

Avantages :

- Détection en temps réel.
- Rendre un système plus puissant contre les attaques.
- Plus de performances matériels grâce à la distribution.
- Plus de travail (détection) que ce qu'une machine peut faire dans un laps de temps raisonnable.
- Plus de données que ne peut contenir une seule machine.
- Le système de détection fonctionnera même en cas de panne.

Inconvénients :

- Le coût.
- la complexité d'implémentation et modification de l'infrastructure de distribution.
- Gestion des ressources (CPUs, GPUs, mémoire)
- Faible contre les attaques 0-day/inconnue.

4.6 Conclusion

Le but de ce chapitre est de résumer les informations collectées et de donner un traitement statistique, en ajoutant les mécanismes d'analyse. En raison du problème posé, il est devenu nécessaire pour un détecteur d'intrusion d'être à jour des évolutions offensives sur Internet, en particulier une évolution pour réaliser des attaques. Notre objectif principal était de proposer un nouveau prototype qui pourrait corriger certains des anciens défauts et donner une bonne image d'un système qui peut gérer la tolérance aux pannes et la résistance à la charge, ainsi que la durée du traitement.

Conclusion Générale

La sécurité informatique est un enjeu clé. Cela comprend d'abord la définition d'une politique de sécurité, puis la mise en œuvre de cette politique. Les méthodes préventives telles que le contrôle d'accès sont essentielles, mais pas suffisantes. Par conséquent, il est nécessaire de recourir à la détection d'intrusion. En particulier, la méthode de détection d'intrusion configurée par la politique de sécurité nous semble très prometteuse. Cette méthode est entièrement basée sur le modèle d'évolution de l'état du système et le modèle de politique de sécurité.

Il est évident que les IDS ont un rôle très important dans la sécurité des réseaux, mais face au changement des méthodes de pénétration et face au développement des attaques ces systèmes restent toujours faibles dans la mesure où il faut détecter de nouvelles attaques manuellement et les suivre puis garder les informations nécessaires de ce dernier dans la base de l'IDS, ce qui pose le réseaux dans un grave danger pour une période inconnue. Notre objectif était d'augmenter la performance du système de détection d'intrusion et être capable d'éviter les problèmes tels que les fausses alertes et les alertes excessives, ce qui pourrait être causé par une attaque comme DDos.

Les IDS classiques ne sont plus aussi performants contre les attaques développées, par rapport à l'optimisation de la détection et la résistance contre les attaques et même la tolérance aux pannes. en effet les attaques d'aujourd'hui telles que les attaques distribuées posent un grand problème face à ces systèmes de détection classique en raison de son intolérance à ce type d'attaque. Cela nous a incité à rechercher une proposition qui pourrait améliorer et apporter des solutions aux problèmes du système de détection classique. Notre proposition est de faire face aux attaques distribuées par un détecteur distribué c'est a dire un système de détection distribué (DIDS).

Future perspectives

Implémentation de notre proposition n'est pas complète vue le temps et les ressources présentes. Parmi les causes qui nous ont limités dans cette implémentation est que l'apprentissage machine exige plus de temps et demande un certain nombre des bases de données avec plus de ressources, Ainsi que la résolution des fausses alertes nécessaires.

L'apprentissage par machine a de nombreux bénéfices sur cette proposition, pour la

détection des attaques habituelles sans avoir besoin de décrypter, détecter et analyser les données dans les en-têtes à nouveau. Il aide également à réduire la quantité de fautes négatives/positives et rend l'IDS plus fiable et solide. Par conséquent, la première étape à effectuer est d'obtenir suffisamment de données pour représenter le problème à résoudre. Ce n'est pas toujours facile. Certaines informations sont plus chères que d'autres. Par exemple, lorsque la partie données est chiffrée, il est plus facile d'obtenir l'en-tête du paquet réseau que les informations de la partie données. La deuxième étape consiste à nettoyer les données collectées (également appelé pré traitement), c'est-à-dire à réduire le contenu strictement intéressant et sa conversion.

D'autre-part, en terme applicative dans la phase d'apprentissage, les informations peuvent comprendre les habitudes de l'utilisateur ou différents types d'attaques. En phase d'exécution, ils permettent de détecter les attaques. Cependant, la question est de savoir s'il est possible de mesurer la quantité de données nécessaire pour avoir le bon modèle, En d'autres termes, il adhère à un certain taux d'erreurs acceptables. Ce problème n'est toujours pas résolu. Cependant, le choix des informations à considérer dépendra du type d'apprentissage.

En effet, le besoin pour plus de ressources et d'infrastructures de machines élevées nous a dressé un obstacle à la poursuite de nos recherches pour l'ajout de l'apprentissage automatique à notre implémentation ; nous avons donc considéré que cette partie constituera un très bon départ pour de futures travaux à nos recherches, afin de rendre la solution proposée plus robuste et plus riche en matière de sécurité des réseaux.

Références

[1] *Dorothy e. denning. Ieee transactions on software engineering, vol.se-13, no.2 february 1987, 222-232*

[2] *Michael Sobirey. Intrusion detection system bibliography. URL : http ://www-rnks.informatik.tu-cottbus.de/sobirey/ids.html, March 1998.*

[3] *Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi. A taxonomy of computer program security flaws. ACM Computing Surveys, 26(3) :211–254, September 1994.*

[4] *what-is-an-intrusion-detection-system URL :https ://www.csoonline.com/article /3255632/what-is-an-intrusion detection-system how-an-ids-spots-threats.html, date :19 fev 2018.*

[5] *Détection des intrusions dans les systèmes d'information : la nécessaire prise en compte des caractéristiques du système surveillé, Ludovic Mé, Supélec, Campus de Rennes, Equipe SSIR 01/04/2015.*

[6] *Intrusion detection system : A comprehensive review, Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, Kuang-Yuan Tung. 25 April 2012.*

[7] *R. Sandhu and P. Samarati, Authentication, access control and intrusion detection, The Com- puter Science and Engineering Handbook (Boca Raton, FL) (A. Tucker, ed.), CRC Press, 1997.*

[8] *Robert J. Ellison, Nancy R. Mead, Thomas A. Longstaff, and Richard C. Linger, The surviv- ability imperative : Protecting critical systems, CrossTalk : The Journal of Defense Software. October 2000.*

[9] *Intrusion Detection : A Survey, Aleksandar LazarevicVipin KumarJaideep Srivastava, ch.2, DAAD19-01, NSF, 2002.*

[10] *K. Scarfone, P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," NIST Special Publication 800-94, Feb. 2007.*

[11] *David R. Safford, Douglas Lee Schales, and David K. Hess. The tamu security package : An ongoing response to internet intruders in an academic environment. In Proceedings of the Fourth USENIX Security Symposium, pages 91–118, Santa Clara, CA, October 1993.*

[12] *Warren S. Sarle. Neural networks and statistical models. In Proceedings of the Nine-teenth Annual SAS Users Group International Conference, April, 1994, pages 1538–1550, Cary, NC, April 1994. SAS Institute.*

[13] *Secure Networks, Inc. Ballista security auditing system. URL http :/www.securenetworks.com/, 1997.*

[14] *Teresa F. Lunt, R. Jagannathan, Rosanna Lee, Sherry Listgarten, David L. Edwards, Peter G. Neumann, Harold S. Javitz, and Alfonso Valdes. IDES : The enhanced prototype–a real–time intrusion–detection expert system. SRI International, 333 Ravenswood Avenue, Menlo Park, CA, October 1988.*

[15] *Network Security Through Data Analysis From Data to Action, Michael Collins, O'REILLY, February 2014.*

[16] *Sandeep Kumar and Eugene Spafford. A pattern matching model for misuse intrusion detection. In Proceedings of the 17th National Computer Security Conference, pages 11–21, October 1994.*

[17] *Aleksandar LazarevicVipin, KumarVipin KumarJaideep, SrivastavaJaideep Srivastava, "Managing Cyber Threats", "Intrusion Detection : A Survey", January 2005,330.*

[18] *Emmanouil Vasilomanolakis, Shankar Karuppayah, Max Mühlhäuser, and Mathias Fischer. Taxonomy and survey of collaborative intrusion detection, May 2015. URL :http ://doi.acm.org/10.1145/ 2716260.*

[19] *Designing Distributed Systems PATTERNS AND PARADIGMS FOR SCALABLE, Brendan Burns, 244p February 20, 2018.*

[20] *M. Mosbah, Modèles et Approches Formels pour les Systèmes Distribués, URL : https ://www.labri.fr/perso/mosbah/Enseignement/ALGODIST/cours1.pdf.*

[21] *Serge Haddad, Fabrice Kordon, Laurent Pautet, Laure Petrucci, Distributed Systems design and algorithm, 22 janvier 2013.*

[22] *Styles et Patrons architectures Distributes LOG8430 : Styles d'architecture (polymtl.ca) 1/33, 2017.*

[23] *Preview Software Architecture Design Tutorial (PDF Version). 29 sep 2019.*

[24] *An Introduction to Distributed Intrusion Detection Systems by Nathan Einwechter, Senior Research Scientist last updated January 8, 2001.*

[25] *Taxonomy and Survey of Collaborative Intrusion Detection, Article in ACM Computing Survey, May 2015, Emmanouil Vasilomanolakis, Shankar Karuppayah, Mathias Fischer.*

[26] *Survey of intrusion detection systems :techniques, datasets and challenge-sAnsam Khraisat, Iqbal Gondal, Peter Vamplew and Joarder Kamruzzaman December 2018.*

[27] *The DIDS (Distributed Intrusion Detection System) Prototype Steven R. Snapp, Stephen E. Snaha, 7 pages, 1992.*

[28] *N. Bashah, I. B. Shanmugam, and A. M. Ahmed, "Hybrid intelligent intrusion detection system," Proceedings of World Academy of Science, Engineering and Technology, vol. 6, June 2005.*

[29] *S. Snapp, J. Brentano, and G. Dias et al. "DIDS (Distributed Intrusion Detection System) – motivation, architecture, and an early prototype," In Proceedings of the 14th National Computer Security Conference, 1991.*

[30] *F. Hosseinpour, A. Meulenberg, S. Ramadass, P. Vahdani, and Z. Moghaddasi, "Distributed agent based model for intrusion detection system based on artificial immune system," Int. J. Digit. Content Technol. its Appl., vol. 7, pp. 206–214, 2013.*

[31] *N. Afzali and R. Azmi, "MAIS-IDS : a distributed intrusion detection system using multi-agent AIS approach," Eng. Appl. Artif. Intell. Vol. 35, pp. 286–298, 2014.*

[32] *Comprendre et anticiper les attaques DDoS,Agence nationale de la sécurité des systèmes d'information (ANSSI), Mars 2015*

[33] *Anas Abou El Kalam. MODÈLES ET POLITIQUES DE SECURITE POUR LES DOMAINESDE LA SANTE ET DES AFFAIRES SOCIALES. Réseaux et télécommunications [cs.NI]. InstitutNational Polytechnique de Toulouse - INPT, 2003. Français.*

[34] *https ://www.netscout.com/what-is-ddos/ip-icmp-fragmentation*

[35] *Jean-Marc Robert, ETSProtection contre les menaces -Détection -A08*

[36] *extensibilité, Le Grand Dictionnaire terminologique, Office québécois de la langue française (le 14 mars 2021).*

[37] *Med El assad, ARCHITECTURE D'UN SYSTEME CLIENT/SERVEUR,2007*

[38] *Ray : A Distributed Execution Engine for the MachineLearning Ecosystem, Philipp Moritz, 16 août 2019, 79 page*

[39] *https ://datatracker.ietf.org/doc/html/rfc793*