TER
Détection des attaques de Déni de Service dans les réseaux IP

Elaboré par : Radoslava TATAROVA & Gaetano GIARMANA
Master 1 Informatique

Encadré par: M. Osman SALEM

2009/2010

REMERCIEMENTS

*Avant tout nous tenons à remercier **M. Osman Salem**, notre encadreur, pour sa disponibilité, son expertise et ses précieux conseils durant toute la durée de ce projet TER.*

Table des matières

I. Introduction

Les systèmes d'information sont aujourd'hui de plus en plus ouverts sur Internet.

Cette ouverture, a priori bénéfique, pose néanmoins un problème majeur : il en découle un nombre croissant d'attaques.

Dans une attaque par déni de service, un utilisateur exploite la connectivité d'Internet pour paralyser les services offerts par le site ciblé. Ce qui 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.

Ce projet vise à proposer une extension pour l'algorithme de détection d'anomalies CUSUM qui est utilisé dans le réseau IP pour la détection, l'identification et la classification d'anomalies (comme : DoS, DDoS). Il s'agit de proposer une amélioration pour augmenter le taux de détection et réduire le taux de fausses alarmes.

II. Attaques et techniques de protection:

Internet est soumis à des fortes variations de trafics, certaines légitimes et d'autres illégitimes telles que les attaques qui exploitent des failles liées au protocole TCP/IP qui est le protocole le plus utilisé dans le réseau et qui n'est malheureusement pas sécurisé. Les principaux types d'attaques sont :

1. 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.

1.1 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…

- 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…).

- 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.

- Progression : il est temps pour l'attaquant de réaliser ce pourquoi il a franchit les précédentes étapes. Le but ultime étant d'obtenir les droits de l'utilisateur root (ou system) sur un système afin de pouvoir y faire tout ce qu'il souhaite (inspection de la machine, récupération d'informations, nettoyage des traces...).

1.2 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 donc aisé de le lire. De même, il est facile de savoir à tout moment quelles pages web sont consultées par 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". Cette attaque n'est intéressante que dans la mesure où la machine de confiance dont l'attaquant à pris l'identité peut accéder au serveur cible en tant que root.

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 « WS_Ping ProPack ».

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.

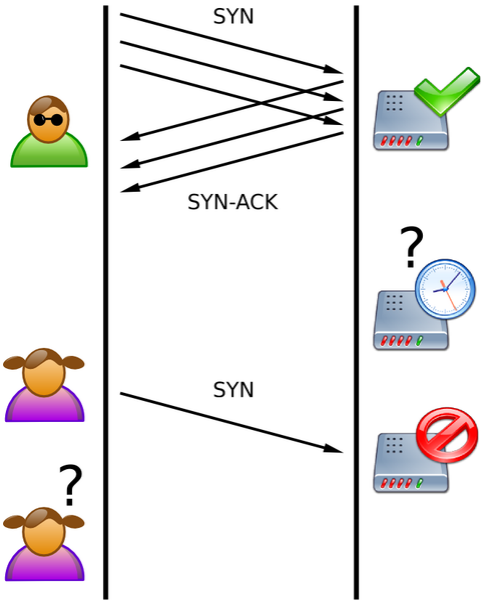

Figure 1 : syn flooding

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.

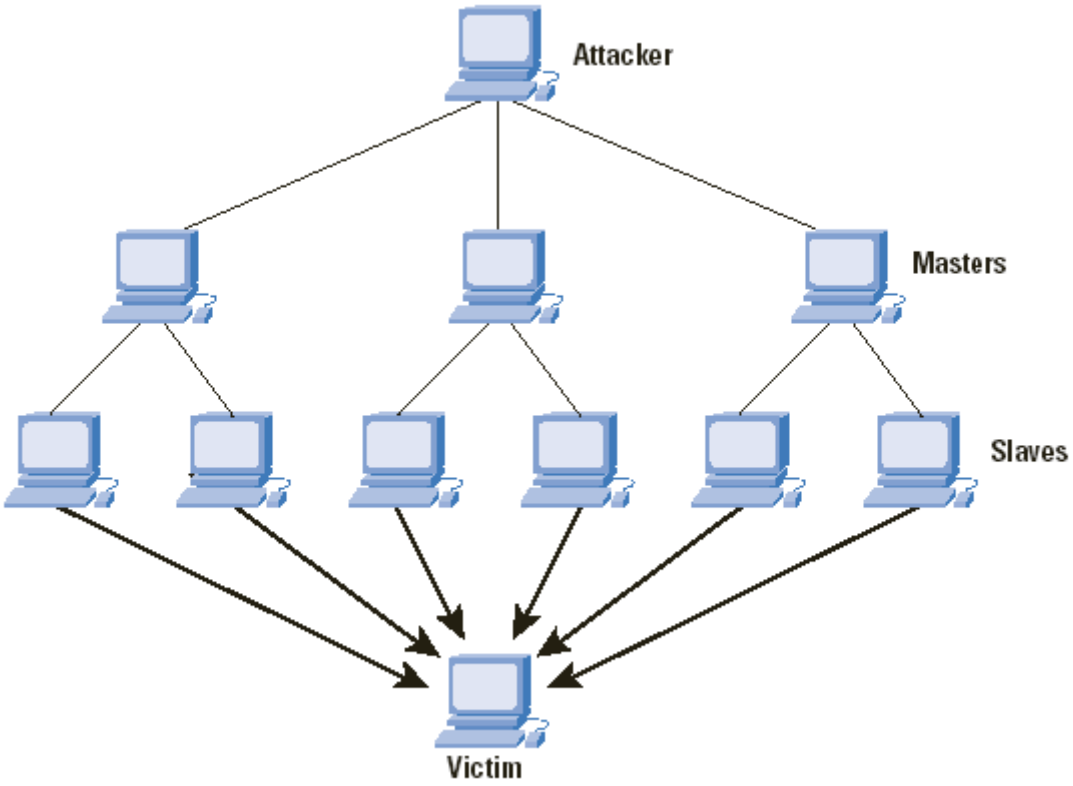

Figure 2 : DDoS

- **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.

2. 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, etc.… dans le but de maintenir la protection du réseau informatique.

Nous détaillerons un peu plus loin de ce rapport chacune de ces méthodes et nous soulignerons leurs limites.

2.1. 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.

2.2. 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.

III. Algorithmes de détection des attaques de déni de service dans les réseaux IP

1. Algorithmes

La sécurité contre les attaques distantes se renforce, notamment par le biais d'équipements réseaux plus puissants (comme des firewalls plus intelligents), mais les attaques locales restent toutefois encore fort efficaces : l'ARP Spoofing, le vol de session, … restent souvent possibles.

L'informatique évolue, les applications sont de plus en plus complexes et les délais laissés aux programmeurs et administrateurs sont souvent très (trop) courts.
Les risques de failles applicatives sont, de ce fait, très grands et peuvent s'avérer dangereux pour des applications largement répandues.

Les attaques distribuées seront toujours redoutables si la plupart des machines personnelles ne sont pas protégées.
Ce qui nous amène à notre seconde partie : comment détecter et empêcher ces attaques ?

Plusieurs études se sont intéressées aux problèmes de détection d'anomalies. Dans cette partie nous présenterons 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.

1.1 Algorithme à seuil adaptatif (adaptive threshold)

C'est un algorithme plutôt simple qui détecte les anomalies basé sur la violation d'un seuil qui est défini de manière adaptative en utilisant des mesures récentes du trafic et basé sur une évaluation du nombre moyen de paquets SYN.

L'algorithme considère seulement les violations du seuil et pas l'intensité de ces violations.

Une application directe de cette méthode rapporterait un taux élevé de faux positifs (fausses alertes). Une solution simple pour améliorer son exécution, serait de signaler une alarme après un certain nombre de violations consécutives du seuil.

1.2. 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 certain seuil global de trafic. Ce comportement est similaire à celui de l'algorithme à seuil adaptatif. À la différence de ce dernier qui considère seulement les violations du seuil, l'algorithme CUSUM considère le volume excessif envoyé au dessus du volume normal et par conséquent explique l'intensité des violations.

IV. Algorithme CUSUM

1. Introduction

Étant une méthodologie 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 à un fonctionnement anormal 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 produites par l'algorithme que nous allons utiliser dans l'interface PHP pour les classifier, les vérifier et les afficher. Nous avons également ajouté une nouvelle façon de détecter une attaque de SYN flooding grâce aux conseils de M. Osman SALEM.

2. Fonctionnement

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

$$G_n = [\ G_{n-1} + (X_n - \mu)\ ]$$

Où G_n 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_n est la variable représentant soit le nombre de paquets SYN.

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

La moyenne se calcule selon la formule suivante :

$$\mu_n = 0.5*\mu_{n-1} + 0.5*X_{n-1}$$

Lorsque G_n dépasse un certain seuil **h** ($G_n \geq h$), une alarme se déclenche et réinitialise G_n 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.

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 «flux». Parallèlement, si une anomalie est détectée, l'algorithme insère une ligne dans la table «alarme» en précisant l'intervalle de temps sur lequel l'anomalie a été détectée.

Nous avons également ajouté une nouvelle façon de détecter une attaque de SYN flooding grâce aux conseils de M. Salem. En effet, M. Salem ayant assisté à une conférence d'Andrew Clark qui travaillait à l'université de Science et technologie de

Queensland Brisbane, en Australie, nous a montré son amélioration de l'algorithme CUSUM que nous avons repris.

Au lieu d'utiliser l'algorithme CUSUM interval par interval, il utilise une «fenêtre» de taille variable selon les circonstances. Au départ, la taille de la fenêtre est minimum (pour notre projet nous avons choisi 3 comme taille minimum pour la fenêtre), puis elle grandit jusqu'à atteindre une taille maximum (qui pour notre projet est 10). En cas d'attaque, la fenêtre revient à sa taille minimum et ne recommencera à grandir que lorsque $\mathbf{G_n}$ < **h_end** où **h_end** est égale au seuil avant l'attaque divisé par 4.

3. Conclusion

L'algorithme CUSUM présente plusieurs avantages notamment la facilité de la mise en ligne, 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 l'incapacité de vérifier tout le trafic dans le cas d'un très haut débit et la nécessité d'avoir un contrôle humain pour gérer les différentes attaques.

V. Interface et base de données

Dans le cadre de notre projet, une interface PHP liée à une base de données était nécessaire pour afficher les différentes alertes générées par CUSUM.

1. Outil et matériaux de réalisation

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

La réalisation du cette application a été faite sous la plateforme WampServer qui permet d'intégrer le langage de script PHP et le MYSQL :

WampServer : est un environnement simple et facilement utilisable de développement web avec MySQL et le langage de script PHP qui est le plus utilisé pour la réalisation de sites web dynamiques.

Il est important de comprendre que PHP est un langage qui s'exécute au niveau serveur.

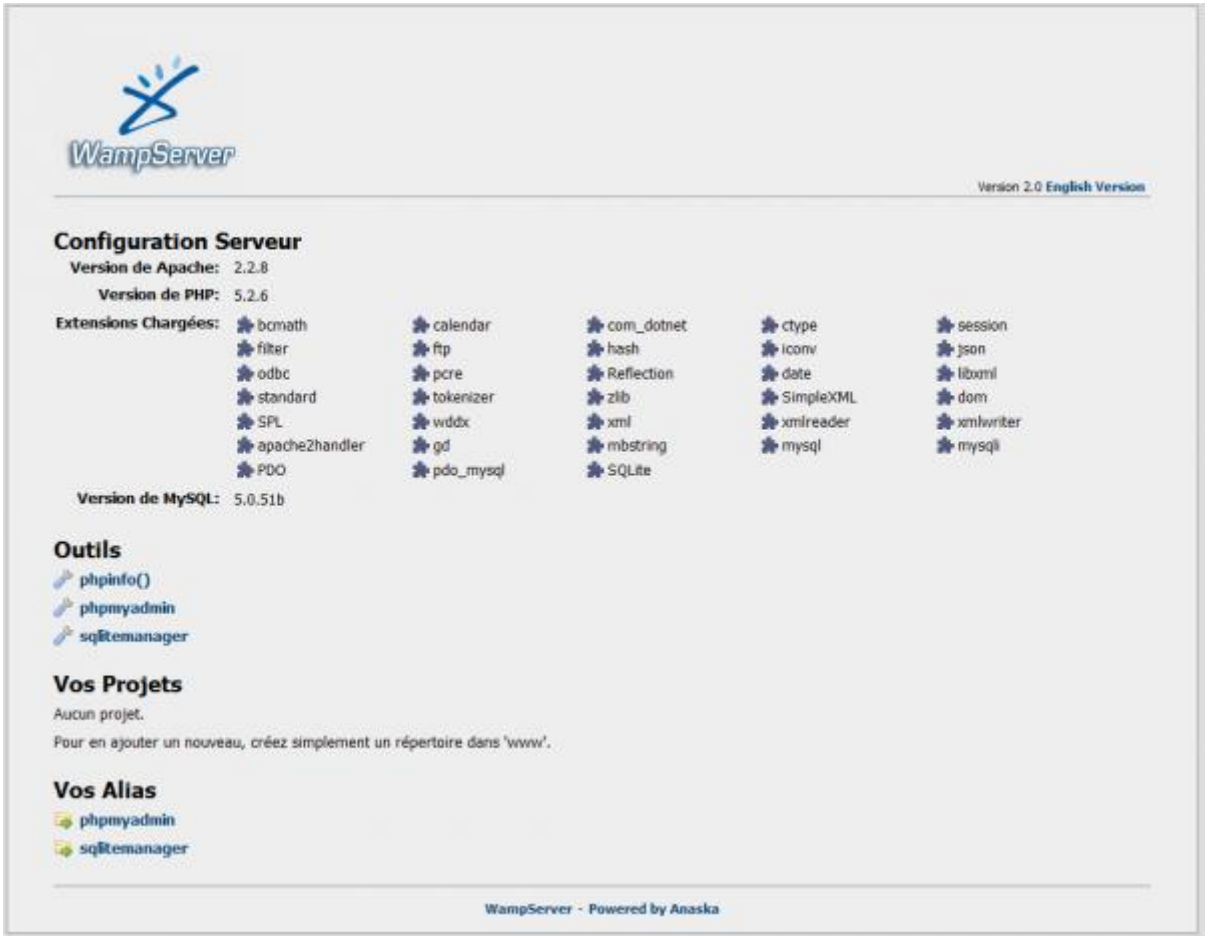

Figure 3 : wanpserver

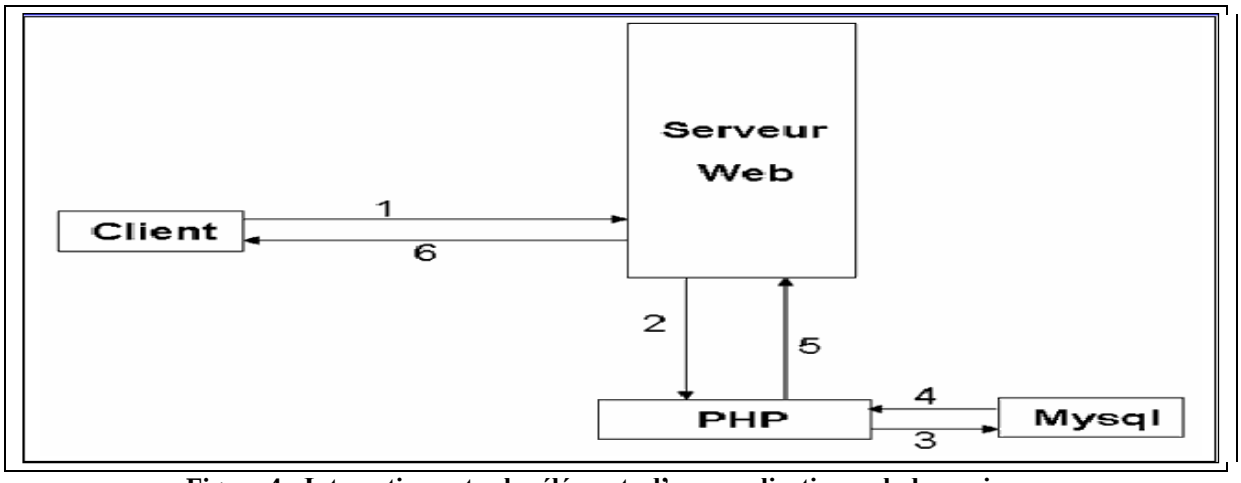

Figure 4 : Interaction entre les éléments d'une application web dynamique

PHP : PHP est un langage de script qui est inclut dans le langage 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 écrivant un programme 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.

2. 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.

La sauvegarde de bases de données représente un problème assez complexe.
Tout d'abord, toutes les bases de données n'ont pas l'option de sauvegarde.

Par exemple, MySQL n'en a pas, c'est pourquoi cette base ne peut être sauvegardée qu'à l'aide d'un logiciel spécial tel que WampServer avec son interface phpMyAdmin. D'autre part, nous avons besoin de sauvegarder une base de très grande taille.

De plus, nous devrons employer une méthode qui permet d'archiver toute les données de la base pour être réutilisées dans la phase d'import et d'analyse.

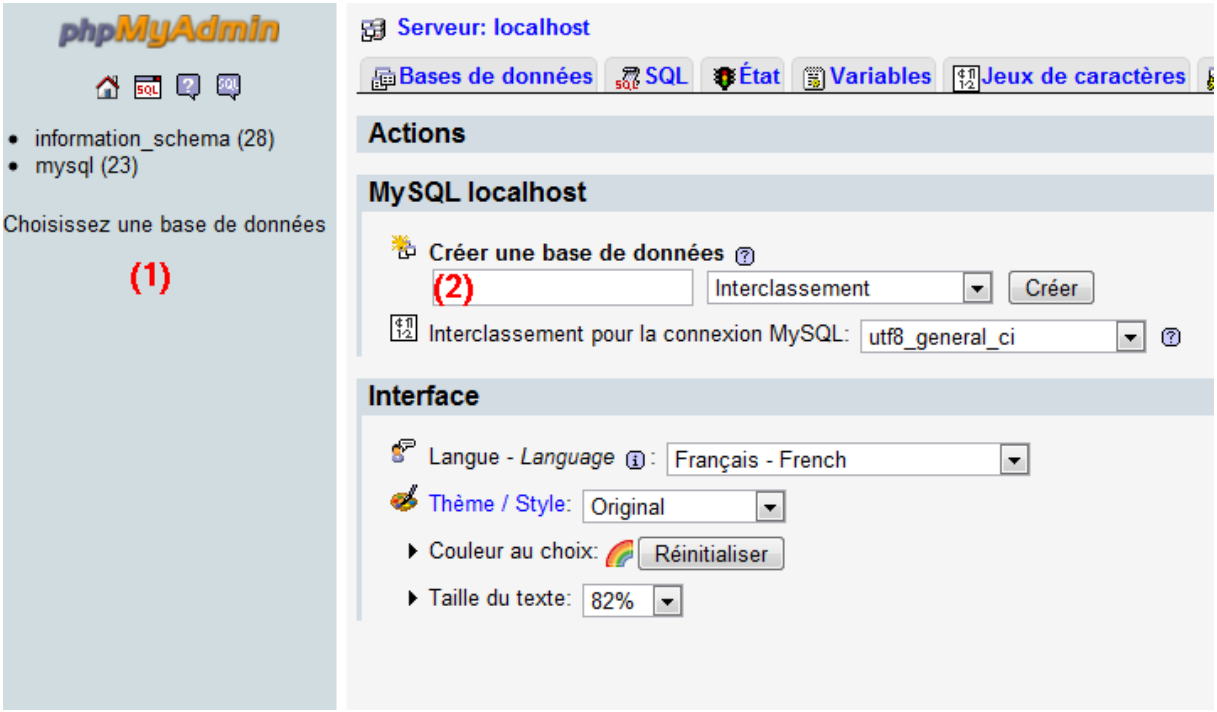

Figure 5 : Interface phpMyAdmin

3. Interface PHP

Dans cette partie nous allons présenter l'interface PHP.

Après la phase d'authentification, l'utilisateur a accès à plusieurs fonctionnalités.

Le menu « tableau », nous offre la possibilité d'exploiter la base de données, et plus précisément la table « alarme », afin d'en extraire des informations statistiques sur les intervals où une attaque a été détectée.

Le menu « code », offre la possibilité à l'utilisateur de récupérer le code source de CUSUM pour l'utiliser ou l'améliorer.

4. 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.

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.

Dans le cadre de notre projet TER, nous voulions détecter les attaques de deni de service de type SYN flooding afin de sécuriser le réseau contre les attaques.

Le développement de l'algorithme CUSUM était le premier pas dans l’avancement de notre projet.

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.

Ce travail nous a beaucoup apporté dans le domaine de la sécurité réseaux.

Cette application que nous avons élaborée présente des avantages comme la détection rapide des anomalies ainsi qu'un taux de fausses alertes limité.

D'autre part, il existe des inconvénients auxquels nous n'avons pas pu remédier et qui pourront faire l'objet d'un futur projet.

VII. Bibliographie

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

[2] 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* »

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

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

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

[6] http://www.siteduzero.com/

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