Master Informatique

Master Mathématiques et Informatique

Spécialité

Informatique

OUTIL DE DETECTION D'ANOMALIES DANS UN RESEAU IP

Elaboré par :

Anis AYACHI

Abdelkarim DHIFALLAH

Encadré par :

Mr. SALEM Osman

Année universitaire : 2008-2009

TABLE DES MATIÈRES

TABLE DES FIGURES

REMERCIEMENT

Ce travail s'inscrit dans le cadre d'un projet de fin d'année en Informatique à l'université PARIS DESCARTES.

A son terme, nous tenons à exprimer notre grande reconnaissance à Mr**. SALEM Osman**, Maître de conférence à l'université PARIS DESCARTES qui nous a suivi de près l'évolution de ce projet et n'a pas cessé de nous prodiguer ses précieux conseils dont nous avions largement bénéficié.

Nous tenons également à remercier Mr. **Nicolas COT** d'avoir accepter de juger ce modeste travail.

Enfin nous remercions tous les enseignants et administrateurs de l'université de PARIS DESCARTES pour nous avoir fourni un enseignement de qualité.

I. INTRODUCTION

L'Internet, parce qu'il tend à devenir le réseau universel de communications caractérisé par une offre multiservice, devient également la cible principale d'attaques et présente une sensibilité accrue à leurs impacts. Il ne se passe plus une semaine sans que l'on apprenne que telle entreprise ou tel institut a subi de lourdes pertes financières en raison d'une déficience de la sécurité de son système d'information. Par conséquent les entreprises ne peuvent plus ignorer ces risques et se croire à l'abri de telles épreuves.

Ces attaques sont fortement pénalisantes puisqu'elles se traduisent par une détérioration de la qualité de service (QoS) dont la gravité est imprévisible. Or, l'Internet est de plus en plus utilisé pour des applications nécessitant une qualité de service (QoS) stable et garantie – par exemple, pour les applications de téléphonie sur IP.

Afin d'appliquer les méthodologies et les notions enseignées jusqu'en Master1 Informatique à l'Université Paris Descartes, nous devons réaliser un Travail d'Etude et de Recherche durant 4 mois. Celui-ci nous permet de nous initier à la recherche, d'appliquer les connaissances acquises durant notre scolarité et de favoriser le travail en groupe

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

II. QUELQUES TYPES D'ATTAQUES RESEAU

L'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. DoS

Le déni de service, en anglophone "Denial of Service", qui est une attaque réalisée dans le but de rendre indisponible durant une certaine période les services ou ressources d'une organisation. Généralement, ce type d'attaque à lieu contre des serveurs, routeurs, machines et accès à une entreprise afin qu'ils deviennent inaccessibles pour leurs clients. Le but de cette attaque n'est pas de voler des informations ou de supprimer des données, il s'agit juste d'empêcher le bon fonctionnement des activités des sociétés présentes sur Internet.

Le principe de cette attaque est d'envoyer des paquets ou des données de très grande taille, afin de provoquer une saturation ou un état instable des équipements victimes (routeurs et serveurs) et de les empêcher ainsi d'assurer les services réseau qu'elles sont sensées offrir. Dans certains cas extrêmes. Le plus souvent, dans ce type d'attaque le cracker lance seul son attaque et cache son identité réseau (adresse IP), pour ne pas être tracé par la victime.

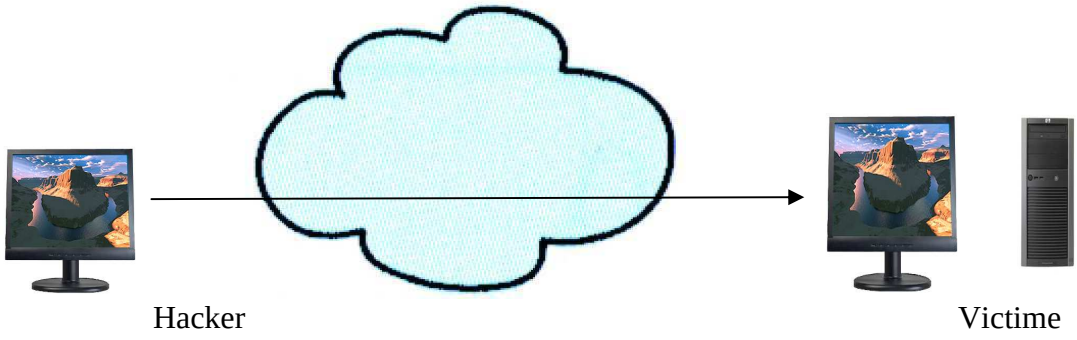

Figure [1] : Attaque DoS

2. DDoS

Le déni de service distribué ou encore “ Distributed Denial of Service “ est un déni de service qui est provoqué par plusieurs machines. Le but recherché de la réalisation de cette attaque sont les mêmes que pour le DoS, avec comme origine d’attaque plusieurs machines à la fois.

Le cracker doit constituer une armé d’attaquants appelés “agents“ ou “zombies“, ces derniers sont contrôlés par un ou plusieurs maîtres appelés aussi “généraux“. Lors de l’attaque, le cracker se connecte au maître, qui envoie un ordre à tous les agents d’attaquer la cible avec une technique choisie par lui-même, lui permettant en outre de cacher son identité à la victime.

Il existe des outils pour organiser et mettre en place l'attaque. En repérant certaines failles courantes sur les machines présentes sur Internet, l'attaquant finit par devenir maître de centaines voir de milliers de machines non protégées. Il faut noter que dans ce type d’attaque les victimes sont aussi les agents en plus de celles qui subissent le déni de services.

Avec le grand nombre des machines non sécurisées sur Internet, on constate que cette attaque est une menasse très grave et dévastatrice puisqu’elle provient d’un réseau tout entier et non pas d’une seule machine.

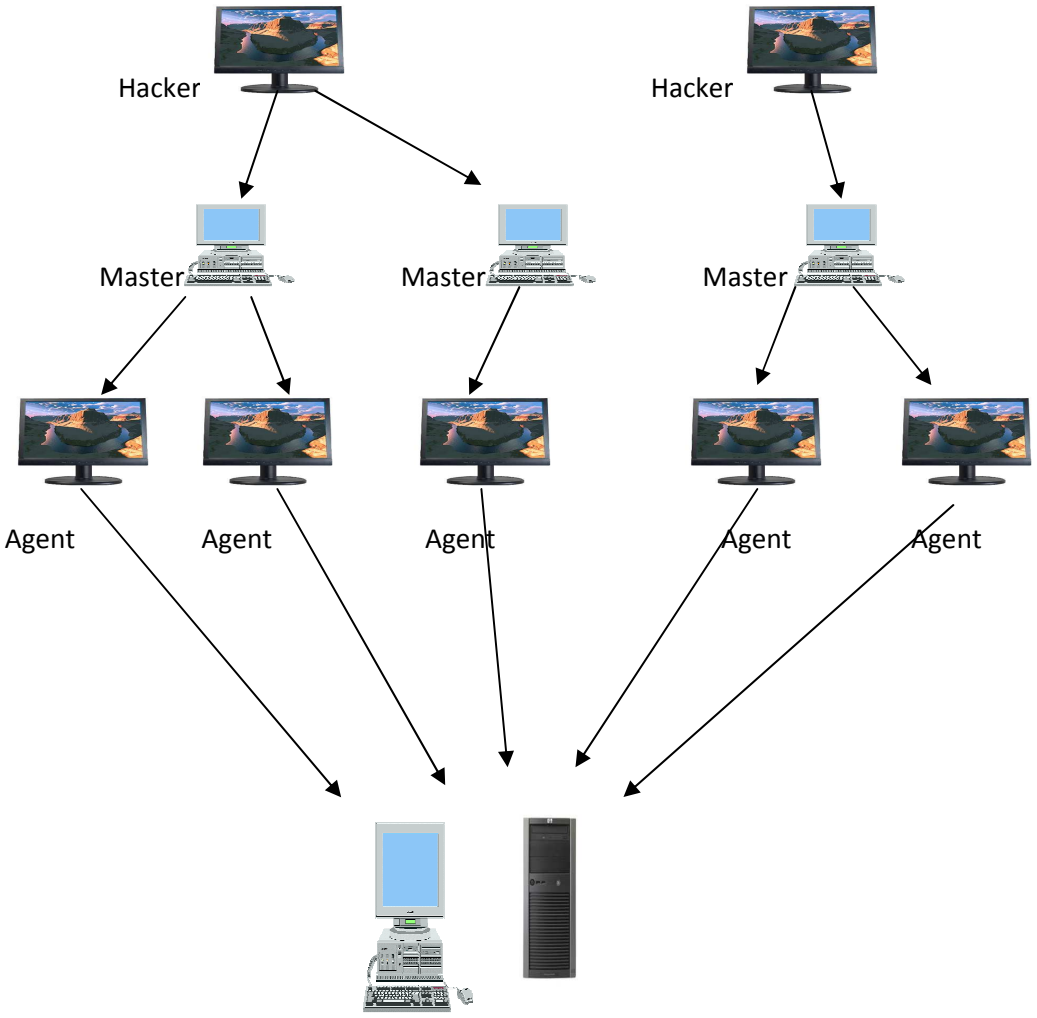

Figure [2] : Attaque DDoS

3. PortScan et NetScan

Le balayage des ports (PortScan) n'est pas vraiment une attaque, en fait c'est une activité qui, en général, procède l'attaque car il s'agit de l'un des principaux moyens pour obtenir des informations sur un ordinateur existant. Il s'agit de balayer les ports TCP/UDP utilisés sur le nœud convoité afin de définir leur état.

Le balayage des ports permet de comprendre quels sont les attaques qui peuvent réussir sur ce système. De plus, les informations obtenues suite au balayage donnent au malfaiteur une idée du système d'exploitation utilisé sur l'ordinateur distant.

Le scan de réseau (NetScan) a pour objectif la découverte de réseau en utilisant une combinaison des tests TCP SYN/ACK, UDP et ICMP, pour but de solliciter une réponse de la cible qui prouvera qu'une adresse IP est effectivement active. Sur des

nombreux réseaux, seul un petit pourcentage des adresses IP sont actives. Il existe plusieurs options pour la découverte des hôtes.

Outils de scan : On trouve énormément des outils de scan, le plus utilisé est :

- Nmap : il a été conçu pour scanner rapidement des grands réseaux, mais il fonctionne bien contre des simples hôtes. Il détermine les hôtes qui sont disponibles sur le réseau, les services qu'ils offrent, le système d'exploitation en cours d'exécution et d'autres caractéristiques.

➔ Nmap est le meilleur scanner disponible aujourd'hui.

4. Principales Attaques

❖ Attaque ICMP Redirect

Cette attaque envois des messages ICMP Redirect à une cible qui peut être un serveur ou un routeur. Le datagramme informera la victime qu'il faut passer par un autre chemin et ceci provoquera une indisponibilité du LAN (Local Area Network).

❖ Attaque Ping Flood (ICMP Flood)

C'est la technique d'attaque par déni de service la plus répondu car elle est utilisé par des particulier et des amateurs qui s'amusent à pinger un host distant. Si ce host Internet ou privé répond à notre Ping, il sera facile de l'inonder des flux afin de le rendre indisponible. L'objectif est de saturer sa bande passante d'accès réseau, mémoire, processeurs…

❖ Attaque SynFlood

Cette technique utilise la faiblesse du protocole TCP, elle consiste à l'envoi massif de demande d'ouverture des sessions SYN. L'objectif est d'arriver au nombre de sessions TCP maximum, lorsque la limite est atteinte, la cible ne pourra plus établir de session TCP causant une indisponibilité de tous ces applications en écoute sur le port TCP.

Pour se protéger, il est donc possible de jouer sur des valeurs (nombre de session maximum établies, le temps d'attente de retour ACK) afin d'accepter plus de sessions et d'être plus réactif sur la durée d'attente. Cependant, le fait de modifier ces valeurs peuvent engendrer un impact de performance local.

❖ Attaque Mail Bombing

Cette attaque, consiste à envoyer plusieurs milliers d'emails à destination d'une entreprise ou d'un utilisateur cible. Le but est le remplissage massive de la boîte à lettre utilisateur, mais surtout de saturer le débit Internet de l'entreprise ciblée. Cette

attaque devient de moins en moins efficace du fait que les entreprises possèdent de plus en plus des tuyaux de boucle local en haut débit, de la QoS (Qualité de service) performante et des relais Anti-Spam pertinents.

III.SYSTEMES DE DETECTIONS D'ANOMALIES

1. Définition

Un IDS (système de détection d'intrusion) est un type de système de sécurité des hôtes et des réseaux. Il est un mécanisme qui écoute le trafic réseau et l'analyse (décrypte) afin de capturer tout type de trafic anormal ou malveillants.

Ce type de système est développe à la réponse de croissance de nombre d'attaque visant les grands sites et réseaux. Aujourd'hui, cet outil est indispensable dans un système de gestion de sécurité d'un réseau ou une hôte qui permet de palier les failles d'un pare-feu.

2. Familles des systèmes de détections

1) NIDS

Il assure la sécurité au niveau du réseau. Ce type de système est placé au point d'entrée du réseau à protéger. Il analyse tout le trafic entrant et sortant de ce réseau et déclenche une alarme en cas de détection d'une activité malicieuse.

2) HIDS

Il assure la sécurité au niveau d'une hôte individuel. Ce type de système est placé sur la machine elle-même; il capte et analyse tout le trafic entrant et sortant qui lui concerne afin de déclencher une alarme en cas de détection d'une activité malicieuse.

3. Les différents comportements de post-détection des IDS

Lors de détection d'une activité malicieuse, l'IDS doit réagir pour annoncer ce qu'il a découvert. Pour cela, il existe différentes méthodes de réaction des IDS, ces réactions sont appelés comportements de post-détection. La réponse du IDS est généralement la génération d'alertes soit par une notification visuelle qui consiste à afficher une alerte dans un ou plusieurs écrans ou par exemple exécution d'un programme externe qui à pour rôle de notifier l'administrateur de réseau qu'il y a une intrusion, soit par sms, email ou autre méthode et dans ce cas, c'est à l'administrateur de réagir.

4. Exemples

- **Snort**

Snort est un IDS de type NIDS, il est gratuitement téléchargeable sur internet.

Cet IDS est le plus répandu dans le marché. Il est capable de faire une analyse de trafic réseau en temps réel et de détecter différents type d'attaque comme scan des ports.

❖ **Tripwire**

Tripwire est un outil qui permet de s'assurer que les fichiers sensibles sur un hôte ne sont pas modifiés. Pour assurer cette fonctionnalité, cet outil possède une base de données contenant la signature numérique des fichiers que l'utilisateur souhaite surveiller.

Lors de la phase de vérification (contrôle) d'intégrité, l'outil recalcule la signature numérique de chaque fichier et la compare avec la valeur dans sa base. En cas de différence, une alerte sera lancée.

IV.ALGORITHMES DE DETECTION D'ANOMALIES

1. CUSUM

Cusum est l'algorithme le plus connu parmi la famille des algorithmes de détection d'anomalie. Son but est de détecter tout changement considéré remarquable ou brusque de modèle dans des processus stochastiques.

L'objectif de la détection de ce changement, est de pouvoir conclure le plus rapidement possible, le passage d'un état de fonctionnement normal A1 à un état de fonctionnement considéré anormal A2.

Description de l'algorithme :

La première étape consiste à segmenter l'intervalle du traitement en intervalles de temps de taille Δ. À chaque intervalle de temps, on calcule la valeur X_n qui présente le nombre des SYN ou la différence nb(SYN) – nb (FIN) dans l'intervalle concerné.

La deuxième étape consiste à faire une fonction simple qui nous permet la détection d'une éventuelle rupture. Cette fonction est récursive et est appliquée dans chaque intervalle Δ_i.Si le résultat de cette fonction dépasse un seuil bien déterminé, une alarme est déclenchée.
Cette fonction récursive est sous la forme suivante:

$$G_n = [G_{n-1} + X_n - M]^+ ;$$
$$G_0 = 0;$$

où X_n présente le nombre SYN ou SYN-FIN, et M présente la valeur moyenne des intervalles précédents.
La notation (+) indique que si

$G_{n-1} + X_n - M \succ 0$ Alors $G_n = G_{n-1} + X_n - M$

Sinon, $G_n = 0$

2. NADA

Network Anomaly Detection Algorithm (NADA) a été développé par :

- **Silvia Farraposo :** School of Technology and Management- Polytechnic Institute of Leiria Alto- Vieiro, Morro do Lena (Portugal)
- **Philippe Owezarski :** LAAS- CNRS / Université de Toulouse (France)
- **Edmundo Monteiro :** DEI/CISUC- University of Coimbra Polo II (Portugal)

NADA peut travailler sur n'importe quelle série temporelle (le nombre de paquets, le nombre d'octets ou le nombre de nouveaux flots par unité de temps), son objectif potentiel est la détection des anomalies c'est à dire déterminer si une anomalie est en train de se produire y compris les plus réduites.

Cet algorithme considère qu'une anomalie est responsable d'une variation sur au moins un des critères considérés, à au moins une échelle de temps et à un niveau d'agrégation donnés. Les échelles de temps Δ choisies pour analyser le trafic vont de quelques microsecondes jusqu'à plusieurs heures. Pour cela, il compare les variations du trafic entre l'instant Δ_{i+1} et l'instant Δ_i.

Cet algorithme présente deux phases :

- Phase d'apprentissage : elle peut s'étendre sur un ou plusieurs intervalles de temps, c'est une phase de pré-analyse.
- Phase d'analyse et de détection : dans la quelle on applique l'équation ci-dessous

$$\mathbf{X=\{x_1, x_2, \ldots, x_n\}} \quad \mathbf{x_i = \{\#packet/\ \#bytes/\ \#flows\ \}/\ \Delta}$$

$$\mathbf{P= \{p_1, p_2, \ldots, p_{n-1}\}} \quad \boldsymbol{Pi= x_{i+1} - x_i}$$

$$\begin{cases} \boldsymbol{Pi \geq E(p) + k\sigma \rightarrow anomalie} \\ \boldsymbol{Pi < E(p) + k\sigma \rightarrow normal} \end{cases}$$

Avec $E(p)$ la moyenne, σ l'écart-type et K une valeur positive, on a constaté que des grandes valeurs de k conduisent à un grand nombre des faux négatifs, alors que des petites valeurs conduisent à un grand nombre des faux positifs. $K=2$ est la meilleure valeur pour configurer NADA. Ce dernier détecte une anomalie si la valeur de série temporelle dans l'intervalle en cour est supérieur ou égale à un seuil (seuil= moyenne de tous les intervalles précédents + 2*écart-type).

V.NOTRE SYSTEME DE DETECTION D'ANOMALIES

1. Description

Notre système de détection d'anomalie proposé est de type NIDS. Pour le cycle de développement et de test, on a utilisé une trace de taille 3Go. Cette trace contient différent type d'attaque.

2. Algorithme

Dans ce projet, on a décidé d'utiliser l'algorithme NADA qui est pour l'instant le seul à permettre de simultanément détecter, classifier et identifier les anomalies du trafic. NADA travaille sur trois séries temporelles différentes : le nombre des octets (bytes), le nombre des flots (flows) et le nombre des paquets (packets), et par conséquences la détections des trois types d'attaques différentes :

- Attaque de type Bytes (octets)
- Attaque de type Flows (flots)
- Attaque de type Packets (paquets)

3. Architecture

Notre Système est constitué de quatre modules :

- Module d'enregistrement.
- Module de détection d'intrusion.
- Module d'interrogation.
- Module de réponse.

Module d'enregistrement: ce module permet de lire les informations à partir de la trace et l'enregistrer dans la base des données afin de simplifier la manipulation de ces donner dans la phase d'interrogation.

Module de détection d'intrusion: ce module travaille sur les données de la trace en appliquant l'algorithme mentionné précédemment afin de détecter l'intrusion.

Module de réponse: ce module permet l'affichage d'une alerte suite d'une détection d'intrusion mentionnée par le module de détection d'intrusion.

Module d'interrogation: ce module permet à l'utilisateur de consulter et de vérifier les données présentes dans sa base de données. Par exemple suite à une attaque de type DoS, l'utilisateur peut consulter tout les informations sur cette attaque en entrant seulement les informations mentionnées par le module de réponse.

4. Environnement de Développement

Après une période de recherche et d'étude des différents systèmes existants sur le marché, et après le choix de l'algorithme de détection d'intrusion à utiliser, nous avons attaqué la partie la plus délicate : c'est le choix de l'environnement de développement (c'est-à-dire les différents technologies et langages à utiliser).

Notre choix été, le langage C pour le traitement de la trace, MySQL pour le stockage d'informations qui proviennent du programme C, les technologies HTML,CSS, Javascript et PHP pour la réalisation de l'interface graphique pour l'interaction de l'utilisateur avec les système.

Le langage C :

Le langage C étant un langage simple, il permet d'écrire des logiciels qui n'ont besoins d'aucun support d'exécution (ni bibliothèque logicielle, ni machine virtuelle), au comportement prédictible en temps d'exécution comme en consommation de mémoire vive.
Ce langage est extrêmement utilisé dans des domaines comme les calculs intensifs, l'écriture de systèmes d'exploitation et tous les modules où la rapidité de traitement est importante.

HTML :

Hypertexte Markup Language, est le format de données conçu pour représenter les pages web. HTML permet aussi de structurer sémantiquement et de mettre en forme le contenu des pages, d'inclure des ressources multimédias dont des images, des vidéos, des formulaires de saisie

Javascript :

Sert à contrôler les données saisies dans les formulaires et interagir avec le document HTML.

CSS :

(Cascading Style Sheet) sert à décrire la présentation des documents HTLM.L'objectif majeur de cette technologie est de permettre la stylisation hors des documents.
Parmi les avantages majeurs, on peut citer :

- La structure de document et la présentation peuvent être générées dans des fichiers séparés.
- Le code HTML est considérablement réduit en taille et en complexité, puisqu'il ne contient plus de balises ni d'attributs de présentation.

PHP :

(Hypertext preprocessor) est un langage de script, spécialement conçu pour le développement d'applications web. Il peut intégrée facilement du code HTML. Ce qui distingue PHP des langages de script comme le Javascript, est que le code est exécuté coté serveur.

Le choix de ce langage été pour ces raisons :

- La simplicité d'écriture de script.
- La possibilité d'inclure du code HTML au sein de script PHP.
- La simplicité d'interfaçage avec la base de données MySQL qui est utilisée dans notre outil.

MySQL :

MySQL est le serveur de base de données le plus utilisé dans le monde. Son architecture logicielle le rend extrêmement rapide et facile à personnaliser. Les principaux avantages de MySQL sont sa rapidité, sa robustesse et sa facilité d'utilisation et d'administration. Un autre avantage majeur de MySQL est sa documentation très complète et bien construite.

5. Fonctionnement

Dans cette section on va détailler les étapes de notre travail et le fonctionnement de notre interface graphique.

- ✓ **Module d'enregistrement :**

Dans ce module on a crée une base de donnée (Database), nommée « anomalie » dans laquelle on a crée deux tableaux (Table) :

- Table « alarme » : composée de trois champs :
  a. intervall : l'intervalle dans lequel l'attaque a été faite.
  b. ip_dest : l'adresse IP attaquée.
  c. type : le type de l'attaque (Bytes, Flows ou Packets).
- Table « trace » : composée de 14 champs :
  a. dst_ip : adresse IP source.
  b. src_ip : adresse IP destination.
  c. src_port : port source.
  d. dst_port : port destination.
  e. proto : type de protocole.
  f. in_out.
  g. start_time : debut de l'attaque.
  h. end_time : fin de l'attaque.
  i. nb_packets : nombre de paquets.
  j. nb_syn : nombre de SYN.
  k. nb_synack : nombre de SynAck.

l. nb_fin : nombre de FIN.

m. nb_rst : nombre de RST.

n. nb_bytes : nombre de Bytes.

Cette dernière nous fournis des informations sur le trafic dans chaque intervalle de temps.

On a ajouté les quelques lignes de code C ci-dessous, qui nous permettent de se connecter à notre base de donnée.

```
MYSQL *con;
con = mysql_real_connect(conn, localhost, "dhifallah",\
 "eclipse", "anomalie", 0, NULL,0);
if (con==NULL) {
    printf ("\aErreur de connexion du client !\n");
    exit(EXIT_FAILURE);
 }

 else
      printf ("Connexion établi !\nWelcome !\n");
```

mysql_real_connect() : méthode prédéfinie dans l'API MYSQL qui prend en paramètres :

- L'adresse du serveur où MySQL est installé.
- Nom du user: dhifallah
- Mot de passe: eclipse
- Base de données : anomalie

On a également ajouté des lignes de code C pour l'insertion des données dans nos deux tableaux :

« Table trace »

```
buf=(char*)malloc (500);

sprintf(buf, "INSERT INTO trace (ip_dest, ip_src, port_src, port_dest,\
 protocol, nb_syn, nb_ack,nb_packet,nb_byte,nb_fin, nb_rst, in_out, \
 start_time, end_time) VALUES ('%u.%u.%u.%u', '%u.%u.%u.%u', '%hu', '%hu',\
 '%u', '%u', '%u', '%u', '%u', '%u', '%u', '%hu', '%u', '%u')\n", \
 (rec->daddr>>24)&0xFFU, (rec->daddr>>16)&0xFFU,(rec->daddr>>8)&0xFFU,\
 (rec->daddr)&0xFFU,(rec->saddr>>24)&0xFFU, (rec->saddr>>16)&0xFFU,\
 (rec->saddr>>8)&0xFFU,(rec->saddr)&0xFFU, rec->dst_port, rec->src_port,\
 rec->proto, rec->nb_syn, rec->nb_synack, rec->nb_packets,rec->nb_bytes,\
 rec->nb_fin,rec->nb_rst,rec->in_out,rec->start_time,rec->end_time);

 mysql_query(&conn, buf);
 free(buf);
```

« Table alarme»

```
switch (t) {
 case 0:
      printf("L'adresse IP de la machine attaquÃ©e bytes \
      IP= %u.%u.%u.%u de niveau X\n",(IPdest>>24)&0xFFU, \
      (IPdest>>16)&0xFFU,(IPdest>>8)&0xFFU,(IPdest)&0xFFU);
      sprintf(buf,"INSERT INTO alarme(intervall,ip_dest,type)\
      VALUES ('%llu','%u.%u.%u.%u','BYTES')",interval_cnt,\
      (IPdest>>24)&0xFFU,(IPdest>>16)&0xFFU,(IPdest>>8)&0xFFU,(IPdest)&0xFFU);
 break;
 case 1:
      printf("L'adresse IP de la machine attaquÃ©e flows \
      IP= %u.%u.%u.%u de niveau Y\n",(IPdest>>24)&0xFFU, \
      (IPdest>>16)&0xFFU,(IPdest>>8)&0xFFU,(IPdest)&0xFFU);
      sprintf(buf,"INSERT INTO alarme(intervall,ip_dest,type) \
      VALUES ('%llu','%u.%u.%u.%u','FLOWS')",interval_cnt,\
      (IPdest>>24)&0xFFU,(IPdest>>16)&0xFFU,(IPdest>>8)&0xFFU,(IPdest)&0xFFU);
 break;
 case 2:
      printf("L'adresse IP de la machine attaquÃ©e packets \
      IP= %u.%u.%u.%u de niveau Z\n",(IPdest>>24)&0xFFU, \
      (IPdest>>16)&0xFFU,(IPdest>>8)&0xFFU,(IPdest)&0xFFU);
      sprintf(buf,"INSERT INTO alarme(intervall,ip_dest,type)\
      VALUES ('%llu','%u.%u.%u.%u','PACKETS')",interval_cnt,\
      (IPdest>>24)&0xFFU,(IPdest>>16)&0xFFU,(IPdest>>8)&0xFFU,(IPdest)&0xFFU);
 break;

                  }
mysql_query(conn,buf);
```

✓ **Module de détection d'intrusion :**

L'exécution de notre code principal, détermine les anomalies (les adresses IP des machines attaquées et le type d'attaque)

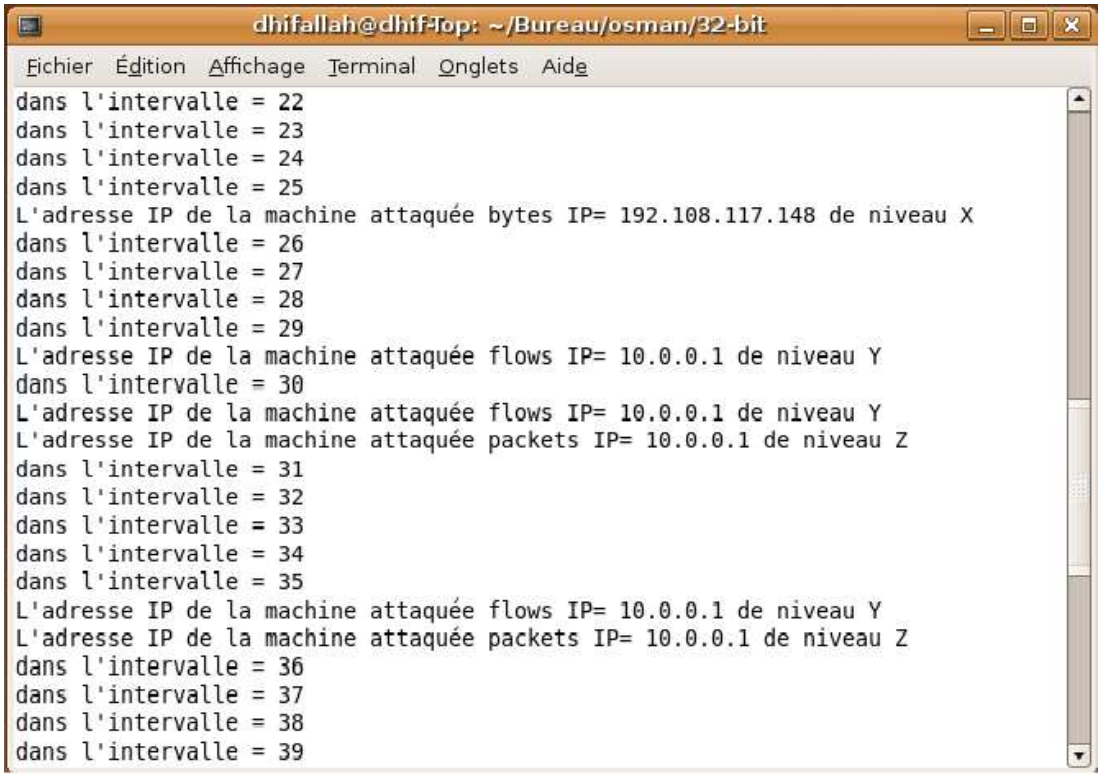

Figure [3]: détection des anomalies

Et par la suite le remplissage du tableau « alarme »

id	intervall	ip_dest	type
1	8	79.6.216.227	BYTES
2	13	77.197.79.167	BYTES
3	26	192.108.117.148	BYTES
4	30	10.0.0.1	FLOWS
5	31	10.0.0.1	FLOWS
6	31	10.0.0.1	PACKET
7	36	10.0.0.1	FLOWS
8	36	10.0.0.1	PACKET
9	60	10.0.0.1	FLOWS
10	60	10.0.0.1	PACKET
11	61	10.0.0.1	FLOWS
12	61	10.0.0.1	PACKET
13	69	192.108.117.148	BYTES
14	90	10.0.0.1	FLOWS
15	90	10.0.0.1	PACKET
16	112	10.0.0.1	FLOWS
17	112	10.0.0.1	PACKET
18	120	10.0.0.1	FLOWS
19	120	10.0.0.1	PACKET

19 rows in set (0.01 sec)

Figure [4] : table « alarme »

D’après ce tableau, on constate 19 attaques qui ont été faite sur quatre adresse IP différente, l’adresse IP 10.0.0.1 a été attaquée a plusieurs reprises avec deux types d’attaques différents (Flows et Packets).

Analyse des traces :

Ces traces sont obtenues en utilisant le programme « **Gnuplot** »

Gnuplot : un programme souple qui peut produire des représentations graphiques en deux ou trois dimensions des fonctions numériques. Le programme fonctionne sur tous les ordinateurs et systèmes d'exploitation principaux et peut envoyer les graphiques à l’écran ou dans des fichiers dans de nombreux formats.

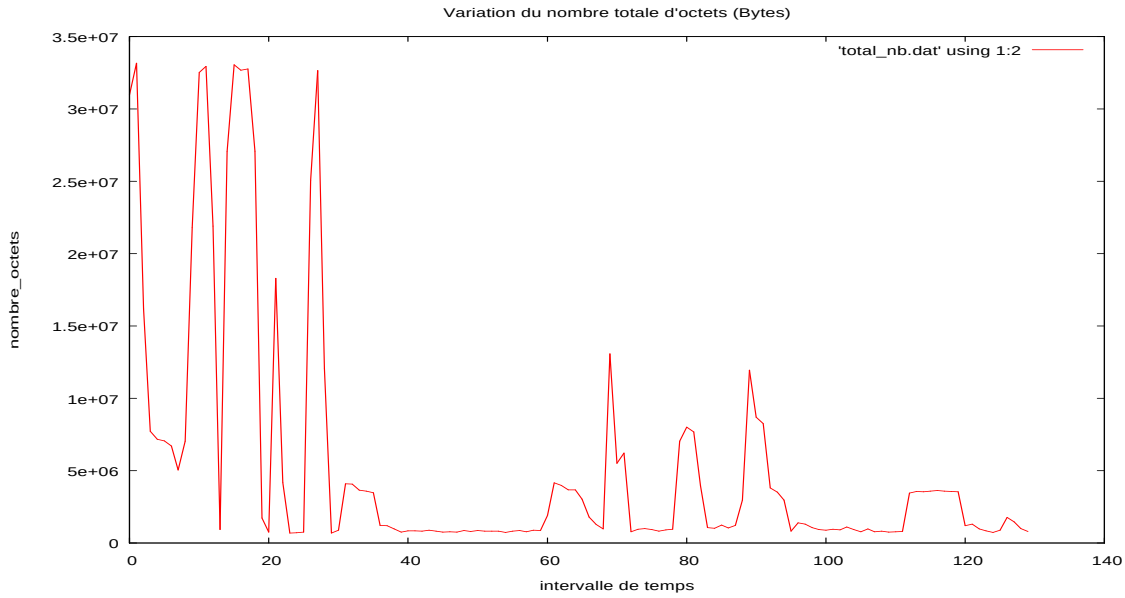

Figure [5] : variation de nombre totale d’octets

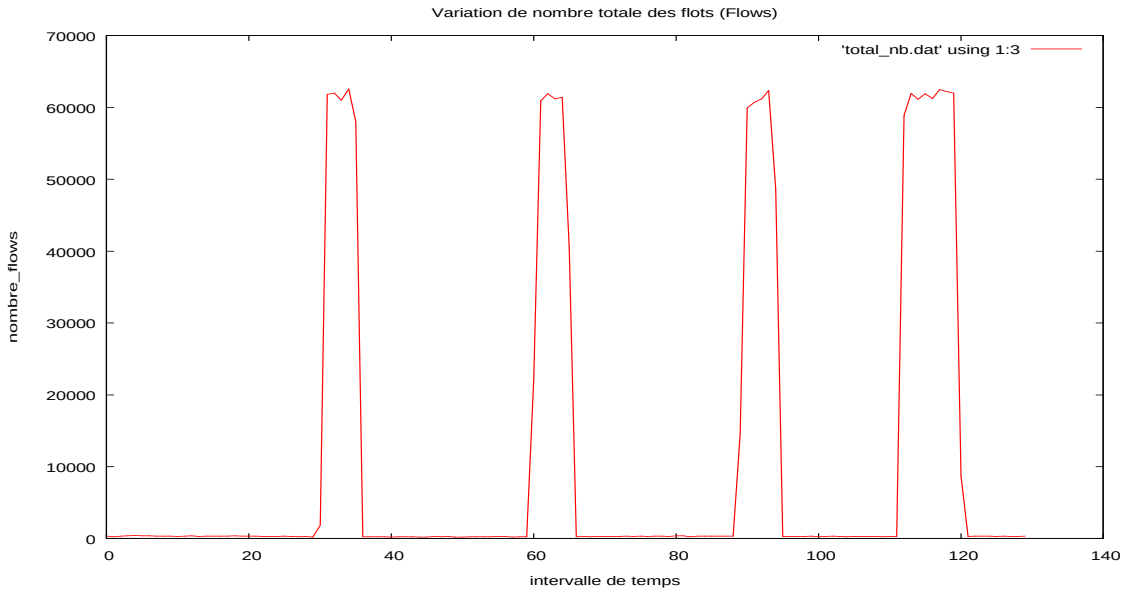

Figure [6] : variation de nombre totale des flots

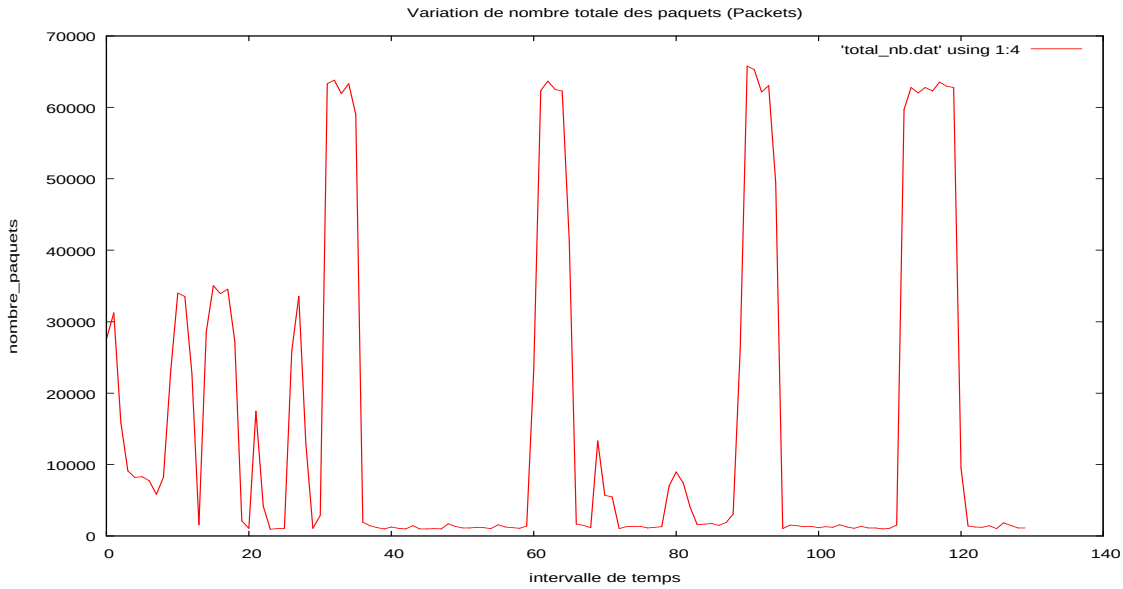

Figure [7] : variation de nombre totale des paquets

Ces trois courbes représentent dans l'ordre :

- Nombre totale d'octets (Bytes) dans notre trace : le nombre vari au cours du temps et il atteint un maximum de $3.3*10^7$ octets.

- Nombre totale de flots (Flows) dans notre trace : le nombre est différent d'un intervalle à un autre et il atteint un maximum de ~62000 flots.
- Nombre totale de paquets (Packets) dans notre trace : ce nombre vari au cours du temps et présente un maximum de 65000 paquets.

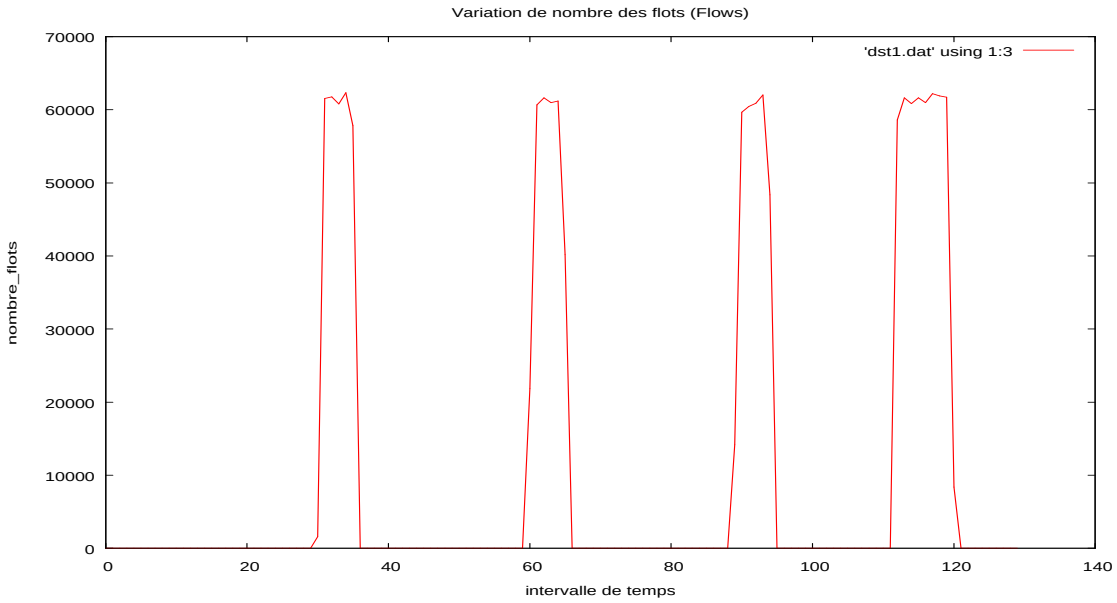

Figure [8] : variation de nombre des flots de la machine attaquée

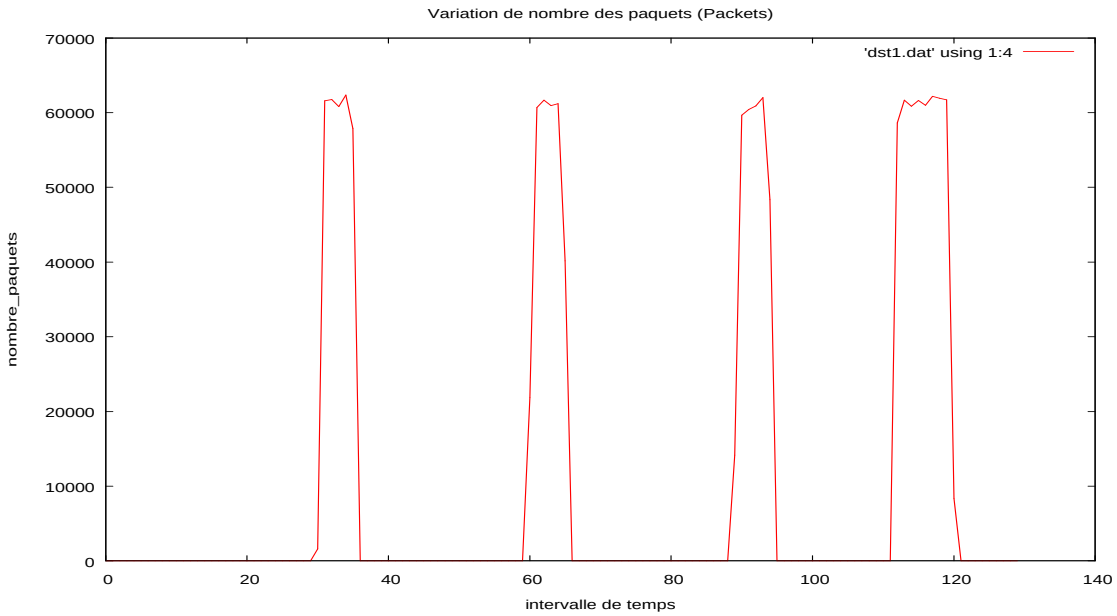

Figure [9] : variation de nombre des paquets de la machine attaquée

Ces deux courbes représentent respectivement le nombre de flots (Flows) et le nombre de paquets (Packets) de l'adresse IP attaquée 10.0.0.1. On remarque une variation du nombre de flots brutale dans les intervalles 30, 31, 36, 60, 112 et 120, et de même pour le nombre de paquets dans les intervalles 31, 36, 60, 112 et 120. Ces variations conduisent à la détection des anomalies dans chacun de ces intervalles comme il a été déjà mentionné dans le tableau « alarme ».

✓ **Module d'interrogation :**

Notre outil de détection d'intrusion, fourni à son utilisateur un ensemble d'interfaces qui lui permettent d'interroger le système.

Dans cette partie du rapport, on décrit les différentes interfaces du système ainsi que leurs utilités et le principe de leurs fonctionnements.

Interface Home

Cette interface constitue la page d'accueil de notre système d'interaction. Dans cette interface, on trouve des informations générales concernant le trafic (notre trace).
Parmi ces informations, on peut trouver :

- Le nombre total d'alerte détecté.
- Le nombre d'adresse IP source distincts.
- Le nombre d'adresses IP destination distincts.

- Le nombre de ports sources distincts.
- Le nombre de ports destinations distincts.
- Le nombre de protocoles existants dans la trace.

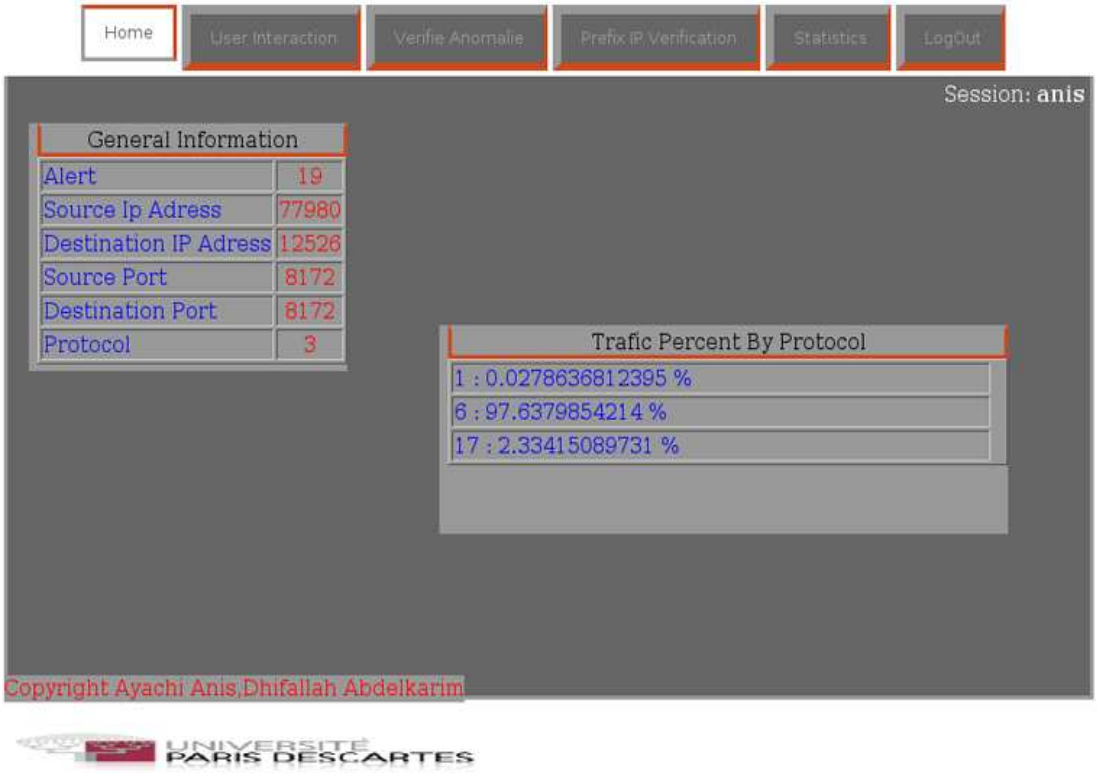

Figure [10]: interface home

La recherche des données présentées dans l'interface, se fait à travers des requêtes SQL appliqués sur la table « trace » de notre base.

L'exécution de ces requêtes se fait après l'opération de connexion à la base de données. Le code de connexion à la base de données est le suivant :

$connect=mysql_connect('localhost','root','') or die("erreur de connexion à la base");
mysql_select_db('anomalie',$connect);

La requête suivante présente la requête SQL qui a pour rôle de trouver le nombre total distinct d'adresse IP source :
select count (distinct (ip_src)) from trace;

Interface Login

Cette interface permet à l'utilisateur de se connecter au système afin de pouvoir l'interroger.

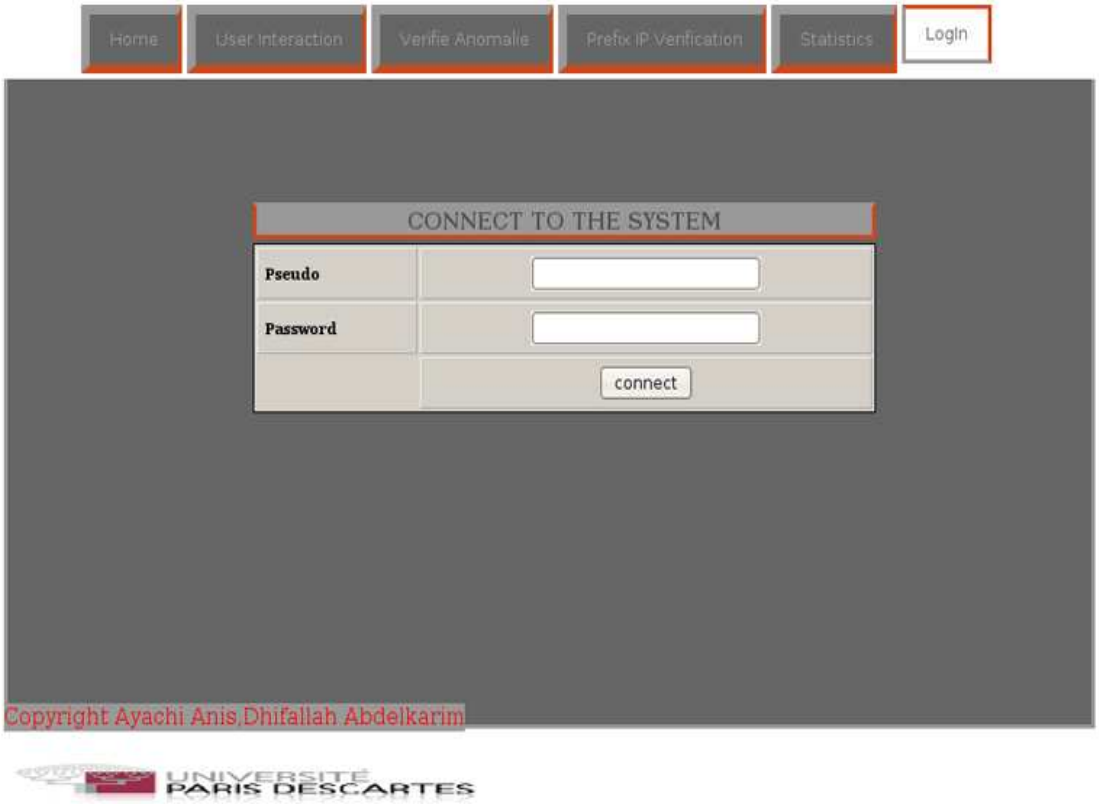

Figure [11] : connexion au système

Pour utiliser le système, l'utilisateur doit se connecter en entrant son login et mot de passe.

Après la vérification du login et du mot de passe, une session utilisateur est créé qui a comme attributs login et rôle de l'utilisateur. Cet attribut de nom rôle a pour rôle de permettre ou restreindre l'accès à l'interface« User Interaction », puisque seulement l'utilisateur qui a comme rôle administrateur a le droit d'accéder à cette interface.

Pour permettre l'utilisation de session, la page « connect.php » doit commencer par *session_start()*, cette fonction met en évidence l'utilisation de la session.

La requête SQL qui permet de vérifier l'existence de l'utilisateur dans la table « user » et de trouver son rôle est :

select role from users where pseudo='$pseudo' and password='$password';

L'association des valeurs pseudo et rôle aux attributs pseudo, rôle de la session se fait comme suit :

```
$_SESSION['pseudo']=$pseudo ;
$_SESSION['role']= $row[0];
```

Avec:
$result=mysql_query("select role from users where pseudo='$pseudo' and password='$password'");
$row=mysql_fetch_array($result);

Interface User Interaction

Cette interface permet seulement aux utilisateurs dont le rôle est administrateur d'y accéder. Il permet aux administrateurs d'ajouter, supprimer, modifier des comptes d'autres utilisateur.

Pour supprimer un utilisateur, il suffit de fournir son pseudo.

Figure [12] : suppression d'un utilisateur

La requête SQL qui permet de faire la suppression est :

delete from users where pseudo='$pseudo';
où *$pseudo* est le pseudo de l'utilisateur à supprimer.

Pour ajouter un nouvel utilisateur, un administrateur fournit le pseudo et le mot de passe et le rôle de cet utilisateur.

Figure [13]: ajout d'un nouvel utilisateur

La requête SQL qui permet de faire l'ajout est :

insert into users values ('$pseudo','$password','$role');
où *$pseudo* est le pseudo de l'utilisateur, *$password* est le mot de passé de l'utilisateur et *$role* est le rôle de l'utilisateur qui peut être "simple user" ou "administrator".

Interface Vérifie Anomalie

Cette interface permet aux utilisateurs de vérifier quelques types d'anomalie comme « Net scan, Dos Attack »

Pour vérifier une anomalie, l'utilisateur peut choisir le type d'anomalie à vérifier (ici nous proposons quatre types d'anomalie qui sont Net Scan, Port Scan, DOS, DDOS). Selon ce type, l'utilisateur fournit les informations nécessaires pour interroger le système.

Pour la vérification d'un « Net Scan », l'utilisateur donne l'adresse IP source et le numéro du port destination. La réponse du système est la liste des adresses

IP destination dont ils ont eux une communication avec l'adresse source et le port destination données par l'utilisateur.

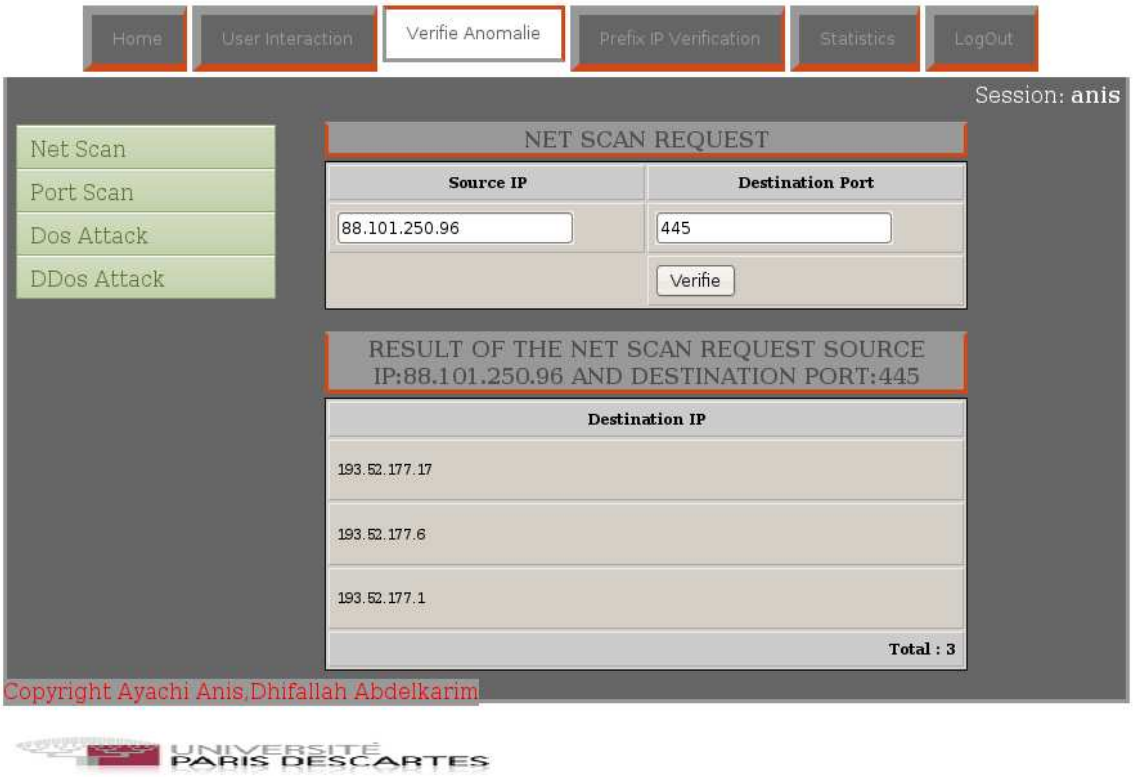

Figure [14] : vérification d'un NetScan

La requête SQL qui permet de vérifier le « Net Scan » est :

select distinct(ip_dest) from trace where ip_src='$src_ip' and port_dest='$port_dest' and protocol='6' and nb_syn!=0;

Pour la vérification d'un « Port Scan », l'utilisateur donne l'adresse IP source et destination concerné. La réponse du système est la liste des ports destinations pour cette combinaison.

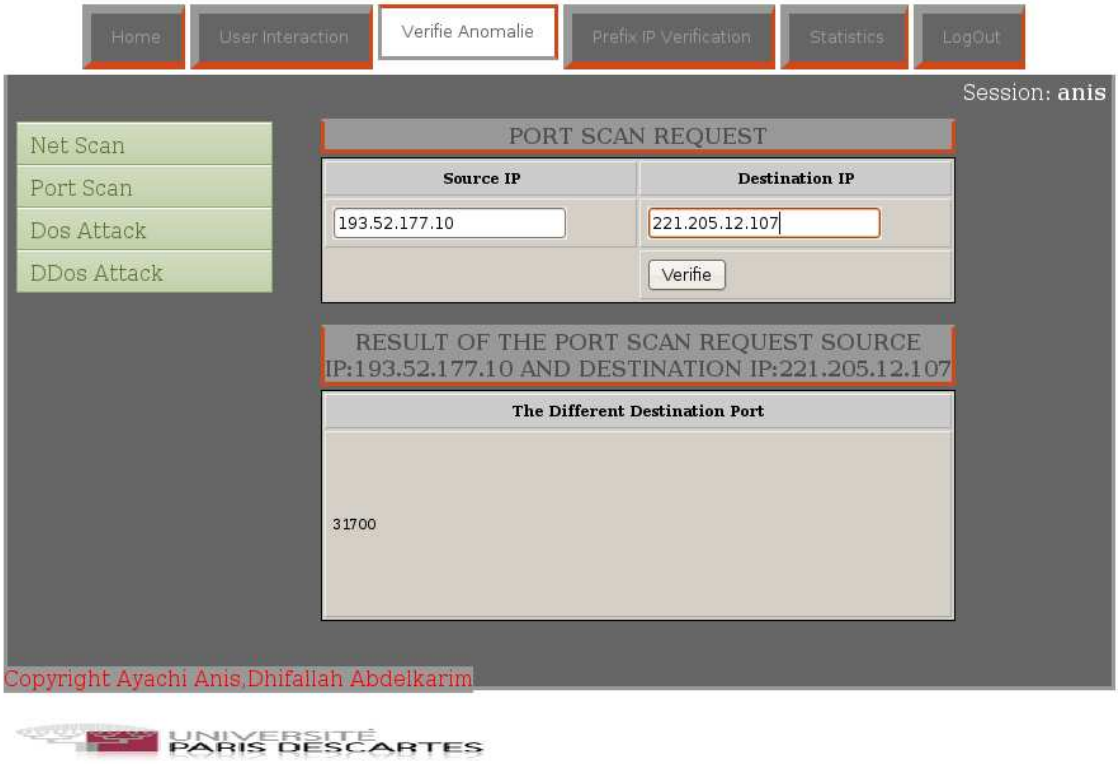

Figure [15]: vérification d'un PortScan

La requête SQL qui permet de vérifier le « Port Scan » est :

select distinct (port_dest) from trace where ip_src='$ip_src' and ip_dest='$ip_dest' and protocol='6' and nb_syn!=0;

Pour la vérification d'une attaque de type « DoS », l'utilisateur fournit l'adresse IP source et destination et le port destination. La réponse du système est le nombre total des paquets de type SYN reçu sur le port destination de la machine qui a cet adresse IP destination et provenant de l'adresse IP source qui sont mentionnés par l'utilisateur.

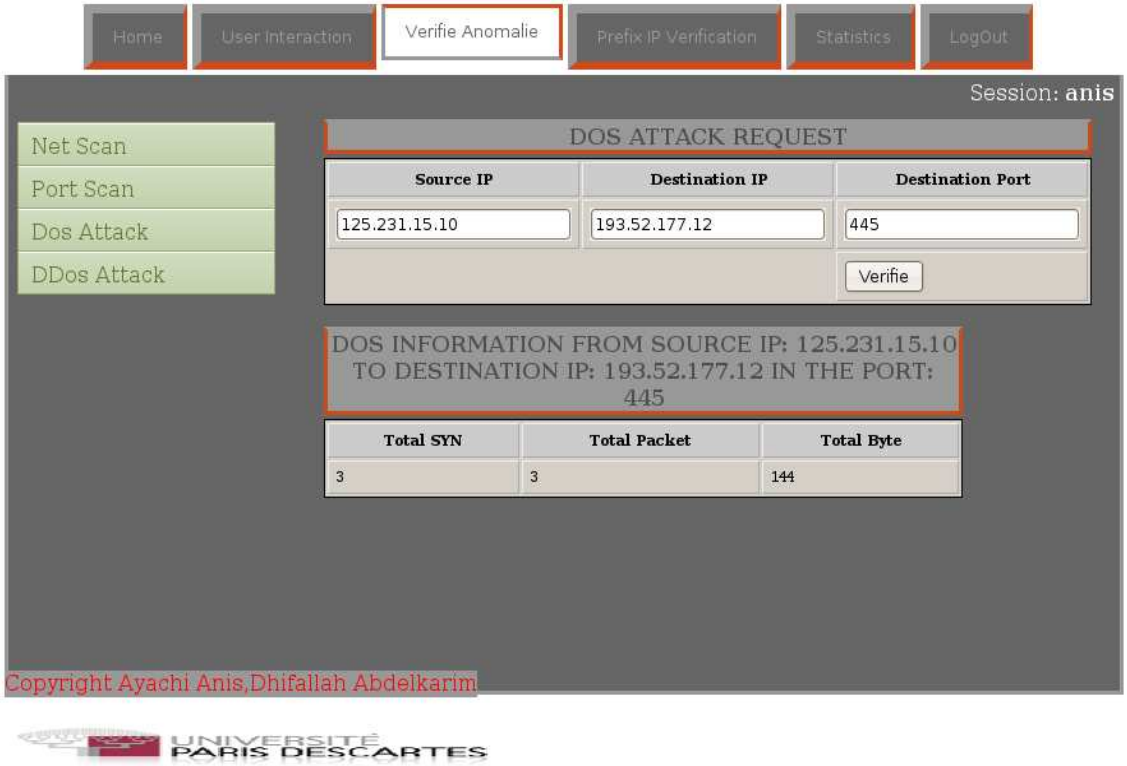

Figure [16] : vérification d'une attaque DoS

La requête SQL qui permet de vérifier l'attaque « DoS » est :

Select sum(nb_syn),sum(nb_packet),sum(nb_byte) from trace where ip_src='$ip_src' and ip_dest='$ip_dest'and port_dest='$port_dest';

Pour la vérification d'une attaque de type « DDoS », l'utilisateur fournit l'adresse IP destination et le port destination concerné.

La réponse du système est les adresses IP sources avec le nombre des paquets de type SYN communiquées avec l'adresse IP destination et le port destination données par l'utilisateur.

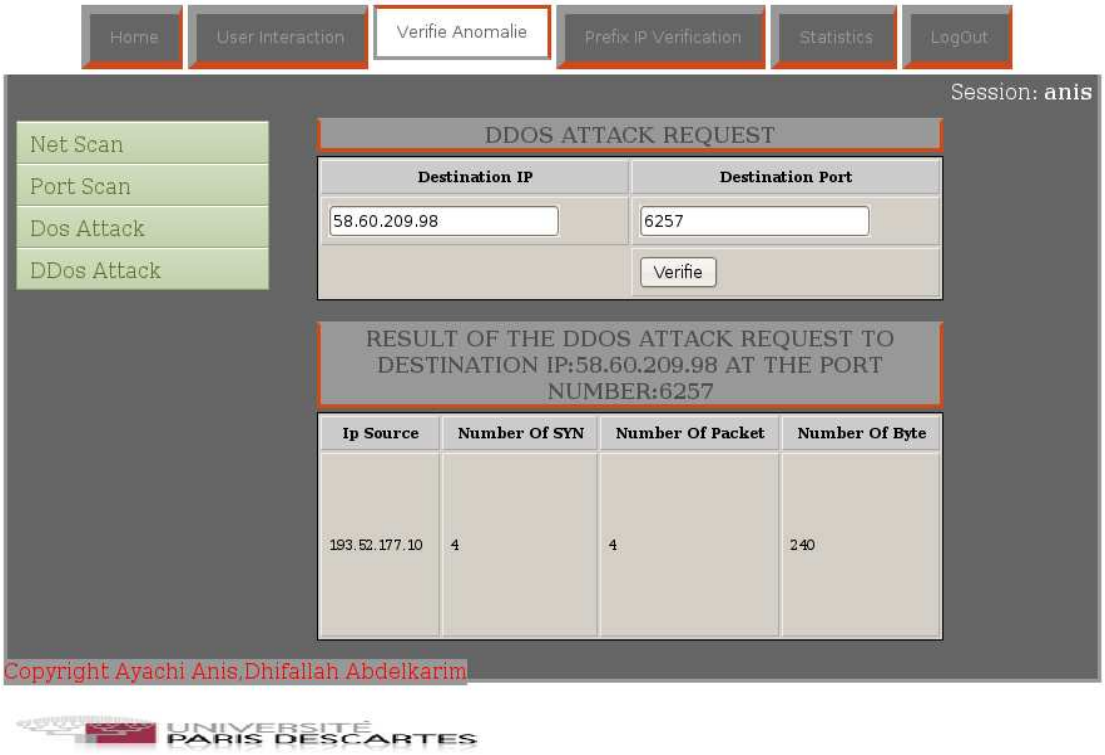

Figure [17] : vérification d'une attaque DDoS

La requête SQL qui permet de vérifier l'attaque « DDoS » est :

select ip_src, sum (nb_syn) from trace where ip_dest='$ip_dest' and port_dest='$port_dest' group by ip_src;

L'affichage des informations se fait par ordre décroissant de somme des paquets de type SYN.

Interface Prefix IP Verification

Cet interface permet à l'utilisateur de voir les machines les plus actifs dans un « réseau » mentionné comme préfix selon un critère bien déterminer comme (nombre_SYN, nombre_packet), cette interrogation se fait à travers d'un préfix entré par l'utilisateur pour filtrer les réseaux voulu.

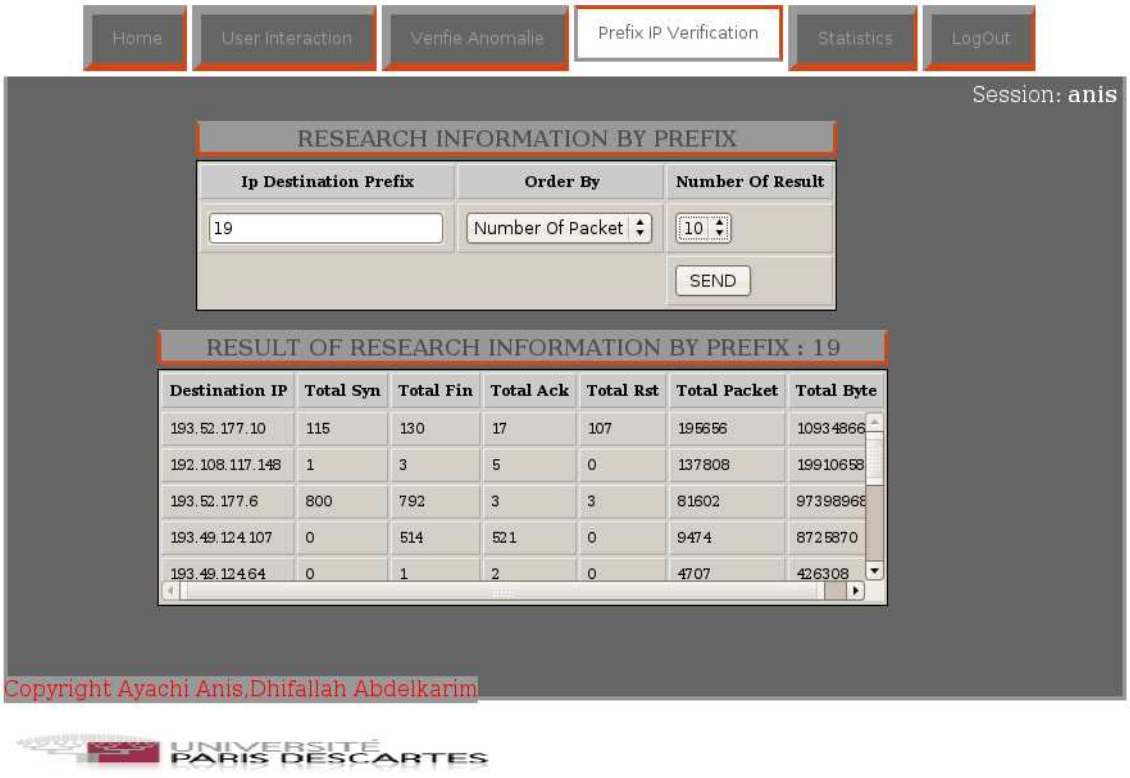

Figure [18] : recherche d'informations en utilisant le préfix d'adresse IP

Pour lancer une requête en utilisant un préfix, l'utilisateur fournit comme entrée un préfix de réseau, et choisit le critère dont les adresses IP destination seront affiché par ordre décroissant.
Pour faire ce type de traitement, on a besoin de deux requêtes SQL et quelques traitements supplémentaires en utilisant un tableau associatif et un tableau multidimensionnel. Les deux requêtes SQL sont:

select ip_dest,sum($order) from trace where ip_dest like '$prefix%' group by ip_dest;

select sum(nb_syn),sum(nb_fin),sum(nb_ack),sum(nb_rst),sum(nb_packet), sum(nb_byte) from trace where ip_dest='$cle' group by ip_dest;

Interface Statistics

Cette interface permet à l'utilisateur de chercher :

- Les adresses IP destination les plus actives selon un critère bien déterminé comme (nombre paquet, nombre SYN).
- Les adresses IP source les plus actives selon un critère bien déterminé comme (nombre paquet, nombre SYN).
- Les ports destination les plus actives selon un critère bien déterminé comme (nombre paquet, nombre SYN).
-

Pour rechercher les adresses IP destination les plus active selon un critère bien déterminer qui est le nombre de (paquet, syn, fin, rst, byte, ack), l'utilisateur, à travers de son interface, mentionne le critère voulu.

La réponse de système sera l'affichage des adresses IP destination avec le nombre total de critère demandé par l'utilisateur.

L'affichage se fait par ordre décroissant selon la valeur de critère.

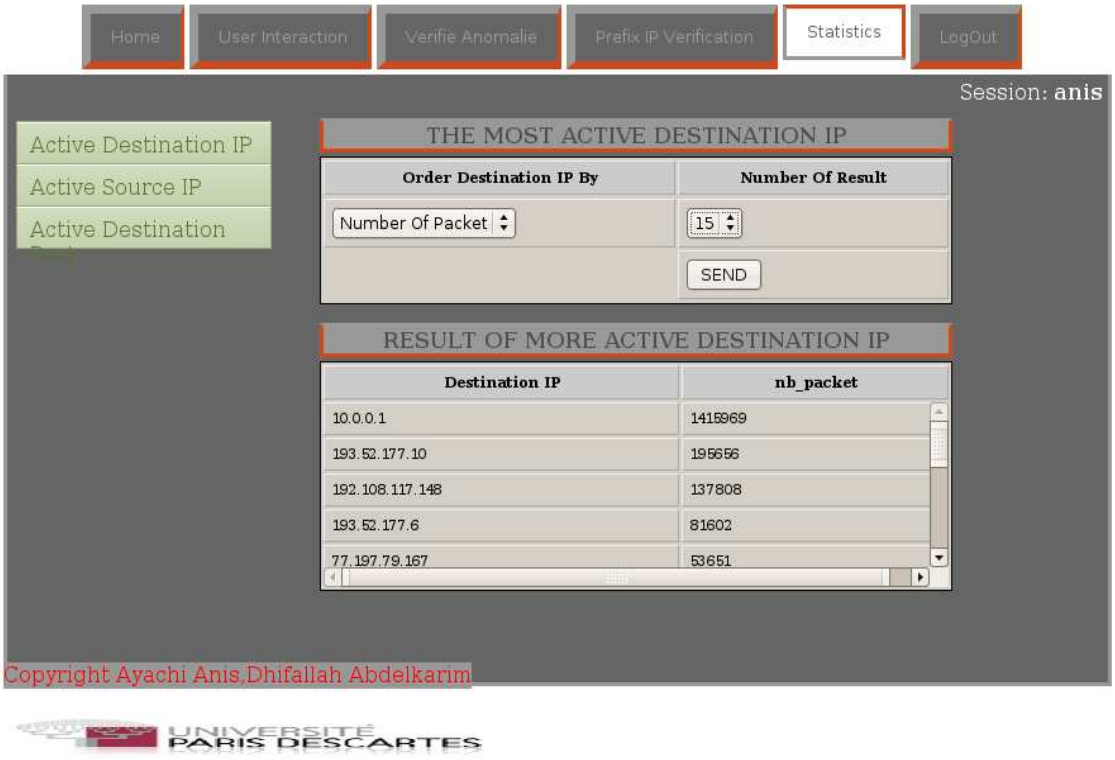

Figure [19] : les adresses IP destinations les plus actives selon un critère

Pour faire ce type de traitement, on a besoin d'une requête SQL et un traitement supplémentaire en utilisant un tableau associatif. La requête SQL est la suivante :

select ip_dest,sum($order) from trace group by ip_dest;

Pour rechercher les adresses IP source les plus actives selon un critère bien déterminé qui est le nombre de (paquet, syn, fin, rst, byte, ack), l'utilisateur à travers son interface, mentionne le critère voulu.

La réponse de système sera l'affichage des adresses IP source avec le nombre totale de critère demandé par l'utilisateur.

L'affichage se fait par ordre décroissant selon la valeur de critère.

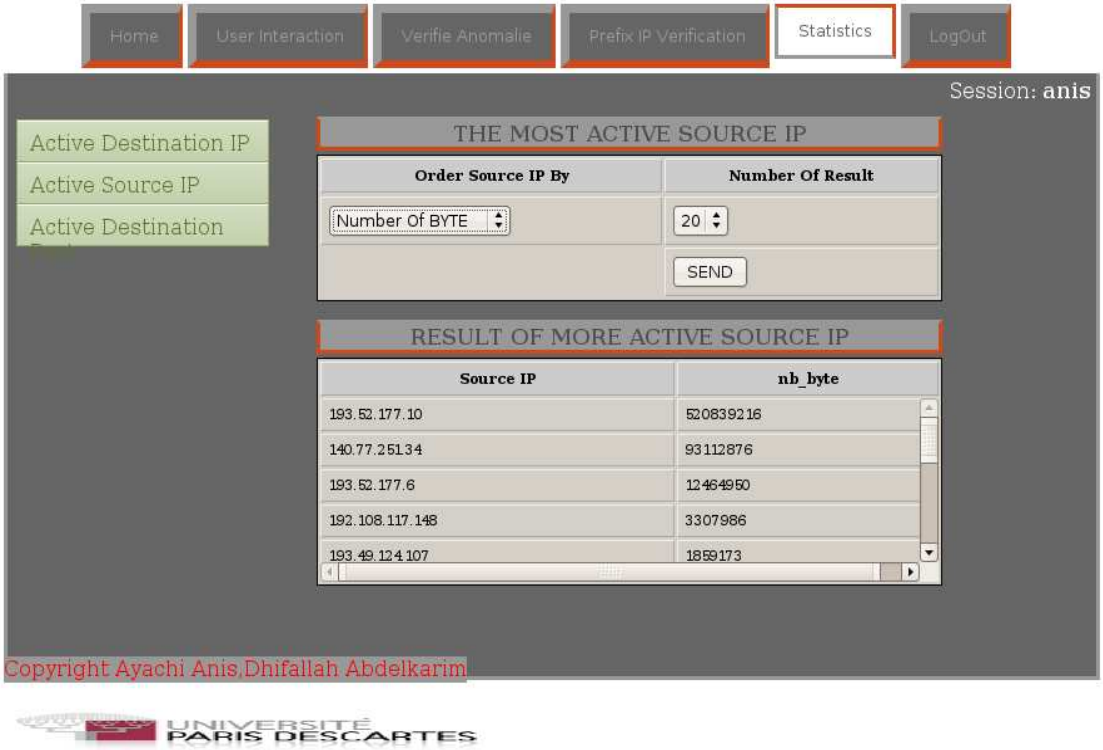

Figure [20] : les adresses IP sources les plus actives selon un critère

Pour faire ce type de traitement, on a besoin d'une requête SQL et un traitement supplémentaire en utilisant un tableau associatif. La requête SQL est la suivante:

select ip_src,sum($order) from trace group by ip_src;

✓ **Module de réponse :**

Ce module a pour rôle d'avertir l'utilisateur de différentes intrusions détectées par le système. Il est sous la forme d'une interface simple qui permet d'extraire à partir de la base de données des informations concernant les alertes. Ces informations sont :

- IP destination
- Type d'attaque
- Intervalle de temps

ALARME INFORMATION

Destination IP	Type	Intervall
79.6.216.227	BYTES	8
77.197.79.167	BYTES	13
192.108.117.148	BYTES	26
10.0.0.1	FLOWS	30
10.0.0.1	FLOWS	31
10.0.0.1	PACKET	31
10.0.0.1	FLOWS	36
10.0.0.1	PACKET	36
10.0.0.1	PACKET	60
10.0.0.1	FLOWS	60

Figure [21] : informations concernant les différentes intrusions détectées

La requête SQL qui permet de faire ce type de traitement est :

select ip_dest,type,intervall from alarme order by intervall;

VI.CONCLUSION

Le système de détection d’intrusion s’avère indispensable en complément d’outils de sécurité plus conventionnels. Il permet en effet de détecter des comportements qui peuvent mettre en cause la confidentialité ou la disponibilité d’un système**.**

Dans notre projet, nous avons développé un système de détection d'intrusion de type NIDS. Les résultats de test de ce système sur notre trace, sont satisfaisables. Cette satisfaction, ne signifie pas que notre système est parfait, il serait merveilleux de pouvoir crier victoire mais...comme vous le savez déjà qu'une sécurité parfaite n'existe pas.

Malgré les recherches énormes qui ont été déjà réalisés ces dernières années, aucune recherche n'a abouti à un système de détection d'intrusion permettant de garantir une sécurité à 100%.

References

[1] Bryan Gatenby: Denial of Service Attack Detection & Mitigation.

[2] Haining Wang Danlu Zhang Kang G. Shin: Detecting SYN Flooding Attacks

[3] Marina Thottan and Chuanyi Ji: Anomaly Detection in IP Networks.

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

[5] PratyushChandra: Intrusion Détection Systems.

[6] Nathalie Dagorn: Détection et prévention d'intrusion : présentation et limites (Rapport de recherche).

[7] BoonPing Lim Md. Safi Uddin:"Statistical-based SYN-flooding Detection Using Programmable Network Processor".

[8] Silvia Farraposo1, Philippe Owezarski2, Edmundo Monteiro :"Détection, classification et identification d'anomalies de trafic".

[9] Vasilios A. Siris, Fotini Papagalou: Application of anomaly detection algorithms for detecting SYN flooding attacks.

[10] Ludovic Mé: Détection des intrusions dans les systèmes d'information.

[11] Ludovic Mé : Méthodes et outils de la détection d'intrusion.

[12] http://www.authsecu.com.