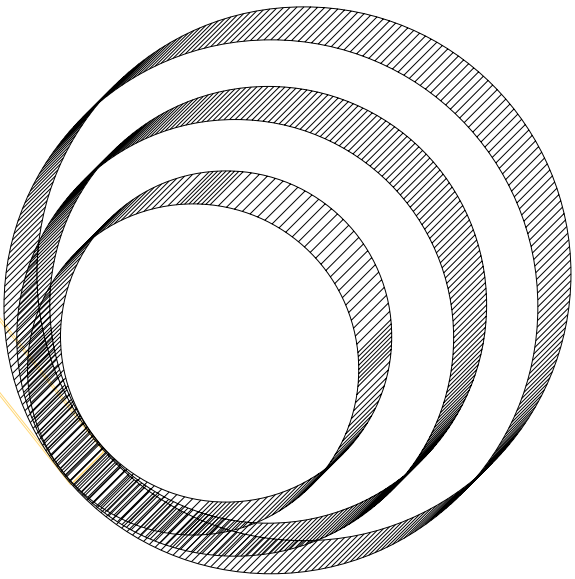
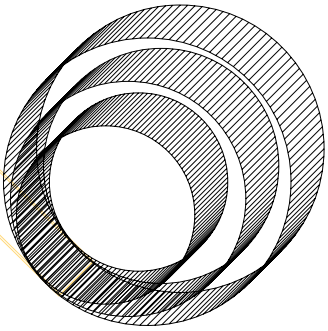




UNIVERSITÉ PARIS DESCARTES

TER Detection d'anomalies sur le réseau

Elaboré par: Jabou Chaouki
Schillings Michaël
Hantach Anis

Encadré par : Mr Osman Salem

2008/2009

Remerciement

Nous avons l'honneur d'exprimer nos profondes gratitudes et nos sentiments les plus sincères de respect et de reconnaissance à tous ceux qui nous ont aidés de près ou de loin à l'achèvement de ce projet TER et à l'élaboration du présent rapport.

Nous remercions en particulier :

Mr. Osman Salem, notre encadreur, pour ses suggestions et ses précieux conseils qui nous ont guidés tout au long de notre projet TER.

Nos reconnaissances vont également à nos amis pour leurs aides qui nous ont été d'un grand apport tout au long de ce travail.

Table des matières

Table des Figures

I Introduction :

Internet relie des centaines de millions d'ordinateurs à travers le monde en fonctionnant sur plusieurs plateformes matérielles et logicielles. Il sert d'innombrables besoins personnels et professionnels pour les personnes et les sociétés. Cependant, cette interconnexion des ordinateurs permet également aux utilisateurs malveillants d'utiliser ces ressources à des fins abusives et de lancer par exemple des attaques dites de "déni de service" (DoS) à l'encontre de serveurs web.

Dans une attaque par déni de service, un utilisateur exploite la connectivité d'Internet pour paralyser les services offerts par le site ciblé. Cela se traduit souvent pas une inondation (flood) de la victime avec de très nombreuses requêtes. Une attaque DoS peut être soit effectuée par une source unique, soit par plusieurs sources, où de multiples programmes se coordonnent pour lancer leurs attaques simultanément, on parle alors de DDoS (Distributed Denial of Service). Plusieurs outils sophistiqués qui automatisent les procédures d'attaques sont facilement disponibles sur Internet, et des instructions détaillées permettent même à un amateur de les utiliser efficacement. Chaque année les attaques de déni de service causent d'importants dommages financiers, de sorte qu'il est essentiel de mettre au point des techniques de détection et de répondre aux attaques rapidement. Le développement de ces techniques exige une bonne connaissance des attaques.

Aujourd'hui, les outils de suivi permettent de détecter une attaque et d'identifier les propriétés de base telles que le taux de trafic et les types de paquets. Toutefois, caractériser les attaques comme étant simples ou multi-source et identifier le nombre d'attaquants est difficile.

Tout au long de ce document, nous développons le travail réalisé dans le cadre de notre projet.

Dans un premier temps, notre tâche a été de nous informer sur les types d'attaques utilisées sur Internet et connaître leur mode opératoire. Nous avons donc étudié plusieurs outils d'attaques parmi les plus célèbres, notamment ceux utilisant des attaques de type déni de service. La compréhension de leur fonctionnement était une étape nécessaire et en toute logique nous avons poursuivi notre travail sur l'étude de méthodes de protection.
La mise au point d'un système de détection d'intrusions étant le principal sujet de ce travail, nous avons consacré une grande partie de notre temps à la recherche d'algorithmes et de méthodes de détection en essayant de trouver les plus efficaces afin d'en tirer des idées pour l'amélioration de CUSUM.

La deuxième partie de ce rapport présente donc les systèmes de détection d'intrusions, différentes approches, différents algorithmes pour les mettre en œuvre en concluant sur leurs avantages et inconvénients par rapport à d'autres outils de protection.

Dans la troisième partie nous présentons en détails l'algorithme CUSUM, son fonctionnement concret et plus spécialement son implémentation dans le cadre de notre projet en expliquant les étapes de développement et le résultat que l'on obtient.
Enfin, pour permettre une meilleure administration du programme CUSUM et pour offrir de meilleures options d'exploitation des résultats, nous avons développé une interface PHP en interaction avec une base de données. La dernière partie de ce rapport explique le fonctionnement de cette interface et les choix qui ont été faits concernant les fonctionnalités de la base de données et de l'interface.

II Attaques et techniques de protection:

1-Introduction :

De nos jours la sécurité du réseau informatique est devenue indispensable face aux attaques qui se multiplient rapidement. Pour contrarier ces attaques, les systèmes de sécurité visent à prévenir de ces dernières et à corriger les vulnérabilités exploitées. Il est alors nécessaire d'établir l'identification des menaces potentielles et de connaître les différents procédés des attaquants afin de sécuriser le réseau. La compréhension des techniques d'intrusion permet de définir des méthodes efficaces de surveillance et de détection des attaques.

Dans la première partie de ce chapitre nous exposons les différentes étapes, types et outils d'attaques sur le réseau informatique, ensuite nous présentons les solutions possibles de protection contre ces attaques.

2- Les Attaques

Une attaque est l'exploitation d'une faille d'un système informatique connecté à un réseau.

Pour réussir leur exploit, les attaquants tentent d'appliquer un plan d'attaque bien précis pour aboutir à des objectifs distincts.

2.1 Les différentes étapes d'une attaque :

La plupart des attaques, de la plus simple à la plus complexe fonctionnent suivant le même schéma :

- **Identification de la cible :** cette étape est indispensable à toute attaque organisée, elle permet de récolter un maximum de renseignements sur la cible en utilisant des informations publiques et sans engager d'actions hostiles. On peut citer par exemple l'interrogation des serveurs DNS…

- **Le scanning :** l'objectif est de compléter les informations réunies sur une cible visées. Il est ainsi possible d'obtenir les adresses IP utilisées, les services accessibles de même qu'un grand nombre d'informations de topologie détaillée (OS, versions des services, règles de pare-feu…).

- **L’exploitation :** cette étape permet à partir des informations recueillies d’exploiter les failles identifiées sur les éléments de la cible, que ce soit au niveau protocolaire, des services et applications ou des systèmes d’exploitation présents sur le réseau.

- **La progression :** il est temps pour l’attaquant de réaliser son objectif. Le but ultime étant d’obtenir les droits de l’utilisateur root sur un système afin de pouvoir y faire tout ce qu’il souhaite (inspection de la machine, récupération d’informations, nettoyage des traces...).

2.2 Les différents types d’attaques :

Une personne mal intentionnée dispose d'une panoplie d'attaques pour s’approprier, bloquer ou modifier des ressources. En voici quelques unes :

Le sniffing : grâce à un logiciel appelé "sniffer", il est possible d’intercepter toutes les trames que notre carte réseau reçoit et qui ne nous sont pas destinées. Si quelqu’un se connecte par Telnet par exemple à ce moment-là, son mot de passe transitant en clair sur le net, il sera aisé de le lire. De même, il est facile de savoir à tout moment quelles pages web regardent les personnes connectées au réseau, les sessions ftp en cours, les mails en envoi ou réception. Un inconvénient de cette technique est de se situer sur le même réseau que la machine ciblée.

L’IP spoofing : cette attaque est difficile à mettre en œuvre et nécessite une bonne connaissance du protocole TCP. Elle consiste, le plus souvent, à se faire passer pour une autre machine en falsifiant son adresse IP de manière à accéder à un serveur ayant une "relation de confiance" avec la machine "spoofée".

Les programmes cachés ou virus : il existe une grande variété de virus. On ne classe cependant pas les virus d’après leurs dégâts mais selon leur mode de propagation et de multiplication. On recense donc les vers (capables de se propager dans le réseau), les troyens (créant des failles dans un système), Les bombes logiques (se lançant suite à un événement du système (appel d’une primitive, date spéciale)).

Les scanners : un scanner est un programme qui permet de savoir quels ports sont ouverts sur une machine donnée. Les Hackers utilisent les scanners pour savoir comment ils vont procéder pour attaquer une machine. Leur utilisation n'est heureusement pas seulement malsaine, car les scanners peuvent aussi permettre de prévenir une attaque. Le plus connu des scanners réseau est « Nmap ».

SYN Flooding : une connexion TCP s'établie en trois phases. Le SYN Flooding exploite ce mécanisme d'établissement en trois phases. Les trois étapes sont l'envoi d'un SYN, la réception d'un SYN-ACK et l'envoi d'un ACK. Le principe est de laisser sur la machine cible un nombre important de connexions TCP en attentes. Pour cela, l’attaquant envoie un très grand nombre

de demandes de connexion, la machine cible renvoie les SYN-ACK en réponse au SYN reçus. L'attaquant ne répondra jamais avec un ACK, et donc pour chaque SYN reçu la cible aura une connexion TCP en attente.
Etant donné que ces connexions semi-ouvertes consomment des ressources mémoires au bout d'un certain temps, la machine est saturée et ne peut plus accepter de connexion. Ce type de déni de service n'affecte que la machine cible

DDoS : "Distributed Denial of Service" ou déni de service distribué est un type d'attaque très évolué visant à faire planter ou à rendre muette une machine en la submergeant de trafic inutile. Plusieurs machines à la fois sont à l'origine de cette attaque (c'est une attaque distribuée) qui vise à anéantir des serveurs, des sous réseaux, etc. D'autre part, elle reste très difficile à contrer ou à éviter. C'est pour cela que cette attaque représente une menace que beaucoup craignent.

- **Mode opératoire :**

Les DDoS sont devenus à la portée de tout un chacun depuis quelques ans. En effet dans les premiers temps, cette attaque restait assez compliquée et nécessitait de bonnes connaissances de la part des attaquants; mais ceux-ci ont alors développé des outils pour organiser et mettre en place l'attaque. Ainsi le processus de recherche des hôtes secondaires (ou zombies) a été automatisé. On cherche en général des failles courantes (buffer overflows sur wu-ftpd, les RPCs...) sur un grand nombre de machines sur Internet et l'attaquant finit par se rendre maître (accès administrateur) de centaines voir de milliers de machines non protégées. Il installe ensuite les clients pour l'attaque secondaire et essaye également d'effacer ses traces (corruption des fichiers logs, installation de rootkits). Une fois le réseau en place, il n'y a plus qu'à donner l'ordre pour inonder la victime finale de paquets inutiles. Il est intéressant de noter que les victimes dans ce type d'attaques ne sont pas que celles qui subissent le déni de service; tous les hôtes secondaires sont également des machines compromises jusqu'au plus haut niveau (accès root), tout comme l'hôte maître. La menace provient du fait que les outils automatisant le processus ont été très largement diffusés sur Internet. Il n'y a plus besoin d'avoir des connaissances pointues pour la mettre en place, il suffit de "cliquer" sur le bouton.

3- Les outils

Dans cette partie nous présentons quelques outils d'attaque réseaux. Nous avons choisi les plus utilisés servant à effectuer une attaque de déni de service. Dans le cas d'une attaque distribuée, un réseau typique se compose d'un point central et de nombreux hôtes distants, encore appelés démons. Pendant le déroulement de l'attaque, le hacker se connecte au point central qui envoie alors un ordre à tous les hôtes distants (via UDP, TCP ou ICMP). Ensuite, les hôtes distants vont attaquer la cible finale suivant la technique choisie par l'attaquant.

3.1 TFN (Tribal Flood Network)

C'est le premier outil d'attaque de type Déni de Service qui a obtenue une large visibilité. Il utilise une architecture à deux couches. Un client qui contrôle des agents/démons. Ces agents réalisent l'attaque distribuée sur la victime/cible et avec le type d'attaque désiré par le client. Les agents TFN fonctionnent comme des services réseaux cachés sur les machines piratées, capables de recevoir les commandes du client noyées parmi le flux courant des communications du réseau. Les adresses du client et des agents sont falsifiées dans tous les communications et les attaques.

Architecture du TFN

- Protocole de la communication entre Client/Agent : ICMP
- L'attaque sur les protocoles : IP / TCP / UDP / ICMP

Le TFN client est exécuté à travers la commande ligne pour envoyer des commandes aux TFN Agents.

3.2 TFN2K

C'est la version évoluée de TFN, au début sur une architecture à deux couches (trois couches désormais), mais avec l'ajout de chiffrement BlowFish de ces communications entre le client et ses agents, qui le rend plus dur à détecter. L'architecture du TFN2K ressemble à celle du TFN.

- Les communications entre maitre et agent se font en TCP, UDP, ICMP ou un protocole aléatoire entre les trois.
- Les méthodes d'attaque sont TCP, UDP, ICMP paquet flood.
- L'en-tête du paquet entre maître et agent est aléatoire sauf pour ICMP
- Le client renvoie chaque commande 20 fois pour s'assurer que le démon la recevra au moins une fois.
- Toutes les commandes sont chiffrées en utilisant l'algorithme CAST-256. La clé est définie au moment de compilation et utilisée comme mot de passe pour lancer tfn2k client.

3.3 Trin00

Trin00 s'appuie sur une architecture à trois couches avec un client, qui envoie des commandes à des serveurs maîtres qui se chargent chacun d'un sous réseau d'agents. Cette couche intermédiaire rend plus difficile à identifier l'origine de l'attaque. Cependant, Trin00

réussi moins bien à dissimuler ses communications au sein du trafic réseau. Trin00 n'utilise qu'une seule forme d'attaque DoS (UDP), contrairement à TFN.

4- Protection du système

La variété et la disponibilité des outils d'attaques augmentent le risque des intrusions. Par conséquent les administrateurs s'appuient sur diverses solutions comme les pare-feux, les systèmes de détection d'intrusions... dans le but de maintenir la protection du réseau informatique. Nous détaillons dans la suite chacune de ces méthodes et nous soulignons leurs limites.

4.1 Pare feux

Un pare-feu (Firewall) est un système physique ou logique qui inspecte les paquets entrants et sortants du réseau afin d'autoriser ou d'interdire leur passage en se basant sur un ensemble de règles appelées ACL. Il existe principalement trois types de pare-feux :

- Pare-feu avec filtrage des paquets : ce pare feu filtre les paquets en utilisant des règles statiques qui testent les champs des protocoles jusqu'au niveau transport.

- Pare-feu à filtrage des paquets avec mémoire d'états : ce modèle conserve les informations des services utilisés et des connexions ouvertes dans une table d'états. Il détecte alors les situations anormales suite à des violations des standards de protocole,

- Pare-feu proxy : ce pare feu joue le rôle d'une passerelle applicative. En analysant les données jusqu'au niveau applicatif, il est capable de valider les requêtes et les réponses lors de l'exécution des services réseaux.

Malgré leurs grands intérêts, les pare-feux présentent quelques lacunes. En effet, un attaquant peut exploiter les ports laissés ouverts pour pénétrer le réseau local. Ce type d'accès est possible même à travers des pare-feux proxy. Il suffit d'utiliser un protocole autorisé tel que HTTP pour transporter d'autres types de données refusées.
Ainsi l'opération supplémentaire d'encapsulation/décapsulation des données permet à l'attaquant de contourner le pare feu. Les scripts constituent aussi des sources d'intrusion que les pare feux échouent à détecter.

4.2 Systèmes de détection d'intrusions

Un système de détection d'intrusions (IDS) tente d'identifier les menaces dirigées contre le réseau de l'entreprise. Il s'appuie sur plusieurs sources d'informations comme les fichiers d'audit, les journaux de sécurité et le trafic réseau.

Il s'est avéré que l'outil mentionné précédemment ne peut pas prévenir toutes les attaques et par suite assurer seul une sécurité idéale du réseau. Étant donnée l'impossibilité de stopper toutes les attaques, les systèmes de détection d'intrusions constituent une bonne solution pour détecter celles qui passent inaperçues. Placés après les pare-feux, les IDS constituent la dernière barrière de sécurité. Ils analysent le trafic qui passe à travers les pare-feux et supervisent les activités des utilisateurs sur le réseau local. Par ailleurs, placés avant les pare-feux, les IDS découvrent les attaques à l'entrée du réseau. Les IDS s'appuient généralement sur deux sources d'information : les paquets transitant sur le réseau et les informations collectées sur les machines.

On parle alors de deux types de systèmes de détection d'intrusions : les IDS basés réseau et les IDS basés hôte. Ces deux catégories d'IDS emploient généralement deux principes de détection l'approche comportementale et l'approche basée sur la connaissance.

5- Conclusion

Dans ce chapitre nous avons vu comment un attaquant peut compromettre un système informatique en suivant une stratégie bien définie et en utilisant des outils adaptés. Pour remédier à ces problèmes, des solutions de sécurité efficaces sont mises en œuvre par les administrateurs. Dans une optique d'optimisation de cette sécurisation, les systèmes de détection d'intrusions présentent un bon moyen de garantir cette sécurité des réseaux.
C'est pourquoi nous entamerons dans le chapitre suivant l'étude des différents systèmes de détection d'intrusions, leur fonctionnement et leurs limites.

III Système de détection d'intrusions

1- Introduction

Un IDS (*Intrusion Detection System*) est un outil qui permet d'écouter le traffic réseau de façon furtive dans le but de détecter des activités anormales qui pourraient être assimilées à des intrusions et permettant ainsi d'avoir une action de prévention. Malgré le grand intérêt des IDSs, l'évolution des techniques d'attaques et d'autre part la complexité des données limitent l'efficacité de prévention de ces derniers. Etant dans l'incapacité d'arrêter toutes les attaques, la détection d'intrusions tente de percevoir celles que les autres systèmes de sécurité n'arrivent pas à déceler.

À travers ce chapitre nous commençons par classifier les systèmes de détection d'intrusions selon deux approches. Par la suite nous présentons quelques IDSs parmi les plus connus ainsi que quelques algorithmes.

2- Classification des systèmes de détection d'intrusions :

Plusieurs critères permettent de classer les systèmes de détection d'intrusions, la méthode d'analyse étant le principal. Deux méthodes dérivant de cette dernière existent aujourd'hui : l'approche comportementale et l'approche par scénarios.
On peut citer aussi d'autres critères de classification des IDSs : la fréquence d'utilisation, les sources de données à analyser, le comportement de l'IDS après intrusion.

2.1 Approche comportementale

L'approche comportementale est fondée sur une description statistique des sujets. L'objectif est de détecter les actions anormales effectuées par ces sujets (par exemple, des heures de connexion anormales, un nombre anormal de fichiers supprimés ou un nombre anormal de mots de passe incorrects fournis au cours d'une connexion).
Le comportement normal des sujets est appris en observant le système pendant une période donnée appelée phase d'apprentissage (par exemple, un mois). Le comportement normal, appelé comportement sur le long terme, est enregistré dans la base de données et comparé avec le comportement présent des sujets, appelé comportement à court terme. Une alerte est générée si une déviation entre ces comportements est observée. Dans cette approche, le comportement sur le long terme est, en général, mis à jour périodiquement pour prendre en compte les évolutions possibles des comportements des sujets.
Nous considérons traditionnellement que l'avantage principal de l'approche comportementale est de pouvoir être utilisée pour détecter de nouvelles attaques. Autrement dit, en signalant

toute déviation par rapport au profil, il est possible de détecter a priori toute attaque qui viole ce profil, même dans le cas où cette attaque n'était pas connue au moment de la construction du profil.

Cependant, cette approche présente également plusieurs inconvénients. Tout d'abord, le diagnostic fournit par une alerte est souvent flou et nécessite une analyse complémentaire.
Ensuite, cette approche génère souvent de nombreux faux positifs car une déviation du comportement normal ne correspond pas toujours à l'occurrence d'une attaque. Citons à titre d'exemples, en cas de modifications subites de l'environnement de l'entité modélisée, cette entité changera sans doute brutalement de comportement. Des alarmes seront donc déclanchées.
Pour autant, ce n'est peut-être qu'une réaction normale à la modification de l'environnement.

En outre, les données utilisées en apprentissage doivent être exemptes d'attaques, ce qui n'est pas toujours le cas. Enfin, un utilisateur malicieux peut habituer le système (soit pendant la phase d'apprentissage, soit en exploitation si l'apprentissage est continu) à des actions malveillantes, qui ne donneront donc plus lieu à des alertes. Le problème de la détection d'intrusions est couramment approché d'une façon radicalement différente qui est l'approche par scénario.

2.2 Approche par scénario

La détection d'intrusions peut également s'effectuer selon une approche par scénario. Il s'agit de recueillir des scénarios d'attaques pour alimenter une base d'attaques. Le principe commun à toutes les techniques de cette classe consiste à utiliser une base de données, contenant des spécifications de scénario d'attaques (on parle de signatures d'attaque et de base de signatures). Le détecteur d'intrusions compare le comportement observé du système à cette base et remonte une alerte si ce comportement correspond à une signature prédéfinie. Le principal avantage d'une approche par scénario est la précision des diagnostics qu'elle fournit par rapport à ceux avancés par l'approche comportementale. Il est bien entendu que l'inconvénient majeur de cette approche est qu'elle ne peut détecter que des attaques dont elle dispose de leur signature. Or, définir de façon exhaustive la base de signatures est une des principales difficultés à laquelle se heurte cette approche. La génération de faux négatifs est à craindre en présence des nouvelles attaques. En effet, contrairement à un système de détection d'anomalies, ce type de détecteur d'intrusions nécessite une maintenance active : puisque par nature il ne peut détecter que les attaques dont les signatures sont dans sa base de données, cette base doit être régulièrement (sans doute quotidiennement) mise à jour en fonction de la découverte de nouvelles attaques. Aucune nouvelle attaque ne peut par définition être détectée.
D'autre part, il existe de nombreuses attaques difficiles à détecter car elles nécessitent de corréler plusieurs événements. Dans la plupart des produits commerciaux, ces attaques élaborées sont décomposées en plusieurs signatures élémentaires. Cette décomposition peut

générer de nombreux faux positifs si un mécanisme plus global n'est pas développé pour corréler les alertes correspondant à ces différentes signatures élémentaires. Chacune de ces deux approches peut conduire à des faux-positifs (détection d'attaque en absence d'attaque) ou à des faux-négatifs (absence de détection en présence d'attaque).

2.3 Autres critères

Parmi les autres critères de classification existants, nous pouvons citer entre autres :

– les sources de données à analyser :
Les sources possibles de données à analyser sont une caractéristique essentielle des systèmes de détection d'intrusions. Les données proviennent, soit de fichiers générés par le système d'exploitation, soit de fichiers générés par des applications, soit encore d'informations obtenues en écoutant le trafic sur le réseau.

– le comportement de l'IDS après intrusion :
Une autre façon de classer les systèmes de détection d'intrusions, consiste à voir quelle est leur réaction lorsqu'une attaque est détectée. Certains se contentent de déclencher une alarme (réponse passive).

– la fréquence d'utilisation :
Une autre caractéristique des systèmes de détection d'intrusions est leur fréquence d'utilisation : périodique ou continue. Certains systèmes de détection d'intrusions analysent périodiquement les traces d'audit à la recherche d'une éventuelle intrusion ou anomalie passée. Cela peut être suffisant dans des contextes peu sensibles.

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

3- Les différents types d'IDS

Un IDS a pour fonction d'analyser en temps réel ou différé les évènements en provenance des différents systèmes, de détecter et de prévenir en cas d'attaque. Les buts sont nombreux :

– collecter des informations sur les intrusions,
– gestion centralisée des alertes,

– effectuer un premier diagnostic sur la nature de l'attaque permettant une réponse rapide et efficace,

Les systèmes de détection d'intrusion ou IDS peuvent se classer selon trois catégories majeures selon qu'ils s'attachent à surveiller :

– le trafic réseau : on parle d'IDS réseau ou NIDS (Network based IDS)
– l'activité des machines : on parle d'IDS Système ou HIDS (Host based IDS)
– une application particulière sur la machine : on parle d'IDS Application (Application based IDS).

Les NIDS : ces outils analysent le trafic réseau ; ils comportent généralement une sonde qui "écoute" sur le segment de réseau à surveiller et un moteur qui réalise l'analyse du trafic afin de détecter les signatures d'attaques ou les divergences face au modèle de référence. Les IDS réseaux à base de signatures sont confrontés actuellement à un problème majeur qui est l'utilisation grandissante du cryptage. En effet, le cryptage rend l'analyse du contenu des paquets presque impossible. La plupart des NIDS sont aussi dits IDS inline car ils analysent le flux en temps réel. Pour cette raison, la question des performances est très importante.
De tels IDS doivent être de plus en plus performants afin d'analyser les volumes de données de plus en plus importants pouvant transiter sur les réseaux.

Les HIDS : les IDS Systèmes analysent quant à eux le fonctionnement ou l'état des machines sur lesquelles ils sont installés afin de détecter les attaques. Pour cela ils auront pour mission d'analyser les journaux systèmes, de contrôler l'accès aux appels systèmes, de vérifier l'intégrité des systèmes de fichiers ... Ils sont très dépendants du système sur lequel ils sont installés. Il faut donc des outils spécifiques en fonction des systèmes déployés. Ces IDS peuvent s'appuyer sur des fonctionnalités d'audit propres ou non au système d'exploitation, pour en vérifier l'intégrité, et générer des alertes. Il faut cependant noter qu'ils sont incapables de détecter les attaques exploitant les faiblesses de la pile IP du système, typiquement les Dénis de service comme SYN FLOOD ou autre.

4- Quelques systèmes de détection d'intrusions

Actuellement, ils existent plusieurs systèmes de détections, certains sont commercialisés, d'autres sont encore dans les laboratoires de recherche. Dans ce qui suite, nous allons décrire deux systèmes de détection d'intrusion : *SNORT* et *PRELUDE*. Il est à noter que ces outils sont distribués comme logiciels libres.

4.1 Prelude-NIDS

Prelude NIDS est l'un des détecteurs d'intrusions les plus connus. Prelude est disponible et libre sur les plateformes Linux, FreeBSD et Windows. Prelude possède une architecture modulaire et distribuée. Modulaire, car ses composants sont indépendants, et peuvent être facilement mis à jour. Distribuée, car ces composants indépendants interagissent les uns avec les autres. Cela permet d'avoir divers composants installés sur différentes machines et de réduire ainsi la surcharge d'applications. Ces différents composants sont les sondes et les managers. Les sondes peuvent être de deux types : réseau ou local. Une sonde réseau analyse tout le trafic, pour y détecter d'éventuelles signatures d'attaques. La sonde locale assure la surveillance d'une seule machine, il analyse le comportement du système pour y détecter des tentatives d'exploitation de vulnérabilités internes. Les sondes signalent les tentatives d'attaques par des alertes. Ces alertes sont reçues par le manager qui les interprète et les stocke. Nous pouvons également décrire un autre système de détection aussi connu qui est Snort.

4.2 Snort-NIDS

Snort est un NIDS écrit par Martin Roesch, disponible sous licence GNU, son code source est accessible et modifiable. Il permet d'analyser le trafic réseau de type IP. Snort peut être configuré pour fonctionner en quatre modes :

- Le mode sniffer : dans ce mode, Snort analyse les paquets circulant sur le réseau et les affiche d'une façon continue sur l'écran ;
- Le mode « packet logger » : dans ce mode Snort journalise le trafic réseau dans des répertoires sur le disque ;
- Le mode détecteur d'intrusion réseau (NIDS) : dans ce mode, Snort analyse le trafic du réseau, compare ce trafic à des règles déjà définies par l'utilisateur et établi des actions à exécuter ;
- Le mode Prévention des intrusions réseau (IPS), c'est SNORT-inline. Il analyse du trafic en temps réel.

Plusieurs autres IDS existent comme Haystack, MIDAS et IDES. Ces IDS utilisent des techniques différentes dans le but d'améliorer la capacité de détection. Parmi les techniques utilisées, nous pouvons citer le data mining et l'agent mobile.

5- Quelques algorithmes de détection d'intrusions

Plusieurs études se sont intéressées aux problèmes de détection d'anomalies. Dans cette partie nous présentons deux algorithmes. Un algorithme à seuil adaptatif (adaptive threshold) et celui de la somme cumulative (CUmulative SUM, CUSUM) pour la détection de point de rupture. Les deux algorithmes apprennent de manière adaptative le comportement normal. Par conséquent ils n'exigent pas de spécification de signatures d'attaques.

5.1 Algorithme à seuil adaptatif (adaptive threshold)

C'est un algorithme plutôt simple qui détecte les anomalies basé sur le dépassement d'un seuil qui est défini d'une manière adaptative, en utilisant des mesures récentes du trafic, et il se base sur une évaluation du nombre moyen de paquets SYN. L'algorithme considère seulement les dépassements du seuil et pas leur intensité.

Une application directe de cette méthode rapporterait un taux élevé de faux positifs (fausses alertes). Une alarme est signalée après un certain nombre de dépassements consécutifs du seuil.

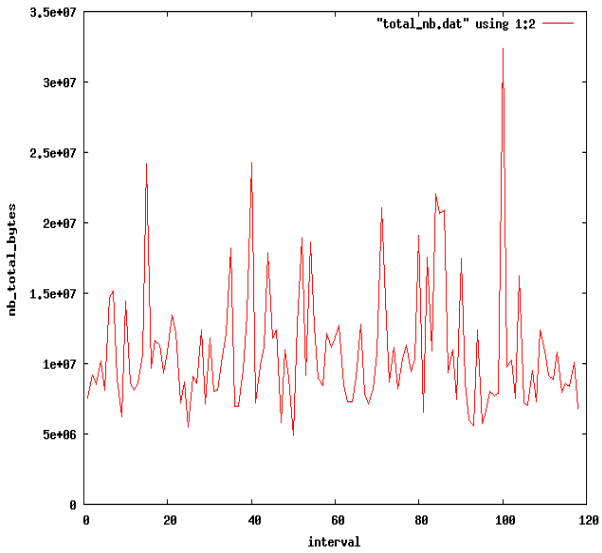
5.2 CUSUM

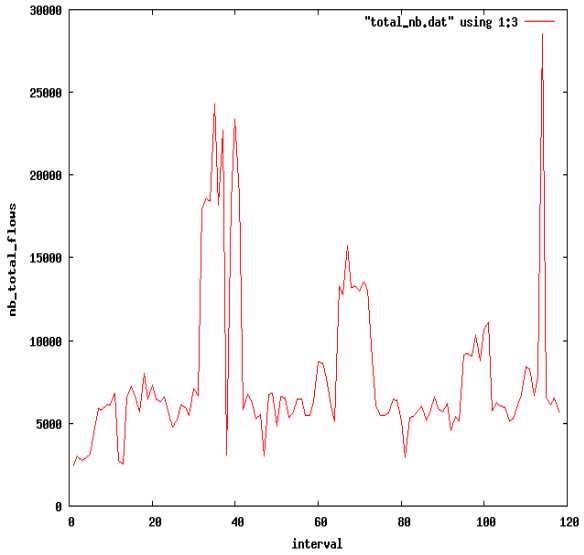
CUSUM est un algorithme employé couramment pour la détection d'anomalies qui se base sur la théorie de détection de point de rupture. En particulier, une alarme est signalée quand un volume accumulé dans un intervalle de temps précis dépasse un seuil bien défini. Ce comportement est similaire à celui de l'algorithme à seuil adaptatif. À la différence de ce dernier qui considère seulement les dépassements du seuil, l'algorithme CUSUM considère le volume excessif envoyé au dessus du volume normal et par conséquent explique l'intensité des violations.

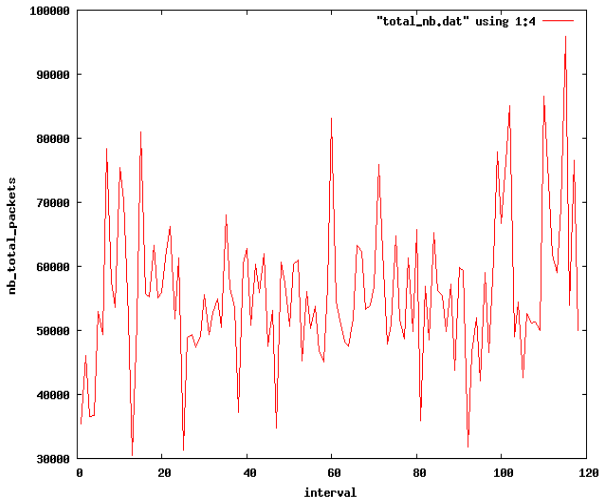
6- Résultats obtenus

Pour une meilleure interprétation des resultats de notre algorithme nous avons tracé les graphes suivants qui montre les variations de trafic selon le nombre d'octets, de paquets, ou de flux en fonction du temps.

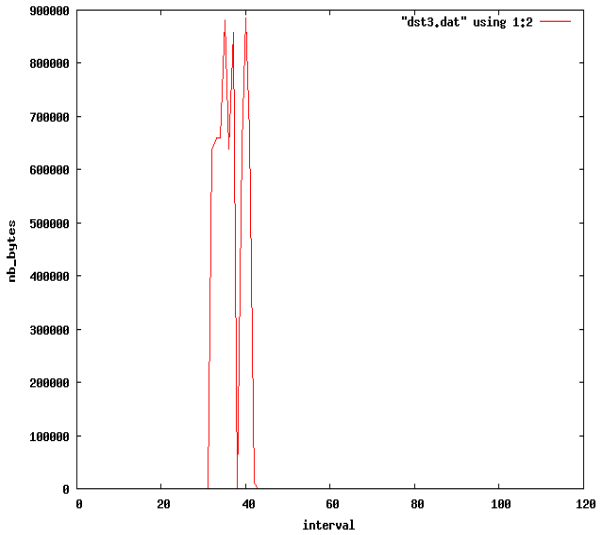
Courbes de variation pour tout le trafic :

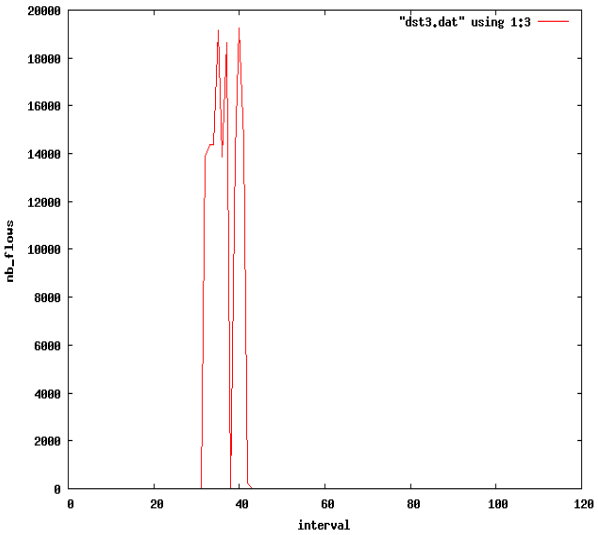


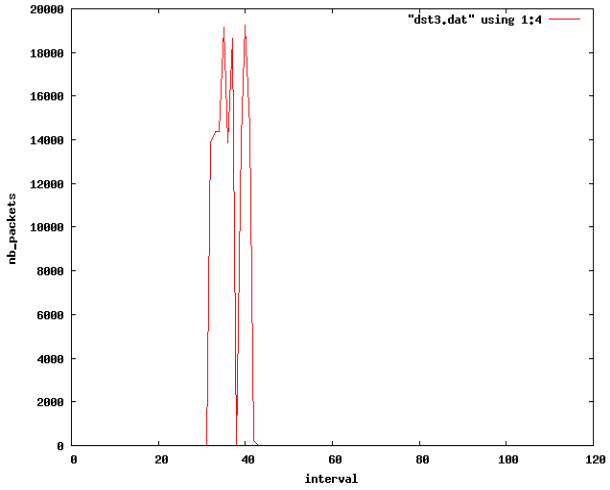




Courbes de variation pour l'adresse destination : 10.0.0.1







⇨ Nous pouvons remarquer sur les trois courbes précédentes une augmentation brusque du trafic (point de rupture) ce qui nous mène à conclure que cette adresse est attaquée.

7- Conclusion

La plupart des IDS sont fiables, ce qui explique qu'ils sont souvent intégrés dans les solutions de sécurité. Les avantages qu'ils présentent face aux autres outils de sécurités les favorisent, mais d'un autre côté cela n'empêche pas que les meilleurs IDS présentent aussi des lacunes et quelques inconvénients. Nous comprenons donc bien qu'ils sont nécessaires mais ne peuvent pas se passer de l'utilisation d'autres outils de sécurité visant à combler leurs défauts.

Par exemple, les systèmes de détection d'intrusions fournissent une bonne sécurité mais qui n'est pas automatisée. Cela signifie que l'utilisation des IDS se fait toujours conjointement à une expertise humaine.

IV Algorithme CUSUM

1- Introduction

Étant une méthode non paramétrique, l'algorithme CUSUM détecte les points de rupture c'est à dire le passage, à l'instant t inconnu, d'un fonctionnement normal (hypothèse 1) à un fonctionnement anormal (hypothèse 2) du système sous surveillance. Ainsi, le dépassement d'un seuil *h*, fixé après une série de tests visant à minimiser le taux de fausses alertes, va engendrer une alarme.

Dans ce chapitre, nous allons expliquer le fonctionnement de cet algorithme et les différents tests effectués en appliquant ce dernier sur les traces qui nous ont été fournies.
La première version de notre algorithme s'est limitée à la surveillance du trafic, à la détection et à l'affichage des alertes sur la console. Dans la deuxième version nous avons rajouté la connexion à la base de données afin de sauvegarder les différents flux et les alertes remontées par l'algorithme que nous allons utiliser dans l'interface PHP pour les classifier, les vérifier et les afficher.

2- Fonctionnement

Comme dit précédemment, l'algorithme CUSUM se base sur le principe de la somme cumulative qui se présente sous cette formule :

$$S_i = [\ S_{i-1} + (X_i - k*\mu)\ ]^+$$

Où S_i est la somme cumulative positive qui est initialisée à 0 lors du lancement de l'algorithme. Ce qui signifie que si elle est négative, elle est immédiatement mise à 0.

X_i est la variable représentant soit le nombre de flux, de paquets ou d'octets.

μ est la moyenne des S_i calculée à l'instant t (i allant de 0 à t).

La moyenne se calcule selon la formule suivante :

$$\mu_i = 0.5*\mu_{i-1} + 0.5*S_{i-1}$$

k est un paramètre à faire varier pour améliorer les performances de la détection (réduire le nombre de fausses alertes...).

Lorsque S_i dépasse un certain seuil *h* ($S_i \geq h$), une alarme se déclenche et réinitialise S_i à 0 sachant que nous pouvons modifier le seuil jusqu'à trouver la valeur optimale avec laquelle on a le minimum de fausses alertes.

2.1 Première version de l'algorithme

Après la programmation et la compilation de l'algorithme nous l'avons appliqué sur les traces d'un trafic qui nous étaient fournies pour détecter les différentes anomalies présentes dans ces dernières.

Nous présentons par la suite quelques captures d'écrans de l'exécution du programme.

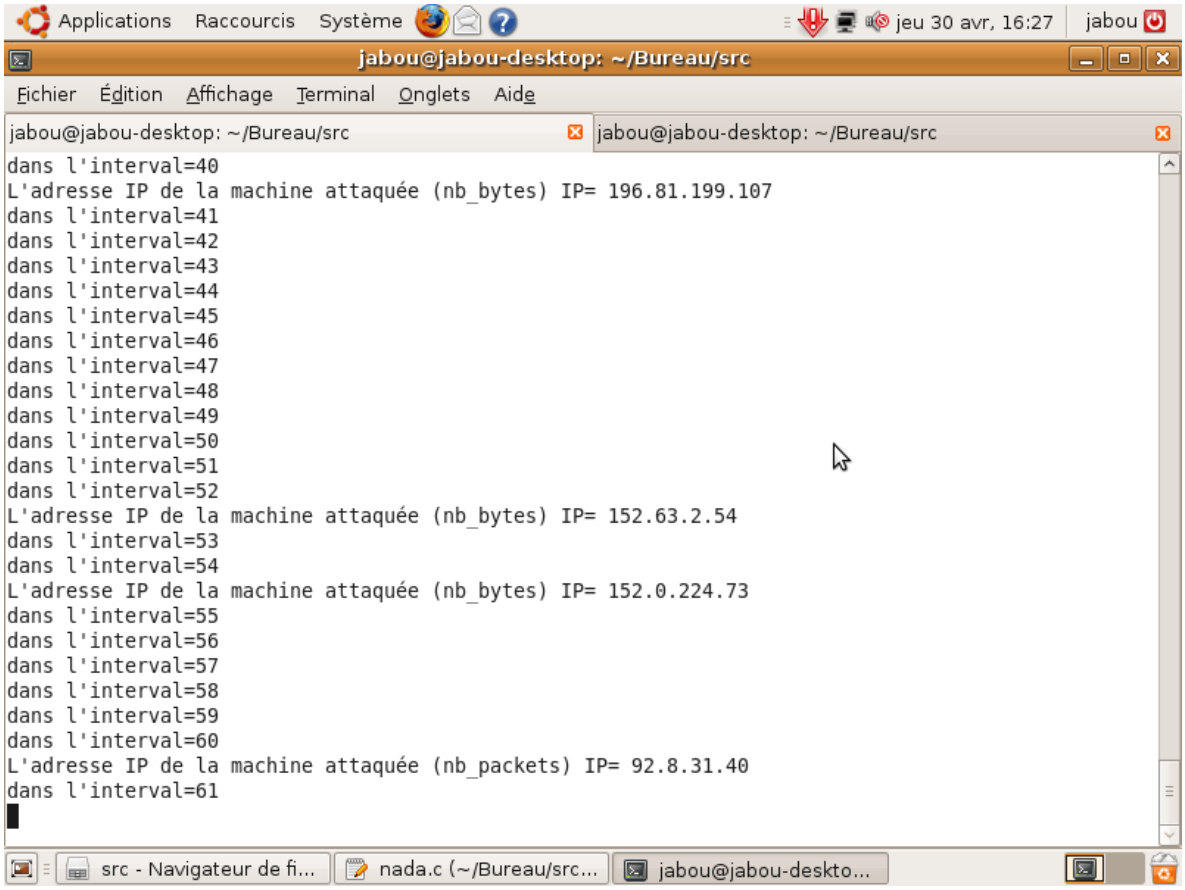


Figure 1 : Capture d'écran de l'exécution du programme

2.2 Deuxième version de l'algorithme

Dans cette version, nous avons ajouté l'interaction de l'algorithme avec la base de données.

Dans un premier temps, le programme se connecte à la base. Une fois connecté, il insère chaque flux dans la table «flow». Parallèlement, si une anomalie est détectée, l'algorithme insère une ligne dans la table «alerte» en précisant l'adresse IP destination, le type d'alerte et l'intervalle de temps sur lequel l'anomalie a été détectée.

←T→	num_interval	num_alerte	dst_ip	type_alerte_str nb_bytes=0, nb_flows=1, nb_packets=2
	6	1	26.235.51.91	nb_bytes
	32	2	10.0.0.1	nb_flows
	40	3	196.81.199.107	nb_bytes
	52	4	152.63.2.54	nb_bytes
	54	5	152.0.224.73	nb_bytes
	60	6	92.8.31.40	nb_packets
	65	7	10.0.0.2	nb_flows
	71	8	142.190.141.188	nb_bytes

Figure 2 : table alerte

num_interval	num_flow	dst_ip	src_ip	src_port	dst_port	proto	in_out	padding	start_time	start_utime	end_time	end_utime	nb_packets	nb_syn	nb_synack	nb_fin	nb_rst	nb_bytes
1	1	192.108.117.148	193.55.112.40	42309	14662	6	1	NULL	1194598784	226456	1194598842	267799	352	0	0	0	0	18580
1	2	193.55.112.40	192.108.117.148	14662	42309	6	1	NULL	1194598784	226537	1194598842	225883	615	0	0	0	0	801405
1	3	150.217.122.27	12.165.217.192	12435	2745	6	1	NULL	1194598784	231065	1194598844	87839	1020	0	0	0	0	1464892
1	4	23.171.59.168	150.217.122.77	2179	42613	6	1	NULL	1194598784	231130	1194598844	36576	3298	0	0	0	0	4120034
1	5	150.217.70.134	194.72.25.14	53	35432	17	1	NULL	1194598784	231413	1194598835	79418	12	0	0	0	0	1728
1	6	150.217.70.132	17.17.189.55	63491	110	6	1	NULL	1194598784	231450	1194598805	571332	1817	0	0	1	0	98503
1	7	17.17.189.55	150.217.70.132	110	63491	6	1	NULL	1194598784	231867	1194598805	571651	3226	0	0	1	0	4831931
1	8	150.217.122.154	20.253.70.253	1931	6881	6	1	NULL	1194598784	232153	1194598784	232153	1	1	0	0	0	48
1	9	195.66.165.4	150.217.70.134	35432	53	17	1	NULL	1194598784	232476	1194598784	232476	1	0	0	0	0	73
1	10	150.217.122.27	203.215.208.7	11356	2414	6	1	NULL	1194598784	232615	1194598841	946749	62	0	0	0	0	62581
1	11	2.185.55.44	150.217.123.65	4429	10473	6	1	NULL	1194598784	232877	1194598840	60380	490	0	0	0	0	556135
1	12	203.215.208.7	150.217.122.27	2414	11356	6	1	NULL	1194598784	233241	1194598841	947530	44	0	0	0	0	1892
1	13	150.217.122.77	20.133.44.189	27727	2210	6	1	NULL	1194598784	234217	1194598843	850279	943	0	0	0	0	338246
1	14	150.217.123.65	21.109.178.122	52194	1284	6	1	NULL	1194598784	234358	1194598843	760696	225	0	0	0	0	11093
1	15	150.217.70.134	32.85.213.146	3585	25	6	1	NULL	1194598784	234741	1194598785	136476	3	0	0	1	0	204
1	16	150.217.122.77	16.52.248.123	30939	2008	6	1	NULL	1194598784	235058	1194598844	63988	3669	0	0	0	0	4187030
1	17	150.217.70.48	194.182.25.181	49528	443	6	1	NULL	1194598784	235603	1194598801	683200	1936	0	0	0	0	105545
1	18	21.109.178.122	150.217.123.65	1284	52194	6	1	NULL	1194598784	235753	1194598843	762530	363	0	0	0	0	413824

Figure 3 : table flow

3- Conclusion

L'algorithme CUSUM présente plusieurs avantages notamment la rapidité de détection ainsi que la minimisation du taux de fausses alertes ce qui n'est pas le cas de tous les systèmes de détection d'intrusions.
Malgré l'efficacité de cet algorithme il présente diverses limites. Par exemple la nécessité d'avoir un contrôle humain pour verifier les différentes attaques. Il est également impossible de connaître le type d'attaque, ce qui sera comblé dans la suite grâce à la classification faite au niveau de l'interface PHP.

V Interface et base de données

1- Introduction

Dans le cadre de notre projet, une interface PHP liée à une base de données est nécessaire pour afficher les différentes alertes générées par CUSUM d'une part, et d'autre part pour pouvoir faire des traitements sur les tables (flow et alerte) comme la classification, la vérification…

Nous commençons par une étape de conception dans laquelle nous utiliserons la méthode UML pour pouvoir schématiser les différentes tables de la base. Par la suite nous présenterons l'interface PHP puis la base de données.

2- Conception

La conception était une phase nécessaire pour la réalisation de notre application. Tout d'abord nous avons choisis la méthode utilisée, puis nous avons décrit l'application à travers des diagrammes.

2.1 Méthode de conception utilisée

Les méthodes objet proposent de structurer un système sans centrer l'analyse uniquement sur les données ou uniquement sur les traitements, mais sur les deux (données et traitements) à la fois.

Pour la conception de notre système on a adopté le langage de modélisation unifié UML.

UML est l'acronyme de Unified Modeling Language (ou en français Langage de Modélisation Unifié). Il est né de la fusion des trois méthodes qui ont le plus influencé la modélisation objet au milieu des années 90 : OMT, Booch et OOSE.

Normalisé en fin 1997 par L'OMG (Object Management Group), le langage UML a plusieurs avantages :

- ✓ Il donne une dimension méthodologique à l'approche objet permettant une meilleure maîtrise de sa richesse.
- ✓ Il peut être intégré à n'importe quel processus de développement de logiciel de manière transparente.

- ✓ Il n'impose pas de méthode de travail particulière.
- ✓ Il permet d'améliorer progressivement les méthodes de travail, tout en préservant les modes de fonctionnement.

2.2 Diagrammes

2.2.1 Diagramme de cas d'utilisation

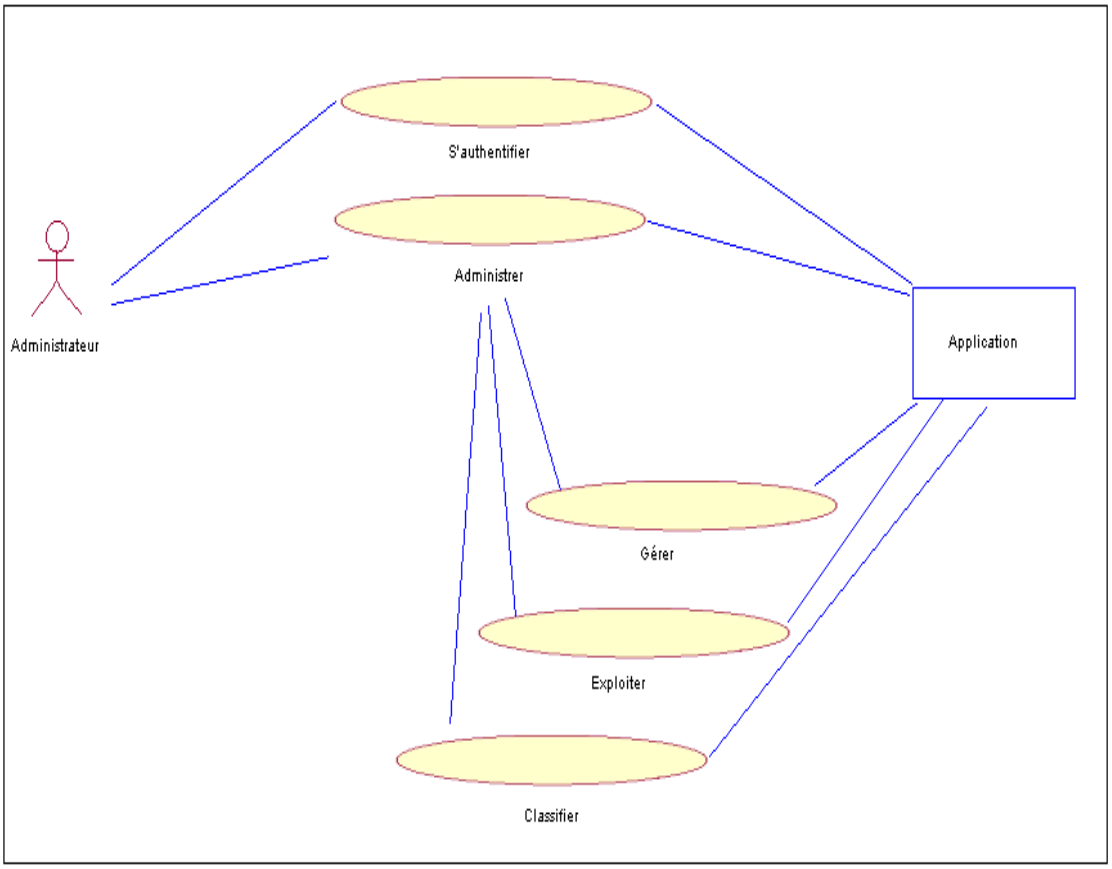


Figure 4 : Diagramme de cas d'utilisation

2.2.2 Diagramme de classe

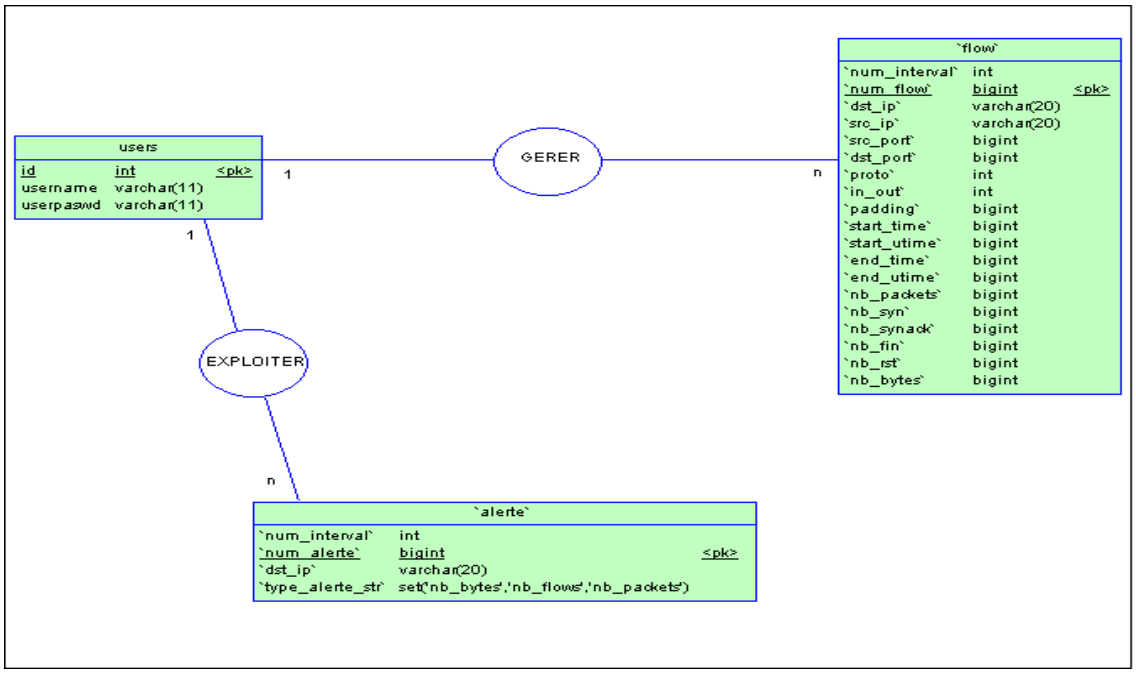


Figure 5 : Diagramme de classe

3- Outils et matériaux de réalisation

Dans cette partie nous allons présenter les principaux outils utilisés pour la mise en place de notre application.

La réalisation de cette application a été faite sous la plateforme d'EasyPHP qui permet d'intégrer le langage de script PHP, MYSQL et le serveur Web Apache :

- **EasyPHP :** est un environnement simple et facilement utilisable de développement web avec MySQL et PHP. Il est important de noter que PHP est un langage qui s'exécute au niveau du serveur Web Apache.

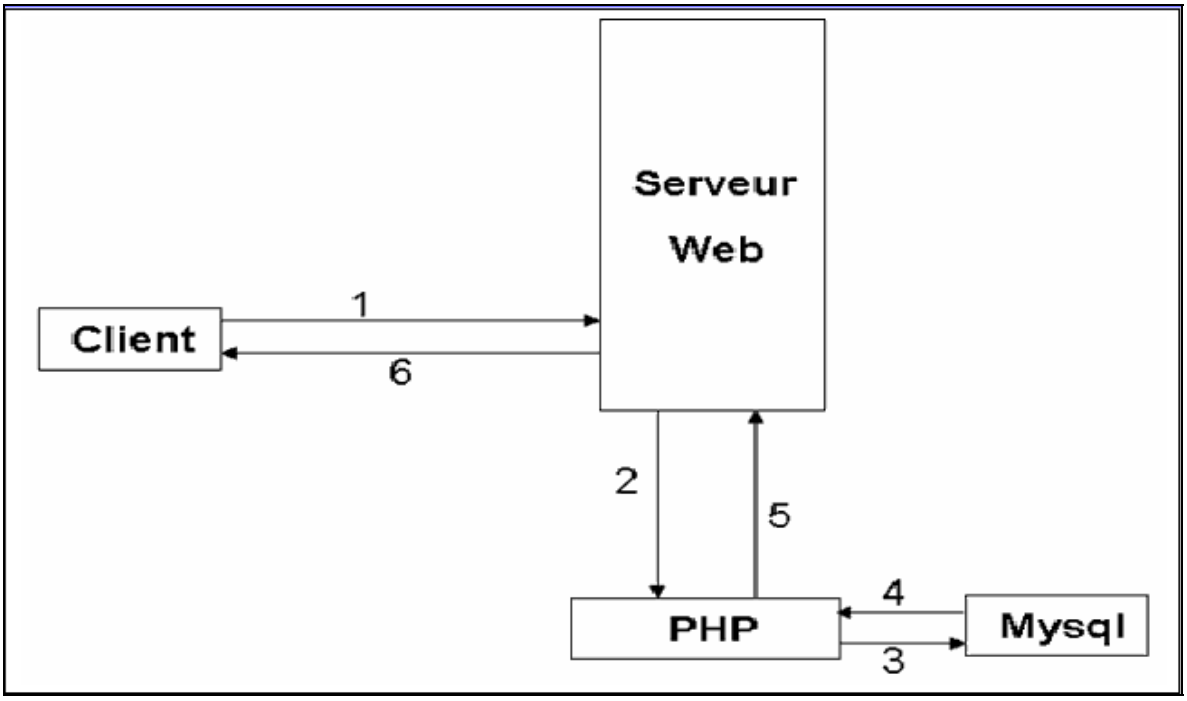


Figure 6 : interaction entre les éléments d'une application web dynamique

- **PHP :** PHP est un langage de script qui permet de générer des pages HTML. Le but du langage PHP est de permettre aux développeurs de site web d'écrire rapidement des pages web dynamiques.

Il offre aussi les possibilités suivantes :

- Interagir avec une base de données par l'intermédiaire de fonctions
- Construire les requêtes en utilisant un script PHP
- Afficher les résultats des requêtes avec HTML
- Envoyer des requêtes SQL au serveur

- **MYSQL :** est un système de gestion de bases de données relationnelles il permet :
  - La création de base de données
  - L'intégration de requête SQL pour la gestion et manipulation de données de la base

4- Base de données MySQL

D'habitude de larges volumes de données sont gardés dans des bases construites selon une approche standardisée et structurée ce qui permet de gérer des données de la manière la plus efficace.

Nous avons besoin de sauvegarder une base de très grande taille dans un fichier extérieur vu que le volume des données très important occasionnerait un ralentissement de l'accès à la base.

4.1 Comment faire une sauvegarde de base des données?

Nous utilisons un script Shell afin de créer une sauvegarde de toutes les bases de données.

Ce script réalise la sauvegarde des données sous un fichier CSV qui peut être ultérieurement restauré, et il effectue également la compression des données sauvegardées sous forme d'archive ".gz".

Commandes utilisées :

- **MySQL dump *:*** qui permet d'exporter une base vers un fichier texte, pour la sauvegarde :

mysqldump –user=$MYSQLUSER --password=$MYSQLPASSWORD

- **gzip :** permet de compresser le fichier de sauvegarde afin de *réduire sa taille*

$DATABASE | gzip > $ARCHIVEPATH/$SAVEDIR

Nous présentons une capture d'écrans du script de sauvegarde des données de la table flows :

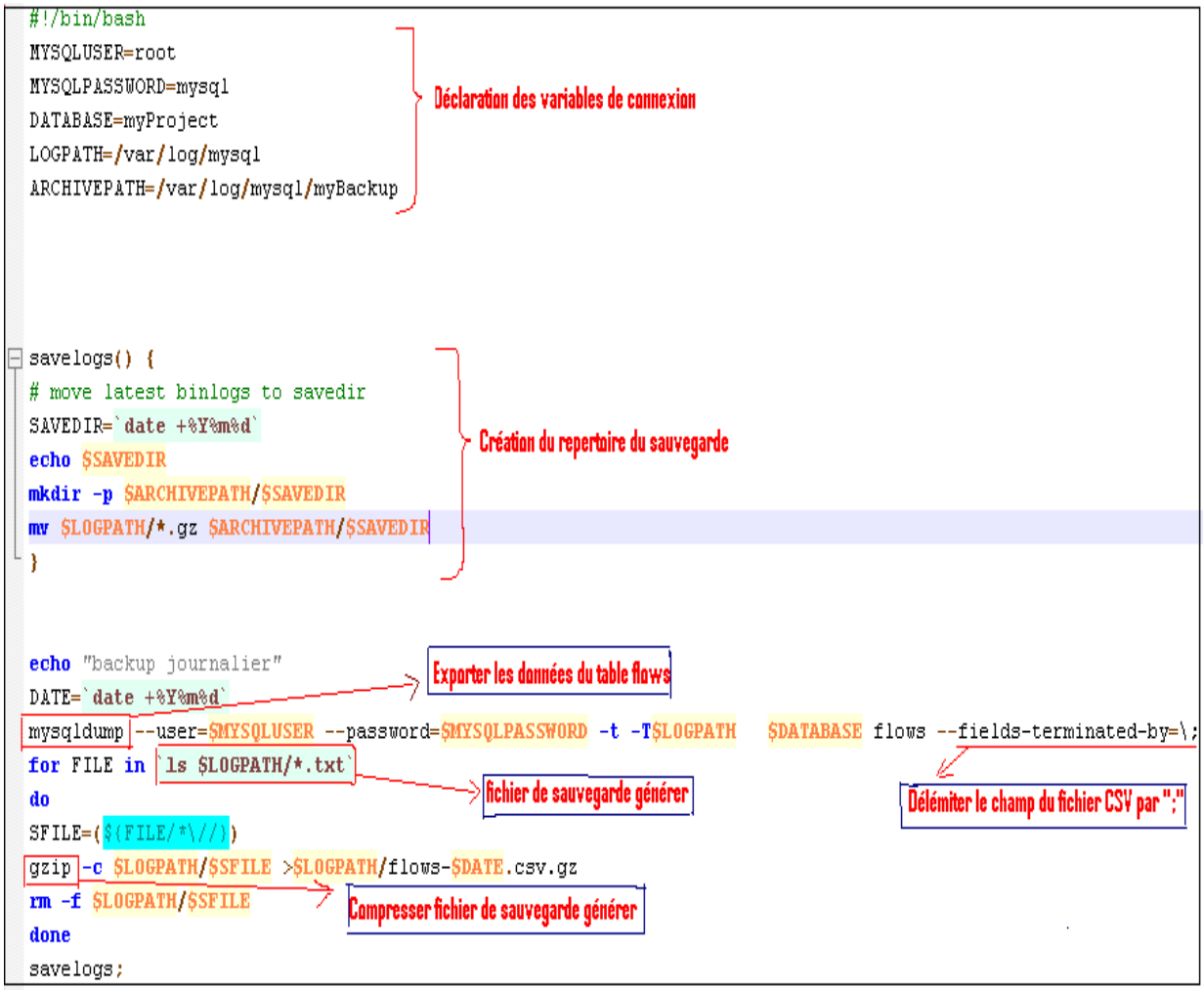


Figure 7 : Script d'export de la base

4.2 Comment restaurer les données des sauvegardes ?

Il est possible d'importer ces sauvegardes par la création d'un script Shell qui permet de :

- Restaurer les données archivées pendant la phase de sauvegarde
- Créer une table temporaire qui permet de manipuler les données importées
- Récupérer les données de fichier de sauvegarde

Commandes utilisées :

- **gunzip:** qui permet de décompresser les fichiers de sauvegarde

gunzip $ARCHIVEPATH/$DATE2/flows-$DATE2.csv.gz

mysqlimport : qui permet d'importer des données depuis des fichiers texte

mysqlimport --user=$MYSQLUSER --password=$MYSQLPASSWORD --fields-terminated-by=";" --local --lines-terminated-by="\n" $DATABASE $ARCHIVEPATH/$DATE2/flows_$DATE2.csv;

Nous présentons une capture d'écrans du script de restauration des données des sauvegardes :

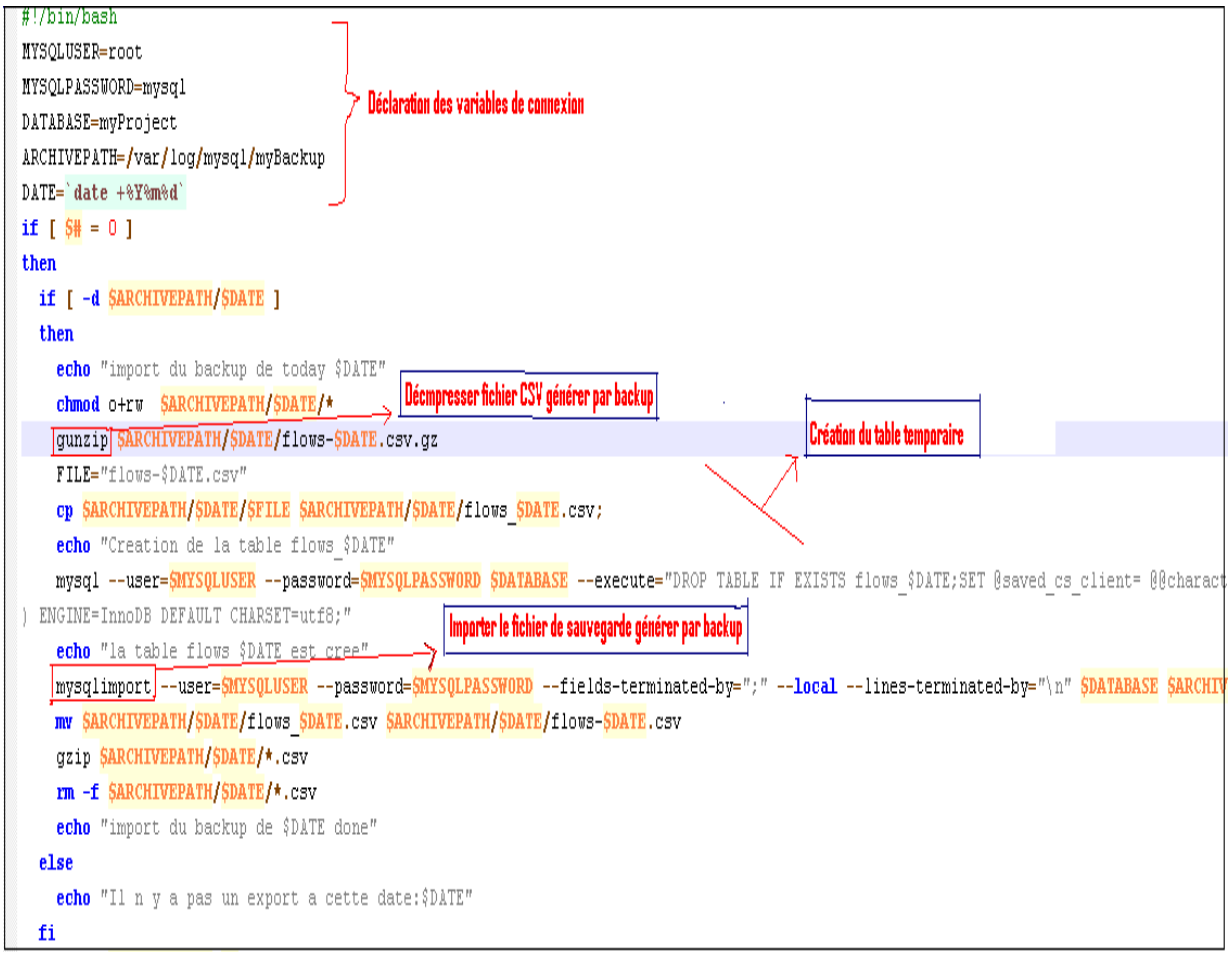


Figure 8 : Script d'import de la base

4.3 Automatisation de sauvegarde de base donnée

La sauvegarde de base de données est automatisée par l'utilisation de **Cron Daily** qui est un *daemon* utilisé pour programmer des tâches qui doivent être exécutées à un moment précis. Chaque utilisateur a un fichier **crontab** lui permettant d'indiquer les actions (et à quelles périodes) elles devront être exécutées.

Il faut pour cela lancer le logiciel crontab dans une console puis taper la commande suivante :

m h j M a root ./backupCSV.sh

Les paramètres m (minute), h (heure), j (jour), M (mois) et a (année) servent à indiquer les dates auxquelles le script devra être lancé. Par exemple pour lancer la commande à minuit tous les jours de tous les mois de chaque année, il suffit de taper « 0 0 * * * root ./backupCSV.sh »

5- Interface PHP

Dans cette partie nous allons présenter l'interface PHP qui va offrir a l'administrateur du réseau quelques options en plus telles que vérifier, classifier ainsi que plein d'autres fonctions.

Après la phase d'authentification, l'utilisateur a accès à plusieurs fonctionnalités. Parmi elles, la gestion des utilisateurs qui réservée à l'administrateur lui permet de rajouter, supprimer, modifier, lister des utilisateurs.

Les trois figures suivantes montrent un aperçu de ces fonctionnalités.

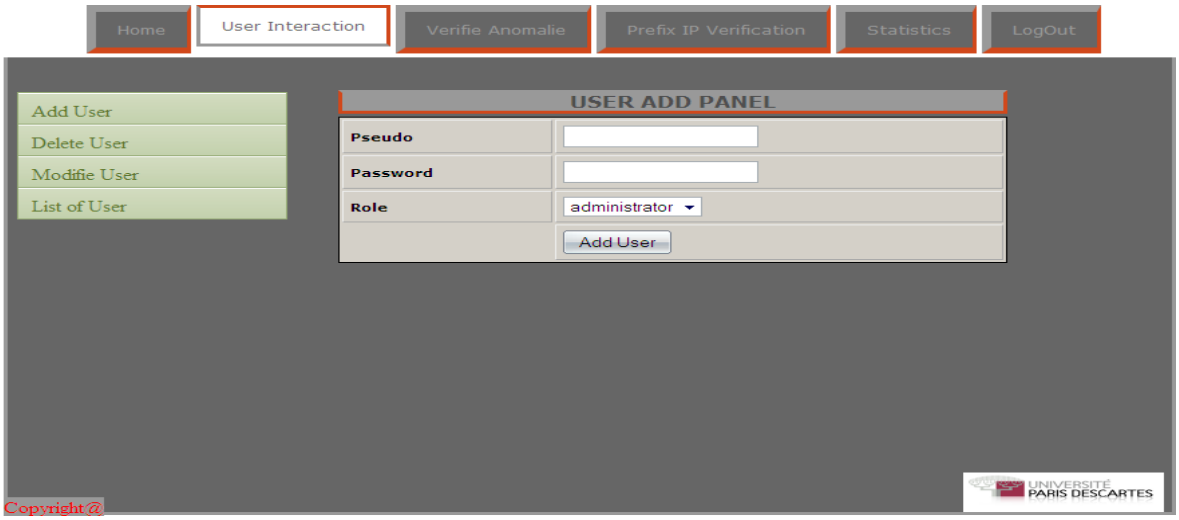


Figure 9 : page d'ajout d'un utilisateur

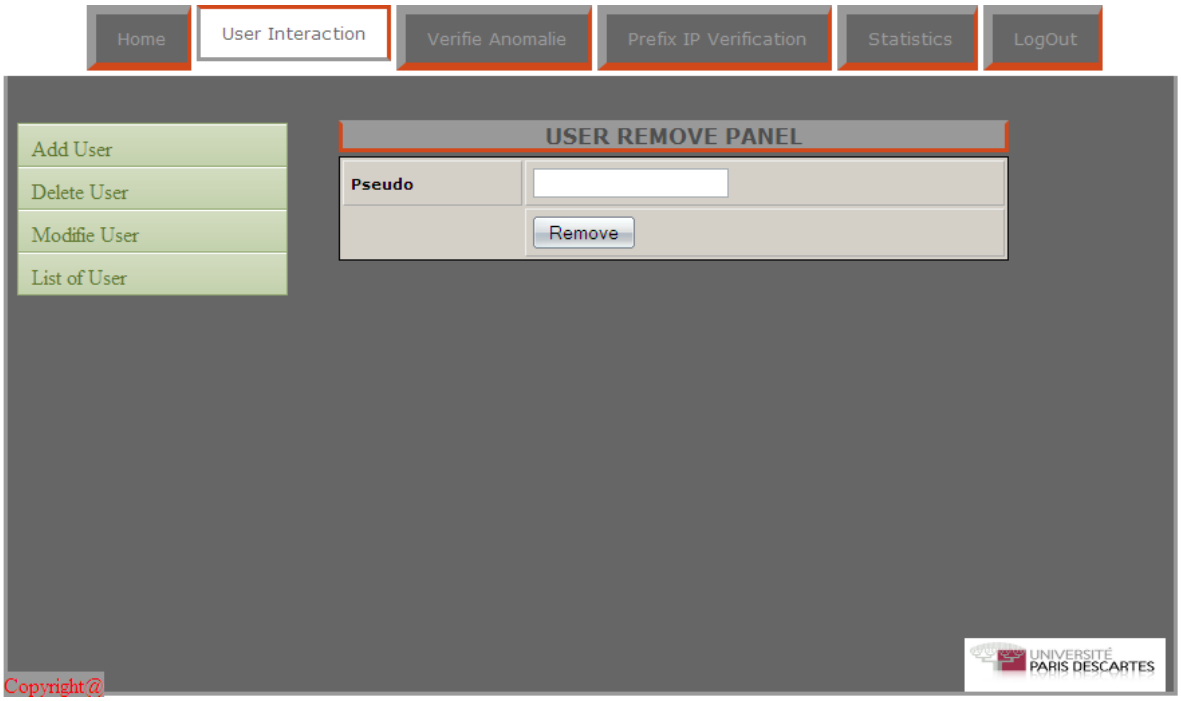


Figure 10 : page de suppression d'un utilisateur

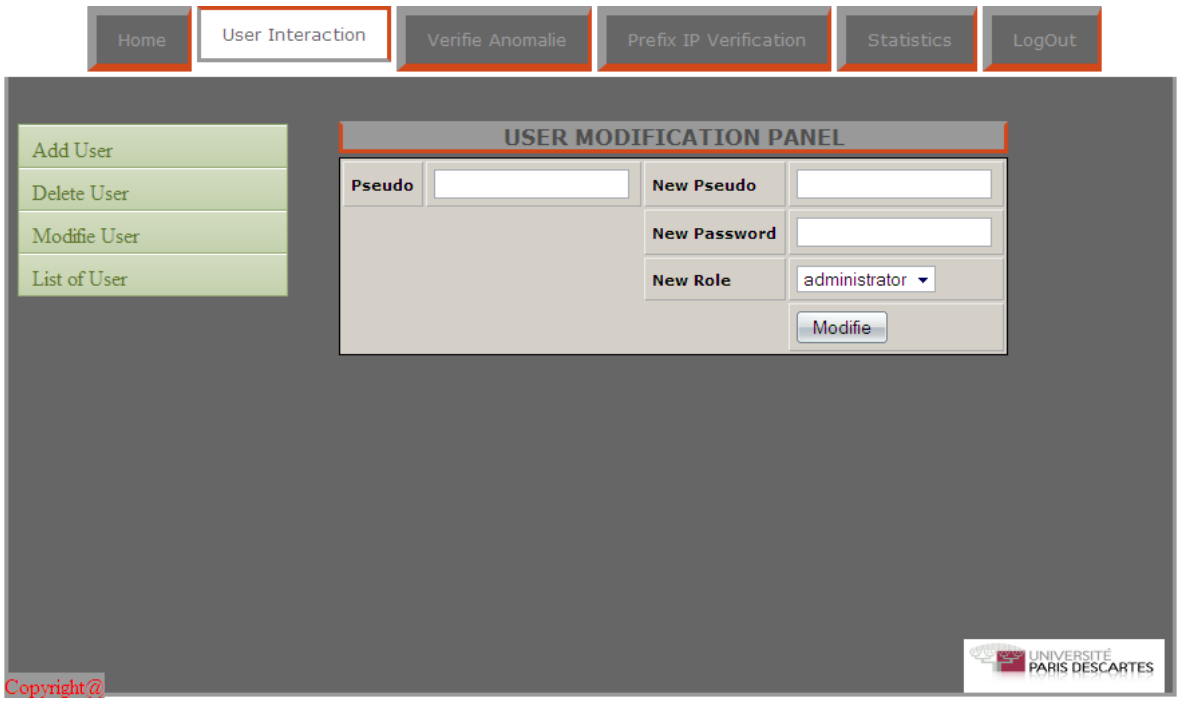


Figure 11 : page de renommage d'un utilisateur

Le menu « statistics », que l'on peut voir sur la capture d'écran suivante, nous offre la possibilité d'exploiter la base de données, et plus précisément la table « flow », afin d'en extraire des informations statistiques sur l'adresse IP destination, l'adresse IP source ou le port destination, ordonnées selon le nombre de paquets, le nombre SYN, FIN, RST, BYTE ou le nombre de ACK.

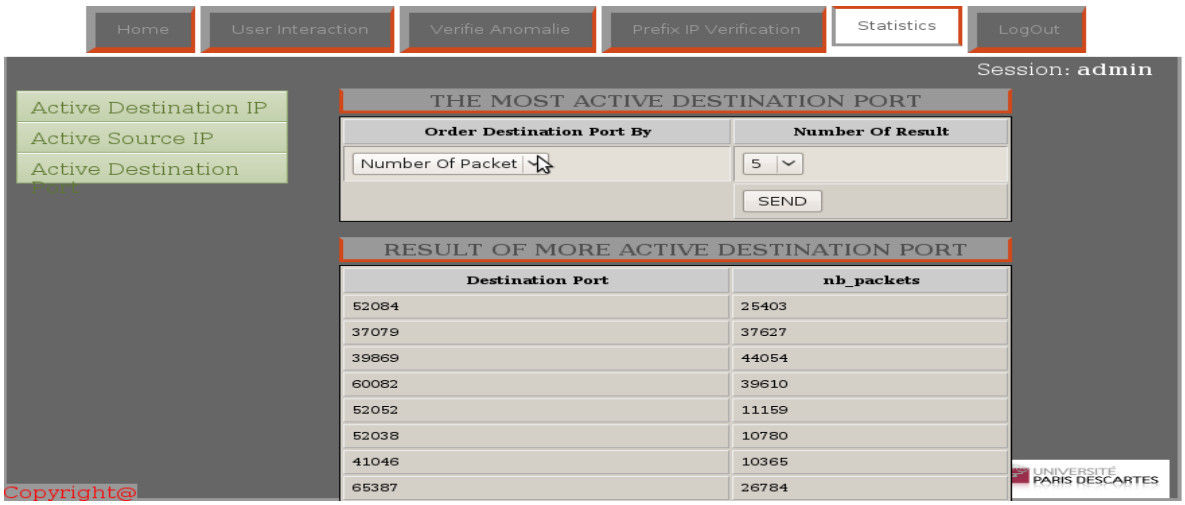


Figure 12 : page de statistics

La capture d'écran suivante présente le menu « Prefix IP verification ». C'est une option qui permet d'exploiter la table flow de la base de données pour sélectionner les informations concernant les adresses commençant par un préfix saisi. Cet affichage peut être ordonné selon le nombre de paquets, le nombre de SYN, etc…

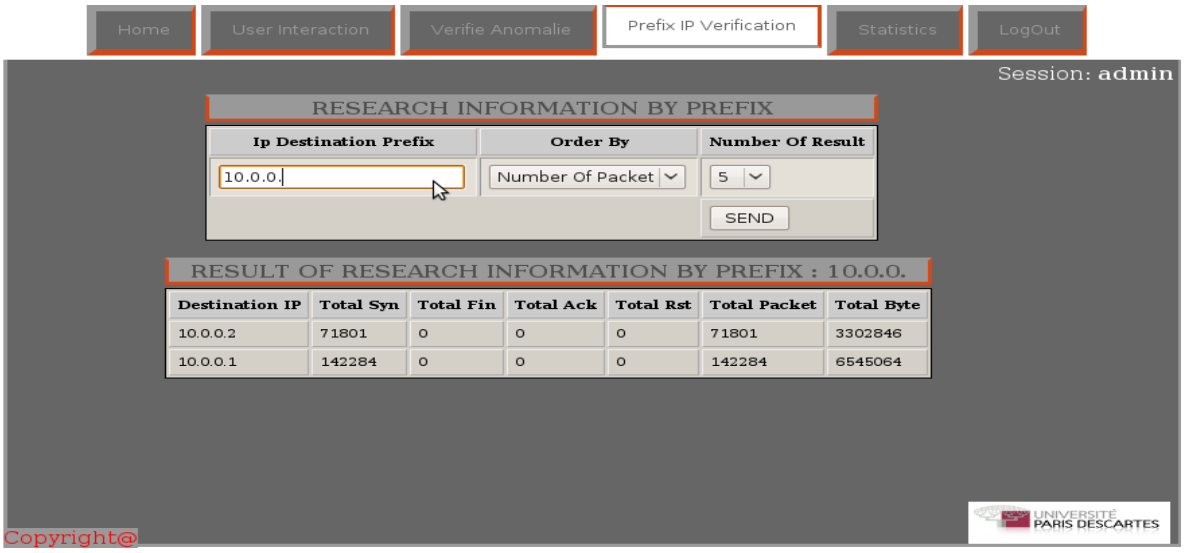


Figure 13 : page Prefix IP Verification

Après le lancement d'une alerte, le menu « Vérifier anomalie » permet à l'administrateur de confirmer que cette attaque n'est pas une fausse alerte. Cette vérification peut s'effectuer pour quatre types d'attaques qui sont les suivants :

- Net scan : l'administrateur est invité à saisir l'adresse IP source et le port destination pour connaître l'adresse IP destination sur laquelle le port est ouvert et est attaqué par l'adresse IP source saisie.

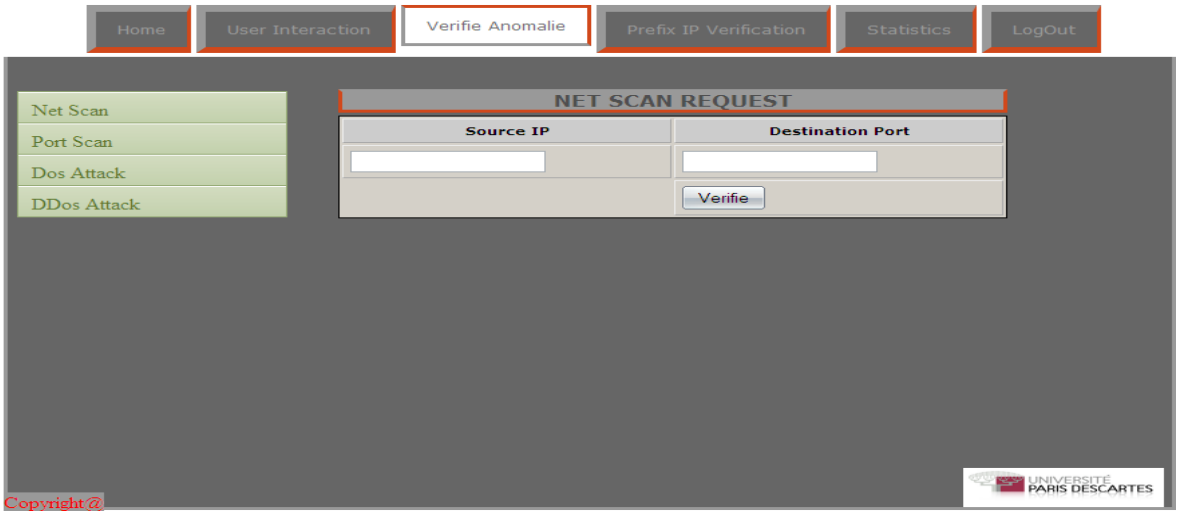


Figure 14 : page Verifie Anomalie –Net Scan

- Port scan : la saisie de l'adresse IP source et de l'adresse IP destination donnera la possibilité à l'administrateur de vérifier si l'adresse IP destination subit un port scan par l'adresse IP source.

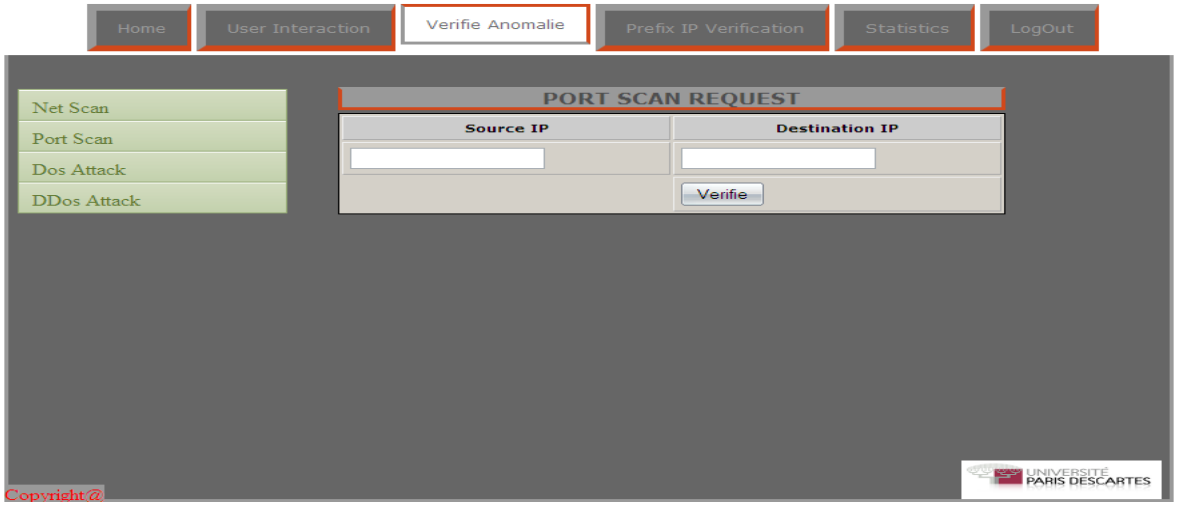


Figure 15 : page Verifie Anomalie –Port Scan

- DoS : après la saisie des adresses IP source et destination ainsi que le port destination, l'administrateur peut savoir si la machine possédant l'adresse IP destination saisie subit une attaque de type DoS sur le port correspondant.

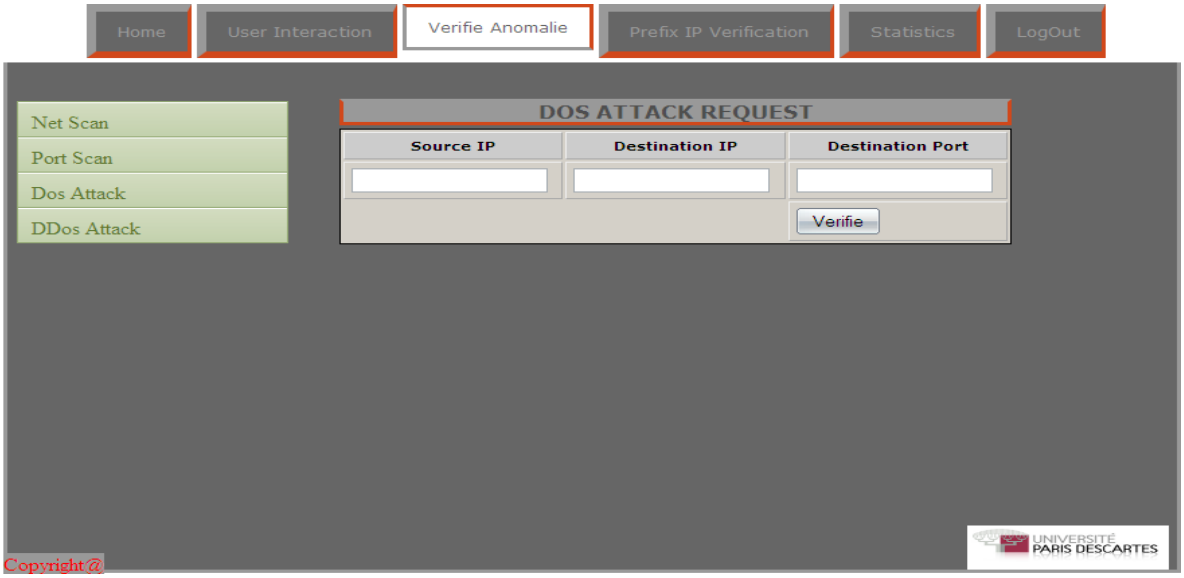


Figure 16 : page Verifie Anomalie –Dos Attack

- DDoS : dans ce cas, la vérification s'effectue de la même façon que pour le DoS, à l'exception près qu'aucune adresse IP source ne doit être indiquée vu qu'une attaque DDoS est lancée par plusieurs sources à destination d'une même machine.

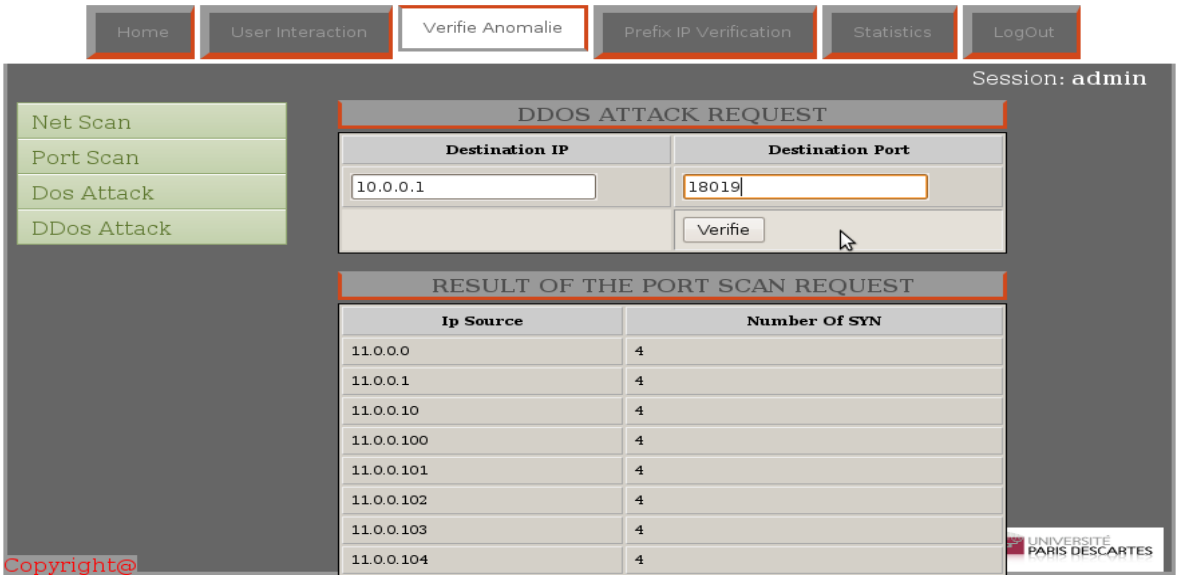


Figure 17 : page Verifie Anomalie –DdoS Attack

6- Conclusion

Pour valoriser notre travail une interface était nécessaire afin de présenter les différentes informations telles que les anomalies, et plus généralement les données stockées dans la base. Elle permet aussi d'offrir la possibilité à l'administrateur d'exploiter et interpréter ces données afin de l'aider à découvrir par exemple quel est le type d'une attaque, son origine et d'autres informations.

À ceci nous avons rajouté une option d'archivage qui était indispensable vu que l'insertion des flows du trafic dans la base de données fait augmenter sa taille très rapidement. De plus nous avons implémenté l'option inverse, c'est à dire la récupération de la base sauvegardée.

VI Conclusion

Sur Internet, les pirates emploient de plus en plus diverses stratégies pour dissimuler leurs caractères intrusifs. Par conséquent, la détection de la reconnaissance active devient plus difficile mais indispensable afin de comprendre les intentions des attaquants.

Nous avons suivi dans le cadre de notre projet TER plusieurs étapes afin de parvenir à notre but, celui de sécuriser le réseau contre les attaques.

Dans le premier chapitre la protection du réseau était notre but premier ce qui nous a menés à étudier les procédés mis en œuvre par les attaquants ainsi que les outils qu'ils utilisent pour lancer les attaques. Nous avons tout naturellement enchainé sur l'étude de quelques techniques de protection comme les pare-feu et les systèmes de détection d'intrusions que l'on a détaillés par la suite.

Dans le second chapitre nous avons pu étudier les systèmes de détection d'intrusions, leurs critères de classification et les différents types que l'on peut trouver. Cette étude nous a été nécessaire pour élaborer une solution adéquate que nous avons développée dans le chapitre suivant.

Le développement de l'algorithme CUSUM était l'objectif du troisième chapitre dans lequel nous avons détaillé le fonctionnement, les étapes de l'implémentation ses avantages et ses limites.

La nécessité de stocker les traces des anomalies produites par CUSUM nous a poussés à réaliser une base de données et par la suite une interface pour l'exploiter. Nous avons développé des fonctionnalités au sein de cette interface pour permettre à un administrateur d'exploiter la base. Cette dernière étant très volumineuse, des besoins d'archivage se sont fait sentir et nous avons remédié à cela en implémentant un script qui effectue cet archivage et s'exécutant automatiquement. Tout ceci a été exposé dans le dernier chapitre.

Ce travail nous a beaucoup apporté dans le domaine de la sécurité des réseaux. Cette application que nous avons élaborée présente des avantages comme la détection rapide des anomalies ainsi qu'un taux de fausses alertes limité.

VII Bibliographie

[1] Silvia Farraposo, Philippe Owezarski, Edmundo Monteiro « Détection, classification et identification d'anomalies de trafic »

[2] Ghislain Verdier, Nadine Hilgert et Jean-Pierre Vila « Une méthode statistique de détection d'anomalie pour les modèles `a espace d'´etat non linéaires »

[3] J. AUSSIBAL, P. BORGNAT, Y. LABIT, G. DEWAELE, N. LARRIEU, L. GALLON, P. OWEZARSKI, P. ABRY, K.BOUDAOUD « *Base de traces d'anomalies légitimes et Illégitimes »*

[4] Yuefeng Li, Mark Looi, Ning Zhong «Advances in intelligent IT»

[5] Alexander Clemm, Lisandro Zambenedetti Granille, Rolf Stadler, International « Managing virtualization of networks and services »

[6] Eric Cole « Hackers beware »

[7] Cornel-Marius Matei, Jacques Duchêne, Igor Nikiforov « Détection de ruptures dans les signaux électromyographiques »

[8] Nadine Hilgert, Ghislain Verdier & Jean-Pierre Vila « Filtrage et détection de rupture dans les modèles à espace d'état »

[9] Sheng-Ya Lin, Jyn-Charn Liu, Wei Zhao « Adaptive CUSUM for Anomaly Detection and Its Application to Detect Shared Congestion »

[10] B. BENMAMMAR, C. LÉVY-LEDUC, F. ROUEFF « Algorithme de détection d'attaques de type (SYN Flooding) »

[11] http://www.eastman-watch.cn/distributed-denial-of-service-attacks-tfn2k-attacks-and-iptables-filtering-test/

[12] http://hackersparadise.synthasite.com/software.php

VIII Annexe

Courbes de variation pour l'adresse destination : 88.123.111.137