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

Mémoire de Master en Informatique

Option

Administration et sécurité des réseaux informatiques

Thème

Détection et Prévention des Intrusions dans un Système Informatique en utilisant SNORT

Présenté par :

Fawzi BAKOURI.

Youcef OUARI

Encadreurs :

Mr Abderrahmane BAADACHE.

Membre du jury :

DR Mawloud OMAR

Président du jury :

Mr Kamel KABYL.

Juin 2012.

Remerciements

Nos vifs remerciements sont adressés :

En premier lieu à nos parents pour nous avoir soutenus et pour tous les sacrifices consentis, ainsi qu'à tous nos proches et amis.

A notre encadreur pour nous avoir guidé tout au long de ce projet.

Aux membres de la commission pour avoir jugé notre modeste travail.

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

Table des matières

Liste des figures ..... **iv**

Introduction générale **1**

Chapitre I : La sécurité informatique **2**

I Introduction 2

I.1 Qu'est-ce que la sécurité ? 2

I.2 Mise en place d'une politique de sécurité 2

I.2.1 Etude des risques 2

I.2.2 Différents aspects de la sécurité 3

I.2.3 Services de la sécurité (Objectifs) 3

I.2.4 Principaux défauts de la sécurité informatique 4

I.3 Attaques informatiques 4

I.3.1 Anatomie d'une attaque 4

I.3.2 Cyber-attaques 5

I.3.4 Différents types d'attaques 6

I.3.4.1 Attaques réseaux 6

I.3.4.2 Attaques applicatives 7

I.4 Démarches pour anticiper et résoudre les problèmes 9

I.4.1 Les firewalls 9

I.4.2 Proxy 10

I.4.3 DMZ (Demilitarized zone) 10

I.4.4 Antivirus 11

I.4.5 NIPS (Systèmes de prévention d'intrusions « réseau ») 11

I.5 Conclusion 12

Chapitre II : Systèmes de détection et prévention des intrusions **14**

II Introduction 14

II.1 Systèmes de détections d'intrusions (IDS) 14

II.1.1 Principes de fonctionnement des IDS 14

II.1.1.1 Principes de détection des intrusions 15

II.1.1.1.1 Approche par scénario ou signature 16

II.1.1.1.2 Approche comportementale (Anomaly Detection) 18

II.2 Différents types d'IDS 18

II.2. 1 Systèmes de détections d'intrusions « Réseau » (NIDS) 19
II.2.2 Systèmes de détections d'intrusions de type « hôte » (HIDS) 19
II.2.3 Systèmes de détection d'intrusions « Hybrides » 19
II.3 Systèmes de prévention d'intrusions (IPS) 19
II.3 Systèmes de prévention d'intrusions « Réseau » (NIPS) 20
II.3.1 Définition 20
II.3.2 Fonctionnement d'un NIPS 20
II.3.3 Systèmes de prévention d'intrusions « Kernel » (KIPS) 21
II.4 Inconvénients des IDS/IPS 21
II.4.1 Besoin de connaissances en sécurité 21
II.4.2 Faux positifs et faux négatifs 21
II.4.3 Pollution/surcharge 22
II.4.4 Consommation de ressources 22
II.4.5 Perte de paquets (limitation des performances) 22
II .5 Conclusion 22
Chapitre III : Mise en œuvre d'un NIPS (Snort/SnortSam) 25
III Introduction 25
III.1 Présentation générale de Snort 25
III.1.2 : Positionnement de SNORT dans le réseau 25
III.1.3 Architecture de Snort 26
III.1.4 Environnement 27
III.1.5 Installation de Snort 28
III.1.6 Modes de fonctionnement 28
III.1.6.1 Le mode « NIDS» 28
III.1.7 Paramétrage de Snort 28
III.1.7.1 Pré-processeurs 28
III.1.7.2 Les plugins de sortie 29
III.1.7.3 Les règles de Snort (Rules) 29
III.1.7.3.1 Création de règles 29
III.1.7.3.2 Mise à jour de règles (rules) 31
III.2 Mise en place de Barnyard2 31
III.2.1 Le plugin « Unified 2 » 31
III.3 La console B.A.S.E 31
III.4 SnortSam 32

III.4.1 présentation 32
III.4.2 Configuration de règles de SNORT (fwsam) 33
III.4.2.1 Fwsam 33
III.5 IPtables/Netfilter 34
III.6 architecture applicative 35
III.7 Outils d'attaques (Anonymous OS) 36
III.8 Conclusion 37
Chapitre IV : Installation de Snort/SnortSam 39
IV Introduction 39
IV.1 Installation de Snort 39
IV.1.2 Lancement de Snort 39
IV.1.3 Modes de fonctionnement 39
IV.1.3.1 Le mode écoute (Sniffer mode) 39
IV.1.3.2 Le mode « detection d'intrusion NIDS » 41
IV.1.4 Mise en œuvre de la base de données MySql 45
IV.1.4.1 Installation 45
IV.1.4.2 Création de la base de données pour Snort 46
IV.1.4.3 Installation de Snort-Mysql 47
IV.1.5 Mise en place de la console B.A.S.E 50
IV.1.6 Mise en place de barnyard 59
IV.1.6.1 installation de barnyard 59
IV.7 Extension de Snort en NIPS 64
IV.7.1 Application du patch SnortSam a Snort 64
IV.7.2 Mise en place de l'agent SnortSam 66
IV.7.2.1 Installation de l'agent SnortSam 66
IV.7.2.2 Configuration de SnortSam 69
IV.7.4 configuration des règles 71
IV.8 Lancement d'attaques 72
IV.8.2 Scan du réseau 'nmap' 74
IV.9 Conclusion 76
***Conclusion générale* 78**
***Bibliographie* 80**

Liste des figures

Figure 1 : Evolution des différents types d'attaques au fil des années. 6
Figure 2 : Schéma illustrant l'attaque man in the middle. 9
Figure 3 : Positionnement de la DMZ dans un réseau informatique. 11
Figure 4 : Principes de fonctionnement d'un IDS . 15
Figure 5 : Principe de détection par un IDS basé sur une base de signature. 17
Figure 6 : Principes de fonctionnement d'un NIPS . 21
Figure 7 : Différentes positions possibles de Snort dans un réseau Informatique. 26
Figure 8 : Fonctionnement de Snort. 27
Figure 9 : Format d'une règle de Snort. 29
Figure 10 : Schéma généralisant l'architecture de l'ensemble Snort/SnortSam 33
Figure 11 : Architecture généralisant le fonctionnement du NIPS (Snort/SnortSam) 35
Figure 12 : Aperçu des outils d'attaque. 36

Introduction générale

Introduction générale

L'évolution de la technologie a beaucoup amélioré la plupart des plateformes de sécurité existantes dans les réseaux informatiques. En effet depuis l'avènement des premiers réseaux IP, les problèmes de sécurité se sont diversifiés et ont conduit au développement de nouvelles techniques de sécurité.
Du cryptage à l'implémentation d'une architecture de sécurité et la mise en place des firewalls (logiciels ou matériels), les réseaux IP ont connu une amélioration au niveau de l'aspect sécurité.

Mais la Sécurité à cent pour cent n 'a jamais et ne sera sans doute jamais atteinte, car dans toute plateforme de sécurité existante, on peut y déceler une faille plus ou moins facile à exploiter. Parmi les outils de sécurité qui ont étaient inventés, on trouve les systèmes de détection et de prévention d'intrusions (IDS/IPS), leur but est de complémenter les firewalls et autres outils de sécurité afin de limiter au maximum ces failles de sécurité.

Les IDS/IPS existent sous forme matérielle (appliance IBM) qui sont couteux est destinés aux gros réseaux informatiques, aussi, ils existent sous forme logicielle comme SNORT, software open source sous linux et qui possède une équipe de développement très compétente, idéal pour les petits réseaux et petites entreprises a budget limité. Il fait l'objet de notre étude dans le cadre de ce projet.

Dans la première partie, nous ferons l'étude de la sécurité informatique et de l'importance de la mise en place d'une politique de sécurité au niveau d'un réseau informatique et de ses objectifs, et cela nous mènera forcément a parler des menaces que subi un réseau informatique sous forme d'attaques de différents types et nous verrons aussi quelques solutions possibles pour remédier a cela.

La Deuxième partie sera consacrée à la présentation des IDS/IPS, leurs emplacements au sein du réseau, les différents types de ces derniers. Nous verrons aussi leur principe de fonctionnement, les différences existantes entre chacun d'entre eux et enfin nous verrons comme n'importe quelle solution informatique, quelques inconvénients que possèdent ces solutions.

La troisième partie (implémentation) sera réservée à l'illustration de l'outil IDS/IPS qui est SNORT : software open source sous Linux que nous avons proposé comme solution. Dans cette partie nous détaillerons les modules et dépendances menant à l'installation de Snort, sa position dans le réseau, sont architecture et nous verrons aussi comment le paramétrer. Nous verrons également comment étendre Snort d'un simple système de détection en système de prévention d'intrusion.

La quatrième et dernière partie sera consacrer a 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 sont comportement.

Partie théorique

Chapitre I : La sécurité informatique

I Introduction

Avec le développement de l'utilisation d'Internet, de plus en plus d'entreprises ouvrent leur système d'information à leurs partenaires ou leurs fournisseurs, il est donc essentiel de connaître les ressources de l'entreprise à protéger et de maîtriser le contrôle d'accès et les droits des utilisateurs du système d'information. Il en va de même lors de l'ouverture de l'accès de l'entreprise sur Internet. Dans le but de minimiser les risques d'attaques et de vol d'informations, il est indispensable d'établir au sein de l'entreprise une politique de sécurité afin de protéger la confidentialité de ses données au maximum possible.

I.1 Qu'est-ce que la sécurité ?

La notion de sécurité informatique couvre l'ensemble des moyens outils, techniques et méthodes pour garantir que seules les personnes ou autres systèmes autorisés interviennent sur le système et ont accès aux données, sensibles ou non. Avec l'ouverture des systèmes informatiques sur l'extérieur et le rôle de support stratégique tenu désormais par ces derniers, la sécurité est un thème majeur bien trop rarement considéré à sa juste valeur.

I.2 Mise en place d'une politique de sécurité

La mise en œuvre d'une politique de sécurité globale est assez difficile, essentiellement par la diversité des aspects à considérer. Une politique de sécurité peut se définir par un certain nombre de caractéristiques : les niveaux où elle intervient, les objectifs de cette politique et enfin les outils utilisés pour assurer cette sécurité. Chaque aspect différent doit être pris en compte, de façon à atteindre les objectifs de sécurité désirés, en utilisant de façon coordonnée les différents outils à disposition.

Nous allons tout d'abord étudier les risques potentiels puis, les différents aspects d'une politique de sécurité, avant de définir les objectifs visés, puis de voir les outils disponibles pour appliquer cette politique.

I.2.1 Etude des risques

Les coûts d'un problème informatique peuvent être élevés et ceux de la sécurité le sont aussi. Il est nécessaire de réaliser une analyse de risque en prenant soin d'identifier les problèmes potentiels avec les solutions avec les coûts associés. L'ensemble des solutions retenues doit être organisé sous forme d'une politique de sécurité cohérente, fonction du niveau de tolérance au risque. On obtient ainsi la liste de ce qui doit être protégé. Il faut cependant prendre conscience que les principaux risques restent : « câble arraché », « coupure secteur », « crash disque », « mauvais profil utilisateur », …

Voici quelques éléments pouvant servir de base à une étude de risque:

• Quelle est la valeur des équipements, des logiciels et surtout des informations ?

• Quel est le coût et le délai de remplacement ?

• Faire une analyse de vulnérabilité des informations contenues sur les ordinateurs en réseau (programmes d'analyse des paquets, logs…).

• Quel serait l'impact sur la clientèle d'une information publique concernant des intrusions sur les ordinateurs de la société ? [1]

I.2.2 Différents aspects de la sécurité

Une politique de sécurité s'élabore à plusieurs niveaux.
On va tout d'abord sécuriser l'accès aux données de façon logicielle (authentification, contrôle d'intégrité). On va également sécuriser l'accès physique aux données : serveurs placés dans des salles blindées (qui empêchent les ondes électromagnétiques d'être captées) 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 leur mot de passe 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.

I.2.3 Services de la sécurité (Objectifs)

Les objectifs d'une politique de sécurité sont de garantir la sécurité des informations et du réseau de l'entreprise. Ces impératifs peuvent être définis à plusieurs niveaux :

- Disponibilité : les données doivent rester accessibles aux utilisateurs (une attaque de type DoS, par exemple, vise à empêcher les utilisateurs normaux d'un service d'y accéder)

- Confidentialité : les données ne doivent être lisibles que des personnes habilitées pour.

- Intégrité : il faut pouvoir garantir que les données protégées n'ont pas été modifiées par une personne non autorisée.

- Non répudiation : on doit pouvoir certifier, quand un fichier a subi des modifications, la personne qui l'a modifié.

L'authentification : consistant à assurer que seules les personnes autorisées aient accès aux ressources.

I.2.4 Principaux défauts de la sécurité informatique

Les défauts de sécurité d'un système d'information les plus souvent constatés sont :

• Installation des logiciels et matériels par défaut.

• Mises à jour non effectuées.

• Mots de passe inexistants ou par défaut.

• Services inutiles conservés (Netbios…).

• Traces inexploitées.

• Pas de séparation des flux opérationnels des flux d'administration des systèmes.

• Procédures de sécurité obsolètes.

• Eléments et outils de test laissés en place dans les configurations en production.

• Authentification faible.

• Télémaintenance sans contrôle fort.

I.3 Attaques informatiques

Tout ordinateur connecté à un réseau informatique est potentiellement vulnérable à une attaque.

Une « attaque » est l'exploitation d'une faille d'un système informatique (système d'exploitation, logiciel ou bien même de l'utilisateur) à des fins non connues par l'exploitant du système et généralement préjudiciables.

Sur Internet, des attaques ont lieu en permanence, à raison de plusieurs attaques par minute sur chaque machine connectée. Ces attaques sont pour la plupart lancées automatiquement à partir de machines infectées (par des virus, chevaux de Troie, vers, etc.), à l'insu de leur propriétaire. Plus rarement il s'agit de l'action de pirates informatiques.

Afin de contrer ces attaques il est indispensable de connaître les principaux types d'attaques afin de mettre en œuvre des dispositions préventives.

I.3.1 Anatomie d'une attaque

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

· **Probe :** consiste en la collecte d'informations par le biais d'outils comme whois, Arin,

DNS lookup. La collecte d'informations sur le système cible peut s'effectuer de plusieurs manières, comme par exemple un scan de ports grâce au programme Nmap pour déterminer la version des logiciels utilisés, ou encore un scan de vulnérabilités à l'aide

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

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

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

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

· **Paralyze :** cette étape peut consister en plusieurs actions. Le pirate peut utiliser le serveur pour mener une attaque sur une autre machine, détruire des données ou encore endommager le système d'exploitation dans le but de planter le serveur. Après ces cinq étapes, le pirate peut éventuellement tenter d'effacer ses traces, bien que cela ne soit rarement utile. En effet, les administrateurs réseaux sont souvent surchargés de logs à analyser. De plus, il est très difficile de supprimer entièrement des traces.

I.3.2 Cyber-attaques

On appelle "cyber-attaque" une tentative d'atteinte à des systèmes informatiques réalisée dans un but malveillant. Elle peut avoir pour objectif de voler des données (secrets militaires, diplomatiques ou industriels, données personnelles, bancaires, etc.), de détruire, endommager ou altérer le fonctionnement normal des dispositifs informatiques, de prendre le contrôle des processus informatiques, ou de tromper les dispositifs d'authentification pour effectuer des opérations illégitimes.

Les dispositifs informatiques ciblés par ces attaques sont des ordinateurs ou des serveurs, isolés ou en réseaux, reliés ou non à internet, des équipements périphériques tels que les imprimantes, ou des outils communicants comme les téléphones mobiles ou les assistants personnels. [3]

Voici un schéma illustrant l'évolution de la cyber-attaque au fil du temps selon le F.B.I, ou on remarque l'évolution des outils d'attaques au fil du temps par rapport aux connaissances techniques requises pour les acteurs.

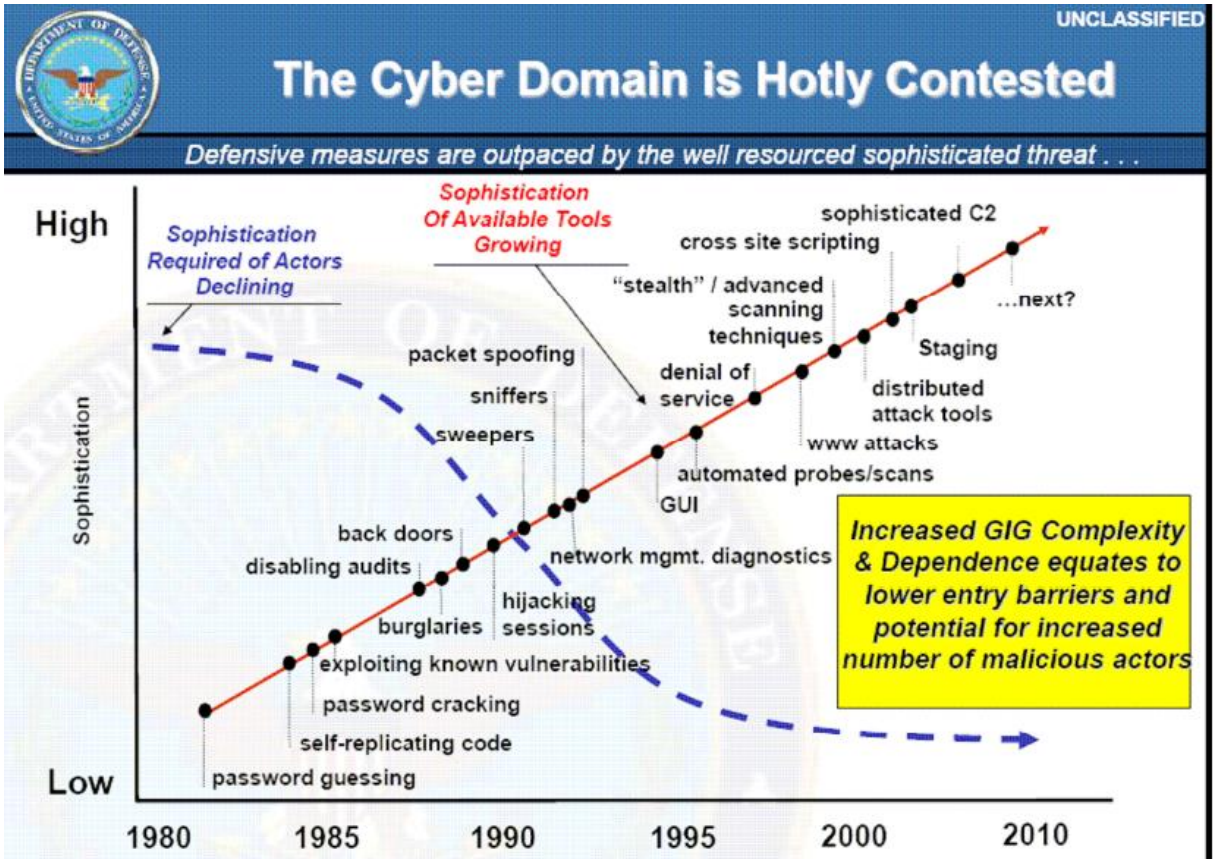


Figure 1 : Evolution des différents types d'attaques au fil des années.

I.3.4 Différents types d'attaques

Il existe un grand nombre d'attaques permettant à une personne mal intentionnée de s'approprier des ressources, de les bloquer ou de les modifier. Certaines requièrent plus de compétences que d'autres, en voici quelques unes :

I.3.4.1 Attaques réseaux

Ce type d'attaque se base principalement sur des failles liées aux protocoles ou à leur implémentation. Observons quelques attaques bien connues :

- **Techniques de scan**

  Les scans de ports ne sont pas des attaques à proprement parler. Le but des scans est de déterminer quels sont les ports ouverts, et donc en déduire les services qui sont exécutés sur la machine cible (ex : port 80/TCP pour un service HTTP). Par conséquent, la plupart des attaques sont précédées par un scan de ports lors de la phase Probe qui est comme nous l'avons vu, la première phase des 5P's dans le déroulement d'une attaque.

Il existe un nombre important de techniques de scan. Idéalement, la meilleure technique de scan est celle qui est la plus furtive afin de ne pas alerter les soupçons de la future victime.

IP Spoofing

Cette a attaque consiste a usurper l'adresse IP d'une autre machine via les datagrammes IP.

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

Déroulement : il existe des utilitaires qui permettent de modifier les paquets IP ou de créer ses propres paquets (ex : hping2). Grâce à ces utilitaires, il est possible de spécifier une adresse. [2]

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

L'ARP spoofing,ou ARP poisoning, est une attaque du réseau local (802.3, 802.11) en utilisant le protocole ARP ,cette technique peut permettre à l'attaquant de détourner les flux de communication transitant sur un réseau local commuté, lui permettant de les écouter, de les corrompre ou de bloquer le trafic.
Cette attaque se fait en envoyant un paquet ARP forgé par l'attaquant C vers une machine A. La machine A envoie ses paquets à l'attaquant C, alors qu'ils étaient destinés à la victime B. De même, l'attaquant C envoie un paquet ARP forgé vers la victime B afin qu'elle envoie ses paquets à l'attaquant C au lieu de les envoyer à la machine A.
Enfin, l'attaquant doit router les paquets de A vers B et inversement pour que la connexion entre la machine A et la victime puisse continuer.
L'attaquant, en détournant le flux, peut ainsi voir les données qui transitent en clair entre les deux machines.

- **DNS ID Spoofing**

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

Remarque : Une attaque que nous allons voir ci-après, le Déni de Service, peut aider à ralentir le trafic du serveur DNS et ainsi permettre de répondre avant lui. [2]

I.3.4.2 Attaques applicatives

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

- **Buffers overflow**

  Les buffers overflows, ou dépassement de la pile, sont une catégorie de bug particulière. Issus d'une erreur de programmation, ils permettent l'exploitation d'un shellcode à distance.

  Ce shellcode permettra à une personne mal intentionnée d'exécuter des commandes sur le système distant, pouvant aller jusqu'à sa destruction.

  L'erreur de programmation est souvent la même : la taille d'une entrée n'est pas vérifiée et l'entée est directement copiée dans un buffer dont la taille est inférieure à la taille de l'entrée. On se retrouve donc en situation de débordement, et l'exploitant peut ainsi accéder à la mémoire.

- **Scripts**

  Principalement web (ex : Perl, PHP, ASP), ils s'exécutent sur un serveur et renvoie un résultat au client. Cependant, lorsqu'ils sont dynamiques (i.e. qu'ils utilisent des entrées saisies par un utilisateur), des failles peuvent apparaître si les entrées ne sont pas correctement contrôlées. L'exemple classique est l'exploitation de fichier à distance, tel que l'affichage du fichier mot de passe du système en remontant l'arborescence depuis le répertoire web.

- **SQL injections**

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

- **Man in the middle**

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

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

La figure suivante nous montre l'emplacement de l'intrus au niveau du réseau local dans le but d'usurper la victime.

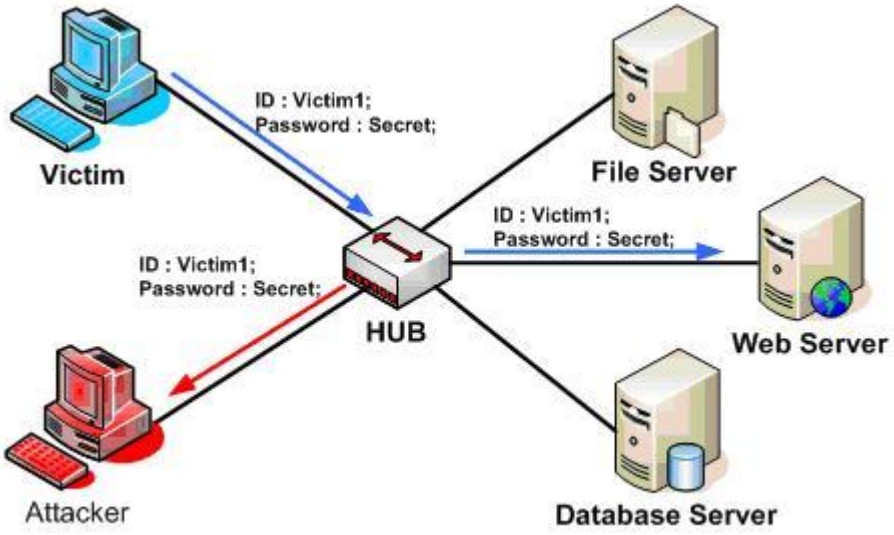


Figure 2 : Schéma illustrant l'attaque *man in the middle*.

- **Déni de service**

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

I.4 Démarches pour anticiper et résoudre les problèmes

Beaucoup d'outils, techniques et stratégies sont développés afin de remédier à l'ensemble des menaces que subit un réseau informatique.
Selon le besoin, les menaces existantes et la sensibilité des données à protéger, un administrateur réseau dois faire la bonne combinaison entre les différentes solutions software et hardware proposées jusqu'à présent.

Voici quelques solutions suggérées dont notre proposition IDS/IPS :

I.4.1 Les firewalls

Les firewalls ne sont pas des IDS à proprement parler mais ils permettent également de stopper des attaques. Nous ne pouvions donc pas les ignorer.

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

Il existe trois types de firewalls :

- **Les systèmes à filtrage de paquets sans état :** analyse les paquets les uns après les autres, de manière totalement indépendante.

- **Les systèmes à maintien d'état (stateful) :** vérifient que les paquets appartiennent à une session régulière. Ce type de firewall possède une table d'états où est stocké un suivi de chaque connexion établie, ce qui permet au firewall de prendre des décisions adaptées à la situation. Ces firewalls peuvent cependant être outrepassés en faisant croire que les paquets appartiennent à une session déjà établie.

- **Les firewalls de type proxy :** Le firewall s'intercale dans la session et analyse l'information afin de vérifier que les échanges protocolaires sont conformes aux normes.

I.4.2 Proxy

Le proxy est un ordinateur faisant office de passerelle entre le réseau d'un particulier ou d'une entreprise et Internet.

Il a deux utilités :

Il fait office de firewall pour tout le réseau, ce qui en fait une sécurité de plus en cas d'attaque, il sert de mémoire cache, téléchargeant les pages web visitées par les utilisateurs du réseau local, ce qui permet de les exécuter ensuite à partir du proxy et non du serveur distant qui héberge le site.

I.4.3 DMZ (Demilitarized zone)

Lorsque certaines machines du réseau interne ont besoin d'être accessibles de l'extérieur (serveur web, un serveur de messagerie, un serveur FTP public, etc.), il est souvent nécessaire de créer un nouveau sous-réseau contenant ces services, accessible aussi bien du réseau interne que de l'extérieur, sans pour autant risquer de compromettre la sécurité de l'entreprise. On parle ainsi de « zone démilitarisé » (notée DMZ pour DeMilitarized Zone) pour désigner cette zone isolée hébergeant des applications mises à disposition du public. La DMZ fait ainsi office de « zone tampon » entre le réseau à protéger et le réseau.

Cette figure illustre l'emplacement de la DMZ au niveau d'un réseau local :

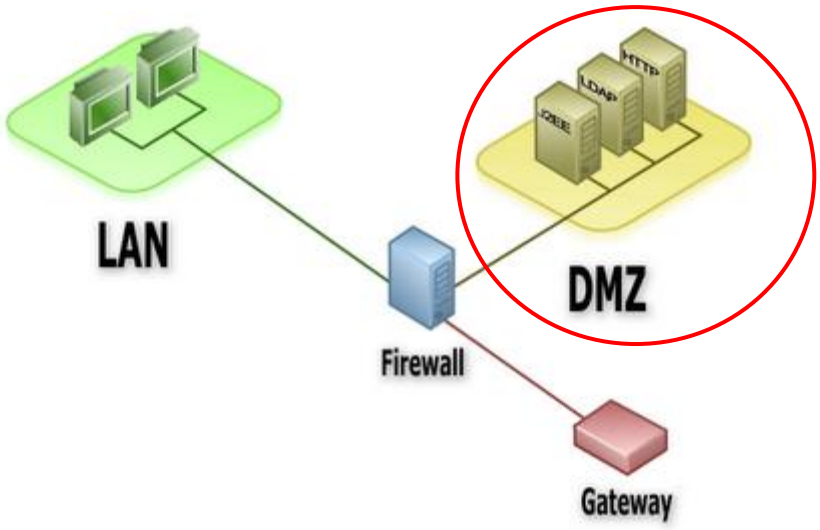


Figure 3 : Positionnement de la DMZ dans un réseau informatique.

I.4.4 Antivirus

Logiciels permettant de détecter, neutraliser et/ou supprimer les virus informatiques et autres programmes malicieux. La détection se fait selon deux principes : une analyse par signatures qui permet de détecter les virus connus pour peu que les définitions de virus soient régulièrement mises à jour, ou une analyse heuristique qui permet de détecter avec des résultats variables les virus inconnus à partir de leur logique de programmation et le cas échéant de leur comportement à l'exécution. Les antivirus fonctionnent eux-mêmes selon deux principes : un scanner qui permet à l'utilisateur de lancer une analyse d'un disque ou d'un fichier lorsqu'il le souhaite ("on demand"), ou un moniteur qui surveille le système en temps réel ("on access") et empêche l'utilisateur d'ouvrir un fichier infecté.

I.4.5 NIPS (Systèmes de prévention d'intrusions « réseau »)

Le NIPS, en plus de détecter les intrusions sur le réseau, peut ainsi réagir automatiquement en supprimant les paquets, relancer la connexion et bloquer des ports grâce à sa communication avec le firewall.

Un NIPS assure une surveillance continue du flux circulant dans le réseau, d'où sont importance et son indispensabilité, pour cela nous allons consacrer tout un chapitre afin d'illustrer sa mise en place, ces fonctionnalités et ses atouts.

I.5 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 formes d'attaques informatiques, que nous avons illustré quelques-unes dans le but de montrer l'intensité du danger. Enfin, nous avons proposé quelques solutions existantes afin de se protéger et réduire les risques.

Chapitre II : Systèmes de détection et de prévention des intrusions.

II Introduction

Suite aux nombreuses menaces ciblant les réseaux informatiques, plusieurs contre-mesures ont été développées, dont les systèmes de détection et de prévention d'intrusions (IDS/IPS). Cette solution fera l'objet de notre étude. A coté des solutions connues comme les firewalls et les anti-virus, les IDS/IPS se font de plus en plus indispensables, pour cela, nous allons essayer d'expliquer leur principe de fonctionnement ainsi les différentes formes que peut avoir cette solution et les atouts qu'elle peut procurer en complétant les solutions citées auparavant.

II.1 Systèmes de détections d'intrusions (IDS)

Définition : Un IDS (Intrusion Detection System) est un ensemble de composants logiciels et matériels dont la fonction principale est de détecter des activités anormales ou suspectes sur la cible analysée (un réseau ou un hôte). Il permet ainsi d'avoir une action de prévention et d'intervention sur les risques. Afin de détecter les attaques que peut subir un système (réseau informatique), il est nécessaire d'avoir un logiciel spécialisé dont le rôle est de surveiller les données qui transitent sur ce système, et qui est capable de réagir si des données semblent suspectes.

II.1.1 Principes de fonctionnement des IDS

La détection repose sur des techniques de sondage (une sonde permet la capture de paquets), des tentatives de compromission de systèmes, d'activités suspectes internes, des activités virales ou encore audit des fichiers de journaux (logs).

Le fonctionnement d'un IDS repose sur 4 principes :

- Le principe de détection que nous allons détailler ultérieurement.
- Le comportement de l'IDS après détection.
- Les sources de données.
- La fréquence d'utilisation de l'IDS. [4]

Voici un schéma illustrant ces principes :

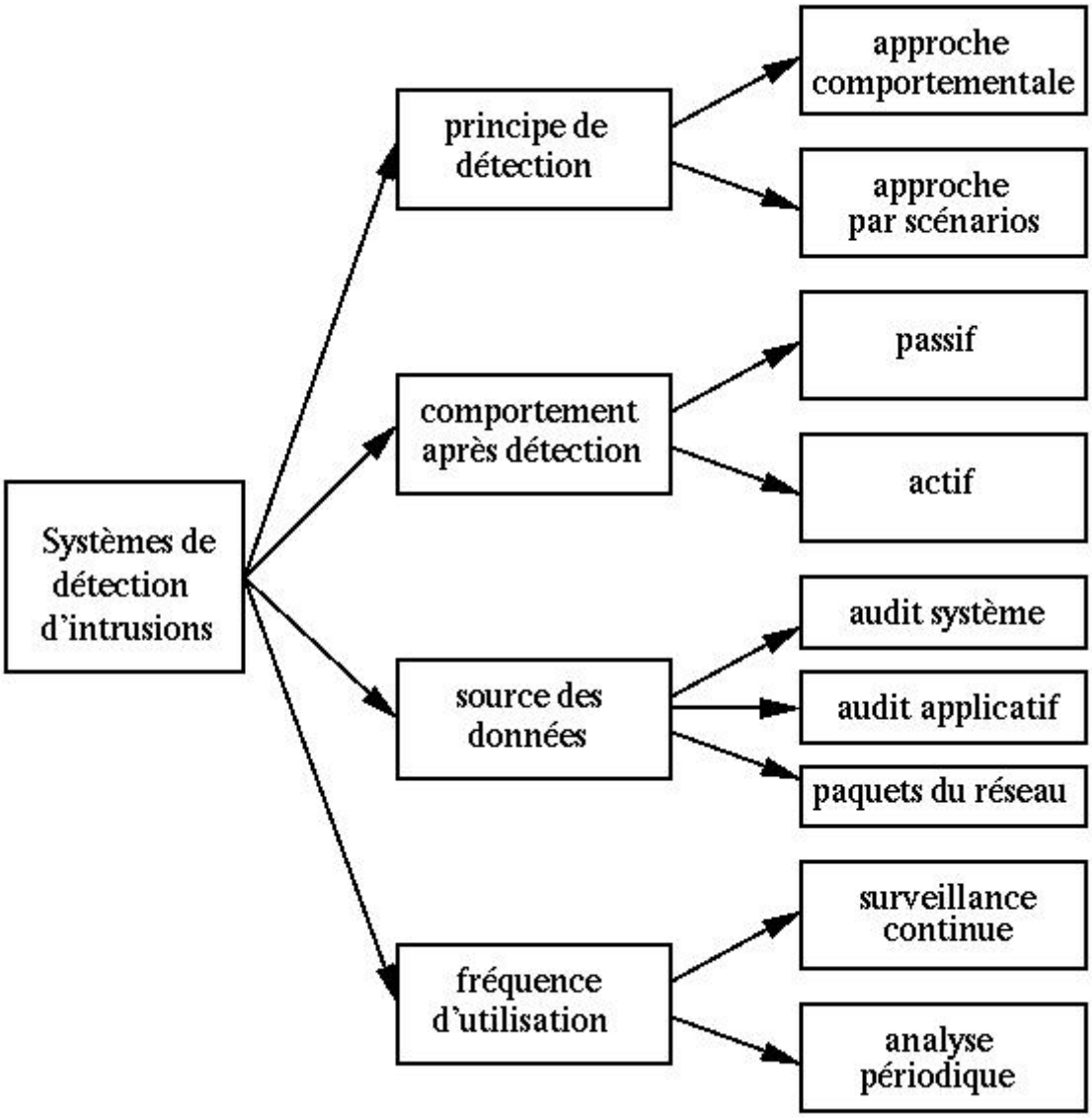


Figure 4 : Principes de fonctionnement d'un IDS .

II.1.1.1 Principes de détection des intrusions

Pour bien gérer un système de détection d'intrusions, il est important de comprendre comment celui-ci fonctionne. Des questions se posent alors :

- Comment reconnaître/définir une intrusion?
- Comment une intrusion est-elle détectée par un tel système ?
- Quels critères différencient un flux contenant une attaque d'un flux normal ?
- Etc.

Ces questions nous ont amené à étudier le fonctionnement interne des IDS.

Il existe plusieurs méthodes permettant de détecter une intrusion :

- La première consiste à détecter des signatures d'attaques connues dans les paquets circulant sur le réseau : **l'Approche par scénario ou par signature.**

- La seconde, consiste quant à elle, à détecter une activité suspecte dans le comportement de l'utilisateur : **l'Approche comportementale ou par Anomalie.**

Ces deux techniques, aussi différentes soient-elles, peuvent être combinées au sein d'un même système afin d'accroître la sécurité.

II.1.1.1.1 Approche par scénario ou signature

Cette technique s'appuie sur la connaissance des techniques utilisées par les attaquants pour déduire des scénarios typiques. Elle ne tient pas compte des actions passées de l'utilisateur et utilise des signatures d'attaques existantes (ensemble de caractéristiques permettant d'identifier une activité intrusive : une chaîne alphanumérique, une taille de paquet inhabituelle, une trame formatée de manière suspecte, …).

Cette technique se base sur :

- **La recherche de motifs (pattern matching)**

C 'est la méthode la plus connue et la plus facile à comprendre. Elle se base sur la recherche de motifs (chaînes de caractères ou suite d'octets) au sein du flux de données. L'IDS comporte une base de signatures où chaque signature contient les protocoles et ports utilisés par une attaque spécifique ainsi que le motif qui permettra de reconnaître les paquets suspects.

Le principal inconvénient de cette méthode est que seules les attaques reconnues par les signatures seront détectées ! Il est donc nécessaire de mettre à jour régulièrement la base de signatures de l'IDS. Un autre inconvénient est que les motifs sont en général fixes. Or une attaque n'est pas toujours identique à 100%. Le moindre octet différent par rapport à la signature provoquera le non détection de l'attaque.
Pour les IDS utilisant cette méthode, il est nécessaire d'adapter la base de signatures en fonction du système à protéger. Cela permet non seulement de diminuer les ressources nécessaires et donc augmenter les performances ; mais également réduire considérablement le nombre de fausses alertes et donc faciliter le travail des administrateurs réseaux qui analyseront les alertes. [5]
De manière analogue, cette technique est également utilisée dans les anti-virus. En effet un anti-virus ne peut reconnaître un virus que si ce dernier est reconnu dans sa base de signatures virale, d'où la mise à jour régulière des anti-virus.

Cette figure montre le principe de fonctionnement de l'IDS en comparant les paquets capturés avec les signatures déjà existantes afin de générer une alerte.

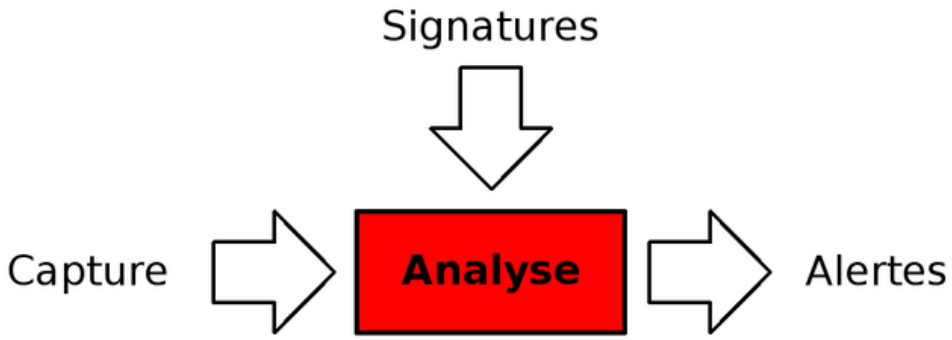


Figure 5 : Principe de détection par un IDS basé sur une base de signature.

- **Recherche de motifs dynamiques**

  Le principe de cette méthode est le même que précédemment mais les signatures des attaques évoluent dynamiquement. L'IDS est de ce fait doté de fonctionnalités d'adaptation et d'apprentissage.

- **Analyse de protocoles**

  Cette méthode se base sur une vérification de la conformité (par rapport aux RFC) des flux, ainsi que sur l'observation des champs et paramètres suspects dans les paquets. L'analyse protocolaire est souvent implémentée par un ensemble de préprocesseurs (programmes ou plug-in), où chaque préprocesseur est chargé d'analyser un protocole particulier (FTP, HTTP, ICMP, …). Du fait de la présence de tous ces préprocesseurs, les performances dans un tel système s'en voient fortement dégradées (occupation du processeur).

  L'intérêt fort de l'analyse protocolaire est qu'elle permet de détecter des attaques inconnues, contrairement au pattern matching qui doit connaître l'attaque pour pouvoir la détecter. [2]

- **Analyse heuristique et détection d'anomalies**

Le but de cette méthode est, par une analyse intelligente, de détecter une activité suspecte ou toute autre anomalie (une action qui viole la politique de sécurité définie dans l'IDS).

Par exemple : une analyse heuristique permet de générer une alarme quand le nombre de pings vers un réseau ou hôte est très élevé ou incessant (Ping de la mort).

II.1.1.1.2 Approche comportementale (Anomaly Detection)

Cette technique consiste à détecter une intrusion en fonction du comportement passé de l'utilisateur. Il faut préalablement dresser un profil utilisateur à partir de ses habitudes et déclencher une alerte lorsque des événements hors profil se produisent.

Cette technique peut être appliquée non seulement à des utilisateurs mais aussi à des applications et services. Plusieurs métriques (paramètres) sont possibles : la charge CPU, le volume de données échangées, le temps de connexion sur des ressources, la répartition statistique des protocoles et applications utilisés, les heures de connexion, … Cependant elle possède quelques inconvénients :

• **Peu fiable :** tout changement dans les habitudes de l'utilisateur provoque une alerte.

• **Nécessite une période de non fonctionnement pour mettre en œuvre les mécanismes d'auto-apprentissage :** si un pirate attaque pendant ce moment, ses actions seront assimilées à un profil utilisateur, et donc passeront inaperçues lorsque le système de détection sera complètement mis en place.

• **L'établissement du profil doit être souple afin qu'il n'y ait pas trop de fausses alertes :** le pirate peut discrètement intervenir pour modifier le profil de l'utilisateur afin d'obtenir après plusieurs jours ou semaines, un profil qui lui permettra de mettre en place son attaque sans qu'elle ne soit détectée.

Il existe des IDS qui utilisent ces deux approches à la fois, utilisant les algorithmes et techniques citées précédemment. En pratique ces techniques sont définies dans les règles des IDS. Ces règles se trouvent dans des fichiers de configuration spécifiques, peuvent être personnalisées afin de mieux répondre aux exigences de la politique de sécurité mise en place par l'administrateur réseau. [6]

II.2 Différents types 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…

Il existe plusieurs types d'IDS, mais on peut les classer en deux familles :

- **Les NIDS** : Network IDS, système de détection d'intrusion réseau
- **Les HIDS** : Host IDS, système de détection d'intrusion de type hôte. [7]

Les autres IDS sont en réalité des dérivées de ces familles : les IDS Hybrides, les IPS (systèmes de prévention d'intrusions).

Les IDS sont disponibles sous deux formats :

- **Les logiciels** : permettent à n'importe quel administrateur de réseau de l'installer sur son OS. Ils sont faciles à installer, configurer et contrôler. Cependant, cet OS (Windows, Linux) est une distribution dont les failles peuvent être connues des

hackers . Il est plus vulnérable si les patchs (programmes de mise à jour) ne sont pas régulièrement installés et que les modules inutilisés de l'OS sont conservés.

- **Les appliances** : sont des « boîtes noires » dédiées (solutions commerciales), connectées au réseau et implémentées d'un OS propriétaire, d'un firewall et des interfaces réseaux requises. Le système d'exploitation est propriétaire, sécurisé et peu connu des hackers.

Au côté sécuritaire de l'Appliance, il faut ajouter les meilleures performances obtenues à son usage : le Hardware et le logiciel ont été conçus spécifiquement pour cette utilisation.

II.2. 1 Systèmes de détections d'intrusions « Réseau » (NIDS)

Objectif : analyser de manière passive les flux en transit sur le réseau et détecter les intrusions en temps réel.
Un NIDS écoute donc tout le trafic réseau, puis l'analyse et génère des alertes si des paquets semblent dangereux.
Les NIDS étant les IDS plus intéressants et les plus utiles du fait de l'omniprésence des réseaux dans notre vie quotidienne, ce document se concentrera essentiellement sur ce type d'IDS.

II.2.2 Systèmes de détections d'intrusions de type « hôte » (HIDS)

Les HIDS, pour Host based IDS, sont en général placés sur des machines sensibles, susceptibles de subir des attaques et possédant des données sensibles pour l'entreprise. Les serveurs, web et applicatifs, peuvent notamment être protégés par un HIDS. Il analyse en temps réel les flux relatifs à une machine ainsi que les journaux.

Un HIDS a besoin d'un système sain pour vérifier l'intégrité des donnés. Si le système a été compromis par un pirate, le HIDS ne sera plus efficace. Pour parer à ces attaques, il existe des KIDS (Kernel Intrusion Detection System) et KIPS (Kernel Intrusion Prevention System) qui sont fortement liés au noyau. Ces types d'IDS sont décrits un peu plus loin. [8]

II.2.3 Systèmes de détection d'intrusions « Hybrides »

Généralement utilisés dans un environnement décentralisé, ils permettent de réunir les informations de diverses sondes placées sur le réseau. Leur appellation « hybride » provient du fait qu'ils sont capables de réunir aussi bien des informations provenant d'un système HIDS qu'un NIDS. [9]

II.3 Systèmes de prévention d'intrusions (IPS)

La fonction principale d'un IPS est d'empêcher toute activité suspecte détectée au niveau d'un système : ils ont la capacité de prévenir une attaque avant qu'elle atteigne sa destination. Contrairement aux IDS, les IPS sont des outils aux fonctions «

actives », qui en plus de détecter une intrusion, tentent de la bloquer. Le principe de fonctionnement d'un IPS est analogue à celui d'un IDS, ajoutant à cela l'analyse des contextes de connexion (système assurant le contrôle du trafic réseau) ainsi, on dit que l'IPS est un IDS actif, il pourra alerter l'administrateur d'éventuelles tentatives d'intrusion ou de l'existence d'un trafic suspect.

Si l'action est conforme à l'ensemble de règles existantes, la permission de l'exécuter sera accordée et l'action sera exécutée. Si l'action est illégale (c'est-à-dire si le programme demande des données ou veut les changer alors que cette action ne lui est pas permise), une alarme est générée. Dans la plupart des cas, les autres détecteurs du réseau (ou une console centrale connectées à l'IPS) en seront aussi informés dans le but d'empêcher les autres ordinateurs d'ouvrir ou d'exécuter des fichiers spécifiques (si on est en réseau).

On peut classer les IPS en deux groupes suivants leurs domaines d'utilisation :

II.3 Systèmes de prévention d'intrusions « Réseau » (NIPS)

II.3.1 Définition

Le NIPS est un IPS qui a pour rôle de surveiller le trafic d'un réseau local d'une entreprise, se dernier existe en deux catégories ; la première et software (Ex : snort/snortsam), la seconde et hardware (Ex : Appliance IBM), sa fonction principale est de bloquer des attaques réseau (déni de service DOS, XSS, SQL Injection...) et pour se faire, il utilise une base de règles (rules) qui lui permet d'identifier les différents types d'intrusions afin de les bloquer en négociant avec le firewall.

II.3.2 Fonctionnement d'un NIPS

Le NIPS est capable de décrypter le contenu du flux IP circulant dans le réseau local (LAN 802.3) ou provenant de l'extérieur (Internet), précisément, il fait la correspondance entre les champs des paquets (payload)1 avec celle des empreintes d'intrusions déjà contenu règles (rules). Si le résultat est positif, le NIPS établie une négociation avec le Firewall (pare-feu) qui va réagir selon la politique dans les règles afin de définir l'action souhaitée (Rejet, Blocage, Quarantaine,…)

La figure suivante nous montre l'emplacement du NIPS dans un réseau ainsi que son fonctionnement qui se base sur l'analyse du contenu du paquet IP.

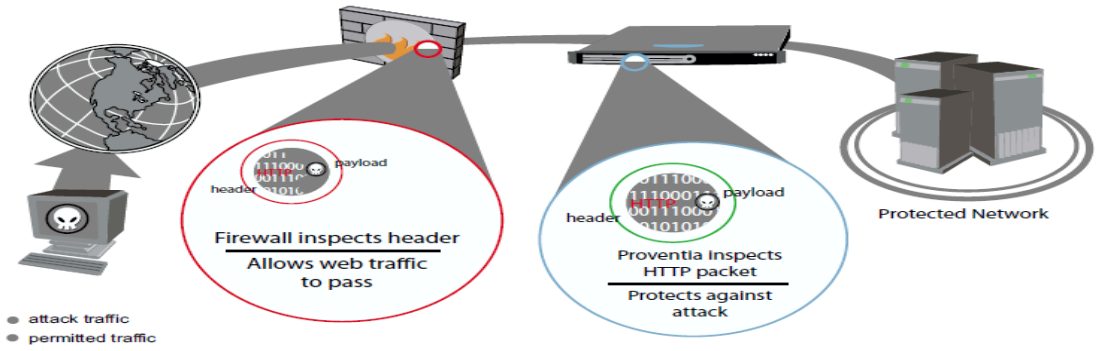


Figure 6 : Principes de fonctionnement d'un NIPS.

II.3.3 Systèmes de prévention d'intrusions « Kernel » (KIPS)

Dans certains cas, l'utilisation d'un détecteur d'intrusions au niveau noyau peut s'avérer parfois nécessaire pour sécuriser une station.
Prenons l'exemple d'un serveur web, sur lequel il serait dangereux qu'un accès en lecture/écriture dans d'autres répertoires que celui consultable via http, soit autorisé. Le KIPS peut reconnaître des motifs caractéristiques du débordement de mémoire, et peut ainsi interdire l'exécution du code. Le KIPS peut également interdire l'OS d'exécuter un appel système qui ouvrirait un shell de commandes. Puisqu'un KIPS analyse les appels systèmes, il ralentit l'exécution. C'est pourquoi ce sont des solutions rarement utilisées sur des serveurs souvent sollicités.

Exemple de KIPS : SecureIIS, qui est une surcouche du serveur IIS de Microsoft.

II.4 Inconvénients des IDS/IPS

II.4.1 Besoin de connaissances en sécurité

La mise en place de sonde sécurité fait appel à de bonnes connaissances en sécurité.

L'installation en elle-même des logiciels est à la portée de n'importe quel informaticien. En revanche l'exploitation des remontées d'alertes nécessite des connaissances plus pointues.

La configuration, et l'administration des IDS nécessitent beaucoup de temps, et de connaissances. C'est un outil d'aide, qui n'est en aucun cas complètement automatisé. [10]

II.4.2 Faux positifs et faux négatifs

· **Faux positif** : une alerte provenant d'un IDS ou IPS mais qui ne correspond pas à une attaque réelle.

· **Faux négatif** : une intrusion réelle qui n'a pas été détectée par l'IDS ou IPS.

Les développeurs des IPS et IDS s'efforcent afin de réduire les faux positifs. Mais la signification et l'importance du «Faux positif» est différente pour l'IPS et l'IDS. La différence provient des objectifs de conception de l'IDS et IPS. Un IDS est conçu pour alerter un analyste de la sécurité d'un comportement suspect par contre, un IPS est censé atténuer les attaques en temps réel.
Quand un IPS dispose d'un faux positif, la principale préoccupation est que le trafic légitime soit bloqué. La plupart des organisations considèrent le blocage du trafic légitime comme un problème beaucoup plus grave que de générer une fausse alerte. Par conséquent, un faux positif d'IPS est une question beaucoup plus grave qu'faux positif d'IDS.

II.4.3 Pollution/surcharge

Les IDS /IPS peuvent être pollués ou surchargés, par exemple par la génération d'un trafic important (le plus difficile et lourd possible à analyser). Une quantité importante d'attaques peut également être envoyée afin de surcharger les alertes de l'IDS. Des conséquences possibles de cette surcharge peuvent être la saturation de ressources (disque, CPU, mémoire), la perte de paquets, le déni de service partiel ou total …

II.4.4 Consommation de ressources

Outre la taille des fichiers de logs, la détection d'intrusion est excessivement gourmande en ressources. En effet un système NIDS doit générer des journaux de comptes-rendus d'activité anormale ou douteuse sur le réseau.

II.4.5 Perte de paquets (limitation des performances)

Les vitesses de transmission sont parfois telles qu'elles dépassent largement la vitesse d'écriture des disques durs, ou même la vitesse de traitement des processeurs.

Il n'est donc pas rare que des paquets ne soient pas traités par l'IDS, et que certains d'entre eux soient néanmoins reçus par la machine destinataire.

II .5 Conclusion

Dans cette partie, on a montré que la détection d'intrusions dans les réseaux ne vient pas concurrencer les mécanismes de sécurité traditionnels mais, au contraire, les compléter. Même si on ne peut pas atteindre la sécurité absolue, on veut au moins pouvoir détecter l'intrusion afin d'y remédier. On a également présenté les principes mis en œuvre par les systèmes de détection d'intrusions pour atteindre leur but. Finalement, on a vu que de nombreux problèmes restent à résoudre avant que la détection d'intrusions soit fiable. Cette technologie n'est pas encore arrivée à la maturité et les outils existants ne sont pas toujours à la hauteur des besoins. Certaines approches théoriques doivent encore être validées dans la pratique. De nouvelles approches demandent encore à être approfondies comme l'immunologie ou les systèmes basés agents.

Partie pratique

Chapitre III : Mise en œuvre d'un NIPS (Snort/SnortSam)

III Introduction

L'ouverture des systèmes et leurs interconnexions avec le réseau Internet ont fait que les attaques soient de plus en plus nombreuses et diversifiées les unes que des autres.

Outre la mise en place de pare-feux et de systèmes d'authentification, il est de nos jours nécessaire de mettre en place un système de prévention d'intrusions (Snort/SnortSam).

Snort/SnortSam sont des logiciels open source travaillant sous l'environnement UNIX et qui forment en les combinant un NIPS

Dans ce qui suit, nous allons commencer par donner une présentation générale de SNORT et SnortSam, ensuite nous allons présenter leurs manipulations : installation, configuration et fonctionnalités. Enfin, nous allons terminer par donner une conclusion et des perspectives pour ce travail.

III.1 Présentation générale de Snort

SNORT est un NIDS écrit par Martin Roesch en 1998 et développé par Sourcefire. 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 la journalisation de paquets sur Protocole Internet (IP).

Snort effectue l'analyse de protocole, la recherche de contenu, et l'appariement de contenu. Il utilise une sonde pour détecter les attaques, le débordement système, les scans de ports, etc.

III.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 :

Voici un schéma illustrant ces différentes positions :

Dans la figure suivante, on voit les différentes positions possibles de l'emplacement de SNORT au niveau du réseau local.

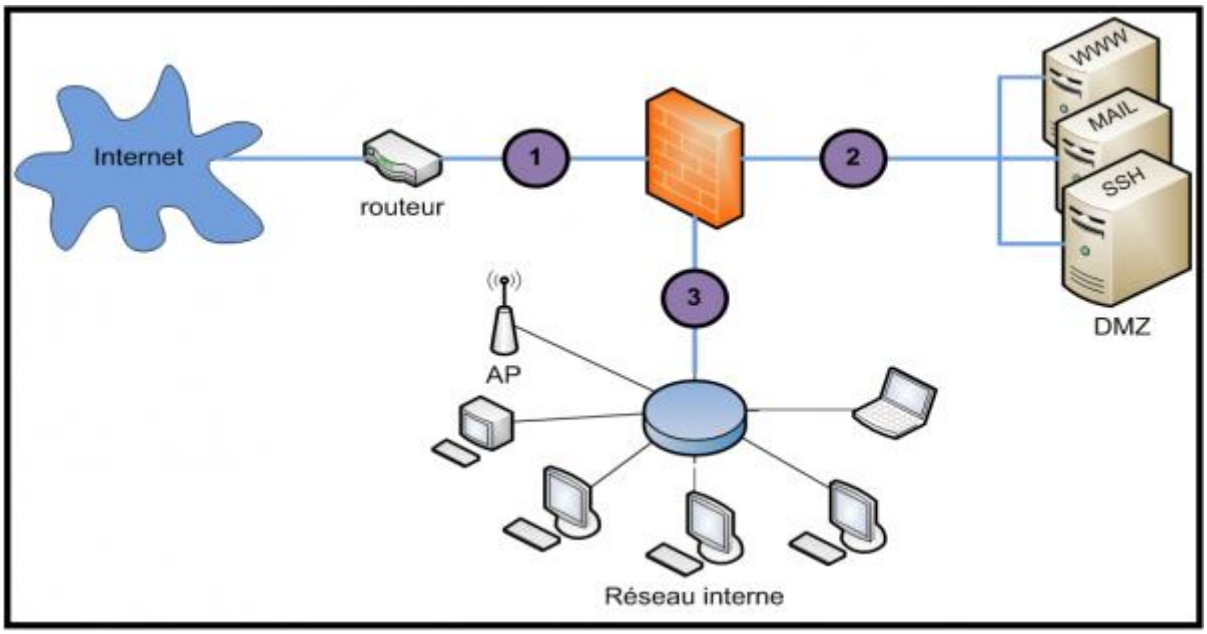


Figure 7 : Différentes positions possibles de Snort dans un réseau Informatique.

(1) 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. Ainsi, beaucoup d'alertes seront remontées ce qui rendra les logs difficilement consultables.

(2) Seul le trafic entre la DMZ et internet ou le réseau interne est analysé. De plus, placer SNORT à cet emplacement nous permet de détecter les attaques non filtré par le pare-feu et donc minimise le trafic réseau à analyser. Cependant le trafic entre le réseau interne et internet n'est pas visible pour SNORT.

(3) sur le réseau interne : Le positionnement du NIDS à cet endroit nous permet d'observer les tentatives d'intrusion 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. [11]

III.1.3 Architecture de Snort

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

- **Un noyau de base :** Au démarrage, ce noyau charge un ensemble de règles, compile, optimise et classe celles-ci. Durant l'exécution, le rôle principal du noyau est la capture de paquets.
- **Une série de pré – processeurs**, ceux-ci améliorent les possibilités de SNORT en matière d'analyse et de recomposition du trafic capturé. Ils reçoivent les paquets

directement capturés, éventuellement les retravaillent puis les fournissent au moteur de recherche de signatures.

- **Une série d'analyses est ensuite appliquée aux paquets :** Ces analyses se composent principalement de comparaisons de différents champs des headers des protocoles (IP, ICMP, TCP et UDP) par rapport à des valeurs précises. [12]

Après la détection d'intrusion, une série de « output plugins » permet de traiter cette intrusion de plusieurs manières : envoie vers un fichier log, envoie d'un message d'alerte vers un serveur syslog, stocker cette intrusion dans une base de données SQL.

Cette figure illustre le principe du fonctionnement de SNORT qui est basé sur différents modules de traitement de flux.

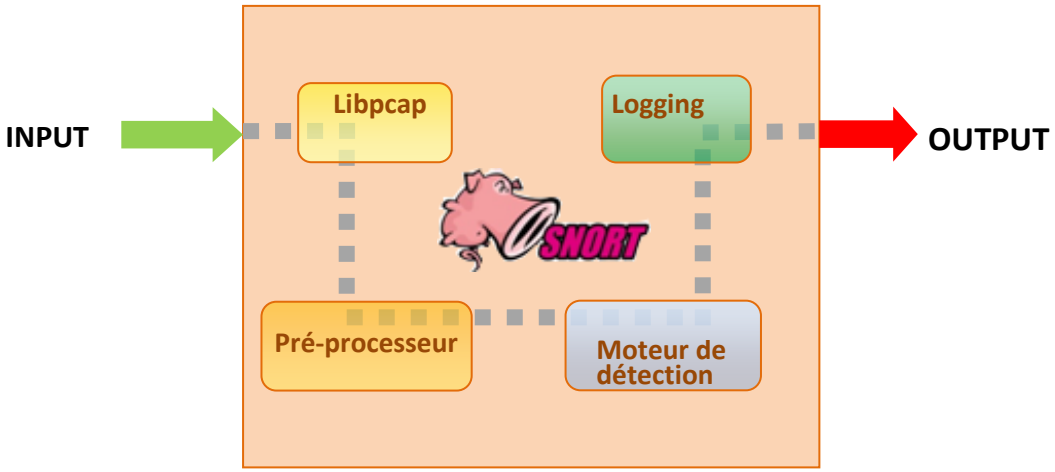


Figure 8 : Fonctionnement de Snort.

III.1.4 Environnement

Pour une sécurité optimale, nous avons préféré de travailler dans un environnement Linux, plus précisément : Ubuntu 10.04 LTS, car il nous fournit un espace de travail unique et nous assure une fiabilité de résultats incomparable.

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

ubuntu

III.1.5 Installation de Snort

- **Pré-requis :**

nous devonsd'abord installer les outils de compilations et les dépendances de Snort :

Libpcap (Packet CAPture): Bibliothèque logicielle pour la capture de paquets IP.
Libnet : C'est une bibliothèque open source. Elle permet de fabriquer et d'injecter facilement des paquets sur un réseau.

III.1.6 Modes de fonctionnement

- **Mode écoute (Sniffer mode) :** Ce mode permet de lancer snort en mode sniffer et permet décrypté le contenu et l'entête dés paquet sur l'interface connecté au reseaux ("snort -vde")
- **Mode "log de paquets" :** Le log de paquet permet l'archivage des paquets circulant sur le réseau de l'IDS. Il permet, grâce à ses arguments des opérations intéressantes permettant de limiter les logs à certains critères, comme une plage d'adresse IP (ex : "snort -l ../log/snort -h 192.168.0.0/24") .
- **Mode "détection d'intrusion" :** Le mode NIDS permet à snort d'adopter un comprtement particulier en cas de détection d'une (succession) de chaînes de caractères dans les paquets interceptés ; selon les règles définies dans des fichiers d'extension ".rules».

III.1.6.1 Le mode « NIDS»

Le véritable intérêt de SNORT est le mode NIDS, il utilise pour cela des règles pour détecter les intrusions. Il existe aujourd'hui environ 1500 règles différentes, chacune s'adaptant à un cas particulier. On peut créer des règles pour observer une activité particulière sur notre réseau : pings, scans, connexions par backdoors, faille dans un script, tentative de prise de contrôle à distance ... Les alertes peuvent être enregistrées dans un fichier particulier ou directement dans le syslog et être rajoutées aux messages système ou encore dans une base de données... Chaque règle se rajoute dans un fichier de configuration prévu à cet effet, vous pouvez soit utiliser celles qui existent déjà, soit en créer de nouvelle. Le fichier de configuration de snort est `/etc/snort/snort.conf`, les fichiers `.rules` contenus dans `/etc/snort/rules/` sont des fichiers contenant, des règles pour un usage bien particulier. Le nom du fichier est, en général explicite, ainsi, ftp.rules contient des règles spécifiques au ftp et `dos.rules` s'utilise pour les tentatives de DoS (Denial Of Service). [13]

III.1.7 Paramétrage de Snort

III.1.7.1 Pré-processeurs

Les préprocesseurs permettent d'étendre les fonctionnalités de SNORT. 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 preprocessor dans les règles de SNORT est :

```
preprocessor <nom>   <options>. [14]
```

Exemple :

```
preprocessor minfrag: 128
```

III.1.7.2 Les plugins de sortie

Plusieurs plugins de sortie d'alerte peuvent être spécifiés dans le fichier de configuration de Snort tell que : syslog , tcpdump ,base de données,xml,mail,unified2 , 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>
```

III.1.7.3 Les règles de Snort (Rules)

Snort permet l'écriture de 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 règle contient comme informations l'action de la règle, le protocole, les adresses IP source et destination et les masques réseau, et les ports source et destination. La section options 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. [12]

III.1.7.3.1 Création de règles

Une règle Snort est composée de deux parties présentées sous le format suivant :

```
Header (options)
```

Action	Protocole	Adresse1	Port1	Direction	Adresse2	Port2	Options (msg, content…)

Figure 9 : Format d'une règle de Snort.

Le champ « action » peut prendre plusieurs valeurs selon l'action à mener par Snort en détectant des paquets réseau répondant au critère définie dans la règle. Ces valeurs sont les suivantes :

- alert : génère une alerte et log le paquet
- log : log le paquet
- pass : ignore le paquet
- activate : active une règle dynamique
- dynamic : définie une règle dynamique
- ..etc

Le champ « protocole » décrit le protocole utilisé pour la communication. Snort supporte les protocoles TCP, UDP, ICMP et IP.

Les champs « direction » renseignent Snort sur la direction des échanges réseau (->, <->, <-).

Les champs « adress/port » décrivent les adresses IP et les ports des machines qui échangent des données sur le réseau.

Quant à la partie options, elle renseigne Snort sur les caractéristiques des paquets à signaler et garantissent une meilleure granularité pour définir et appliquer les règles mais aussi déclencher les actions qu'elles décrivent.

La partie options est constituée de plusieurs champs qui assurent l'analyse du contenu des paquets réseau avec plus de finesse. Notons que la manipulation de ces champs nécessite une grande maîtrise des protocoles réseau pour pouvoir décrire les signatures des attaques à détecter.

Pour chaque option le format est nom option : valeur1 [, valeur2,...] ci-dessous les options utilisées dans la création des règles :

- 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 contente, fixe le décalage du début de la tentative de correspondance de motif

- depth : modifie l'option contente, 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) [15]

III.1.7.3.2 Mise à jour de règles (rules)

Les mises à jour des règles de Snort sont disponible sur le site officiel `http://www.snort.org`, cependant une inscription annuelle est requise.

III.2 Mise en place de Barnyard2

Lorsqu'un comportement suspect est intercepté par Snort en fonction des signatures, Snort inscrit les événements. Il est possible d'inscrire les logs en base de données directement. Néanmoins, dans un souci d'optimisation (libération de ressources), nous utiliserons Barnyard, une couche applicative qui exploite les événements générés par Snort. Ainsi, Snort inscrira les événements dans des logs au format unifié (Fast Unified Logging) et ces derniers seront exploités par Barnyard pour une inscription en base de données. [16]

III.2.1 Le plugin « Unified 2 »

Le plugin de sortie unifié est conçu pour être la méthode la plus rapide possible de la journalisation des événements de Snort. Le plugin de sortie unifié enregistre les événements dans un format binaire, ce qui permet encore d'alléger le mécanisme sur les alertes dés événement.

Le nom *unifié* est un terme impropre, puisque le plugin de sortie unifié crée deux fichiers différents, un fichier *d'alerte*, et un fichier *journal*. Le fichier d'alerte contient les détails de haut niveau d'un événement (par exemple: IP, protocole, port, identifiant de message).

Le fichier journal contient les informations de paquets détaillés (un dump paquet avec l'ID d'événement associé). Les deux types de fichiers sont écrits dans un format binaire.

```
Output unified2: \
         <nom du fichier nom de fichier <base [, <taille limite <en Mo>]
```

III.3 La console B.A.S.E

Par défaut, les alertes de Snort sont enregistrées dans un simple fichier texte. L'analyse de ce fichier n'est pas aisée, même 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. Parmi ceux-ci, le plus en vogue actuellement est BASE (Basic Analysis and Security Engine), un projet open-source basé sur ACID (Analysis Console for Intrusion Databases). La

console BASE est une application Webécrite 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 Apache.
- Php5 : module PHP.
- Php5-mysql : interface PHP/MYSQL
- La bibliothèque ADODB (signifie Active Data Objects Data Base), c'est une bibliothèque destinée à communiquer avec différents systèmes de gestion de base de données (SGDB) comme Mysql, SQL Server,etc. Écrite au début en PHP, il existe également une version en Python.
- Php-mail : extension PHP.

III.4 SnortSam

III.4.1 présentation

Snortsam 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 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. Ce plugin supporte plusieurs types de pare-feu, dans notre cas nous allons utiliser le firewall logique de linux (iptables). [17]

L'avantage de SnortSam, c'est que l'agent avec lequel il fonctionne fourni plusieurs fonctionnalités que d'autres mécanismes de blocage automatisés ne proposent pas, comme:

- Une liste blanche d'adresse IP qui ne seront jamais bloquées.
- Un système de plage horaire pour bloquer les utilisateurs.
- Des règles de blocage spécifiques, comme les règles de blocage dépendant de l'heure.
- Une liste de filtrage de SID autorisés ou non, basée sur l'entité.
- Un moteur de détection d'attaque qui essaye d'atténuer le risque d'un Deni de service interne à l'architecture IDS.
- Un système de blocage préventif des répétitions (mêmes adresses IP) avec une fenêtre modifiable pour améliorer les performances.
- Une communication chiffrée en TwoFish entre Snort et l'agent SnortSam.
- Un vrai support d'OPSEC en utilisant le SDK de Checkpoint (plugin OPSEC)
- Le « Multi-Thread » pour traiter plus vite et simultanément les blocages sur plusieurs appareils
- L'enregistrement des fichiers et la notification d'événements par email.
- Et la possibilité d'utiliser l'architecture client/serveur (snortsam/snort) pour les grands réseaux de manière à optimiser les remontées d'information du réseau.

De plus, SnortSam est un programme complètement gratuit et open-source. Il peut être compilé sur plusieurs plateformes (linux,open bsd,mac..) et l'interaction devrait fonctionner à travers ces différentes plateformes

La figure suivante nous montre la liaison de SNORT avec ses différents agents SnortSam et qui communiquent entre eux en utilisant un algorithme de cryptage *Twofish*.

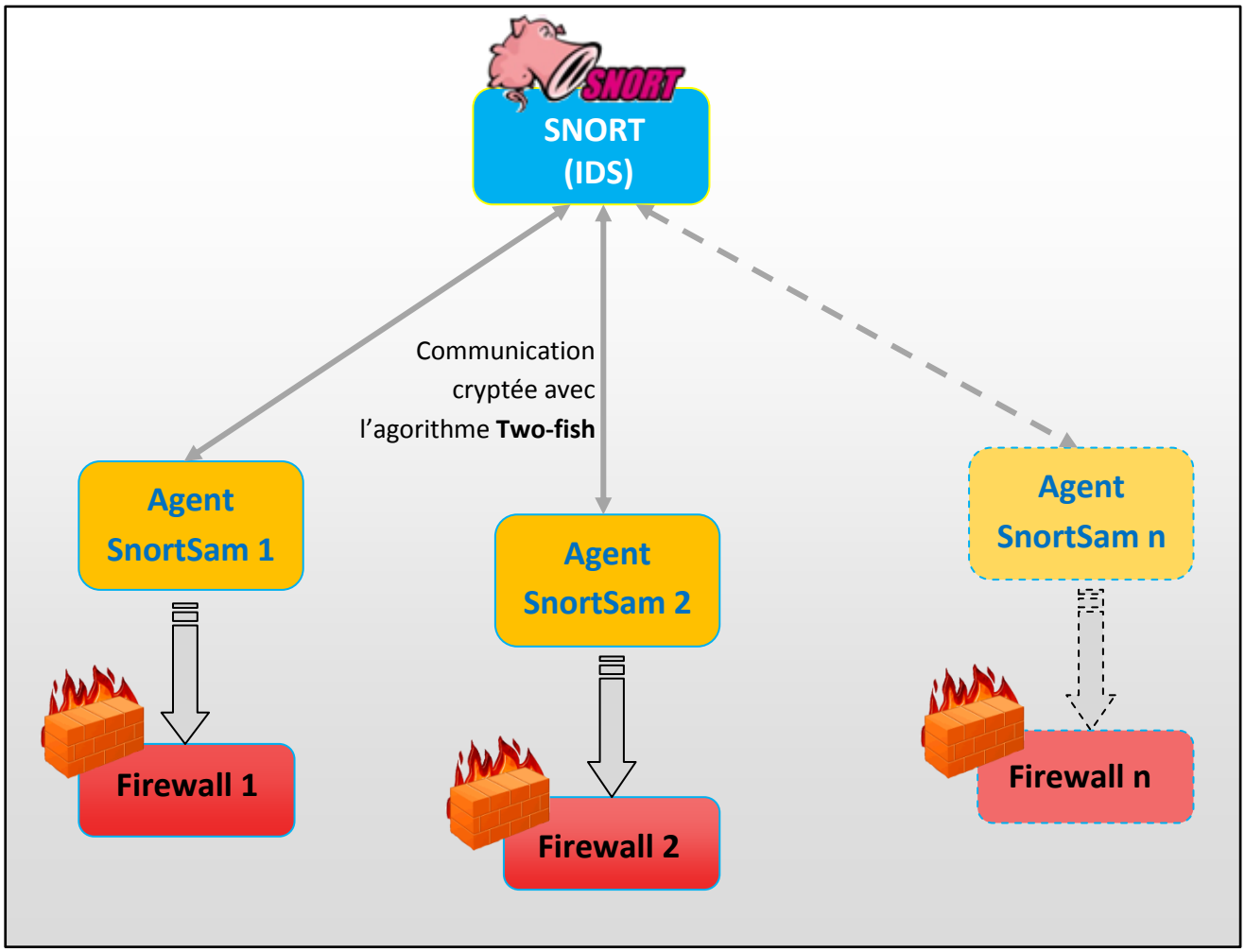


Figure 10 : Schéma généralisant l'architecture de l'ensemble Snort/SnortSam

III.4.2 Configuration de règles de SNORT (fwsam)

L'intérêt de configuré les rules est le blocage des adresse IP coté firewall, et pour cela on injecte le champ « fwsam » dans les lignes des règles.

III.4.2.1 Fwsam

FWSAM est une extension aux règles de snort et qui permet la génération des paramètres de blocage coté firewall, ca structure est la suivante :

```
fwsam: {qui} comment, le temps;
```

Qui : Peut être: src : la source, dst :la destination

Comment : Peut être initialisé à In, out, src, dest, either, both, this ou conn pour spécifier au firewall le type des paquets à rejeter.

Temps : Inique au pare-feu la durée de blocage du paquet IP en question, cependant, la durée peut etre initiée en quelques secondes, minutes, jours, voir meme des mois ou années ! Sinon, une valeur de 0, ou le mot-clé« permanent » ou « infini », permet de bloquer l'hôte de façon permanente. [18]

Example:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:
"WEB-ATTACKS /bin/ps command attempt ";
flow:to_server,established; uricontent: "ps%20"; nocase;
sid:1329; classtype:web-application-attack; rev:4; fwsam:
src[in], 5 minutes;)
```

Cette règle s'appui sure une connexion de l'extérieure vers les serveurs internes. Cela signifie que la source est l'attaquant, la destination c'est nous. Fwsam négocie avec le firewall afin de bloquer l'adresse IP a partir de la source et pour une durée de 5 minutes.

III.5 IPtables/Netfilter

iptables est un logiciel open source Linux utilisé pour mettre en place, maintenir et inspecter les tables des règles de filtrage des paquets IP du noyau Linux. On peut configurer les chaînes et règles dans le pare-feu en espace noyau (et qui est composé par des modules Netfilter).

Netfilter est l'implémentation noyau du firewall sous Linux, c'est l'un des firewalls les plus puissants du marché. Il est souple, simple à configurer et très largement supporté par la communauté.

Iptables est la commande permettant de paramétrer le filtre Netfilter du noyau et donc de configurer son firewall.

Dans notre cas, l'agent SnortSam négocie avec IPtables qui joue le rôle du firewall pour d'éventuel blocage d'adresse IP etc.

La figure suivante nous montre le mécanisme de log assuré par SNORT, le blocage de paquet fait par SnortSam et l'affichage des alertes.

III.6 architecture applicative

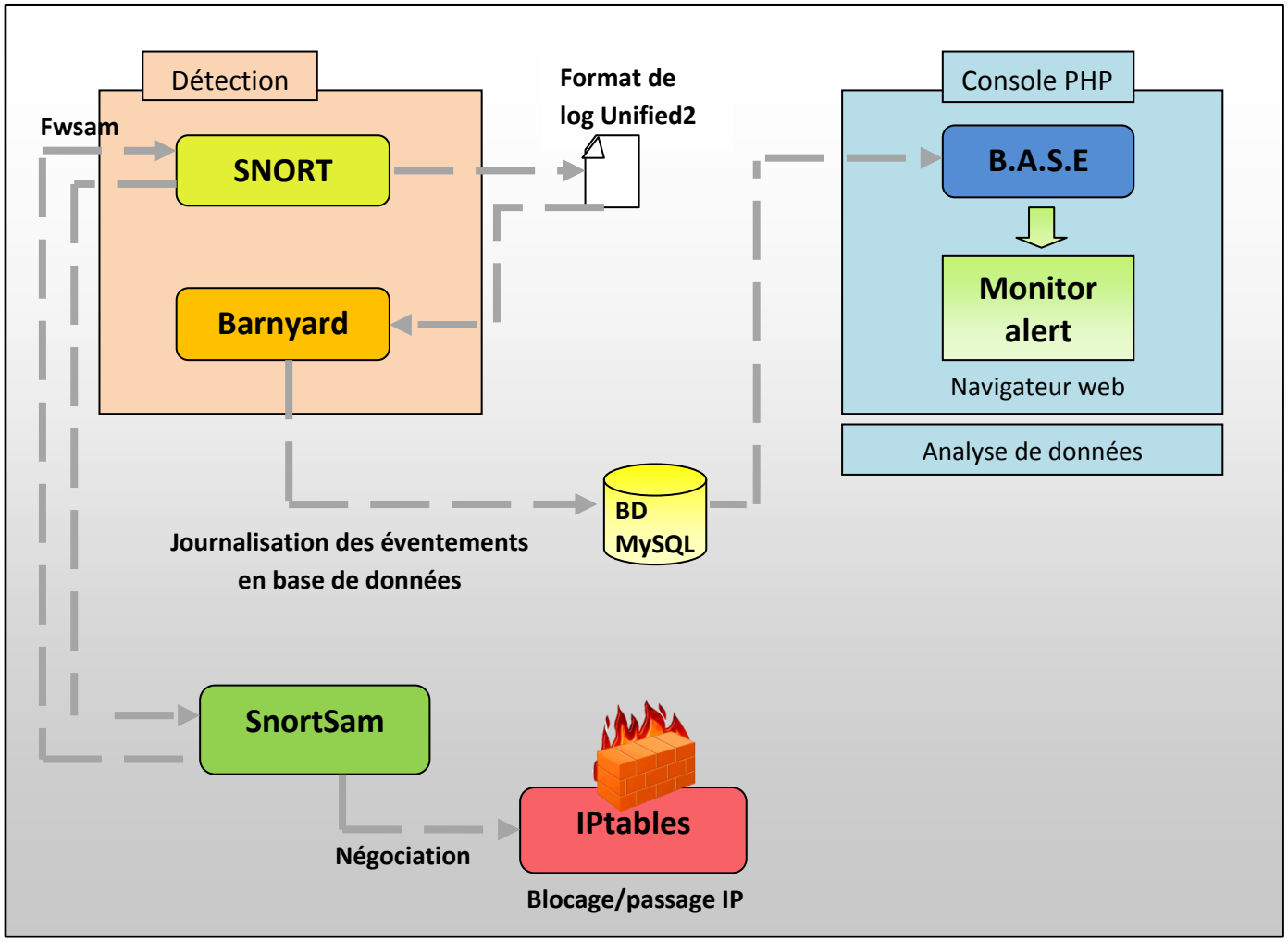


Figure 11 : Architecture généralisant le fonctionnement du NIPS (Snort/SnortSam)

Le mécanisme de NIPS (snort/snortsam) est composé de plusieurs entités, snort capture les paquets du réseau (frontal ou dupliqué) en utilisant Libpcap, a ce moment la, l'interface réseau de Snort est en mode monitor. Snort va logué les événements (alertes) dans un fichier de format unifié (unified 2)

Le module barnyard a pour mission de récupérer les logs unifiés afin de les inscrire dans la base de données MYSQL, pour qu'ils soient accessibles via BASE qui est une console graphique.

Le module BASE utilise un navigateur web pour afficher les événements (alertes) en détails, afin de faciliter la tache a l'administrateur réseau.

En parallèle avec les logs, snort active le plugin fwsam pour communiquer avec l'agent snortsam et lui fournir les informations nécessaires, cependant la connexion entre snort et snortsam est cryptée.

Snortsam analyse les événements reçus auprès de snort, afin de négocier avec le firewall existant pour la mise en œuvre d'une règle de blocage d'adresse IP.

III.7 Outils d'attaques (Anonymous OS)

Pour tester la fiabilité et la puissance notre application, nous avons utilisé des outils d'attaques contenus dans un système d'exploitation portant le nom des célèbres hackers, « Anonymous-OS », c'est un système d'exploitation basé sur Ubuntu 11.10 et qui est équipé de pas mal d'outils d'attaques très répondus, a savoir SQL injection, DoS (deni de service), etc. Donc c'est l'outil idéal pour effectuer des testes (pentesting).

Cette figure nous montre les différents outils d'attaques contenus dans la distribution Anonymous OS :

Figure 12 : Aperçu des outils d'attaque.

Pour notre NIPS (snort /snortsam) nous allons effectué une attaque DoS(Denil of service) Slowloris qui est un outil puissant permettant de faire tomber rapidement un serveur web. Ecrit en perl par RSnake, il affecte en particulier les serveurs Apache.

Principe

Le principe repose sur l'envoi des requêtes HTTP partielles afin de garder les sockets ouverts. Slowloris va instancier des Handshake TCP d'une manière infinie, ce qui va causer la saturation du serveur web cible.

III.8 Conclusion

Snort est un outil très intéressant dans la mise en place d'une sécurité réseau. Grâce aux communautés très actives qui créent les librairies d'attaques. Snort permet de voir avec une bonne acuité de quoi il faut se protéger.

De plus Snort placé dans l'enceinte d'un réseau permet de détecter les failles les plus répandues qui proviennent généralement de l'intérieur de l'entreprise, et non de l'extérieur. Ce système de détection multiplateforme est en perpétuelle évolution et semble un des meilleurs outils dans la connaissance des vulnérabilités auxquelles on est exposé.

Chapitre IV : Installation de Snort/SnortSam

IV Introduction

Dans ce dernier chapitre, nous allons voir un cas pratique concernant Snort, SnortSam, nous allons voir comment installer les différents composant du NIPS(Snort/SnortSam) ainsi que toutes les configurations nécessaires.

Au final, nous allons tester notre configuration on laçant quelques attaques et essayer de les détecter et de les stopper.

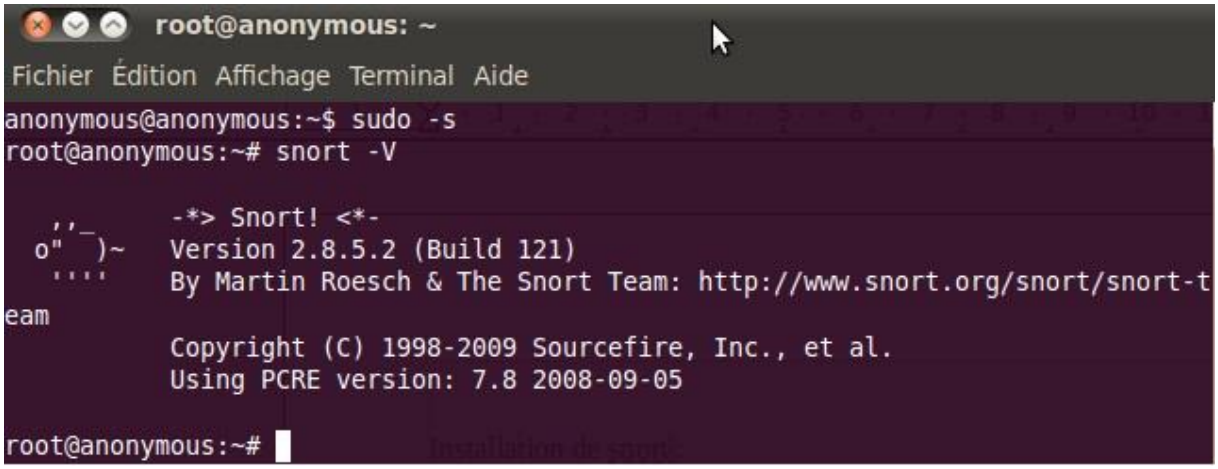
IV.1 Installation de Snort

A partir du shell en tape la commande (sert a récupérer Snort a partir des dépôts d'archives Ubuntu) :

```
# apt-get install snort
```

IV.1.2 Lancement de Snort

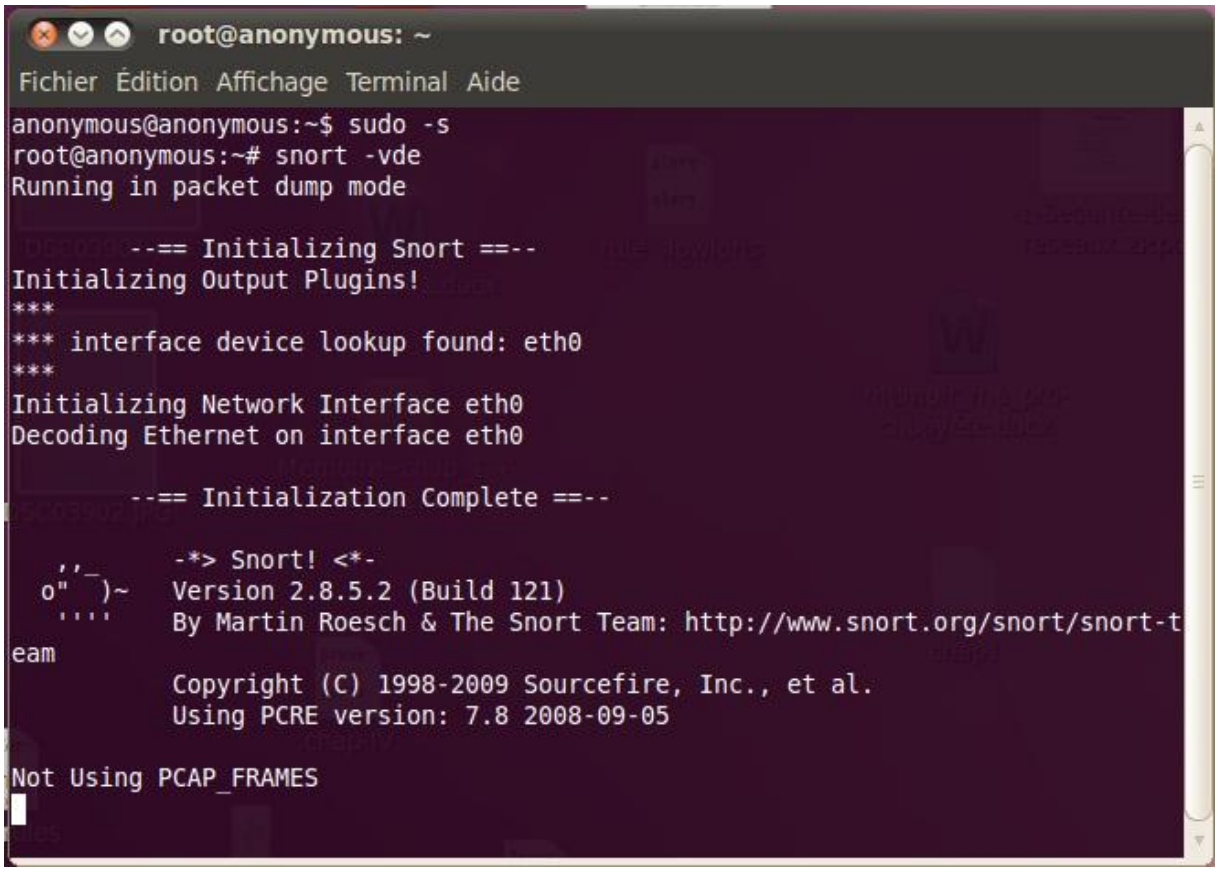
Vérification de la l'installation avec la commande shell: `snort -V` et qui affiche la version de snort (2.8.5.2)

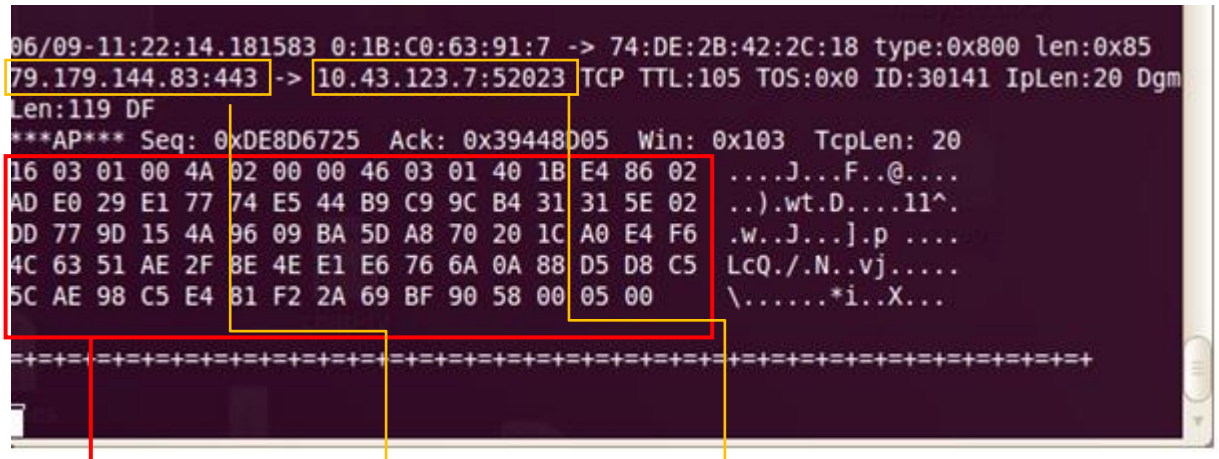


IV.1.3 Modes de fonctionnement

IV.1.3.1 Le mode écoute (Sniffer mode)

Ce mode permet de lire les paquets circulant sur le réseau et les affiche d'une façon continue sur l'écran

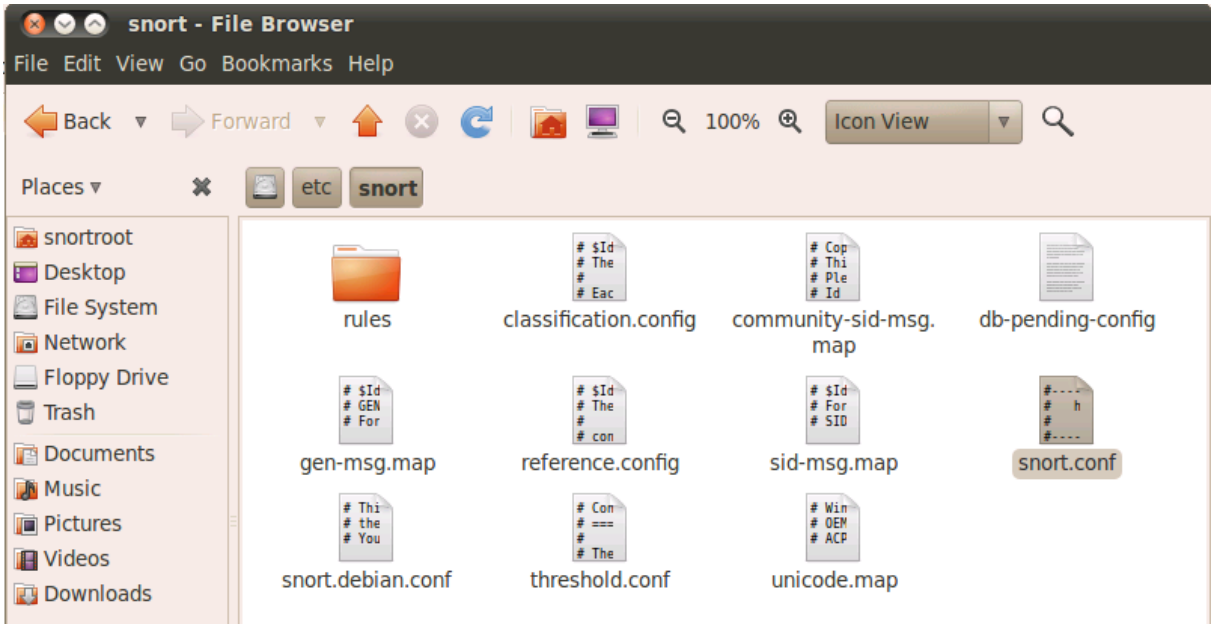




Contenu du paquet IP Adresse source Adresse destination

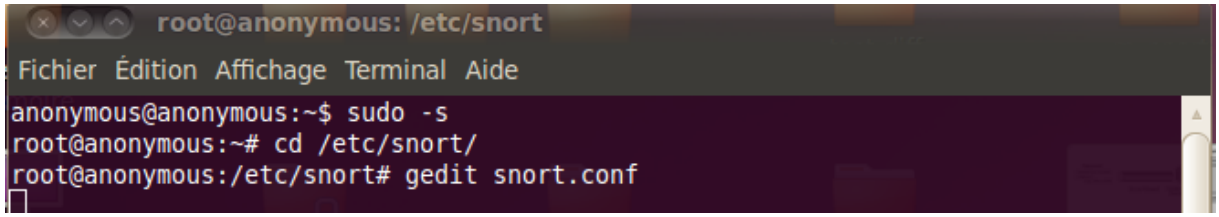
IV.1.3.2 Le mode « detection d'intrusion NIDS »

- Configuration du fichier `snort.conf` :



On édite le fichier snort.conf a l'aide de la commande shell :

```
#gedit /etc/snort/snort.conf
```

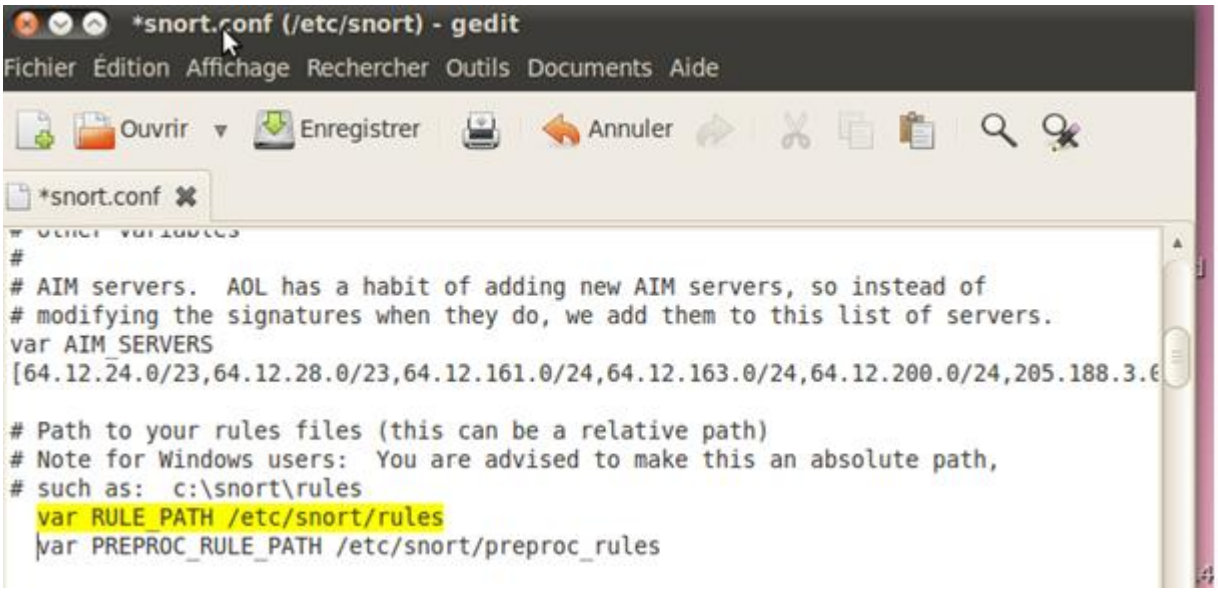


On a choisi l'interface réseaux Ethernet (802.3).
L'interface réseau peut être : `wlan0, eth0, ppp,` etc.

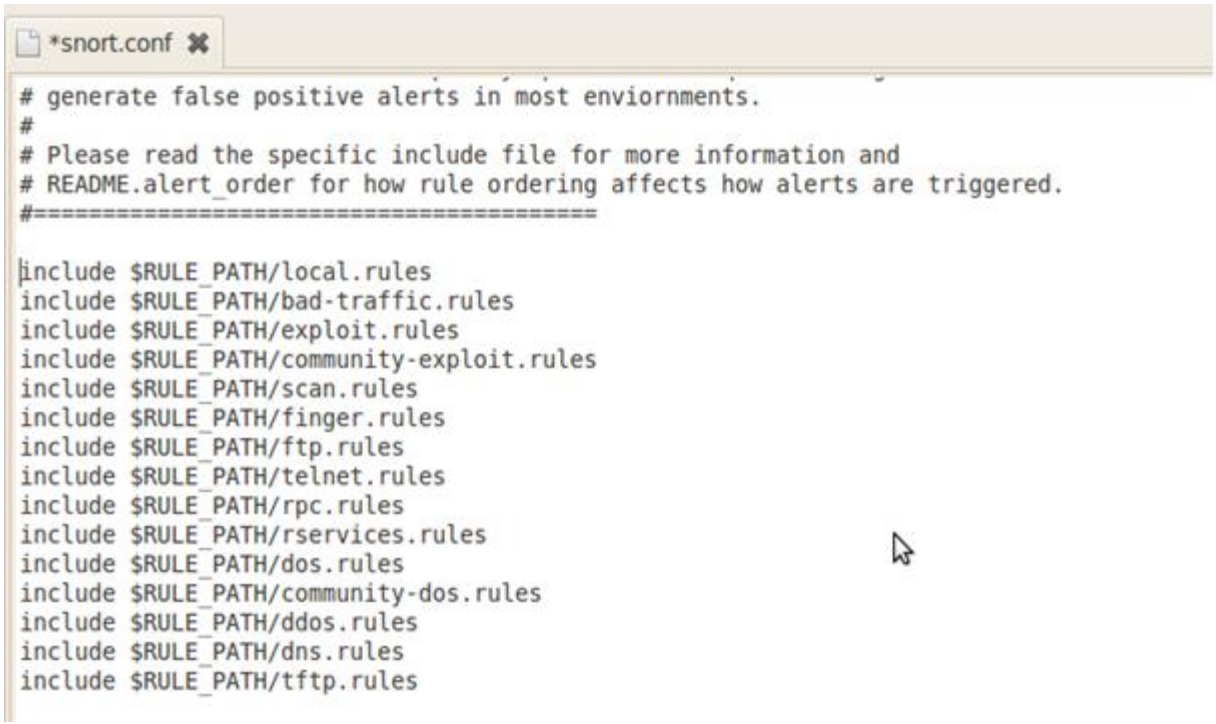
- on indique la classe d'adresse du réseau, comme suit :

```
var HOME_NET 192.168.1.0/24
```

- On indique maintenant le répertoire où sont disposées nos règles (rules) :

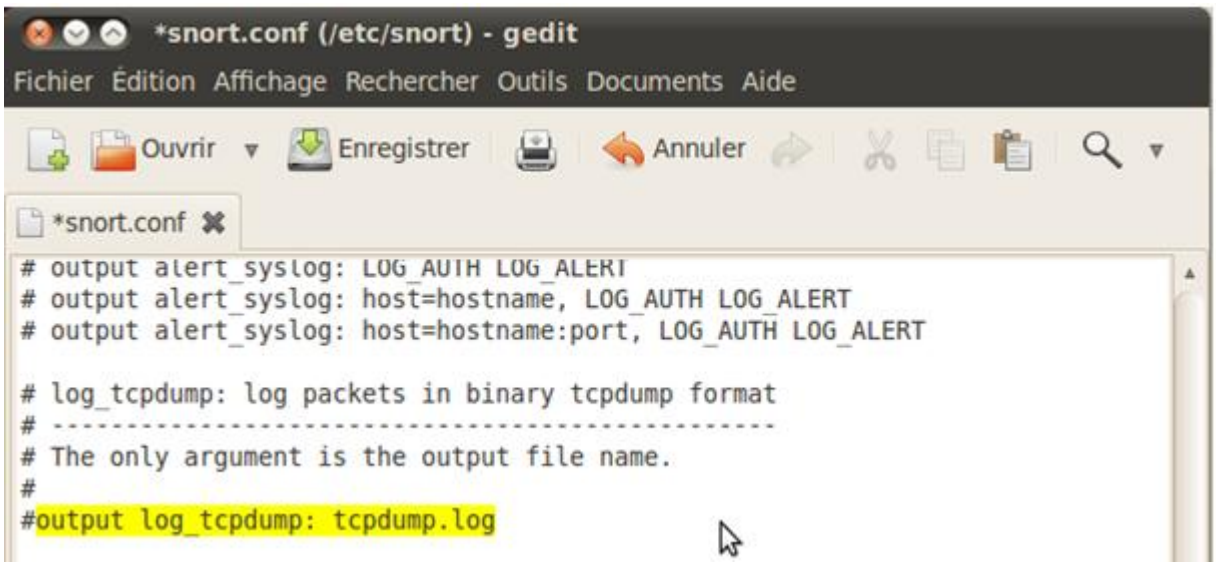


Puis, on inclut les règles qui nous intéressent :



On commente le fichier de sortie `output log_tcpdump` en ajoutant le carectère «#», car dans notre cas, on travail avec un module de sortie du format `unified` en ajoutant :

```
output unified2 : filename snort.log , limit 128
```



*snort.conf

```
# overhead for logging and alerting to various slow storage mechanisms such
as
# databases or the network can now be avoided.
#
# Check out the spo_unified.h file for the data formats.
#
# Two arguments are supported.
#    filename - base filename to write to (current time_t is appended)
#    limit    - maximum size of spool file in MB (default: 128)
#
# output alert_unified: filename snort.alert, limit 128
#output log_unified: filename snort.log, limit 128

output unified2: filename snort.log , limit 128
```

Puis, on spécifie les préprocesseurrs adéquats :

snort.conf (/etc/snort) - gedit

Fichier Édition Affichage Rechercher Outils Documents Aide

Ouvrir Enregistrer Annuler

snort.conf

```
# Performance Statistics
# ----------------------
# Documentation for this is provided in the Snort Manual.  You should read
it.
# It is included in the release distribution as doc/snort_manual.pdf
#
# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt 10000

# http_inspect: normalize and detect HTTP traffic and protocol anomalies
#
# lots of options available here. See doc/README.http_inspect.
# unicode.map should be wherever your snort.conf lives, or given
# a full path to where snort can find it.
preprocessor http_inspect: global \
    iis_unicode_map unicode.map 1252

preprocessor http_inspect_server: server default \
    profile all ports { 80 8080 8180 } oversize_dir_length 500

#
#  Example unique server configuration
#
#preprocessor http_inspect_server: server 1.1.1.1 \
#    ports { 80 3128 8080 } \
```

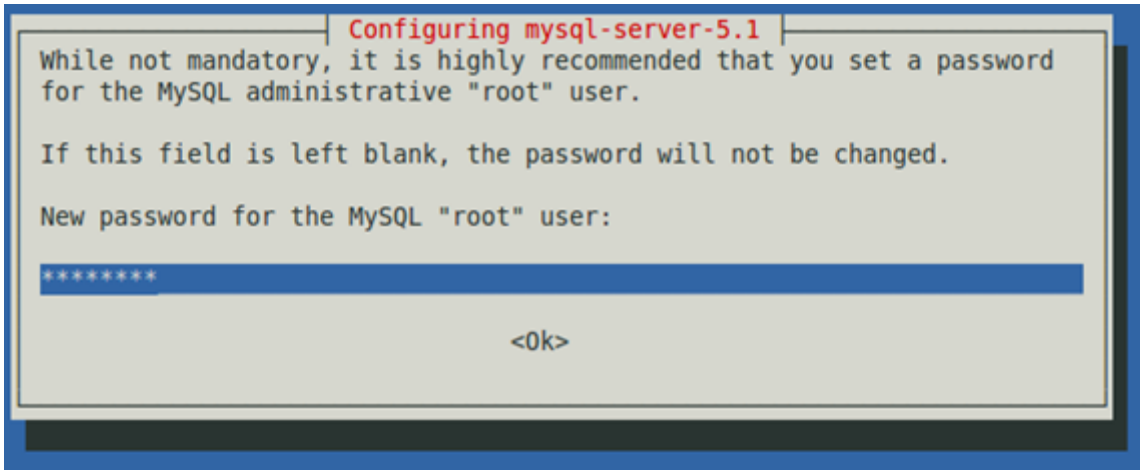
IV.1.4 Mise en œuvre de la base de données MySql

IV.1.4.1 Installation

La première étape consiste à installer le serveur et le client MySQL :

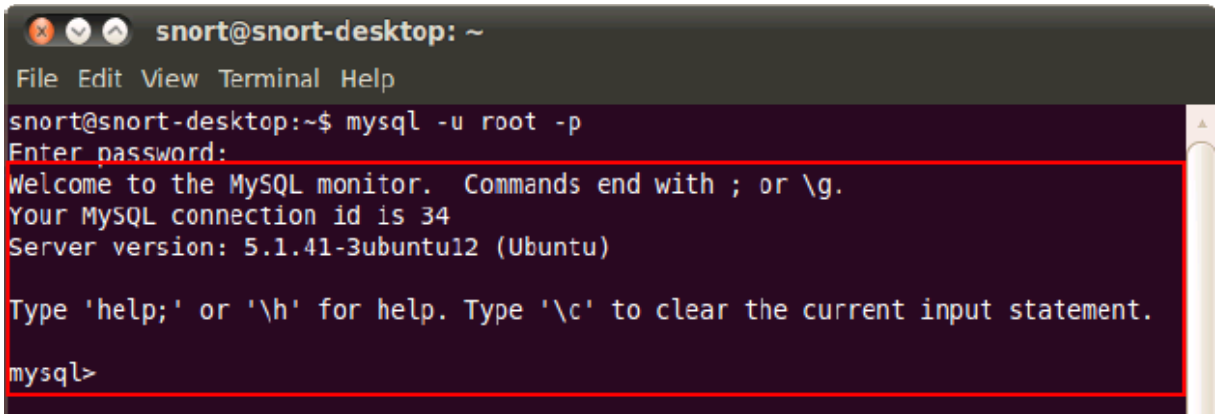
```
sudo apt-get install mysql-server mysql-client
```

Pendant l'installation, on spécifie le mot de passe root pour Mysql:



On accède à la base de données comme suit :

```
$ mysql –u root -p
```

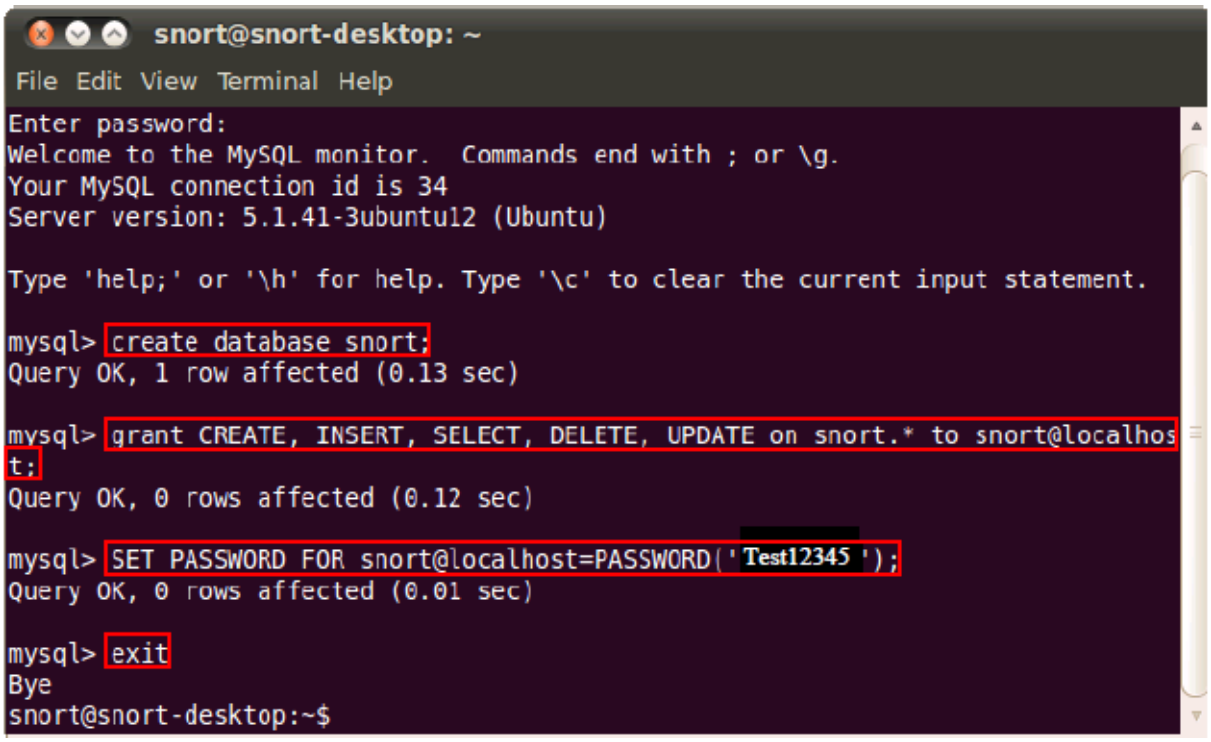


IV.1.4.2 Création de la base de données pour Snort

On lance d'abord MySQL

En suite on crée la base de données Snort:

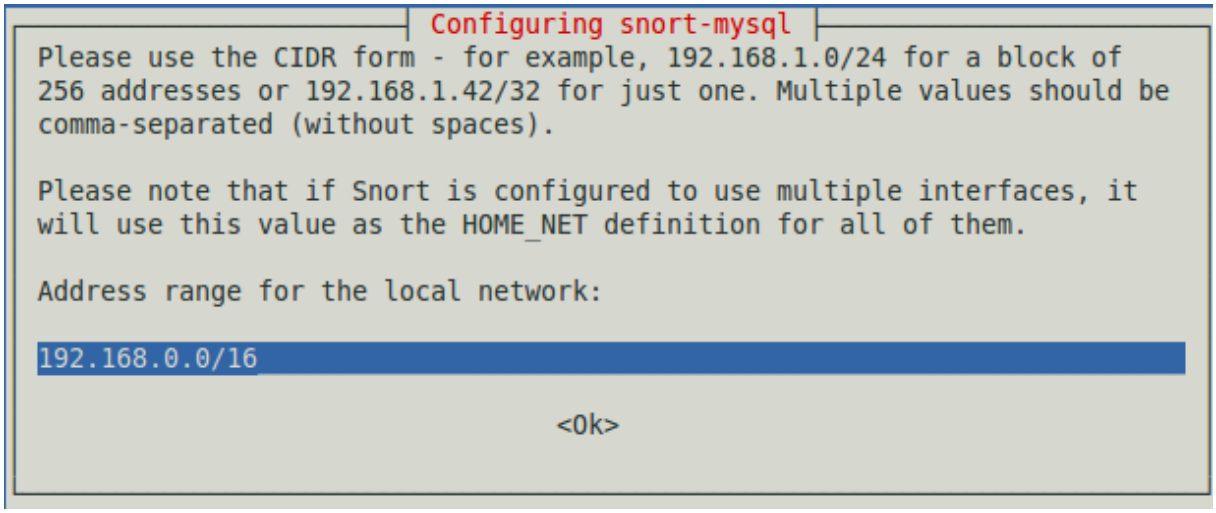
```
Mysql> create database snort;

Mysql> grant CREATE, INSERT, SELECT, DELETE, UPDATE on snort.*
to snort@localhost;

Mysql> SET PASSWORD FOR snort@localhost=PASSWORD('password');

Mysql> exit
```

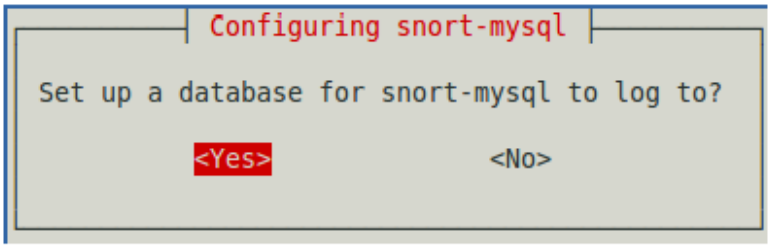


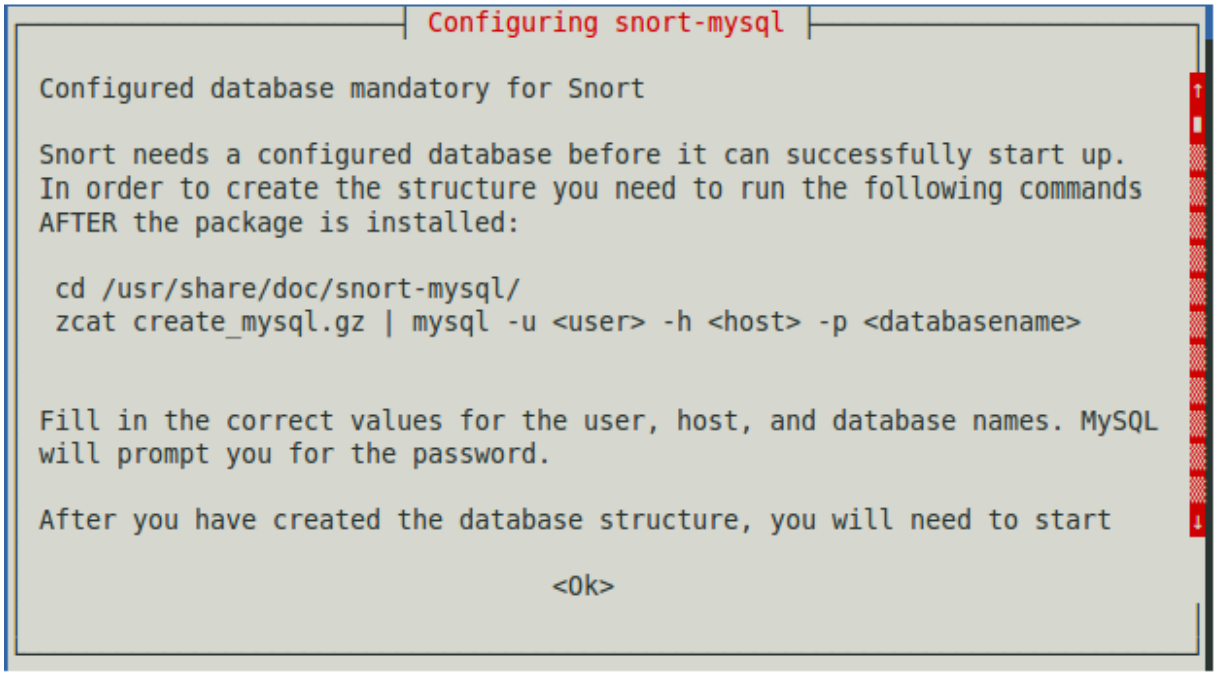
IV.1.4.3 Installation de Snort-Mysql

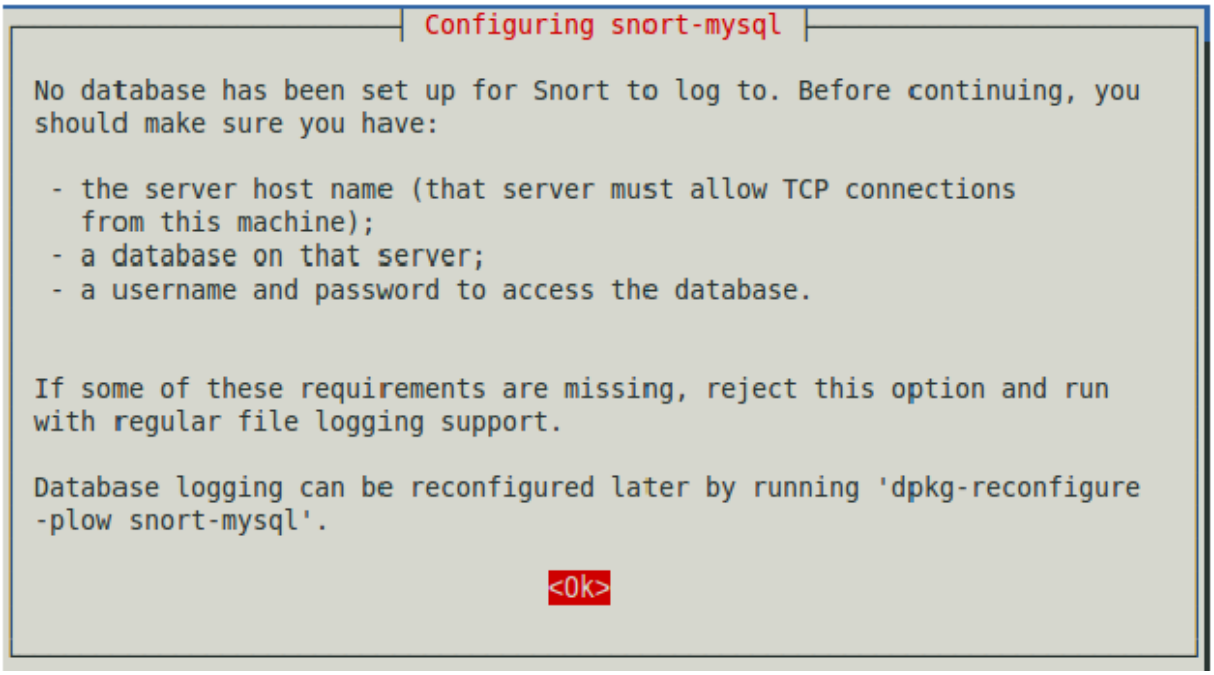
Cette étape consiste à la création des schémas de données pour la base Snort, pour cela, on installe Snort-Mysql :

```
#apt-get install snort-mysql
```









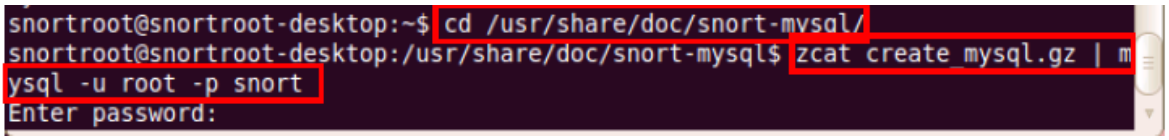
Une fois que nous avons créé les bases de données, nous allons procéder à la création du schéma des données pour la base Snort en rentrant d'abord dans le dossier contenant snort-mysql :

```
# cd /usr/share/doc/snort-mysql/
```

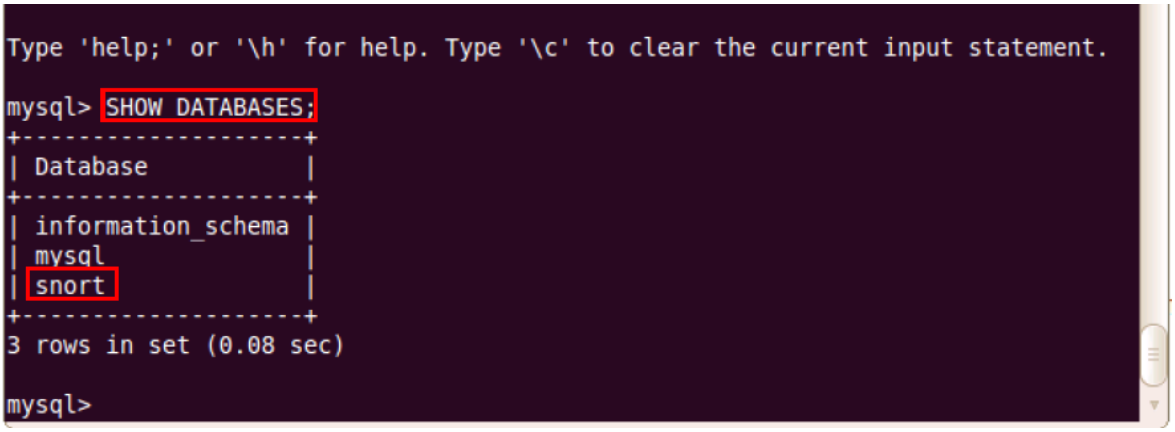
puis, on tape:

```
# zcat create_mysql.gz | mysql -u root -p
```

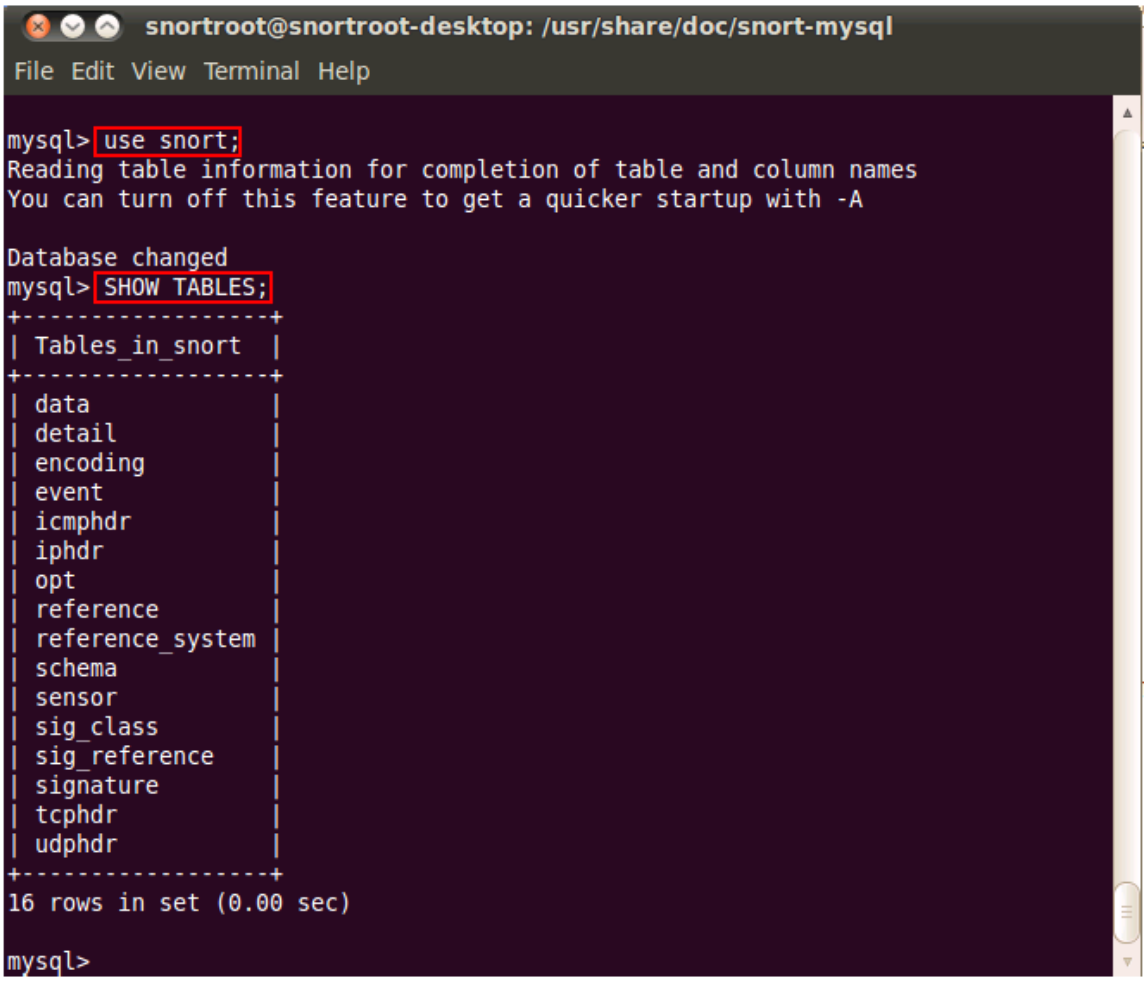
Pour ramener le schema de snort vers la base de données MySQL.



On peut vérifier si les bases sont bien créées et que le schéma a bien été importé :



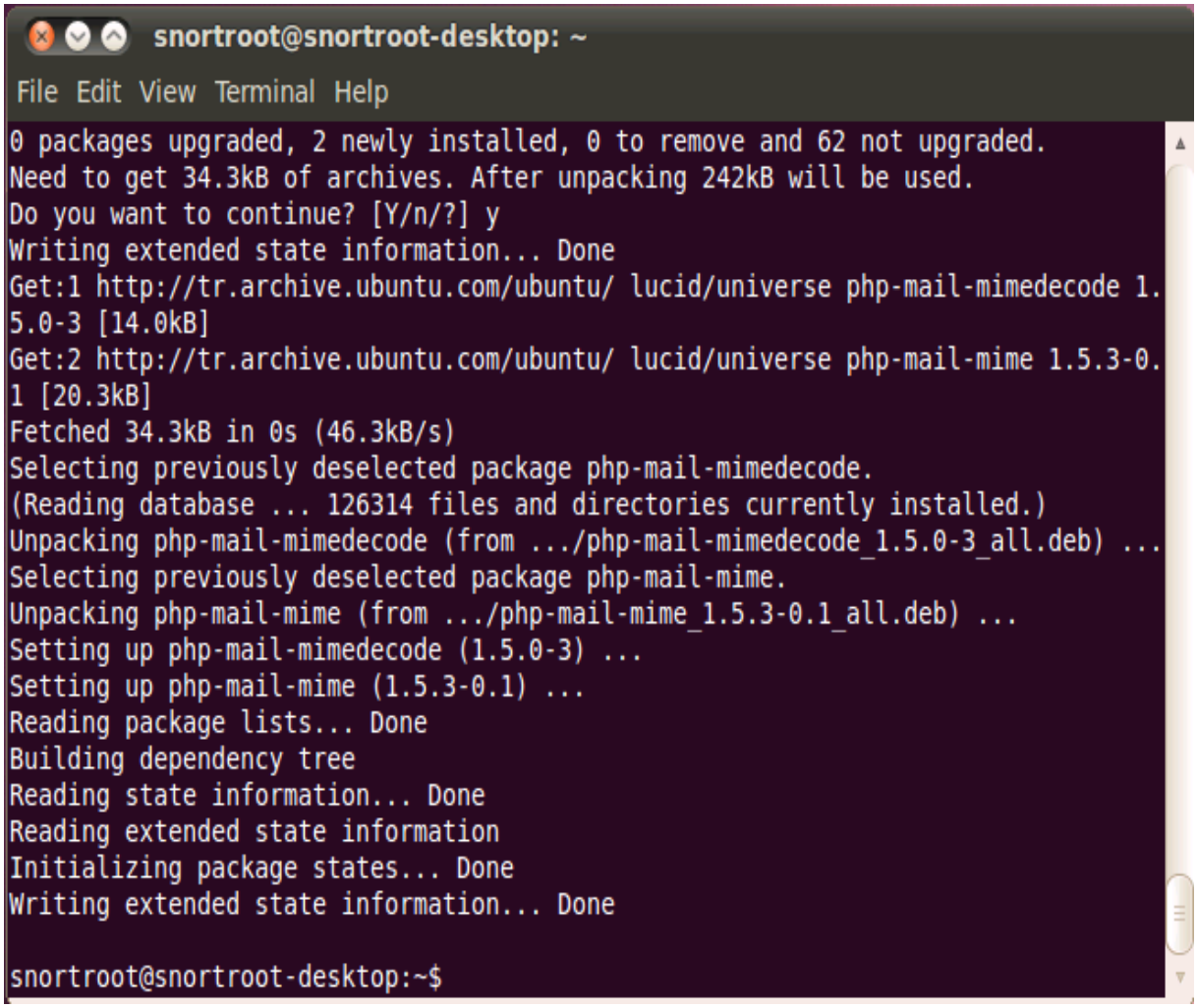
Si tout va bien, on aura la table suivante :



IV.1.5 Mise en place de la console B.A.S.E

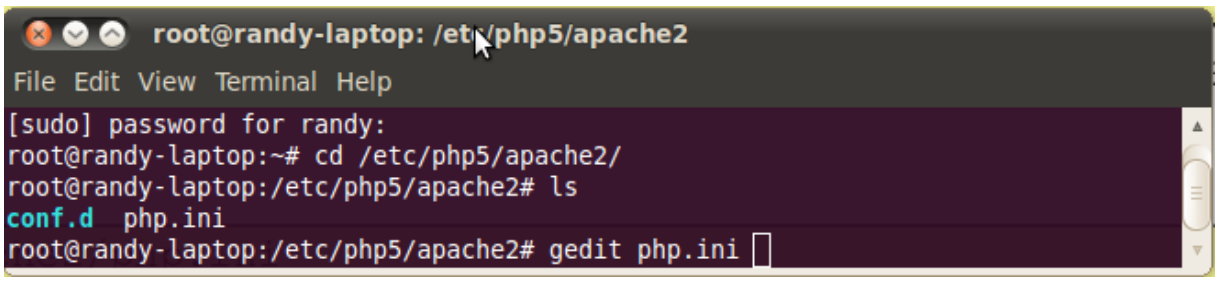
- **Mise en place d'Apache-PHP**

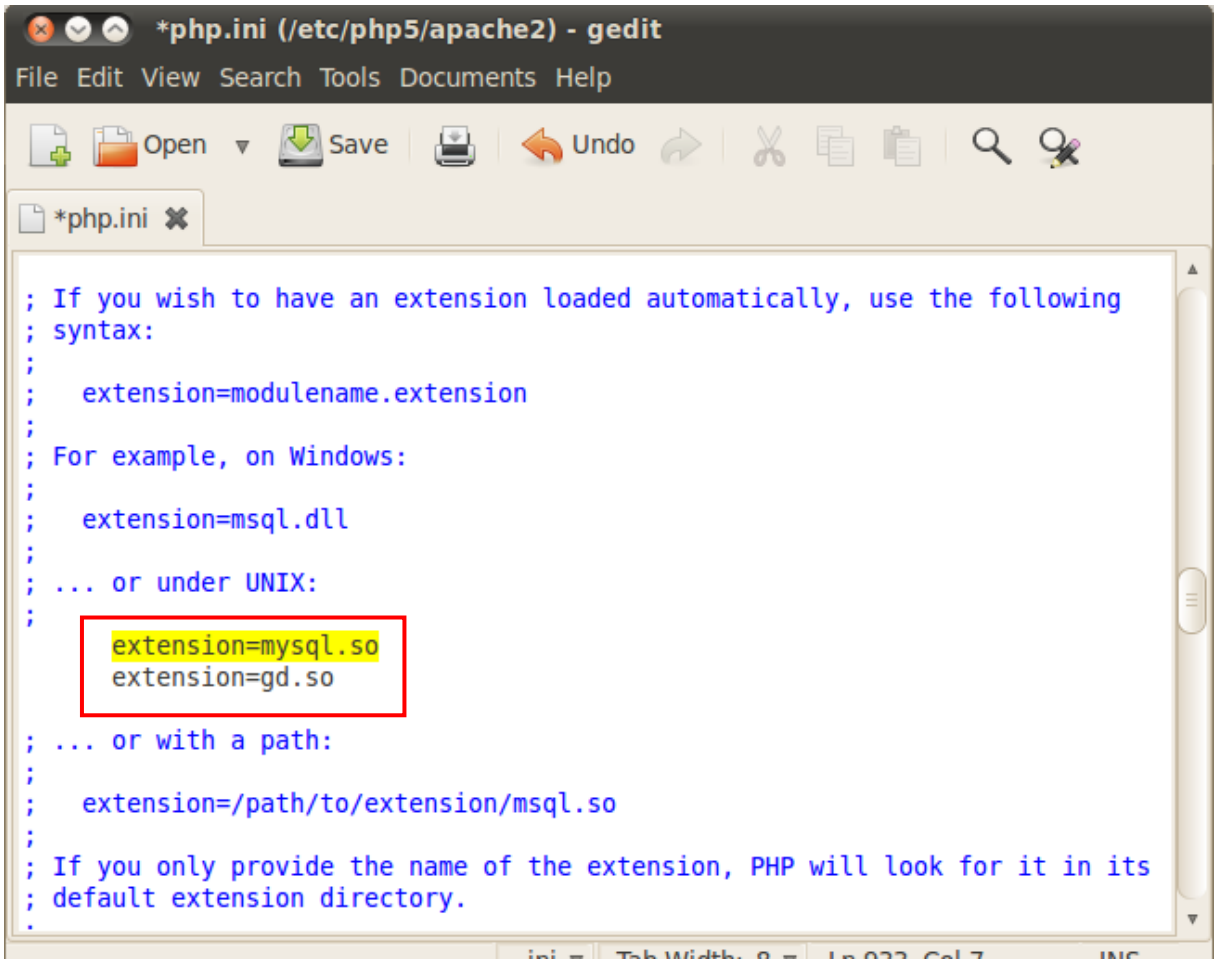
```
# apt-get install apache2 php5 libapache2-mod-
php5 php5-gd php5- mysql libtool libpcre3-dev
php-pear vim ssh openssh-server
```



On configure le fichier php.ini pour que les modifications nécessaires soient apportées à PHP en ajoutant les extensions suivantes :

```
extension=mysql.so

extension=gd.so
```

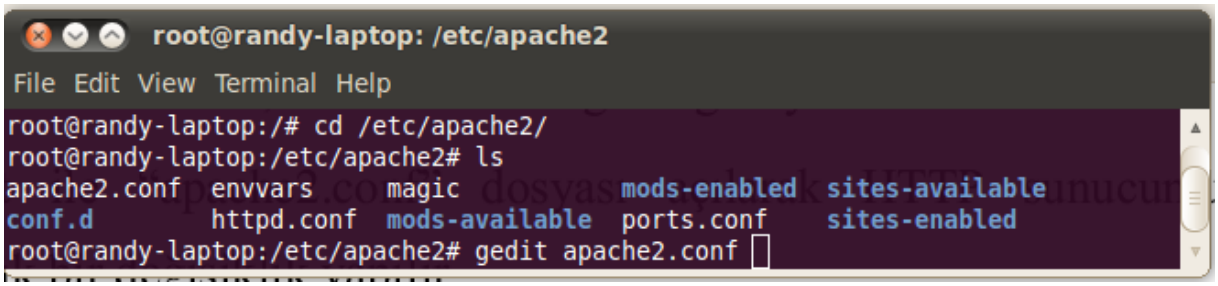

root@randy-laptop: /etc/php5/apache2
File Edit View Terminal Help
[sudo] password for randy:
root@randy-laptop:~# cd /etc/php5/apache2/
root@randy-laptop:/etc/php5/apache2# ls
conf.d php.ini
root@randy-laptop:/etc/php5/apache2# gedit php.ini


*php.ini (/etc/php5/apache2) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
*php.ini
; If you wish to have an extension loaded automatically, use the following
; syntax:
;
;   extension=modulename.extension
;
; For example, on Windows:
;
;   extension=msql.dll
;
; ... or under UNIX:
;
extension=mysql.so
extension=gd.so
; ... or with a path:
;
;   extension=/path/to/extension/msql.so
;
; If you only provide the name of the extension, PHP will look for it in its
; default extension directory.

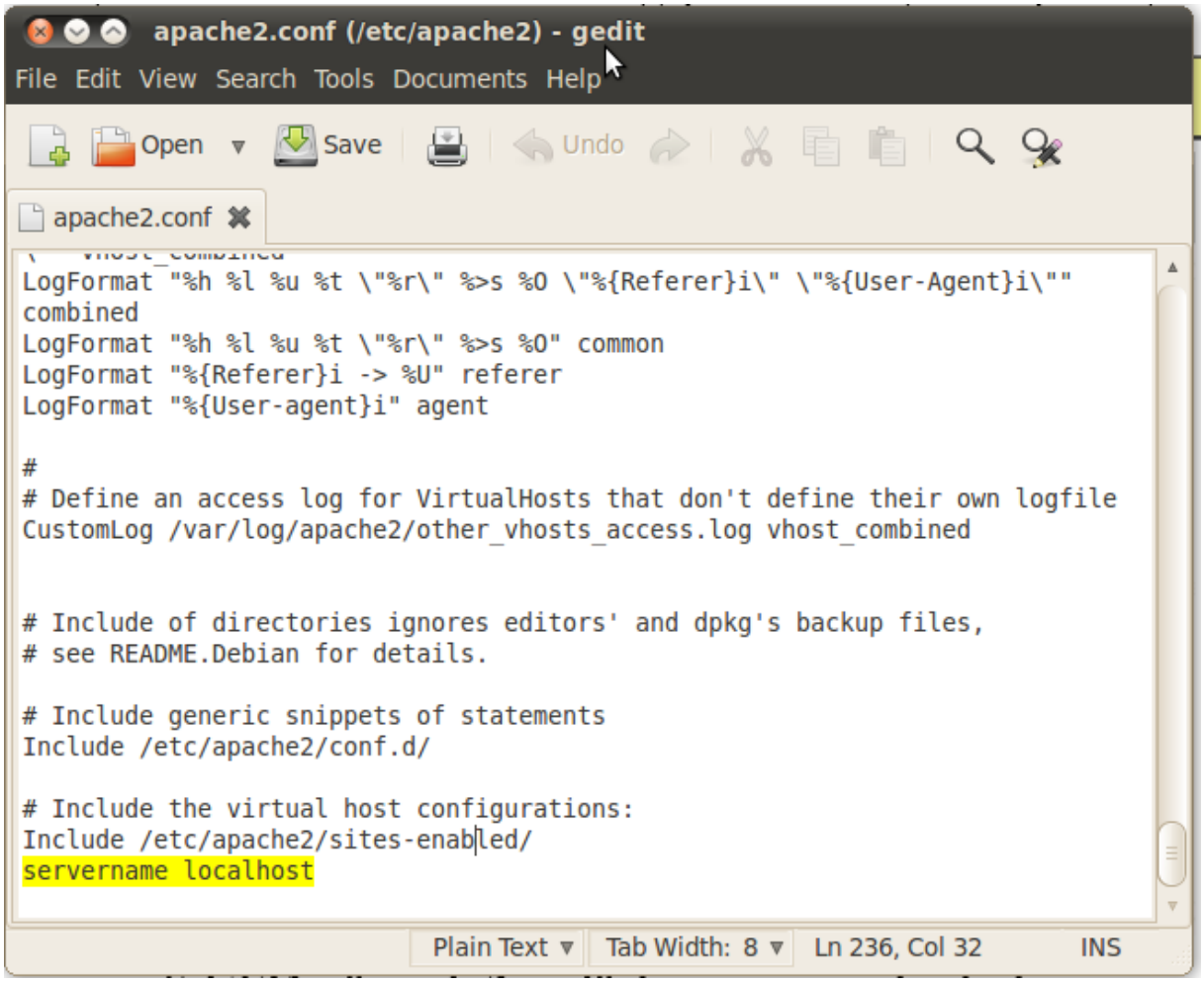
- **Configuration de Apache2**

  On configure le fichier apache2.conf :

```
#gedit /etc/apache2/apache2.conf
```



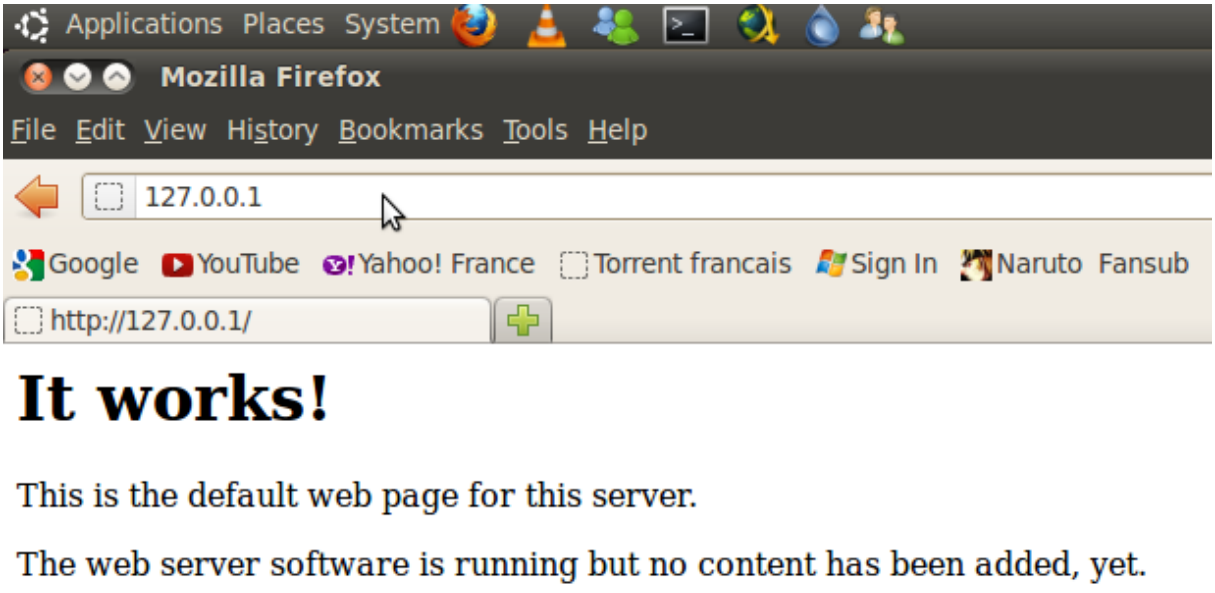
On ajoute le nom du serveur, dans notre cas, c'est « localhost » :



Enfin, on redémarre le service apache :

```
# /etc/init.d/apache2 restart
```

Après, on vérifie le fonctionnement de Appache2 en tapant dans le navigateur web l'adresse 127.0.0.1 (ou directement : localhost) :



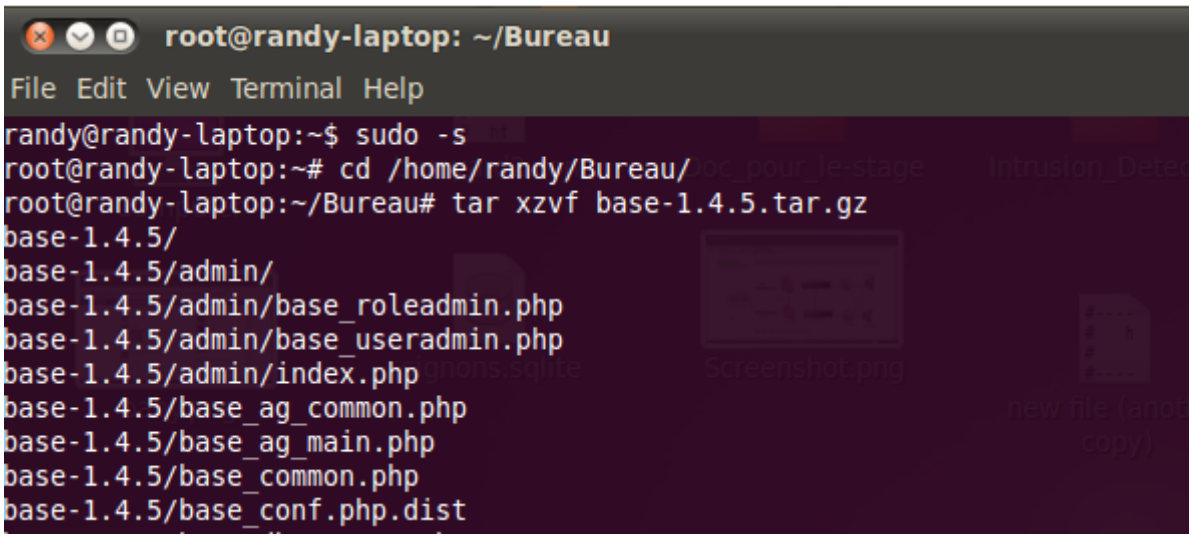
- **Installation de B.A.S.E**

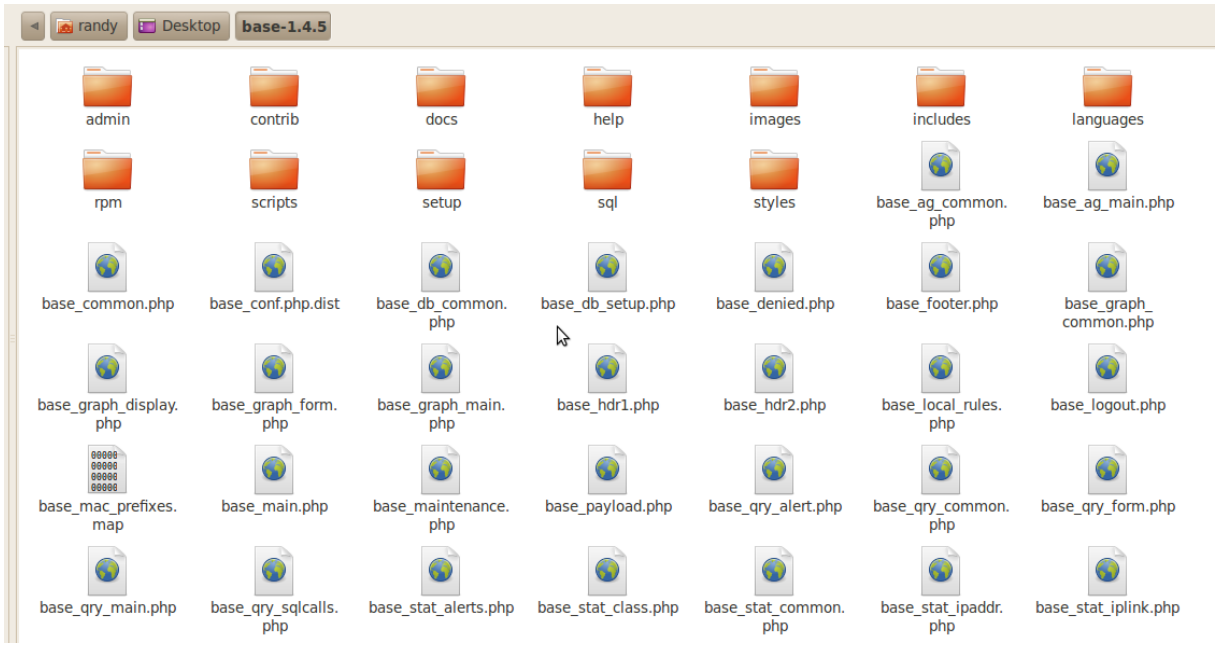
On installe d'abord les dépendances suivantes :

```
# pear install --alldeps Mail

# pear install --alldeps Mail_Mime

# aptitude install php-mail

# aptitude install php-mail-mime
```

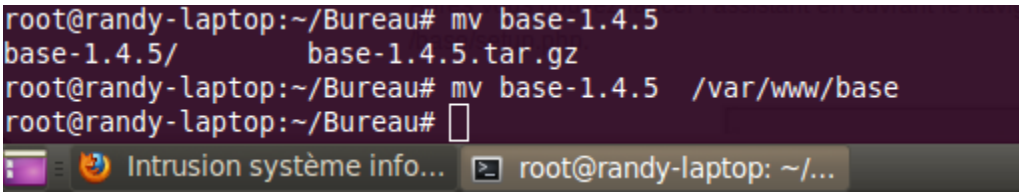
On décompresse l'archive B.A.S.E :

```
tar -xzvf base-1.4.5.tar.gz
```



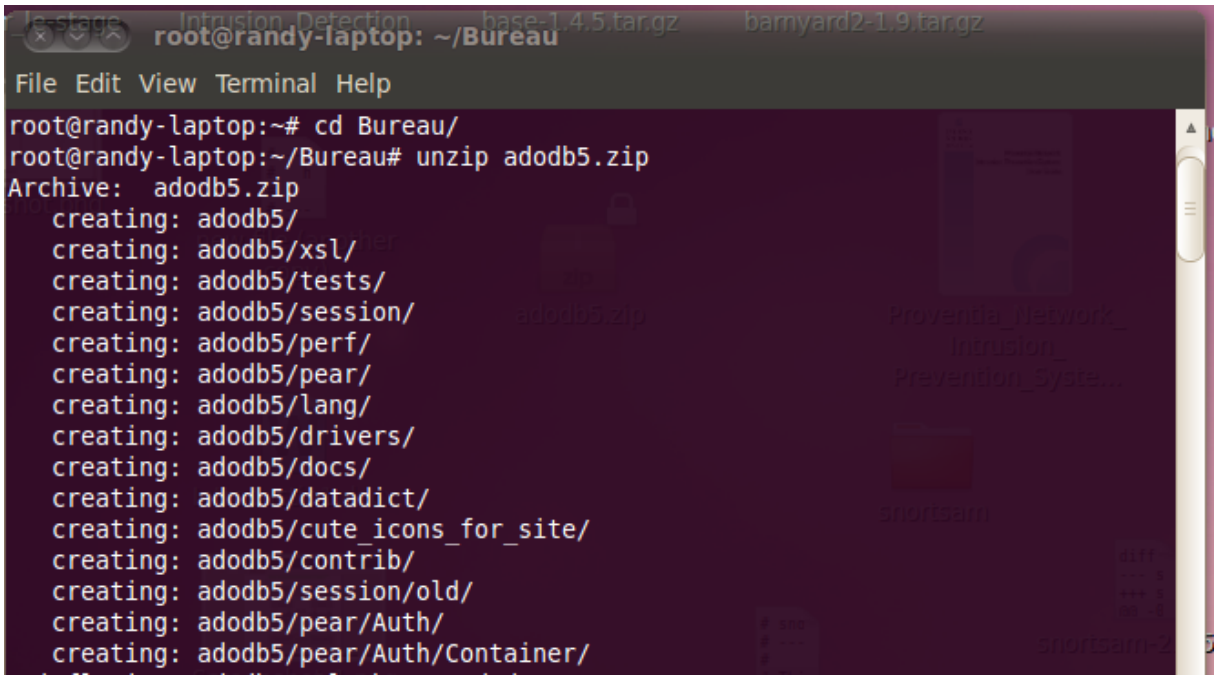


On déplace le dossier BASE décompressé dans le répertoire `/var/www/base`



Maintenant, on installe Adodb

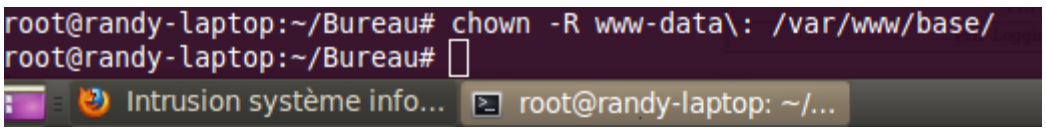
- **Installation de Adodb**
  On décompresse le ficher `adodb5.zip`



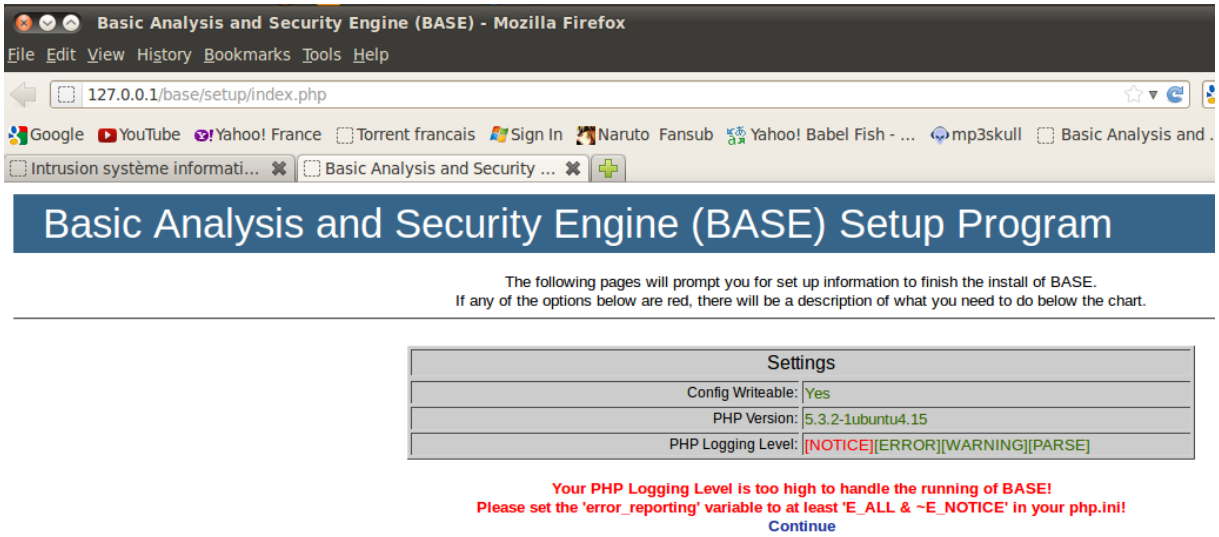
Puis on déplace le dossier Adodb5 vers le répertoire `/var/www/base` :

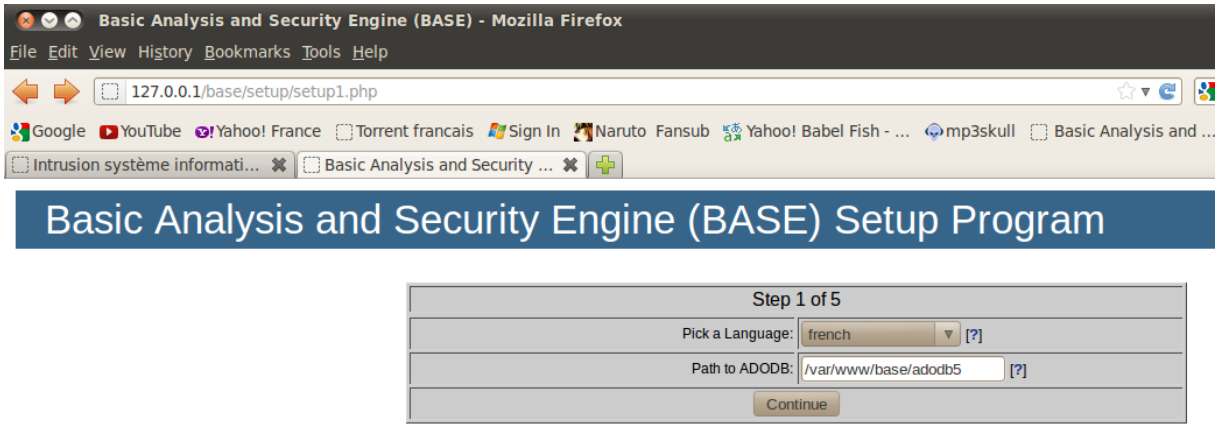
```
#mv adodb5 /var/www/base
```

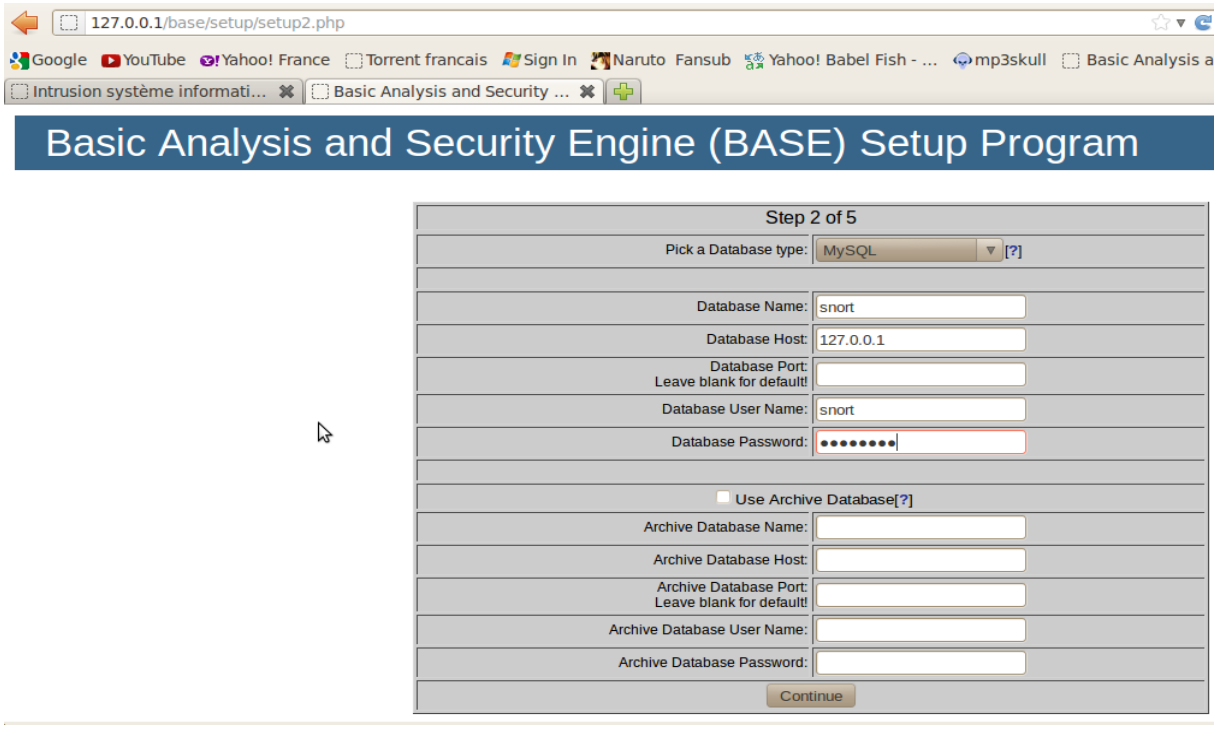
Et on attribut les droits d'écriture a l'utilisateur web dans le contenu du dossier `base`

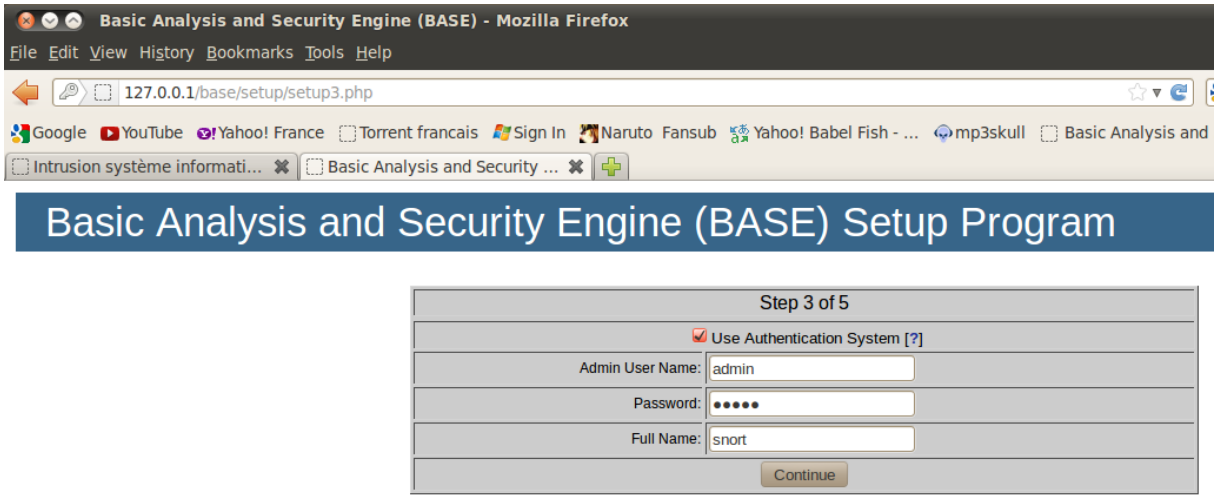


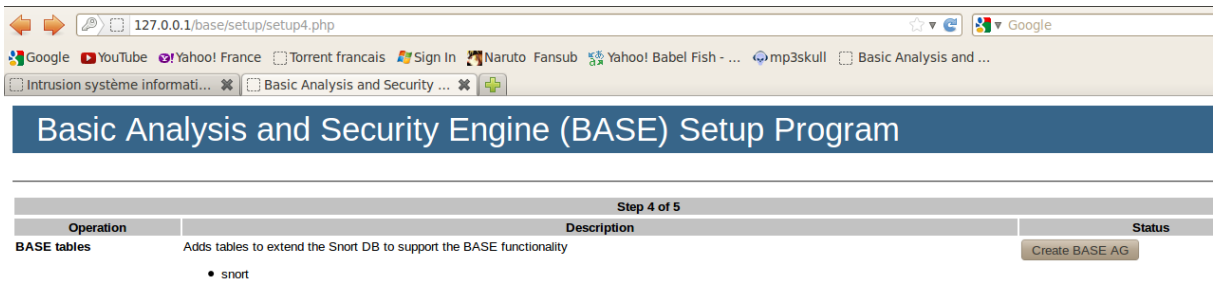
Apres l'installation de B.A.S.E, on lance dans le navigateur : `127.0.0.1/base/`

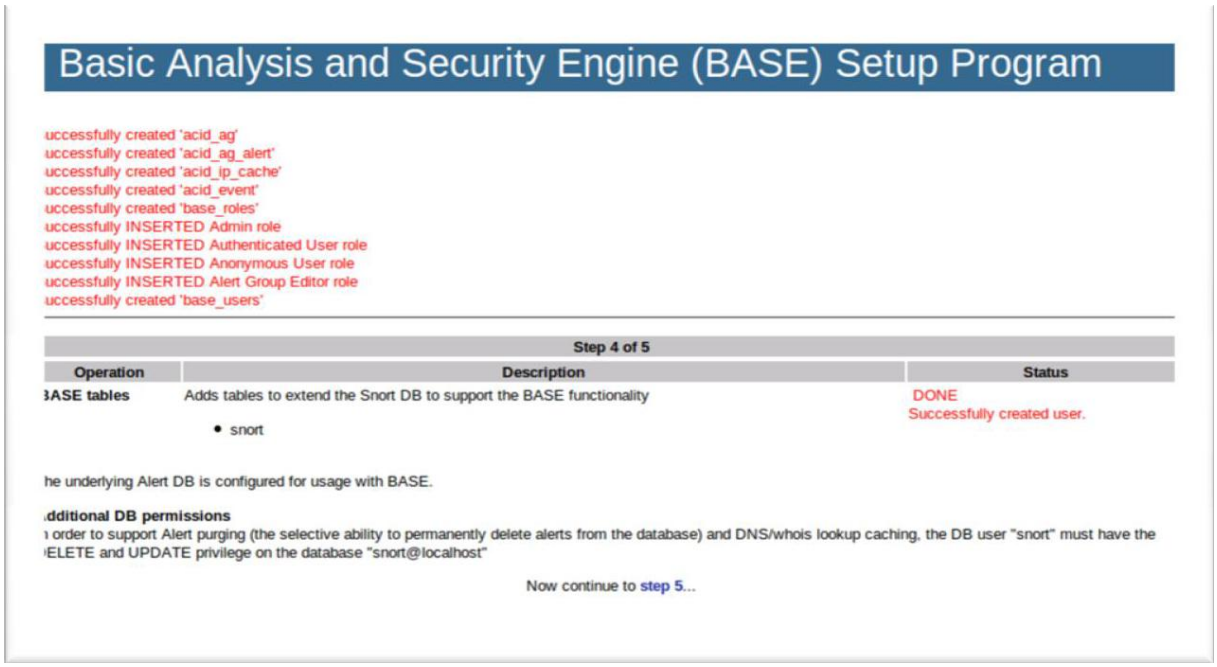


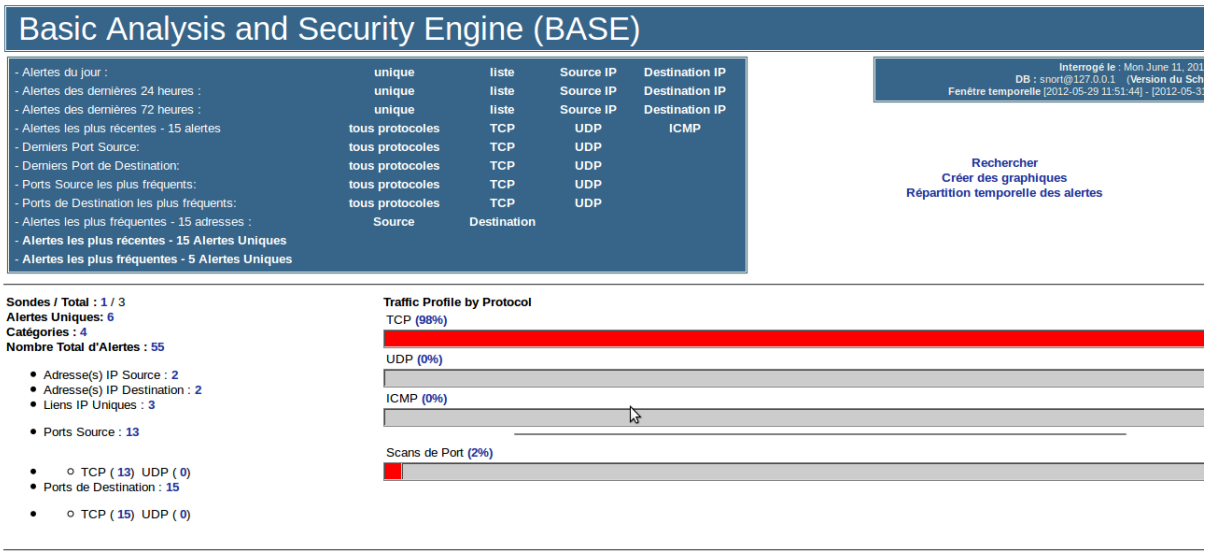



127.0.0.1/base/setup/setup2.php
Basic Analysis and Security Engine (BASE) Setup Program
Step 2 of 5
Pick a Database type: MySQL [?]
Database Name: snort
Database Host: 127.0.0.1
Database Port: Leave blank for default!
Database User Name: snort
Database Password:
Use Archive Database[?]
Archive Database Name:
Archive Database Host:
Archive Database Port: Leave blank for default!
Archive Database User Name:
Archive Database Password:
Continue


Basic Analysis and Security Engine (BASE) - Mozilla Firefox
File Edit View History Bookmarks Tools Help
127.0.0.1/base/setup/setup3.php
Basic Analysis and Security Engine (BASE) Setup Program
Step 3 of 5
Use Authentication System [?]
Admin User Name: admin
Password:
Full Name: snort
Continue


127.0.0.1/base/setup/setup4.php
Basic Analysis and Security Engine (BASE) Setup Program
Step 4 of 5
Operation
Description
Status
BASE tables
Adds tables to extend the Snort DB to support the BASE functionality
Create BASE AG
snort


Basic Analysis and Security Engine (BASE) Setup Program
uccessfully created 'acid_ag'
uccessfully created 'acid_ag_alert'
uccessfully created 'acid_ip_cache'
uccessfully created 'acid_event'
uccessfully created 'base_roles'
uccessfully INSERTED Admin role
uccessfully INSERTED Authenticated User role
uccessfully INSERTED Anonymous User role
uccessfully INSERTED Alert Group Editor role
uccessfully created 'base_users'
Step 4 of 5
Operation
Description
Status
ASE tables
Adds tables to extend the Snort DB to support the BASE functionality
DONE
Successfully created user.
snort
he underlying Alert DB is configured for usage with BASE.
dditional DB permissions
order to support Alert purging (the selective ability to permanently delete alerts from the database) and DNS/whois lookup caching, the DB user "snort" must have the
ELETE and UPDATE privilege on the database "snort@localhost"
Now continue to step 5...


Basic Analysis and Security Engine (BASE)
- Alertes du jour : unique liste Source IP Destination IP
- Alertes des dernières 24 heures : unique liste Source IP Destination IP
- Alertes des dernières 72 heures : unique liste Source IP Destination IP
- Alertes les plus récentes - 15 alertes tous protocoles TCP UDP ICMP
- Derniers Port Source: tous protocoles TCP UDP
- Derniers Port de Destination: tous protocoles TCP UDP
- Ports Source les plus fréquents: tous protocoles TCP UDP
- Ports de Destination les plus fréquents: tous protocoles TCP UDP
- Alertes les plus fréquentes - 15 adresses : Source Destination
- Alertes les plus récentes - 15 Alertes Uniques
- Alertes les plus fréquentes - 5 Alertes Uniques
Interrogé le : Mon June 11, 201
DB : snort@127.0.0.1 (Version du Sch
Fenêtre temporelle [2012-05-29 11:51:44] - [2012-05-3
Rechercher
Créer des graphiques
Répartition temporelle des alertes
Sondes / Total : 1 / 3
Alertes Uniques: 6
Catégories : 4
Nombre Total d'Alertes : 55
Adresse(s) IP Source : 2
Adresse(s) IP Destination : 2
Liens IP Uniques : 3
Ports Source : 13
TCP (13) UDP (0)
Ports de Destination : 15
TCP (15) UDP (0)
Traffic Profile by Protocol
TCP (98%)
UDP (0%)
ICMP (0%)
Scans de Port (2%)

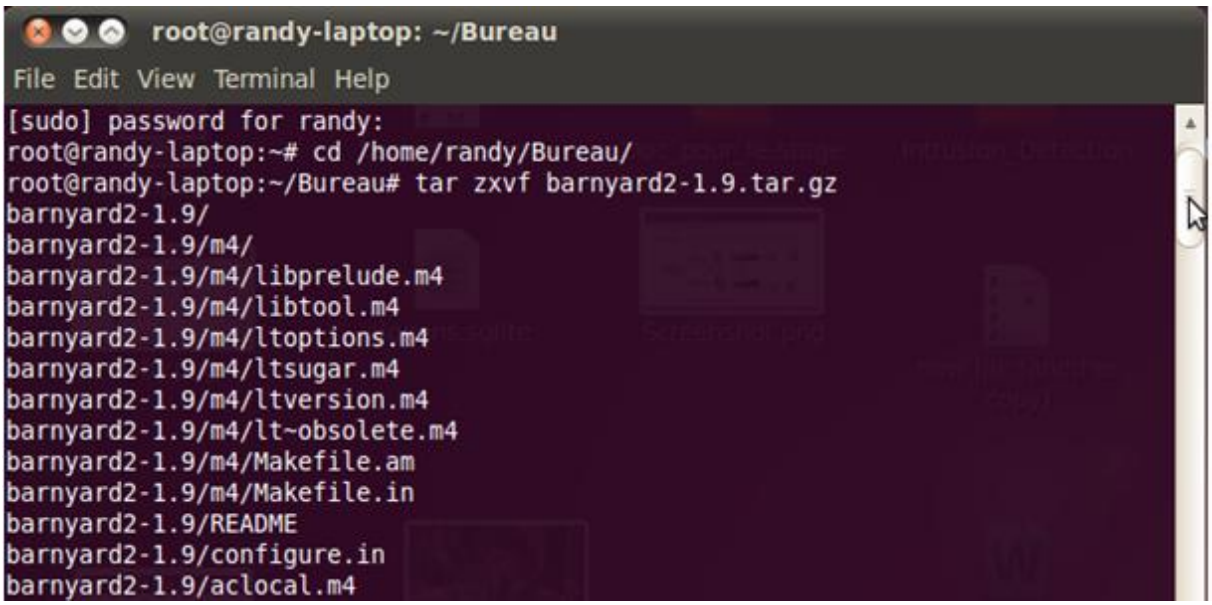
IV.1.6 Mise en place de barnyard

IV.1.6.1 installation de barnyard

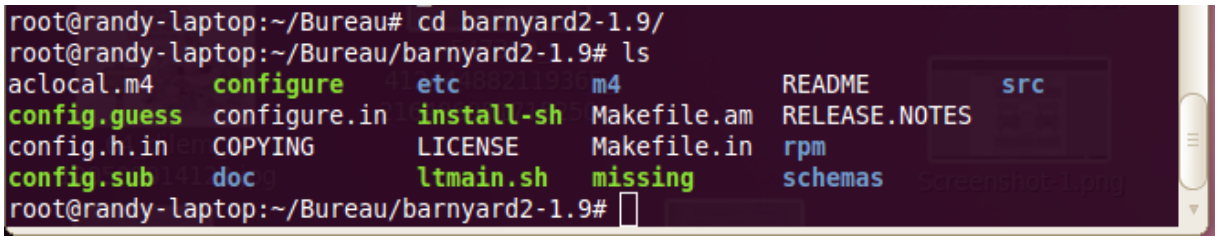
A partir du shell en exécute les commandes suivantes :

```
# wget -O barnyard2-1.7.tar.gz \

http://www.securixlive.com/download/barnyard
2/barnyard2-1.7.tar.gz

# tar zxvf barnyard2-1.7.tar.gz

# cd barnyard2-1.7

# ./configure --with-mysql

# make

# make install

# sudo cp etc/barnyard2.conf /etc/snort
```

Decompression de Barnyard:

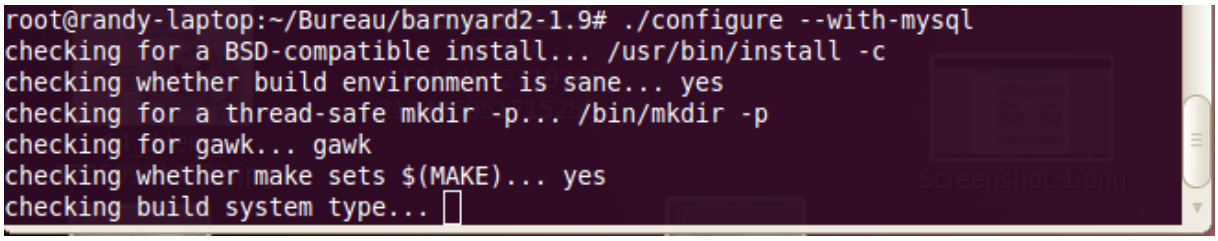


On verifie le contenu de Barnyard a l'aide de la commande `ls` :

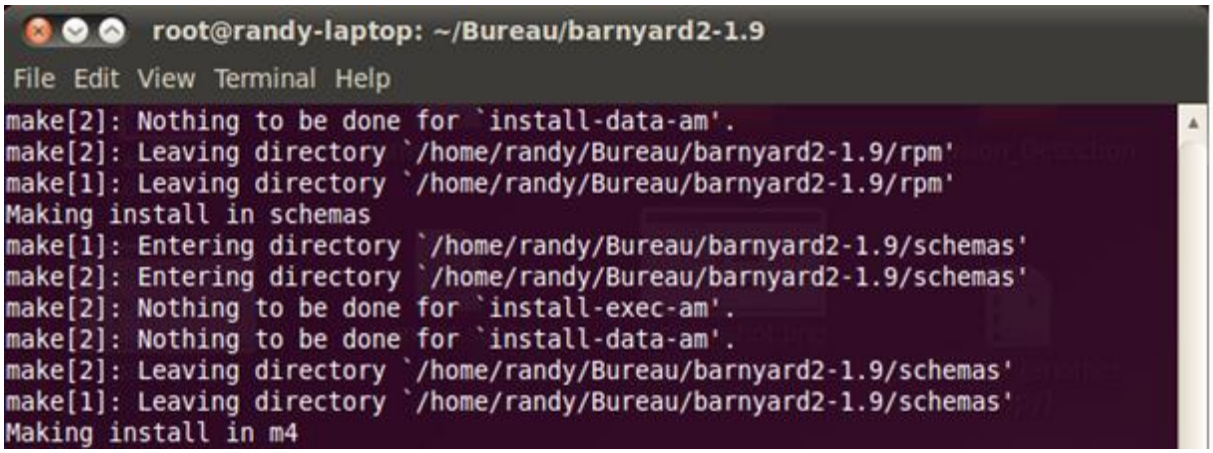


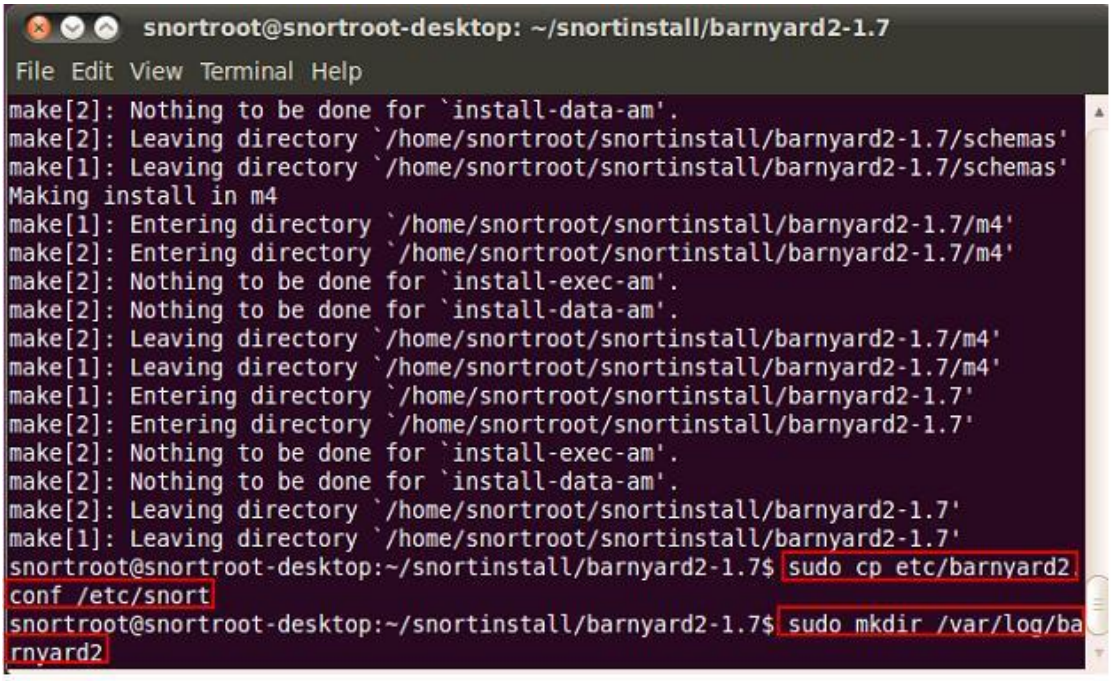
Lancement de l'installation :

```
./configure –with-mysql
```




```
make
make install
```





Enfin, on copie le le fichier barnyard2.conf vers le répertoire `/etc/snort` aifn de paramétrer Snort avec barnyard2 :

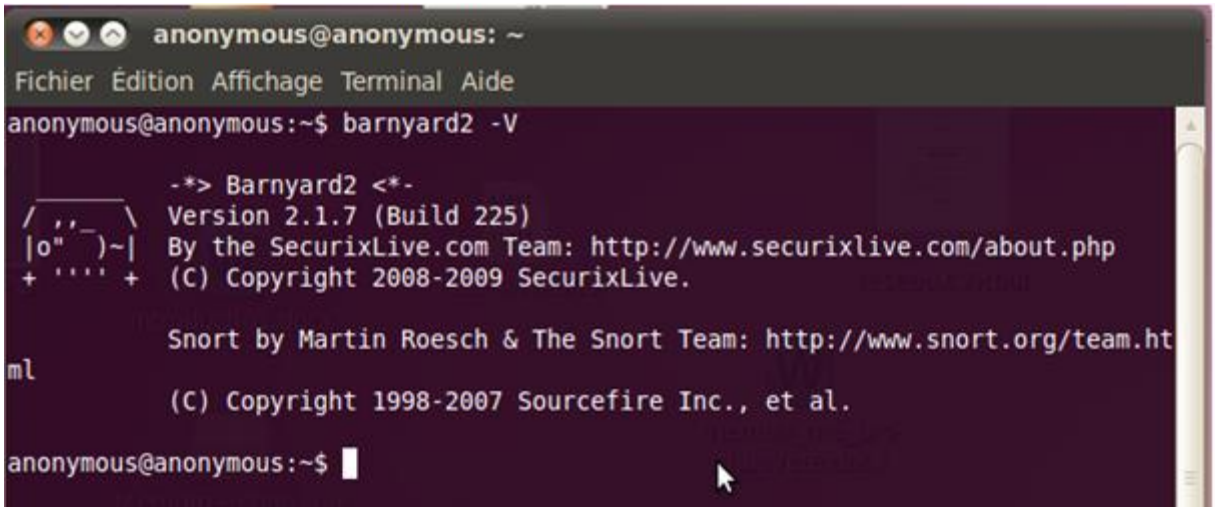
```
# cp etc/barnyard2.conf /etc/snort
```

Et on crée un dossier ou Barnyard2 stocke les logs :

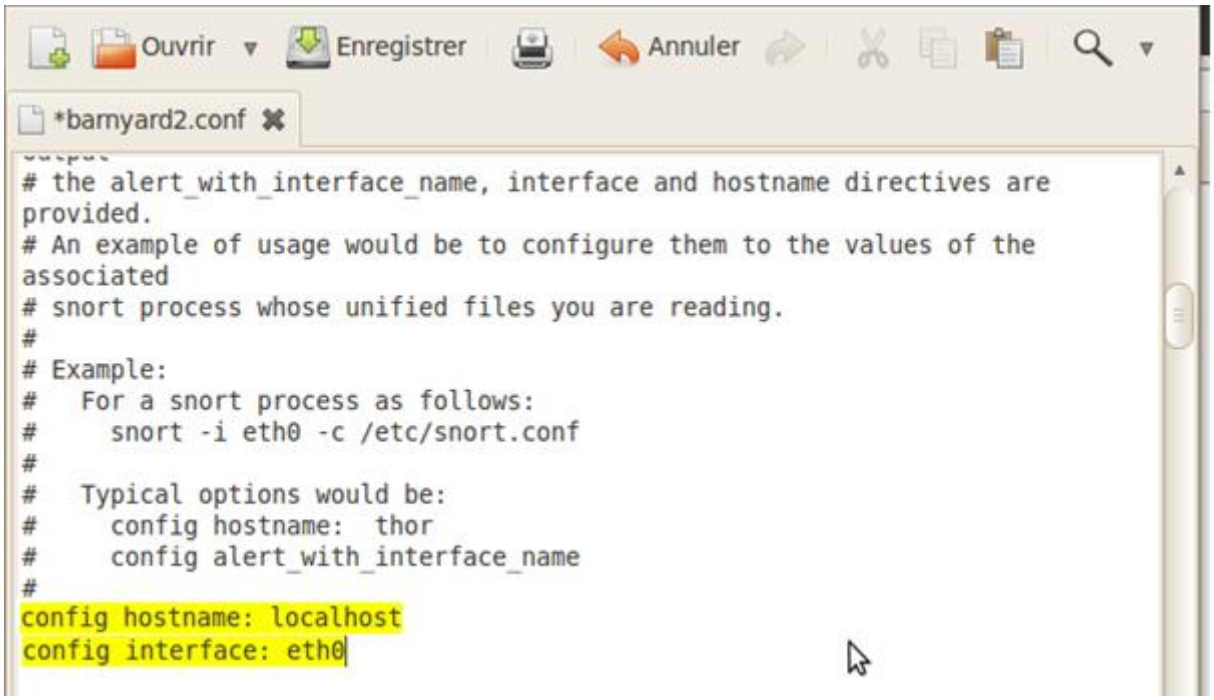
```
# mkdir /var/log/barnyard2
```

Apres l'installation, on vérifié si tout est bon en tapant la version du Barnyard :

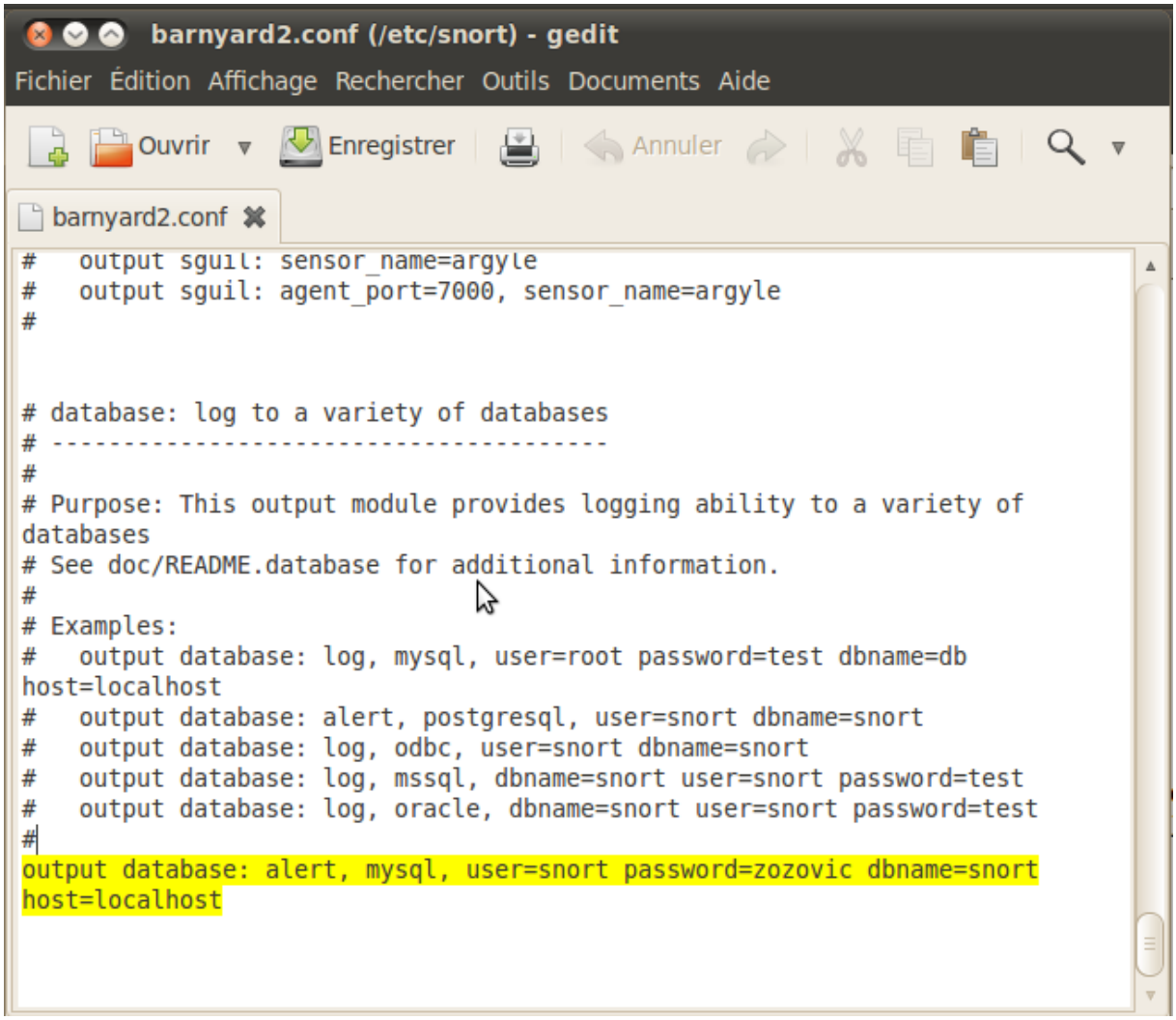
```
Barnyard -V
```



On configure le fichier `barnyard.conf` pour y ajouter le nom du hôte '`localhost`' est l'interface '`eth0`'

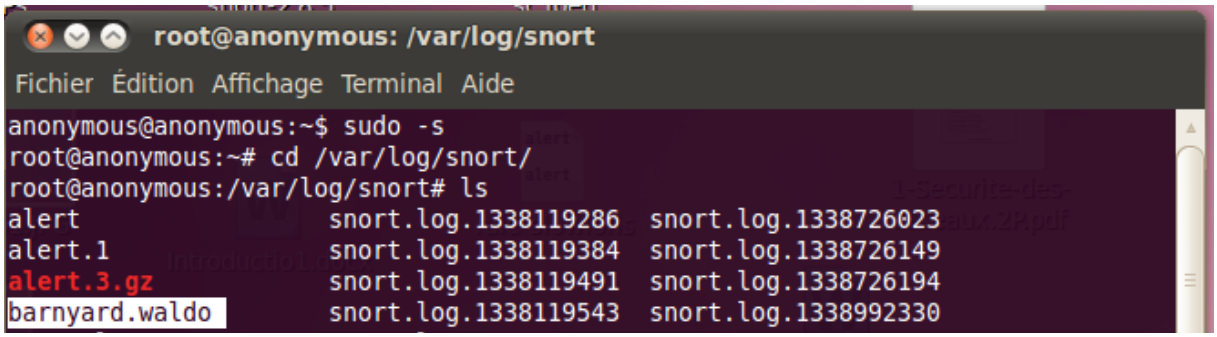


Puis, on configure la sortie vers la base de données MySQL :

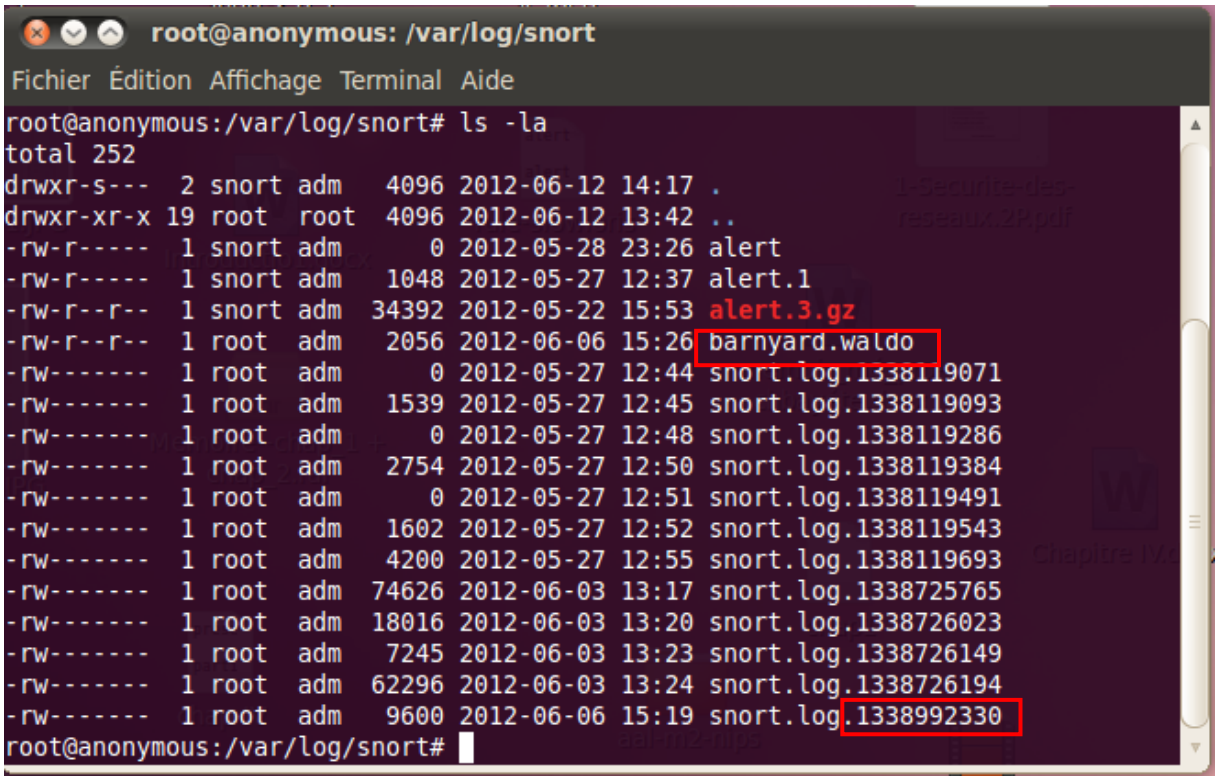


Cette étape a pour but de synchroniser Snort avec Barnyard :

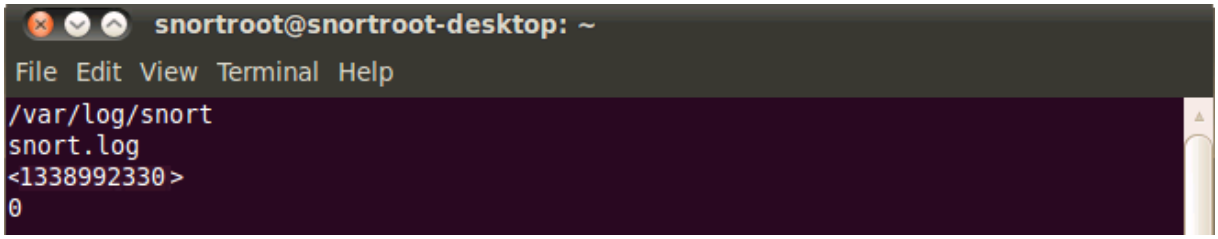
On crée un fichier pourtant le nom de `barnyard.waldo` dans le répertoire : `/var/log/snort`



On fait `ls -la /var/log/`snort pour récuperer le dernier timestamp (le dernier en date):



Puis on l'injecter dans le fichier `barnyard.walo`



IV.7 Extension de Snort en NIPS

IV.7.1 Application du patch SnortSam a Snort

On récupere le patch ncéssaire a la version de l'IDS Snort (dans notre cas V 2.8.5.2) Via l'URL :
http://www.snortsam.net/files/snort-plugin/snortsam-2.8.6.diff.gz

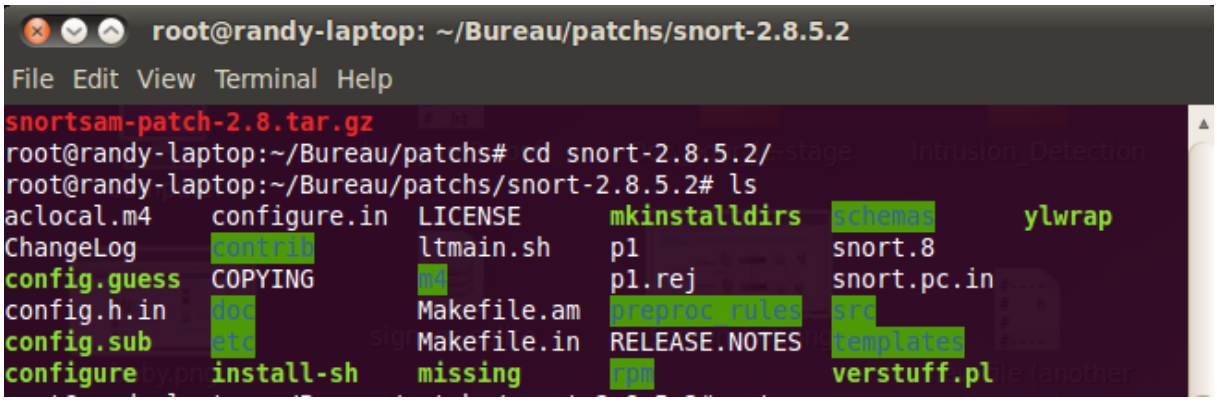
On décompresse :

```
# gunzip snortsam-2.8.6.diff.gz
```

snortsam-2.8.5.diff

```
diff -ruN snort-2.8.4.1.orig/autojunk.sh snort-2.8.4.1/autojunk.sh
--- snort-2.8.4.1.orig/autojunk.sh      1970-01-01 03:30:00.000000000 +0330
+++ snort-2.8.4.1/autojunk.sh   2009-06-23 16:40:44.000000000 +0430
@@ -0,0 +1,7 @@
+#!/bin/sh
+# the list of commands that need to run before we do a compile
+libtoolize --automake --copy
+aclocal -I m4
+autoheader
+automake --add-missing --copy
+autoconf
diff -ruN snort-2.8.4.1.orig/src/Makefile.am snort-2.8.4.1/src/Makefile.am
--- snort-2.8.4.1.orig/src/Makefile.am  2009-06-23 16:40:16.000000000 +0430
+++ snort-2.8.4.1/src/Makefile.am       2009-06-23 16:40:44.000000000 +0430
@@ -51,7 +51,8 @@
 sf_types.h \
 log_text.c log_text.h \
 detection_filter.c detection_filter.h \
-rate_filter.c rate_filter.h
+rate_filter.c rate_filter.h \
+twofish.c twofish.h

 snort_LDADD = output-plugins/libspo.a \
 detection-plugins/libspd.a            \
diff -ruN snort-2.8.4.1.orig/src/fatal.h snort-2.8.4.1/src/fatal.h
--- snort-2.8.4.1.orig/src/fatal.h      1970-01-01 03:30:00.000000000 +0330
+++ snort-2.8.4.1/src/fatal.h   2009-06-23 16:40:44.000000000 +0430
```

On localise le répértoire source de Snort, `cd snort-2.8.5.2/`



Puis, on lance le patch comme suit :

```
# patch -p1 < /home/randy/Bureau/snortsam-2.8.6.diff
```

```
root@randy-laptop:~/Bureau/patchs/snort-2.8.5.2# patch -p1 < /home/randy/Bureau/
snortsam-2.8.5.diff
patching file autojunk.sh
patching file src/Makefile.am
patching file src/fatal.h
patching file src/output-plugins/Makefile.am
patching file src/output-plugins/spo_alert_fwsam.c
patching file src/output-plugins/spo_alert_fwsam.h
patching file src/plugbase.c
patching file src/plugin_enum.h
patching file src/twofish.c
patching file src/twofish.h
root@randy-laptop:~/Bureau/patchs/snort-2.8.5.2#
```

A la fin du patch, le fichier autojunk.sh se crée :

```
root@randy-laptop:~/Bureau/patchs/snort-2.8.5.2# ls -la
total 2204
drwxrwxrwx 11 randy randy   4096 2012-06-11 12:22 .
drwxr-xr-x  5 randy randy   4096 2012-06-11 12:13 ..
-rw-r--r--  1 randy randy 263373 2009-10-19 22:17 aclocal.m4
-rw-r--r--  1 root  root     166 2012-06-11 12:22 autojunk.sh
```

On recompile Snort:

```
# ./autojunk
# ./configure –with-mysql
# make
# make install
```

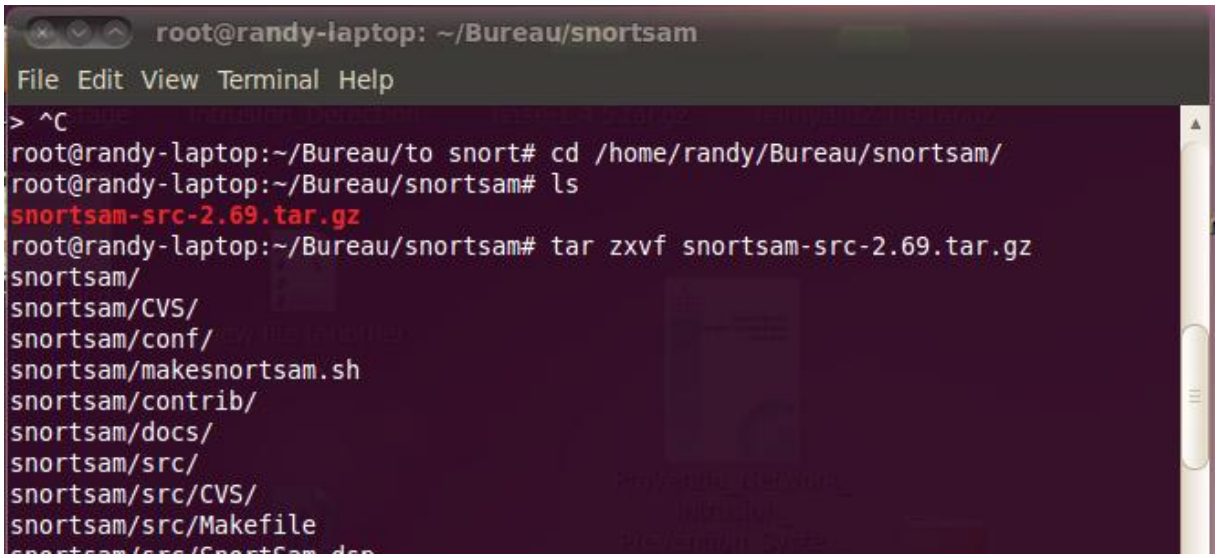
IV.7.2 Mise en place de l'agent SnortSam

IV.7.2.1 Installation de l'agent SnortSam

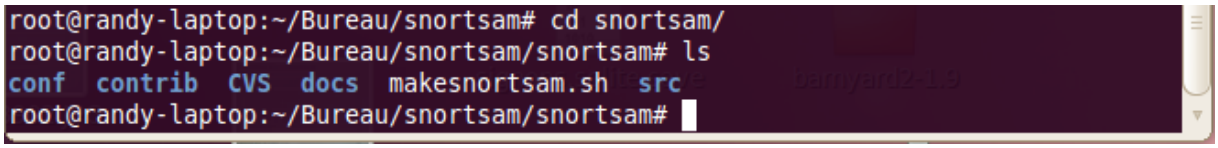
On récupère l'archive snortsam depuis l'URL :
http://www.snortsam.net/files

On écompresse avec :

```
tar xzvf snortsam-src-2.69.tar.gz
```

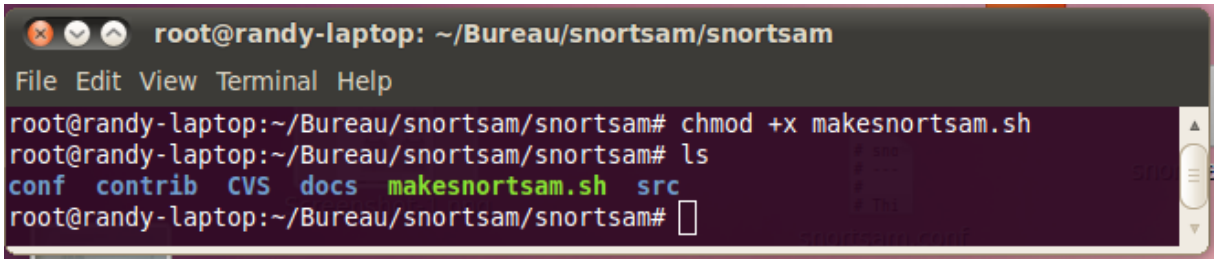


On verifié le contenu du dossier snortsam avec `ls`



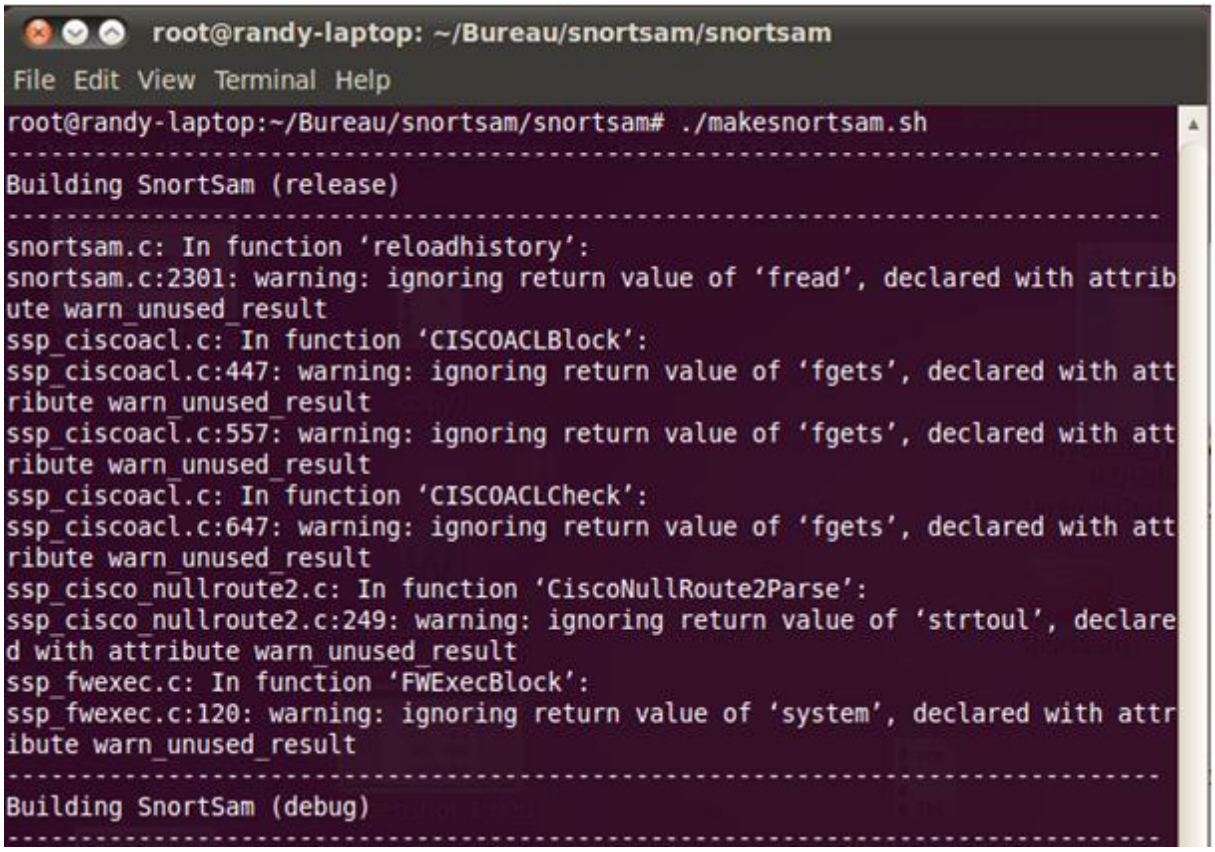
On attribut les droits d'exécution au fichier `makesnortsam.sh` avec la commande :

```
chmod +x makesnortsam.sh
```



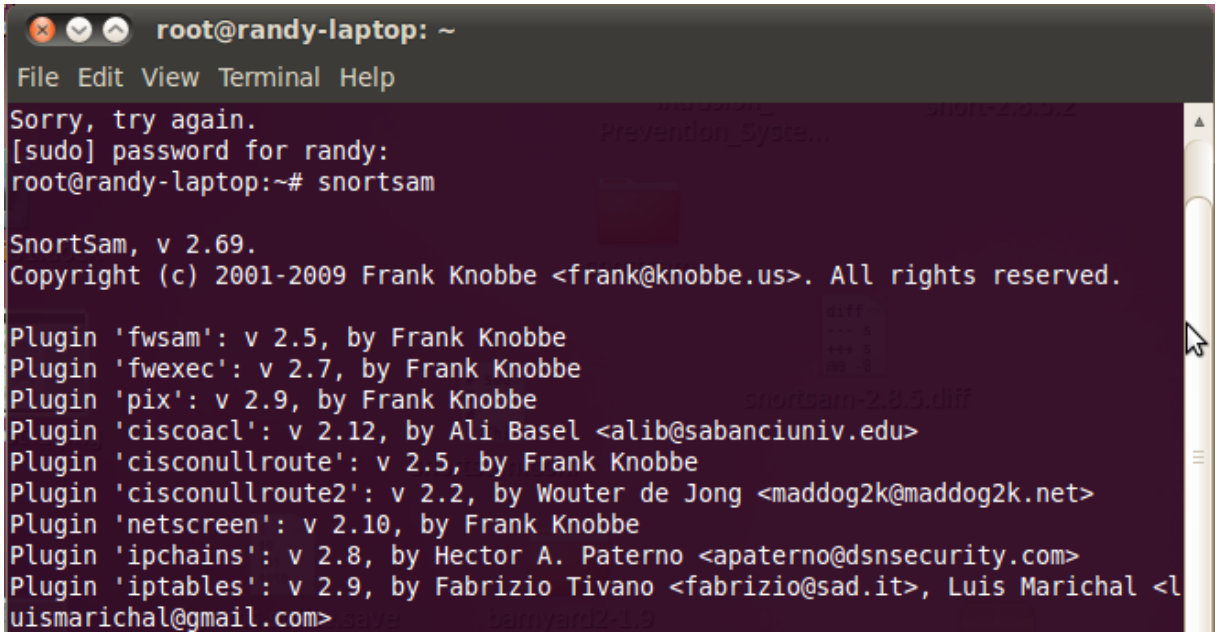
On installe snortsam avec :

```
./makesnortsam.sh
```



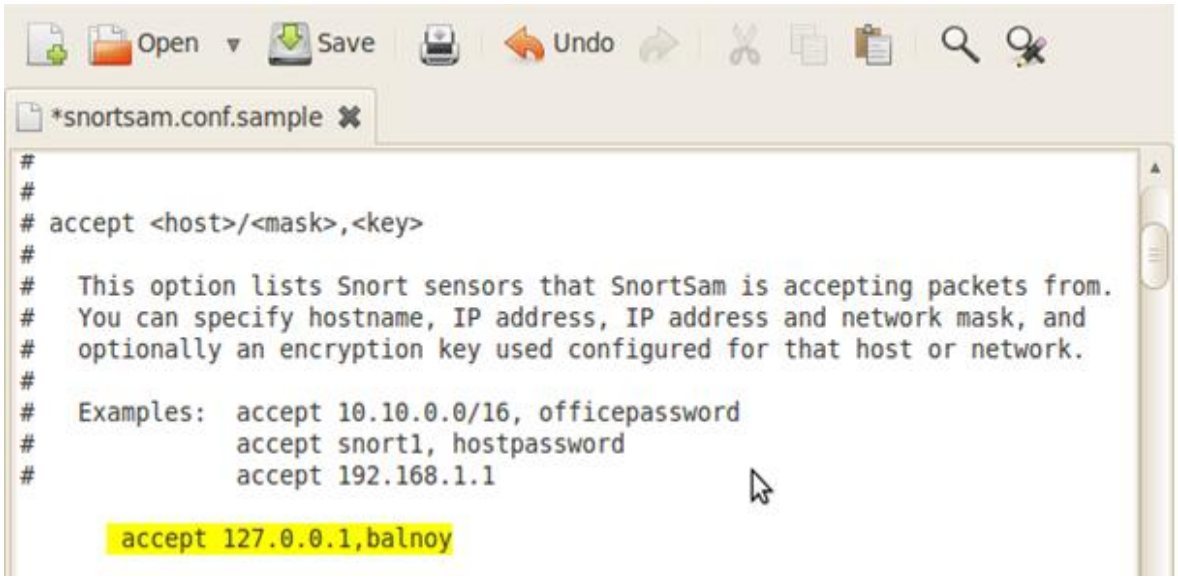
On vérifie si l'installation a pris position :

```
# snortsam -V
```

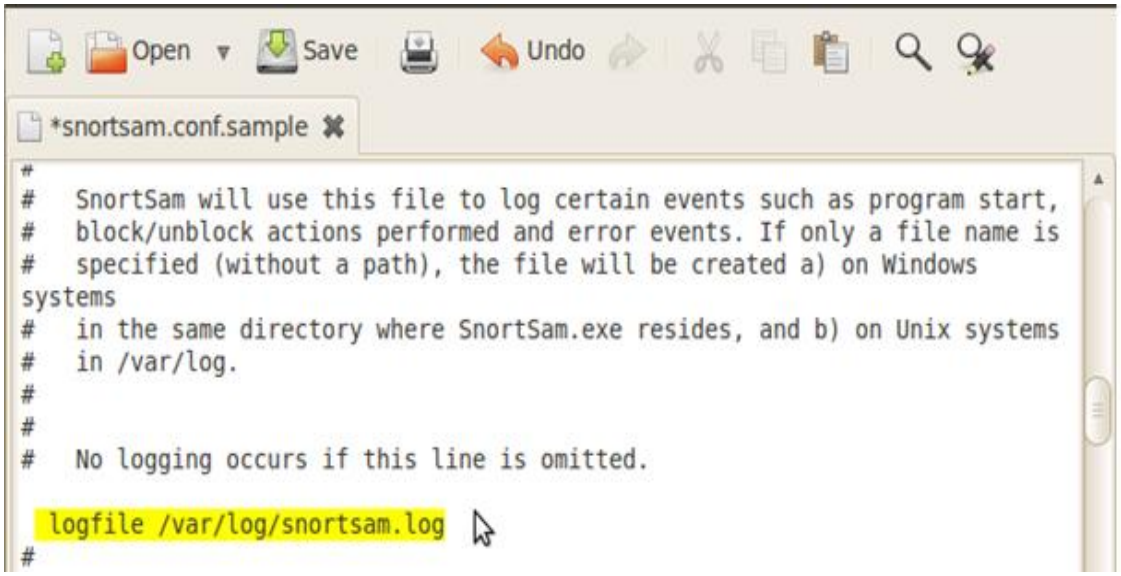


IV.7.2.2 Configuration de SnortSam

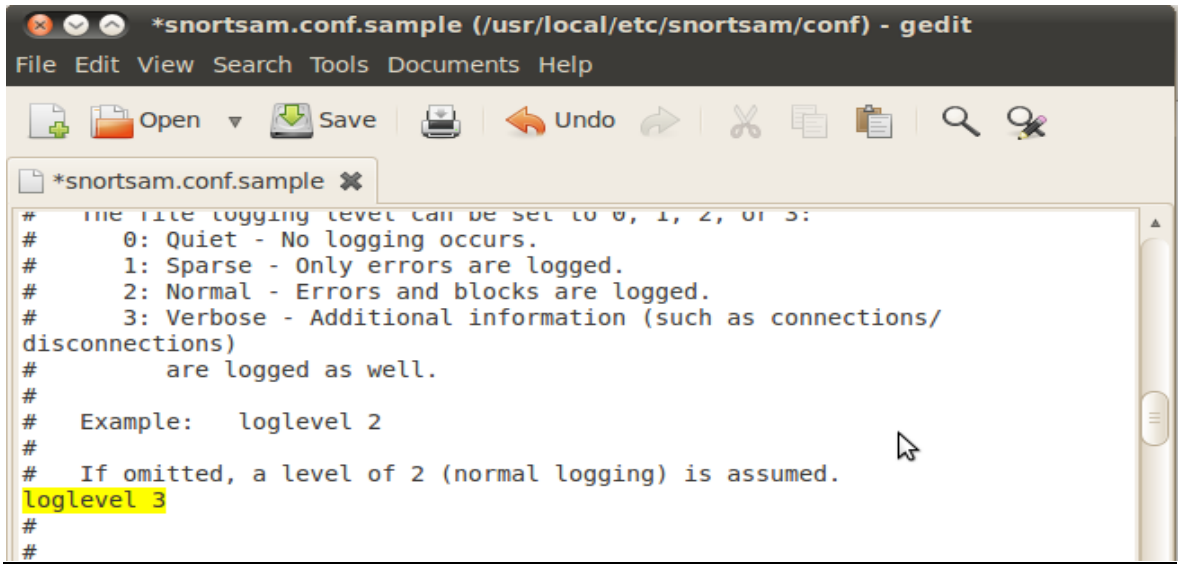
Dans le fichier de configuration snortsam.conf on ajoute certaines configurations. Dans notre cas, l'agent SnortSam est installé dans le meme hote que snort :
On précise l'adresse 127.0.0.1 et la clé de communication (shared key).



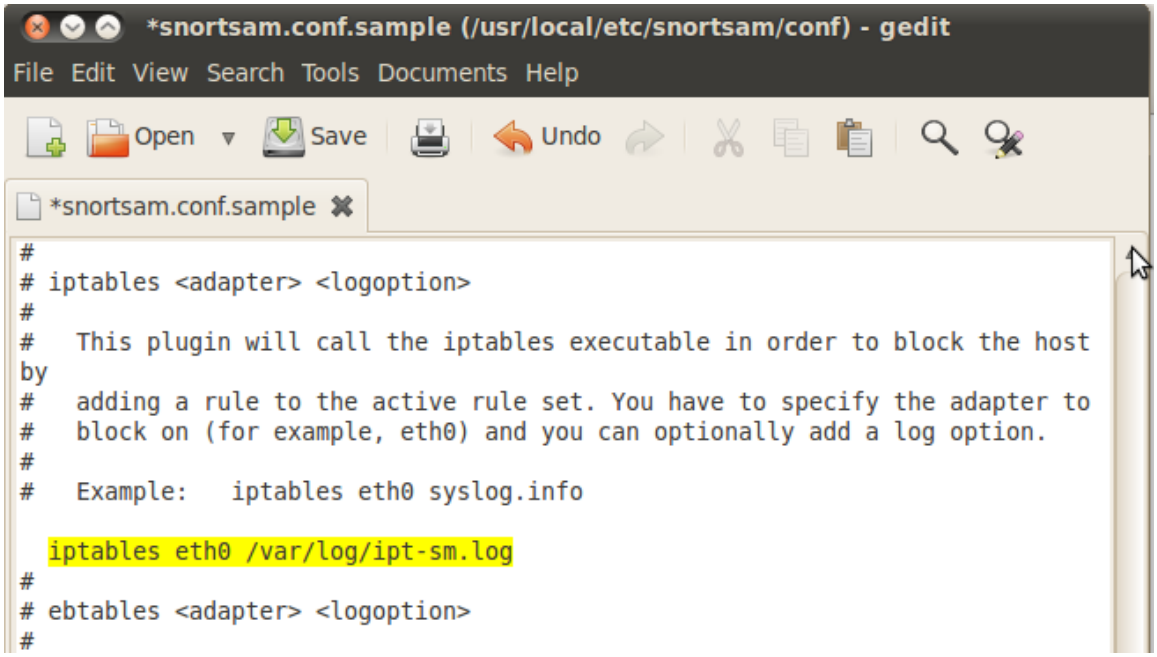
On indique le répertoire du fichier log



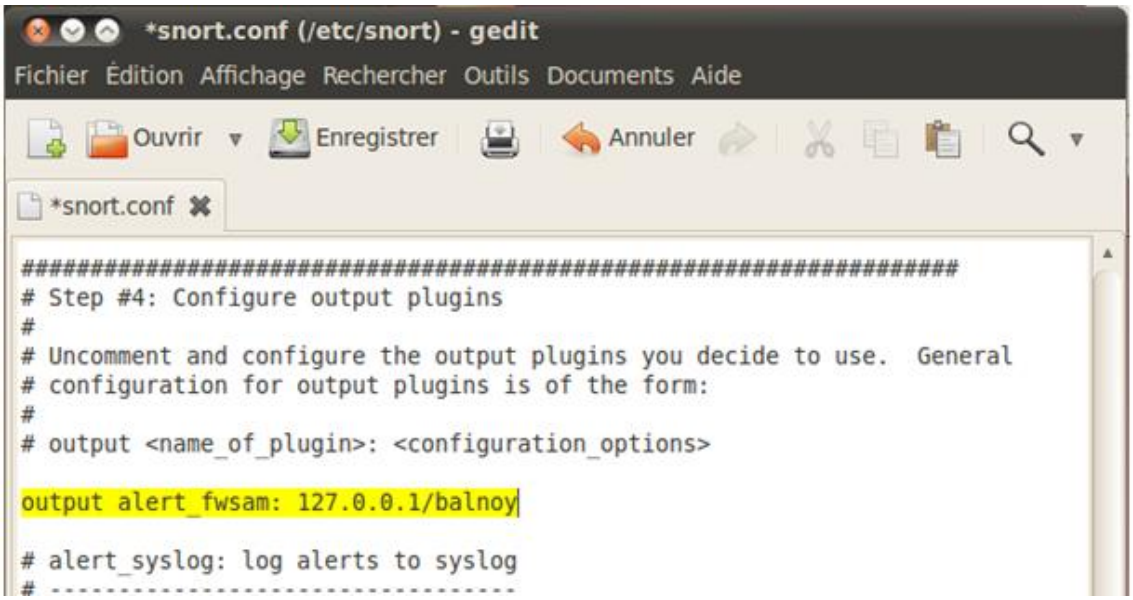
Selon les détails de log afficher, on a choisit le niveau 3 (level 3)



On indique le firewall travaillant avec snortsam ainsi que l'inerface qui les relie :
Dans notre cas : le firewall est iptable via l'interface eth0



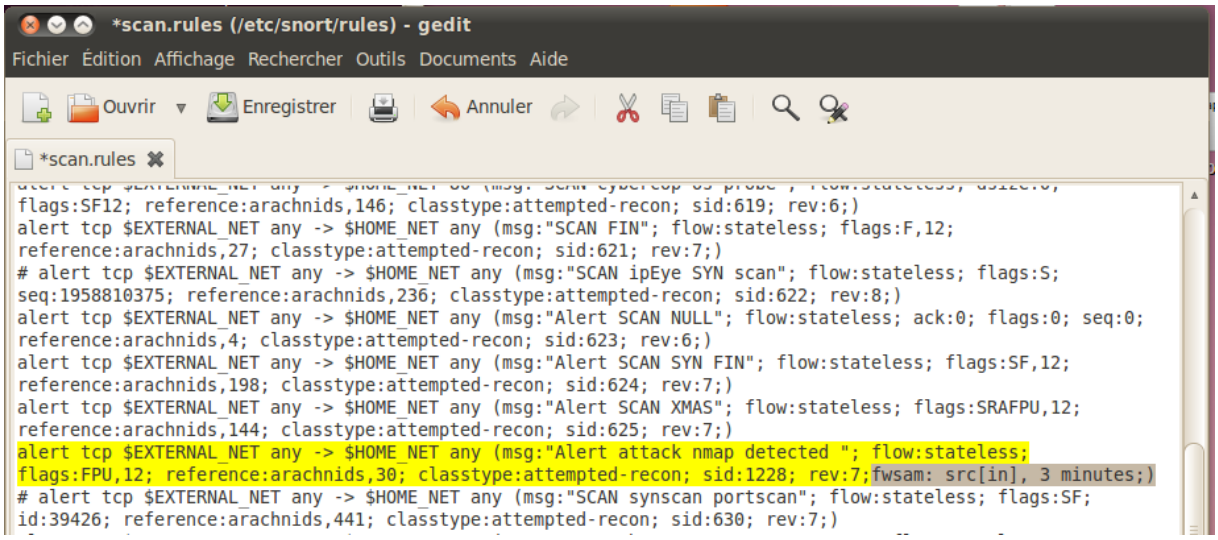
Enfin, pour pouvoir communiquer avec l'agent SnortSam, on specifie dans snort.conf le module de sortie suivant :



IV.7.4 configuration des règles

Pour que notre NIPS (Snort/SnortSam) puisse bloquer les attaques, on édite les règles a la manière qui nous convient.
Dans notre cas, on a choisi de bloquer le scan nmap et l'attaque DoS (deni e service)

On injecte dans scan.rules : `fwsam : src[in], 3 minutes ;`
(blocage de l'attaque entrante pendant 3 minutes)



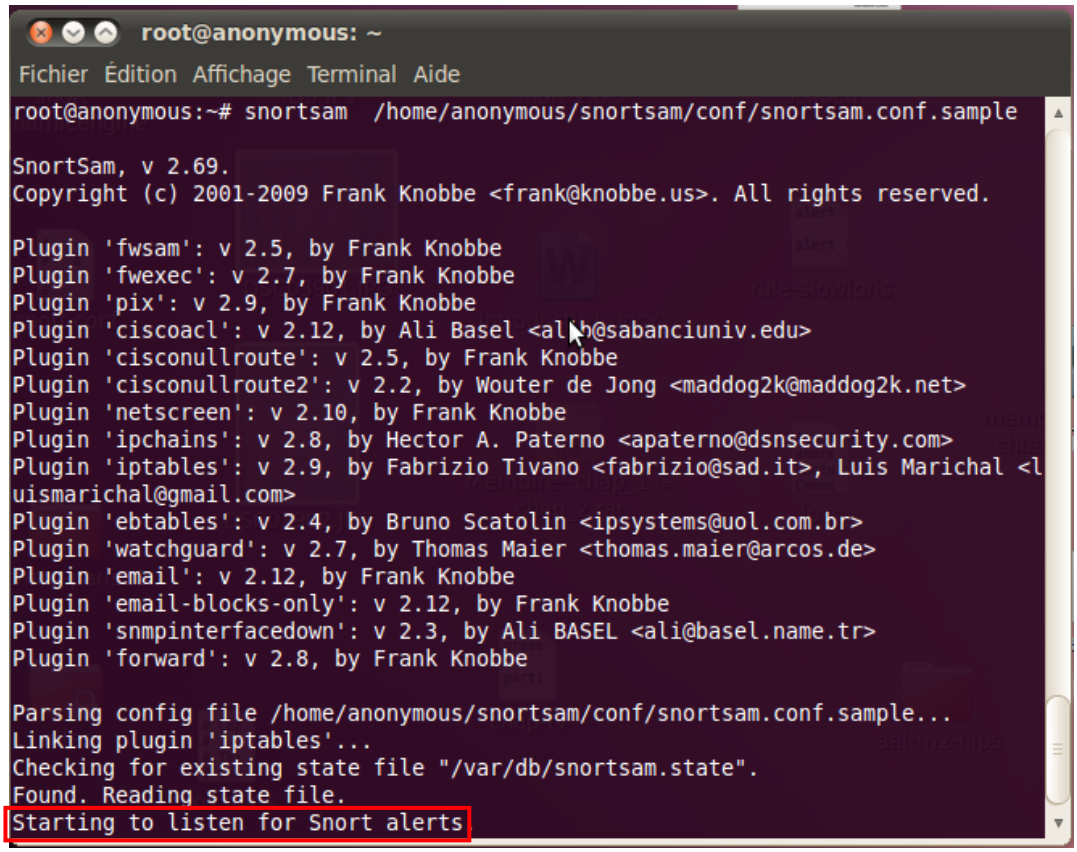
Dans dos.rules on ajoute : `fwsam : src[in], 1 minutes ;`

IV.8 Lancement d'attaques

On interconnecte deux PC via Ethernet (802.3), l'un jouera le rôle de lanceur d'attaques et l'autre essayera de détecter et stopper ces attaques.

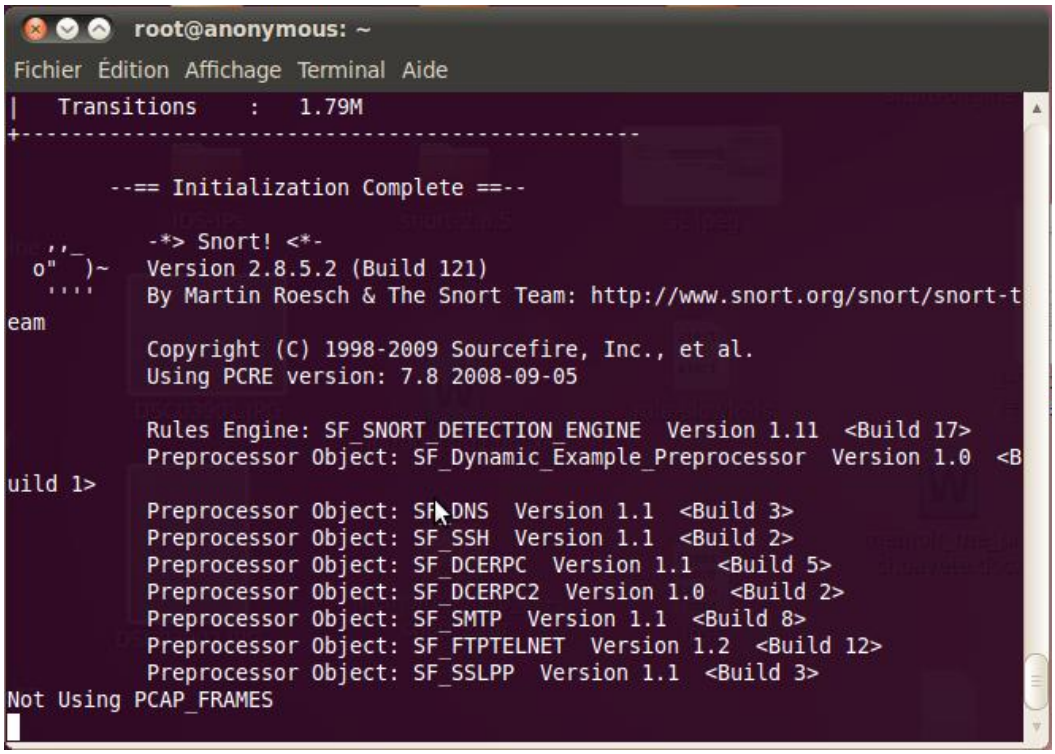
Dans la machine 'victime' on lance dabord snortsam :

```
#snortsam /home/anonymous/snortsam/conf/snortsam.conf.sample
```



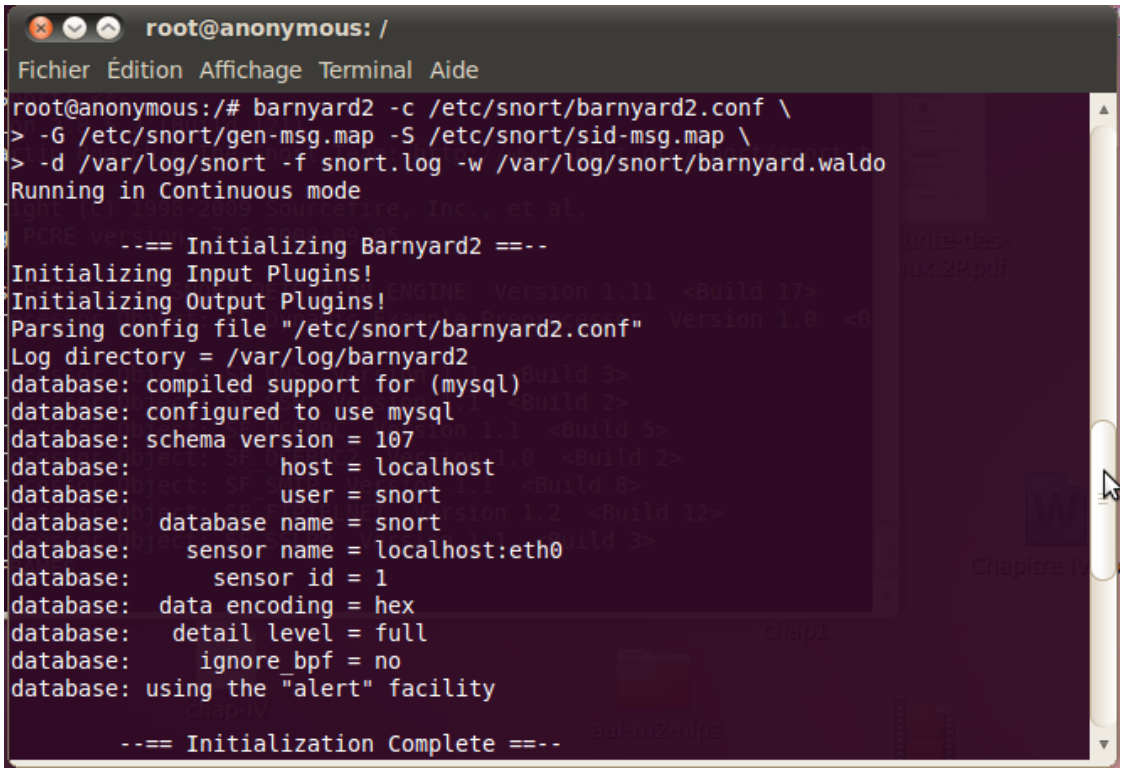
Puis on lance Snort :

```
#snort  -c /etc/snort/snort.conf -i eth0
```



Et enfin Barnyard :

```
#barnyard2 -c /etc/snort/barnyard2. Conf  -G
/etc/snort/gen-msg.map -S /etc/snort/sid-msg.map  -d
/var/log/snort -f snort.log -w
/var/log/snort/barnyard.waldo
```

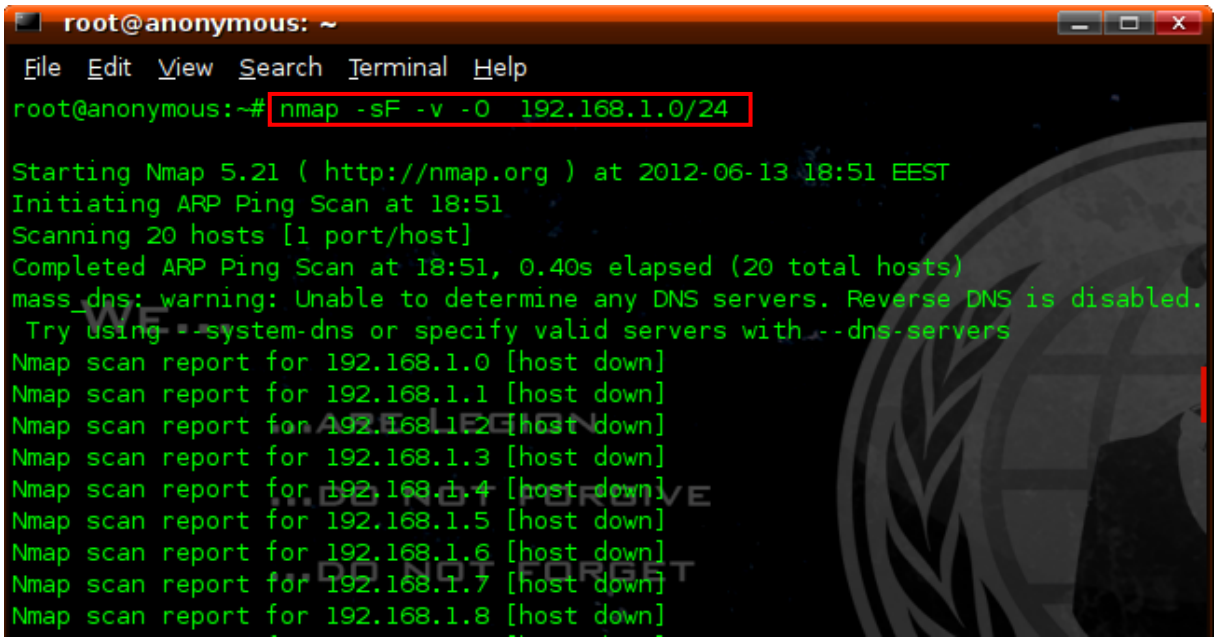


La synchronisation de notre NIPS est terminé.

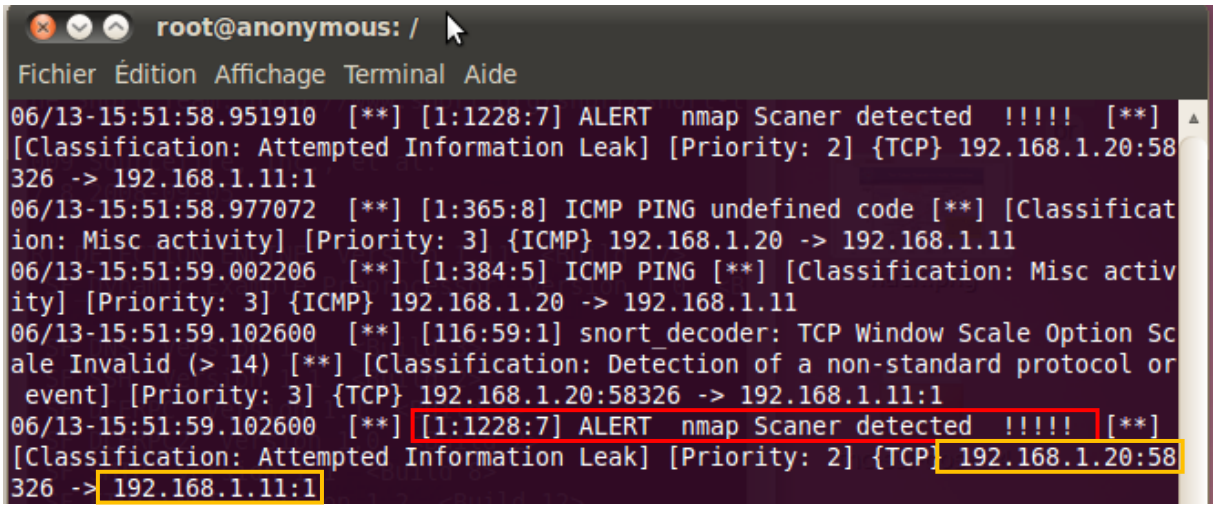
IV.8.2 Scan du réseau 'nmap'

A partir du shell de Anonymous-OS, on lance nmap:

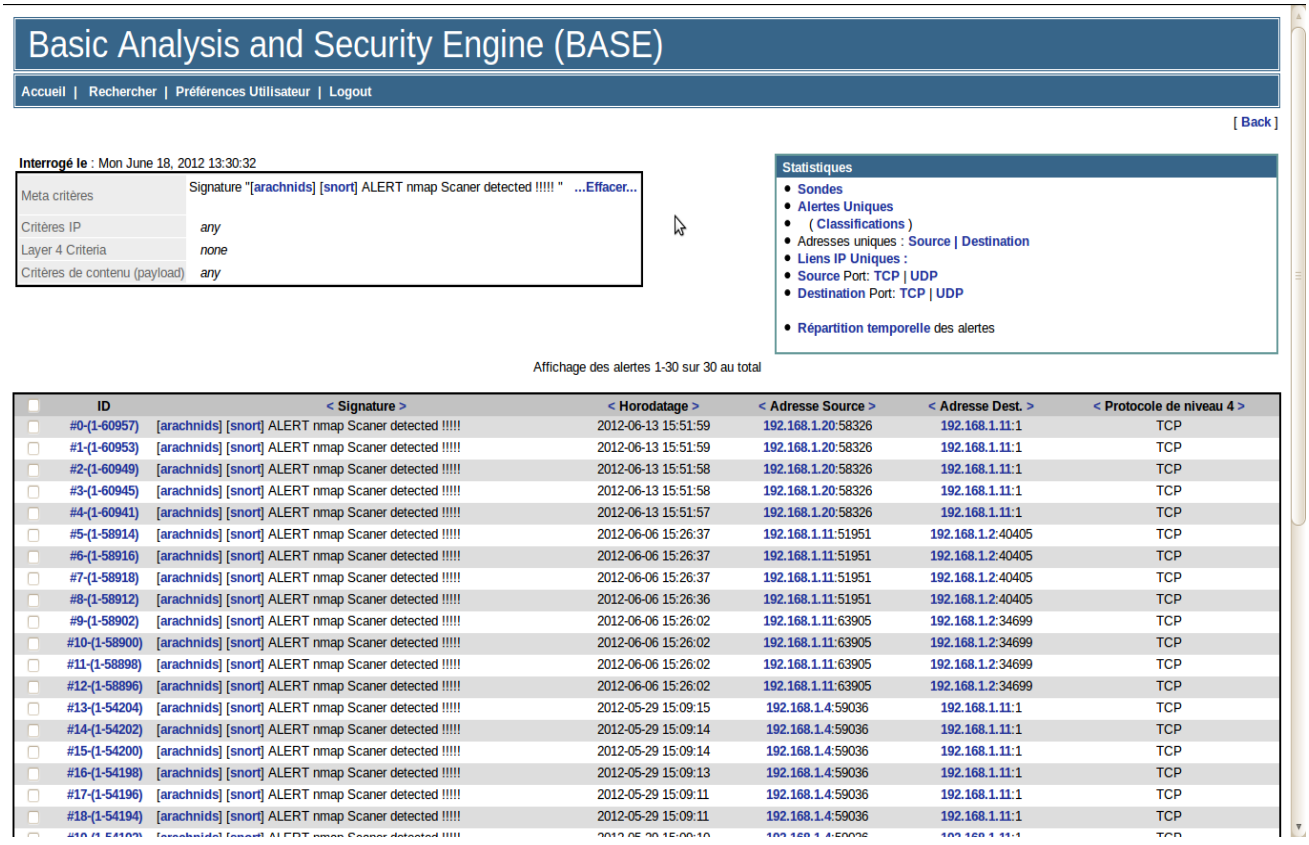
```
nmap -sF -v -O 192.168.1.0/24
```

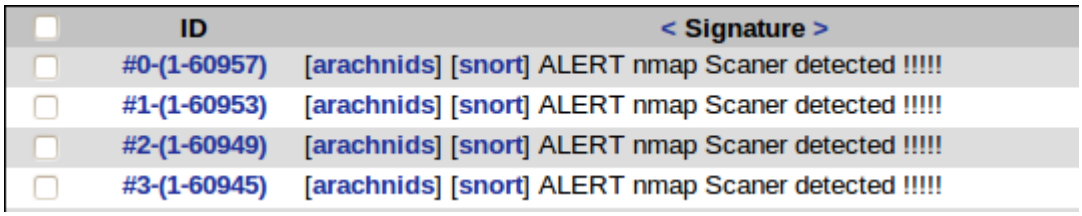


Dans la machine ‘victim’, on remarquera rapidement la détection de ce scan :
Depuis Barnyard :

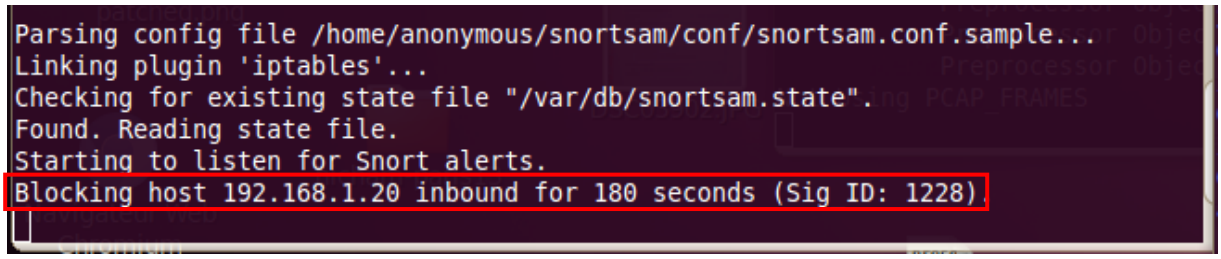


Dans la console BASE, les alertes s’affichent comme suit :





Et on remarque aussi que snortsam a reussi à stopper le scan a partir de la source(192.168.1.20), du coup, aucune information ne sera trasmise a l'attaquant (adresse IP, ports overts, etc)



IV.9 Conclusion

Snort est un outil puissant et efficace qui sécurise notre réseau. Sa puissance devient encore plus importante en l'étendant en NIPS grâce à SnortSam.

Dans cette partie, nous avons illustré l'installation et le mécanisme de fonctionnement de Snort en détails. Nous avons vu à la fin, comment Snort a pu stopper une attaque DoS avec succès !

Snort est un outil Open source, donc gratuit est accessible a n'importe quel utilisateur, il est surtout conseillé aux petites entreprises qui n'ont pas les moyens ou les besoins de procurer des solutions hardwares qui sont très chères, et qui offre le moindre service ou mise a jour avec des tarifs exorbitants.

Conclusion générale

Conclusion générale

Les réseaux d'entreprises ont évolué au fil du temps à une vitesse vertigineuse. L'interconnexion de ces réseaux à Internet les a directement exposés aux menaces informatiques. C'est ce qu'on constate dans la figure[1] affirmée par le bureau d'investigation fédéral américain (F.B.I), ce qui a obligé les entreprises a prendre des mesures sécuritaires afin de remédier a cela.

Les mesures sont exprimées sous formes de politiques de sécurité afin de prendre les mesures nécessaire selon la conception du réseau et les données a sécurisé.

L'étude que nous avons menés nous a conduit a découvrir les systèmes de détection et de préventions des intrusions. Ces systèmes sont indispensables aux entreprises pour assurer leur sécurité. Ils s'ajoutent au réseau standard de l'entreprise pour optimiser la sécurité.

Il faut noter que ces outils sont développés pour compléter les outils déjà existants comme le firewall, et non pas de les remplacer.

Dans notre cas, nous avons pris le ca de SNORT qui est un IDS open source sous LINUX extensible en IPS grâce à SnortSam.

Snort ne garantie pas une sécurité a cent pour cent, néanmoins, il nous procure une certaine immunité vis a vis notre réseau.

En perspective, il faut noter que SNORT reste difficile à mettre en œuvre et à utiliser et ne facilite pas la tâche à l'administrateur.

Autre inconvénient, SNORT est très gourmant en termes de ressources (CPU, RAM, …). Aussi, SNORT doit mettre à jour ces règles régulièrement afin de pouvoir s'adapter aux nouvelles menaces.

Bibliographie

Bibliographie

[1] LESCOP Yves 2002
ylescop.free.fr/mrim/cours/securite.pdf

[2] Les systèmes de détectiond'intrusions
http://dbprog.developpez.com/securite/ids/IDS.pdf

[3] Cyber-attaque Comprendre le phénomène
http://www.risques.gouv.fr/risques/autre-risque/Cyber-risques/

[4] Les systèmes de détection d'intrusion réseau
http://www.litis.univ-lehavre.fr/~duvallet/enseignements/Cours/M2MATIS/SIRES-IDS-4p.pdf

[5] Mémoire de fin d'études du premier cycle licence téléinformatique
http://www.nthierno.unblog.fr/2010/05/02/memoire-de-fin-detudes-du-premier-cycle-licence-teleinformatique-3/

[6] Les systèmes de détection d'intrusion réseau
http://www.litis.univ-lehavre.fr/~duvallet/enseignements

[7] *NT Réseaux IDS & IPS*
http://www.monge.univ-mlv.fr/~duris/NTREZO/.../Baudoin-Karle-IDS-IPS.pdf

[8] La sécurité de la téléphonie sur IP en entreprise
*http://*www.cedric-baillet.fr/IMG/pdf

[9] Les systèmes de détection d'intrusions *réseaux*
http://litis.univ-lehavre.fr/~duvallet/enseignements/Cours/M2MATIS/SIRES-IDS-4p.pdf

[10] rapport final du ter - site perso guillaume
http://lehmann.free.fr/RapportMain.pdf

[11] Mise en place d'une sonde : Snort
http://www.repo.zenk-security.com/

[12] *Snort* - SecuriNets
http://www.securinets.com/sites/default/files/tuto_pdf/Tuto_**Snort**.pdf

[13] Système de *détection d'intrusion : SNORT*
http://www.trustonme.net/didactels/187.html

[14] Sécurité dans les réseaux Ad-hoc avec SNORT
projets-gmi.iup.univ-avignon.fr/projets/Rapport_Sem2_p06.pdf

[15] L'écriture des règles SNORT
www.groar.org/trad/snort/snort-faq/writing_snort_rules.html

[16] *Snort*:Preambule - Aldeid
http://www.aldeid.com/wiki/Snort:Preambule

[17] Intrusion système information : détection faille sécurité
http://www.nbs-system.com

[18] SnortSam Readme
http://doc.emergingthreats.net/bin/view/Main/SnortSamREADMErules

[19] Guide de *configuration Netfilter-iptables*
http://www.ssi.gouv.fr/IMG/pdf/Guide_de_configuration_Netfilter_v1.pdf

[20] Sécurité des Réseaux, S. Breton

Cours Université Paris XIII – Master pro – 2011-2012