REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A /Mira de Bejaïa
Faculté des Sciences Exactes
Département d'Informatique

Mémoire de Master professionnel en Informatique

Option

Administration et sécurité des réseaux informatique

Thème

Détection d'intrusions dans les réseaux LAN : Installation et configuration de l'IDS-SNORT

Présenté par :

M. **TOUATI** Azeddine

Soutenu devant le jury compose de:

Président du jury :	M. **OMAR** Mawloud
Examinateur :	M. **BAADACHE** A/Rahman
Encadreur :	M. **BOUKERRAM** AbdAllah

Promotion 2015/2016

Mes vifs remerciements sont adressés :

En premier lieu le bon Dieu de m'avoir donné la vie, la santé et la force de mener à terme mon projet, à mes parents pour leur soutenu et pour tous les sacrifices consentis, ainsi qu'à tous mes frères et sœurs, à mes beaux frères et mes belles sœurs, à mes cousins et amis.

A mon encadreur pour qui ma guider tout au long de ce projet

Aux membres de la commission d'avoir accepté de jugé ce modeste travail

Et à tous ceux qui ont participé de près ou de loin à l'accomplissement de ce projet.

Table des matières

Tables des figures

ACID	**A**nalyse **C**onsol for **I**ntrusions **D**atabase
ARP	**A**ddress **R**esolution **P**rotocol
CD ROM	**C**ompact **D**isc **R**ead **O**nly **M**emory
CERT	**C**omputer **E**mergency **R**esponse **T**eam
CIDR	**C**lassless **I**nter-**D**omain **R**outing
CPU	**C**entral **P**rocessing **U**nit
DAQ	**D**ata **A**cquisition
DDoS	**D**istributed **D**enial **o**f **S**ervice
DMZ	**D**e**M**ilitarized **Z**one
DNS	**D**omain **N**ame **S**ervice
DVD	**D**igital **V**ersatile **D**isc
FTP	**F**ile **T**ransfer **P**rotocol
H-IDS	**H**ost **B**ased **I**ntrusions **D**etection **S**ystem
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
HTTPS	**H**yper**T**ext **T**ransfer **P**rotocol **S**ecure
ICMP	**I**nternet **C**ontrol Message **P**rotocol
IDS	**I**ntrusions **D**etection **S**ystem
IDWG	**I**ntrusions **D**etection exchange format**W**orking **G**roup
IETF	**I**nternet **E**ngineering **T**ask **F**orce
IGRP	**I**nterior **G**ateway **R**outing **P**rotocol
IP	**I**nternet **P**rotocol
IPS	**I**ntrusions **P**revention **S**ystem
IPSec	**I**nternet **P**rotocol **S**ecurity
IPX	**I**nternetwork **P**acket **E**xchange
ISS	**I**nternet **S**ecure **S**ystem
LAN	**L**ocal **A**rea **N**etwork
NFS	**N**etwork **F**ile **S**ystem
N-IDS	**N**etwork **B**ased **I**ntrusions **D**etection **S**ystem
NMAP	**N**etwork **M**apper
NTP	**N**etwork **T**ime **P**rotocol
OSPF	**O**pen **S**hortest **P**ath First
OSI	**O**pen **S**ystems **I**nterconnection
PHP	**H**ypertext **P**reprocessor
POP3	**P**ost **O**ffice **P**rotocol **V**ersion 3
PSSI	**P**olitique de **S**écurité des **S**ystèmes d'**I**nformation
RIP	**R**outing **I**nformation **P**rotocol
RPC	**R**emote **P**rocedure **C**all
SI	**S**ystème d'**I**nformation
SGBD	**S**ystème de **G**estion de **B**ase de **D**onnées
SSH	**S**ecure **S**hell
SSI	**S**écurité des **S**ystèmes d'**I**nformation
SSL	**S**ecure **S**ocket **L**ayer
TCP	**T**ransmission **C**ontrol **P**rotocol
TLS	**T**ransport **L**ayer **S**ecurity

UDP	**U**ser **D**atagram **P**rotocol
URI	**U**niform **R**esource **I**dentifier
URL	**U**niform **R**esource **L**ocator
USB	**U**niversal **S**erial **B**us
VPN	**V**irtual **P**rivate **N**etwork

Introduction générale

Les réseaux informatiques sont devenus beaucoup plus important qu'ils en aient il y a quelques années. De nos jours les entreprises dès leur création n'hésitent pas à mettre en place un réseau informatique pour faciliter la gestion de leur infrastructure, c'est pour cela que la sécurité de ces réseaux constitue un enjeu crucial.

La sécurité d'un système informatique repose en premier lieu sur la mise en place d'une politique de sécurité. Une fois la politique de sécurité définie, il convient de la mettre en œuvre au sein du système informatique. Deux approches non exclusives sont envisageables : la prévention des attaques et leur détection. La première approche, en appliquant un contrôle a priori sur les actions effectuées au sein du système, s'assure que les utilisateurs ne pourront pas violer la politique. Cette approche évite que le système ne se trouve dans un état corrompu, nécessitant une analyse et une correction. De ce fait, des mécanismes de prévention sont présents sur les systèmes informatiques, il s'agit souvent de contrôle d'accès. Cependant, de tels mécanismes possèdent leurs propres limitations, qui peuvent porter sur des aspects théoriques des modèles sous-jacents ou sur leur implémentation. Ces limitations justifient le recours à des mécanismes de détection d'intrusions (IDS).

Afin de qualifier un IDS, on s'intéresse à sa **fiabilité**, qui est sa capacité à émettre une alerte pour toute violation de la politique de sécurité, et à sa **pertinence**, qui est sa capacité à n'émettre une alerte qu'en cas de violation de la politique de sécurité. Un IDS est parfaitement fiable en absence de faux négatif ; il est parfaitement pertinent en l'absence de faux positif.

Notre travail s'articule autour de ce domaine dont il consiste à sécuriser un réseau LAN à l'aide d'un système de détection d'intrusions.

Le premier chapitre est un chapitre descriptif pour la sécurité des réseaux, sur lequel on va définir les menaces, les logiciels malveillants et une politique de sécurité ainsi les principaux mécanismes de sécurité.

Le second chapitre sera consacré à présenter une architecture globale d'un IDS, la définition et le mode de fonctionnement de ce dernier. Ainsi la classification des IDS et enfin la méthode de détection d'une intrusion.

Le dernier chapitre sera consacré à la mise en œuvre de Snort, nous détaillerons l'installation de Snort sous Linux, ainsi tous les paramétrages nécessaires afin de le rendre fonctionnel. Enfin, nous testerons la fiabilité de notre solution en lançant quelques attaques réelles dans le but de suivre son comportement

Chapitre 1 : Les niveaux de sécurité

Introduction

L'informatique et en particulier l'Internet jouent un rôle grandissant dans le domaine des réseaux. Un grand nombre d'applications critiques d'un point de vue de leur sécurité sont déployées dans tous les secteurs professionnels. La sécurité de ces réseaux constitue un enjeu crucial, le contrôle des informations traitées et partagées au sein de ces réseaux devient alors un problème majeur d'autant plus que les réseaux sont interconnectés entre eux, ce qui complexifie donc la tâche des responsables de la sécurité.

Tout au long de ce chapitre, notre intérêt se portera sur les principales menaces pesant sur la sécurité des réseaux ainsi que les mécanismes de défense.

1. Sécurité des réseaux

1.1. Définition

La sécurité d'un réseau est un ensemble de moyens techniques, organisationnels, juridiques et humains nécessaires et mis en place pour conserver, rétablir, et garantir sa sécurité contre les menaces accidentelles ou intentionnelles. En général, la sécurité d'un réseau englobe celle du système informatique sur lequel il s'appuie. [1]

1.2. Évaluation de la sécurité d'un réseau

La sécurité d'un réseau peut s'évalué sur la base d'un certain nombre de critères de sécurité. On distingue généralement trois principaux critères de sécurité [1]:

- ***Disponibilité*** : Elle consiste à garantir l'accès à un service ou à une ressource.
- ***Intégrité*** : Elle consiste à s'assurer que les données n'ont pas été altérées durant la communication (de manière fortuite ou intentionnelle).
- ***Confidentialité*** : Elle consiste à rendre l'information inintelligible à d'autres personnes que les seuls acteurs concernés.

En plus de ces trois critères, on peut ajouter les critères suivants:

- ***Authentification*** : Elle consiste à assurer l'identité d'un utilisateur, c'est-à dire de garantir à chacun des correspondants que son partenaire est bien celui qu'il croit être.
- ***Non répudiation*** : Elle consiste à garantie qu'aucun des correspondants ne pourra nier la transaction.

L'évaluation de la sécurité d'un système informatique est un processus très complexe basé en général sur une méthodologie. Cette évaluation passe par une analyse de risques. Cette dernière pesant sur un système informatique elle-même s'appuie sur un ensemble de règles définies au préalable [1].

1.3. Les Causes pour sécuriser les réseaux

1.3.1. Les enjeux

a) **Enjeux économique :** Les organismes ou entreprises à but lucratif ont presque toujours la même finalité : c'est de réaliser des bénéfices sur l'ensemble de leurs activités. Cette réalisation est rendue possible grâce à son système d'information considéré comme moteur de développement de l'entreprise. D'où la nécessité de garantir la sécurité de ce dernier. La concurrence fait que des entreprises s'investissent de plus en plus dans la sécurisation de leurs systèmes d'information et dans la qualité de service fournit aux clients [1].

b) **Enjeux politiques :** La plupart des entreprises ou organisations se réfèrent aux documents officiels de sécurité élaborés et recommandés par l'État. Ces documents contiennent généralement des directives qui doivent être appliquées par toute structure engagée dans un processus de sécurisation du réseau. Dans le cadre du chiffrement des données par exemple, chaque État définit des cadres et mesures d'utilisation des algorithmes de chiffrement et les recommande aux entreprises exerçant sur son territoire. Le non-respect de ces mesures et recommandations peut avoir des conséquences graves sur l'entreprise. A ce niveau, l'enjeu est plus politique parce que chaque État souhaite être capable de décrypter toutes les informations circulant dans son espace [1].

c) **Enjeux juridiques** : Dans un réseau, on retrouve de l'information multiforme (numérique, papier, etc.). Le traitement de celle-ci doit se faire dans un cadre bien définit et dans le strict respect des lois en vigueur. En matière de juridiction, le non-respect des lois et exigences relatives à la manipulation des informations dans un système d'information peut avoir des conséquences graves sur l'entreprise. [1]

1.3.2. Les Vulnérabilités

Tous les systèmes informatiques sont vulnérables. Peu importe le niveau de vulnérabilité de ceux-ci. Une vulnérabilité est une faille ou une faiblesse pouvant être exploitée par une personne mal intentionnée pour nuire. [1]

Les vulnérabilités des systèmes peuvent être classées en catégorie (humaine, technologique, organisationnelle, mise en œuvre).

a) **Vulnérabilités humaines** : L'être humain de par sa nature est vulnérable. La plupart des vulnérabilités humaines proviennent des erreurs (négligence, manque de compétences, surexploitation, etc.), car ne dit-on pas souvent que l'erreur est humaine? Un SI étant composé des humains, il convient d'assurer leur sécurité si l'on veut garantir un maximum de sécurité dans le SI. [1]

b) **Vulnérabilités technologiques :** Avec la progression exponentielle des outils informatiques, les vulnérabilités technologiques sont découvertes tous les jours. Ces vulnérabilités sont à la base dues à une négligence humaine lors de la conception et la réalisation. Pour être informé régulièrement des vulnérabilités technologiques

découvertes, il suffit de s'inscrire sur une liste ou des listes de diffusion mises en place par les CERT (Computer Emergency Readiness ou Response Team). [1]

c) **Vulnérabilités organisationnelles :** Les vulnérabilités d'ordre organisationnel sont dues à l'absence des documents cadres et formels, des procédures (de travail, de validation) suffisamment détaillées pour faire face aux problèmes de sécurité du système. Quand bien même ces documents et procédures existent, leur vérification et mises à jour ne sont pas toujours bien assurées. [1]

d) **Vulnérabilités mise en œuvre :** Les vulnérabilités au niveau mise en œuvre peuvent être dues au non prise en compte de certains aspects lors de la réalisation d'un projet. [1]

1.3.3. Les Menaces

Une menace est un événement, d'origine accidentelle ou délibérée, capable s'il se réalise de causer un dommage au sujet étudié. Le réseau informatique comme tout autre réseau informatique est en proie à des menaces de toutes sortes qu'il convient de recenser. [1]

On peut également classer les menaces en deux catégories :

- ***Les menaces passives*** : consistent essentiellement à copier ou à écouter l'information sur le réseau, elles nuisent à la confidentialité des données. Dans ce cas, celui qui prélève une copie n'altère pas l'information elle-même. [1]
- ***Les menaces actives*** : consistent à altérer des informations ou le bon fonctionnement d'un service. [1]

1.3.4. Les logiciels malveillants

Ce sont des logiciels développés par des hackers dans le but de nuire à un système d'informations.

- **Les Virus :** un virus est un segment de programme qui, lorsqu'il s'exécute, se reproduit en s'adjoignant à un autre programme (du système ou d'une application), et qui devient ainsi un cheval de Troie. Puis le virus peut ensuite se propager à d'autres ordinateurs (via un réseau) à l'aide du programme légitime sur lequel il s'est greffé. Il peut également avoir comme effets de nuire en perturbant plus ou moins gravement le fonctionnement de l'ordinateur infecté. [2]

  Les virus peuvent être classés suivant leur mode de propagation et leurs cibles:[3]

  - ✓ **Le virus de boot** : il est chargé en mémoire au démarrage et prend le contrôle de l'ordinateur.
  - ✓ **Le virus d'application** : ils infectent les programmes exécutables, c'est-à-dire les programmes (.exe, .com ou .sys) en remplaçant l'amorce du fichier, de manière à ce que le virus soit exécuté avant le programme infecté. Puis ces virus rendent la main au programme initial, camouflant ainsi leur exécution aux yeux de l'utilisateur.

- ✓ **La macro virus** : il infecte des logiciels de la suite Microsoft Office les documents bureautiques en utilisant leur langage de programmation, qui contaminera tous les documents basés sur lui, lors de leur ouverture.

- **Les Vers :** Un ver est un programme autonome qui se reproduit et se propage à l'insu des utilisateurs. Contrairement aux virus, un ver n'a pas besoin d'un logiciel hôte pour se dupliquer. Le ver a habituellement un objectif malicieux, par exemple :

  - ✓ Espionner l'ordinateur dans lequel il réside ;
  - ✓ Offrir une porte dérobée à des pirates informatiques ;
  - ✓ Détruire des données sur l'ordinateur infecté ;
  - ✓ Envoyer de multiples requêtes vers un serveur internet dans le but de le saturer. [2]

- **Les chevaux de Troie :** Un cheval de Troie est une forme de logiciel malveillant déguisé en logiciel utile. Son but : se faire exécuter par l'utilisateur, ce qui lui permet de contrôler l'ordinateur et de s'en servir pour ses propres fins. Généralement d'autres logiciels malveillants seront installés sur votre ordinateur, tels que permettre la collecte frauduleuse, la falsification ou la destruction de données. [2]

- **Les logiciels espion :** (Espiogiciel ou logiciel espion) est un programme ou un sous-programme, conçu dans le but de collecter des données personnelles sur ses utilisateurs et de les envoyer à son concepteur, ou à un tiers via Internet ou tout autre réseau informatique, sans avoir obtenu au préalable une autorisation explicite et éclairée desdits utilisateurs. [2]

- **Le spam :** correspond à l'envoi intempestif de courriers électroniques, publicitaires ou non, vers une adresse mail. Le spam est une pollution du courrier légitime par une énorme masse de courrier indésirable non sollicité. [4]

1.3.5. Les intrusions

Une intrusion est définie comme une faute malveillante d'origine interne ou externe résultant d'une attaque qui a réussi à exploiter une vulnérabilité. Elle est susceptible de produire des erreurs pouvant provoquer une défaillance vis-à-vis la sécurité, c'est-à-dire une violation de la politique de sécurité du système. Le terme d'intrusions sera employé dans le cas où l'attaque est menée avec succès et où l'attaquant a réussi à s'introduire et/ou compromettre le système. [5]

1.3.6. Les Attaques

a) Définition : Une attaque est définie comme faute d'interaction malveillante visant à violer une ou plusieurs propriétés de sécurité. C'est une faute externe créée avec l'intention de nuire, y compris les attaques lancées par des outils automatiques : vers, virus, etc. La notion d'attaque ne doit pas être confondue avec la notion d'intrusions. [5]

b) Les motivations d'une attaque : Les motivations des attaques peuvent être liées à divers objectif : [6]

- Obtenir un accès au système ;
- Voler des informations, tels que des secrets industriels ou des propriétés intellectuelles ;
- Collectionner des informations personnelles sur un utilisateur
- S'informer sur l'organisation ;
- Récupérer des données bancaires ;
- S'informer sur l'organisation (entreprise de l'utilisateur, etc.) ;
- Troubler le bon fonctionnement d'un service ;
- Utiliser le système de l'utilisateur comme « rebond » pour une attaque ;
- Utiliser les ressources du système de l'utilisateur, notamment lorsque le réseau sur lequel il est situé possède une bande passante élevée ;
- Faire du chantage ;
- Par simple jeu ou par défi ;
- Pour terrorisme ou pour des fins politique ;
- Pour apprendre ;

c) Type d'attaques : Les hackers utilisent plusieurs techniques d'attaques. Ces attaques peuvent être regroupées en trois familles différentes : [7]

- **Les attaques directes :** C'est la plus simple des attaques. Le hacker attaque directement sa victime à partir de son ordinateur. La plus part des hackers utilisent cette technique. En effet, les programmes de hack qu'ils utilisent ne sont que faiblement paramétrable, et un grand nombre de ces logiciels envoient directement les paquets à la victime. [7]

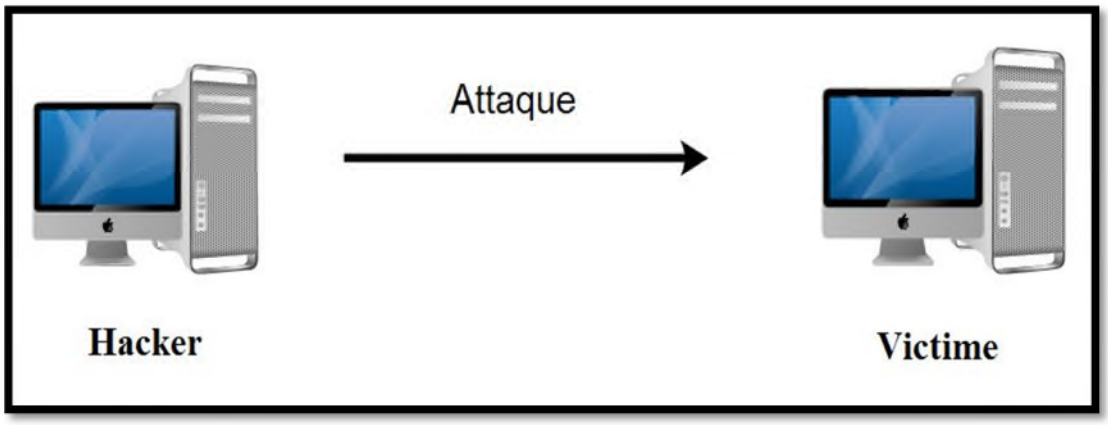


Figure 1.1 : Attaque directe

- **Les attaques indirectes par rebond :** Cette attaque est très prisée des hackers. En effet, le rebond a deux avantages :
  - Masquer l'identité du hacker.
  - Utiliser les ressources de l'ordinateur intermédiaire car il est plus puissant pour attaquer.

Le principe en lui-même, est simple : Les paquets d'attaque sont envoyés à l'ordinateur intermédiaire, qui répercute l'attaque vers la victime. D'où le terme par rebond.

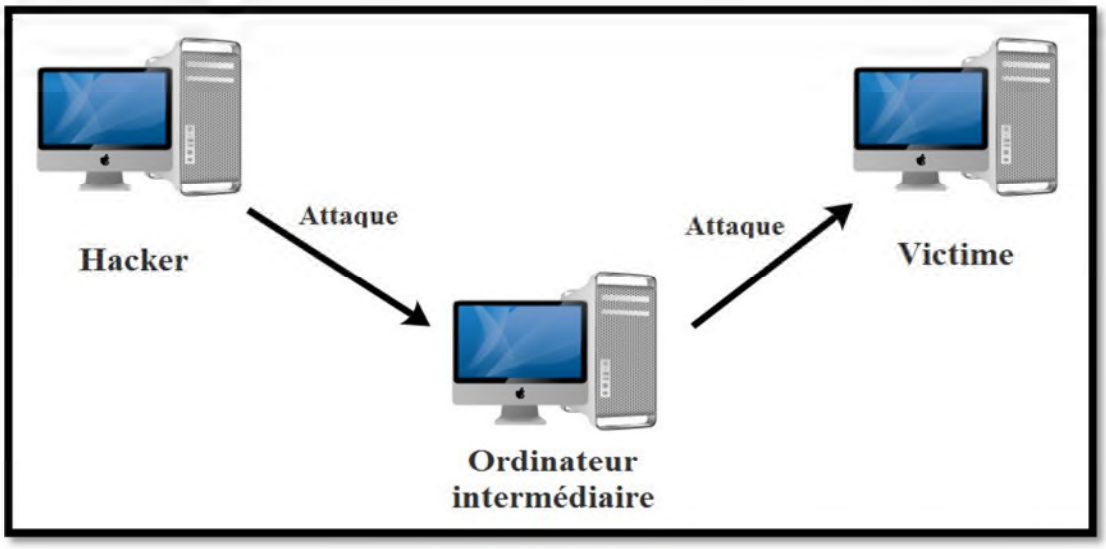


Figure 1.2 : Attaque indirecte par rebond

- **Les attaques indirectes par réponse :** Cette attaque est un dérivé de l'attaque par rebond. Elle offre les mêmes avantages, du point de vue du hacker. Mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête. Et c'est cette réponse à la requête qui va être envoyée à l'ordinateur victime

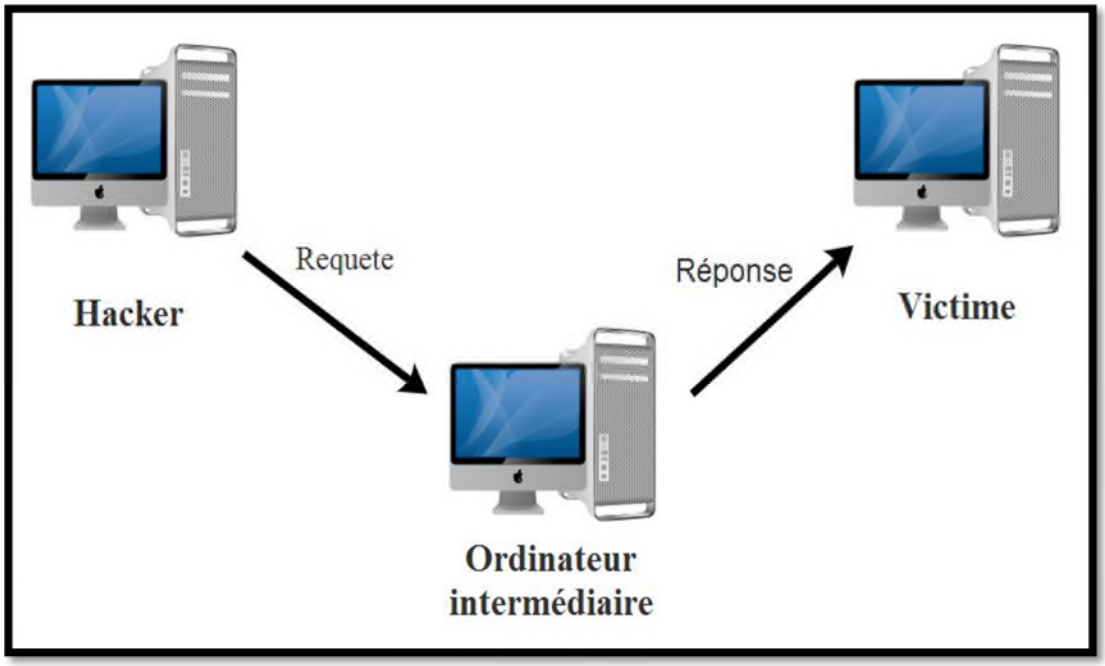


Figure 1.3 : Attaque indirecte par réponse

d) Catégorie des attaques : Il existe quatre catégorie d'attaques : [2]

- **Attaques par interruption :** c'est une attaque protée à la disponibilité. La destruction d'une pièce matérielle (tel un disque dure), la coupure d'une ligne de communication, ou la mise hors service d'un système de gestion de fichiers en sont des exemples.

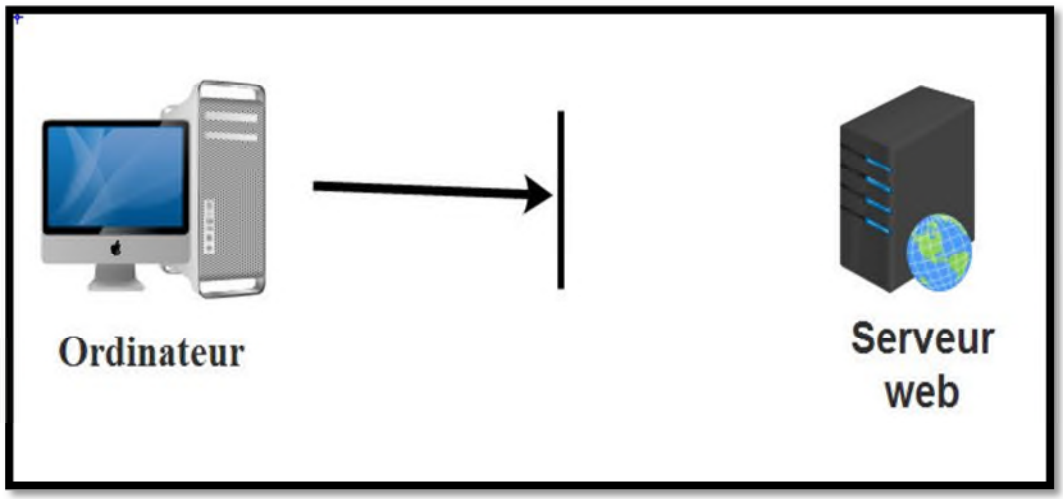


Figure 1.4 : Attaque par interruption

- **Attaque par interception :** C'est une attaque portée à la confidentialité. Il peut s'agir d'une personne, d'un programme ou d'un ordinateur. Une écoute téléphonique dans le but de capturer des données sur un réseau, ou la copie non autorisée de fichiers ou de programme en sont des exemples.

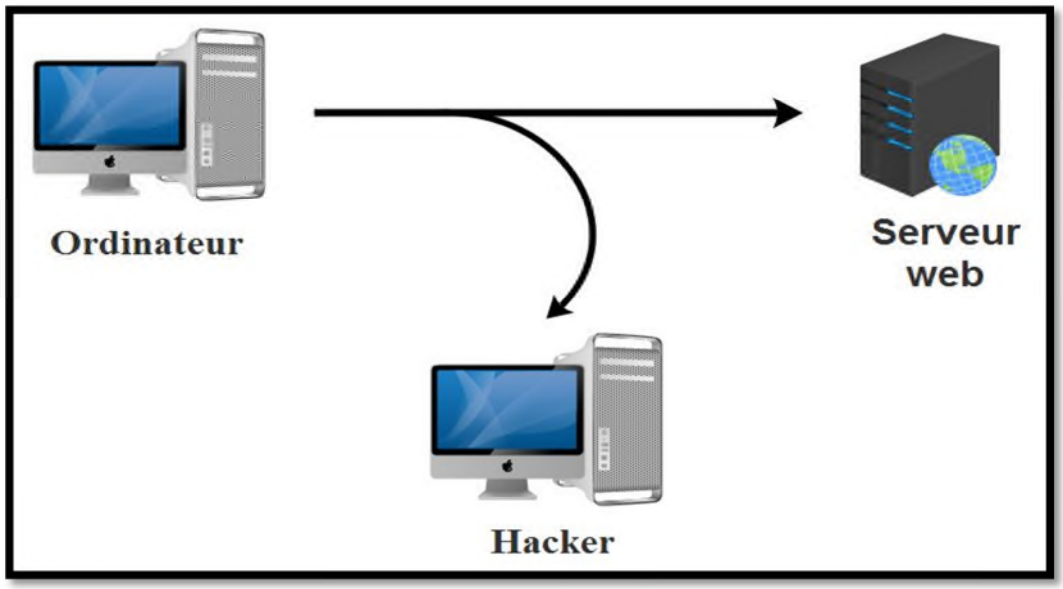


Figure 1.5 : Attaque par interception

.

- **Attaque par modification :** Il s'agit d'une attaque portée à l'intégrité. Changer des valeurs dans un fichier de données, altérer un programme de

façon à bouleverser son comportement ou modifier le contenu de messages transmis sur un réseau sont des exemples de telles attaques

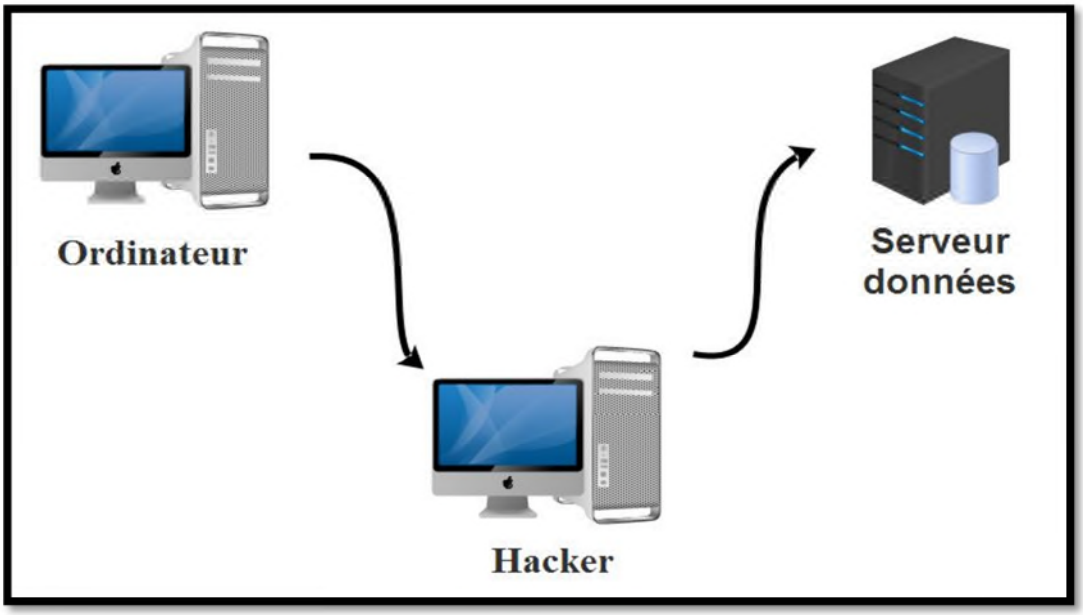


Figure 1.6 : Attaque par modification

- **Attaque par fabrication :** C'est une attaque portée à l'authenticité. Il peut s'agir de l'insertion de faux messages dans un réseau ou l'ajout d'enregistrements à un fichier.

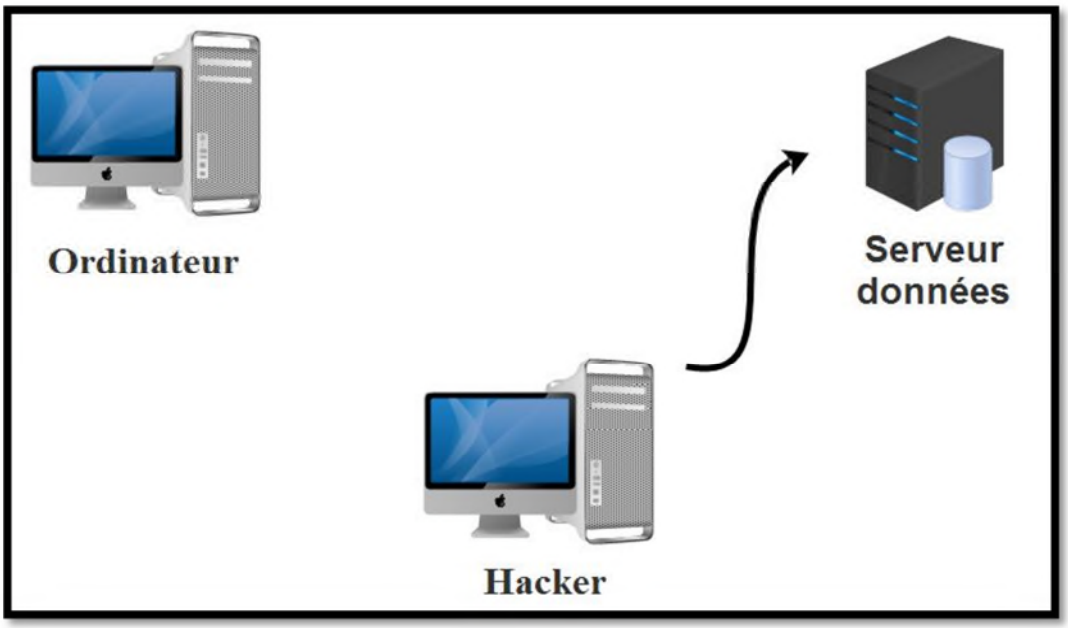


Figure 1.7 : Attaque par fabrication

e) Quelques attaques courantes

Le Déni de Service : Ce genre d'attaques (denial of service en anglais ou DoS) sont des attaques qui visent à rendre une machine ou un réseau indisponible durant une certaine période. Cette attaque est dangereuse quand elle vise les entreprises dépendantes de leur infrastructure réseau. [7]

Le SYN flood : Cette attaque utilise des paquets TCP contenant le flag SYN. Ce flag signifie initier une connexion avec la cible. En envoyant un nombre très important de ces paquets, on oblige le serveur à démarrer un socket de connexion pour chaque requête, il enverra donc des paquets contenant les flags SYN, ACK pour établir la connexion mais ne recevra jamais de réponses. Le serveur ayant arrivé à saturation à cause de la grande fille d'attente de connexion ne pourra plus pouvoir répondre aux connexions légitimes des utilisateurs. [W1]

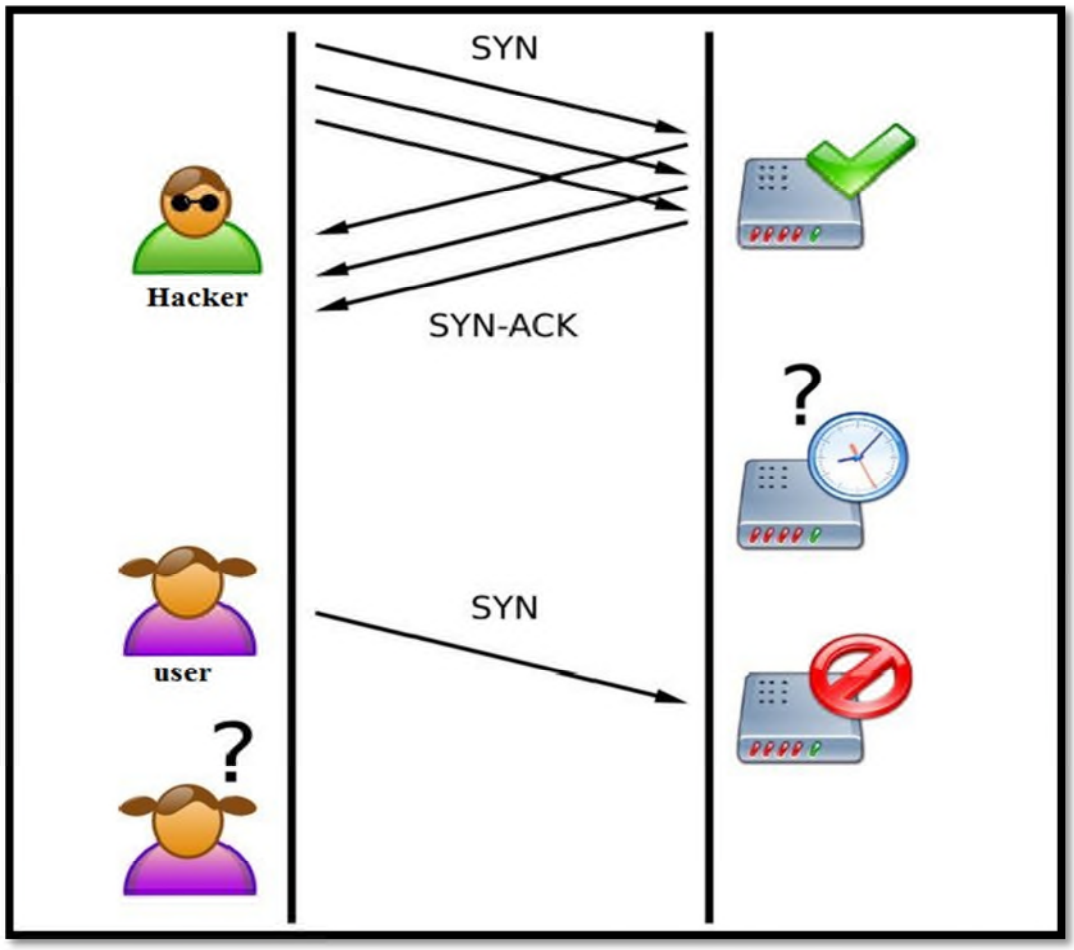


Figure 1.8 : l'attaque SYN Flood

Le PING flood : Elle consiste à simplement envoyer un nombre maximal de PING simultané jusqu'à saturer la victime. On utilise généralement la commande Ping sous Linux mais une des conditions pour que l'attaque soit efficace est de posséder plus de bande passante que la victime. [7]

Le Smurf : Les attaques Smurf profite d'une faiblesse d'IPv4 et d'une mauvaise configuration pour profiter des réseaux permettant l'envoi de paquets au broadcast. Le broadcast est une adresse IP qui permet de joindre toutes les machines d'un réseau. L'attaquant envoie au broadcast des paquets contenant l'IP source de la victime ainsi chaque machine sur le réseau va répondre à la cible à chaque requête de l'attaquant. On se sert du réseau comme un amplificateur pour perpétrer l'attaque, cette méthode porte aussi le nom d'attaque réfléchie permettant à l'attaquant de couvrir ces traces et de rendre l'attaque plus puissante. [7]

Teardrop Attack : Elle consiste à envoyer des paquets IP invalides à la cible, ces paquets peuvent être fragmentes, ou contenir des données corrompues ou qui dépassent la taille réglementaire. Sur certains systèmes comme les Windows avant 98 ou les Linux avant 2.0.32, ces paquets ne peuvent être interprétés et rendrons la machine inopérante.[7]

Les attaques distribuées : La plupart des attaques, cité plus haut, peuvent être exécutés de manière distribuée, on parle de DDoS pour Distributed Denial of Service. Les attaques distribuées se basent sur ce fait : attaquer une cible toute seule se traduit souvent par un échec, alors que si un grand nombre de machines s'attaquent à la même cible alors l'attaque a plus de chance de réussir. [W2]

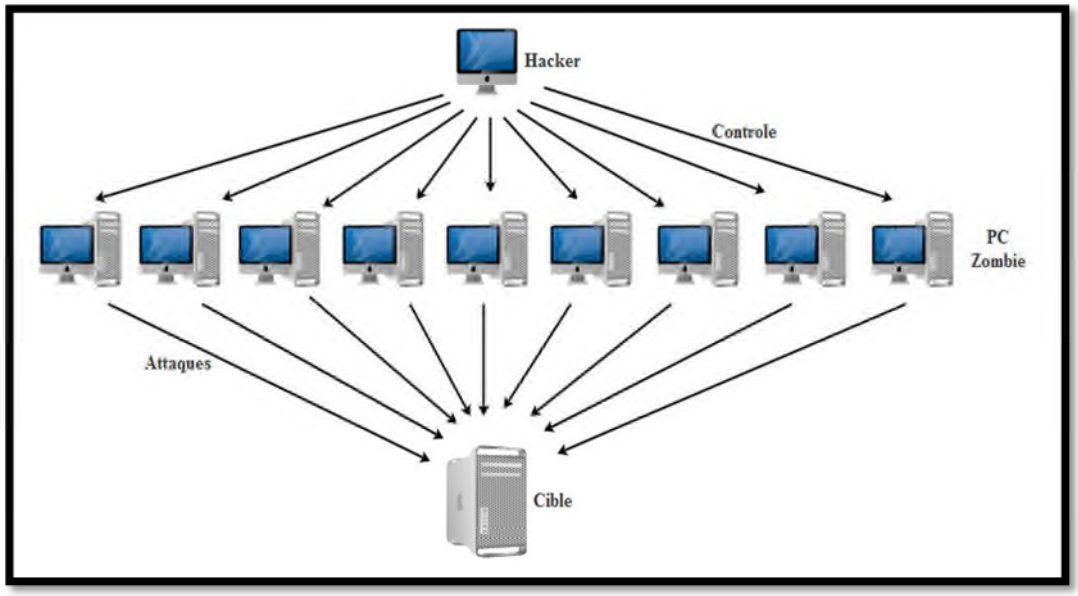


Figure 1.9 : DDoS Attaque

ARP spoof : L'ARP spoof est une attaque très puissante qui permet, en général, de sniffer le trafic sur le réseau en s'interposant entre une ou des victimes et la passerelle. Elle permet même de sniffer et récupérer des mots de passes sur des connexions sécurisés SSL. L'attaque inonde le réseau avec des trames ARP liant l'adresse physique de l'attaquant avec la passerelle. De cette manière, le cache ARP des victimes est corrompu et tout le trafic est redirige vers le poste de l'attaquant. [7]

DNS spoof : De la même manière, on peut corrompre le DNS d'une victime. Normalement, ceci permet de rediriger la victime vers des sites pirates que l'on contrôle mais dans le cadre d'un déni service, on corrompt le cache DNS de fausses informations qui rendront impossible l'accès aux sites web. [7]

IP Spoofing : Il existe plusieurs types d'IP Spoofing. La première est dite Blind Spoofing, c'est une attaque "en aveugle". Les paquets étant forgés avec une adresse IP usurpée, les paquets réponses iront vers cette adresse. Il sera donc impossible à l'attaquant de récupérer ces paquets. Il sera obligé de les "deviner". Cependant, il existe une autre technique que le Blind Spoofing. Il s'agit d'utiliser l'option IP Source Routing qui permet d'imposer une liste d'adresses IP des routeurs que doit emprunter le paquet IP. Il suffit que l'attaquant route le paquet réponse vers un routeur qu'il contrôle pour le récupérer. Néanmoins, la grande majorité des routeurs d'aujourd'hui ne prennent pas en compte cette option IP et jettent tous paquets IP l'utilisant. [7]

Les chevaux de Troie : Leur objectif est le plus souvent d'ouvrir une porte dérobée ("backdoor") sur le système cible, permettant par la suite à l'attaquant de revenir à loisir épier, collecter des données, les corrompre, contrôler voire même détruire le système. Certains chevaux de Troie sont d'ailleurs tellement évolués qu'ils sont devenus de véritables outils de prise en main et d'administration à distance. [7]

Les virus informatiques : Un vers est un programme parasite. Il n'est pas forcément autopropageable. Son but est de grignoter des ressources système : CPU, mémoire, espace disque, bande passante... Ces petits bouts de programme sont dépendants du système d'exploitation ou d'un logiciel. Ils se propagent, comme toutes données binaires, par disquettes, CD ROM, réseaux. [7]

Les buffers overflow : Un buffer overflow est une attaque très efficace et assez compliquée à réaliser. Elle vise à exploiter une faille, une faiblesse dans une application pour exécuter un code arbitraire qui compromettra la cible. [7]

Le Mail Bombing : Elle consiste à envoyer un nombre faramineux d'emails (plusieurs milliers par exemple) à un ou des destinataires. L'objectif étant de : [7]

- Saturer le serveur de mails
- Saturer la bande passante du serveur et du ou des destinataires,
- Rendre impossible aux destinataires de continuer à utiliser l'adresse électronique.

Social Engineering : C'est une technique qui a pour but d'extirper des informations à des personnes. Contrairement aux autres attaques, elle ne nécessite pas de logiciel. La seule force de persuasion est la clé de voûte de cette attaque. Il y a quatre grandes méthodes de social engineering : par téléphone, par lettre, par internet et par contact direct. [7]

1.3.7. Les moyens de sécurisé un réseau

La sécurité d'un réseau c'est la sécurité des éléments qui le compose, il existe plusieurs mécanismes et dispositifs de sécurité, parmi eux :

Les Antivirus

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

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

Les mises à jour système

Pour éviter les dénis de services applicatifs, on doit maintenir tous les logiciels de son système à jour puisque les mises à jour permettent souvent de corriger des failles logicielles, qui peuvent être utilisées par un attaquant, pour mettre l'application hors service, ou pire, le serveur. Il est donc impératif de mettre son système à jour très régulièrement C'est un moyen très simple à mettre en place pour se protéger des attaques applicative.

Editer des options dans les fichiers de configuration qui stocke des données concernant chaque connexion reçue par la machine telle l'adresse IP source, le numéro de port, l'âge de la connexion. En analysant ces données, on peut facilement détecter les comportements suspects et éviter certains types d'attaque.

Les firewalls

En français on dit pare-feu ou garde-barrière, c'est un système permettant de protéger un ordinateur ou un réseau d'ordinateurs des intrusions provenant d'un réseau tiers (notamment internet). Le pare-feu est un système permettant de filtrer les paquets de données échangés avec le réseau, il s'agit ainsi d'une passerelle filtrante comportant au minimum les interfaces réseau suivante :

- une interface pour le réseau à protéger (réseau interne) ;
- une interface pour le réseau externe.

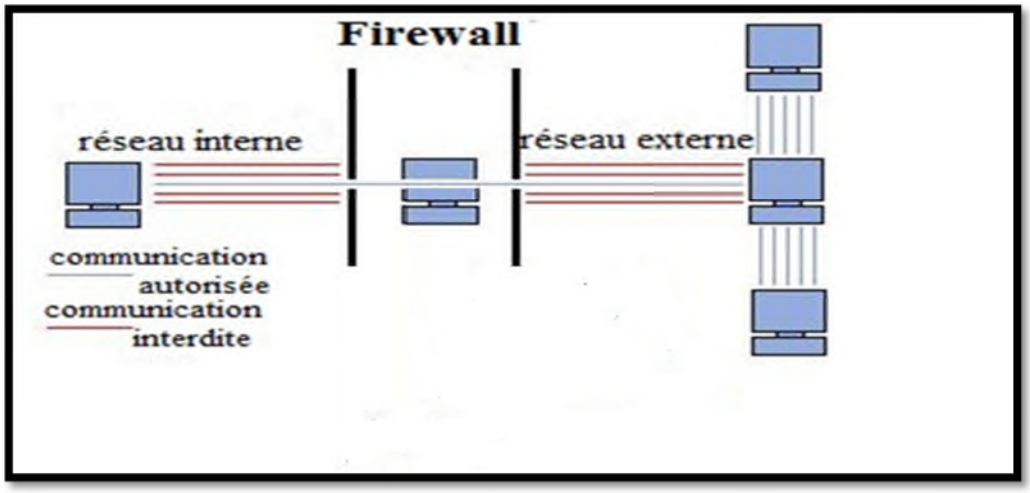


Figure 1.10 : Firewall

Le système firewall est un système logiciel, reposant parfois sur un matériel réseau dédié, constituant un intermédiaire entre le réseau local (ou la machine locale) et un ou plusieurs réseaux externes. Il est possible de mettre un système pare-feu sur n'importe quelle machine et avec n'importe quel système pourvu que :

- La machine soit suffisamment puissante pour traiter le trafic ;
- Le système soit sécurisé ;
- Aucun autre service que le service de filtrage de paquets ne fonctionne sur le serveur.

Fonctionnement d'un système pare-feu

C'est un ensemble de différents composants matériels (physique) et logiciels (logique) qui contrôlent le trafic intérieur/extérieur selon une politique de sécurité. Un système pare-feu fonctionne la plupart du temps grâce à des règles de filtrage indiquant les adresses IP autorisées a communiquer avec les machines aux réseaux, il s'agit ainsi d'une passerelle filtrante.

Il permet d'une part de bloquer des attaques ou connexions suspectes d'accéder au réseau interne.

D'un autre côté, un firewall sert dans de nombreux cas également à éviter la fuite non contrôlée d'informations vers l'extérieur. Il propose un véritable contrôle sur le trafic réseau de l'entreprise, Il permet donc d'analyser, de sécuriser et de gérer le trafic réseau.

Architecture DMZ

Une DMZ (Demilitarized zone) est une zone d'un réseau d'entreprise, située entre le réseau local et Internet, derrière le pare-feu. Il s'agit d'un réseau intermédiaire regroupant des serveurs ou services (HTTP, DHCP, mails, DNS, etc.). Ces serveurs devront être accessibles depuis le réseau interne de l'entreprise et, pour certains, depuis les réseaux externes. Le but est ainsi d'éviter toute connexion directe au réseau interne.

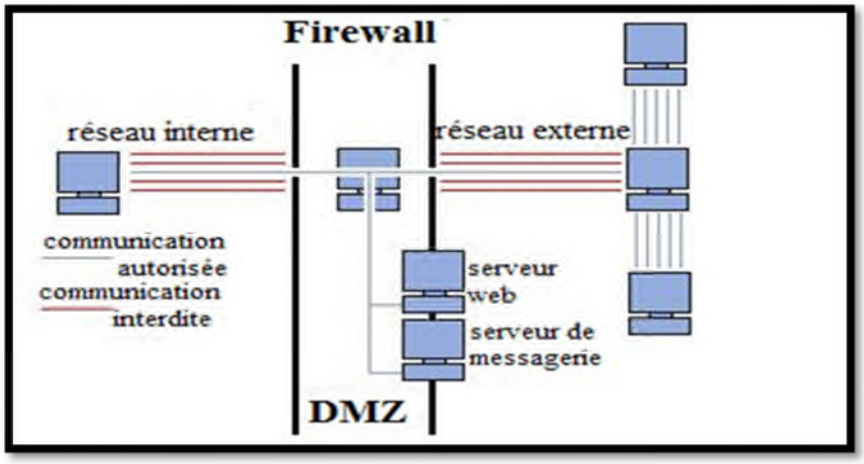


Figure 1.11 : Architecture DMZ

Les VPN

Dans les réseaux informatiques, le réseau privé virtuel (Virtual Private Network en anglais, abrégé en VPN) est une technique permettant aux postes distants de communiquer de manière sûre, tout en empruntant des infrastructures publiques (internet). [8]

Un VPN repose sur un protocole, appelé protocole de tunnelisation, c'est-à-dire un protocole permettant aux données passant d'une extrémité à l'autre du VPN d'être sécurisées par des algorithmes de cryptographie. [8]

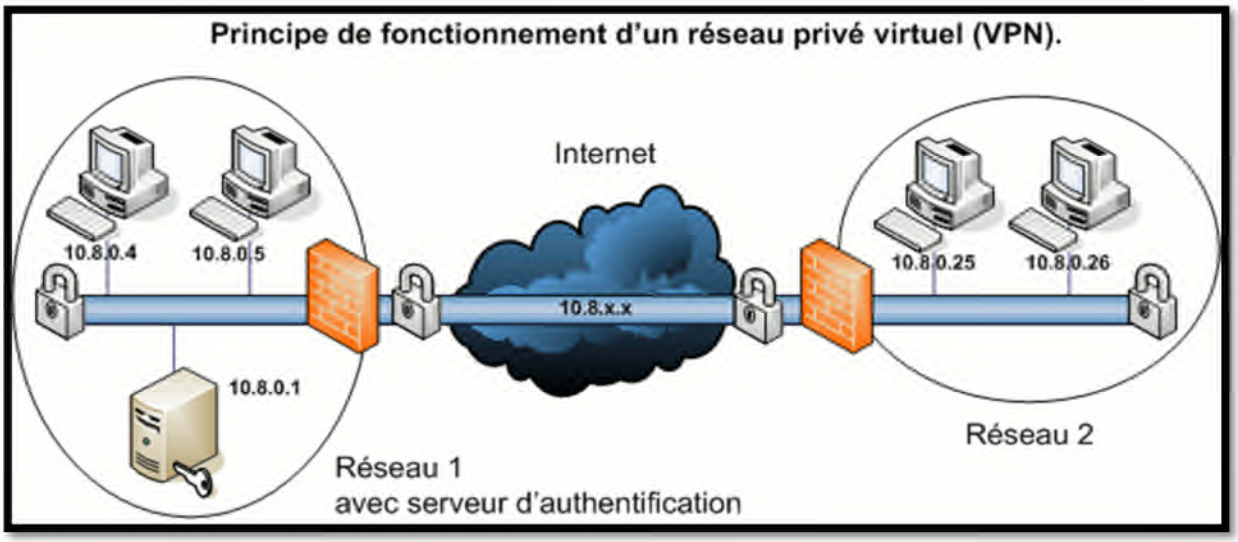


Figure 1.12 : Principe d'un VPN

Les IDS

La détection d'intrusion est définie comme étant un mécanisme écoutant le trafic réseau de manière furtive, afin de repérer des activités anormales ou suspectes et permettant ainsi d'avoir une stratégie de prévention sur les risques d'attaques. Il existe différents types d'IDS, que l'on classe comme suit :

1. **Système de détection d'intrusion réseau (NIDS) :** un NIDS analyse de manière passive les flux transitant sur le réseau et détecte les intrusions en temps réel, en d'autres termes, un NIDS écoute tout le trafic réseau, puis analyse et génère des alertes si des paquets semblent dangereux.
2. **Système de détection d'intrusion de type hôte (HIDS) :** un HIDS est généralement placé sur des machines sensibles, susceptibles de subir des attaques et possédant des données sensibles pour l'entreprise.
3. **Système de détection d'intrusion de type hybride :** il s'agit d'un système capable de réunir des informations provenant d'un système HIDS ainsi que d'un NIDS. Généralement utilisé dans un environnement décentralisé, il permet de réunir les informations de diverses sondes placées sur le réseau.

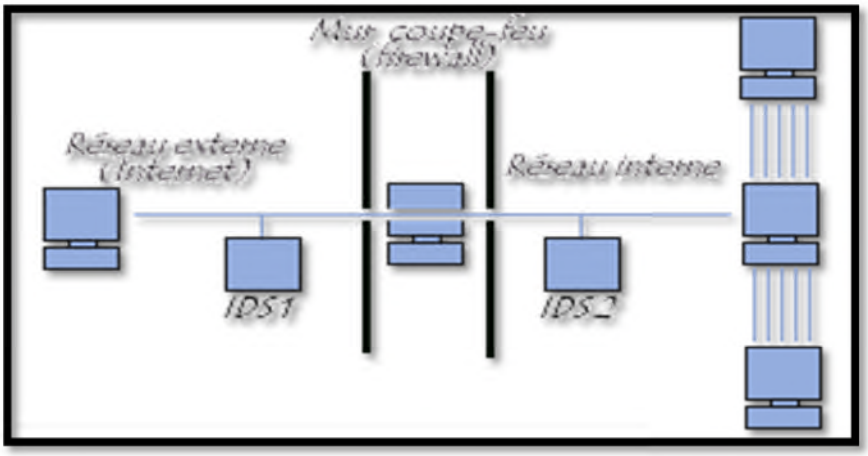


Figure 1.13 : Système de détection d'intrusions

Les IPS

L'IPS est un Système de Prévention/Protection contre les intrusions et non plus seulement de reconnaissance et de signalisation des intrusions comme la plupart des IDS. La principale différence entre un IDS (réseau) et un IPS (réseau) tient principalement en 2 caractéristiques :

- le positionnement en coupure sur le réseau de l'IPS et non plus seulement en écoute sur le réseau pour l'IDS (traditionnellement positionné comme un sniffer sur le réseau).
- la possibilité de bloquer immédiatement les intrusions et quel que soit le type de protocole de transport utilisé et sans reconfiguration d'un équipement tierce, ce qui induit que l'IPS est constitué en natif d'une technique de filtrage de paquets et de moyens de blocages.

La Sensibilisation du personnel

Les politiques de sécurité informatique des entreprises s'appuient généralement sur des techniques de protection et des plans d'urgence mais négligent souvent un aspect : le

personnel. La stratégie idéale dans le domaine de la sécurité informatique ne se limite pas à des techniques de protection et à des consignes complexes. Elle nécessite également une formation appropriée du personnel. Faute d'une sensibilisation de ce dernier, les mesures de sécurité informatique ne sont qu'à moitiés efficaces.

Audits de sécurité

Un audit de sécurité consiste à s'appuyer sur un tiers de confiance (généralement une société spécialisée en sécurité informatique) afin de valider les moyens de protection mis en œuvre, au regard de la politique de sécurité.

L'objectif de l'audit est ainsi de vérifier que chaque règle de la politique de sécurité est correctement appliquée et que l'ensemble des dispositions prises forme un tout cohérent.

Un audit de sécurité permet de s'assurer que l'ensemble des dispositions prises par l'entreprise sont réputées sûres.

Contrôle d'accès

L'accès au système d'information exige une identification et une authentification préalable. L'utilisation de comptes partagés ou anonymes est interdite. Des mécanismes permettant de limiter les services, les données, les privilèges auxquels à accès l'utilisateur en fonction de son rôle dans l'organisation doit être mis en œuvre. [6]

Les accès aux serveurs et aux réseaux doivent être journalisés L'attribution et la modification des accès et privilèges d'un service doivent être validées par le propriétaire du service.

Pour les services sensibles, un inventaire régulièrement mis à jour en sera dressé. Il importe de bien différencier les différents rôles et de n'attribuer que les privilèges nécessaires.

Les protocoles de sécurité

Un protocole est un ensemble de règles et de procédures à respecter pour émettre et recevoir des données sur un réseau. Sur Internet, les protocoles utilisées font partie d'une suite de protocoles TCP/IP, tel que la plus part de ces protocoles ne sont pas sécurisés lors de la transmission des données sur le réseau. Les protocoles sécurisés ont été mis au point, afin d'encapsuler les messages dans des paquets de données chiffrées. On cite parmi ces protocoles les suivants :

- **Protocole SSH (Secure Shell) :** c'est un protocole qui permet à des services TCP/IP d'accéder à une machine à travers une communication chiffrée appelée « tunnel ».
- **Protocole SSL (Secure Socket Layer) :** c'est un procédé de sécurisation des échanges, il a été conçu pour assurer la sécurité des transactions effectuées via Internet.
- **Protocole HTTPS :** HTTPS n'est rien d'autre que HTTP encapsulé dans la couche de chiffrement TLS (Transport Layer Security). En général le serveur est

authentifié par un certificat X509, l'internaute peut s'authentifier par l'intermédiaire d'un serveur RADIUS, ou par un des autres procédés proposés par les logiciels serveur.

- **IPSec (IP Security) :** IPSec (*Internet Protocol Security*) est conçu pour sécuriser le protocole IPv6. La lenteur de déploiement de ce dernier a imposé une adaptation d'IPSec à l'actuel protocole IPv4. On établit un tunnel entre deux sites (voir figure 10.6), et IPSec gère l'ensemble des paramètres de sécurité associés à la communication. Deux machines passerelles, situées à chaque extrémité du tunnel, négocient les conditions de l'échange des informations : quels algorithmes de chiffrement, quelles méthodes de signature numérique ainsi que les clés utilisées pour ces mécanismes. La protection est apportée à tous les trafics et elle est transparente aux différentes applications.

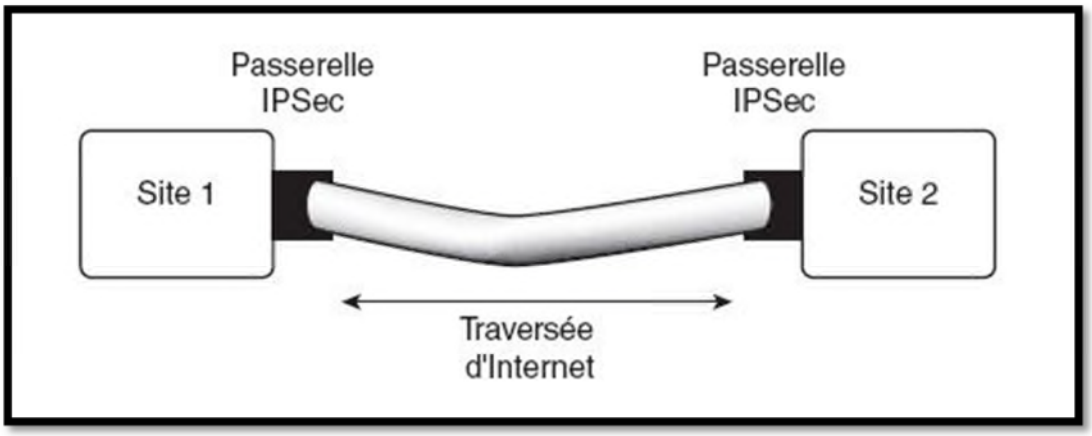


Figure 1.14 : Un tunnel IPSec entre deux sites d'entreprise [2]

Les Algorithme de chiffrements

Il existe deux grandes familles d'algorithmes de chiffrements, ceux à clés symétriques et ceux à clés asymétriques.

- **Algorithme de chiffrement symétrique :** il consiste à utiliser la même clé pour le chiffrement ainsi que pour le déchiffrement. Il est donc nécessaire que les deux interlocuteurs se soient mis d'accord sur une clé privée, ou ils doivent utiliser un canal sécurisé pour l'échanger.
- **Algorithme de chiffrement asymétrique :** c'est une méthode cryptographique faisant intervenir une paire de clés asymétrique (une clé publique et une clé privée). Elle utilise cette paire de clés pour le chiffrement et le déchiffrement. La clé publique est rendue publique et elle est distribuée librement, la clé privée quant à elle n'est jamais distribuée et doit être gardé secrète.

1.3.8. Mise en œuvre d'une politique de sécurité

La politique de sécurité des systèmes d'information est un plan d'actions définies pour maintenir un certain niveau de sécurité. Elle reflète la vision stratégique de la direction de l'entreprise en matière de sécurité des systèmes d'informations (SSI).

Une politique de sécurité s'élabore à plusieurs niveaux :

- sécuriser l'accès aux données de façon logicielle (authentification, contrôle d'intégrité).
- sécuriser l'accès physique aux données : serveurs placés dans des salles blindées avec badge d'accès…
- Un aspect très important pour assurer la sécurité des données d'une entreprise est de sensibiliser les utilisateurs aux notions de sécurité, de façon à limiter les comportements à risque : si tout le monde peut accéder aux salles de serveurs, peut imposte qu'elles soient sécurisées !
- De même, si les utilisateurs laissent leurs mots de passes écrit à côté de leur PC, son utilité est limitée…
- Enfin, il est essentiel pour un responsable de sécurité de s'informer continuellement, des nouvelles attaques existantes, des outils disponibles…de façon à pouvoir maintenir à jour son système de sécurité et à combler les brèches de sécurité qui pourraient exister.

Conclusion

Dans ce chapitre, nous avons présenté un aperçu sur la sécurité informatique dans un réseau et l'importance de la mise en place d'une politique de sécurité en traçant les besoins et les objectifs voulus afin de remédier aux menaces constantes que subi un réseau informatique. Ces menaces se manifestent généralement sous forme d'attaques informatiques que nous avons illustrées dans le but de montrer l'intensité de danger. Enfin nous avons proposé quelques solutions existantes afin de se protéger et réduire les risques.

Chapitre 2 : Les systèmes de détection d’intrusions

Introduction

Une propriété de valeur doit être protégée contre le vol et la destruction. Certaines maisons sont équipées de systèmes d'alarme qui peuvent décourager des voleurs, prévenir les autorités dans le cas d'une effraction et même avertir les propriétaires que leurs maisons est en feu. De telles mesures sont nécessaires pour assurer l'intégrité des maisons et la sécurité de leurs propriétaires. [W3]

La même assurance d'intégrité et de sécurité devrait également être appliquée aux systèmes et données informatiques. L'internet a facilité le flux d'informations, personnelles, financières et autres. En même temps, il a également promu autant de dangers. Les utilisateurs malveillants recherchent des proies vulnérables comme les systèmes sans correctifs, les systèmes affectés par des chevaux de Troie et les réseaux exécutant des services peu sûrs. Des alarmes sont nécessaires pour prévenir les administrateurs et les membres de l'équipe de sécurité qu'une effraction s'est produite afin qu'ils puissent répondre en temps réel au danger. Les systèmes de détection d'intrusions ont été conçus pour jouer le rôle d'un tel système d'alarme. [W3]

Dans ce chapitre nous présentons tout d'abord la notion de système de détection d'intrusions ainsi que son architecture. Je présente également la classification des IDS, dans ce cadre plusieurs critères sont pris en compte nous commençons par la classification selon la méthode d'analyse qui découpe les IDS en deux approches (comportementale et par signatures), enfin nous allons mettre le point sur la détection d'intrusions réseau.

1. Définition

La détection des intrusions est le processus de surveillance des événements qui se trouvent dans un système des ordinateurs ou du réseau et les analysant pour détecter les signes des intrusions, défini comme des tentatives pour compromettre la confidentialité, l'intégrité, la disponibilité ou éviter des mécanismes de sécurité de l'ordinateur ou du réseau. L'intrusion est causée par les attaques accédant au système via Internet, autorisée l'utilisateur du système qui essaye de gagner les privilèges supplémentaires pour lesquels ils n'ont pas été autorisés, et autorisé les utilisateurs qui abusent les privilèges donnés. Le système de détection des intrusions est un logiciel ou un matériel qui automatise des surveillances et les processus analysés. [9]

Les IDS protègent un système contre les attaques, les mauvaises utilisations et les compromis. Ils peuvent également surveiller l'activité du réseau, analyser les configurations du système et du réseau contre toute vulnérabilité, analyser l'intégrité de données et bien plus. Selon les méthodes de détection que vous choisissez de déployer, il existe plusieurs avantages directs et secondaires au fait d'utiliser un IDS. [W3]

Un IDS a quatre fonctions principales : l'analyse, la journalisation, la gestion et l'action.

- **Analyse:** Analyse des journaux du système pour identifier des intentions dans la masse de données recueillie par l'IDS. Il y a deux méthodes d'analyses : L'une basée sur les signatures d'attaques, et l'autre sur la détection d'anomalies.

- **Journalisation:** Enregistrement des évènements dans un fichier de log. Exemples d'évènements : arriver d'un paquet, tentative de connexion.
- **Gestion:** Les IDS doivent être administrés de manière permanente. On peut assimiler un IDS à une caméra de sécurité.
- **Action:** Alerter l'administrateur quand une attaque dangereuse est détectée.

2. Présentation d'un système de détection d'intrusions

Dans cette section nous allons décrire les systèmes de détection d'intrusions.

2.1. Architecture d'un IDS

Plusieurs schémas ont été proposés pour décrire les composants d'un système de détection d'intrusions. Parmi eux, nous avons retenu celui issu des travaux d'Intrusions Detection exchange formatWorking Group (IDWG) de l'Internet Engineering Task Force (IETF) comme base de départ, car il résulte d'un large consensus parmi les intervenants du domaine. [10]

L'objectif des travaux du groupe IDWG est la définition d'un standard de communication entre certains composants d'un système de détection d'intrusions. La figure illustre ce modèle et permet d'introduire un certain nombre de concepts :

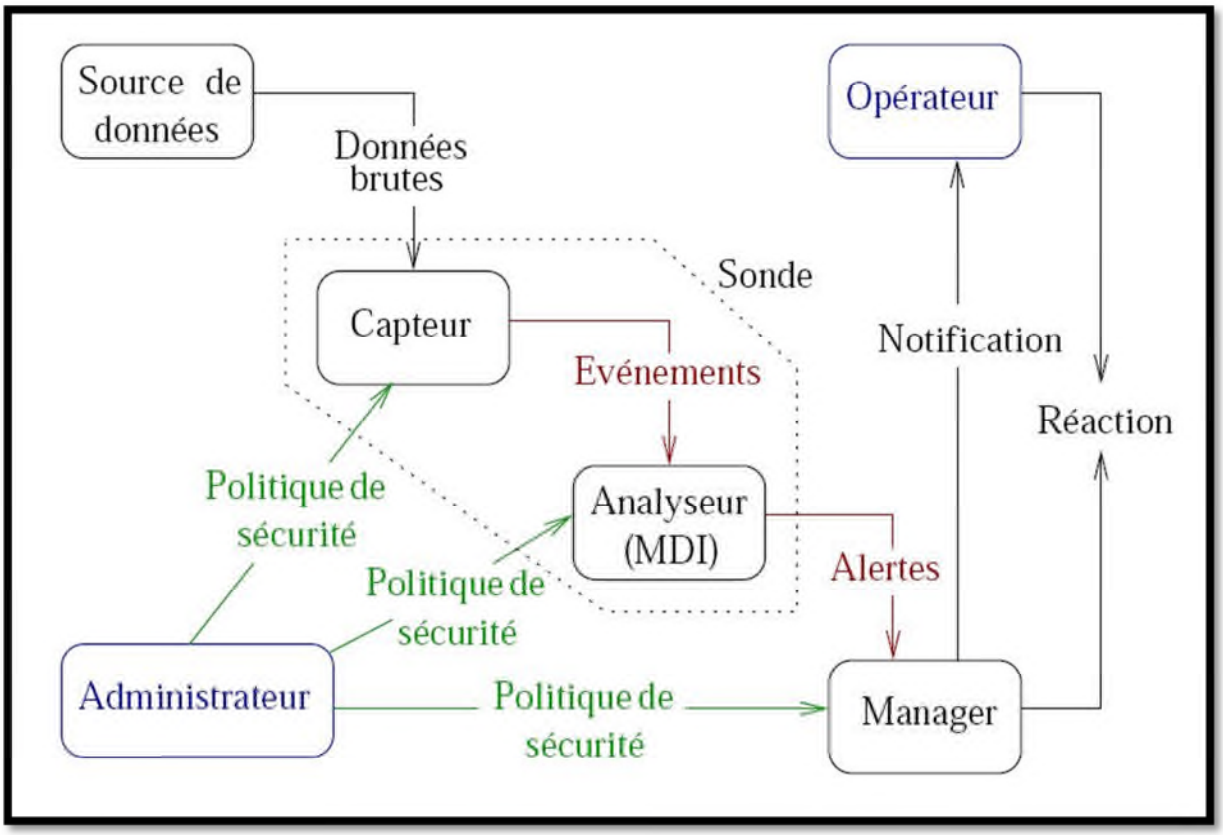


Figure 2.1 : Modèle générique de la détection d'intrusions proposé par l'IDWG

L'architecture IDWG d'un système de détection d'intrusions contient des capteurs qui envoient des événements à un analyseur. Les capteurs couplés avec un analyseur forment une sonde, cette dernière envoie des alertes vers un manager qui la notifie à un opérateur humain.

2.1.1. Les différents éléments de cette architecture

- **Administrateur :** Personne chargée de mettre en place la politique de sécurité, et par conséquent, de déployer et configurer les IDS.
- **Alerte :** message formaté émis par un analyseur s'il trouve des activités intrusives dans une source de données.
- **Analyseur :** c'est un outil logiciel qui met en œuvre l'approche choisie pour la détection (comportementale ou par scénarios), il génère des alertes lorsqu'il détecte une intrusion.
- **Capteur :** logiciel générant des événements en filtrant et formatant les données brutes provenant d'une source de données.
- **Evénement :** message formaté et renvoyé par un capteur. C'est l'unité élémentaire utilisée pour représenter une étape d'un scénario d'attaques connu.
- **Manager :** composant d'un IDS permettant à l'opérateur de configurer les différents éléments d'une sonde et de gérer les alertes reçues et éventuellement la réaction.
- **Notification :** la méthode par laquelle le manager d'IDS met au courant l'opérateur de l'occurrence d'alerte.
- **Opérateur :** personne chargée de l'utilisation du manager associé à l'IDS. Elle propose ou décide de la réaction à apporter en cas d'alerte. C'est, parfois, la même personne que l'administrateur.
- **Réaction :** mesures passive ou actives prises en réponse à la détection d'une attaque, pour la stopper ou pour corriger ses effets.
- **Sonde :** un ou des capteurs couplés avec un analyseur.
- **Source de données :** dispositif générant de l'information sur les activités des entités du système d'information.

Dans ce modèle qui représente le processus complet de la détection ainsi que l'acheminement des données au sein d'un IDS. L'administrateur configure les différents composants (capteur(s), analyseurs(s), manager(s)) selon une politique de sécurité bien définie. Les capteurs accèdent aux données brutes, les filtrent et les formatent pour ne renvoyer que les événements intéressants à un analyseur. Les analyseurs utilisent ces événements pour décider de la présence ou non d'une intrusion et envoient dans le cas échéant une alerte au manager, qui notifie l'opérateur humain, une réaction éventuelle peut être menée automatiquement par le manager ou manuellement par l'opérateur. [11]

2.2. Vocabulaire de la détection d'intrusions

La détection d'intrusions utilise un vocabulaire bien définis qui ne se trouve pas dans le modèle précédent et qui est comme suit :

- **Attaque ou intrusion :** action qui permet de violer la politique de sécurité.
- **Audit de sécurité :** c'est l'ensemble des mécanismes permettant la collecte d'informations sur les actions faites sur un système d'information.

- **Détection d'intrusions :** recherche de traces laissées par une intrusion dans les données produites par une source.
- **Faux positif :** alerte en l'absence d'attaque.
- **Faux négatif :** absence d'alerte en présence d'attaque.
- **Vulnérabilité :** faille de conception, d'implémentation ou de configuration d'un système logiciel ou matériel.
- **Log (trace d'audit) :** c'est un fichier système à analyser.
- **Exploit :** terme utilisé pour désigner un programme d'attaque.
- **Scénario :** suite constituée des étapes élémentaires d'une attaque.
- **Signature :** suites des étapes observables d'une attaque, utilisée par certains analyseurs pour rechercher dans les activités des entités, des traces de scénarios d'attaques connus.
- **Système de détection d'intrusions :** ensemble constitué d'un ou plusieurs capteurs, un ou plusieurs analyseurs et un ou plusieurs managers.
- **Corrélation :** c'est l'interprétation conceptuelle de plusieurs événements (alertes) visant à leur assigner une meilleure sémantique et à réduire la quantité globale d'événements (d'alertes).

2.3. Caractéristiques d'un système de détection d'intrusions [12]

Les caractéristiques suivantes sont souhaitables dans un IDS :

- Fonctionnement en permanence avec une supervision manuelle minimale.
- Etre tolérant aux pannes dans le sens où il doit récupérer après une défaillance ou une réinitialisation de la machine.
- Résister aux tentatives de corruption, c'est-à-dire, il doit pouvoir détecter s'il a subit lui-même une modification indésirable
- . utiliser un minimum de ressources pour implémenter une politique de sécurité spécifique d'un réseau.
- Etre facilement configurable pour implémenter une politique de sécurité spécifique d'un réseau.
- S'adapter au cours du temps aux changements du système surveillé et du comportement des utilisateurs.
- Etre difficile à tromper.

Comme la taille des réseaux a tendance à croitre, on peut ajouter les caractéristiques suivantes :

- Etre scalable.
- Etre robuste, c'est-à-dire l'arrêt d'un composant ne doit pas entrainer une défaillance totale.

2.4. Emplacement d'un système de détection d'intrusions

Il existe plusieurs endroits stratégiques où il convient de placer un IDS.

Le schéma suivant illustre un réseau local ainsi que les positions que peut y prendre un IDS :

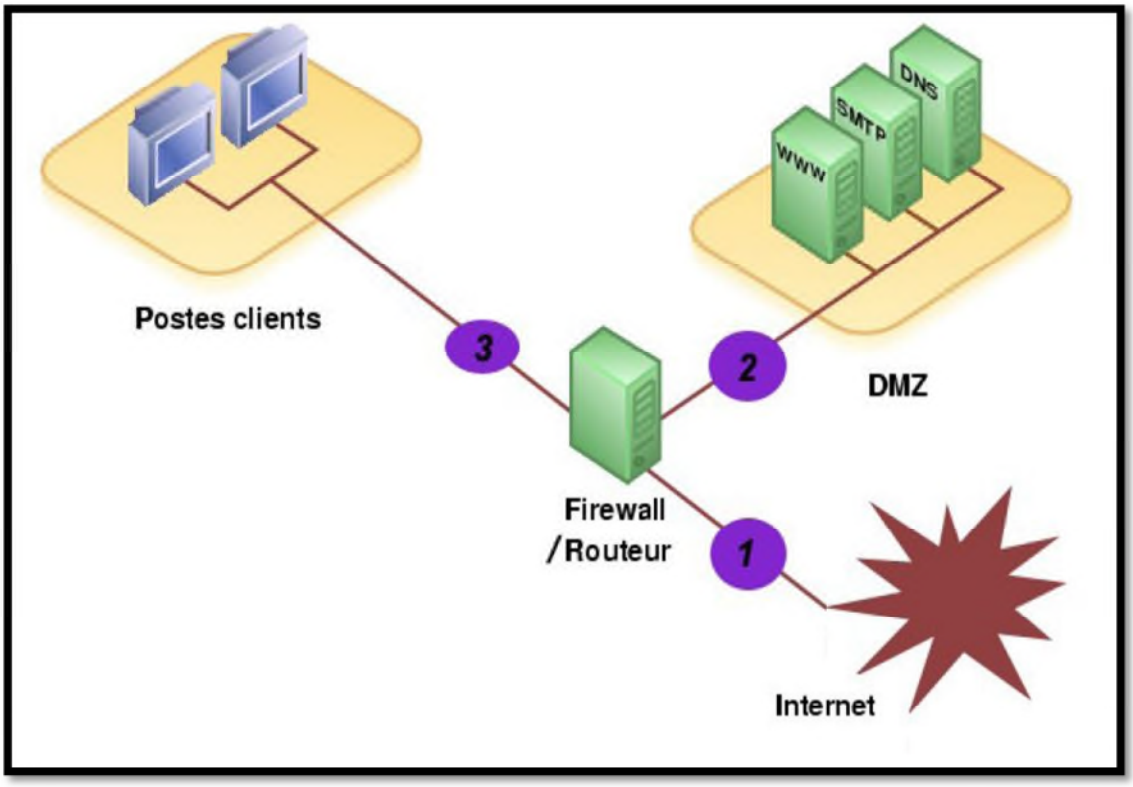


Figure 2.2 : Endroits typiques pour un système de détection d'intrusions

- **Position (1):** Sur cette position, l'IDS va pouvoir détecter l'ensemble des attaques frontales, provenant de l'extérieur, en amont du firewall. Ainsi, beaucoup (trop?) d'alertes seront remontées ce qui rendra les logs difficilement consultables.

- **Position (2):** Si l'IDS est placé sur la DMZ, il détectera les attaques qui n'ont pas été filtrées par le firewall et qui relèvent d'un certain niveau de compétence. Les logs seront ici plus clairs à consulter puisque les attaques bénins ne seront pas recensées.

- **Position (3):** L'IDS peut ici rendre compte des attaques internes, provenant du réseau local de l'entreprise. Il peut être judicieux d'en placer un à cet endroit étant donné le fait que 80% des attaques proviennent de l'intérieur. De plus, si des trojans ont contaminé le parc informatique (navigation peu méfiante sur internet) ils pourront être ici facilement identifiés pour être ensuite éradiqués.
-

2.5. Classification des systèmes de détection d'intrusions

Nous pouvons classifier les systèmes de détection d'intrusions selon cinq critères : [11]

- La source des données à analyser.
- Le lieu de l'analyse des données.
- La fréquence de l'analyse.
- Le comportement en cas d'attaque détectée.

- La méthode de détection utilisée.

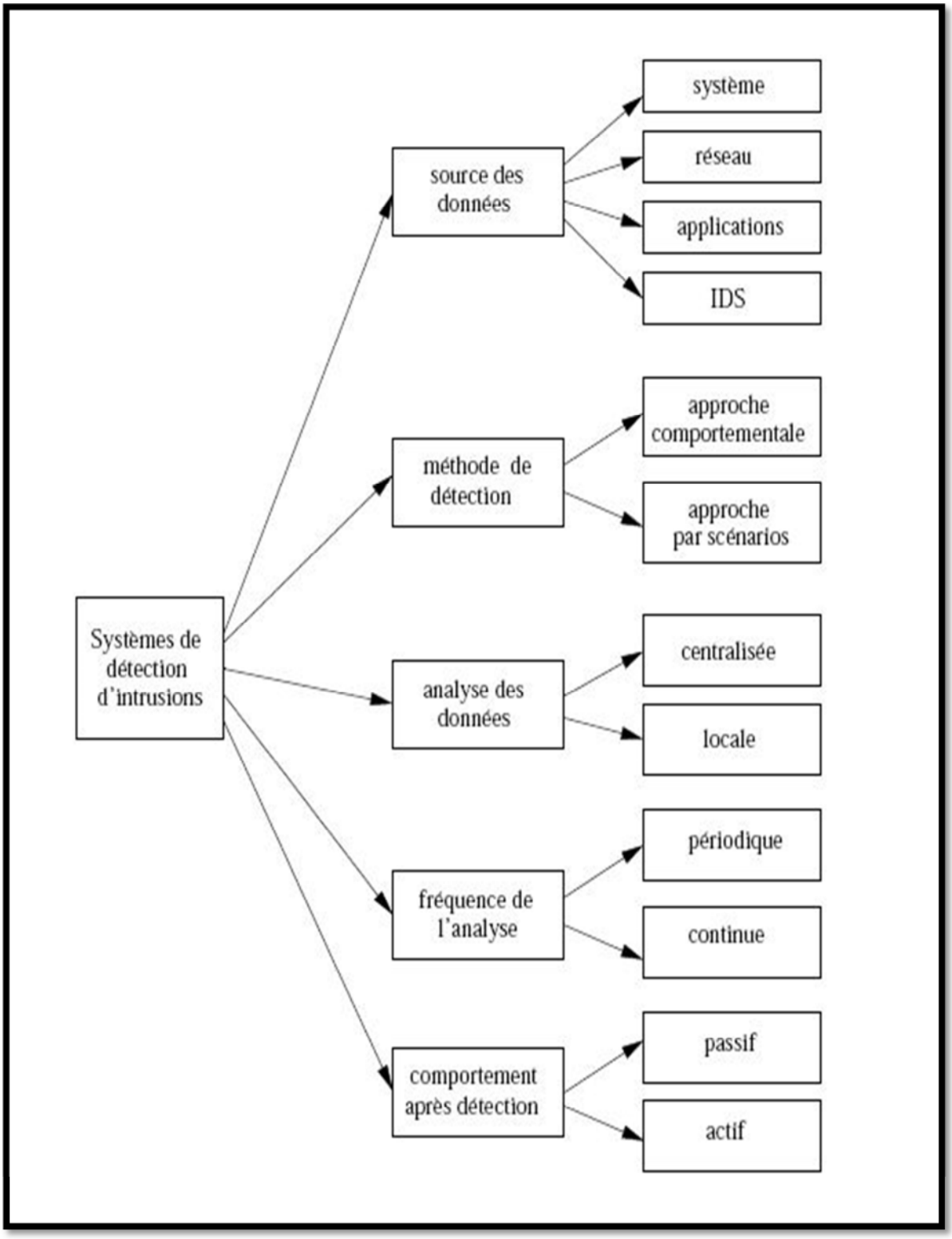


Figure 2.2 : Classification des systèmes de détection d'intrusions. [11]

2.5.1. Source des données à analyser

Les sources possibles de données à analyser sont une caractéristique essentielle des systèmes de détection d'intrusions puisque ces données constituent la matière première du processus de détection. Les données proviennent soit de logs générés par le système d'exploitation, soit de logs applicatifs, soit d'informations provenant du réseau, soit encore d'alertes générées par d'autres IDS. [11]

Source d'information système

Un système d'exploitation fournit généralement plusieurs sources d'information : [11]

- **Commandes systèmes :** presque tous les systèmes d'exploitation fournissent des commandes pour avoir un « instantané » de ce qui se passe.

- **Accounting :** l'accounting fournit de l'information sur l'usage des ressources partagées par les utilisateurs (temps processeur, mémoire, espace disque, débit réseau, applications lancées, ...).

- **Audit de sécurité :** tous les systèmes d'exploitation modernes proposent ce service pour fournir des événements système, les associer à des utilisateurs et assurer leur collecte dans un fichier d'audit. On peut donc potentiellement disposer d'informations sur tout ce que font (ou ont fait) les utilisateurs : accès en lecture à un fichier, exécution d'une application, etc.

Source d'information réseau

Des dispositifs matériels ou logiciels (snifer) permettent de capturer le trafic réseau. Cette source d'information est particulièrement adaptée lorsqu'il s'agit de rechercher les attaques en déni de service qui se passent au niveau réseau ou les tentatives de pénétration à distance. Le processus d'interception des paquets peut être rendu quasiment invisible pour l'attaquant car on peut utiliser une machine dédiée juste reliée à un brin du réseau, configurée pour ne répondre à aucune sollicitation extérieure et dont personne ne soupçonnera l'existence. Néanmoins, il est difficile de garantir l'origine réelle de l'attaque que l'on a détectée car il est facile de masquer son identité en modifiant les paquets réseau. [11]

Source d'information applicative

Les applications peuvent également constituer une source d'information pour les IDS. Les capteurs applicatifs sont de deux natures : [11]

- **Capteur interne :** le filtrage sur les activités de l'application est alors exécuté par le code de l'application.
- **Capteur externe :** le filtrage se fait à l'extérieur de l'application. Plusieurs méthodes sont utilisées : un processus externe peut filtrer les logs produits par l'application ou bien l'exécution de l'application peut être interceptée (au niveau de ses appels de librairies ou d'un proxy applicatif).

Prendre ses informations directement au niveau de l'application présente plusieurs avantages. Premièrement, les données interceptées ont réellement été reçues par l'application. Il est donc difficile d'introduire une désynchronisation entre ce que voit passer le capteur applicatif et ce que reçoit l'application contrairement à ce qu'il peut se passer avec les capteurs réseau. Ensuite, cette source d'information est généralement de plus haut niveau que les sources système et réseau. Cela permet donc de filtrer des événements qui ont une sémantique plus riche. Finalement, si l'on prend l'exemple d'une connexion web chiffrée par SSL, un capteur réseau ne verra passer que des données pseudo-aléatoires tandis qu'un capteur associé au serveur web pourra analyser le texte en clair de la requête. [11]

Source d'information basée IDS

Une autre source d'information, souvent de plus haut niveau que les précédentes, peut être exploitée. Il s'agit des alertes remontées par des analyseurs provenant d'un IDS. Chaque alerte synthétise déjà un ou plusieurs événements intéressants du point de vue de la sécurité. Elles peuvent être utilisées par un IDS pour déclencher une analyse plus fine à la suite d'une indication d'attaque potentielle. De surcroît, en corrélant plusieurs alertes, on peut parfois détecter une intrusion complexe de plus haut niveau. Il y aura alors génération d'une nouvelle alerte plus synthétique que l'on qualifie de méta-alerte. [11]

2.5.2. Localisation de l'analyse des données

On peut également faire une distinction entre les IDS en se basant sur la localisation réelle de l'analyse des données : [11]

- **Analyse centralisée :** certains IDS ont une architecture multi-capteurs (ou multisondes). Ils centralisent les événements (ou alertes) pour analyse au sein d'une seule machine. L'intérêt principal de cette architecture est de faciliter la corrélation entre événements puisqu'on dispose alors d'une vision globale. Par contre, la charge des calculs (effectués sur le système central) ainsi que la charge réseau (due à la collecte des événements ou des alertes) peuvent être lourdes et risquent de constituer un goulet d'étranglement.

- **Analyse locale :** si l'analyse du flot d'événements est effectuée au plus près de la source de données (généralement en local sur chaque machine disposant d'un capteur), on minimise le trafic réseau et chaque analyseur séparé dispose de la même puissance de calcul. En contrepartie, il est impossible de croiser des événements qui sont traités séparément et l'on risque de passer à côté de certaines attaques distribuées.

2.5.3. Fréquence de l'analyse

Une autre caractéristique des systèmes de détection d'intrusions est leur fréquence d'utilisation : [11]

- **Périodique :** certains systèmes de détection d'intrusions analysent périodiquement les fichiers d'audit à la recherche d'une éventuelle intrusion ou anomalie passée. Cela peut être suffisant dans des contextes peu sensibles (on fera alors une analyse journalière, par exemple).

- **Continue :** la plupart des systèmes de détection d'intrusions récents effectue leur analyse des fichiers d'audit ou des paquets réseau de manière continue afin de proposer une détection en quasi temps-réel. Cela est nécessaire dans des contextes sensibles (confidentialité) et/ou commerciaux (confidentialité, disponibilité). C'est toutefois un processus coûteux en temps de calcul car il faut analyser à la volée tout ce qui se passe sur le système.

2.5.4. Comportement après détection

Une autre façon de classer les systèmes de détection d'intrusions consiste à les classer par type de réaction lorsqu'une attaque est détectée : [11]

- **passive :** la plupart des systèmes de détection d'intrusions n'apportent qu'une réponse passive à l'intrusion. Lorsqu'une attaque est détectés, ils génèrent une alarme et notifient l'administrateur système par e-mail, message dans une console, voire même par beeper. C'est alors lui qui devra prendre les mesures qui s'imposent
- **active :** d'autres systèmes de détection d'intrusions peuvent, en plus de la notification à l'opérateur, prendre automatiquement des mesures pour stopper l'attaque en cours. Par exemple, ils peuvent couper les connexions suspectes ou même, pour une attaque externe, reconfigurer le pare-feu pour qu'il refuse tout ce qui vient du site incriminé. Des outils tels que RealSecure ou NetProwler proposent ce type de réaction. Toutefois, il apparait que ce type de fonctionnalité automatique est potentiellement dangereux car il peut mener à des dénis de service provoqués par l'IDS. Un attaquant déterminé peut, par exemple, tromper l'IDS en usurpant des adresses du réseau local qui seront alors considérées comme la source de l'attaque par l'IDS. Il est préférable de proposer une réaction facultative à un opérateur humain (qui prend la décision finale).

2.5.5. Méthode de détection

Deux approches de détection ont été proposées : [11]

- **L'approche comportementale** : cette approche se base sur l'hypothèse selon laquelle nous pouvons définir un comportement normal de l'utilisateur et que toute déviation par rapport à celui-ci est potentiellement suspect.
- **L'approche par signature** : elle s'appuie sur un modèle constitué des sections interdites dans le système d'informatique, ce modèle s'appuie sur la connaissance des techniques employées par les attaquants : on tire des scénarios d'attaque et on recherche dans les traces d'audit leur éventuelle survenue.

2.5.5.1. L'approche comportementale

Les détecteurs d'intrusions comportementaux reposent sur la création d'un modèle de référence représentant le comportement de l'entité surveillée en situation de fonctionnement normal. Ce modèle est ensuite utilisé durant la phase de détection afin de pouvoir mettre en évidence d'éventuelles déviations comportementales. Pour cela, le comportement de l'entité surveillée est comparé à son modèle de référence. Une alerte est levée lorsqu'une déviation trop importante (notion de seuil) vis-à-vis de ce modèle de comportement normal est détectée.

Le principe de cette approche est de considérer tout comportement n'appartenant pas au modèle de comportement normal comme une anomalie symptomatique d'une intrusion ou d'une tentative d'intrusion. [13]

On peut distinguer deux catégories de profils :

2.5.5.1.1. Profils construits par apprentissage

Parmi les méthodes proposées pour construire les profils par apprentissage, les plus marquantes sont les suivantes : [11]

- **méthode statistique :** le profil est calculé à partir de variables considérées comme aléatoires et échantillonnées à intervalles réguliers. Ces variables peuvent être le temps processeur utilisé, la durée et l'heure des connexions, etc. Un modèle statistique est alors utilisé pour construire la distribution de chaque variable et pour mesurer, au travers d'une grandeur synthétique, le taux de déviation entre un comportement courant et le comportement passé.

- **système expert :** ici, c'est une base de règles qui décrit statistiquement le profil de l'utilisateur au vu de ses précédentes activités. Son comportement courant est comparé aux règles, à la recherche d'une anomalie. La base de règles est rafraîchie régulièrement.

- **réseaux de neurones :** la technique consiste à apprendre à un réseau de neurones le comportement de l'entité à surveiller. Par la suite, lorsqu'on lui fournira en entrée les actions courantes effectuées par l'entité, il devra décider de leur normalité.

- **analyse de signatures :** Il s'agit de construire un modèle de comportement normal des services réseaux. Le modèle consiste en un ensemble de courtes séquences d'appels système représentatifs de l'exécution normale du service considéré. Des séquences d'appels étrangères à cet ensemble sont alors considérées comme l'exploitation potentielle d'une faille du service.

Pour toutes ces méthodes, le comportement de référence utilisé pour l'apprentissage étant rarement exhaustif, on s'expose à des risques de fausses alarmes (faux positifs). De plus, si des attaques ont été commises durant cette phase, elles seront considérées comme normales (risque de faux négatifs).

2.5.5.1.2. Profils spécifiant une politique de sécurité (policy-based)

Pour les IDS dits policy-based, il n'y a pas de phase d'apprentissage. Leur comportement de référence est spécifié par une politique de sécurité : la détection d'une intrusion intervient chaque fois que la politique est violée. Le profil est ici une politique de sécurité qui décrit la suite des appels systèmes licites d'une application. [11]

L'approche comportementale possède un certain nombre d'avantages et d'inconvénients :

Les avantages :

- l'analyse comportementale n'exige pas des connaissances préalables sur les attaques.
- Elle permet la détection de la mauvaise utilisation des privilèges.
- Elle permet de produire des informations qui peuvent être employées pour définir des signatures pour l'analyse basée connaissance.

Les inconvénients

- Les approches comportementales produisent un taux élevé des alarmes type faux positif en raison des comportements imprévisibles des utilisateurs et des réseaux.
- Ces approches nécessitent des phases d'apprentissage pour caractériser les profils de comportement normaux.
- Les alarmes génériques par cette approche ne sont pas significatives.

2.5.5.2. L'approche par scénarios

On construit des scénarios d'attaque en spécifiant ce qui est caractéristique de l'attaque et qui doit être observé dans les traces d'audit. L'analyse des traces d'audit se fait à la recherche de ces scénarios. Les méthodes proposées à ce jour sont les suivantes : [11]

- **- système expert :** le système expert comporte une base de règles qui décrit les attaques. Les événements d'audit sont traduits en des faits qui sont interprétables par le système expert. Son moteur d'inférence décide alors si une attaque répertoriée s'est ou non produite.
- **analyse de signatures :** il s'agit là de la méthode la plus en vue actuellement. Des signatures d'attaques sont fournies à des niveaux sémantiques divers selon les outils (de la suite d'appels système aux commandes passées par l'utilisateur en passant par les paquets réseau). Divers algorithmes sont utilisés pour localiser ces signatures connues dans les traces d'audit. Ces signatures sont toujours exprimées sous une forme proche des traces d'audit. Si l'on prend l'exemple des NIDS, les algorithmes de recherche de motifs utilisés permettent d'obtenir de bonnes performances en vitesse de traitement mais génèrent de nombreuses fausses alertes.
- **automates à états finis :** plusieurs IDS utilisent des automates à états finis pour coder le scénario de reconnaissance de l'attaque. Cela permet d'exprimer des signatures complexes et comportant plusieurs étapes. On passe d'un état initial sûr à un état final attaqué via des états intermédiaires. Chaque transition entre états est déclenchée par des conditions sur les événements remontés par les capteurs.

L'approche par scénario possède un certain nombre d'avantages et d'inconvénients :

Les avantages

- l'analyse basée connaissance est très efficace pour la détection d'attaque avec un taux très bas des alarmes de type faux positif.
- Les alarmes générées sont significatives.

Les inconvénients

- Cette analyse basée connaissance permet seulement la détection des attaques qui sont connues au préalable. Donc, la base de connaissances doit être constamment mise à jour avec les signatures des nouvelles attaques.
- Le risque que l'attaquant peut influencer sur la détection après la reconnaissance des signatures.

2.6. Les différentes sortes d'IDS

Les différents IDS se caractérisent par leur domaine de surveillance. Celui-ci peut se situer au niveau d'un réseau d'entreprise, d'une machine hôte, d'une application… Nous allons tout d'abord étudier la détection d'intrusion basée sur l'hôte, puis basée sur une application, avant de nous intéresser aux IDS réseaux, NIDS et NNIDS (Network IDS et Node Network IDS). [14]

2.6.1. La détection d'intrusions basée sur l'hôte [14]

Les systèmes de détection d'intrusions basés sur l'hôte ou HIDS (Host IDS) analysent exclusivement l'information concernant cet hôte. Comme ils n'ont pas à contrôler le trafic du réseau mais "seulement" les activités d'un hôte ils se montrent habituellement plus précis sur les types d'attaques subies.

De plus, l'impact sur la machine concernée est sensible immédiatement, par exemple dans le cas d'une attaque réussie par un utilisateur. Ces IDS utilisent deux types de sources pour fournir une information sur l'activité de la machine : les logs et les traces d'audit du système d'exploitation. Chacun a ses avantages : les traces d'audit sont plus précises et détaillées et fournissent une meilleure information alors que les logs qui ne fournissent que l'information essentielle sont plus petits. Ces derniers peuvent être mieux contrôlés et analysés en raison de leur taille, mais certaines attaques peuvent passer inaperçues, alors qu'elles sont détectables par une analyse des traces d'audit.

Ce type d'IDS possède un certain nombre d'avantages : il est possible de constater immédiatement l'impact d'une attaque et donc de mieux réagir. Grâce à la quantité des informations étudiées, il est possible d'observer les activités se déroulant sur l'hôte avec précision et d'optimiser le système en fonction des activités observées.

De plus, les HIDS sont extrêmement complémentaires des NIDS. En effet, ils permettent de détecter plus facilement les attaques de type "Cheval de Troie", alors que ce type d'attaque est difficilement détectable par un NIDS. Les HIDS permettent également de détecter des attaques impossibles à détecter avec un NIDS, car elles font partie de trafic crypté.

Néanmoins, ce type d'IDS possède également ses faiblesses, qui proviennent de ses qualités : du fait de la grande quantité de données générées, ce type d'IDS est très sensible aux attaques de type DoS, qui peuvent faire exploser la taille des fichiers de logs.

Un autre inconvénient tient justement à la taille des fichiers de rapport d'alertes à examiner, qui est très contraignante pour le responsable sécurité. La taille des fichiers peut en effet atteindre plusieurs Mégaoctets. Du fait de cette quantité de données à traiter, ils sont assez gourmands en CPU et peuvent parfois altérer les performances de la machine hôte. Enfin, ils ont moins de facilité à détecter les attaques de type hôte que les IDS réseaux.

Les HIDS sont en général placés sur des machines sensibles, susceptibles de subir des attaques et possédantes des données sensibles pour l'entreprise. Les serveurs, web et applicatifs, peuvent notamment être protégés par un HIDS.

Pour finir, voici quelques HIDS connus:

- **Tripwire.**
- **WATCH.**
- **DragonSquire.**
- **Tiger.**
- **Security Manager.**
- **Etc**.

2.6.2. Détection d'Intrusions basée sur une application [14]

Les IDS basés sur les applications sont un sous-groupe des IDS hôtes. Ils contrôlent l'interaction entre un utilisateur et un programme en ajoutant des fichiers de log afin de fournir de plus amples informations sur les activités d'une application particulière. Puisque vous opérez entre un utilisateur et un programme, il est facile de filtrer tout comportement notable. Un ABIDS se situe au niveau de la communication entre un utilisateur et l'application surveillée.

L'avantage de cet IDS est qu'il lui est possible de détecter et d'empêcher des commandes particulières dont l'utilisateur pourrait se servir avec le programme et de surveiller chaque transaction entre l'utilisateur et l'application. De plus, les données sont décodées dans un contexte connu, leur analyse est donc plus fine et précise.

Par contre, du fait que cet IDS n'agit pas au niveau du noyau, la sécurité assurée est plus faible, notamment en ce qui concerne les attaques de type "Cheval de Troie". De plus, les fichiers de log générés par ce type d'IDS sont des cibles faciles pour les attaquants et ne sont pas aussi sûrs, par exemple, que les traces d'audit du système.

Ce type d'IDS est utile pour surveiller l'activité d'une application très sensible, mais son utilisation s'effectue en général en association avec un HIDS. Il faudra dans ce cas contrôler

le taux d'utilisation CPU des IDS afin de ne pas compromettre les performances de la machine.

2.6.3. La Détection d'Intrusions Réseau [14]

Le rôle essentiel d'un IDS réseau est l'analyse et l'interprétation des paquets circulant sur ce réseau. L'implantation d'un NIDS sur un réseau se fait de la façon suivante : des capteurs sont placés aux endroits stratégiques du réseau et génèrent des alertes s'ils détectent une attaque. Ces alertes sont envoyées à une console sécurisée, qui les analyse et les traites éventuellement. Cette console est généralement située sur un réseau isolé, qui relie uniquement les capteurs et la console.

Les avantages des NIDS sont les suivants : les capteurs peuvent être bien sécurisés puisqu'ils se contentent d'observer le trafic et permettent donc une surveillance discrète du réseau, les attaques de type scans sont facilement détectées, et il est possible de filtrer le trafic.

Les NIDS sont très utilisés et remplissent un rôle indispensable, mais ils présentent néanmoins de nombreuses faiblesses. En effet, la probabilité de faux négatifs (attaques non détectées comme telles) est élevée et il est difficile de contrôler le réseau entier. De plus, ils doivent principalement fonctionner de manière cryptée d'où une complication de l'analyse des paquets. Pour finir, à l'opposé des IDS basés sur l'hôte, ils ne voient pas les impacts d'une attaque

Voici quelques exemples de NIDS :

- **NetRanger.**
- **Dragon.**
- **NFR.**
- **Snort.**
- **ISSRealSecure.**

2.6.4. Les IDS hybrids [15]

Les IDS hybrides rassemblent les caractéristiques des NIDS et HIDS. Ils permettent, en un seul outil, de surveiller le réseau et les terminaux. Les sones sont placés en des points stratégiques, et agissent comme NIDS et/ou HIDS suivant leurs emplacements. Toutes ces sondes remontent alors les alertes à une machine qui va centraliser le tout, et lier les informations d'origines multiples. Ainsi, on comprend que les IDS hybrides sont basés sur une architecture distribuée, ou chaque composant unifie son format d'envoi cela permet de communiquer et d'extraire des alertes plus pertinentes.

Les avantages des IDS hybrides sont multiples :

- Moins de faux positifs
- Meilleure corrélation (la corrélation permet de générer de nouvelles alertes à partir de celle existantes).
- Possibilité de réaction sur les analyseurs.

3. Les principales tâches d'un IDS

Un IDS permet de repérer des anomalies dans le trafic réseau comme suit :

- Détecter les tentatives de découvertes du réseau.
- Détecter dans certains cas, si l'attaque a réussi ou non.
- Détecter le Dénis de Service.
- Détecter le niveau d'infection du système informatique et les zones réseaux touchées.
- Repérer les machines infectées.
- Alerter de façon centrale pour toutes les attaques.
- Réagir aux attaques et corriger les problèmes éventuels.
- Etc.

Si on compare les HIDS et NIDS, les HIDS présente un avantage considérable par rapport à un NIDS dans le cas où le trafic est crypté. En effet, un NIDS n'a pas connaissance des clés de cryptage et ne peut appliquer ses algorithmes de détection au niveau des données chiffrées. La détection est effectuée à l'extrémité de la chaine de communication, une fois le flux est décrypté. Ceci est réalisé en mettant en œuvre un agent HIDS directement sur le serveur cible. Les flux chiffrés sont ainsi décodés par la cible et transmis ensuite au monteur d'analyse de l'NIDS.

4. les limites d'un IDS

Parmi les faiblesses des IDS on trouve : [16]

- Nombreux faux positifs.
- Configuration complexe et longue.
  - Nombreux faux positifs après configuration.
- Pas de connaissance de la plate-forme.
  - De ses vulnérabilités.
  - Du contexte métier.
- Les attaques applicatives sont difficilement détectables.
  - Injection SQL.
  - Exploitation de CGI mal conçus.
- Des évènements difficilement détectables.
  - Scans lents / distribués.
  - Canaux cachés / tunnels.
- Pollution des IDS.
  - Consommation des ressources de l'IDS.
  - Perte de paquets.
  - Déni de service contre l'IDS / l'opérateur.
  - Une attaque réelle peut passer inaperçue.
- Attaque contre l'IDS lui-même.
- Ils ne peuvent pas compenser les trous de sécurité dans les protocoles réseaux.
- Ils ne peuvent pas compenser des manques significatifs dans votre stratégie de sécurité, votre politique de sécurité ou votre architecture de sécurité.

5. Quelques outils

Il existe plusieurs IDS sur le marché, parmi eux :

- **ISS RealSecure**

Internet Security Systems (ISS) fournit ISS RealSecure IDS, une plate-forme de détection d'intrusions intégrée. ISS RealSecure IDS utilise une approche basée sur des normes pour comparer les entrées de trafic réseau et journaux d'accueil des méthodes connues et probables des attaquants. ISS RealSecure IDS intègre avec de nombreuses applications de réseau et de gestion des systèmes. [W4]

RealSecure combine en un seul agent trois fonctionnalités essentielles :

- Un moteur de détection d'intrusions.
- Un firewall personnel.
- Un module de contrôle d'applications et de communications.

- **Enterasys DRAGON**

Edité par Enterasys Networks, c'est un système de détection des intrusions considérée comme des leaders du marché du fait ses performances, ses facultés d'adaptation à tout type d'environnement et ses capacité d'analyse.

Les solutions Dragon sont constituées de sondes NIDS (Network Sensor), d'agents HIDS (Host Sensor) et d'un système de management qui assure les fonctions d'exploitation des événements de la suite Dragon.

Le Network Sensor est un NIDS disponible en version logicielle ou en boitier dédié. Depuis la version 6, Enterasys décline les Appliances et les logiciels en trois versions selon la bande passante à analyser. Les versions matérielles sont adossées à leur équivalent logiciel.

Le Host Sensor est une agent HIDS qui détecte les attaques contre le système sur lequel il est installé en contrôlant les journaux systèmes et d'audit ainsi qu'en utilisant des mécanismes d'analyse de signatures.

Dragon détecte les intrusions sur l'ensemble de l'infrastructure informatique ou qu'elles se produisent et permet d'avoir ainsi une visibilité globale sur le système d'information. Cela permet notamment d'optimiser les ressources humaines nécessaires à l'analyse des journaux issus des différents firewalls ou serveur Web en fédérant tous ces journaux au niveau d'une console Dragon unique qui analysera automatiquement les données afférentes. [17]

- **SNORT**

SNORT est un IDS particulièrement répondu car fourni en open source. Il est donc gratuit et facile à se procurer, outre sa gratuité, son avantage est qu'il dispose d'une très grosse base de signatures réalisée par la communauté des utilisateurs.

C'est également le gage d'obtenir rapidement des mises à jour de la base dès qu'une nouvelle menace est signalé. Il a été conçu à l'origine pour le système linux mais il a également été porté nous Windows. On trouve dans le commerce plusieurs livres dédiés à l'installation et à l'utilisation de SNORT.

SNORT est généralement utilisé en conjonction avec un autre logiciel open source nommé BASE qui est la console de gestion et d'analyse.

Pour ce qui est des points négatifs, on peut néanmoins considérer que SNORT est moins puissant en termes de moteur d'analyse que des solutions commerciales telles que celle d'ISS.

Par ailleurs, bien que la base de signatures soit étendue, elle nécessite un travail constant de la part de l'administrateur qui doit les télécharger manuellement. Il n'y a pas de procédure de mise à jour automatiques. [18]

Conclusions

Ce chapitre nous a permis de découvrir les systèmes de détection d'intrusions leurs fonctionnement et leur capacités. La plupart des IDS sont fiables, ce qui explique qu'ils sont souvent intégrés dans les solutions de sécurité. Les avantages qu'ils présentent face aux autres outils de sécurités les favorisent, mais d'un autre côté cela n'empêche pas que les meilleurs IDS présentent aussi des lacunes et quelques inconvénients. Nous comprenons donc bien qu'ils sont nécessaires mais ne peuvent pas se passer de l'utilisation d'autres outils de sécurité visant à combler leurs défauts. Nous allons voir dans le chapitre suivant comment réussir une bonne configuration de ces derniers afin de mieux sécurisé le réseau.

Chapitre 3 : la mise en place d’un IDS (SNORT)

.

Introduction

Outre la mise en place d'un pare-feu et d'un système d'authentification, il est de nos jours nécessaire de mettre en place un système de détection d'intrusions (SNORT).

SNORT est un logiciel open source écrit par Martin Roesch, disponible sous licence GNU, son code source est accessible et modifiable à partir de l'URL : «http://www.snort.org».

Dans ce qui suit, nous allons commencer par donner une présentation générale de SNORT, en suite nous allons présenter leur manipulation : installation, configuration et fonctionnalités. Enfin, nous allons terminer par donner une conclusion et des perspectives pour ce travail.

1. Présentation général de Snort

Snort est un NIDS écrit par Martin Roesch, disponible sous licence GNU, son code source est accessible et modifiable à partir de l'URL : « http://www.snort.org ».

Snort a la capacité d'effectuer l'analyse du trafic en temps réel et effectuer l'analyse de protocole, la recherche de contenu. Il utilise une sonde pour détecter les attaques, le débordement système, les scans de port etc.

Snort peut être configuré pour fonctionner en trois modes :

Le mode sniffer : dans ce mode, SNORT lit les paquets circulant sur le réseau et les affiche d'une façon continue sur l'écran.
Le mode « packet logger » : dans ce mode SNORT journalise le trafic réseau dans des répertoires sur le disque.
Le mode détecteur d'intrusions réseau (NIDS) : dans ce mode, SNORT analyse le trafic du réseau, compare ce trafic à des règles déjà définies par l'utilisateur et établi des actions à exécuter.

1.2. Positionnement de Snort dans le réseau

L'emplacement physique de la sonde SNORT sur le réseau a un impact considérable sur son efficacité.

Dans le cas d'une architecture classique, composée d'un Firewall et d'une DMZ, trois positions sont généralement envisageables :

Avant le Firewall ou le routeur filtrant : dans cette position, la sonde occupe une place de premier choix dans la détection des attaques de sources extérieures visant l'entreprise. SNORT pourra alors analyser le trafic qui sera éventuellement bloqué par le Firewall. Les deux inconvénients de cette position du NIDS sont:

- le risque engendré par un trafic très important qui pourrait entraîner une perte de fiabilité

- étant situé hors du domaine de protection du firewall, le NIDS est alors exposé à d'éventuelles attaques pouvant le rendre inefficace.

Sur la DMZ : dans cette position, la sonde peut détecter tout le trafic filtré par le Firewall et qui a atteint la zone DMZ. Cette position de la sonde permet de surveiller les attaques dirigées vers les différents serveurs de l'entreprise accessible de l'extérieur.
Sur le réseau interne : le positionnement du NIDS à cet endroit nous permet d'observer les tentatives d'intrusions parvenues à l'intérieur du réseau d'entreprise ainsi que les tentatives d'attaques à partir de l'intérieur. Dans le cas d'entreprises utilisant largement l'outil informatique pour la gestion de leur activités ou de réseaux fournissant un accès à des personnes peu soucieuses de la sécurité (réseaux d'écoles et d'universités), cette position peut revêtir un intérêt primordial.

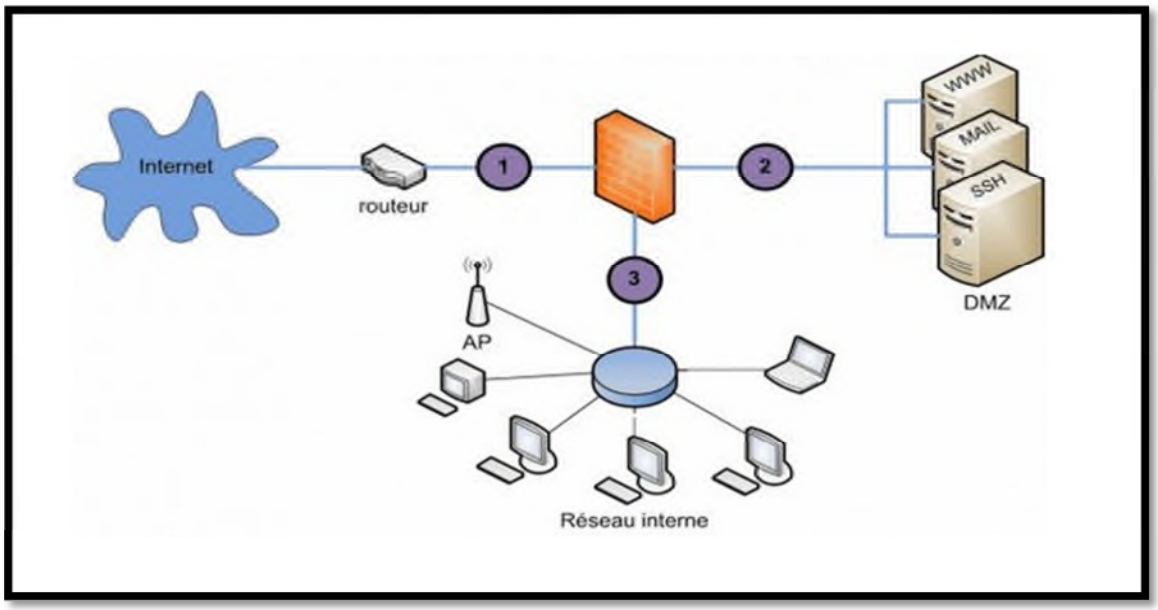


Figure3.1 : Différentes positions possible de Snort dans un réseau informatiques

1.3. Architecture de Snort

L'architecture de SNORT est modulaire et est composée de :

Décodeur de paquet : en anglais (Packet Decoder) il capture les paquets de données des interfaces réseaux, les prépare afin d'être prétraitées ou envoyées au moteur de détection.
Pré processeur : en anglais (Pre processor**)** ce sont des composants utilisés avec SNORT afin d'améliorer les possibilités d'analyse, et de recomposition du trafic capturé. Ils reçoivent les paquets, les retraitent et les envoient au moteur de détection.
Moteur de détection : en anglais (Detection Engine) c'est le composant le plus important de SNORT. Son rôle consiste à détecter les éventuelles intrusions qui

existent dans un paquet. Pour se faire, le moteur de recherche se base sur les règles de SNORT. En effet, ce moteur consulte ces règles et les compare une à une avec le paquet de données. S'il y a conformité, le détecteur l'enregistre dans le fichier log et/ou génère une alerte. Sinon le paquet est laissé tomber.

Système d'alerte et d'enregistrement des logs : en anglais (Logging and Alerting System) il permet de générer les alertes et les messages log suivant ce que le moteur de détection a trouvé dans le paquet analysé.

Output modules : en anglais (plugins) permet de traiter l'intrusion générée par le système d'alertes et de notation de plusieurs manières : envoie vers un fichier log, génère un message d'alerte vers un serveur syslog, ou stocke cette intrusion dans une base de données comme MySQL ou Oracle.

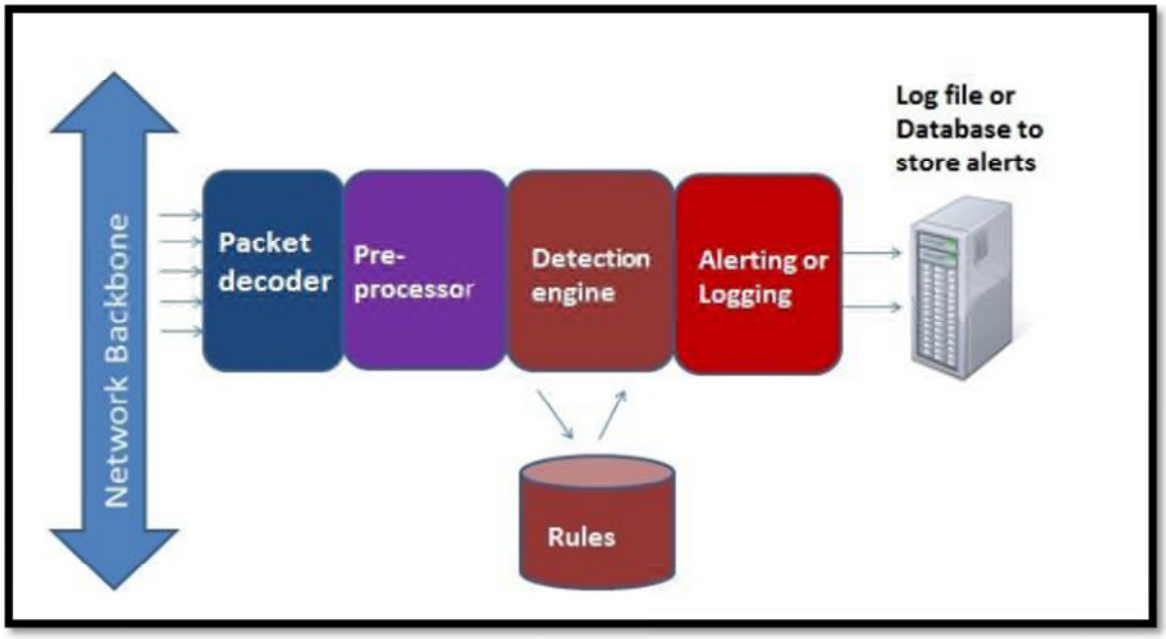


Figure 3.2 : Architecture de SNORT

1.4. Environnement

Pour une sécurité optimale, nous avons préféré de travailler dans un environnement Linux car il nous fournit un espace de travail unique et nous assure une fiabilité de résultats

Ubuntu est une distribution Gnu/Linux récente, développée par la société Canonical Ltd, fondée par Mark Shuttleworth, basée sur une Débian, il est constitué de logiciels libres, est disponible gratuitement y compris pour les entreprises.

1.5. Paramétrage de SNORT [W6]

1.5.1. Pré-processeurs

Les préprocesseurs permettent d'étendre les fonctionnalités de SNORTT. Ils sont exécutés avant le lancement du moteur de détection et après le décodage du paquet IP. Le paquet IP peut être modifié ou analysé de plusieurs manières en utilisant le mécanisme de préprocesseur.

Les préprocesseurs sont chargés et configurés avec le mot-clé preprocessor. Le format de la directive preprocesseur dans les règles de SNORT est : **preprocessor <nom> <options>.**

1.5.1.1. Mainfrag

Le préprocesseur mainfrag examine les paquets fragmentés pour une valeur limite de taille spécifiée. Quand les paquets sont fragmentés, c'est généralement causé par des routeurs entre la source et la destination. Généralement, il n'y a pas d'équipement réseau commercial qui fragmente les paquets en tailles plus petites que 512 octets, donc nous pouvons utiliser ce fait pour activer la surveillance du réseau pour de petits fragments qui sont généralement indicatifs de quelqu'un essayant de cacher son trafic derrière la fragmentation.

Format : **minfrag: <valeur limite>**
Exemple : **preprocessor mainfrags : 128**

1.5.1.2. HTTP Decode

HTTP Decode est utilisé pour traiter les chaînes HTTP URI et convertir leurs données en chaînes ASCII non obscurcies. Ceci est fait pour vaincre les scanners évasifs d'URL web et les attaques hostiles qui pourraient autrement échapper aux chaînes d'analyse de contenu utilisées pour examiner dans le trafic HTTP des activités suspicieuses. Le module préprocesseur prend des numéros de ports HTTP (séparés par des espaces) pour être normaliser en arguments (typiquement 80 et 8080).
Format : **http_decode: <liste de ports>**
Exemple : **preprocessor http_decode: 80 8080**

1.5.1.3. Détecteur de balayage du port (Portscan Detector)

Ce que le préprocesseur de scans de ports de Snort fait :

Journaliser le début et la fin de scans de ports depuis une unique IP source vers la facilité de journalisation standard.

Si un fichier journal est spécifié, il journalise les IP destinations et les ports scannés ainsi que le type de scan.

Un scan de ports est défini comme des tentatives de connexion TCP vers plus de P ports en T secondes ou des paquets UDP envoyés à plus de P ports en T secondes. Les ports

peuvent être répartis au travers de tout nombre d'adresses IP destinations, et peuvent tous être le même port si ils sont répartis au travers de multiples IP. Cette version fait les scans de ports unique-unique et unique-plusieurs. La prochaine version majeure fera les scans de ports distribués (multiple-unique ou multiple-multiple). Un scan de ports est également défini comme un unique paquet de "scan furtif", tel que NULL, FIN, SYNFIN, XMAS, etc. Ceci signifie que dans scan-lib dans la distribution standard de snort vous devez commenter la section pour les paquets de scans furtifs. Le bénéfice est que avec le module de scans de ports ces alertes ne devraient apparaître qu'une fois, plutôt qu'une fois pour chaque paquet. Si vous utilisez la fonctionnalité de journalisation externe vous pouvez voir la technique et le type dans le fichier journal.

Format : **portscan: <réseau à surveiller> <nombre de ports> <période de détection> <répertoire/fichier>**
Exemple : **preprocessor portscan: 192.168.1.0/24 5 7 /var/log/portscan.log**

1.5.1.4. Port scan Ignorer hosts

Un autre module de Patrick Mullen qui modifie le fonctionnement du système de détection de scans de ports. Si vous avez des serveurs qui tendent à activer le détecteur de scans de ports (tels que des serveurs NTP, NFS et DNS), vous pouvez dire au module d'ignorer les SYN TCP et les scans de ports UDP de certains systèmes. Les arguments de ce module sont une liste de blocs IP/CIDR à ignorer.

Format : **portscan-ignorehosts: <liste de systèmes>**
Exemple : **preprocessor portscan-ignorehosts: 192.168.1.5/32 192.168.3.0/24**

1.5.1.5. Défragmentation (Defrag)

Le module defrag (de Dragos Ruiu) permet à Snort d'effectuer de la défragmentation IP, rendant plus difficile pour les pirates de contourner simplement les capacités de détection du système. Il est très simple dans son usage, requérant simplement l'addition d'une directive préprocesseur au fichier de configuration sans aucun argument. Ce module ***supercedes*** généralement la fonctionnalité du module minfrag (c.-à-d. vous n'avez pas besoin d'utiliser minfrag si vous utilisez defrag).

Format : **defrag**
Exemple : **preprocessor defrag**

1.5.1.6. Stream

Ce module est toujours en BETA test, utilisez avec précautions. Le plugin Stream fournit la fonctionnalité de réassemblage de sessions TCP à Snort. Les sessions TCP sur les ports configurés avec de petits segments seront réassemblées en un flux de données que Snort peut évaluer proprement pour des activités suspicieuses. Ce plugin prend plusieurs arguments : Timeout - le temps maximal en secondes pendant lequel une session sera gardée vivante si elle n'a pas vu un paquet pour elle.

port - un port serveur à surveiller. Nous ne voulons pas surveiller toutes les sessions tcp (le voulons nous ?)

maxbytes - le nombre maximal d'octets dans notre paquet reconstruit

Format : **stream: timeout <temps maximal>, ports <ports>, maxbytes <maximum d'octets>**
Exemple: **preprocessor stream: timeout 5, ports 21 23 80 8080, maxbytes 16384**

1.5.2. Les plugins de sortie

Les modules de sortie sont une nouveauté de la version 1.6. Ils permettent à Snort d'être bien plus flexible dans le formatage et la présentation des sorties à ses utilisateurs. Les modules de sortie son exécutés quand les sous-systèmes d'alerte ou de journalisation de Snort sont appelés, après les préprocesseurs et le moteur de détection. Le format des directives dans le fichier de règles est très similaire à celui des préprocesseurs.

Plusieurs plugins de sortie peuvent être spécifiés dans le fichier de configuration de Snort. Quand plusieurs plugins du même type (journal, alert) sont spécifiés, ils sont "empilés" et appelés en séquence quand un événement se produit. Comme avec les systèmes standards de journalisation et d'alerte, les plugins de sortie envoient leurs données à /var/log/snort par défaut ou vers un répertoire désigné par un utilisateur (en utilisant l'option de la ligne de commande "-l").

Les modules de sortie sont chargés au moment de l'exécution en spécifiant le mot clé *output* dans le fichier de règles : **output <nom>: <options>**
Exemple : **output alert_syslog: LOG_AUTH LOG_ALERT**

1.5.2.1. Alerte syslog (Alert_syslog)

Ce module envoie les alertes à la facilité syslog (comme l'option -s de la ligne de commande). Ce module permet également à l'utilisateur de spécifier la facilité de journalisation et la priorité dans le fichier de règles de Snort, en donnant aux utilisateurs une plus grande flexibilité dans la journalisation des alertes.
Mots clés disponibles :

Options:

LOG_CONS, LOG_NDELAY, LOG_PERROR, LOG_PID, LOG_AUTH, LOG_AUTHPRIV, LOG_DAEMON, LOG_LOCAL, LOG_USER

Priorités :

LOG_EMERG, LOG_ALERT, LOG_CRIT, LOG_ERR, LOG_WARNING, LOG_NOTICE, LOG_INFO, LOG_DEBUG

Format : **alert_syslog: <facilité> <priorité> <options>**
Exemple : **output alert_syslog: LOG_AUTH LOG_ALERT LOG_PID**

1.5.2.2. Alerte rapide (Alert_fast)

Ceci imprimera les alertes de Snort dans un format rapide d'une ligne vers un fichier de sortie spécifié. C'est une méthode d'alerte plus rapide que les alertes *full* car elle n'a pas besoin d'afficher toutes les entêtes des paquets vers le fichier de sortie

Format : **alert_fast: <nom du fichier de sortie>**
Exemple : **output alert_fast: alert.fast**

1.5.2.3. Alerte pleine (Alert_full)

Affiche les messages d'alerte de Snort avec l'intégralité des entêtes des paquets. Cette facilité d'alerte est généralement plutôt lente car elle requière que le programme fasse beaucoup plus d'analyse de données pour formater les données à être imprimées. Les alertes seront écrites dans le répertoire de journalisation par défaut (/var/log/snort) ou le répertoire de journalisation spécifié sur la ligne de commande.

Format : **alert_full: <nom du fichier de sortie>**
Exemple : **output alert_full: alert.full**

1.5.2.4. Alerte smb

Ce plugin envoie des messages d'alerte WinPopup aux noms de machines NETBIOS indiqués dans le fichier spécifié comme argument à ce plugin de sortie. Il doit être noté que l'utilisation de ce plugin **n'est pas** encouragée puisqu'il exécute un exécutable binaire externe (smbclient) au même niveau de privilèges que Snort, communément root. Le format du fichier des stations de travail est une liste de noms NETBIOS des systèmes qui souhaitent recevoir les alertes, un par ligne dans ce fichier.

Format : **alert_smb: <nom du fichier des stations de travail à alerter>**
Exemple : **output alert_smb: workstation.list**

1.5.2.5. Alerte unixsock

Configure un socket du domaine UNIX et y envoie les rapports d'alertes. Des programmes / processus externes peuvent écouter cette socket et recevoir les alertes Snort et les données des paquets en temps réel. C'est actuellement une interface expérimentale.

Format : **alert_unixsock**
Exemple : **output alert_unixsock**

1.5.2.6. Log _tcpdump

Le module log_tcpdump enregistre les paquets vers un fichier au format tcpdump. Ceci est utile pour effectuer des analyses post traitement sur le trafic collecté avec le grand nombre d'outils qui sont disponibles pour examiner des fichiers au format tcpdump. Ce module ne prend qu'un seul argument, le nom du fichier de sortie.

Format : **log_tcpdump: <nom du fichier de sortie>**
Exemple : **output log_tcpdump: snort.log**

1.5.3. Les bases de SNORT

Snort permet d'écrire des règles personnelles et utilise un langage simple et léger de description de règles qui est flexible et assez puissant.
Les règles Snort sont divisées en deux sections logiques, l'entête de la règle et les options de la règle :

L'entête de la règle : il contient comme information l'action de la règle, le protocole, les adresses IP source et destination, les masque réseaux et les ports source et destination.
Option de la règle : contient les messages d'alerte et les informations sur les parties du paquet qui doivent être inspectées pour déterminer si l'action de la règle doit être acceptée.

Exemple : **alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg: "mountd access";)**

1.5.3.1. Les inclusions

Le mot clé include permet à d'autres fichiers de règles d'être inclus dans le fichier de règles indiqué sur la ligne de commande de Snort. Il fonctionne beaucoup comme un "#include" du langage de programmation C, lisant le contenu de fichier nommé et les mettant en place dans le fichier à la place où l'include apparaît.

Format : **include: <répertoire/nom du fichier include>**
Exemple : **#include $RULE_PATH/emerging-botcc-BLOCK.rules**

1.5.3.2. Les variables

Des variables peuvent être définies dans Snort. Ce sont de simples substitutions des variables fixées avec le mot clé var comme dans la Figure 2.

Format : **var: <nom> <valeur>**
Exemple : **var MY_NET [192.168.1.0/24,10.1.1.0/24] alert tcp any any -> $MY_NET any (flags: S; msg: "SYN packet";)**

1.5.4. Les entêtes de règle

1.5.4.1. L'action de règle

L'entête de règle contient l'information qui défini le "qui, où, et quoi" d'un paquet, ainsi que quoi faire dans l'événement où le paquet avec tous les attributs indiqués dans la règle

devrait se présenter. Le premier élément dans une règle est l'action de règle. L'action de règle dit à Snort quoi faire quand il trouve un paquet qui correspond aux critères de la règle. Il y a cinq actions accessibles par défaut dans Snort, alert, log, pass, activate, et dynamic.

alert - génère une alerte en utilisant la méthode d'alerte sélectionnée, et alors journalise le paquet

log - journalise le paquet

pass - ignore le paquet

activate - alerte et alors active une autre règle dynamic

dynamic - reste passive jusqu'à être activée par une règle activate, alors agit comme une règle log

Vous pouvez aussi définir vos propres types de règles et associer un ou plusieurs plugins de sortie avec eux. Vous pouvez alors utiliser le type de règle comme action dans les règles Snort.

Cet exemple créera un type qui journalisera juste vers tcpdump :

```
ruletype suspicious
{
   type log
   output log_tcpdump: suspicious.log
}
```

Cet exemple créera un type de règle qui journalisera vers syslog et une base de données mysql :

```
ruletype redalert
{
   type alert
   output alert_syslog: LOG_AUTH LOG_ALERT
   output database: log, mysql, user=snort dbname=snort host=localhost
}
```

1.5.4.2. Les protocoles

Le champ suivant dans une règle est le protocole. Il y a trois protocoles IP que Snort analyse actuellement pour des comportements suspicieux, tcp, udp, et icmp. Dans le futur il pourra y en avoir plus, tels que ARP, IGRP, GRE, OSPF, RIP, IPX, etc.

1.5.4.3. Les adresses IP

La section suivante de l'entête de règle s'occupe comme information de l'adresse IP et du port pour une règle donnée. Le mot clé "any" peut être utilisé pour définir n'importe quelle

adresse. Snort n'a pas de mécanisme pour fournir de la résolution de nom pour le champ de l'adresse IP dans le fichier de règles. Les adresses sont formées par une simple adresse IP numérique et un bloc CIDR. Le bloc CIDR indique le masque réseau qui doit être appliqué à l'adresse de la règle et à tout paquet qui est testé par rapport à la règle. Un masque de bloc CIDR de /24 indique un réseau de classe C, /16 un réseau de classe B, et /32 indique l'adresse spécifique d'une machine. Par exemple, la combinaison d'adresse/CIDR 192.168.1.0/24 devrait signifier le bloc d'adresses de 192.168.1.1 à 192.168.1.255. Toute règle qui utilise cette désignation pour, disons, l'adresse destination devrait correspondre à toute adresse dans cet intervalle. Les désignations CIDR nous donne une façon rapide de désigner de larges espaces d'adresses avec juste quelques caractères.

Dans la Figure 1, l'adresse IP source était fixée pour correspondre à tout ordinateur parlant, et l'adresse destination était fixée pour correspondre au réseau de classe C 192.168.1.0 .

Il y a un opérateur qui peut être appliqué aux adresses IP, l'opérateur de négation. Cet opérateur dit à Snort de correspondre à toute adresse IP sauf celles indiquées par la liste d'adresses IP. L'opérateur de négation est indiqué par un "!". Par exemple, une modification facile à l'exemple initial est de le faire alerter sur tout trafic qui provient de l'extérieur du réseau local avec l'opérateur de négation :

alert tcp !192.168.1.0/24 any -> 192.168.1.0/24 111 (content: "|00 01 86 a5|"; msg: "external mountd access";)

Ces adresses IP de la règle indiquent "tout paquet tcp avec une adresse source ne provenant pas du réseau interne et à destination du réseau interne".

Vous pouvez aussi spécifier des listes d'adresses IP. Une liste IP spécifiée en entourant une liste d'adresses IP et de blocs CIDR séparés par des virgules entre crochets. Pour le moment, une liste IP ne peut pas inclure d'espaces entre les adresses. Voir la Figure 3 pour un exemple de liste IP en action :

alert tcp ![192.168.1.0/24,10.1.1.0/24] any -> [192.168.1.0/24,10.1.1.0/24] 111 (content: "|00 01 86 a5|"; msg: "external mountd access";)

1.5.4.4. Les numéros de port

Les numéros de ports peuvent être spécifiés de nombre de façons, incluant "any" (ndt : tous les ports), des définitions de ports statiques, des intervalles et des négations. Les ports "any" sont les valeurs génériques, signifiant littéralement tous les ports. Les ports statiques sont indiqués par un seul numéro de port, tel que 111 pour portmapper, 23 pour telnet, ou 80 pour http, etc. Les intervalles de ports sont indiqués avec l'opérateur d'intervalle ":". L'opérateur d'intervalle peut être appliqué dans nombre de façons pour prendre différentes significations :

log udp any any -> 192.168.1.0/24 1:1024 (journalise le trafic udp provenant de tout port et à destination de ports dans l'intervalle de 1 à 1024)

log tcp any any -> 192.168.1.0/24:6000 (journalise le trafic tcp depuis tout port et allant vers les ports inférieurs ou égaux à 6000)
log tcp any :1024 -> 192.168.1.0/24:500 (journalise le trafic tcp depuis les ports privilégiés inférieurs ou égaux à 1024 allant vers les ports supérieurs ou égaux à 500)

La négation de port est indiquée en utilisant l'opérateur de négation "!". L'opérateur de négation peut être appliqué à tous les autres types de règles (excepté "any", qui se traduirait en aucun, combien ce serait Zen...). Par exemple, si pour quelque raison tordue vous voulez tout journaliser sauf les ports X Windows : **log !192.168.1.0/24 any <> 192.168.1.0/24 23**

1.5.4.5. L'opérateur de direction

L'opérateur de direction "->" indique l'orientation, ou la "direction", du trafic auquel la règle s'applique. L'adresse IP et les numéros de ports du côté gauche de l'opérateur de direction est considéré comme le trafic provenant du système source, et les informations d'adresse et de port du côté droit de l'opérateur est le système destination. Il y a aussi un opérateur bidirectionel, qui est indiqué par le symbole "<>". Ceci dit à Snort de considérer les paires adresse/port ou bien en source ou bien en destination de l'orientation. C'est utile pour enregistrer/analyser les deux côtés d'une conversation, tel que des sessions telnet ou POP3. Un exemple de l'opérateur bidirectionel étant utilisé pour enregistrer les deux côtés d'une session telnet : **log !192.168.1.0/24 any <> 192.168.1.0/24 23**

1.5.4.6. Les règles activate/dynamic

Les paires de règles activate/dynamic donnent à Snort une capacité puissante. Vous pouvez maintenant avoir une règle qui en active une autre pour un nombre fixé de paquets quand son action est accomplie. Ceci est très utile si vous voulez configurer Snort pour effectuer l'enregistrement de ce qui suit quand une règle spécifique "se désactive". Les règles d'activation se comportent juste comme les règles alert, excepté qu'elles ont un champ d'option *obligatoire* : "activates". Les règles dynamiques se comportent juste comme les règles log, mais elles ont un champ d'option différent : "activated_by". Les règles dynamiques ont également un second champ obligatoire, "count". Quand la règle "activate" se désactive, elle active la règle dynamique qui lui est liée (indiquée par les numéros d'option activates/activated_by) pour "count" paquets (50 dans ce cas).

Mettez les ensembles et elles ressemblent à ceci :

activate tcp !$HOME_NET any -> $HOME_NET 143 (flags: PA; content: "|E8C0FFFFFF|\bin|; activates: 1; msg: "IMAP buffer overflow!";)
dynamic tcp !$HOME_NET any -> $HOME_NET 143 (activated_by: 1; count: 50;)

Ces règles disent à Snort d'alerter quand il détecte un débordement de tampon dans IMAP et collecte les 50 prochains paquets destinés au port 143 provenant de l'extérieur de $HOME_NET et destinés à $HOME_NET. Si le débordement de tampon arrive et a réussi, il y a de très bonnes possibilités que des données utiles seront contenues dans les prochains 50

(ou quel que soit) paquets allant au même port de service sur le réseau, ainsi il y a avantage à collecter ces paquets pour une analyse ultérieure.

1.5.5. Les options de règle

Les options de règle forment le coeur du moteur de détection d'intrusions de Snort, combinant facilité d'utilisation, puissance et flexibilité. Toutes les options de règle de Snort sont séparées les unes des autres par un caractère point virgule ";". Les mots clés des options de règle sont séparés de leurs arguments avec un caractère deux points ":". Au moment de la rédaction, il y a 15 mots clé d'option de règle disponibles dans Snort :

msg : affiche un message dans les alertes et journalise les paquets

logto : journalise le paquet dans un fichier nommé par l'utilisateur au lieu de la sortie standard

ttl : teste la valeur du champ TTL de l'entête IP

tos : teste la valeur du champ TOS de l'entête IP

id : teste le champ ID de fragment de l'entête IP pour une valeur spécifiée

ipoption : regarde les champs des options IP pour des codes spécifiques

fragbits : teste les bits de fragmentation de l'entête IP

dsize : teste la taille de la charge du paquet contre une valeur

flags : teste les drapeaux TCP pour certaines valeurs

seq : teste le champ TCP de numéro de séquence pour une valeur spécifique

ack : teste le champ TCP d'acquittement pour une valeur spécifiée

itype : teste le champ type ICMP contre une valeur spécifiée

icode : teste le champ code ICMP contre une valeur spécifiée

icmp_id : teste la champ ICMP ECHO ID contre une valeur spécifiée

icmp_seq : teste le numéro de séquence ECHO ICMP contre une valeur spécifique

content : recherche un motif dans la charge d'un paquet

content-list : recherche un ensemble de motifs dans la charge d'un paquet

offset : modifie l'option content, fixe le décalage du début de la tentative de correspondance de motif

depth : modifie l'option content, fixe la profondeur maximale de recherche pour la tentative de correspondance de motif

nocase : correspond à la procédure de chaîne de contenu sans sensibilité aux différences majuscules/minuscules

session : affiche l'information de la couche applicative pour la session donnée

rpc : regarde les services RPC pour des appels à des applications/procédures spécifiques

resp : réponse active (ferme les connexions, etc)

react : réponse active (bloque les sites web)

1.5.5.1. Messages (Msg)

L'option de règle msg dit au moteur de journalisation et d'alerte le message à imprimer avec une sauvegarde du paquet ou une alerte. C'est une simple chaîne texte qui utilise "\" comme caractère d'échappement pour indiquer un caractère discret qui pourrait autrement rendre confus l'analyseur de règles de Snort (tel que le caractère point-virgule ";").

Format : **msg: "<message texte>";**

1.5.5.2. Logto

L'option logto dit à Snort de journaliser tous les paquets qui déclenchent cette règle vers un fichier journal de sortie spécial. C'est spécialement pratique pour combiner les données de choses comme les activités NMAP, les scans CGI HTTP, etc. Il doit être noté que cette option ne marche pas quand Snort est en mode de journalisation binaire.

Format : **logto: "<nom de fichier>";**

1.5.5.3. TTL

Cette règle d'option est utilisée pour fixer la valeur time-to-live à tester. Le test qu'il effectué est réussi seulement sur une correspondance exacte. Ce mot de clé d'option était destiné à être utilisé pour détecter les tentatives de traceroute.

Format : **ttl: "<nombre>";**

1.5.5.4. TOS

Le mot clé "tos" vous permet de vérifier le champ TOS de l'entête IP pour une valeur spécifique. Le test effectué est réussi seulement sur une correspondance exacte.

Format : **tos: "<nombre>";**

1.5.5.5. ID

Ce mot clé d'option est utilisé pour tester une correspondance exacte dans le champ ID de fragment de l'entête IP. Quelques programmes de pirates (et d'autres programmes) fixent ce champ spécifiquement pour différents besoins, par exemple la valeur 31337 est très populaire avec certains pirates. Ceci peut être retourné contre eux en mettant en place une simple règle pour tester ceci et quelques autres "nombres de pirates".

Format : **id: "<nombre>";**

1.5.5.6. Options IP (Ipoption)

Si des options IP sont présentes dans un paquet, cette option va chercher l'utilisation d'une option spécifique, tel que le routage par la source. Les arguments valides de cette option sont :

rr - Record route (ndt : enregistrement de la route)

eol - End of list (ndt : fin de liste)

nop - No op (ndt : pas d'opération)

ts - Time Stamp (ndt : horaire)

sec - IP security option (ndt : option de sécurité IP)

lsrr - Loose source routing (ndt : routage vague par la source)

ssrr - Strict source routing (ndt : routage stricte par la source)

satid - Stream identifier

Les options IP les plus fréquemment regardées sont les routages vague et stricte par la source qui ne sont utilisés dans aucune application Internet. Seulement une seule option peut être spécifiée par règle.

Format : **ipopts: <option>;**

1.5.5.7. Fragment bits (Fragbits)

Cette règle inspecte les bits de fragment et le bit réservé dans l'entête IP. Il y a trois bits qui peuvent être vérifiés, le bit Reserved Bit (RB), le bit More Fragments (MF) et le bit Dont Fragment (DF). Ces bits peuvent être vérifiés pour une variété de combinaisons. Utilisez les valeurs suivantes pour indiquer les bits spécifiques :

R - Reserved Bit

D - DF bit

M - MF bit

Vous pouvez également utiliser des modificateurs pour indiquer des critères logiques de correspondance pour les bits spécifiés :

+ - Tous les drapeaux, correspond si au moins les bits spécifiés sont positionnés

* - Au moins un drapeau, correspond si au moins un des bits spécifiés est positionné

! - Aucun drapeau, correspond si aucun des bits spécifiés n'est positionné

Format : **fragbits: <valeurs des bits>;**

Exemple**: alert tcp !$HOME_NET any -> $HOME_NET any (fragbits: R+; msg: "Reserved IP bit set!";)**

1.5.5.8. La charge du paquet (Dsize)

L'option dsize est utilisée pour tester la taille de la charge du paquet. Il peut être fixé à toute valeur, utilise en plus les signes supérieur/inférieur pour indiquer des intervalles et des limites. Par exemple, si vous savez qu'un certain service a un tampon d'une certaine taille, vous pouvez fixer cette option pour regarder les tentatives de débordement de tampons. Cela a l'avantage supplémentaire d'être une façon bien plus rapide de tester contre les débordements de tampons qu'une vérification de contenu de la charge.

Format : **dsize: [>|<] <nombre>;** (note : Les opérateurs > et < sont optionnels !)

1.5.5.9. Contenu (Content)

Le mot clé content est une des fonctionnalités les plus importantes de Snort. Il autorise l'utilisateur de fixer des règles qui recherchent un contenu spécifique dans la charge du paquet et déclencher une réponse basée sur les données. A chaque fois que l'option content de correspondance de motif est exécutée, la fonction de correspondance de motif de Boyer-Moore est appelée et le test (plutôt cher en temps de calcul) est effectué contre le contenu du paquet. Si une donnée correspondant exactement à la chaîne de donnée en argument est contenue n'importe où dans la charge de paquet, le test est réussi et le reste des options de la règle est exécutée. Sachez que ce test fait la différence entre majuscules et minuscules.

La donnée d'option pour le mot clé content est quelque peu complexe; il peut contenir du texte et des données binaires mélangées. Les données binaires sont généralement entourées dans des caractères barre verticale ("|") et représentées en tant que bytecode. Un bytecode représente une donnée binaire comme des nombres hexadécimaux et c'est une bonne méthode de raccourci pour décrire des données binaires complexes. La Figure 7 contient un exemple de mélange de textes et de données binaires dans une règle Snort.

Exemple: **alert tcp any any -> 192.168.1.0/24 143 (content: "|90C8 C0FF FFFF|/bin/sh"; msg: "IMAP buffer overflow!";)**

Format : **content: "<chaîne de contenu>";**

1.5.5.10. Offset

L'option de règle offset est utilisée comme un modificateur des règles utilisant le mot clé d'option content. Ce mot clé modifie la position de début de recherche pour la fonction de correspondance de motif depuis le début de la charge du paquet. Il est très utile pour des choses comme des règles de détection de scans CGI où la chaîne de recherche de contenu n'est jamais trouvée dans les quatre premiers octets de la charge. Soin doit être pris à ne pas fixer la valeur offset trop strictement sous peine de manquer potentiellement une attaque ! Ce mot clé d'option de règle ne peut pas être utilisé sans spécifier également l'option de règle content.

Format : **offset: <nombre>;**

1.5.5.11. Profondeur (Depth)

Depth est une autre modification de l'option de règle contenu. Ceci fixe la profondeur maximale de recherche dans la fonction contenu de correspondance de motif de recherche depuis le début de sa région de recherche. Il est utile pour limiter la fonction de correspondance de motif d'exécuter des recherches inefficaces une fois que la région de recherche probable pour un ensemble de contenus donné a été dépassée. (C'est à dire que si vous recherchez "cgi-bin/php" dans un paquet relatif au web, vous n'avez probablement pas besoin de perdre du temps à rechercher dans la charge après les 20 premiers octets!) Voir la Figure 8 pour un exemple de règle de recherche combinant content, offset et depth.

Format : **depth: <nombre>;**

Exemple: **alert tcp any any -> 192.168.1.0/24 80 (content: "cgi-bin/phf"; offset: 3; depth: 22; msg: "CGI-PHF access";)**

1.5.5.12. Aucun cas (Nocase)

L'option nocase est utilisée pour désactiver la différence majuscules/minuscules dans une règle "content". Elle est spécifiée seule dans une règle et tout caractère ASCII qui est comparé à la charge du paquet est traité comme si il était soit en majuscule soit en minuscule.

Format : **nocase;**

Exemple: **alert tcp any any -> 192.168.1.0/24 21 (content: "USER root"; nocase; msg: "FTP root user access attempt";)**

1.5.5.13. Drapeaux (Flags)

Cette règle teste les drapeaux TCP pour une correspondance. Il y a actuellement 8 drapeaux variables qui sont disponibles dans Snort :

F - FIN (le bit le moins significatif dans l'octet des drapeaux TCP)

S – SYN, R – RST, P – PSH, A – ACK, U - URG

2 - bit réservé 2

1 - bit réservé 1

Il y a aussi des opérateurs logiques qui peuvent être utilisés pour spécifier des critères de correspondance pour les drapeaux indiqués :

+ - Tous les drapeaux, correspond si au moins les bits spécifiés sont positionnés

* - Au moins un drapeau, correspond si au moins un des bits spécifiés est positionné

! - Aucun drapeau, correspond si aucun des bits spécifiés n'est positionné

Les bits réservés peuvent être utilisés pour détecter des comportements non usuels, tels que des tentatives d'empreintes digitales de piles IP ou d'autres activités suspicieuses.

Format : **flags: <valeurs de drapeaux>;**

Exemple: **alert any any -> 192.168.1.0/24 any (flags: SF; msg: "Possible SYN FIN scan";)**

1.5.5.14. Séquence TCP (Seq)

Cette option de règle se réfère aux numéros de séquence TCP. Essentiellement, il détecte si le paquet a un numéro de séquence statique fixé, et donc plutôt peu utilisé. Elle a été incluse pour le bien de l'exhaustivité.

Format : **seq: <nombre>;**

1.5.5.15. Acquittement (Ack)

Le mot clé ack d'option de règle se réfère au champ d'acquittement de l'entête TCP. Cette règle a un propos pratique jusqu'ici : détecter les pings TCP NMAP. Un ping TCP NMAP fixe ce champ à zéro et envoie un paquet avec le drapeau ACK TCP pour déterminer si un système réseau est actif.

Format : **ack: <nombre>;**

Exemple: **alert any any -> 192.168.1.0/24 any (flags: A; ack: 0; msg: "NMAP TCP ping";)**

1.5.5.16. Type ICMP (Itype)

Cette règle teste la valeur du champ type ICMP. Il est fixé en utilisant la valeur numérique du champ. Pour une liste des valeurs disponibles, regardez dans le fichier decode.h

inclus avec Snort ou dans toute référence ICMP. Il doit être noté que les valeurs peuvent être fixées en dehors de l'intervalle pour détecter des valeurs de type ICMP invalides qui sont quelques fois utilisées dans des dénis de services et des attaques en inondations.

Format : **itype: <nombre>;**

1.5.5.17. Code ICMP (Icode)

Le mot clé d'option de règle icode est plutôt identique à la règle itype, fixez juste une valeur numérique ici et Snort va détecter tout trafic en utilisant cette valeur de code ICMP. Les valeurs hors intervalle peut également être fixé pour détecter le trafic suspicieux.

Format : **icode: <nombre>;**

1.5.5.18. Session

Le mot clé session est nouveau depuis la version 1.3.1.1 et est utilisé pour extraire les données utilisateurs depuis les sessions TCP. Il est extrêmement utile pour voir ce que les utilisateurs tapent dans telnet, rlogin, ftp et même les sessions web. Il y a deux mots clé disponibles en argument pour l'option de règle session, printable ou all. Le mot clé printable affiche seulement les données que l'utilisateur verrait normalement ou serait capable de taper. Le mot clé all substitue les caractères non imprimables avec leurs équivalents hexadécimaux. Cette fonction peut ralentir considérablement Snort, donc elle ne devrait pas être utilisée dans des situations de charge importante, et est probablement mieux adaptée au post-traitement de fichiers journaux binaires (format tcpdump).

Format : **session: [printable|all];**

Exemple: **log tcp any any <> 192.168.1.0/24 23 (session: printable;)**

1.5.5.19. Icmp_id

L'option icmp_id examine le numéro ICMP ID d'un paquet ECHO ICMP pour une valeur spécifique. C'est utile car quelques programmes de canaux cachés utilisent des champs ICMP statiques quand ils communiquent. Ce plugin particulier a été développé pour activer la règle de détection de stacheldraht écrite par Max Vision, mais c'est certainement utile pour détecter un nombre potentiel d'attaques.

Format : **icmp_id: <nombre>;**

1.5.5.20. ICMP séquence (Icmp_seq)

L'option Icmp_seq examine le champ séquence ICMP d'un paquet ECHO ICMP pour une valeur spécifique. C'est utile car quelques programmes de canaux cachés utilisent des champs

ICMP statiques quand ils communiquent. Ce plugin particulier a été développé pour activer la règle de détection de stacheldraht écrite par Max Vision, mais c'est certainement utile pour détecter un nombre potentiel d'attaques. (Et oui, je sais que l'information pour ce champ est presque identique à la description de icmp_id, c'est pratiquement la même fichue chose!)

Format : **icmp_seq: <nombre>;**

1.5.5.21. Rpc

Cette option regarde les requêtes RPC et décode automatiquement l'application, la procédure et la version de programme, en indiquant un succès quand les trois variables correspondent toutes. Le format de l'option d'appel est "application,procédure,version". Les caractères génériques sont valides pour la procédure et les numéros de version et sont indiquées par une "*".

Format : **icmp_seq: <nombre,[nombre|*],[nombre|*]>;**

Exemple :

alert tcp any any -> 192.168.1.0/24 111 (rpc: 100000,*,3; msg:"RPC getport (TCP)";)
alert udp any any -> 192.168.1.0/24 111 (rpc: 100000,*,3; msg:"RPC getport (UDP)";)
alert udp any any -> 192.168.1.0/24 111 (rpc: 100083,*,*; msg:"RPC ttdb";)
alert udp any any -> 192.168.1.0/24 111 (rpc: 100232,10,*; msg:"RPC sadmin";)

1.5.5.22. Réponses (Resp)

Le mot clé resp met en oeuvre les réponses flexibles (FlexResp) au trafic qui correspond à une règle Snort. Le code FlexResp permet à Snort de fermer activement les connexions en infraction. Les arguments suivants sont valides pour ce module :

rst_snd - envoie des paquets TCP-RST à la socket expéditrice

rst_rcv - envoie des paquets TCP-RST à la socket destinataire

rst_all - envoie des paquets TCP_RST dans les deux directions

icmp_net - envoie un paquet ICMP_NET_UNREACH à l'expéditeur

icmp_host - envoie un paquet ICMP_HOST_UNREACH à l'expéditeur

icmp_port - envoie un paquet ICMP_PORT_UNREACH à l'expéditeur

icmp_all - envoie tous les paquets ci-dessus à l'expéditeur

Les options peuvent être combinées pour envoyer des réponses multiples au système cible. Les arguments multiples sont séparés par des virgules.

Format : **resp: <resp_modifier[, resp_modifier...]>;**

Exemple :

alert tcp any any -> 192.168.1.0/24 1524 (flags: S; resp: rst_all; msg: "Root shell backdoor attempt";)
alert udp any any -> 192.168.1.0/24 31 (resp: icmp_port,icmp_host; msg: "Hacker's Paradise access attempt";)

1.5.5.23. Liste de contenu (Content-list)

Le mot clé content-list permet à de multiples chaînes de contenu d'être spécifiées à la place d'une unique option de contenu. Les motifs qui seront recherchés doivent pour chacun être spécifiés sur une seule ligne du fichier content-file, mais sinon ils sont traités identiquement aux chaînes de contenue spécifiées comme un argument à la directive standard content. Cette option est la base pour le mot clé react.

Format : **content-list: "<nom de fichier>";**

1.5.5.24. Réaction (React)

Le mot clé react basé sur les réponses flexibles (FlexResp) met en oeuvre la réaction flexible au trafic qui correspond à une règle Snort. La réaction de base est de bloquer les sites intéressants que les utilisateurs veulent accéder : New York Times, slashdot, ou quelque chose de vraiment important - napster et les sites pornos. Le code Flex Resp permet à Snort de fermer activement les connexions en infraction et/ou d'envoyer un message visible au navigateur (modificateur d'avertissement disponible bientôt). Le message peut inclure votre propre commentaire. Les arguments suivants (modificateurs de base) sont valides pour cette option :

block : ferme la connexion et envoie message visible

warn : envoie le message d'avertissement visible (sera disponible bientôt), l'argument de base peut être combiné avec les arguments suivants (modificateurs supplémentaires) :

msg : inclut le texte de l'option msg dans le message visible de bloquage

proxy: <port_nr> - utilise le port du relais pour envoyer le message visible

Des arguments additionnels multiples sont séparés par une virgule. Le mot clé react doit être placé en dernier dans la liste des options.

Format : **react: <react_basic_modifier[, react_additional_modifier...]>;**

Exemple:

tcp any any <> 192.168.1.0/24 80 (content-list: "adults"; msg: "Not for children!"; react: block, msg;)

alert tcp any any <> 192.168.1.0/24 any (content-list: "adults"; msg: "Adults list access attempt"; react: block;)

1.6. Dependences de Snort

1.6.1. Barnyard2

Barnyard2 est un fork du projet étable barnyad, conçue spécifiquement pour Snort et le nouveau format de fichier unified2. Barnyard2 est en développement actif et continue de s'adapter en fonction des commentaires des utilisateurs. [W5]

La version actuelle de 2 à 1,8 a les caractéristiques suivantes:

Traitement des sorties Délestage de vos fichiers d'alerte de Snort à un processus dédié, en minimisant les paquets perdus en soi, Snort.

Analyse unified2 fichiers.

Utilise la syntaxe de configuration similaire à celle de Snort pour simplifier le déploiement.

Supporte tous les plugins de sortie de Snort (sauf alert_sf_socket) ainsi que deux plugins supplémentaires (Sguil et CEF).

Barnyard2 a été écrit à partir de zéro et tirant parti hors des routines fondamentales de Snort et continuellement est aligné sur les dernières versions de Snort. Il est distribué sous la licence GPLv2

1.6.2. BASE

Par défaut, les alertes de Snort sont enregistrées dans un simple fichier texte. L'analyse de ce fichier n'étais pas aisée, meme en utilisant des outils de filtre et de tri. C'est pour cette raison qu'il est vivement conseillé d'utiliser des outils de monitoring. Parmis ceux-ci, le plus vogue actuellement est BASE(Basic Analysis and Security Engine), un projet open-source basé sur Acid(Analysis Console for Intrusion Database). La console BASE est une application Web écrite en PHP qui interface la base de données dans laquelle Snort stocke ses alertes.
Pour fonctionner, BASE a besoin d'un certain nombre de dépendances :

Un SGBD installé : par exemple MySQL.
Snort : compilé avec le support de ce SGBD.
Un serveur http : par exemple Apche.
PHP5 : module PHP.
PHP5-MySQL : interface PHP/MySQL.
La bibliothèque ADODB(Active DATA Objects DATA Base) : c'est une bibliothèque destinée à communiquer avec différents systèmes de gestion de base de données (SGBD) comme MySQL, SQL Server,etc. Ecrite au début en PHP, il existe également une version en Python.
PHP-mail : extension PHP

PHP-gd
PHP-pear

2. Installation de Snort

Installer les conditions préalables des dépôts Ubuntu:

sudo apt-get install -y build-essential libpcap-dev libpcre3-dev libdumbnet-dev bison flex zlib1g-dev

Modifier de **/etc/network/interfaces** comme un admin:

sudo pico /etc/network/interfaces

Rajouter les deux lignes suivantes pour chaque interface réseau :

post-up ethtool -K eth0 gro off
post-up ethtool -K eth0 lro off

Création d'un répertoire pour enregistrer les fichiers téléchargé:

mkdir ~/snort_src
cd ~/snort_src

Snort utilise la bibliothèque d'acquisition de données (DAQ) aux appels abstraits par paquets bibliothèques de capture. DAQ est téléchargé et installer à partir du site Snort:

cd ~/snort_src
wget https://www.snort.org/downloads/snort/daq-2.0.6.tar.gz
tar -xvzf daq-2.0.6.tar.gz
cd daq-2.0.6
./configure
make
sudo make install

L'installation de Snort peut être effectuée :

cd ~/snort_src
wget https://www.snort.org/downloads/snort/snort-2.9.8.2.tar.gz
tar -xvzf snort-2.9.8.2.tar.gz
cd snort-2.9.8.2
./configure --enable-sourcefire
make
sudo make install

Exécuter la commande suivante pour mettre à jour des bibliothèques partagées :

sudo ldconfig

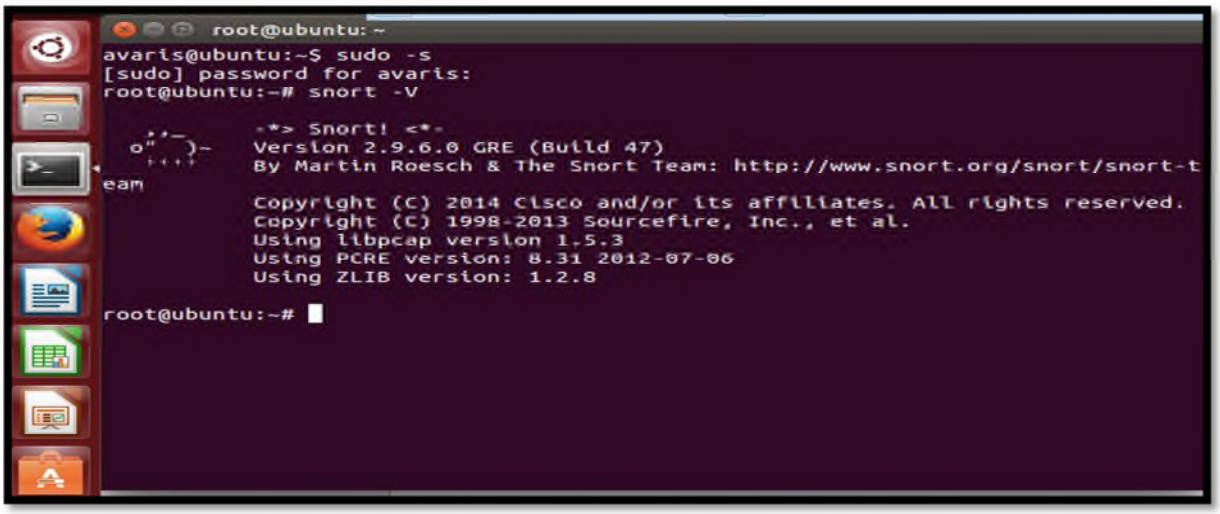
Placer le binaire local de Snort dans **/usr/local/bin/snort** et créer un lien symbolique vers **/use/sbin/snort** :

sudo ln -s /usr/local/bin/snort /usr/sbin/snort

Tester que le binaire Snort fonctionne avec la commande suivante :

/usr/sbin/snort –V

On aura un résultat comme sur la figure suivante :



Configuration des règles de Snort, pour des raisons de sécurité, afin que Snort fonctionne en tant qu'utilisateur non privilégié.

Création de l'utilisateur Snort et le groupe :

sudo groupadd snort
sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort

Création d'un certain nombre de fichiers et dossier dont Snort a besoin lors de son exécution en mode NIDS, on modifier en suite la propriété de ces fichier pour le nouvel utilisateur Snort. Les fichiers de configuration de Snort sont stocké dans **/etc/snort**, les règles dans **/etc/sort/rules** et **/usr/local/lib/snort_dynamicrules**, et on stocke ces journaux dans **/var/log/snort,** tous sa avec les commande suivantes **:**

creation des repertoire de Snort:

sudo mkdir /etc/snort
sudo mkdir /etc/snort/rules
sudo mkdir /etc/snort/rules/iplists
sudo mkdir /etc/snort/preproc_rules
sudo mkdir /usr/local/lib/snort_dynamicrules
sudo mkdir /etc/snort/so_rules

Création des fichiers qui stocke les règles

sudo touch /etc/snort/rules/iplists/default.blacklist
sudo touch /etc/snort/rules/iplists/default.whitelist
sudo touch /etc/snort/rules/local.rules

Création des repertoires de journalisation

sudo mkdir /var/log/snort
sudo mkdir /var/log/snort/archived_logs

Régler les autorisations **:**

sudo chmod -R 5775 /etc/snort
sudo chmod -R 5775 /var/log/snort
sudo chmod -R 5775 /var/log/snort/archived_logs
sudo chmod -R 5775 /etc/snort/so_rules
sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules

#changement de propriété sur les dossiers **:**

sudo chown -R snort:snort /etc/snort
sudo chown -R snort:snort /var/log/snort
sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules

Déplacer les fichiers suivants vers **/etc/snort** avec les commandes suivantes :

cd ~/snort_src/snort-2.9.8.2/etc/
sudo cp *.conf* /etc/snort
sudo cp *.map /etc/snort
sudo cp *.dtd /etc/snort

cd ~/snort_src/snort-2.9.8.2/src/dynamic-preprocessors/build/usr/local/lib/snort_dynamicpreprocessor/
sudo cp * /usr/local/lib/snort_dynamicpreprocessor/

Modification du fichier de configuration Snort. Le fichier de configuration Snort est stocké a **/etc/snort/snort.conf**, et contient tous les paramètres que Snort va utiliser quand il est exécuté en mode NIDS. Ceci est un fichier volumineux (bien plus de 500 lignes), et contient un certain nombre d'options pour la configuration de Snort. Nous sommes intéressés par seulement quelques réglages à l'heure actuelle.

Mettre en commentaire toutes les règles de Snort avec la commande suivante :

sudo sed -i 's/include \$RULE_PATH/#include \$RULE_PATH/' /etc/snort/snort.conf

Modifier quelques lignes dans le fichier **snort.conf**, on ouvre le fichier **Snort.conf** avec la commande suivante :

sudo pico /etc/snort/snort.conf

La ligne 45 et on modifie la ligne de **ipvar $HOME_NET any** vers :

ipvar HOME_NET 192.168.1.0/24

```
# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.1.0/24
```

Insertion des chemins et des répertoires que nous avons créés plus tôt comme suit :

var RULE_PATH /etc/snort/rules # ligne 104
var SO_RULE_PATH /etc/snort/so_rules # ligne 105
var PREPROC_RULE_PATH /etc/snort/preproc_rules # ligne 106

var WHITE_LIST_PATH /etc/snort/rules/iplists # ligne 113
var BLACK_LIST_PATH /etc/snort/rules/iplists # ligne 114

```
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules

# If you are using reputation preprocessor set these
# Currently there is a bug with relative paths, they are relative to where snor$
# not relative to snort.conf like the above variables
# This is completely inconsistent with how other vars work, BUG 89986
# Set the absolute path appropriately
var WHITE_LIST_PATH /etc/snort/rules/iplists
```

Activer les règles qu'on veut utiliser en enlevant le « # » qui se trouve en début de la ligne, maintenant on va éditer un seul fichier règles **local.rules** et pour que Snort l'utilise on doits lui enlevé le # qui rend la ligne comme commentaire

include $RULE_PATH/local.rules

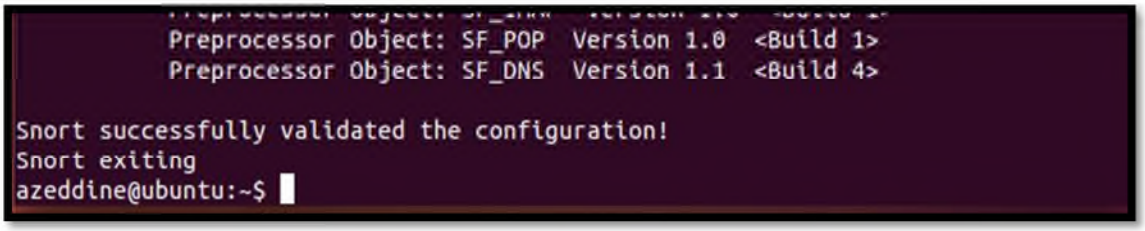
```
# site specific rules
include $RULE_PATH/local.rules

#include $RULE_PATH/app-detect.rules
#include $RULE_PATH/attack-responses.rules
#include $RULE_PATH/backdoor.rules
#include $RULE_PATH/bad-traffic.rules
```

Pour s'assurer du bon fonctionnement de Snort on exécute la commande suivante :

sudo snort -T -c /etc/snort/snort.conf

Réponse : résultat satisfaisant :



Pour écrire sur le fichier **local.rules** on ouvre d'abord le fichier à l'aide de la commande :

sudo pico /etc/snort/rules/local.rules

puis on écris les règles qui nous intéresse le plus, exemple :

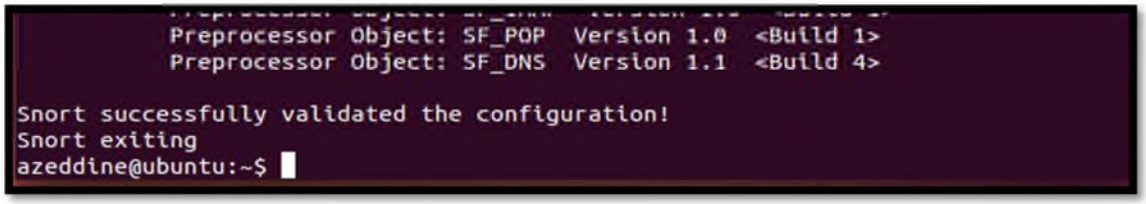
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:10000001; rev:001;)

Cette règle dit que pour tous les paquets ICMP qu'il voit dans tout le réseau à HOME_NET, génère une alerte avec le test ICMP texte.

Apres avoir apporté des modifications aux fichiers qui reniflent des charges, on doit tester le fichier de configuration à nouveau:

sudo snort -T -c /etc/snort/snort.conf

On aura un résultat comme sur la figure suivante :



3. Installation de Barnyard2

Tout d'abord, installer certaines conditions préalables:

sudo apt-get install -y mysql-server libmysqlclient-dev mysql-client autoconf libtool

Lors de l'exécution de la commande précédente nous aurons besoin de configurer le mot de passe MySQL root.

Modifier **snort.conf**. Nous avons besoin d'ajouter une ligne qui indique Snort les événements de sortie sous forme binaire de sorte que Barnyard2 puisse les lire. Après la ligne 520 dans **/etc/snort/snort.conf** ,on ajoute la ligne suivante et on enregistre le fichier:

output unified2: filename snort.u2, limit 128

Cette ligne indique à Snort des événements de sortie dans le format binaire unified2.

Télécharger, configurer et installer Barnyard2:

```
cd ~/snort_src
wget https://github.com/firnsy/barnyard2/archive/v2-1.13.tar.gz -O barnyard2-2-1.13.tar.gz
tar zxvf barnyard2-2-1.13.tar.gz
cd barnyard2-2-1.13
autoreconf -fvi -I ./m4
```

Selon l'architecture du système (x86 ou x64), exécute une des deux lignes suivantes:

```
./configure --with-mysql --with-mysql-libraries=/usr/lib/x86_64-linux-gnu
./configure --with-mysql --with-mysql-libraries=/usr/lib/i386-linux-gnu
```

Une fois terminer on continue avec les commandes suivantes:

```
make
sudo make install
sudo ln -s /usr/include/dumbnet.h /usr/include/dnet.h
sudo ldconfig
```

Barnyard2 est maintenant installé dans **/usr/local/bin/barnyard2**. configurer Snort pour qu'il puisse utiliser Barnyard2, nous avons besoin de quelques fichiers:

```
cd ~/snort_src/barnyard2-2-1.13
sudo cp etc/barnyard2.conf /etc/snort

sudo mkdir /var/log/barnyard2
sudo chown snort.snort /var/log/barnyard2

sudo touch /var/log/snort/barnyard2.waldo
sudo chown snort.snort /var/log/snort/barnyard2.waldo
sudo touch /etc/snort/sid-msg.map
```

Créer la base de données pour enregistrer les alertes de Barnayrd2, pour cela on exécute les commandes suivantes :

```
$ mysql -u root -p
mysql> create database snort;
mysql> use snort;
mysql> source ~/snort_src/barnyard2-2-1.13/schemas/create_mysql
mysql> CREATE USER 'snort'@'localhost' IDENTIFIED BY 'snortpass';
mysql> grant create, insert, select, delete, update on snort.* to 'snort'@'localhost';
mysql> exit
```

Maintenant que la base de données Snort a été créée, modifier le fichier de configuration de barnayrd2 et entrer les détails sur la base de données créé :

sudo pico /etc/snort/barnyard2.conf

Ajouter à la fin du fichier la ligne suivante :

output database: log, mysql, user=snort password=MYSQLSNORTPASSWORD dbname=snort host=localhost

Étant donné que le mot de passe est dans le fichier barnyard2.conf, nous devrions empêcher les autres utilisateurs de le lire:

sudo chmod o-r /etc/snort/barnyard2.conf

Maintenant Barnyard2 est configuré pour fonctionner avec Snort. Pour tester, nous allons exécuter Snort et Barnyard2 et générer des alertes :

sudo /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i eth0 –D

Exécute la commande suivante pour dire à Barnyard2 de sauvegarder ces événements dans l'instance de la base de données de Snort :

sudo barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 –w /var/log/snort/barnyard2.waldo -g snort -u snort

Pour vérifier si Barnyard2 a enregistré ces évènements dans la base de données de Snort on exécute la commande suivante:

mysql -u snort -p -D snort -e "select count(*) from event"

Le nombre d'évènements doit être supérieur à 0

Nombre d'évènements

4. Installation de B.A.S.E

Installer d'abord les conditions préalables :

sudo apt-get install -y apache2 libapache2-mod-php5 php5 php5-mysql php5-common php5-gd php5-cli php-pear

En suite installer ADODB :

```
cd ~/snort_src
wget http://sourceforge.net/projects/adodb/files/adodb-php5-only/adodb-518-for-php5/adodb518a.tgz/download -O adodb518.tgz
tar -xvzf adodb518.tgz
sudo mv adodb5 /var/adodb
```

Installation de B.A.S.E :

Exécuter les commandes suivantes :

```
cd ~/snort_src
wget http://sourceforge.net/projects/secureideas/files/BASE/base-1.4.5/base-1.4.5.tar.gz
tar -zxvf base-1.4.5.tar.gz
sudo mv base-1.4.5 /var/www/html/base/
cd /var/www/html/base
sudo cp base_conf.php.dist base_conf.php
sudo chown -R www-data:www-data /var/www/html/base
sudo chmod o-r /var/www/html/base/base_conf.php
sudo pico /var/www/html/base/base_conf.php
```

La dernière commande va nous permettre d'ouvrir le fichier **base_conf.php**, insérer les configurations nécessaires :

```
$BASE_urlpath = '/base';            # ligne 50
$DBlib_path = '/var/adodb/';        #ligne 80
$alert_dbname   = 'snort';          # ligne 102
$alert_host     = 'localhost';
$alert_port     = '';
$alert_user     = 'snort';
$alert_password = 'MYSQLSNORTPASSWORD';             # ligne 106
```

Redémarrer le service apache2:

sudo service apache2 restart

5. lancement d'attaques

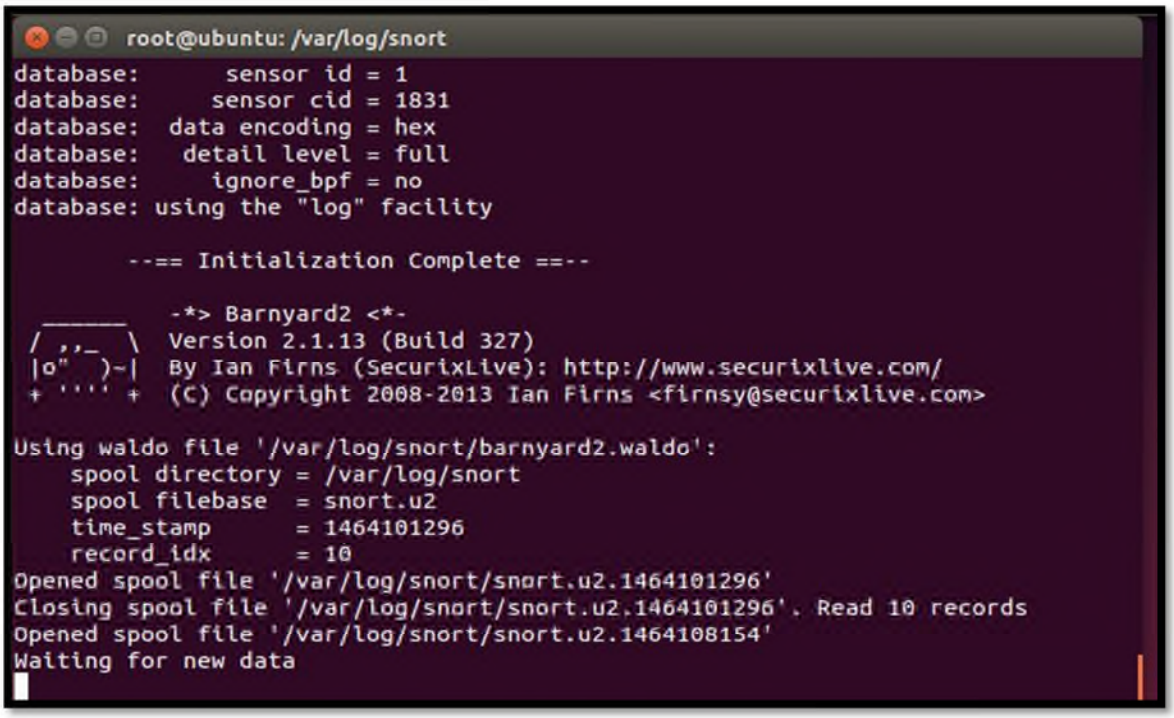
Pour faire le scénario d'attaque, on a besoins de deux PC, l'un jouera le rôle de lanceur d'attaques et l'autre essayera de détecter l'attaque.

Dans la machine victime on lance snort :

sudo /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i eth0 –D

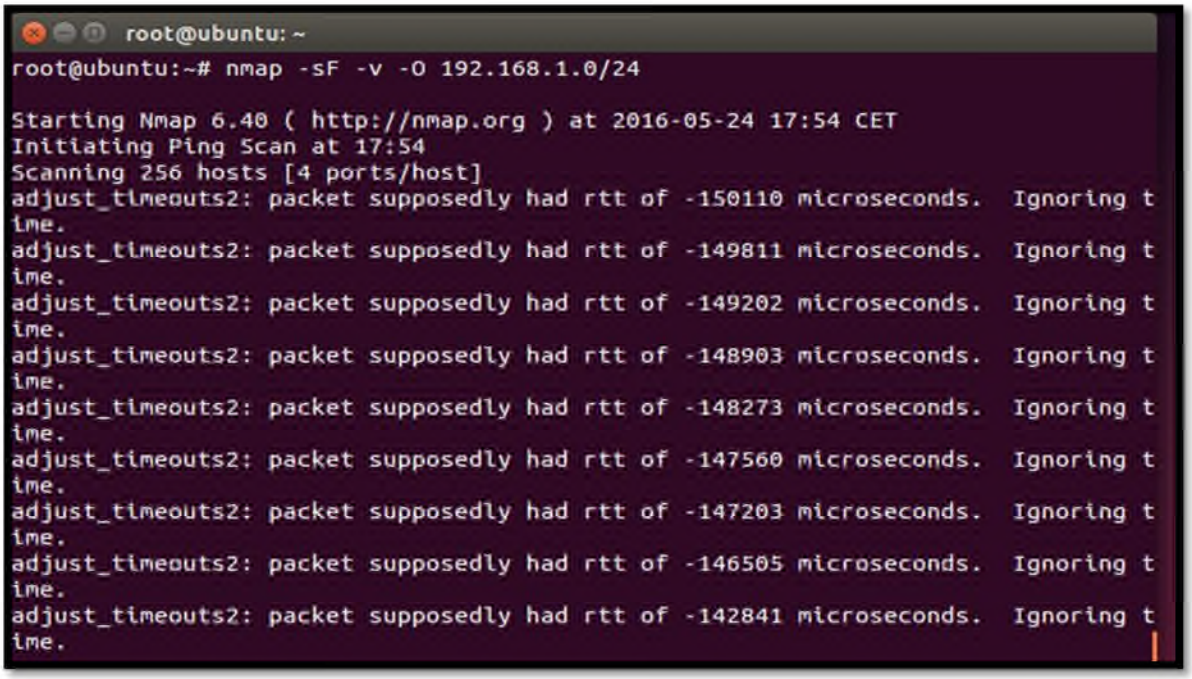
Lancer Barnyard2:

sudo barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort

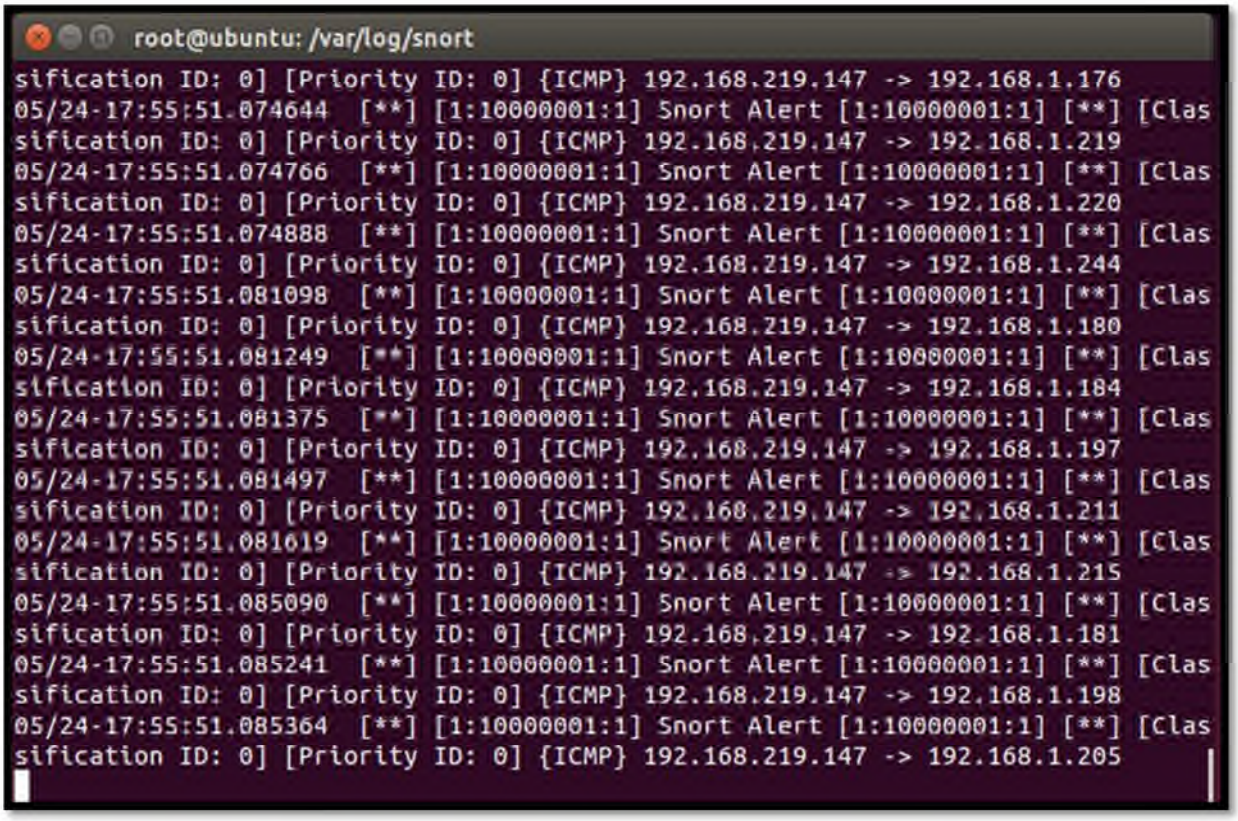


Après sa on lance un scan depuis la machine attaquante :

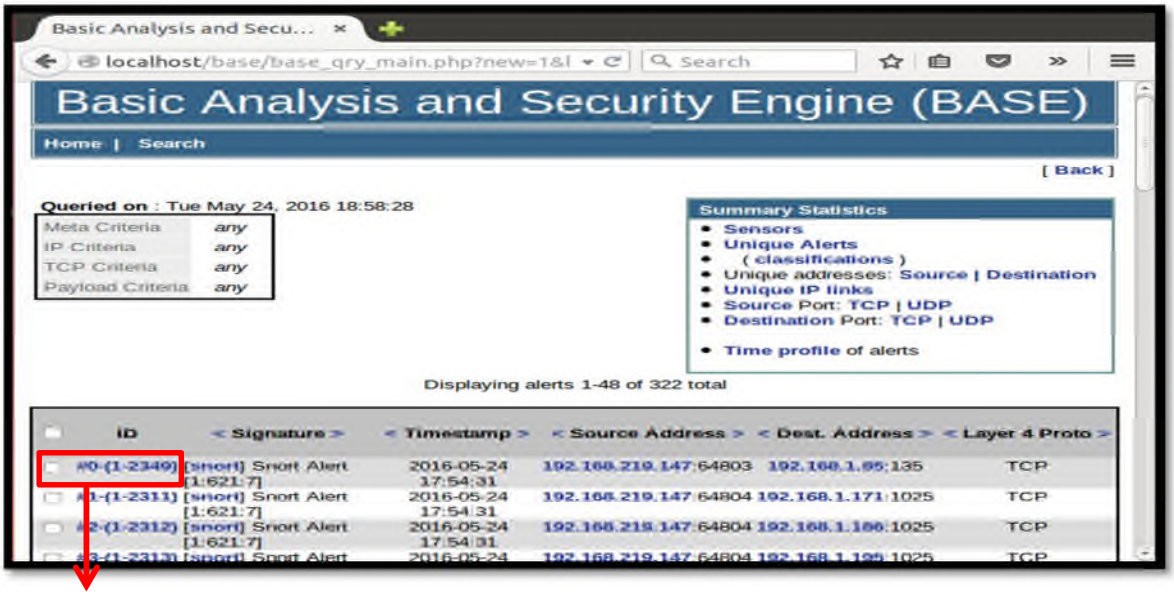
nmap -sF -v -O 192.168.1.0/24



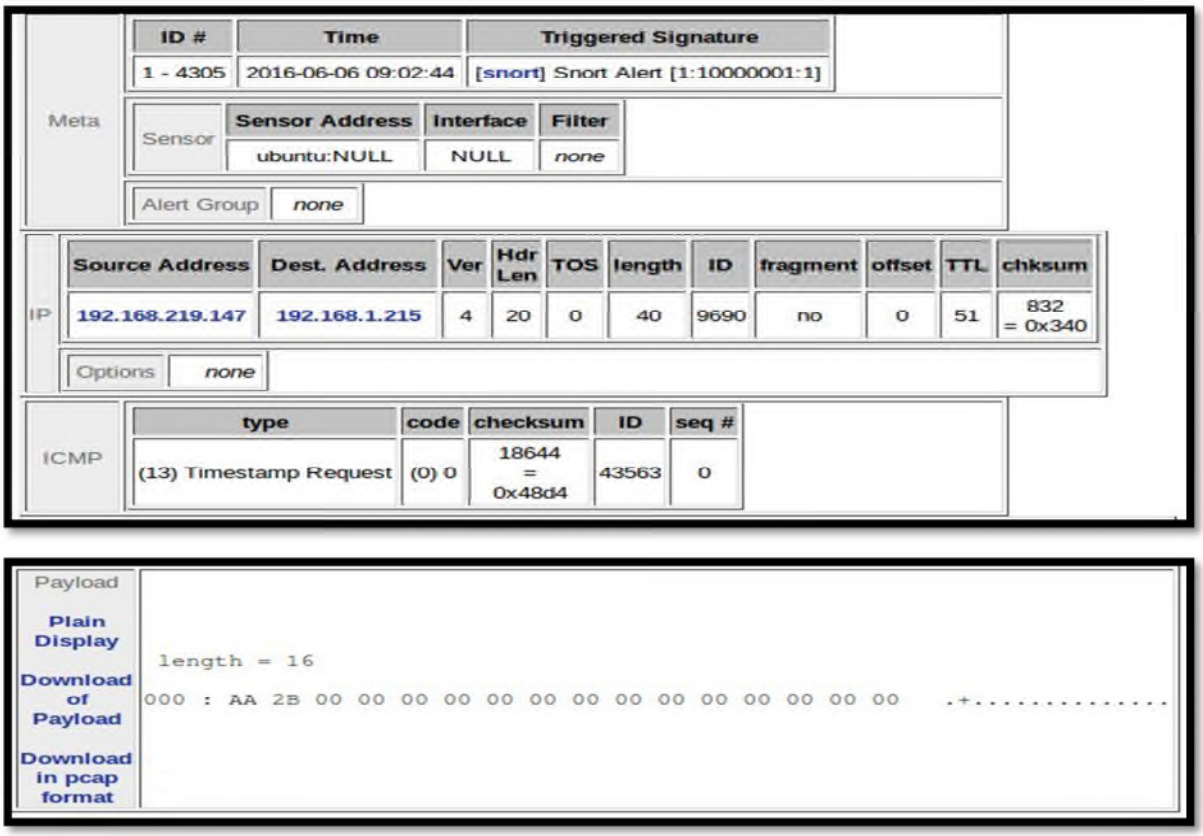
Dans la machine victime on remarque rapidement la détection de ce scan depuis Barnyard2 :



Dans la console BASE les alertes s'affichent comme suit :



Click ici pour avoir plus d'information sur cette alerte, le résultat sera comme suit :



Meta

ID #	Time	Triggered Signature
1 - 4305	2016-06-06 09:02:44	[snort] Snort Alert [1:10000001:1]

Sensor

Sensor Address	Interface	Filter
ubuntu:NULL	NULL	none

Alert Group: none

IP

Source Address	Dest. Address	Ver	Hdr Len	TOS	length	ID	fragment	offset	TTL	chksum
192.168.219.147	192.168.1.215	4	20	0	40	9690	no	0	51	832 = 0x340

Options: none

ICMP

type	code	checksum	ID	seq #
(13) Timestamp Request	(0) 0	18644 = 0x48d4	43563	0

Conclusion

Dans cette partie nous avons réalisé expérimentalement et de façon manuelle l'installation de Snort, avec la quasi-totalité de ses fonctions et intégrer des règles de sécurité qui sont nécessaire pour la détection des attaques afin de renforcer la sécurité au niveau du réseau LAN.

Conclusion générale

Les réseaux informatiques ont évolué au fil du temps à une vitesse vertigineuse. L'interconnexion de ces réseaux à internet les a directement exposés aux menaces informatiques.

Ce projet ma permet d'approfondir mes connaissances, notamment en termes de configuration dans un environnement « Linux ». De plus j'ai aussi enrichi mes connaissances dans le domaine de la sécurité d'un réseau local, grâce à la mise en place d'un système de détection d'intrusions.

L'étude que nous avons menée nous a conduits à découvrir l'une des mesures de sécurité à déployer, pour assurer la sécurité d'un réseau informatique. Nous avons eu l'initiative de mettre en place un système de détection d'intrusions dans le but de renforcer la sécurité au sein d'un réseau LAN, il s'agit de l'IDS « Snort ».

Nous avons présenté l'aspect théorique sur la sécurité des réseaux informatique et toutes les notions qui s'y rapportent, l'aspect pratique quant à lui a fait l'objet d'implémentation de la solution proposée, suivi des différents tests d'évaluation réalisés, permettant de garantir le succès de la démarche de configuration, l'utilisation de cet outils dans un réseau est fonctionnel

En termes de perspectives; la mise en place de plugins qui facilite l'administration de l'IDS Snort tel que **SortSam** qui est un plugin de Snort qui fonctionne avec deux parties : le plugin pour Snort et un agent intelligent qui tourne comme un service sur le(s) firewall(s). Il permet de bloquer des adresses IP sur le pare-feu en analysant les alertes relevé par Snort et indiquant aux agents SnortSam les adresses que ces derniers fournissent au pare-feu afin de les bloquer. J'envisage aussi de mettre en place le plugin **PulledPork** qui permet de faire de façon automatique les mises à jour des règles de Snort.

[2] Laurent Poinsot «Introduction à la sécurité informatique», support de cours, Université Paris 13.

[3] les virus informatique clusif 2005, page 10

[4] Les virus et les spam, Page 37 (https://www.sophos.com/fr-fr/medialibrary/PDFs/case%20studies/fr/comviru_vrius_bfr.pdf?la=fr-FR)

[5] Philippe Biondi, Architecture expérimentale pour la détection d'intrusions dans un système informatique, Article de recherche, Avril-Sptembre 2001

[6] Laurent Bloch-Christophe Wolfhugel. Sécurité informatique .EYROLLES, 2eme edition. 2005.

[7] Le grand livre de la sécurité informatique. SecuriteInfo, Editions du 6 nouvembre 2006

[8] K.GHERBI, Réseaux virtuel privé.

[9] M.Tran Van Tay, le système de détection des intrusions et le système d'empêchement des intrusions (ZERO DAY), Rapport de stage de fin d'étude, institut de la francophonie pour l'informatique, université de Québec à Montréal, Féverier 2005.

[10] Hervé Debar, Benjamin Morin, Frédéric Cuppens, Fabien Autrel, Ludovic Mé, Bernard Vivinis Salem Benferhat, Mireille Ducassé, Rodolphe Ortalo, Détection d'intrusions : corrélation d'alertes. Article de synthèse, Caen, France, 2004.

[11] Cédric Michel, Langage de description d'attaques pour la détection d'intrusions par corrélation d'événements ou d'alertes en environnement réseau hétérogène, thèse de doctorat de l'Université de Rennes1,16 Décembre 2003.

[12] Tarek Abbes. (2004) « Classification du trafic et optimisation des règles de filtrage pour la détection d'intrusions ». Thèse de doctorat de l'université Henri Poincaré.Nancy1.2004

[13] Jonathan-Christofer Demay, Génération et évaluation de mécanisme de détection d'intrusions au niveau applicatif, Thèse de doctorat, école doctorale Matisse, université de Rennes1 Juillet 2011.

[14] Baudoin, Karle, NT Reseau, IDS et IPS , 2004

[15] Hadaoui Rebiha , un IDS basé sur un algorithme inspirer du fonctionnement de colonies de Fourmies, Mémoire de magistère, université M'Hamed Bougara de Boumerdes

[16] Yann Berthier, Jean-Baptiste Marchand, Détection d'intrusions et analyse forensique.

[17] Thierry Evangilista, Les IDS Les systèmes de détection d'intrusions informatiques édition DUNOD, Paris 2004.

[18] Jean-Marc ROYER, sécuriser l'informatique de l'entreprise, Enjeux, menaces, prévention et parades, Edition ENI

[W1] http://blog.octo.com/syn-flood/

[W2] https://www.securiteinfo.com/attaques/hacking/ddos.shtml

[W3]http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-fr-4/ch-detection.html

[W4]http://publib.boulder.ibm.com/tividd/td/TRM/SC23482300/en_US/HTML/adapter_guide06.html

[W5] https://doc.ubuntu-fr.org/barnyard2

[W6]http://repo.hackerzvoice.net/depot_madchat/reseau/ids%7Cnids/L'%E9criture%20de%20r%E8gles%20Snort.htm

Résumé:

Suite à notre étude sur la sécurité dans les réseaux Lan, on se rend bien compte que ce n'est pas facile d'assurer une sécurité à un réseau et de le protéger contre d'éventuelles intrusions.

L'évolution des outils permettant de réaliser des attaques informatiques a mis les différents réseaux informatiques en danger. Avoir un réseau complètement sécurisé est pratiquement irréalisable. Par conséquent, il est nécessaire de pouvoir détecter les intrusions lorsqu'elles se produisent. Cela est rendu possible grâce aux mécanismes de détection d'intrusions. La détection d'intrusions consiste à découvrir l'utilisation d'un système informatique à des fins non légales en émettant une alerte.

Nous avons réalisé l'installation et la configuration de SNORT qui est un système de détection d'intrusions particulièrement répondu car fourni en open source, outre sa gratuité, son avantage est qu'il dispose d'une très grosse base de signatures réalisée par la communauté des utilisateurs.

Mots-clés : Détection d'intrusion, Snort, Barnyard2, IDS.

Abstract

Following our study of security in LAN networks, one realizes that it is not easy to provide security to a network and protect against possible intrusions.

The development of tools for performing computer attacks has different computer networks at risk. Having a completely secure network is virtually impossible. Therefore, it is necessary to detect intrusions when they occur. This is made possible thanks to the intrusions detection mechanisms. Intrusions detection is to discover the use of a computer system to no-legal purposes by issuing a warning.

We completed the installation and configuration of SNORT is an intrusion detection system responded particularly as provided in open source, besides its free, its advantage is that it has a very large signature database conducted by the user community.

Keywords: Intrusion detection**,** Snort, Barnyard2, IDS.