REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLEB DE BLIDA
FACULTE DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDES

En vue de l'obtention d'un diplôme de master en Informatique

THEME

Une approche contextuelle pour la corrélation d'alertes et détection d'intrusions à base d'ontologie

Promotrice :
Mlle. Boustia N

Réalisé par :
M. Djellata Dhiaa Eddine
M.djarboua Omar Malike

nateur Fareh
dant Benstiti
ury

Promotion : Juin 2016

REMERCIMENT

Pour vivre dans cette vie il faut avoir une devise, Il m'a offre cette chance, Il m'a donnée une Âme, Il a fixé ma destinée, C'est lui ma seul et unique vérité, Je lui dirai j'ai et j'aurai toujours foi en lui Car c'est mon seul unique dieu.

Alors " Ahmade Allah" de m'avoir donné Cette chance, " Ahmade Allah" de m'avoir soutenue jusqu'à La fin, " Ahmade Allah" pour le courage que vous m'aviez donnée pour tenir jusqu'au là

Merci mon dieu «ALLAH ».

Au terme de ce travail, nous tenons à exprimer toute notre reconnaissance et remerciements à notre promotrice Mlle Boustia, qui a fait preuve d'une grande patience et a été d'un grand apport pour la réalisation de ce travail. Ses conseils ainsi que ses orientations nous ont permis de mener à terme ce projet.

Nous remercions tous ceux qui sans eux nous n'aurions pu commencer le travail dans de bonnes conditions.

Enfin, nous adressons nos plus sincères remerciements à nos parents, nos frères, et à toute la famille, à tous nos proches et amis, qui nous ont toujours soutenus et encouragés au cours de la réalisation de ce mémoire.
Merci à tous et à toutes.

Dédicace

Je dédie ce modeste mémoire

A tous ceux qui nous ont aidé

A spécialement ma famille pour les soutiens inconditionnelles dans tous les domaines.

A Tous ceux qui vivent pour la recherche et le savoir

A Toutes les personnes qui ont contribué de loin ou de prêt a la réalisation de ce travail

A Tous ceux qui auront l'occasion de lire ce travail.

Djarboua Amor malik.

Dédicace

Je dédie ce modeste travail

A Mes parents qui ont tant souffert pour m'éduquer et m'éclaircir le chemin de la réussite.

A Mes chers frères.

A Tout ma famille, A Tous mes amis sans exception.

A Tous ceux qui vivent pour la recherche et le savoir

A Toutes les personnes qui ont contribué de loin ou de prêt a la réalisation de ce travail

A Tous ceux qui auront l'occasion de lire ce travail.

Djellata Dhiaa eddine.

Résumé

La détection d'intrusions vise à identifier les actions et les tentatives qui essaient de contourner la politique de sécurité pour compromettre la confidentialité, l'intégrité ou la disponibilité d'une ressource, et à lever des alertes en cas de détection.

Les outils de détection d'intrusions (IDS) actuellement en opération produisent de trop nombreuses alertes. Les informations qu'elles contiennent manquent de précision et sont, de plus, parcellaires et de très bas niveau. Ces alertes sont par conséquent d'un intérêt limité pour un opérateur humain. La recherche sur la corrélation d'alertes est très prometteuse.

Par la corrélation d'alertes nous pouvons espérer réduire le volume d'informations à traiter, améliorer la qualité du diagnostic proposé et dégager une meilleure vision globale de l'état de sécurité du système en cas d'intrusion.

Cet état de l'art présente un mélange prometteur : les systèmes de détection d'intrusion basés sur l'ontologie.

Mots clés : système de détection d'intrusions IDS, Sécurité, Intrusion, Corrélation d'alerte, Ontologie.

Abstract

Intrusion detection is to identify actions and attempts trying to circumvent the security policy to compromise the confidentiality, integrity or availability of a resource, and to raise alerts in case of detection.

Intrusion detection tools (IDS) currently in operation produce too many alerts. The information they contain inaccurate and are, moreover, fragmented and of very low level. These alerts are therefore of limited value to a human operator. Research on alert correlation is very promising.

By alert correlation can we hope to reduce the volume of information to process, improve the quality of the proposed diagnosis and generate a better overall view of the system safe state in case of intrusion.

This state of the art presents a promising mixture: intrusion detection systems based on ontology.

Keywords: intrusion detection system IDS, Security, Intrusion, alarm correlation, Ontology.

Table des matières

Chapitre III : Réalisation de corrélation d'alerte

Liste des abréviations

IDS	Intrusion Detection System.
NIDS	Network-based system IDS.
HIDS	Host-based system IDS.
KIDS	Kernel Intrusion Detection System.
KIPS	Kernel Intrusion Prevention System.
IPS	Intrusion Prevention System.
TCP	Transmission Control Protocol.
IP	Internet Protocol.
OSI	Open Systems Interconnection.
SIM	Security Information Manager.
OWL	Ontology Web Language.
XML	eXtensible Markup Language.
RDF	Resource Description Framework.
RDFS	Resource Description Framework Schema.
SPARQL	SPARQL Protocol and RDF Query Language.

Liste des figures

Introduction général

Actuellement, le monde connaît une avance technologique considérable dans tous les domaines, et cela grâce à l'informatique.

Aujourd'hui, l'informatique joue un rôle important dans le développement des entreprises et d'autres établissements.

Les systèmes informatiques et les réseaux sont devenus des outils indispensables pour la société actuelle. Ils sont aujourd'hui déployés dans tous les secteurs professionnels.

La sécurité des systèmes informatiques vise à protéger l'accès et la manipulation des données et les ressources d'un système par des mécanismes d'authentification, d'autorisation, de contrôle d'accès, etc. Néanmoins avec l'ouverture et l'interconnexion des systèmes informatiques, des attaques exploitant les failles de ces systèmes et contournant leurs mécanismes de sécurité sont toujours possibles. Il n'est donc pas toujours possible d'agir préventivement, c'est-à-dire de définir une politique de sécurité en terme de confidentialité, d'intégrité et de disponibilité des données et ressources du système à protéger, et de mettre en œuvre des mécanismes implantant cette politique. Afin de détecter toute tentative de violation des mécanismes de la sécurité, une surveillance permanente ou régulière des systèmes est nécessaire. Il existe plusieurs moyens d'assurer cette surveillance, parmi eux on trouve les systèmes d'informations sont les Systèmes de Détection d'Intrusion (IDS).

Un IDS est un système de détection d'intrusion qui consiste à scruter le trafic réseau et collecter tous les événements, les données à partir d'infrastructure informatique, analyser et générer des alarmes, pour identifier les attaques en cours et les tentatives malveillantes. Les IDS ont comme tout système des avantages et des inconvénients. L'un des problèmes majeurs est le nombre de fausses alertes. Ce taux peut être réduit en procédant à une corrélation d'alertes.

Dans ce travail, nous proposons une approche formelle pour la corrélation et la réduction de fausses alertes dans les systèmes de détection d'intrusions afin d'augmenter leur efficacité.

Notre document est structuré comme suit:

Introduction générale dans laquelle nous avons défini la problématique et notre contribution à la solution.

Le premier chapitre présente un état de l'art sur les systèmes de détection d'intrusions (IDS), alors que le deuxième chapitre introduit les ontologies.

Notre approche de corrélation d'alerte basée sur les ontologies est décrite au troisième chapitre. Le dernier chapitre décrit l'implémentation de notre approche

Nous terminerons ce mémoire par une conclusion générale et des perspectives pour continuer à améliorer ce travail.

CHAPITRE I

LE SYSTEME DE DETECTION D'INTRUSIONS

1. Introduction

Historiquement, internet a débuté par un réseau privé connectant les gouvernements, les militaires et les chercheurs académiques. C'est pourquoi il y avait de faibles besoins de protocoles sécurisés. A la base, on ne pensait pas au virus, au vers, au spam, au phishing, aux zombies, aux spywares, aux attaques par déni de service,... De plus, le coût de subir une telle attaque n'était pas significatif. Petit à petit, le réseau s'est ouvert au monde et sa taille a empêché la création d'un mécanisme totalement sécurisé. De plus, une facilité venant des programmes tout faits et supportant une automatisation des attaques a permis de réaliser rapidement des agressions par n'importe qui.

L'idée intuitive de la sécurité informatique est de limiter l'accès à un système informatique. Avec une sécurité parfaite, les informations ne sont jamais compromises puisqu'un utilisateur non-autorisé n'y a jamais accès. Néanmoins, la sécurité parfaite n'est pas réaliste. C'est pourquoi on tente toujours de prévenir, détecter et répondre à une attaque afin de ne pas permettre qu'une même agression se reproduise. La prévention permet de limiter les cas où une offensive se produit. Pour cela, on utilise des techniques d'authentification, de chiffrement, ou même de camouflage faisant croire à un éventuel attaquant que la tentative ne peut pas aboutir. Puisque la prévention n'est pas parfaite, la détection permet l'identification de caractéristiques qui violent les politiques de sécurité.

La détection d'intrusions a été introduite en 1980 par J.P. Anderson**[2]** qui a été le premier a montrer l'importance de l'audit de sécurité dans le but de détecter les éventuelles violations de la politique de sécurité d'un système, c'est-a-dire une violation d'une des propriétés de confidentialité, d'intégrité ou de disponibilité du système. En 1987, Denning [3] publie un modèle de détection d'intrusions. En 1988, il existait au moins trois prototypes **[4]**, La recherche dans ce domaine s'est ensuite développée, le nombre de prototypes s'est énormément accru.

Le but peut varier : la protection de documents confidentiels, trouver de nouvelles attaques, satisfaire des contraintes légales telles que la confidentialité est de signaler les intrusions ou les attaques, suivant la conception de l'IDS, a l'administrateur de sécurité pour que celui-ci puisse prendre les mesures de réaction adéquates, comme remettre le système dans un état sur, ... etc. Après avoir détecté une violation, il est intéressant de réagir. Les réponses peuvent être différentes mais le but est de réduire l'apparition de pénétrations en

améliorant la prévention et la détection du système, ainsi que de réparer les dommages générés par l'intrusion avec éventuellement des poursuites judiciaires.

L'évaluation et la comparaison des systèmes de détection d'intrusions est un problème en soi de par la diversité des sources de données possibles et la représentativité des données utilisées lors des tests notamment.

2. Définitions

2.1. Intrusion

Une intrusion est toute utilisation d'un système informatique à des fins autres que celles prévues, généralement dues à l'acquisition de privilèges de façon illégitime.

L'intrus est généralement vu comme une personne étrangère au système informatique qui a réussi à en prendre le contrôle, mais les statistiques montrent que les utilisations abusives (du détournement de ressources à l'espionnage industriel) proviennent le plus fréquemment de personnes internes ayant déjà un accès au système. [4]

Une intrusion dans un système informatique est aussi définie par Heady et al. Comme:

« N'importe quel ensemble d'actions essayant de compromettre l'intégrité, la confidentialité ou l'accessibilité d'une ressource ». [7] [6]

En dépit de différentes formes d'intrusions, elles peuvent être regroupées dans deux classes :

-Les intrusions connues ; Ces intrusions sont des attaques bien définies qui généralement exploitent des failles connues du système cible.

-Les intrusions inconnues ou anomalies ; Ces intrusions sont considérées comme des déviations du profil normal d'un système. Elles sont détectées dés qu'un comportement anormal du système est observé. [8] [9]

2.2. Détection d'intrusions

La détection d'intrusions consiste à analyser les informations collectées par les mécanismes d'audit de sécurité, à la recherche d'éventuelles attaques. [4]

C'est la capacité à identifier les individus utilisant un système informatique sans autorisation (cracker) et identifier ceux qui ont un accès légitime au système mais qui abusent de leurs privilèges (menace interne). [7]

On dit qu'une intrusion a eu lieu quand une victime vient d'enregistrer des pertes au sens large, ou des conséquences relatives à l'attaque. Ces attaques sont motivées par la présence de vulnérabilités dans le système, qui sont exploitées par les intrus pour atteindre leurs objectifs. D'une manière formelle une attaque est une action conduite par un ou plusieurs intrus, contre une ou plusieurs victimes, tout en ayant un objectif à atteindre.
Cette action est une série d'événements qui occasionne des conséquences sur la sécurité du système. **[10]**

2.3. Définition d'un IDS

Un Système de détection d'intrusions (Intrusion Detection System (IDS)) est un équipement permettant de surveiller l'activité d'un réseau ou d'un hôte donné, afin de détecter toute tentative d'intrusion et éventuellement de réagir à cette tentative **[11].**
Sa fonctionnalité est la détection des techniques de sondage (balayages de ports, finger-printing), des tentatives de compromission de systèmes, d'activités suspectes internes, des activités virales ou encore audit des fichiers de journaux (logs).
Un IDS peut lancer deux types d'alertes:

- **Faux positif** : une alerte provenant d'un IDS, mais qui ne correspond pas à une attaque réelle.
- **Faux négatif** : une intrusion réelle qui n'a pas été détectée par l'IDS. **[12]**

Debar **[13]** simplifie le système de détection d'intrusions dans un détecteur qui analyse les informations en provenance du système surveillé (voir figure I.1). **[14]**

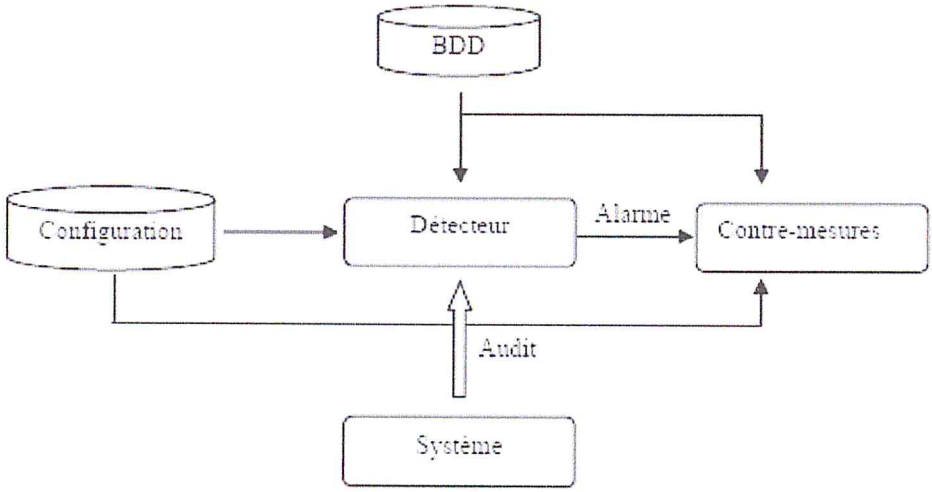

Figure I.1. *Modèle simplifié d'un système de détection d'intrusions.*

Le détecteur analyse trois types d'informations : les informations de long terme relatives aux techniques utilisées dans la détection (Base de données de signatures), les informations de configuration qui déterminent l'état courant du système, et les informations d'audit qui décrivent les événements survenus dans le système.
Philip **[15]** définit trois critères pour évaluer l'efficacité des systèmes de détection d'intrusions. **[14]**

-L'exactitude ; On parle de l'exactitude quand les systèmes de détection d'intrusions déclarent comme malicieux une activité légale. Ce critère correspond au faux positif.

-La performance ; La performance de système de détection d'intrusions est le taux de traitement des événements. Si ce taux est faible, la détection en temps réel est donc impossible.
-La complétude ; On parle de la complétude quand le système de détection d'intrusions rate la détection d'une attaque. Ce critère est le plus difficile parce qu'il est impossible d'avoir une connaissance globale sur les attaques. Ce critère correspond au faux négatif.
Debbar **[13]** a rajouté également les deux critères suivants : **[14]**

-La tolérance aux fautes ; Le système de détection d'intrusions doit lui-même résister aux attaques, particulièrement au déni de service. Ceci est important parce que plusieurs systèmes

de détection d'intrusions s'exécutent sur des matériels ou logiciels connus vulnérables aux attaques.

-La réaction à temps ; Le système de détection d'intrusions doit s'exécuter et propager les résultats de l'analyse le plutôt possible, pour permettre à l'officier de sécurité de réagir avant que des graves dommages n'aient lieu. Ceci implique plus qu'un calcul de performances, parce qu'il ne s'agit pas seulement de temps de traitement des événements, mais aussi de temps nécessaire pour la propagation et la réaction à cet événement.

2.4. Composition d'IDS : [1]

Il existe trois composants essentiels dans un IDS :

1. Senseur : Le senseur est responsable de la collecte d'informations du système, telles que des paquets d'un réseau ou des données de log.

2. Analyseur : L'analyseur reçoit l'ensemble des informations venant des senseurs. Il est responsable de les analyser et d'indiquer si une attaque a lieu ainsi qu'éventuellement sa réponse. C'est essentiellement cette partie de l'IDS que ce document exposera.

3. Interface utilisateur : L'interface utilisateur permet aux utilisateurs de l'IDS de visualiser ou/et de définir le comportement du système.

3. Caractéristiques des systèmes de détection d'intrusions :

Un système de détection d'intrusions se doit de présenter les caractéristiques suivantes: **[16]**

- Être en mesure d'effectuer une surveillance permanente et d'émettre une alarme en cas de détection ;
- Fournir suffisamment d'informations pour réparer le système et déterminer l'étendu des dommages et la responsabilité de l'intrus ;
- Être modulable et configurable pour s'adapter aux plates-formes et aux architectures réseaux
-Être en mesure d'assurer sa propre défense, comme supporter que tout ou partie du système soit hors service ;
- Avoir un faible taux de faux positifs ;
- Être en mesure de tirer les leçons de son expérience et être fréquemment mis à jour avec de nouvelles signatures d'attaques ;
- Être en mesure de gérer les informations apportées par chacune des différentes machines et discuter avec chacune d'entre elles ;

- Être capable d'apporter une réponse automatique en cas d'attaques, mêmes coordonnées ou distribuées ;
- Être en mesure de travailler avec d'autres outils, et notamment ceux de diagnostic de sécurité du système ;
- Être en mesure de retrouver les premiers évènements de corruption pour réparer correctement le système d'informations ;
- Ne pas créer de vulnérabilités supplémentaires ;
- Surveiller l'administrateur système.

Lorsque le nombre de systèmes à superviser augmente et que, par conséquent, les attaques potentielles augmentent également, nous devons, alors, attendre du système de détection d'intrusions les caractéristiques suivantes : **[17]**

- Il doit être capable de superviser un nombre important de stations tout en fournissant des résultats de manière rapide et précise ;
- Il doit fournir « un service minimum de crise » c'est à dire que si certains composants de système de détection d'intrusions cessent de fonctionner, les autres composants doivent être affectés le moins possible par cet état de dégradation ;
- Il doit autoriser des reconfigurations et des installations de patchs d'une manière dynamique. Si un grand nombre de stations est supervisé, il devient pratiquement impossible de redémarrer le système de détection d'intrusions sur tous les hôtes lorsque l'on doit effectuer un changement.

4. Classification des systèmes de détection d'intrusions

Pour classifier les systèmes de détection d'intrusions, on peut se baser sur plusieurs variables. La principale différence retenue est l'approche utilisée, qui peut être soit comportementale, soit par scénarios. Nous verrons ensuite d'autres paramètres permettant de classer les différents systèmes de détection d'intrusions (voir figure I.2). **[17][14] [19][18]**

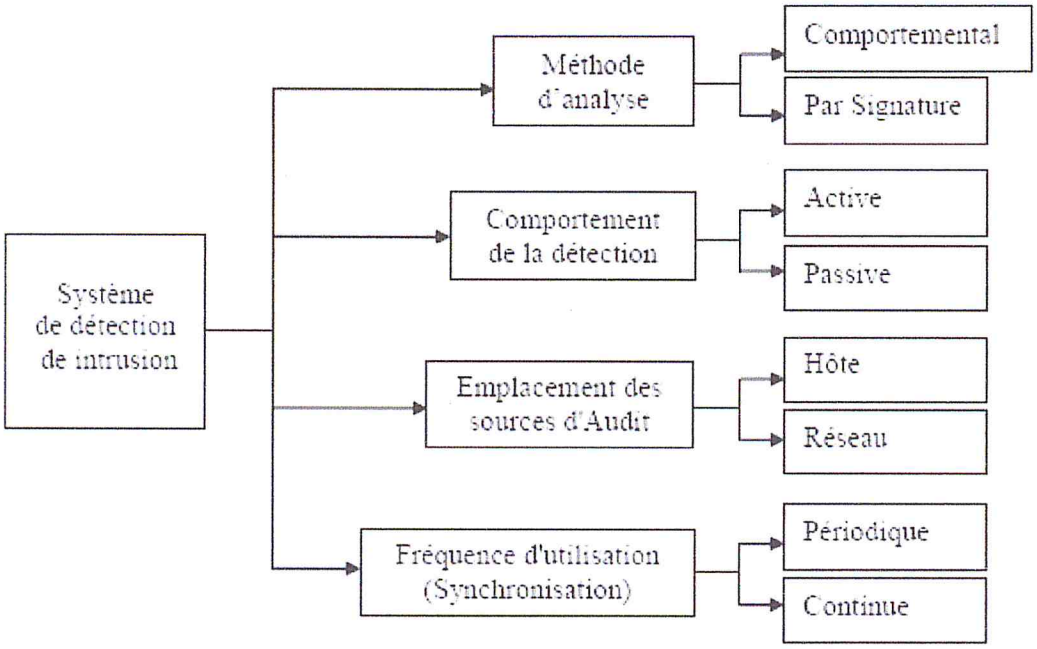

Figure I.2. *Taxonomie des systèmes de détection d'intrusions.*

5. La méthode de détection

Deux approches ont été proposées à ce jour: l'approche comportementale (anomaly detection) et l'approche par scénario (misuse detection ou knowledge based detection).

La première se base sur l'hypothèse que l'on peut définir un comportement «normal» de l'utilisateur et que toute déviation par rapport à celui-ci est potentiellement suspecte.

La seconde s'appuie sur la connaissance des techniques employées par les attaquants : on en tire des scénarii d'attaque et on recherche dans les traces d'audit leur éventuelle survenue. **[17] [20][18]**

5.1. Approche comportementale

Une approche, proposée par Anderson **[3]** puis reprise et étendue par Denning **[2]**, consiste à utiliser des méthodes basées sur l'hypothèse selon laquelle l'exploitation d'une vulnérabilité du système implique un usage anormal de celui-ci. **[21]**

La détection d'anomalies consiste à définir, dans une première phase, un certain comportement du système, des utilisateurs, des applications, etc. considéré comme «normal». Dans une seconde phase, on observe l'entité ainsi modélisée et tout écart par rapport au comportement de référence est signalé comme étant suspect **[20].** Cette approche recouvre en

fait deux problèmes distincts : la définition du comportement «normal» (souvent appelé profil) d'une part, la spécification des critères permettant d'évaluer le comportement observé par rapport à ce profil d'autre part. **[17]** Les différentes approches de détection d'anomalies se distinguent essentiellement par le choix des entités modélisées dans le profil et l'interprétation qui est faite des divergences par rapport à ce profil. **[17][21][20] [18].** Cette approche présente des avantages très importants, tel que la capacité à détecter de nouvelles attaques. **[17]** Cependant, l'approche comportementale souffre de quelques défauts : **[17][21][20]**

- Le choix des différents paramètres du modèle statistique est assez délicat et soumis à l'expérience de l'officier de sécurité ;
- En cas de profonde modification de l'environnement du système cible, le modèle statistique déclenche un flot ininterrompu d'alarmes, du moins pendant une période transitoire ;
- Un utilisateur peut changer lentement de comportement dans le but d'habituer le système à un comportement intrusif ;
- Il est difficile de dire si les observations faites pour un utilisateur particulier correspondent à des activités que l'on voudrait prohiber ;
- Pour un utilisateur au comportement erratique, toute activité est « normale ». Une attaque par déguisement sur son compte ne pourra pas être détectée ;
- Il n'y a pas de prise en compte des tentatives de collusion entre utilisateurs, alors même que cet aspect est très important, notamment dans le cas des réseaux.

5.2. Approche par scénarios

Le problème de la détection d'intrusions est également couramment approché d'une façon radicalement différente, en visant à détecter des signes de scénarii d'attaques connues. Le principe commun à toutes les techniques de cette classe consiste à utiliser une base de données, contenant des spécifications de scénarii d'attaques (on parle de signatures d'attaques et de base de signatures). Le détecteur d'intrusions confronte le comportement observé du système à cette base et lève une alerte si ce comportement correspond à l'une des signatures **[20].** La terminologie « approche par scénarios » vient du fait que l'on s'appuie sur la connaissance des techniques utilisées par les attaquants pour déduire des scénarios typiques **[17][21] [20] [4].** Cette approche présente des avantages et des inconvénients.

Ce type de détecteur d'intrusions nécessite une maintenance active : puisque par nature il ne peut détecter que les attaques dont les signatures sont dans sa base, cette base doit être

régulièrement (sans doute quotidiennement) mise à jour en fonction de la découverte de nouvelles attaques. Aucune nouvelle attaque ne peut par définition être détectée, ce qui implique un taux plus élevé de faux négatifs. Le problème se pose essentiellement pour les attaques très récentes, dont les signatures n'ont pas encore pu être incluses dans la base. Il y a donc un besoin permanent de veille technologique et de maintenance, ce qui engendre un coût global d'utilisation élevé. La construction de cette base représente ainsi un problème à part entière et un système de détection d'intrusions de ce type doit s'accompagner d'outils efficaces de maintenance de la base.

De manière générale, les détecteurs de scénarii se montrent fiables pour signaler les attaques référencées dans la base. Théoriquement, leur taux de faux positifs devrait rester très faible, car par définition une alerte n'est levée que dans le cas où la signature d'une attaque est observée. Cependant, pour des raisons de performance, les signatures sont souvent trop simples ce qui peut correspondre à des actions tout à fait légitimes. Le taux de faux positif reste donc élevé avec les outils existant aujourd'hui. De plus, une éventuelle connaissance de la base de signatures (particulièrement dans le cas des patterns) permet en principe à l'attaquant de construire précisément un scénario non détectable. Cela ne fait que renforcer encore l'exigence de maintenir régulièrement la base. Ce type de détecteur reste assez facile à mettre en œuvre, ne nécessitant pas de phase d'apprentissage (ce qui élimine le risque de sur-apprentissage ou de déformation volontaire du profil). **[20]**

Il semble donc indispensable d'hybrider l'approche comportementale avec l'approche par scénarios de manière à profiter des avantages de l'une et de l'autre.

6-Les types d'IDS : [22]

6.1. Système de détection d'intrusion réseau (NIDS)

Un NIDS écoute donc tout le trafic réseau, puis l'analyse et génère des alertes si des paquets semblent dangereux.

Le but des NIDS est d'analyser de manière passive les flux en transit sur le réseau et détecter les intrusions en temps réel.

6.2. Système de détection d'intrusion machine ou de type hôte (HIDS)

Un HIDS se base sur une unique machine, n'analysant cette fois plus le trafic réseau, mais l'activité se passant sur cette machine. Il analyse en temps réel les flux relatifs à une machine ainsi que les journaux.

Un HIDS a besoin d'un système sain pour vérifier l'intégrité des données. Si le système a été compromis par un pirate, le HIDS ne sera plus efficace. Pour parer à ces attaques, il existe des KIDS (Kernel Intrusion Detection System) et KIPS (Kernel Intrusion PreventionSystem) qui sont fortement liés au noyau.

6.3. Système de détection d'intrusion (hybrides)

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

L'exemple le plus connu dans le monde Open-Source est Prelude. Ce framework permet de stocker dans une base de données des alertes provenant de différents systèmes relativement variés. Utilisant Snort comme NIDS, et d'autres logiciels tels que Samhain en tant que HIDS, il permet de combiner des outils puissants tous ensemble pour permettre une visualisation centralisée des attaques.

6.4. Système de prévention d'intrusion (IPS)

C'est un ensemble de composants logiciels et matériels dont la fonction principale est d'empêcher toute activité suspecte détectée au sein d'un système.

Contrairement aux IDS simples, les IPS sont des outils aux fonctions « actives », qui en plus de détecter une intrusion, tentent de la bloquer. Cependant, les IPS ne sont pas la solution parfaite comme on pourrait le penser. Plusieurs stratégies de prévention d'intrusions existent :

- **host-based memory and process protection** : surveille l'exécution des processus et les tue s'ils ont l'air dangereux (buffer overflow).
- **session interception / session sniping** : termine une session TCP avec la commande TCP Reset : « RST ». Ceci est utilisé dans les NIPS (Network Intrusion Prevention System) ;
- **gateway intrusion detection** : si un système NIPS est placé en tant que routeur, il bloque le trafic ; sinon il envoie des messages à d'autres routeurs pour modifier leur liste d'accès.

Un IPS possède de nombreux inconvénients:

1- Un IPS bloque toute activité qui lui semble suspecte donc possibilité de bloquer un trafic inoffensif. Par exemple, un IPS peut détecter une tentative de déni de service alors qu'il s'agit simplement d'une période chargée en trafic. Les faux positifs sont donc très dangereux pour les IPS.

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

6.5. Les firewalls

Les firewalls ne sont pas des IDS à proprement parler, mais ils permettent également de stopper des attaques.

Les firewalls sont basés sur des règles statiques afin de contrôler l'accès des flux. Ils travaillent en général au niveau des couches basses du modèle OSI (jusqu'au niveau 4), ce qui est insuffisant pour stopper une intrusion. Par exemple, lors de l'exploitation d'une faille d'un serveur Web, le flux HTTP sera autorisé par le firewall puisqu'il n'est pas capable de vérifier ce que contiennent les paquets.

6.7. Les techniques complémentaires

- **Les scanners de vulnérabilités** : systèmes dont la fonction est d'énumérer les vulnérabilités présentes sur un système. Ces programmes utilisent une base de vulnérabilités connues.
- **Les systèmes de leurre** : le but est de ralentir la progression d'un attaquant, en générant des fausses réponses telles que renvoyer une fausse bannière du serveur Web utilisé.
- **Les systèmes de leurre et d'étude (Honeypots)** : le pirate est également leurré, mais en plus, toutes ses actions sont enregistrées. Elles seront ensuite étudiées afin de connaître les mécanismes d'intrusion utilisés par le hacker. Il sera ainsi plus facile d'offrir des protections par la suite.
- **Les systèmes de corrélation et de gestion des intrusions** (SIM - Security Information Manager) : centralisent et corrèlent les informations de sécurité provenant de plusieurs sources (IDS, firewalls, routeurs, applications…). Les alertes sont ainsi plus faciles à analyser.

- **Les systèmes distribués à tolérance d'intrusion** : l'information sensible est répartie à plusieurs endroits géographiques, mais des copies de fragments sont archivées sur différents sites pour assurer la disponibilité de l'information.

Cependant, si un pirate arrive à s'introduire sur le système, il n'aura qu'une petite partie de l'information et celle-ci lui sera inutile.

7. Les types d'attaques : [22]

7.1. Les attaques réseau : Ces attaques sont souvent dues à une faille du protocole ou de son implémentation. Pour réaliser une telle attaque, une première étape est la récolte d'informations. La récolte d'informations peut se faire grâce au social engineering ou à des outils de scan. Le but de ces récoltes est par exemple l'obtention de la version des programmes, les services fonctionnels, les adresses IP ou les ports ouverts dans le système.
Les scans peuvent se faire de plusieurs manières selon la volonté d'être plus ou moins visible lors du scan, et plus ou moins informé de la topologie du réseau. Il existe plusieurs méthodes telles que le scan simple, le scan SYN, le scan XMAS, le scan NULL, le scan FIN, le scan à l'aveugle, le scan très lent et le scan passif.
Après la récolte d'informations, l'attaque peut se réaliser. Il existe plusieurs techniques d'attaques connues telles que: IP Spoofing, ARP Spoofing, DNS Spoofing, les fragments attack, TCP Session Hijacking, Man in the middle, le déni de service...

7.2. Les attaques applicatives : Ces attaques sont souvent dues à une faille d'un logiciel ou d'une configuration. La première est essentiellement liée à des injections SQL ou à des buffers overflows.
Ces derniers permettent l'utilisation de shellcode qui donne la possibilité d'exécuter un code à distance. Les injections SQL sont des introductions de code SQL malveillant dans des requêtes de base de données permettant d'obtenir des informations privées. Enfin, les problèmes dus à une mauvaise configuration sont très répandus. En effet, beaucoup d'administrateurs réseaux préfèrent laisser la configuration par défaut que de la modifier en risquant de mal l'établir.

Conclusion :

Les attaques utilisées par les pirates sont très variées. Certaines utilisent des failles réseau et d'autres des failles de programmation. Nous pouvons donc facilement comprendre que la détection d'intrusions doit se faire à plusieurs niveaux et plusieurs types.

Dans ce chapitre, nous avons décrit ce qu'est qu'un système de détection d'intrusions. Nous avons également étudié les différents types d'IDS selon différents critères de classification tels que les méthodes de détection.

Afin de formaliser notre système de détection d'intrusion, nous utilisons une ontologie. Pour cela, nous introduisons ce concept dans le prochain chapitre.

CHAPITRE II

L'ontologie

1. Introduction :

Le terme «ontologie» trouve son origine chez les philosophes pour indiquer la métaphysique et l'étude de l'existant. Mais au fil du temps, le terme a évolué pour se retrouver dans le domaine médical où 1'ontologie désigne le traitement de la genèse des maladies. Finalement, la notion d'ontologie s'est retrouvée dans le domaine de 1' informatique.

La construction des ontologies représente un espace multidisciplinaire comportant l'apprentissage machine, la représentation des connaissances, l'exploration des données et le traitement du langage naturel. Cela engendra la naissance de plusieurs travaux de recherche dans le domaine de l'ingénierie des ontologies, surtout sur le plan de la construction des ontologies ainsi que de la réutilisation de celles-ci. Cependant l'utilisation des ontologies s'étend à plusieurs domaines, et surtout en intelligence artificielle puisqu'elle permet essentiellement de représenter les connaissances.

2. Définition d'ontologie :

2.1. Définitions dans la littérature :

La littérature est pleine de définitions différentes du terme ontologie. Chaque communauté adopte sa propre interprétation selon l'usage qui en est fait et le but visé.

Définition 1 :

Neches et al. **[25]** furent les premiers à proposer une définition :

« Une ontologie définit les termes et les relations de base du vocabulaire d'un domaine ainsi que les règles qui indiquent comment combiner les termes et les relations de façon à pouvoir étendre le vocabulaire » **[25]**

Définition 2 :

La définition la plus connue est celle de Gruber: « *Une ontologie est une* ***spécification explicite*** *d'une conceptualisation* » **[24] [23]**

Selon lui, une ontologie est modélisée en mettant en place cinq types d'éléments :

• Les concepts qui sont représentés sous forme taxonomique. On retrouve également des super/sous-concepts ainsi que des concepts composés et élémentaires.

• Les relations qui sont similaires aux associations du diagramme de classe UML. En effet, les relations correspondent aux liens entre les concepts.

• Les instances qui désignent les objets.
• Les axiomes: l'utilisation des axiomes revient à accentuer la vérité.
• Les fonctions qui correspondent à une relation spécifique. **[26] [28]**

En plus, Gruber **[26] [28]** a proposé quelques critères de base pour réaliser une ontologie, sachant que ces critères sont légèrement variables selon l'objectif de la conception de l'ontologie:
• La clarté: les termes utilisés au sein de 1' ontologie doivent être clairs, compréhensibles et objectifs.
• La cohérence: une ontologie doit déduire des faits. Aussi, la cohérence doit exister sur le plan de la logique entre les concepts et les relations.
• L'extensibilité: une ontologie doit supporter certains ajouts et mises à jour prévus.
• Le biais d'encodage minimal: la conceptualisation et le développement d'une ontologie doivent s'effectuer indépendamment des langages de codage et aussi indépendamment des symboles particuliers. Cela tient essentiellement à la diversité des sources de partage des ontologies.
• L'engagement ontologique minimal: une ontologie doit mettre l'accent sur le minimum d'engagement satisfaisant pour couvrir un domaine prédéfini. Gruber a présenté l'engagement ontologique minimal comme suit: «l'engagement ontologique s'articule sur la cohérence des termes».

Définition 3 :
«Une ontologie est une spécification formelle et explicite d'une conceptualisation partagée» **[27]**. Cette définition indique que l'ontologie se traduit par un langage compréhensible par la machine.

Définition 4 :
« Une ontologie est une **compréhension partagée** d'un domaine d'intérêt » **[23]**
« Une ontologie est une description formelle d'entités et leurs propriétés, relations, contraintes, comportement » **[30]**

Définition 5 :

«En IA, une ontologie représente un ***artefact d'ingénierie****, constitué par un* ***vocabulaire spécifique*** *utilisé pour décrire une certaine réalité, accompagné d'un* ***ensemble d'hypothèses implicites*** *concernant la* ***signification*** *des* ***mots*** *de ce* ***vocabulaire*** *»* **[23]**

« Les ontologies sont des spécifications partielles et formelles d'une conceptualisation commune » **[29]**

3. Type d'ontologie : [23]

3.1 Top-Ontologie :

C'est le niveau le plus élevé. Il structure les connaissances de haut niveau avec des catégories dont l'organisation dépend des réflexions philosophiques. Elle contient des objets et non des structures, ne s'instancie pas, et se spécialise, c'est à dire, elle spécifie les objets les plus généraux du domaine, les autres objets en seront des spécialisations et non des instances.
Exemple de ce type : SUMO, DOLCE, ……etc.

3.2 Core-Ontologie :

Ce type d'ontologie fournit des concepts structurant du domaine et les relations entre ces concepts. Par exemple, dans le domaine médical, il y'a des concepts de diagnostic, structure anatomique, et des relations comme celles liées à la localisation d'une pathologie sur une structure anatomique.

Cette Ontologies générée par un type thesaurus qui est Spécifié d'un vocabulaire de référence (physique, math, …) et d'une Hiérarchies de termes, relations sémantiques entre les termes (Synonymes, composition, ...), Exemple de ce type : Mikrokosmos,……etc.

3.3. Ontologie du domaine :

Elle fournit les concepts du domaine tels qu'ils sont manipulés par des professionnels du domaine, tel que l'ontologie descriptive qui est sémantiquement riche. Elle définie des concepts et de relations entre ces concepts pour construire des ontologies spécialisées à partir d'ontologies plus générales (core-ontologies) décrivant un domaine, en trouve dans ce type : ontologies diverses (KSL serveur).

Les type sont schématisé d'un la figure (II.1) suivante : **[23]**

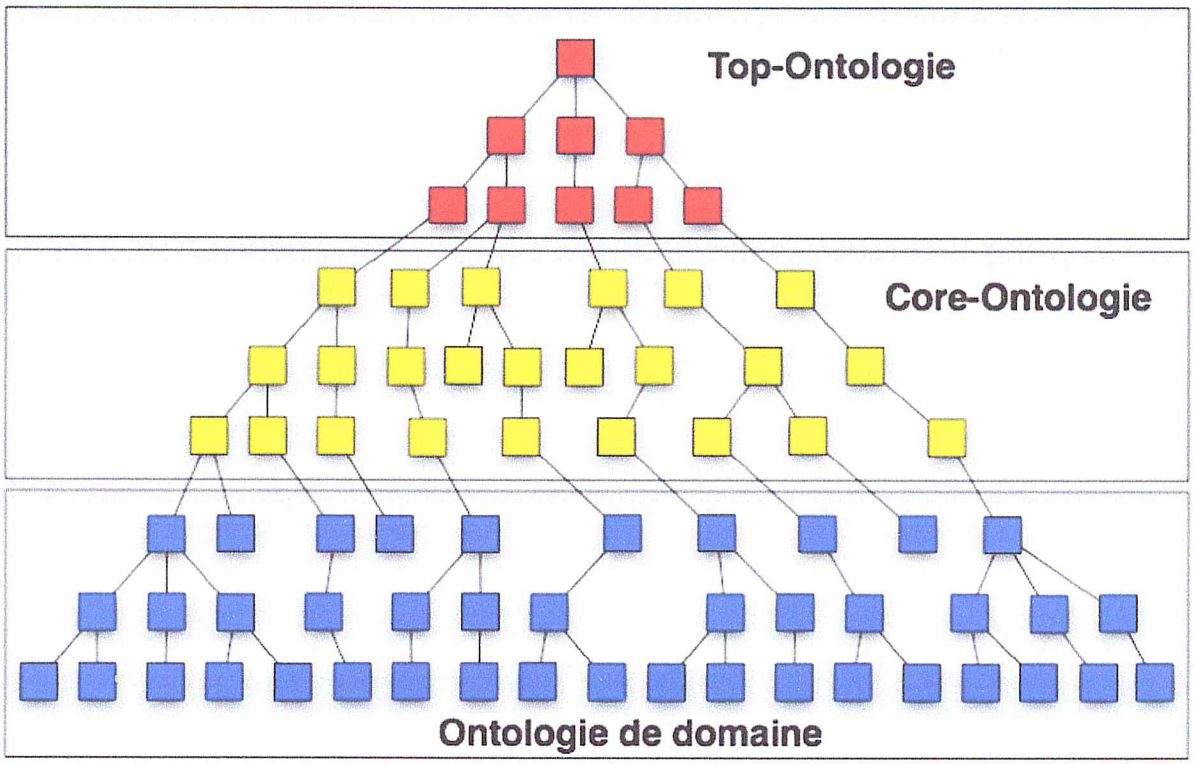

Figure II.1. *Schéma des types*

4.- Langages de spécification d'ontologies :

Dans le contexte du Web sémantique, plusieurs langages d'ontologies ont été développés pendant les dernières années. Certains d'entre eux sont basés sur la syntaxe de XML, tels que XOL, SHOE, OML, RDF, et RDF Schéma. Les deux derniers sont créés par des groupes de travail du W3C. Trois autres langages sont établis sur RDF(S) pour améliorer ses caractéristiques : OIL, DAML+OIL et OWL. La Figure II.3 représente la pyramide des langages du Web sémantique.

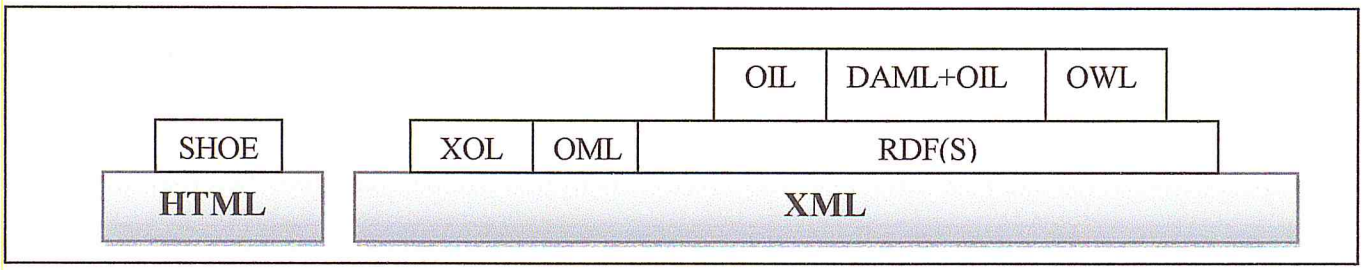

Figure II.3. *La pyramide des langages du Web sémantique*

4.1. RDF :

RDF **[32]** est un langage pour la représentation de métadonnées à propos des ressources. Le modèle RDF permet cette représentation par des assertions sous la forme d'un triplet (*ressource*, *propriété*, *valeur*), ou encore (*sujet*, *prédicat*, *objet*) :

-Ressources : les ressources sont tous les objets *décrits* par RDF. Généralement, ces ressources peuvent être aussi bien des pages Web que tout objet ou personne du monde réel. Les ressources sont alors identifiées par leur URI (Uniform Resource Identifier).

-Propriétés : une propriété est un attribut, un aspect, une caractéristique qui s'applique à une ressource. Il peut également s'agir d'une mise en relation avec une autre ressource.

-Valeurs : les valeurs en question sont les valeurs particulières que prennent les propriétés. La valeur pouvant être une autre ressource ou bien un littéral.

Exemple : Sami a 23 ans et habite Constantine

```
<rdf :RDF>
<rdf :Description about='Sami'>
<rdf :Property about='ville'>
Constantine
</rdf :Property>
<rdf :Property about='age'>
23
</rdf :Property>
</rdf :Description>
</rdf :RDF>
```

4.2. RDF(S) :

RDFS **[33]** est un langage permettant de définir des schémas de méta-données. Il définit le sens, les caractéristiques et les relations d'un ensemble de propriétés. La principale nouvelle notion est la distinction entre une classe (concept d'une ontologie) et une instance (individu d'une ontologie). Quelques notions définies sont : (rdfs : Class), (rdfs : subClassOf), (rdfs : domain), et (rdfs : range).

Sur l'exemple de Sami, nous définissons le concept de personne, une taxinomie de concepts, et l'instance Sami.

```
<rdf :RDF>
<rdfs :Class rdf :about='Personne'>
<rdfs :subClassOf rdf :resource='Thing'/>
</rdfs :Class>
<rdf :Property about='age'>
<rdfs :domain rdf :resource='Personne'/>
<rdfs :range rdf :resource='xsd :integer'/>
</rdf :Property>
<rdf :Property about='ville'>
<rdfs :domain rdf :resource='Personne'/>
<rdfs :range rdf :resource='xsd :string'/>
</rdf :Property>
</rdf :RDF>
<Personne rdf :ID='Sami'>
<age rdf :resource='23' />
<ville rdf :resource='Constantine' />
</Personne>
```

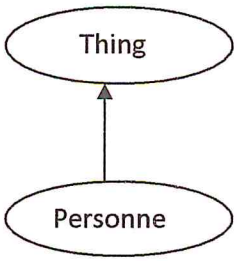

Figure II.4. *Taxinomie de concept personne*

4.3. OWL : OWL **[34]** est un langage fondé sur la syntaxe RDF/XML et héritier des travaux de DAML+OIL. OWL introduit l'aspect sémantique qui manque RDF, et offre, par ses primitives plus riches, au machine une capacité d'interprétation plus grande que celle de RDF et RDFS.

OWL se compose de trois sous-langages OWL Lite, OWL DL et OWL Full, qui offrent des capacités d'expression croissantes, chacun est une extension par rapport à son prédécesseur plus simple.

5. Les ontologies dans la sécurité d'informatique

Dans ces dernières années, les ontologies dans la sécurité informatique sont largement utilisées vue le besoin d'un partage sémantique de connaissance. Nous allons présenter trois utilisations des ontologies dans le domaine de sécurité informatique.

-RBAC (*Role-Based Access Control*)

-KAoS (*Knowledge Acquisition in autOmated Specification*)

-Rei (prononcé en anglais « ray », un mot japonais qui signifie "universel")

5.1. Contrôle d'accès s à base de rôles (RBAC) :

RBAC est un modèle de contrôle d'accès à un système d'information dans lequel chaque décision d'accès est basée sur le rôle auquel l'utilisateur est attaché. Un rôle découle généralement de la structure d'une entreprise. Les utilisateurs exerçant des fonctions similaires peuvent être regroupés sous le même rôle.

5.2. KAoS :

KAoS **[35]** est une approche de gestion d'agent basée sur la politique, comprenant les problèmes typiques de sécurité (les politiques d'autorisation, le cryptage, politique d'accès et de contrôle de ressources) et des notions supplémentaires (politiques de conversation d'agent, représentations et mécanismes d'application pour des politiques de mobilité, des politiques d'enregistrement du domaine, des politiques d'engagements, etc.).

5.3. Rei :

L'ontologie de politique de Rei est présentée en détails dans **[37].** Rei se compose de plusieurs ontologies : ReiPolicy, ReiMetaPolicy, ReiEntity, ReiDeontic, ReiConstraint, ReiAnalysis, et ReiAction. Chaque ontologie décrit les classes et les propriétés liées à ce domaine.

6. Conclusion :

Les ontologies apparaissent désormais comme une clé pour la manipulation automatique de l'information au niveau sémantique. Au fur et à mesure des recherches, des idées se dégagent autour du contenu des ontologies, des méthodes à utiliser pour les construire et des modèles et langages servant à leur représentation.

Dans ce chapitre, nous avons essayé d'éclaircir la notion d'ontologie en présentant plusieurs définitions. Nous avons cités quelques ontologies utilisées dans le domaine de sécurité.

Le chapitre suivant présente notre contribution, à savoir une ontologie pour la corrélation d'alertes et la réduction du taux de fausses alertes.

CHAPITRE III

Réalisation de corrélation d'alerte.

1. Introduction

Dans ce chapitre, nous proposons une approche de corrélation d'alertes et qui permettra en même temps de réduire le taux de fausses alertes. Cette approche sera décrite via une ontologie.

2. Approche de corrélation d'alertes :

Notre système de corrélation représenté dans la figure (III.1) est structuré en quatre (4) étapes :

- **Etape 1** : Dans cette étape les alertes générées via des IDS sont recueillies et transférées dans le composant d'intégration d'alerte. Dans cette étape toutes les informations nécessaires pour le raisonnement sur ces alertes sont recueillies à partir de ressources d'information différentes, à savoir :
  -Capture de contexte.
  -Base de données des vulnérabilités communes.
  -Les bases de données d'attaques.
- **Etape 2 :**
  Consiste en les deux tâches suivantes :
  1)- l'intégration et la conversion de l'ensemble des alertes générées par les déférents IDS dans un format unifié, analysé par l'unité de corrélation d'alertes.
  2)- l'intégration de toutes les informations contextuelles reçues implicitement ou explicitement à partir des différents outils mis en œuvre dans le système.
- **Etape 3 :**
  Dans le but d'automatiser le processus de corrélation, nous avons défini un ensemble d'ontologies extensibles: alerte, attaque, contexte et vulnérabilité.
  Les relations entre ces ontologies ont été établies selon les informations recueillies auprès de diverses ressources telles que les bases de données de vulnérabilités.

- **Etape 4 :**

  Consiste à corréler les informations existantes dans les ontologies, qui se font par l'intermédiaire du moteur de corrélation en utilisant la description de l'ontologie.

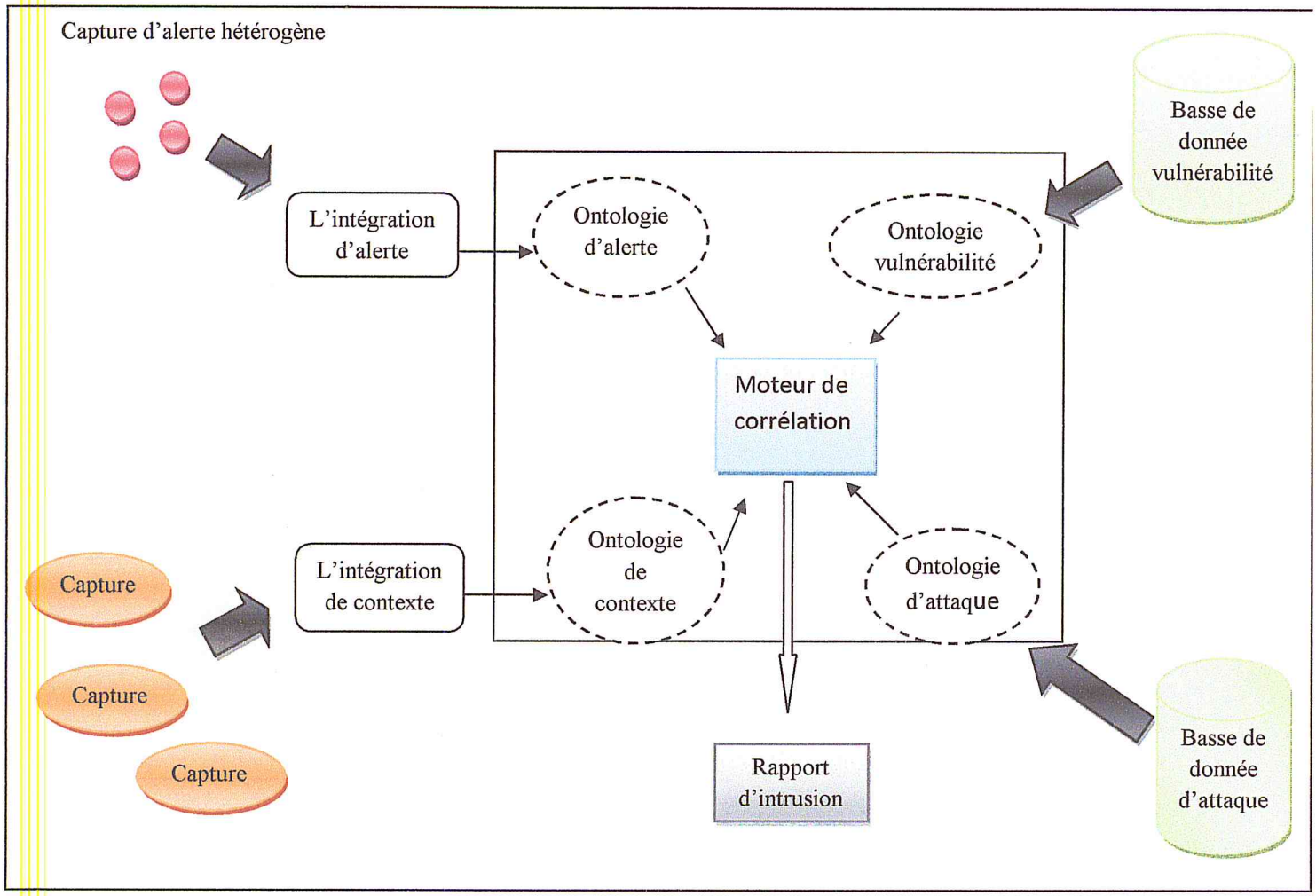

Figure III.1. *Cadre de corrélation d'alerte au courant de contexte sur la base de l'ontologie*

3. Les sources d'information :

Nous allons maintenant décrire chacun des composants en détail :

3.1. Capteurs d'alerte :

Génèrent des alertes basées sur les comportements malveillants suspects qu'ils observent sur les systèmes qu'ils contrôlent. Le plus typique des types déployés du capteur sont NIDS qui génèrent des alertes en examinant individuellement un paquet traité dans les réseaux, il peut inclure des IDS basé sur l'hôte qui génèrent des alertes en fonction de l'activité du système. Enfin il peut également inclure d'autre type de capture lié à la sécurité

3.2. Capteurs de contexte :

C'est un terme générique pour toute source d'information qui peut fournir des informations contextuelles sue les systèmes qui sont surveillés. Le concept du contexte est connu pour permettre aux analystes de définir et utiliser un aspect particulier qu'ils pensent être adaptés à la surveillance de leur système, ceci peut inclure plusieurs type d'information tel que la configuration (réseau, hôte d'application), les vulnérabilités, l'emplacement...

3.3. Base de vulnérabilité :

Au début, les informations sur les vulnérabilités sont recueillies des basées de données public bien connus tel que les Common Vulnérabilités et Exposition (CVE) **[39]** ou NVD. Ensuite les vulnérabilités peuvent être associés au cas de contexte par exemple (hôte, application, réseaux) par l'intermédiaire de l'analyse de vulnérabilité ou la gestion d'actif.

3.4. Scénarios et modèles d'attaque :

Les modèles et les informations sur les attaques peuvent être obtenu à partir de bases de données normalisées telles que l'énumération Motif Attaque commune et Classification (CAPES) **[40]** ou des connaissances spécialisées. Afin de modéliser les attaques, l'un des langages de modélisation d'attaque existants tels que LAMBDA **[41]** ou STATL **[42]** peut être utilisé. Cependant, il se trouve en dehors de la portée de notre travail.

4-Alerte et l'intégration de contexte :

Différent type de capture IDS produisent des alertes de différents formats qui pourraient ne pas facilement être interprétable par l'unité de corrélation. Par conséquent, il est nécessaire de prétraité ces flux d'alertes et de l'exporté dans un format compréhensible par l'unité de corrélation. Dans les environnements de production, cela se fera en utilisant des pilotes spécifiques qui vont faire correspondre des champs d'alertes avec des attributs de classe. Tous les champs des alertes doivent être traduits dans la classe qui leur est attribué au plus haut niveau dans la taxonomie des alertes. L'utilisation de représentations standards comme IDMEF**[43]** ou l'expression de Common Event(CEE) **[44]** devraient facilité la présentation. Dans notre travail, nous utilisons IDMEF.

Le composant d'intégration de contexte de notre approche intégrée également toutes les informations contextuelle dans différent format reçu implicitement ou explicitement a partir de divers outils mis en œuvre dans le système.

Une fois le processus d'intégration terminé, le processus de corrélation peut commencer.

5.1 Ontologie Alerte :

Toutes les alertes intégrées sont transférés dans cette ontologie ainsi que ses instances. Il y a une relation de dépendance avec l'ontologie contexte et une relation d'association avec l'ontologie Attaque (**figure III.3**). Chaque alerte est suspectée d'être une attaque dans un contexte particulier. La classe Alerte comprend des attributs génériques tels que, la source, la cible, l'heure, ...

La classe alerte dispose de plusieurs sous classe parmi eux : Netalert, HostAlert, AppAlert correspondant aux alertes générées par NIDS, HIDS et IDS basées sur des applications ou des journaux d'application, contenant chacune des attributs spécifiques générées par des capteurs. Dans notre cas, le contexte peut avoir multiple facettes. Il est probable qu'une seule alerte peut être lié à de multiples instances de contexte de diverses sous classes (par exemple un utilisateur, un réseau et une demande), et ainsi l'association entre l'alerte et le contexte sera plusieurs à plusieurs.

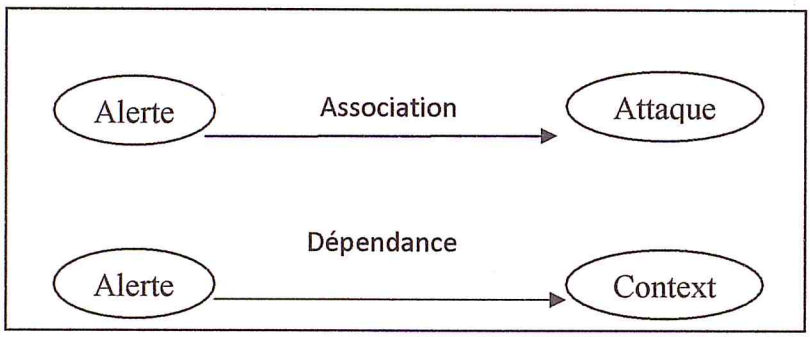

Figure III.3. *Relation d'ontologie d'alerte.*

5.2 Ontologie Contexte:

L'information contextuelle intégrée est transféré dans l'ontologie de contexte. Nous avons partagé des informations contextuelles en deux catégories :

- Informations statiques de contexte qui change rarement au fil de temps par exemple (l'architecture du réseau, hôte/profil d'utilisateur, et le type de système d'exploitation)

- Information dynamique de contexte qui change continuellement au fil du temps par exemple (le type de trafic, l'utilisation du système, le temps de jour/semaine).

5.3 Ontologie Vulnérabilité:

Cette ontologie représente la liste des vulnérabilités liée à des faits existants dans un contexte particulier, généralement instancié d'une source de vulnérabilité public tell que CVE. Cette ontologie a une relation de composition avec l'ontologie du contexte puisque chaque vulnérabilité V est spécifique à un type particulier de système, qui est représenté en tant que sous-classe de contexte (typiquement hôte). Ainsi, une vulnérabilité V peut être associé a tous les actifs (contexte) qui sont vulnérable à elle. Cette ontologie a aussi une relation d'association avec l'ontologie Attaque, car généralement toute vulnérabilité V est exploitable par une ou plusieurs attaques (**Figure III.4**).

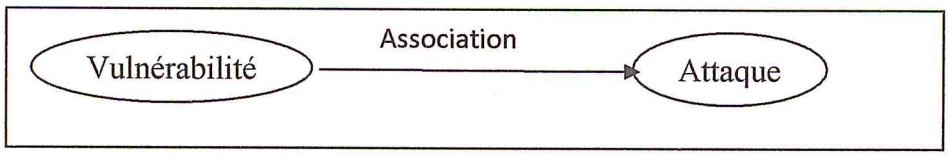

Figure III.4. *Relation d'ontologie de vulnérabilité*

5.4 Ontologie Attaque :

L'ontologie Attaque comprend des informations relatives au scénario d'attaque connu, il comprend aussi des attributs d'attaque génériques tels que Vector, Objective.... La classe Vector représente la méthode qui est utilisé par une attaque pour infecter les systèmes informatique. La classe Objective inclus les sous-classes comme des fuites d'information (InfoLeakage), exécution de base à distance (DataAnnihilation), Spamming et PrivilegeEsc. L'ontologie Attaque a une relation de dépendance avec l'ontologie de contexte, et une relation d'association avec les ontologies Alerte et Vulnérabilité (**Figure III.5**), car fondamentalement chaque attaque a besoin d'un contexte particulier pour s'exécuter, elle pourrait avoir besoin d'exploiter des vulnérabilités particulières [V1 ;:::::;Vn], en résultat, elle pourrait déclenché des alertes [a1 ;:::::;an].

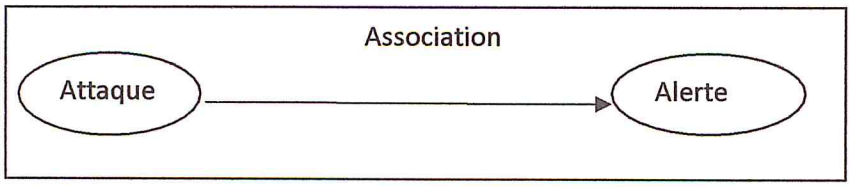

Figure III.5. *Relation d'ontologie d'attaque.*

6-Moteur de corrélation :

Afin de mettre en œuvre notre approche de corrélation d'alerte, nous utilisons OWL-DL (Ontology Web Language Description Logic) pour concevoir et alimenter nos ontologies. L'utilisation d'OWL-DL fournit une grande souplesse pour ce cadre en permettant la réutilisation ou l'adaptation de requêtes de données exprimées dans cette logique pour divers scénarios de surveillance, de déploiement, détection en ligne ou après le coup d'analyse avancée du réseau. Le moteur de corrélation aura des informations sur seulement trois des quatre ontologies que nous avons définis, Alerte, contexte et vulnérabilités. C'est en suivant les relations mentionnées ci-dessus entre les classes correspondant que le moteur de corrélation sera en mesure de déduire l'existence d'une instance d'attaque qui pourrait correspondre à un sous-ensemble particulier d'alertes, de contextes et de vulnérabilités. Ce processus de corrélation peut être considéré comme deux parcours :

6.1. Filtrage basé sur le Contexte et la vulnérabilité. Étant donné une alerte (ou alertes), déterminer quels instances de contextes sont impliqués, quelles sont leurs vulnérabilités connues, et enfin déterminer quels scénarios d'attaque pourraient être exploité.

6.2. Reconstruction d'attaque : Pour chaque scénario d'attaques possibles liées aux alertes, Essayez de faire correspondre la séquence des alertes précédente avec les étapes de l'attaque.

Le résultat de ce processus devrait, fournir à l'analyste de la sécurité une liste réduite de descriptions de haut niveau des attaques potentielles en cours (ou achevés) qui comprend quelques scénarios non pertinents et les faux positifs.

Le résultat de ce processus devrait, fournir à l'analyste de la sécurité une liste réduite de descriptions de haut niveau des attaques potentielles en cours (ou achevés) qui comprend quelques scénarios non pertinents et les faux positifs.

Afin de mettre en œuvre les deux volets de cette approche. Nous utilisons un ensemble des règles logiques exprimé en Sémantique Règle Web Langue (Semantic Web Rule Language (SWRL)) et Semantic Query-Enhanced Règle Web Langue (Semantic Query-Enhanced Web Rule Language (SQWRL)). Même si diverses approches de corrélation spécifiques pourraient être mises en œuvre dans notre modèle générique, nous utilisons certains aspects de l'approche décrite dans **[45]**, pour illustrer l'utilisation de notre approche.

7. Exemple mise en œuvre de l'approche de Valeur et al.

L'approche de corrélation d'alertes proposée dans **[45]**, comprend un ensemble complet de mesures qui couvre divers aspects du processus de corrélation d'alertes. Afin de montrer comment mettre en œuvre notre approche automatiquement nous expliquons les détails de certaines de ces étapes.

7.1. Alerte fusion. Alerte fusion est le processus de fusion des alertes qui représentent la détection indépendante du même événement malveillants par différents IDS. Une condition importante pour fusionner deux ou plusieurs alertes est qu'ils doivent être dans le même laps de temps. Nous avons une règle 1 dans le moteur de corrélation afin d'effectuer la fusion d'alerte:

Rule1_fusion:

Alert(?a1) ^ Alert(?a2) ^ Analyser(?an1) ^ Analyser(?an2) ^ detectTime(?dt1) ^ detectTime(?dt2) ^ source(?s1) ^ source(?s2) ^ Traget(?tar1) ^ Traget(?tar2) ^ classific(?cl1) ^ classific(?cl2) ^ Assesement(?as1) ^ Assesement(?as2) ^ swrlb:equal(?s1,?s2) ^ swrlb:equal(?tra1,?tra2) ^ swrlb:equal(?cl1,?cl2) ^ swrlb:equal(?as1,?as2) ^ swrlb:subtractTimes(?td,?dt1,?dt2) ^ swrlb:lessThan(?td,"5s") → sqwrl:select(?a1)

7.2. Vérification alerte. : Le processus de reconnaissance et de réduction des alertes non pertinentes qui se réfèrent à l'attaque ratée. La principale raison de l'échec d'attaque est l'indisponibilité de l'exigence contextuelle de l'attaque, c'est-à-dire l'absence de la vulnérabilité requise dans le contexte de l'attaque. Identification des attaques ratées permet au moteur de corrélation de réduire les effets des alertes non pertinentes dans son processus décisionnel. Les règles 2 et 3 dans le moteur de corrélation de notre approche permettent d'effectuer la vérification des alertes basées sur les vulnérabilités du système ciblé.

Rule_2_vérification:

Alert(?a) ^ Host(?h) ^ OS (?o) ^Vulnerability(?v) ^ classific(?cl) ^ reference(?ref) ^ hasTarget(?a, ?h) ^ hasClassific(?a, ?cl) ^ hasOS (?h, ?o) ^ hasReference(?c, ?ref) ^ hasVulnerability(?o, ?v) ^ hasName(?ref, ?n1) ^ hasName(?v, ?n2) ^ swrlb:equal(?n1,?n2) → sqwrl:select(?a)

Rule_3_vérification:

Alert(?a) ^ Host(?h) ^ App(?ap) ^ Vulnerability(?v) ^ classific(?cl) ^ reference(?ref) ^ hasTarget(?a, ?h) ^ hasClassific(?a, ?cl) ^ hasApp(?h, ?ap) ^ hasReference(?c, ?ref) ^ hasVulnerability(?ap, ?v) ^ hasName(?ref, ?n1) ^ hasName(?v, ?n2) ^ swrlb:equal(?n1,?n2) → sqwrl:select(?a)

7.3. La reconstruction du processus d'attaque. La reconstruction du processus est la fusion d'une série d'alertes qui se réfèrent à une attaque lancée par un attaquant contre une cible unique est une autre étape de l'approche dans le processus de corrélation d'alertes de **[45]**. De même pour le processus de fusion d'alerte, les alertes doivent se produire dans la même fenêtre de temps pour être en corrélation. Règle 4 exécute le processus de reconstruction du processus.

Rule_4:

Alert(?a) ^ Host(?h1) ^ Host(?h2) ^ detectTime(?t1) ^ detectTime(?t2) ^ hasSource(?a, ?h1) ^ hasTarget(?a, ?h2) ^ hasDetectTime(?a, ?dt) ^ swrlb:greaterThanOrEqual(?dt,?t1) ^ swrlb:lessThanOrEqual(?dt,?t2) → sqwrl:select(?a, ?h1, ?h2)

En résumé, les règles 1, 2 et 3 permettent de réduire les alertes non pertinentes. A cet effet, on récupère les informations requises par les ontologies alertes, contexte et vulnérabilité. Ensuite pour les alertes et les scénarios d'attaque pertinente, le processus de reconstruction est effectué par application de la règle 4, où le moteur tente d'établir une correspondance entre les alertes filtrées et les étapes d'attaques dans l'ontologie d'attaque. Une fois qu'on trouve toute correspondance entre les deux ontologies, il va afficher l'ensemble du scénario d'attaque.

8. Conclusion :

Dans ce chapitre, une approche de corrélation d'alertes basée sur les ontologies a été présentée. Le but de cette approche est d'être flexible pour lui permettre d'être utilisé dans les différents scénarios de déploiement que les analystes de sécurité sont susceptibles de faire face.

L'idée principale est d'utiliser et d'exploiter un modèle ontologie contenant des classes de base et des sous-classes pour les concepts de contexte des actifs informatiques, d'alerte, de la vulnérabilité et de l'attaque. Ces ontologies sont alors instanciées par soit automatiquement par les pilotes de source spécifique, ou manuellement pour les informations statiques (telles que le contexte, de la vulnérabilité et de l'information d'attaque). Le moteur de corrélation est alors mis en œuvre en utilisant des règles logiques.

CHAPITRE VI

Implémentation

1. Introduction :

Comme nous l'avons déjà mentionné, les systèmes de détection d'intrusions actuels souffrent d'imperfections. Afin de les surmonter, nous avons proposé une approche de corrélation d'alerte pour réduire le taux de fausses alertes.

Dans ce chapitre, nous présentons notre application ainsi que les outils utilisés

2. Les outils et les langages de développement :

2.1. Le langage OWL-DL : [46]

Le langage OWL-DL concerne les utilisateurs souhaitant une expressivité maximum sans sacrifier la complétude de calcul (toutes les inférences sont sûres d'être prises en compte) et la décidabilité (tous les calculs seront terminés dans un intervalle de temps fini) des systèmes de raisonnement. Le langage OWL-DL comprend toutes les structures de langage de OWL avec des restrictions comme la séparation des types (une classe ne peut pas être en même temps un individu ou une propriété, une propriété être un individu ou une classe). OWL-DL se nomme ainsi pour sa correspondance avec la *logique de description*, un champ de la recherche portant sur un fragment décidable particulier de la logique de premier ordre. Le langage OWL-DL, offre les propriétés de calcul souhaitées pour les systèmes de raisonnement.

Les deux figures qui exprimé des classe et des propriétés de OWL :

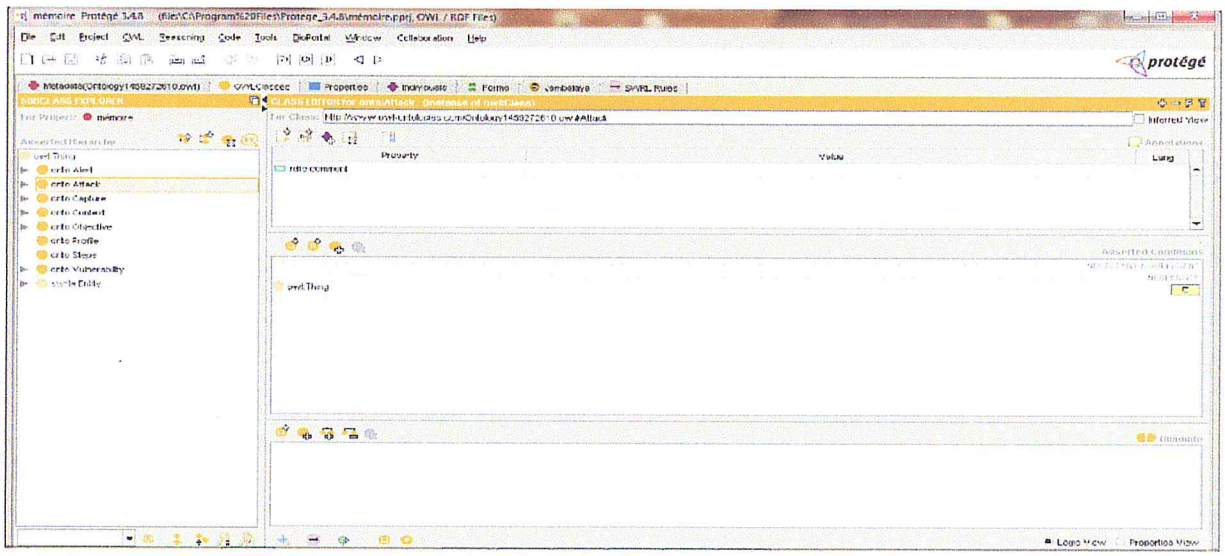

Figure VI.1. *Fenêtre OWL classes dans Protégé.*

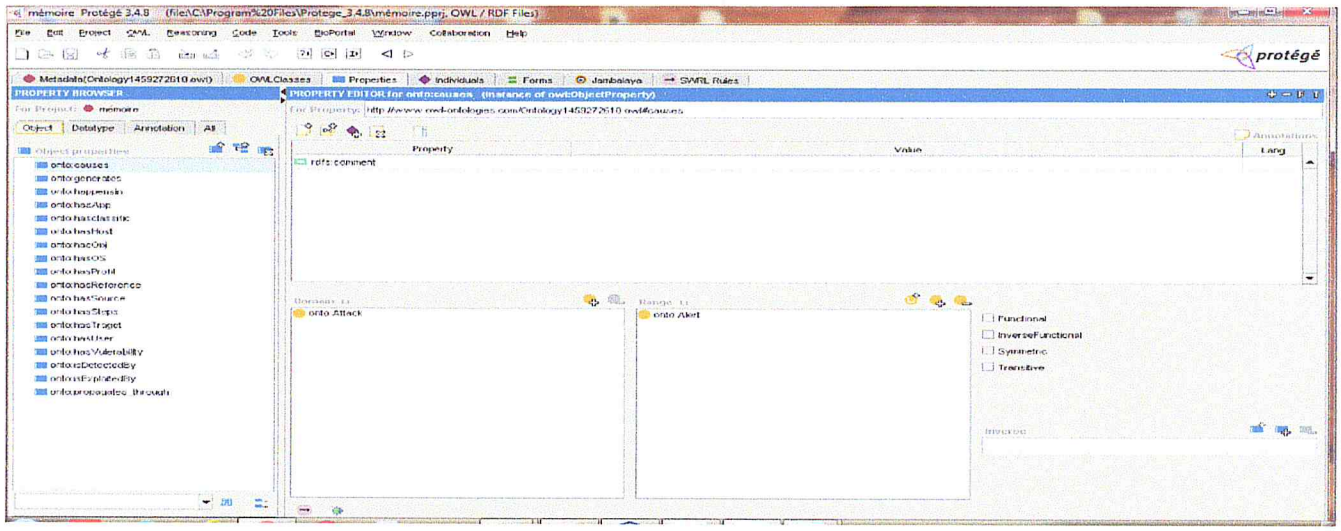

Figure VI.2. *Fenêtre OWL propriétés dans Protégé.*

2.2. Éditeur protégé :

Protégé2000 **[38]** est une interface modulaire, développée au Stanford Medical Informatics de l'Université de Stanford7, permettant l'édition, la visualisation, le contrôle (Vérification des contraintes) d'ontologies, l'extraction d'ontologies à partir de sources textuelles, et la fusion semi-automatique d'ontologies. Le modèle de connaissances de
Protégé2000 est issu du modèle des frames et contient des classes (concepts), des slots (propriétés) et des facettes (valeurs des propriétés et contraintes), ainsi que des instances des classes et des propriétés. De nombreux plug-ins sont disponibles ou peuvent être ajoutés par l'utilisateur.

2.3. Les règles logiques SWRL Rule :

SWRL **[47]** (Semantic Web Rule Language) est un langage de règles pour le web sémantique, combinant le langage OWL-DL et le langage RuleML (Rule Markup Language (Unary/Binary Datalog)).

En comparaison avec DLP (Description Logic Programs), une autre proposition relativement récente de la communauté web sémantique, permettant d'intégrer des règles et OWL, SWRL prend une approche d'intégration diamétralement opposée. DLP est l'intersection de la logique de Horn et d'OWL, tandis que SWRL est (approximativement) l'union des deux. Pour DLP, le langage résultant est une logique descriptive d'une forme inhabituelle et peu expressive. Au

contraire, SWRL garde la puissance d'OWL DL, mais au prix de la décidabilité et des implémentations concrètes par exemple

Exemple de règle :

Alert(?a1) ^ Alert(?a2) ^ Analyser(?an1) ^ Analyser(?an2) ^ detectTime(?dt1) ^ detectTime(?dt2) ^ source(?s1) ^ source(?s2) ^ Traget(?tar1) ^ Traget(?tar2) ^ classific(?cl1) ^ classific(?cl2) ^ Assesement(?as1) ^ Assesement(?as2) ^ swrlb:equal(?s1,?s2) ^ swrlb:equal(?tra1,?tra2) ^ swrlb:equal(?cl1,?cl2) ^ swrlb:equal(?as1,?as2) ^ swrlb:subtractTimes(?td,?dt1,?dt2) ^ swrlb:lessThan(?td,"5s") sqwrl:select(?a1)

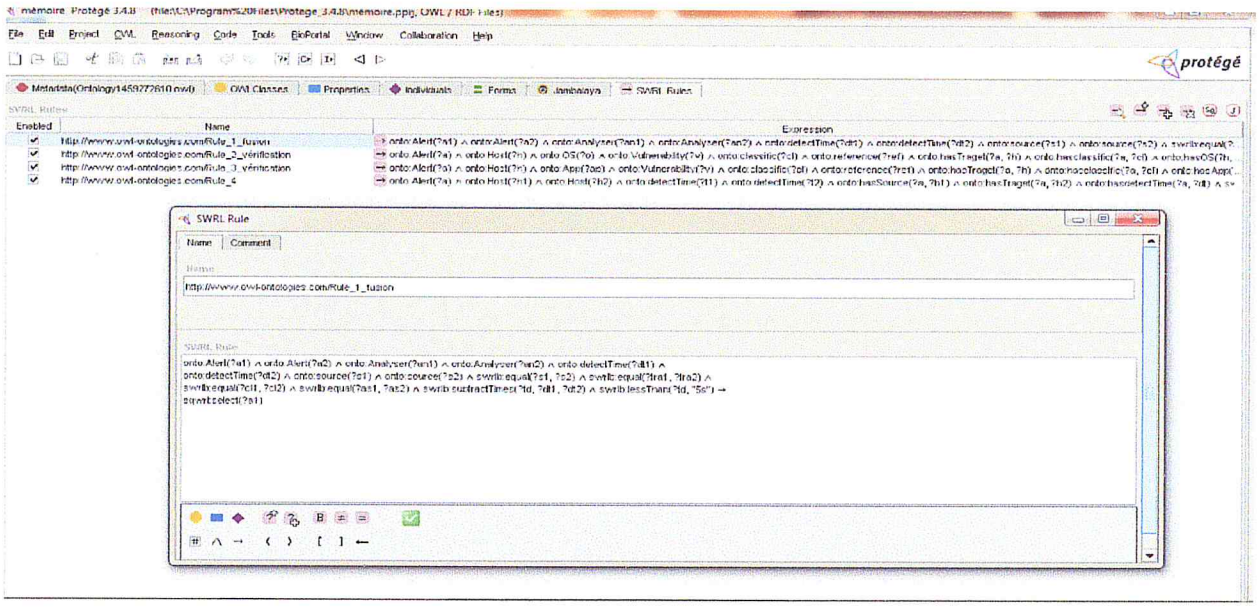

Figure VI.3. *Fenêtre de SWRL dans Protégé.*

2.4. Moteur d'inférence jesstab :

Jess **[48]** est un moteur d'inférence très populaire crée par Sandia National laboratories, c'est l'un des meilleurs moteur d'inférence sur le marché, sa puissance (il a été choisit pour être

embarqué dans les systèmes informatiques des nouveaux destroyers américain) ses mises à jour périodiques et son API java.

Ce moteur d'inférence possède un langage propre pour l'expression des connaissances sous forme de règle. Il peut être utilisé depuis protégé grâce à l'existence d'un pont (OWLJessBridge) qui permet de traduire un modèle d'ontologie dans le langage de Jess, d'exécuter les règles dans Jess et finalement de récupérer le résultat dans protégé.

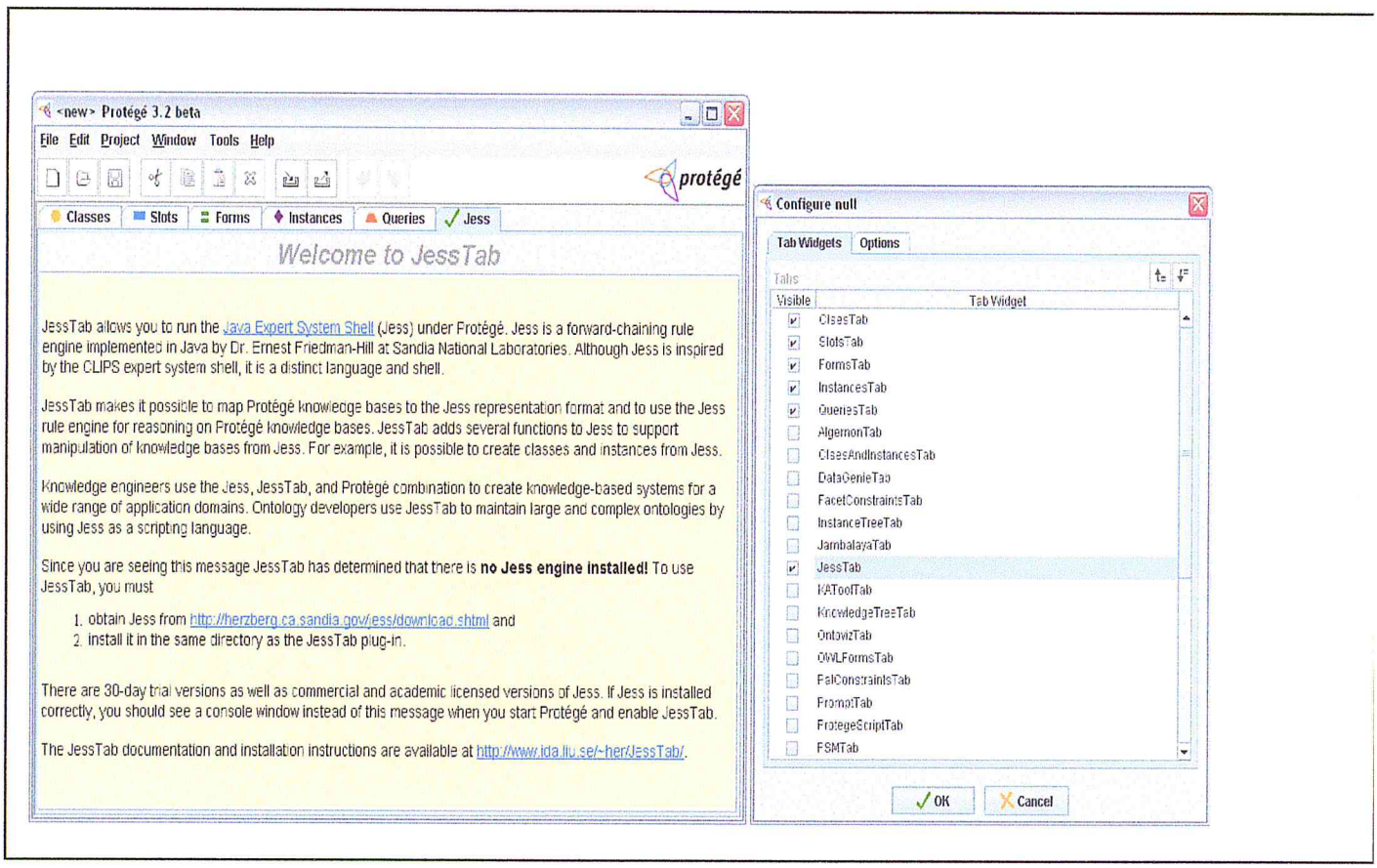

Figure VI.4. *Activation de Jesstab en protégé.*

5-SAPRQL : [49]

SAPRQL est un module d'interrogation et devenu une recommandation dans le carde de l'activité Web sémantique, est un langage et un protocole de requête pour l'interrogation de méta données sous forme des fichiers RDF ou plus exactement un langage d'interrogation de triples RDf, le langage SAPRQL s'inspire clairement dans sa syntaxe et dans ses fonctionnalités du langage SQL. Il aussi quelques traits de ressemblance mineurs avec prolog.

SPARQL possède quatre formes de résultat :

- SELECT : retourne un tableau de résultats.
- CONSTRUCT : retourne un graphe RDF, basé sur un modèle dans la requête .
- DESCRIBE : retourne un graphe RDF, basé sur ce que le processeur de requêtes est configuré pour renvoyer.
- ASK : pose une requête booléenne.

La forme SELECT retourne directement un tableau de solutions comme un ensemble de résultats, tandis que DESCRIBE et CONSTRUCT utilisent les résultats de l'appariement pour construire des graphes RDF.

Exemple :

```
PREFIX ns: <http://www.owlontologies.com/Ontology1459272610.owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?Alert
WHERE {
        ?Alert rdf:type ns:Alert.

}
```

Et les outils de programmation

2.6. JAVA :

Java est le nom d'une technologie mise au point par Sun Microsystems qui permet de produire des logiciels indépendants de toute architecture matérielle. Java est à la fois un langage de programmation et une plateforme d'exécution. Le langage Java a la particularité principale d'être portable sur plusieurs systèmes d'exploitation tels que Windows, Mac OS ou Linux. C'est la plateforme qui garantit la portabilité des applications développées en Java. Il permet de créer des applications autonomes et de doter les documents html de nouvelles fonctionnalités : animations interactives, applications intégrées, modèles 3D, etc. Ce langage est orienté objet et comprend des éléments spécialement conçus pour la création d'applications multimédia.

2.7. Eclipse :

Eclipse IDE est un environnement de développement intégré libre (le terme*Eclipse* désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

3. Implémentation :

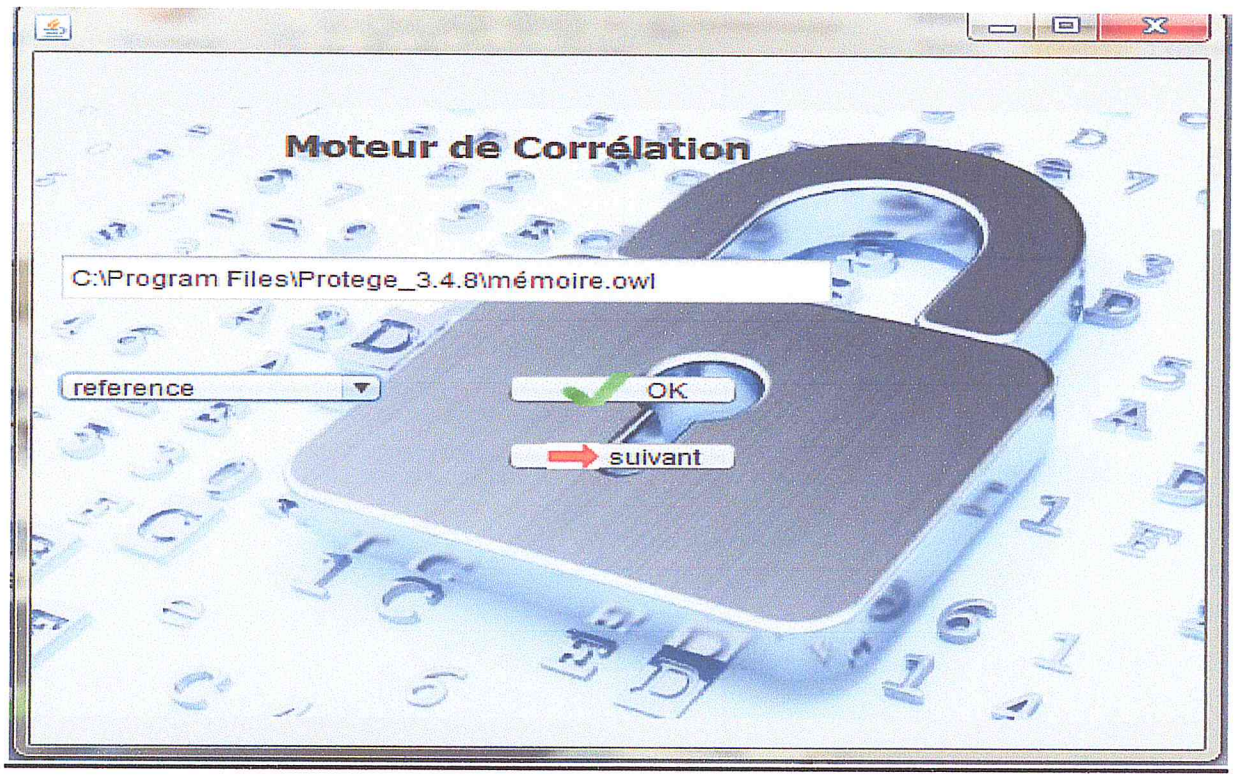

Figure VI.5. *Interface principale.*

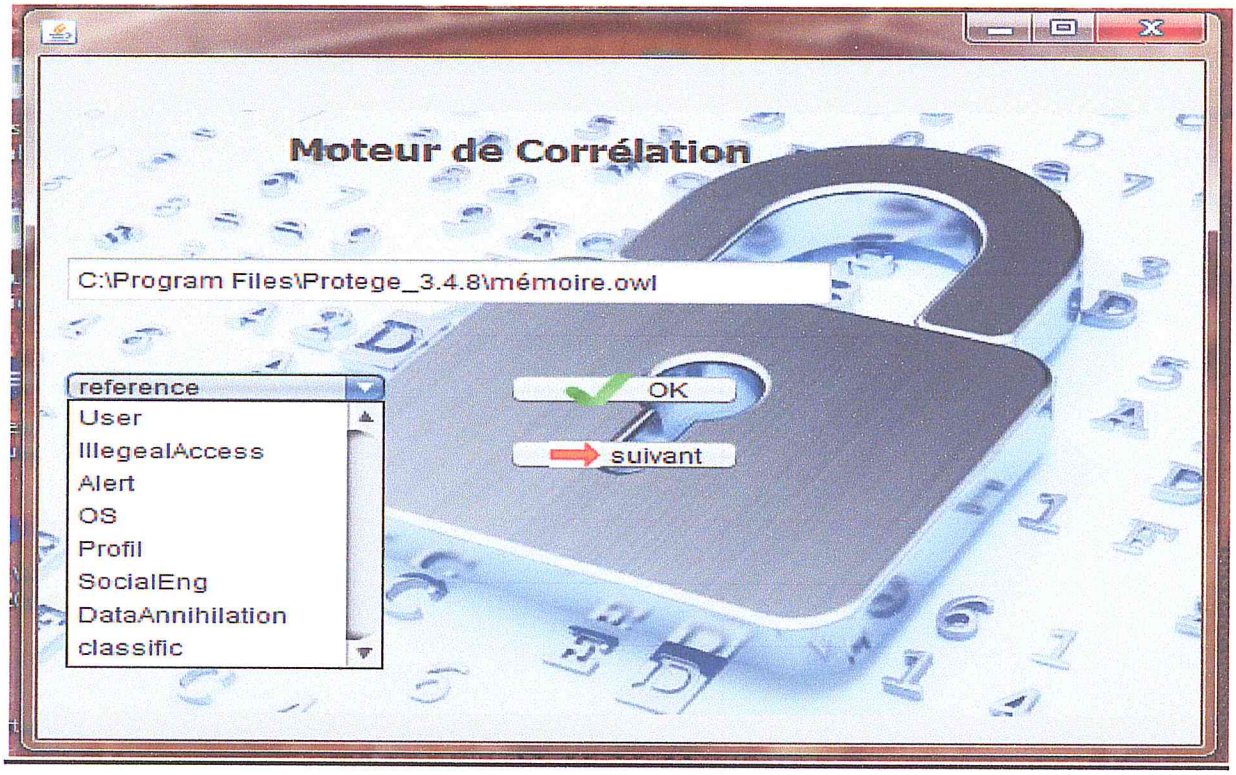

Figure VI.6. *Sélectionné des classes.*

3. Conclusion :

A travers ce chapitre, nous avons décrit l'implémentation de notre approche de corrélation d'alertes afin de réduire le nombre de fausses alertes.

Conclusion générale

Notre travail rentre dans le cadre de la sécurité informatique. Le but de notre travail est de rendre les systèmes de détection d'intrusion plus efficace en proposant une approche de corrélation d'alertes afin de diminuer le taux de fausses alertes (Faux positif).

Pour ce faire, nous avons commencé par introduire les systèmes de détection d'intrusions (IDS). Nous avons présenté par la suite le concept d'ontologie.

L'idée principale de notre travail est d'utiliser et d'exploiter un modèle ontologique contenant des classes de base et des sous-classes pour les concepts de contexte, d'alerte, de vulnérabilité et d'attaque. Ces ontologies sont alors peuplées soit automatiquement par les pilotes de source spécifique (comme pour les capteurs d'alerte), ou manuellement pour les informations statiques (telles que le contexte, de la vulnérabilité et de l'information d'attaque). Le moteur de corrélation est alors mis en œuvre en utilisant des règles logiques écrites dans Semantic Web Rule Language (SWRL) et Semantic Query-Enhanced Règle Web Langue (SQWRL) basée sur la logique de description d'OWL (OWL-DL).

Les règles décrites dans le présent document ne fournissent qu'une validation limitée sur la viabilité et l'efficacité de notre approche en termes d'alerte non-pertinente et la réduction des faux d'alertes dans les systèmes de détection d'intrusions en résumé, nous disant que les objectifs initiaux sont atteints.

Le travail n'est pas finalisé et peut être amélioré de différent manière. Dans le futur, nous proposons de tester notre approche avec une base d'alerte réelle telle que DARPA 2000.

Nous voulons aussi l'améliorer pour qu'elle puisse réduire le taux du faux négatif (alertes réelles non détectées) et détecter de nouvelles attaques.

Bibliographie :

[1]- « ***Les systèmes de détection d'intrusion basés sur du machine learning*** » .Liran Lerman. Thèse. Université Libre de Bruxelles

[2]- J. Anderson, « ***Computer security threat monitoring and surveillance*** ». 1980.

[3]- Dorothy E. Denning. « ***An intrusion detection model*** ». IEEE Transactions on software engeneering, SE-13 :222–232, 1987.

[4] - Philipe Biondi : « ***Architecture expérimentale pour la détection d'intrusions dans un système informatique*** ». Avril-Septembre 2001.

[5]- « ***Détection d'intrusions comportementale par diversification de COTS : application au cas des serveurs web*** ». Frédéric Majorczyk. Thèse de doctorat, Université de Rennes 1 (Rennes) (2008).

[6]- R.Heady, G.Luger, A.Maccabe et M.Sevilla. « ***The architecture of a network level intrusion detection system*** ». Août 1990.

[7]- Serge Fenet, « ***Vers un paradigme de programmation pour les applications distribuées basé sur le comportement des insectes sociaux : application à la sécurité réseaux*** ». Thèse de doctorat de l'Université Claude Bernard-Lyon 1. 2002.

[8] - Karima Boudaoud, « ***Détection d'intrusions : Une nouvelle approche par système multiagents*** ». Thèse de doctorat de l'Université de Genève. 2002.

[9] - Nicolas Nobelis, «**Un modèle de Case-Based Reasoning pour la détection d'intrusion** ». Rapport de stage DEA RSD / ESSI3 SAR. Avril-Septembre 2004.

[10]- Adda Mehdi et Guerrout Farida : « ***Contribution à la Sécurité Informatique, Mise en Oeuvre d'un IDS (Intrusion Detection System)*** ». Mémoire d'ingéniorat. Juin 2001.

[11] - Nicolas Baudoin et Marion Karle, « ***NT Réseaux : IDS et IPS*** ». Ingénieurs 2000. 2003-2004.

[12] - D. Burgermeister & J. Krier (2006), « ***Les systèmes de détection d'intrusion*** », www.developpez.com

[13]- H.Debar, M.Dacier et A.Wespi, « ***A revised taxonomy for intrusion detection systems*** ». Annales des télécommunications. July-August 2000.

[14] - Kenaza Tayeb, « ***Détection d'intrusion coopérative basée sur la fusion de données*** ». Thèse de Magister de l'INA. 2006.

[15]- A.Phillip, Porras et Alfonso Valdes, « ***Live traffic analysis of tcp/ip getwayes*** ». Proc. ISOC Symposium on Network and Distributed System Security (NDSS98). (San Diego, CA, March, 98), Internet Society.

[16]- http://marc.boget.free.fr/stage-html1/Memoire090.html

[17] - Boussad Ait Salem et Mourad Brahimi, « ***Conception et réalisation d'un système de détection d'intrusions*** ». Mémoire d'ingéniorat. 2005.

[18] - Ludovic Mé et Cédric Michel, « ***La détection d'intrusion : bref aperçu et derniers développements*** ». Mars 1999

[19] - Jean-Marc Percher et Bernard Jouga, « ***Détection d'intrusion dans les réseaux AD HOC*** ». Ecole Supérieure d'Electronique de l'Ouest (ESEO), France.

[20] - Jacob Zimmermann et Ludovic Mé, « ***Les systèmes de détection d'intrusions : principes algorithmiques*** ». Supélec, équipe SSIR.
Url: http://www.supelecrennes.fr/rennes/si/equipe/lme/.

[21] - Ludovic Mé et Véronique Alanou, « ***Détection d'intrusion dans un système informatique : Méthodes et outils*** ».

[22]- D. Burgermeister & J. Krier (2006), « ***Les systèmes de détection d'intrusion*** », www.developpez.com.

[23]- P. Bernard ESPINASSE (2010), « **Introduction aux Ontologies** », l'Université d'Aix-Marseille, http://www.lsis.org/

[24] -T. Gruber, « ***A translation approach to portable ontology specification*** », 1993.

[25] - R. Neches, R.E. Fikes, T. Finin, T. Gruber, T. Senator, W.R. Swartout, « ***Enabling technology for knowledge sharing*** », AI Magazine, 1991.

[26] - Kais Salhi. «***LA FUSION DES ONTOLOGIES***». Université du Québec à MONTRÉAL (**JUILLET 2014**).

[27] - W. N. Borst, « ***Construction of engineering ontologies*** ». University of Tweenty, Enschede, Centre for Telematica and Information Technology, 1997.

[28]- T.R. Gruber, ***"Toward principles for the design of ontologies used for knowledge sharing"***. International Journal of Human Computer Studies. 1995.

[29]- N. Guarino, ***"Understanding, building, and using ontologies"***. International Journal of Human-Computer Studies, 46: 293-310. 1997.

[30]- M. Gruninger and M.S. Fox, ***"Methodology for the Design and Evaluation of Ontologies"***. In: Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-95, Montreal, 1995.

[31]- A. Gómez-Pérez, M. Fernandez-Lopez, O. Corcho, ***"OntoWeb: Ontology –based information exchange for knowledge management and electronic commerce"***, Deliverable 1.3: A survey on ontology tools, IST-2000-29243, 31 st May, 2002.

[32]- Recommendation W3C RDF, 2004. http://www.w3.org/TR/rdf-syntax-grammar/.

[33]- Recommendation W3C RDF Schema, 2004. http://www.w3.org/TR/rdf-schema/.

[34]- Recommendation W3C OWL, 2004. http://www.w3.org/TR/owl-ref/.

[35]- A. Uszok, J. M. Bradshaw, R. Jeffers, N. Suri, P. J. Hayes, M. R. Breedy, L. Bunch, M. Johnson, S. Kulkarni, and J. Lott, ***"Kaos policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement"***. In Proceedings of POLICY 2003, Como, Italy. 2003.

[36]- G. Tonti, J. M. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok, ***"Semantic web languages for policy representation and reasoning: A comparison of kaos, rei, and ponder"***. In Proceedings of the International Semantic Web Conference 2003, pages 419–437, Sanibel Island, Florida. 2003.

[37]- L. Kagal, ***"Rei : A policy language for the me-centric project"***. Technical report, HP Labs. 2002.

[38]- Natalya F. Noy and Deborah L. McGuinness, ***"Ontology Development 101: A Guide to Creating Your First Ontology"***. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001

[39] -CVE: Common vulnerabilities exposures (CVE), the key to information sharing. http://cve.mitre.org/

[40] -CAPEC: Common attack pattern enumeration and classification (capec). http://capec.mitre.org/

[41] -Cuppens, F., Ortalo, R.: LAMBDA: « ***A language to model a database for detection of attacks. In: Proceedings of the Third International Workshop on Recent Ad- vances in Intrusion Detection*** ». RAID '00, London, UK, UK, Springer-Verlag (2000)

[42] -Eckmann, S.T., Vigna, G., Kemmerer, R.A.: « ***STATL: An attack language for state-based intrusion detection*** ». Journal of Computer Security 10(1) (2002) 71-103

[43] -Debar, H., Curry, D., Feinstein, B.: « ***The intrusion detection message exchange*** » format (idmef). (2007)

[44] -Mitre Corporation: « ***A standardized common event expression (CEE) for event Interoperability*** »

[45] -Valeur, F., Vigna, G., Kruegel, C., Kemmerer, ***« R. Comprehensive approach to intrusion detection alert correlation. Dependable and Secure Computing, IEEE*** » Transactions on 1(3) (2004) 146-169

[46]- http://www.yoyodesign.org/doc/w3c/owl-guide.

[47]- Bijan Parsia, Evren Sirin , Bernardo Cuenca Grau, Edna Ruckhaus" ***Cautiously ApproachingSWRL***".UniversityofMaryland, http://archive.wikiwix.com/cache/?url=http%3A%2F%2Fwww.mindswap.org%2Fpapers%2FCautiousSWRL.pdf.

[48]- http://www.jessrules.com/docs/71/.

[49]- http://web-semantique.developpez.com/.