الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

en Télécommunication
Spécialité : Réseaux & Télécommunications

Présenté par

KAAB Oussama

&

LABOUDI Mohamed Amine

Détection et atténuation des attaques DDOS dans un réseau SDN à l'aide de l'apprentissage automatique

Proposé par : Mlle BOUCHAIR Maria & Mr ZAIR Samir

Année Universitaire 2021-2022

Remerciements

Avant tout, nous tenons à exprimer notre profonde gratitude à notre promoteur mr zair pour la confiance qu'il nous a accordé en acceptant de nous encadrer dans ce mémoire. Nous le remercions pour son implication, ses conseils et l'intérêt qu'il a porté à notre travail.

Nous adressons nos vifs remerciements aux membres des jurys pour avoir accepté d'examiner et juger ce travail. Nous tenons aussi à remercier nos chères familles pour leurs soutient, encouragements et leurs bienveillance pour notre bien-être et notre succès. On tient à remercier nos amis(e)s pour leur sincère amitié et confiance. Nous leur devons toute notre reconnaissance et notre attachement. A tous ces intervenants, on présente nos sincères remerciements, notre respect et notre gratitude.

KAAB.O & LABOUDI.A

Dédicaces

Dédicaces 1

Je dédie ce modeste travail à ma chère mère

A mon cher père

A mes chères petites sœurs

A mes chers grands parents

Ainsi qu'à mes tantes et mes oncles

A mon binôme amine et toute sa famille

Mes cher(e)s ami(e)s

oussama

Dédicaces 2

Je dédie ce modeste travail à ma chère mère fadhila

A mon cher père ahmed

A mon cher frère abdelwahab

A mes chères tantes et mes oncles

A mon binôme oussama et toute sa famille

Mes cher(e)s ami(e)s

Amine

ملخص:

الشبكات المحددة بالبرمجيات (SDN) هو مفهوم ناشئ مصمم لاستبدال الشبكات التقليدية عن طريق تفكيك التكامل الرأسي. السيطرة المركزية هي أكبر فائدة منSDN ، ولكن نقطة فشل واحدة هي أيضًا فاشلة إذا كان هجوم رفض الخدمة الموزع (DDoS) يجعله بعيد المنال. توفر هذه الذاكرة حلاً فعالًا يعتمد على التعلم الآلي (ML) al- gorithms لاكتشاف هجمات DDoS والتخفيف منها بمساعدة Mininet ووحدة تحكم Ryu لمحاكاة الشبكة. في المقابل، تمت محاكاة هجمات DDoS باستخدام Hping3 الأداة. تثبت هذه الذاكرة أن نوع الطوبولوجيا مهم ويمكن أن يؤثر على النسبة المئوية لنجاح هجمات DDoS في مثل هذه الشبكات. تم اختبار وتقييم ستة خوارزميات ML خاضعة للإشراف (LR و K-NN و NB و SVM و DT و RF) باستخدام مجموعة بيانات اصطناعية. تظهر النتائج أن DT و RF هما الأفضل مقارنة بالخوارزميات الأخرى بدقة 100٪. يُظهر النظام المقترح كفاءته في الكشف عن هجمات DDoS والتخفيف من حدتها باستخدام مصنف RF وخمس ميزات فقط. في الوقت نفسه، تم توفير التخفيف من خلال إضافة قاعدة تدفق إلى المفتاح لإسقاط حركة المرور الضارة .

كلمات المفاتيح: Réseau défini par logiciel (SDN), attaque par déni de service distribué (DDoS), apprentissage automatique (ML)

Résumé : Le réseau défini par logiciel ou SDN (Software-Defined Networking) est un concept émergent conçu pour remplacer les réseaux traditionnelles en brisant l'intégration verticale. Le contrôle central est le plus grand avantage du SDN, mais un point de défaillance unique est également un echec si une attaque par déni de service distribué (DDoS) le rend inaccessible. Ce mémoire fournit une solution efficace basée sur des algorithmes d'apprentissage automatique (ML) pour détecter et mitiger les attaques DDoS à l'aide de Mininet et du contrôleur Ryu pour simuler le réseau, tandis que les attaques DDoS ont été simulées à l'aide de l'outil Hping3. Ce mémoire prouve que le type de la topologie est significatif et peut affecter le pourcentage de réussite des attaques DDoS dans tels réseaux. Six algorithmes supervisés (LR, K-NN, NB, SVM, DT et RF) ont été testés et évalués à l'aide d'un ensemble de données synthétiques. Les résultats montrent que DT et RF sont les

meilleurs par rapport aux autres algorithmes avec 100% de précision. Le système proposé montre son efficacité pour détecter et atténuer les attaques DDoS avec le classificateur RF et cinq critères uniquement tandis que l'atténuation a été

fournie en ajoutant une règle de flux au commutateur pour supprimer le trafic malveillant.

Mots clés : Réseau défini par logiciel (SDN), attaque par déni de service distribué (DDoS), apprentissage automatique (ML)

Abstract :

Software-Defined Networking (SDN) is an emerging concept designed to substitute traditional networking by breaking up vertical integration. Central control is the biggest benefit of SDN, but a single point of failure is also a failure if a distributed denial of service (DDoS) attack makes it unattainable. This memory provides an efficient solution based on machine learning (ML) al- gorithms to detect and mitigate DDoS attacks with the help of Mininet and the Ryu controller to simulate the network. In contrast, DDoS attacks were simulated using of Hping3 tool. This memory proves that the type of topology is significant and can affect the percentage of success of DDoS attacks in such networks. Six supervised ML algorithms (LR, K-NN, NB, SVM, DT, and RF) were tested and evaluated using a synthetic dataset. The results show that DT and RF are the best compared to the other algorithms with 100% of accuracy. The proposed system shows its efficiency in detecting and mitigating DDoS attacks with the RF classifier and only five features. At the same time, the mitigation was provided by adding a flow rule to the switch to drop the malicious traffic.

Keywords : Software-Defined Networking (SDN), Distributed Denial of Service Attack (DDoS), Machine learning (ML).

Listes des acronymes et abréviations

AI Artifical Intelligence

ANN Artificial Neural Network

API Application Programming Interface

ARP Address Resolution Protocol

BGP Border Gateway Protocol

CHARGEN Character Generator Protocol

CLI Command Line Interface

CNN Convolutional Neural Network

CPU Central Processing Unit

CSV Character Separated Values

DoS Denial of Service

DDoS Distributed Denial of Service

DDS Data Distribution Service

DL Deep Learning

DNS Domain Name System

DPID Delivery Point Identifier

DT Decision Tree

EIGRP Enhanced Interior Gateway Routing Protocol

HTTP Hypertext Transfert Protocol

ICMP Internet Control Message Protocol

IT Information Technology

ICMP Internet Control Message Protocol

IDS Intrusion Detection System

IDPS Intrusion Detection and Prevention System IoTInternet ofThings

IP Internet Protocol

IPS Intrusion Prevention Systems ISP Internet Service Provider

K-NN k-Nearest Neighbours

LAN Local Area Network

LDAP Lightweight Directory Access Protocol LR Logistic Regression

LSTM Long Short-Term Memory MAC Media Access Control

ML Machine Learning

MSSQL Microsoft Structured Query Language

NAT Network Address Translation

NB Naive Bayes

NetBIOS Network Basic Input Output System

NFG Network Flow Guard

NOS Network Operating System

NTP Network Time Protocol

ONOS Open Network Operating System

OSI Open Systems Interconnection

OSPF Open Shortest Path First

POF Protocol-Oblivious Forwarding

QoS Quality of Service

RFR andom Forest

RNN Random Neural Network

SDN Software-Defined Network

SNMP Simple Network Management Protocol

SOM Self-Organized Mapping

SSDP Simple Service Discovery Protocol

STPS panning Tree Protocol

SVM Support Vector Machine

TCAM Ternary Content Addressable Memory

TCP Transmission Control Protocol

TCL Tae Command Language

Table des matières

chapitre 2 : Attaque DDOS dans les réseaux SDN et approche ML

Chapitre 3 : Application, tests et résultats

Liste des figures

Liste des tableaux

Introduction générale

Dans les réseaux existants, les flux de trafic sont envoyés via des périphériques réseau tels que des routeurs et des commutateurs répartis dans le monde entier. Les périphériques réseau sont responsables du contrôle et de la propagation du trafic. Bien que ces réseaux traditionnels soient répandus, ils présentent certains inconvénients. Premièrement, il n'offre pas aux chercheurs la flexibilité d'expérimenter, d'ajouter de nouvelles fonctionnalités et un protocole, Deuxièmement, les réseaux existants ne peuvent pas être programmés et ne peuvent donc pas accepter de nouvelles commandes pour améliorer la fonctionnalité. Troisièmement, comme chaque appareil contient à la fois un plan de contrôle et un plan de données, le coût des appareils réseau est très élevé . Cependant, le SDN (Software Defined Networking) résout les problèmes des réseaux existants. Le SDN est un réseau programmable et virtualisé qui vous aide à insérer de nouvelles idées dans votre recherche. SDN supprime le plan de contrôle du plan de données. Le plan de contrôle est responsable du traitement des informations, tandis que le plan de données est responsable du transfert de données . Le SDN peut être déployé dans de nombreux réseaux différents, tels que les réseaux privés, les réseaux d'entreprise et les réseaux étendus. Malheureusement, le SDN présente de nombreux défis qui doivent être relevés. L'évolutivité, les performances et la sécurité sont quelques-uns des défis auxquels le SDN est confronté.

La structure centralisée du contrôleur pourrait entraîner de nombreux problèmes de sécurité. L'un de ces défis critiques est l'impact des attaques par déni de service distribué (DDoS) sur les réseaux SDN. De telles attaques peuvent rapidement faire tomber tout le réseau en faisant tomber le contrôleur. Étant donné que les paquets d'attaque sont envoyés avec de nombreuses adresses IP source usurpées, l'attaque DDoS peut causer des problèmes aux commutateurs et au contrôleur . De plus, dans le flot d'attaques DDoS, les attaquants utilisent de nombreuses adresses IP source d'usurpation différentes, ce qui rend impossible d'arrêter

les attaques en bloquant le trafic en fonction de la seule adresse IP source. La mise en œuvre pratique des méthodes de détection et de réponse DDoS a été la clé du fonctionnement régulier du réseau. Ce problème est plus présent dans les réseaux SDN en raison de son point de défaillance unique (contrôleur).

Chapitre 1 Notions de sécurité réseau, SDN

1.1 Introduction

Ce chapitre représente un aperçu de certains concepts dont nous avons besoin pour accompagner cette thèse. Il définit la sécurité du réseau et les attaques. Il montre le but de la sécurité du réseau. Il explique la différence entre les réseaux traditionnels et les réseaux SDN. Il explique également les principes fondamentaux d'OpenFlow. Enfin, il présente les attaques DDoS et les contre-mesures.

1.2 Sécurité du réseau

1.2.1 Définition

La première question à se poser est de savoir ce que nous entendons par « sécurité du réseau ». Plusieurs domaines d'activité possibles viennent à l'esprit dans ce vaste sujet, et chacun mérite un long article. Tout d'abord, la sécurité réseau est un sous-ensemble de la sécurité informatique [1].

La sécurité dans les réseaux commence par la protection physique. En termes de modèle OSI, il existe différents niveaux auxquels le chiffrement peut être effectué. Cela peut être fait au niveau des couches les plus basses, physique et liaison de données ou des couches supérieures, telles que le réseau (par exemple, Internet IP), le transport, la présentation, l'application, ou même par l'utilisateur [2].

Les aspects pratiques de la mise en réseau de la sécurité comprennent la détection des intrusions informatiques, l'analyse du trafic et la surveillance du réseau [3].

1.2.2 Objectifs

La sécurité du réseau se compose de plusieurs concepts, à savoir :

- **Confidentialité :** Il s'agit de permettre aux utilisateurs autorisés d'accéder à des données sensibles et protégées. Les informations et données sensibles ne doivent être divulguées qu'aux utilisateurs autorisés [4].

- **Intégrité :** fait référence aux méthodes permettant de s'assurer que les données sont réelles, exactes et protégées contre toute modification non autorisée par un utilisateur [4].

- **Disponibilité :** La disponibilité fait référence à la capacité d'accéder à des informations ou à des ressources dans un emplacement spécifié, et au format correct [5].

- **Authenticité :** Un processus qui assure et confirme l'identité de l'utilisateur fait référence à l'authentification. Le processus démarre lorsque l'utilisateur tente d'accéder à des informations ou à des données. L'utilisateur doit justifier de droits d'accès et d'identité [5].

- **Non-répudiation :** Fait référence à une méthode garantissant la transmission des messages entre les parties en utilisant la signature numérique ou le cryptage. La preuve des données authentiques et de l'origine des données peut être obtenue en utilisant un hachage de données [4].

- **Protection contre l'analyse du trafic :** l'analyse du trafic implique l'interception et l'examen des messages pour déduire des informations des schémas de communication qui peuvent être effectuées même après le cryptage des messages.

1.2.3 Attaques réseau

Les attaques de réseau sont un ensemble d'activités malveillantes qui perturbent, refusent, dégradent ou détruisent les données et les services des réseaux informatiques. Une attaque réseau cible l'intégrité, la confidentialité ou la

disponibilité des systèmes de réseau informatique en exploitant le flux de données sur les réseaux [3]. Il existe deux principaux types d'attaques réseau :

- **Actif** : les attaquants obtiennent un accès non autorisé et modifient les données, les supprimant, les cryptant ou les endommageant d'une autre manière. Les attaques actives incluent : l'attaque par usurpation d'identité, l'attaque par trou de ver, la modification, le déni de services [3].

1. Spoofing : Si un nœud malveillant ne parvient pas à montrer son identité, l'expéditeur change sa topologie

2. Modification : si le nœud malveillant modifie la route, l'expéditeur envoie le message via la route longue. Cette attaque provoque un retard de communication entre l'expéditeur et le destinataire [3].

3. Wormhole : Cette attaque peut être définie comme l'attaque par tunnel. À un moment donné, un attaquant reçoit le paquet et le connecte à un nœud malveillant sur le réseau. Ainsi, un nouvel utilisateur suppose qu'il a trouvé le chemin le plus court dans le réseau [3].

4. Fabrication : le mauvais message de routage est créé par un nœud malveillant. En d'autres termes, la route entre les appareils génère des informations incorrectes [3].

5. Déni de services : dans les attaques par déni de service, un nœud malveillant envoie le message à d'autres nœuds et utilise la bande passante du réseau. L'objectif principal du nœud malveillant est de rendre le réseau occupé. Si un message de nœud non authentifié arrive, le récepteur ne recevra pas cet avis car il est occupé et l'initiateur doit attendre cinq avertissements pour la réponse du récepteur [3].

- **Passif** : les attaquants accèdent à un réseau et surveillent ou volent des informations sensibles sans modifier les données, les laissant intactes. Les attaques passives comprennent l'analyse du trafic, l'écoute clandestine et la surveillance.

1. Analyse du trafic : un attaquant tente de percevoir le canal de communication entre l'expéditeur et le destinataire lors d'une attaque d'analyse du

trafic. Un attaquant peut déterminer la quantité de données circulant entre l'expéditeur et le destinataire. L'analyse du trafic n'apporte aucune modification aux données [3].

2. Eavesdropping : Il s'agit d'une attaque passive qui a eu lieu sur un réseau mobile ad-hoc. L'objectif principal de cette attaque est d'extraire des informations secrètes ou sensibles par la communication. Ces informations confidentielles peuvent être la clé privée ou publique d'un expéditeur ou d'un destinataire ou toute autre donnée confidentielle [3].

3. Monitoring : L'attaquant peut lire les données confidentielles dans cette attaque, mais il est incapable de mettre à jour ou de modifier les données [3].

1.2.4 Protection du réseau

Il existe de nombreuses façons d'infiltrer un réseau, de sorte que les professionnels de l'informatique peuvent utiliser de nombreuses techniques et stratégies différentes pour en sécuriser un. Certains des types les plus courants de solutions de sécurité réseau incluent :

- **Logiciel antivirus :** Un logiciel antivirus peut être installé sur tous les périphériques réseau pour les analyser à la recherche de programmes malveillants. Il doit être mis à jour régulièrement pour résoudre tout problème ou vulnérabilité.

- **Cryptage :** Le cryptage est un bon moyen de protéger les données lorsqu'elles sont transmises via un réseau ou un système informatique distribué. La protection offerte par le chiffrement est déterminée par la méthode de chiffrement utilisée, sa mise en œuvre et les règles administratives régissant son utilisation. La combinaison de la technologie de cryptage avec des mécanismes de contrôle d'accès au réseau dans un centre de sécurité réseau peut répondre à des exigences de sécurité supplémentaires telles que l'identité de l'utilisateur, l'autorisation d'accès et l'audit de sécurité [6].

- **Pare-feu :** En protégeant le réseau et les ressources du serveur contre les accès non autorisés et les attaques malveillantes, les pare-feu réseaux constituent

une ligne de défense principale [7]. Les pare-feu sont généralement utilisés à la périphérie du réseau ou au point d'entrée du réseau privé. Les pare-feu du réseau sont inspectés pour le trafic Internet entrant et sortant [7]. Les pare-feu peuvent autoriser ou bloquer le trafic d'entrée ou de sortie en fonction d'un ensemble de règles. Pour ce faire, les pare-feu réseau disposent d'un moteur basé sur des règles qui vérifie séquentiellement les paquets entrants jusqu'à ce qu'une correspondance soit identifiée.

- **IDS (Intrusion Détection System) :** Un système logiciel ou matériel de surveillance et d'analyse des événements se produisant dans un réseau ou un système informatique pour déterminer si une attaque s'est produit [8]. En d'autres termes, un IDS déclenchera toujours l'alarme en cas d'attaque, mais pas s'il n'y a pas d'attaque. Les IDS sont toujours présumés exempts d'erreurs. Cependant, un IDS n'est pas exempt d'erreurs ; il fait généralement deux types d'erreurs possibles : les fausses alarmes (déclencher l'alarme lorsqu'aucune attaque ne se produit) et les attaques manquantes (ne pas déclencher l'alarme lorsqu'il y a une attaque) [8].

La mise en œuvre de réseaux sécurisés nécessite de comprendre les vulnérabilités et les menaces courantes qui font du Web l'environnement idéal pour les attaquants.

1.3 SDN (Software-Defined Network)

1.3.1 Définition

Le SDN offre un certain nombre d'avantages, tels que la gestion à distance et centrale, des interfaces ouvertes et standard et une configuration dynamique ; Néanmoins, il introduit une nouvelle gamme d'attaques, qui autrement n'étaient pas présentes dans les réseaux traditionnels. Fondamentalement, en raison de sa nature centralisée et dominante, le contrôleur du SDN devient un point de défaillance unique et est vulnérable aux attaques par déni de service (DoS). En particulier, attaquer la disponibilité du contrôleur SDN à l'aide de DoS permet à l'attaquant de faire tomber l'ensemble du réseau. Dans les réseaux traditionnels, où les plans de

contrôle et de données sont étroitement couplés, les attaques DoS ciblent un serveur distant , Dans les réseaux SDN, les attaquants pourraient attaquer les plans de contrôle et de données afin d'épuiser les ressources de l'infrastructure réseau, provoquant ainsi une interruption de service. [9]

1.4 Objective du SDN

L'objectif de ces innovations est de simplifier l'administration du réseau et à l'instar de ce que la virtualisation a réalisé dans le monde des serveurs, de rendre la consommation des ressources réseaux par les applications plus flexible.

- ***Scalabilité*** **:** Ceci définit la capacité du SDN, plus spécifiquement dans le plan de contrôle, àgérer et traiter une charge de travail croissante. La scalabilité vise à élargir la capacité du SDN en mettant en œuvre des mécanismes tels que le « devolving » [12], le « clustering » [13] et le « high processing » [14] pour faire face à la charge croissante.
- ***Haute disponibilité*** **:** L'HD est un aspect important des services d'aujourd'hui qui devrait être disponible chaque fois qu'un client demande un service ou une ressource donnée. La disponibilitéest habituellement exprimée en pourcentage du temps de disponibilité au cours d'une année donnée. L'indisponibilité des services peut généralement se produire en raison de pannes de réseau ou de pannes de système [15], [16]. Les fournisseurs de réseaux déploient généralement des services de sauvegarde pour offrir une HD en implémentant du matériel serveur redondant, des composants d'OS serveur et de réseau.
- ***Sécurité*** **:** La sécurité du SDN consiste à protéger les informations contre le vol ou l'endommagement du matériel et des logiciels ainsi que contre l'interruption des services **[**17], [18]. La sécurisation du SDN englobe la sécurité physique du matériel, ainsi que la prévention des menaces logiques qui peuvent provenir du réseau ou des données. Les vulnérabilités du SDN sont laporte d'entrée vers des attaques de sécurité intentionnelles ou accidentelles.
- ***Fiabilité*** **:** Le SDN est considéré comme fiable lorsqu'il notifie les défaillances des

données en temps réel. Dans un tel réseau, il devrait y avoir une fiabilité minimale spécifiée pour l'acheminement des données critiques. Dans les implémentations actuelles, les contrôleurs SDN [19] doivent être capables de répondre en temps réel aux exigences de fiabilité et de ponctualité de l'acheminement.

- ***Performance*** **:** Le rendement (performance) fait référence à la quantité de tâches accomplies par les composantes SDN par rapport au temps et aux ressources (par ex., CPU et RAM) utilisés [20]. Ilexiste de nombreuses façons différentes de mesurer le rendement d'un réseau [21], [22], car chaque réseau est de nature et de conception différentes. En ce qui concerne le SDN, les mesures importantes sont la bande passante, le débit, la latence et le jitter.
- ***Elasticité*** **:** L'élasticité dans le SDN est la capacité d'assurer et de maintenir un niveau de service acceptable même en cas de défaillance d'un service, d'un réseau ou d'un nœud. Lorsqu'un élément SDN est défectueux, le réseau doit fournir un service opérationnel continu avec les mêmes performances. Afin d'accroître l'élasticité du SDN, il faut cerner les défis et les risques potentiels et y faire face pour protéger les services [23]

1.4.1 Comparaison entre les réseaux SDN et les réseaux traditionnels

Les réseaux ont traditionnellement été décrits par leur topologie physique ou leurs serveurs, commutateurs et routeurs connectés. Cela signifie qu'une fois que vous avez établi votre réseau, apporter des modifications coûte cher et prend du temps. Cette mise en réseau est incompatible avec le concept d'un centre de données "lights-out"1 ou d'un environnement Cloud qui nécessite de la flexibilité pour répondre aux demandes changeantes de la charge de travail. De plus, la navigation dans les commutateurs matériels est devenue plus complexe à mesure que la taille et les exigences parallèlement à des réseaux massifs, la configuration manuelle des commutateurs logiciels de réseau individuels s'est avérée extrêmement difficile et chronophage. C'est là que le SDN entre en scène [10].

La mise en réseau traditionnelle utilise une approche distribuée pour le plan de contrôle, des protocoles tels que ARP, STP, OSPF, EIGRP, BGP et autres fonctionnent séparément pour chaque périphérique réseau [10]. Bien que ces périphériques réseau soient connectés, il n'y a pas de système centralisé qui gère ou synthétise l'ensemble du réseau [10]. La différence la plus cruciale entre le réseautage traditionnel et le SDN est que le réseautage traditionnel est basé sur le matériel, tandis que le SDN est souvent basé sur le logiciel. Étant donné que le SDN est basé sur un logiciel, il est plus adaptable, permettant aux utilisateurs de mieux contrôler et gérer les ressources à distance dans le plan de contrôle [10]. Les commutateurs, routeurs et autres matériels physiques sont utilisés dans les réseaux traditionnels pour créer des connexions et faire fonctionner le réseau. Les contrôleurs SDN utilisent une interface nord pour communiquer avec les interfaces de programmation d'application (API) [10]. Grâce à cette connectivité, au lieu d'employer les protocoles requis pour les réseaux traditionnels, les développeurs d'appareils peuvent programmer le réseau directement. Dans les réseaux traditionnels, tous les plans de données et plans de contrôle sont situés dans une unité physique, qui est ensuite partagée pour augmenter la charge de trafic et la pression sur le CPU et la mémoire dans deux processus [10]. La séparation des plans de contrôle et des plans de données dans le SDN peut être facilement surveillée et gérée par le contrôleur et le réseau,

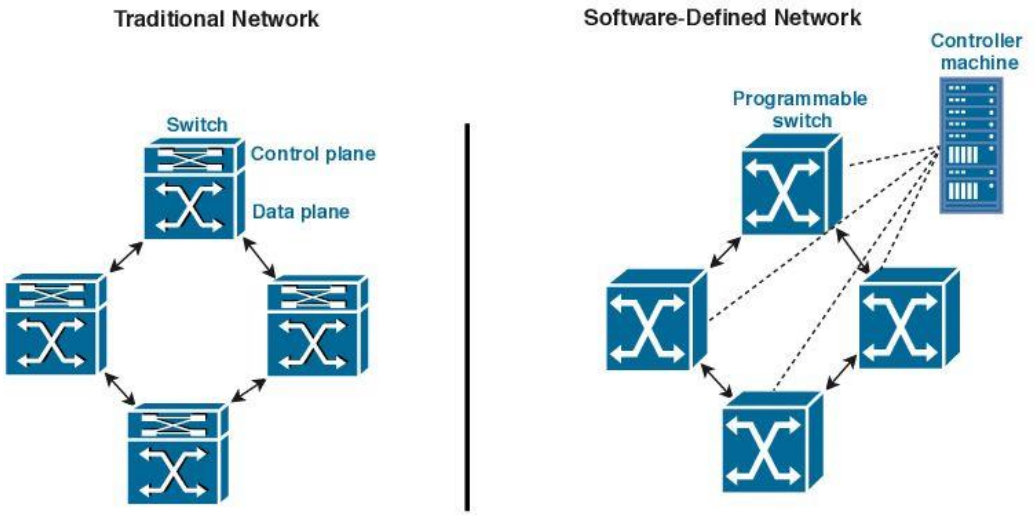


Figure I -1 L'architecture du SDN par rapport au réseau traditionnel

1.4.2 L'architecture du SDN

L'architecture SDN comprend trois couches : couche d'application couche de contrôle et couche d'infrastructure illustrées par la Figure I.1

Les applications (sécurité, gestion de la bande passante et de la charge…) existent dans la couche application et envoient leurs exigences de réseau au plan de contrôle via une NBI (Northbound interface). SDN utilise NBI pour communiquer avec les applications et le logique métier présent dans la couche application afin d'aider les responsables réseau à gérer le trafic et à déployer les services par programme. Dans SDN, le plan de contrôle est un contrôleur logiciel centralisé qui maintient la vue centralisée de l'ensemble du réseau et permet à l'administrateur réseau de diriger le système sous-jacent sur la manière de transférer le trafic.

Le contrôleur SDN est le cerveau du réseau qui interagit avec le plan de données via une SBI (Southbound Interface). Le SBI du SDN relaie les informations aux commutateurs et routeurs déployés dans un plan de données. Le plan de données (plan de transfert) se compose de certains éléments du réseau (par exemple, des commutateurs) chargés de transférer le trafic conformément aux règles définies par le contrôleur. L'architecture SDN fournira un ensemble d'interfaces de programmation d'application qui simplifie la mise en œuvre de services de réseau tels que le routage, la multidiffusion, le contrôle d'accès, etc. L'architecture SDN permet à un administrateur réseau d'implémenter un réseau gérable, hautement évolutif et adaptable, qui répondra aux besoins dynamiques de l'entreprise [11].

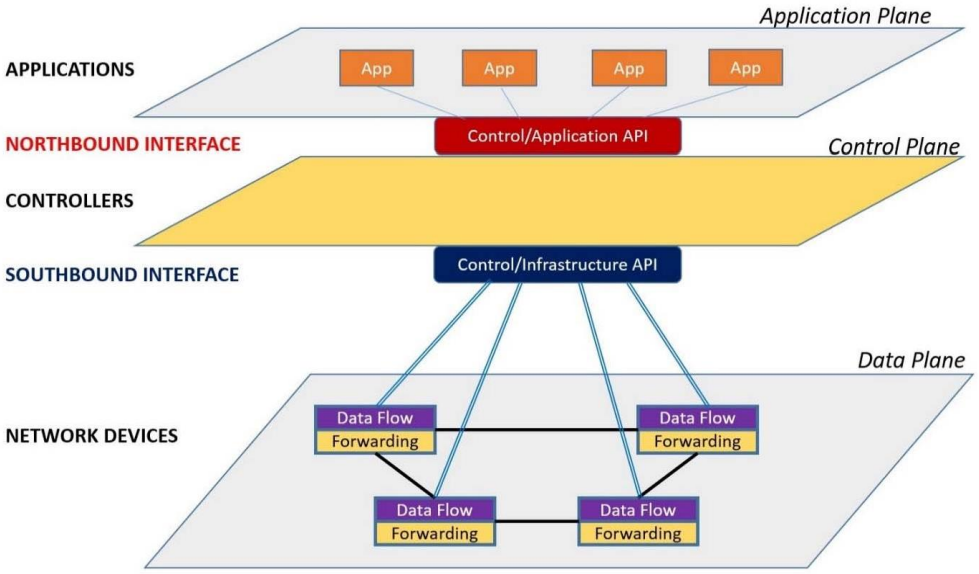


Figure I -2 Architecture du SDN

- **Application plan**

Dans l'architecture SDN, il s'agit de la couche la plus élevée. Cette couche gère toutes les applications métier et de sécurité. Les services logiciels essentiels gérés par cette couche comprennent la mesure, le routage, la qualité de service, l'équilibrage de charge, les systèmes de détection d'intrusion (IDS), les systèmes de prévention d'intrusion (IPS), la mise en œuvre de pare-feu et la gestion de la mobilité. De plus, cette couche communique avec une couche inférieure en utilisant les interfaces d'application vers le nord [11].

- **API Northbound**

Il sert de lien entre les plans de contrôle et d'application. Le NOS met plusieurs API à la disposition des développeurs d'applications. Il aide à programmer le réseau et masque les détails internes du réseau. Les API telles que FML, Procera2, NetKAT3 et Frenetic 4, entre autres, sont largement utilisées

- **Control plan**

Cette couche sert de lien entre la couche d'application et la couche données. Cette couche contient le système d'exploitation réseau (NOS), également appelé

contrôleur de réseau, qui contrôle la fonctionnalité globale du réseau. Un contrôleur logiquement centralisé est chargé de gérer l'ensemble du réseau et de faire des choix de routage, de transfert de flux et de suppression de paquets via la programmation [12]. Cette couche communique avec la couche située en dessous à l'aide d'API orientées vers le sud telles que OpenFlow et NetConf. Le contrôleur est un environnement logiquement centralisé et physiquement distribué qui communique entre eux en utilisant des interfaces vers l'ouest et vers l'est [12].

- **API southbound**

Les API sud du SDN sont utilisées pour communiquer avec le contrôleur SDN, les commutateurs réseau et les routeurs. Ce sont des protocoles qui permettent un contrôle plus efficace du plan de données. Bien que divers protocoles existent, notamment OpenFlow, ForCES, OpFlex5 et Protocol-Oblivious Forwarding (POF), de nombreuses organisations travaillent à normaliser OpenFlow, qui est devenu le protocole de facto [13].

- **Infrastructure layer**

La couche de données ou le plan de données est un autre terme pour le plan d'infrastructure. Il comprend des composants réseau qui interagissent avec le flux de données, tels que des ordinateurs physiques et virtuels, comme la couche physique du modèle OSI. La fonction principale de cette couche est de transférer les paquets en fonction des politiques/règles du contrôleur assignées et développées. Cette couche comprend des périphériques réseau physiques tels que des commutateurs, des routeurs et des points d'accès, ainsi que des commutateurs virtuels tels qu'OpenvSwitch6, Indigo7, Pica88, Nettle9 et OpenFlow [14].

1.4.3 Structure traditionnelle et SDN

❖ Reseau traditionnem

Une structure réseau traditionnel consiste en des périphériques connectés les uns aux autres pour la transmission de données. Dans l'architecture de réseaux traditionnels, le chemin de données et la prise de décision d'un processus de

commutation ou de routage sont collectés sur le même périphérique. Dans les réseaux traditionnels, un commutateur comprend à la fois le plan de contrôle et le plan de données. Les commutateurs traditionnels permettent à la fois de décider où envoyer le trafic via le plan de contrôle et de le transférer via un plan de données. Comme illustré dans la Figure I .3 [15].

Figure I . 3 Structure réseau traditionnelle

❖ **Reseau deffini par logiciel**

SDN extrait la fonctionnalité de couche inférieure en séparant le plan de contrôle et le plan de données. Dans le plan de contrôle SDN, les fonctions sont placées sur le contrôleur SDN et ce dernier n'est autre que le logiciel SDN exécuté par le serveur. Les commutateurs SDN (physiques ou virtuels) ont un plan de données qui transfère le trafic (trames) conformément aux règles définies par le contrôleur SDN. Le contrôleur SDN communique avec un plan de données via le protocole OpenFlow.

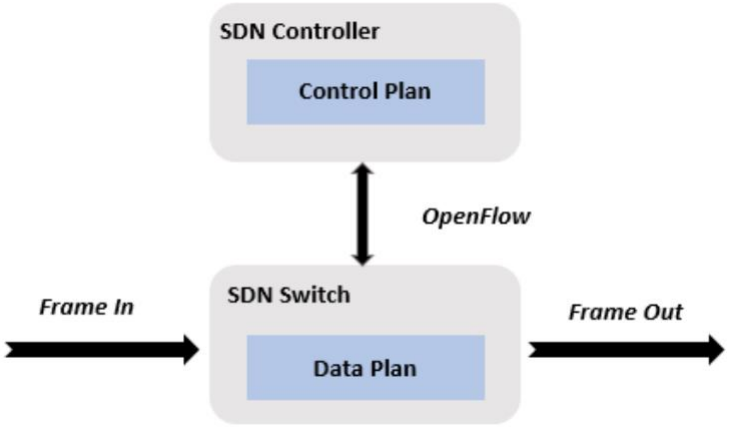


Figure I.4 Réseau défini par logiciel

1.4.4 Les Avantages du SDN

Les avantages du SDN pour les organisations sont nombreux. Une liste des principaux avantages est présentée dans les sous-sections suivantes :

- **Programmable du réseau**

Le SDN peut gérer l'ensemble du réseau par programmation. Le SDN permet d'éviter plus facilement la publication de plans et de protocoles personnalisés sur chaque appareil d'un réseau individuellement. La programmabilité est véritablement possible sur le seul plan de commande, permettant de modifier le comportement d'une seule unité ou de l'ensemble du réseau. En conséquence, le contrôleur peut rapidement améliorer la fonctionnalité de conception du trafic tout en réduisant la congestion du réseau [16].

- **Prix réduit**

La majorité des produits SDN sont gratuits. Certains systèmes, tels que NSX de VMware et la virtualisation de réseau Hyper V 11 de Microsoft, ne nécessitaient que les frais de licence pour le SDN service à payer [16].

- **Protection enrichie**

Le financement d'une machine virtuelle dans un environnement virtualisé est une tâche ardue. Le SDN, en revanche, fournit une surveillance sensible sur tous les appareils [16].

❖ **Gestion efficace du réseau**

Le SDN permet au gestionnaire de réseau de modifier la qualité du réseau à distance. En modifiant les caractéristiques du réseau en fonction de l'atterrissage de la tâche dans le réseau, un contrôle du réseau simple et fiable est possible [16].

1.5 Protocole Open Flow

OpenFLow est défini par L'ONF (Open networking foundation) comme étant un protocole de communication qui permet d'accéder directement au plan de données des périphériques réseau tels que les routeurs et les switch, à la fois physiques et virtuels à travers le réseau. Autrement dit, il permet de contrôler le comportement du plan de données d'une façon centralisée et programmable [17].

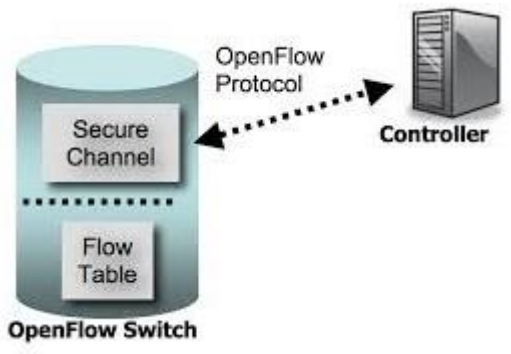


Figure I.5 Le protocole OpenFlow

1.5.1 Architecture OpenFlow

La figure suivante détaille les éléments d'un réseau OpenFlow. Le réseau est formé essentiellement d'un ou plusieurs commutateurs OpenFlow, un ou plusieurs contrôleurs, et finalement le protocole OpenFlow définissant tous les messages

échangés entre les contrôleurs et les commutateurs et permettant de standardiser la communication.

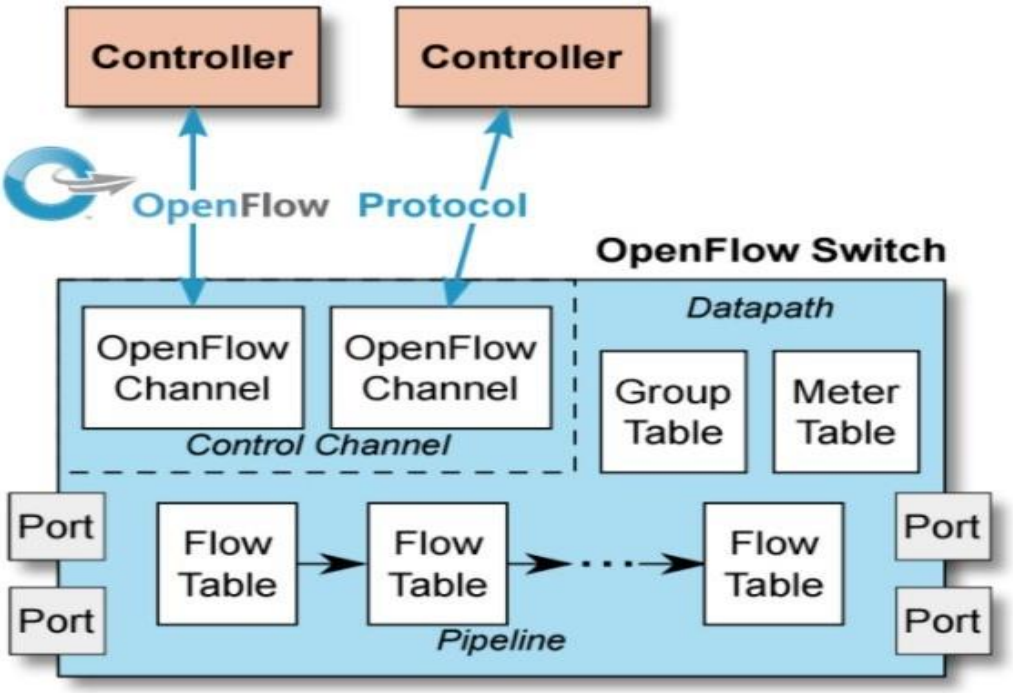


Figure I 6 architecture openflow

1.5.2 Canal OpenFlow (OpenFlow Channel)

Le canal OpenFlow est l'interface qui connecte chaque commutateur OpenFlow à un contrôleur. Cette interface permet au contrôleur de recevoir les messages du commutateur et de pouvoir le gérer à travers le réseau. Le canal doit être sécurisé afin d'assurer le bon déroulement des communications entre le commutateur et le contrôleur. Pour cela l'échangede message se fait au cours d'une session TCP établie via le port 6653 du serveur contrôleur ou à travers une connexion SSL/TLS (Secure Sockets Layer/ Transport Layer Security) [18].

1.5.3 Commutateur Open Flow

Un Switch Open Flow contient une ou plusieurs tables de Flux et une Table de Groupe, qui traitent les paquets entrants et les commutent vers la destination, il

s'appuie surun Canal sécurisé pour communiquer avec le contrôleur externe. Le contrôleur gère le Switch en utilisant le protocole OpenFLow, qui lui permet d'ajouter, d'effacer et de mettre à jour les entrées dans la ou les Table de Flux [18].

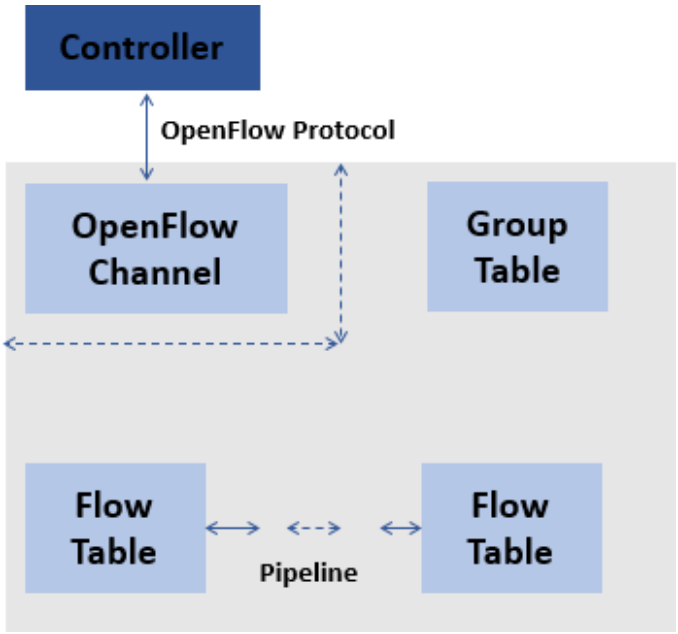


Figure I .7 Commutateur OpenFlow

1.6 Tables OpenFLow

Cette section décrit les composants de tables OpenFLow à savoir, les Table de Flux,les Tables de groupes, ainsi que les mécanismes de correspondance.

1.6.1 Tables de Flux

Chaque Table de Flux dans un Switch contient un ensemble d'entrées.

Champ de correspondance	Compteurs	Instructions

Figure I.8 Champs d'entré dans une table de flux

Les principaux éléments d'une entrée dans une table de Flux :

- **Le champ de correspondance 'Match Fields'** : c'est la partie sur laquelle le

contrôleur se base pour faire correspondre les paquets, cela consiste à vérifier le port d'entrée ou l'entêtedu paquet afin d'y appliquer une action X.

- **Compteurs** : Il est possible de disposer d'un certain nombre de statistiques. Dont on sert pour la gestion des entrées dans les tables de flux.
- **Instructions** : c'est une opération pouvant contenir en elle-même un ensemble d'actions à appliquer sur le paquet [18].

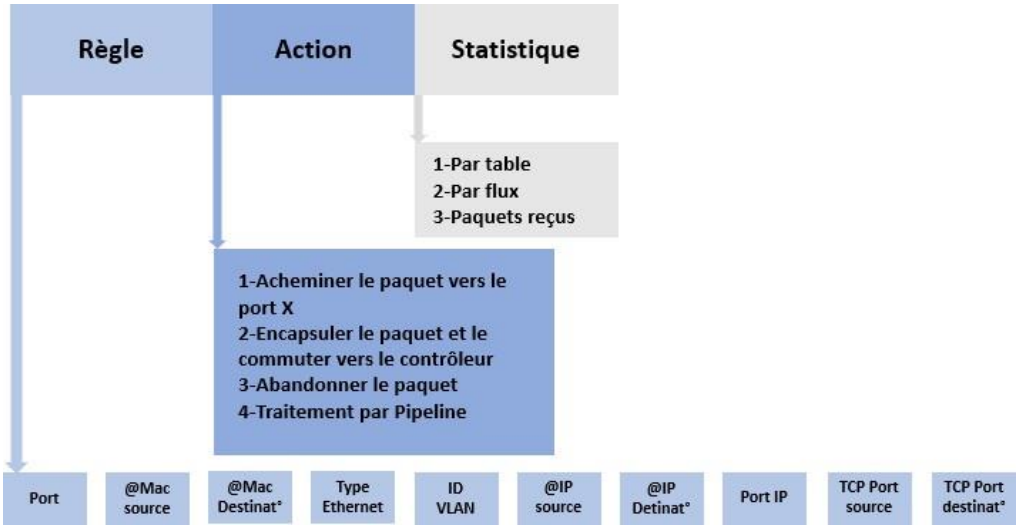


Figure I.9 Table de flux

Les instructions associées à une entrée de flux peuvent être soit des actions à appliquer au paquet correspondant soit des modifications du traitement (pipeline). Les actions à appliquer sur un paquet peuvent être :

- **Output :** Envoi du paquet par un port spécifié.
- **Set-field :** Modifier la valeur d'un champ d'en-tête spécifique, comme le TTL (*Time To live*), l'adresse MAC, ou l'ID du VLAN (*Virtual Local Area Network*), etc…
- **Drop :** Supprimer le paquet.
- **Group :** Traiter le paquet selon un groupe spécifique (entrée spécifique de la table degroupe)

Tandis que les instructions de modification du pipeline peuvent être :

- **Apply-Actions :** Pour appliquer immédiatement les actions sur le paquet.
- **Clear-Actions :** Pour supprimer une liste des actions du paquet.

- **Write-Actions :** Ajouter une liste d'actions au paquet.
- **Goto-Table :** Indique que le paquet doit être acheminé vers une table d'indice supérieur.

1.6.2 Table de groupe :

La table de groupe contient des entrées, et chaque entrée une liste d'actions appelée Conteneur d'actions les actions d'un ou plusieurs **Conteneurs d'actions** sont appliquées sur les paquets envoyés au groupe.

Chaque entrée dans la table de groupe contient :

Identifiant de Groupe	Type de Groupe	Compteurs	Conteneurs d'Actions

Figure I10 Champs d'entré dans une table de groupe

- **L'identifiant de groupe** : c'est un entier de 32 bits.
- **Le Type de groupe** : sert à déterminer la sémantique du groupe.
- **Les compteurs** : mis à jour quand un paquet est traité par un groupe.
- **Conteneur d'action** : un ensemble d'actions et de paramètres associés, qui sont définies pour les groupes

Le protocole OpenFLow supporte trois types de messages : Messages Contrôleur versSwitch, messages asynchrone et messages symétrique, chaque type a une sous-catégorie [18].

1.6.3 Message OpenFlow

❖ **Messages depuis le Contrôleur vers Switch**

Sont initiés par le contrôleur, ils servent à gérer ou vérifier l'état du switch, ces type de messages peuvent ou non demander une réponse de la part du switch.

Modify-State : Ce type de message est envoyé pour gérer l'état dans les Switch. Sa fonction primaire est d'ajouter, modifier ou effacer les entrées dans les tables.

Read-state : Ces messages sont utilisés par le contrôleur pour collecter différentes informations du switch, comme sa configuration actuelle, des statistiques et des capacités.

Packet-Out : sont utilisés pour transférer les paquets reçus par les messages Packet-In.

- **Messages asynchrones**

Les messages asynchrones sont envoyés par le switch vers le contrôleur pour indiquer un changement d'état ou l'arrivé d'un paquet.

Packet-In : Avec ce type de message le switch transfère le contrôle du paquet aucontrôleur.

Flow-removed : informe le contrôleur de la suppression d'une entrée dans la table de Flux

Port-Status : Informe le contrôleur d'un changement sur un port du switch.

- **Messages symétriques**

Les messages symétriques sont envoyés sans aucune sollicitation ni du switch ni du contrôleur.

Echo : ont comme utilité la vérification de la connectivité entre switch et contrôleur.

Hello : Ces messages sont échangés entre les deux une fois la connexion établie.

Error : Utilisé pour signaler de part et d'autre des problèmes de connexion.

- **Etablissement d'une connexion commutateur-contrôleur**

Nous allons citer ci-dessous trois cas d'un établissement d'une connexion entre le commutateur OpenFlow et le contrôleur :

- **Cas n°1**

Tout d'abord, il faut renseigner l'adresse IP du contrôleur au niveau du commutateur, ilpeut y avoir plusieurs pour la redondance. Lors du démarrage du commutateur OpenFlow, ce dernier envoie un paquet « OFPT_HELLO » avec le numéro de version d'OpenFlow supportée. Le contrôleur vérifie la version d'OpenFlow supportée par le commutateur et lui répond par un message « OFPT_HELLO » en indiquant la version d'OpenFlow avec laquelle ils communiqueront. La connexion est ainsi établie.

La figure suivante présente les principales étapes de la connexion entre le contrôleur etle commutateur OpenFlow.

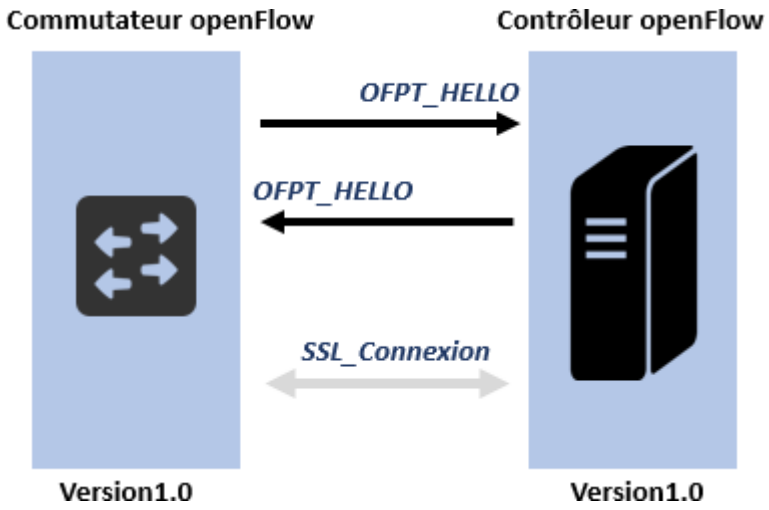


Figure I.11 Connexion au contrôleur OpenFlow

❖ **Cas n°2**

Comme dans le cas précédent le commutateur OpenFlow envoie un paquet « OFPT_HELLO » avec la version du protocole utilisé, le contrôleur s'aperçoit qu'il ne supportepas la version OpenFlow du commutateur. Il lui retourne donc un paquet « OFPT_ERROR » en indiquant que c'est un problème de compatibilité, comme illustré dans la figure suivante :

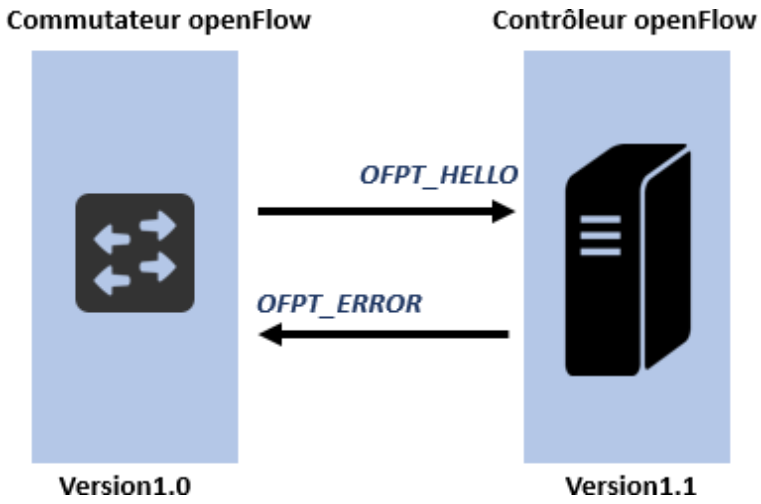


Figure I 12 Échec de la connexion au contrôleur OpenFlow

❖ **Cas n°3**

Comme dans les cas précédents le commutateur envoie un paquet « OFPT_HELLO » au contrôleur, si celui-ci ne répond pas il tente alors de joindre les éventuels autres contrôleurs qui lui ont été paramétrés. S'il ne parvient pas à les joindre, il se met alors en mode urgence « EMERGENCY MODE ». Le commutateur utilise sa table de flux par défaut, si toutefois un paquet ne correspond à aucun enregistrement dans la table il le supprime, comme le montre lafigure ci-dessous : [19]

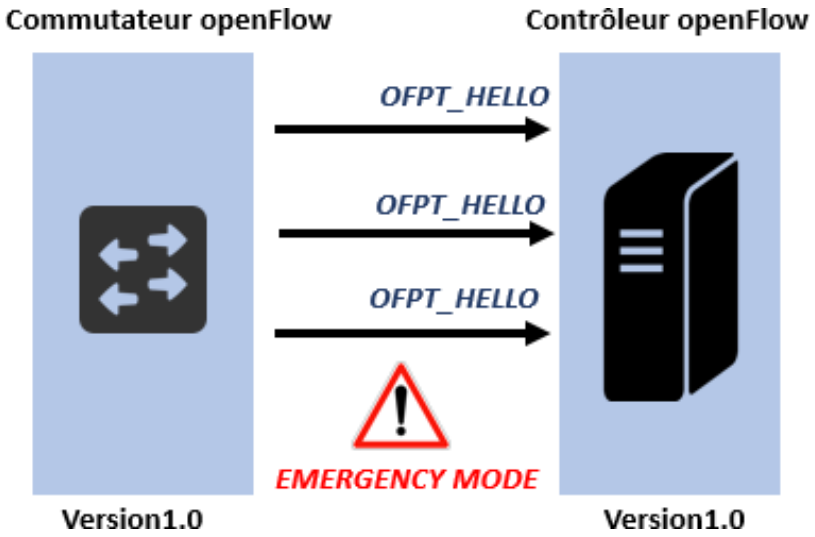


Figure I.13 Mode d'urgence

1.6.4 Fonctionnement du réseau OpenFlow

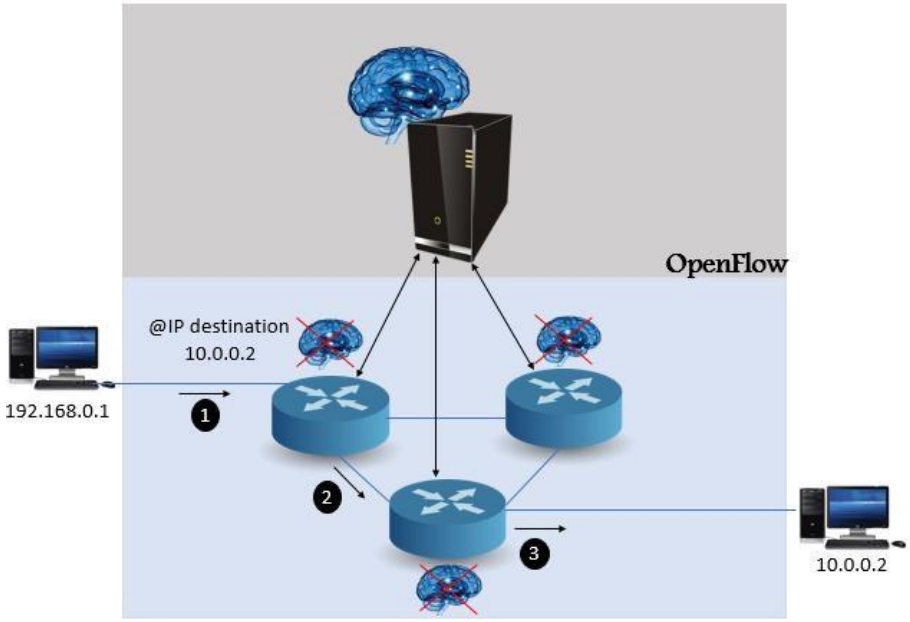


Figure I 14 Schéma d'un simple réseau SDN

Nous pouvons résumer le fonctionnement du réseau OpenFlow par la figure suivante :

Les commutateurs analysent les trames et déterminent leur interface de sortie, mais ce comportement découle de règles émises par le contrôleur. Le commutateur qui réceptionne la trame (1) signale l'évènement au contrôleur à travers un message de type « PACKET_IN », et reçoit en retour des règles au cours d'un échange par un message « PACKET_OUT » donnant l'instruction à suivre (2), afin de décider le transfert à travers quelle interface (3) vers le commutateur suivant approprié.

Openflow utilise la notion de pipeline pour le traitement des flux. Le pipeline définit comment les paquets vont interagir avec les tables de flux. À chaque entrée dans une table de flux, un paquet est inspecté pour déterminer si une entrée correspondante existe, si elle existe, les actions associées sont appliquées. Une action possible est de pouvoir envoyer le paquet vers une autre table de flux [20].

1.7 Traitement du pipeline

Le schéma suivant illustre le fonctionnement de pipeline :

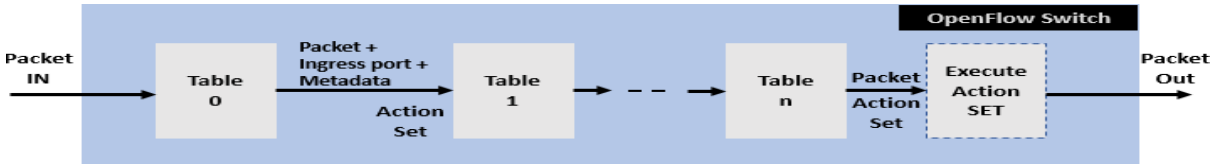


Figure I.15 traitement d'un paquet a travers le pipeline

Les tables de flux d'un commutateur OpenFlow sont numérotées dans l'ordre, en commençant par 0. Le traitement du Pipeline commence toujours à la première table de flux, le paquet est d'abord comparé aux entrées de la table de flux « 0 » et peut continuer sur des tables de flux supplémentaires. Si l'entrée correspondante est trouvée, les instructions associées à l'entrée de flux spécifiques sont exécutées. Si aucune correspondance ne se trouve pas dans une table de flux, le résultat dépend de la configuration de commutateur ; le paquet peut être transmis au contrôleur sur le canal OpenFlow, abandonné, ou peut continuer sur la prochaine table de flux.

Les instructions de traitement du pipeline permettent l'envoi des paquets à la table suivante pour un traitement ultérieur et permet à l'information sous forme de métadonnées d'être transporter entre les tables. Le traitement des tables du pipeline s'arrête lorsque le champ d'instructions associé à l'entrée correspondante dans la table de flux ne spécifie pas une table de flux suivante à ce stade, le paquet est modifié et transmis [18].

1.8 Contrôleur Ryu :

Le composant le plus important du SDN est le contrôleur, à travers lequel les différentes applications réseau sont programmées. Les contrôleurs diffèrent les uns des autres par le langage de programmation qu'ils prennent en charge, la version du protocole dans lequel ils fonctionnent et la prise en charge du multithreading, en plus d'autres différences liées au domaine d'utilisation de ces contrôleurs, que ce soit dans les centres de données, le cloud computing, etc. L'un des plus célèbres de ces contrôleurs est le contrôleur open source RYU, qui est programmé en Python et prend en charge jusqu'à OFF 1.5 du protocole OF [19].

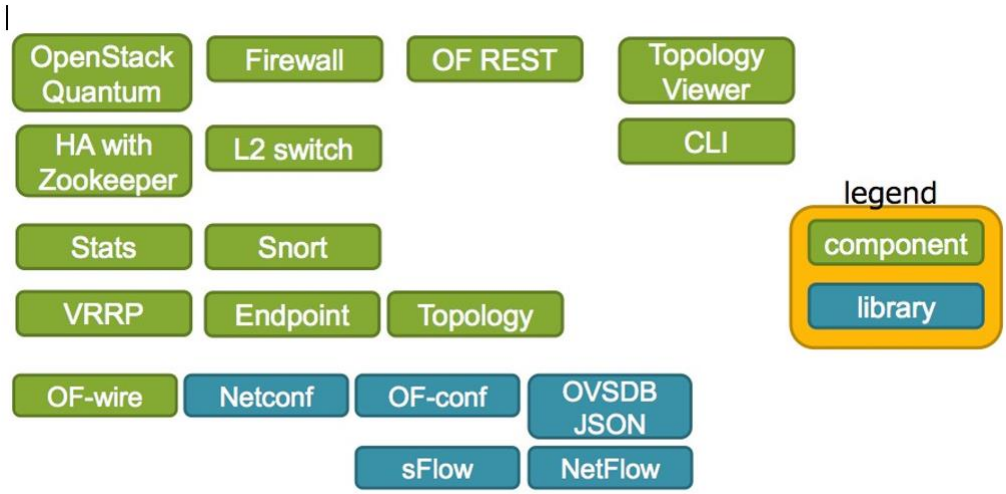


Figure I.16 Composants et bibliothèques inclus dans Ryu

La Figure I.16 montre les composants du contrôleur RYU qui facilitent le développement d'applications réseau, Facilitant la gestion du réseau, y compris OF config qui est utilisé pour les paramètres du protocole OF, et la bibliothèque Open Virtual Switch Data Base (OVSDB) qui est utilisée pour le commutateur de paramètres à gérer. Le protocole OF, y compris la création, la suppression et la modification de règles dans une table de flux et la bibliothèque NET Config qui aide à implémenter les paramètres sur les périphériques réseau [20]

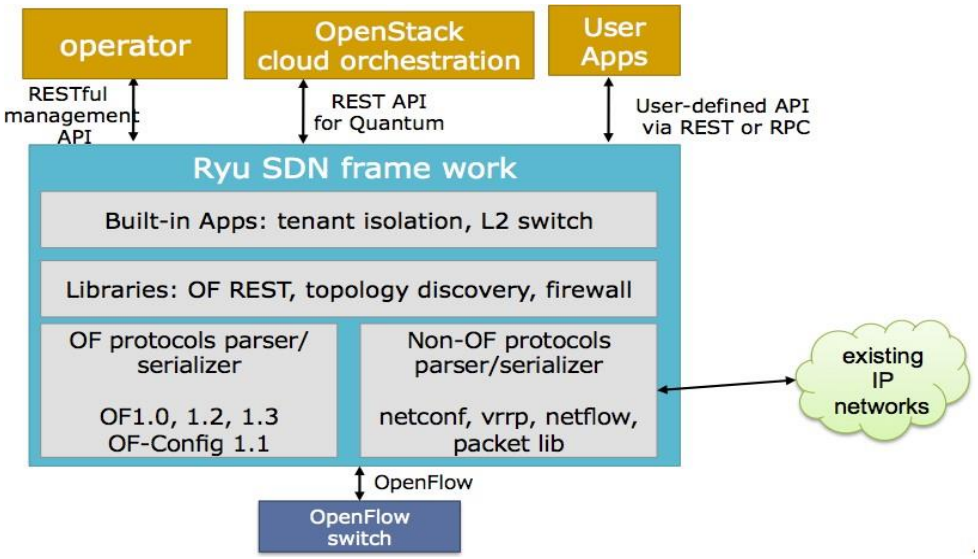


Figure I.17 architecture de ryu contrôleur

1.8.1 Types de contrôleurs SDN

- ***Beacon*** [21]**:** Beacon est un contrôleur SDN qui a été introduit en 2010. Il a été utilisé dans plusieurs recherches projets. C'est un contrôleur basé sur Java. Il peut fonctionner sur de nombreuses plates-formes, y compris Linux multi cœurs haut de gamme, et les téléphones Android.
- ***DISCO*** [22]**:** Disco est un contrôleur distribué. Il est principalement utilisé pour les réseaux WAN et les réseaux superposés. Chaque contrôleur est responsable d'un domaine réseau. Les régulateurs communiquent entre eux par l'intermédiaire d'un canal inter-contrôleur. DISCO peut s'adapter dynamiquement à différentes topologies de réseaux hétérogènes.
- ***IRIS*** [23]**:** IRIS est une plate-forme de contrôleur SDN pour contrôler la base OpenFlow. Il peut gérer un grand réseau. IRIS prend en charge les architectures qui sont évolutives horizontalement. Ainsi, les serveurs peuvent être ajoutés dynamiquement au cluster de contrôleurs. Cela augmente le facteur de performance du contrôle plane.
- ***Maestro*** [24]**:** Maestro est le premier système de contrôle OpenFlow qui exploite le parallélisme. Dans Maestro, les programmeurs peuvent modifier la fonctionnalité du data plane en écrivant des programmes simples à filetage unique. Maestro a son propre ensemble des conceptions et des techniques qui aident le protocole OpenFlow. Il s'agit d'un contrôleur basé sur Java et très portable pour différents systèmes d'exploitation et architectures.
- ***OpenDaylight*** [25]**:** OpenDaylight s'inspire de Beacon. Il s'agit d'un contrôleur basé de Java dérivé de Beacon. Il supporte OpenFlow et d'autres Southbound APIs. L'OpenDaylight est présent dans sa propre machine virtuelle Java (JVM).
- ***NOX*** [26]**:** NOX est la première plate-forme SDN OpenFlow Controller pour la construction d'applications de contrôle réseau. Il a été initialement développé par Nicira Networks, aux côtés d'OpenFlow. Plus tard, NOX a été donné à la communauté SDN. Les applications peuvent être en Python ou en C++ et peuvent être chargées dynamiquement.
- ***POX*** [27]**:** Pox est similaire au NOX Controller. POX est un contrôleur SDN qui permet le

développement et le prototypage rapides du réseau. Il suit le protocole OpenFlow, et qui sert a joué le role d'un framework entre les commutateurs OpenFlow.

- ***Floodlight*** [28]**:** Floodlight est un contrôleur SDN Open source. Il s'agit d'un contrôleur OpenFlow de classe entreprise, basé sur Java. Il fonctionne à la fois avec les commutateurs physiques et les commutateurs virtuels qui utilisent le protocole OpenFlow.

Le tableau 1 effectue une comparaison entre les propriétés des différents contrôleurs SDN.

Propriétés	**Contrôleurs SDN**								
	Beacon	DISCO	IRIS	Maestro	OpenDaylight	NOX	POX	Floodlight	Ryu
Support du Parallélisme	Oui	Oui	Oui	Oui	Oui	Non	Non	Oui	Oui
Protocole	OpenFlow	OpenFlow	OpenFlow OVSDB	OpenFlow	OpenFlow OVSDB	OpenFlow	OpenFlow	OpenFlow	OpenFlow OVSDB
L'architecture	Centralisée	Distribuée	Centralisée	Centralisée	Distribuée	Centralisée	Centralisée	Centralisée	Centralisée
Langage de programmation	Java	Java	Java	Java	Java	C++	Python	Java	Python
API supportés	Ad-hoc API	REST API	REST API	Ad-hoc API	REST API	Ad-hoc API	Ad-hoc API	REST API	Ad-hoc API
Plateformes supportées	Linux Windows	Linux Windows	Linux Windows	Linux Windows Mac OS	Linux	Linux Windows Mac OS	Linux Windows Mac OS	Linux Windows Mac OS	Linux

Tableau 1 Comparaison entre les propriétés des contrôleurs SDN

1.9 Défis du SDN

Même si le SDN a été identifié comme la principale solution aux défis croissants de l'infrastructure du réseau, il en est encore à ses balbutiements. Des avantages tels qu'une fonctionnalité accrue, un coût réduit et une efficacité accrue ont été mis en évidence, mais différents défis doivent également être résolus. Certains de ces défis sont :

1.9.1 Évolutivité

Le principal problème avec le SDN est l'évolutivité. Il fait référence à la capacité de s'étendre pour s'adapter à la croissance du réseau. Malheureusement, le contrôleur peut devenir un goulot d'étranglement pour l'évolutivité. L'introduction d'une infrastructure de contrôleur distribuée ou peer-to-peer peut partager la charge de communication du contrôleur [29]. Cependant, afin d'orienter les communications entre les contrôleurs utilisant l'est et API westbound, une vue globale du réseau est requise [29]. Outre l'évolutivité du contrôleur, il existe plusieurs autres problèmes d'évolutivité, tels que la surcharge de configuration du flux et la résilience aux pannes [30].

1.9.2 Flexibilité et performances

Une question fondamentale du SDN est de savoir comment traiter avec compétence les flux de traitement de paquets de haut niveau. Dans ce cas, la flexibilité et les performances sont les deux variables les plus importantes à prendre en compte. La vitesse de traitement d'un nœud de réseau, prenant en compte à la fois le débit et la latence, est appelée performance . La technique SDN pour gérer les nouveaux paquets doit ajouter une possibilité de programmation. Cependant, en même temps, cela entraîne des problèmes de performances. Dans [31] ils démontrent que les contrôleurs actuels sont incapables de gérer un grand nombre de flux dans des liaisons 10Gbps.

1.9.3 Sécurité

Par rapport aux réseaux traditionnels, la surface d'attaque du SDN est augmentée en séparant le plan de contrôle du plan de données [32]. Selon l'analyse de sécurité, le cadre SDN est vulnérable à une variété de vulnérabilités de sécurité, notamment [33] :

- Accès non autorisé, tel qu'un accès non authentifié à une application ou un accès non autorisé au contrôleur.

- Fuite de données, telle que la découverte des règles de flux (attaque du canal latéral du tampon d'entrée) et la découverte de la politique de transfert (analyse du temps de traitement des paquets).

• Modification des données, telle que la modification des règles de flux pour modifier les paquets. Voici les principales sources de danger possibles dans le SDN :

• Flux de trafic falsifiés ou fabriqués.

• Attaques sur les vulnérabilités des commutateurs.

• Attaques par déni de service sur les communications du plan de contrôle.

• Attaques et vulnérabilités du contrôleur.

• Aucune méthode n'est en place pour s'assurer que le contrôleur et les applications de gestion sont fiables.

• Attaques des postes administratifs et faiblesses de ceux-ci.

La figure montre les attaques possibles de l'architecture SDN :

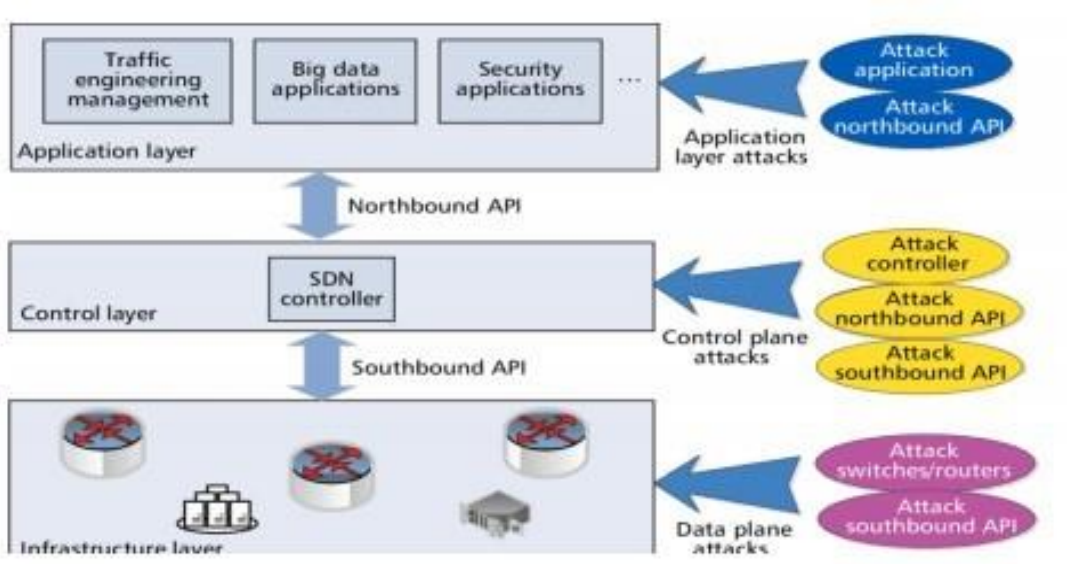


Figure I 18 Attaques potentielles sur l'architecture SDN

1.10 Déni de service (DOS)

Le jeudi 6 août 2009, l'un des services de réseautage social et de microblogging en ligne les plus suivis, Twitter 12, a été bloqué pendant plusieurs heures, faisant taire ses millions de Tweeters [34]. L'incident était le résultat de ce qu'il a décrit comme une attaque par déni de service "en cours". Le premier mot officiel qui est arrivé était une

déclaration plutôt laconique : "Le site est en panne - Nous en déterminons la cause et fournirons une mise à jour sous peu". Cependant, son co-fondateur Biz Stone a rapidement mis à jour ce message : "En ce jeudi matin autrement heureux, Twitter est la cible d'une attaque par déni de service."

1.10.1 Définition

Une attaque par déni de service (DoS) est un seul attaquant contre une cible. Ce type d'attaque refuse l'accès d'autres utilisateurs légitimes aux services ou ressources partagés. Les attaques DoS accomplissent cela en inondant la cible de trafic ou en lui envoyant des informations qui déclenchent un plantage afin que le serveur ne puisse pas faire la distinction entre les demandes valides et non valides. Dans les deux cas, l'attaque DoS prive les utilisateurs légitimes (c'est-à-dire les employés, les membres ou les titulaires de compte) du service ou de la ressource qu'ils attendaient.

Il existe deux méthodes générales d'attaques DoS : les services d'inondation ou les services de plantage. Les attaques par inondation se produisent lorsque le système reçoit trop de trafic à mettre en mémoire tampon, ce qui les ralentit et finit par s'arrêter. D'autres attaques DoS exploitent des vulnérabilités qui provoquent le plantage du système ou du service cible. Dans ces attaques, une entrée est envoyée qui tire parti des bogues de la cible qui par la suite plantent ou déstabilisent gravement le système à accéder ou à utiliser.

1.10.2 Attaques par déni de service distribué (DDoS)

Contrairement aux attaques DoS simples, les attaques DDoS tirent parti de nombreux hôtes compromis et visent à épuiser les ressources réseau le plus rapidement possible. Une fois le réseau perturbé, il ne peut fournir aucun service aux utilisateurs légitimes [34]. La stratégie des DDoS consiste à recruter plusieurs machines agents (esclaves). Ces machines sont généralement utilisées pour envoyer les paquets d'attaque afin d'inonder la cible. L'attaquant exploite automatiquement les machines vulnérables en utilisant des outils spéciaux pour cela, puis il utilisera ces machines infectées pour recruter de nouveaux agents à l'avenir.

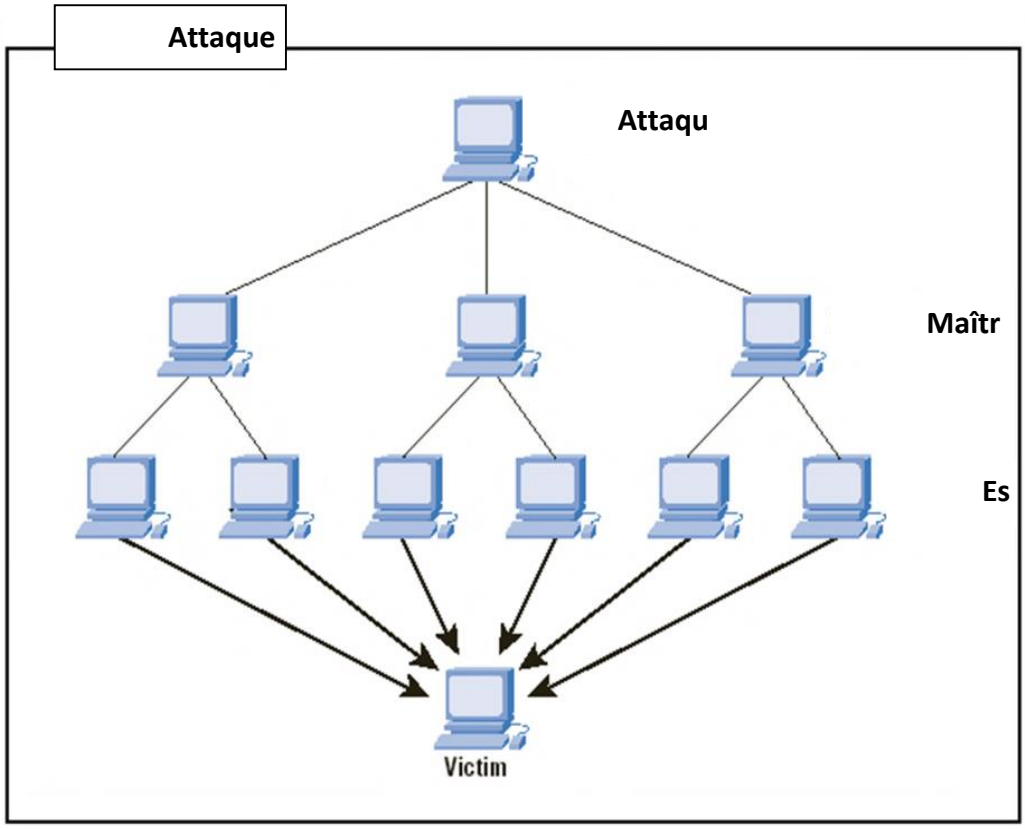


Figure I 19 Exemple d'architecture d'une attaque DDoS

1.10.3 Principe de l'attaque DDoS

Un attaquant établit via des techniques de compromission diverses, un réseau d'hôtesinfectés (par des virus, chevaux de Troie ou vers), appelé réseau de zombies. Ces derniers sont contrôlés à distance par l'attaquant.

Les ordinateurs zombies analysent et infectent constamment plus d'hôtes et créent ainsi plus de zombies. L'attaquant communique avec les hôtes (agents) via des connexions sur des ports spécifiques ou en utilisant les mécanismes de Covert Channeling tel que LOK. Cela lui permet de coordonner et synchroniser les agents (zombies) de manière à ce que l'attaque soit initiée de manière simultanée et dirigée par le biais du réseau de zombies vers la cible (hôte, serveur, réseau), ce qui augmente les chances de réussite (pour l'attaquant) ,

1.10.4 Catégories de cibles d'attaques DDoS

Bande passante : consiste à saturer la capacité réseau d'un serveur, le rendant ainsi

injoignable

Ressources : catégorie d'attaque consistant à épuiser les ressources système de la machinel'empêchant ainsi de répondre aux requêtes légitimes

Exploitation de la faille logicielle « Exploit » : ciblant une faille logicielle particulièreafin de rendre la machine indisponible, ou d'en prendre le contrôle.

1.10.5 Les principaux types d'attaques DoS et DDoS

Nom de l'attaque	Niveau OSI	Type d'attaque	Explications du principe de l'attaque
ICMP Echo Request Flood	L3	Ressources	Aussi appelé Ping Flood, envoi massif de paquets (ping) impliquant la réponse de la victime (pong) ayant le même contenu que lepaquet d'origine.
IP Packet Fragment Attack	L3	Ressources	Envoi de paquets IP référençant volontairementd'autres paquets qui ne seront jamais envoyés, saturant ainsi la mémoire de la victime.
SMURF	L3	Bande Passant e	Attaque ICMP en broadcast usurpant l'adressesource pour rediriger les multiples réponses vers la victime
IGMP Flood	L3	Ressources	Envoi massif de paquets IGMP (protocole de gestion du multicast)
Ping of Death	L3	Exploit	Envoi de paquets ICMP exploitant un bogue d'implémentation dans certains systèmes d'exploitation
TCP SYN Flood	L4	Ressources	Envoi massif de demandes de connexion TCP
TCP Spoofed SYN Flood	L4	Ressources	Envoi massif de demandes de connexion TCP en usurpant l'adresse source
TCP SYN ACK Reflection Flood	L4	Bande passant e	Envoi massif de demandes de connexion TCP vers un grand nombre de machines, en usurpantl'adresse source par l'adresse de la victime. La bande passante de la victime sera saturée par lesréponses à ces requêtes.
TCP ACK Flood	L4	Ressources	Envoi massif d'accusés de réception desegments TCP
TCP Fragmented Attack	L4	Ressources	Envoi de segments TCP référençant volontairement d'autres segments qui ne

			seront

			jamais envoyés, saturant ainsi la mémoire de lavictime
UDP Flood	L4	Bande Passant e	Envoi massif de paquets UDP (ne nécessitant pas d'établissement de connexion préalable)
UDP Fragment Flood	L4	Ressources	Envoi de datagrammes UDP référençant volontairement d'autres datagrammes qui neseront jamais envoyés, saturant ainsi la mémoire de la victime
Distributed DNS Amplification Attack	L7	Bande Passant e	Envoi massif de requêtes DNS usurpant l'adresse source de la victime, vers un grand nombre de serveurs DNS légitimes. La réponse étant plus volumineuse que la question, s'ensuitune amplification de l'attaque
DNS Flood	L7	Ressources	Attaque d'un serveur DNS par l'envoi massif de requêtes
HTTP(S) GET/POST Flood	L7	Ressources	Attaque d'un serveur web par l'envoi massif derequêtes
DDoS DNS	L7	Ressources	Attaque d'un serveur DNS par l'envoi massif de requêtes depuis un grand ensemble de machinescontrôlées par l'attaquant

Tableau 2 Principales catégories d'attaques DoS et DDoS

1.10.6 Exemples d'attaques DoS et DDoS

a ***L'attaque TCP SYN Flooding***

❖ ***Etablissement d'une session TCP :***

TCP est un protocole orienté connexion. L'établissement d'une session permet de s'assurer que les applications sont prêtes à recevoir les données et que toutes les données sont bien reçues par le destinataire [2].

Une connexion TCP s'établie en trois phases selon le mécanisme de poignée de main en trois temps (Three-ways handshake). Ces trois étapes sont l'envoi d'un SYN, la réception d'un SYN-ACK et l'envoi d'un ACK.

❖ **SYN Flooding :**

L'attaque SYN Flood est l'une des attaques les plus répandues qui exploite le mécanisme de poignée de main en trois temps (Three-ways handshake) du protocole TCP. Elle consiste en l'envoi d'un grand nombre de demandes de connexions au serveur cible (SYN) à partir de plusieurs machines et ne pas y répondre. Lors d'une demande de connexion, le serveur est en attente et bloque pendant un certain temps une partie de ses ressources pour cette nouvelle connexion. Le but est d'envoyer plus de demandes qu'il ne peut en traiter dans un temps donné. Ainsi, le serveur gaspille toutes ses ressources réseau à répondre à des requêtes qui ne mènent nulle part et il ne pourra plus subvenir aux besoins de vrais clients [2][18].

b ***L'attaque Smurf***

L'attaque Smurf s'appuie sur le ping (**ping** est un outil exploitant le protocole ICMP, permettant de tester les connexions sur un réseau en envoyant un paquet et en attendant la réponse) et les serveurs de diffusion (broadcast) pour paralyser le réseau (Un serveur broadcast est un serveur capable de dupliquer un message et de l'envoyer à toutes les machines présentes sur le même réseau).

On falsifie d'abord l'adresse IP source pour se faire passer pour la machine cible. On envoie alors un ping sur un serveur de broadcast. Il le fera suivre à toutes les machines qui sont connectées qui renverront chacune un pong au serveur qui fera suivre à la machine cible [24].

c ***L'attaque UDP Flood***

L'attaque UDP Flood consiste en l'envoi d'une grande quantité de paquets UDP sur des ports aléatoires de la machine victime. Ainsi, celle-ci sera obligée de répondre par l'envoi de nombreux paquets ICMP, la rendant inaccessible par d'autres clients. Ce qui conduit, au bout d'un certain temps à une congestion du réseau ainsi qu'à une saturation des ressources de la machine cible. Cette congestion est généralement plus importante qu'avec le TCP Flood car UDP ne possède pas de mécanisme de contrôle

de congestion (time out), donc la totalité de la bande passante peut être saturée (effondrement de la totalité du réseau).

Comme solution pour contrer cette attaque : bloquer les paquets d'attaque en utilisant des règles de filtrage (Firewall) [58].

d ***L'attaque ARP***

L'attaque ARP est l'une des attaques les plus célèbres, elle consiste à exploiter une faiblesse du protocole ARP. Son principe consiste à s'interposer entre deux machines du réseau et de transmettre à chacune un paquet ARP falsifié indiquant que l'adresse ARP (adresse MAC) de l'autre machine a changé, l'adresse fournie étant celle de l'attaquant.

Les deux machines cibles vont ainsi mettre à jour leurs tables dynamiques appelées cache ARP. On parle ainsi « ARP cache poisoning » ou « ARP spoofing » ou encore « ARP redirect » pour désigner ce type d'attaque. De cette manière, à chaque fois qu'une des deux machines souhaitera communiquer avec la machine distante, les paquets seront envoyés à l'attaquant qui les transmettra de manière transparente à la machine destinatrice [58].

1.10.7 **Les outils d'attaques DDoS**

Le tableau **3** décrit les outils les plus utilisés pour déclencher (lancer) des attaques DDoS

Outil	**But**
TFN	Architecture client/agent Protocole de communication : ICMP Attaque sur les protocoles : IP/TCP/UDP/ICMP Les agents TFN sous le contrôle du TFN client, réalisent l'attaque distribuée sur la victime/cible et avec le type d'attaque désiré par le client. Les adresses du client et des agents sont usurpées (falsifiées)

TFN2K	Version évoluée de TFN Communications entre le client et ses agents sont chiffrées : ce quile rend plus dur à détecter Protocoles de communication : TCP/UDP/ICMP
Trin00	Architecture à trois couches : un client qui envoie des commandes à des serveurs maîtres qui se chargent chacun d'un sous réseau d'agents. Attaque sur le protocole : UDP

Tableau 3 Exemple d'outils d'attaques DDoS

1.10.8 **Impact potentiel des attaques DDoS**

L'impact potentiel des attaques DDoS se résume en des points suivants :

- Perturber voire immobiliser totalement les services
- Perte de visibilité sur les ressources
- Ces attaques peuvent aussi bien servir des intentions criminelles que d'un simple acte de malveillance destiné à causer du tort à la cible
- Leur rayon d'action s'élargit, ciblant tous types d'organisations et tous secteurs d'activités en proie à des attaques DDoS
- Risque de perte de compétitivité (cas d'indisponibilité de service)
- Impact sur la confiance des investisseurs, des clients et des collaborateurs [18].

1.10.9 **Prévention des attaques DDoS**

Dans le cas d'une attaque DDoS cela se complique énormément puisque les adresses IP sources sont nombreuses et variées.

En effet, les attaques DDoS sont relativement compliquées à arrêter une fois que les hôtes (agents) ont commencé à attaquer la cible.

Le meilleur moyen de les prévenir consiste à les détecter avec les IDS pendant la phase préliminaire, lorsque l'attaquant communique avec les agents (hôtes).

1.11 Conclusion

En tant que technologie de réseau émergente, le SDN a un bel avenir, la sécurité étant l'un des problèmes les plus urgents. Malheureusement, les attaques DDoS sont devenues plus sophistiquées au fil du temps, et chaque année, des attaques à grande échelle avec une fréquence et une taille croissantes sont lancées contre d'énormes entreprises, des centres de données et d'autres entreprises [9].

Dans ce chapitre, nous avons présenté les notions de sécurité réseau, les réseaux SDN et les attaques DDoS. Une définition du réseau SDN a été fournie, son architecture, ses contrôleurs, ses avantages et ses défis. Nous avons également fait une brève comparaison entre les réseaux traditionnels et les réseaux SDN. Enfin, nous avons discuté des attaques DDoS et de leurs effets, des botnets et de la taxonomie, contenant plusieurs types d'attaques DDoS.

Chapitre 2 Attaque DDOS dans les réseaux SDN et approche ML

2.1 Introduction

Le principal avantage du SDN est sa capacité à découpler les plans de données et de contrôleur, offrant aux développeurs plus de flexibilité et facilitant la maintenance des applications dans ces situations. De plus, il a pu séparer l'infrastructure réseau de ses applications pour découpler le plan de contrôle et être logiquement centralisé. En conséquence, le logiciel peut créer des services désormais implémentés dans le matériel (IDS, pare-feu et routeurs). Cette centralisation et ce découplage du SDN sont possibles car le contrôleur de réseau et les commutateurs disposent d'interfaces programmables bien définies via des API qui leur permettent de communiquer entre eux.

Malgré le potentiel d'innovation introduit par le SDN, il introduit de nouveaux défis pour la sécurité du réseau. Les qualités essentielles d'un réseau de communication sécurisé, telles que le secret, l'intégrité, la disponibilité des informations, l'authentification et la non-répudiation, peuvent être compromises par ces problèmes. Un nombre important de paquets sont transmis à un hôte connecté ou à un ensemble de serveurs. La collecte de paquets DDoS authentiques et contrefaits peut épuiser la capacité de traitement du contrôleur. Cela rendra le contrôleur inaccessible aux paquets de nouveaux arrivants légitimes, détruisant potentiellement l'architecture SDN. Étant donné que les intrus peuvent se propager et se situer sur plusieurs commutateurs, les attaques DDoS peuvent être difficiles à détecter. Étant donné que les commutateurs ne peuvent pas détecter

complètement les attaques, le processus de détection peut ne pas améliorer le processus de détection. Les attaques DDoS, telles que l'inondation des communications contrôleur-commutateur, peuvent avoir un impact significatif sur un réseau SDN.

Les dispositifs tels que les contrôleurs, les capteurs et les substrats de communication dans les infrastructures critiques pourraient exacerber ces problèmes dans les scénarios IoT [39]. Cependant, ces dernières années, certaines solutions aux problèmes de sécurité du SDN ont été proposées dans la littérature [39]. Les algorithmes d'apprentissage automatique sont un moyen qui peut être utilisé pour relever ces défis. Dans ce chapitre, nous présenterons une description des mécanismes de défense contre les attaques DDoS dans les réseaux SDN.

L'objectif du SDN est de fournir des interfaces ouvertes pour le développement de logiciels capables de contrôler la connectivité fournie par un ensemble de ressources réseau et le flux de trafic réseau qui les traverse, ainsi que l'éventuelle inspection et modification du trafic dans le réseau.

Comme le montre la Figure II.1 , l'architecture SDN adoptée se compose de trois niveaux.

L'adoption de cette architecture en couches offre de nombreux avantages mais aussi divers inconvénients en termes de sécurité. En bref, les problèmes de sécurité avec le SDN peuvent être trouvés à trois niveaux : le niveau de communication entre les différents éléments architecturaux, le niveau des composants et le niveau d'audit et de journalisation.

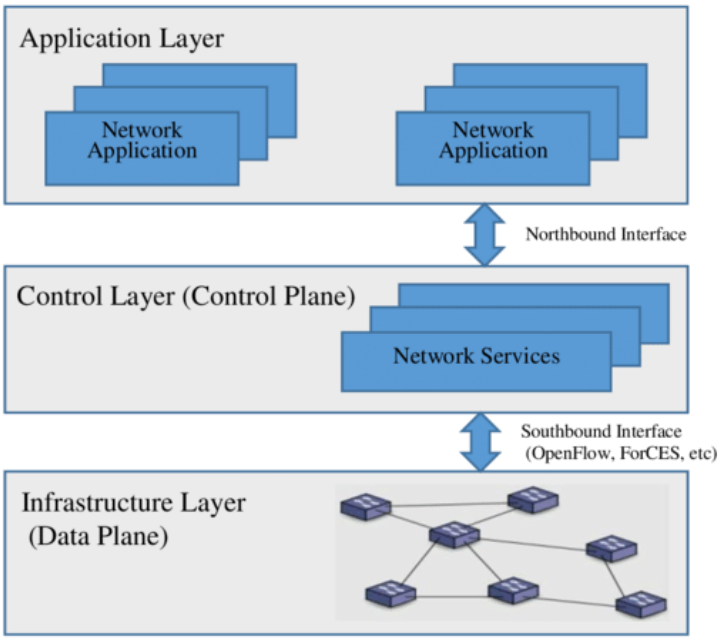


Figure II 1 : Architecture et composants du SDN

2.2 Approche d'apprentissage automatique

Nous appliquons des approches d'apprentissage automatique pour analyser les performances du système et détecter les occurrences étranges qui ne sont pas compatibles avec le comportement typique du réseau. Les actions anormales sont reconnues par des modèles mathématiques construits à l'aide de techniques d'apprentissage automatique, en particulier dans les systèmes de réseau où les données à haute densité circulent et appliquent rapidement des politiques de prévention. Les caractéristiques des approches d'apprentissage automatique utilisées dans ce travail sont brièvement décrites dans cette section.

2.3 Notion de l'intelligence artificielle

Le terme intelligence artificielle est né en 1955 et est souvent attribué à John McCarthy,professeur de mathématiques au Dartmouth Collège. McCarthy a nié le mérite de ce terme, maisil est associé à ses origines car il a organisé un atelier d'été en 1956 qui a lancé une recherche organisée sur l'IA. Il a défini l'IA comme :"La science et l'ingénierie de la fabrication de machines intelligentes" [40].

Nous pouvons définir que l'intelligence artificielle est la science qui permet auxmachines de penser et de prendre des décisions comme les humains.

2.4 Types de l'intelligence artificielle

L'intelligence artificielle peut être structurée en trois niveaux évolutifs (voir la figure II.2 qui sont [41] : L'intelligence artificielle faible (Artificial Narrow Intelligence), l'intelligence artificielle générale (Artificial General Intelligence) et super intelligence artificielle (Artificial Super Intelligence).

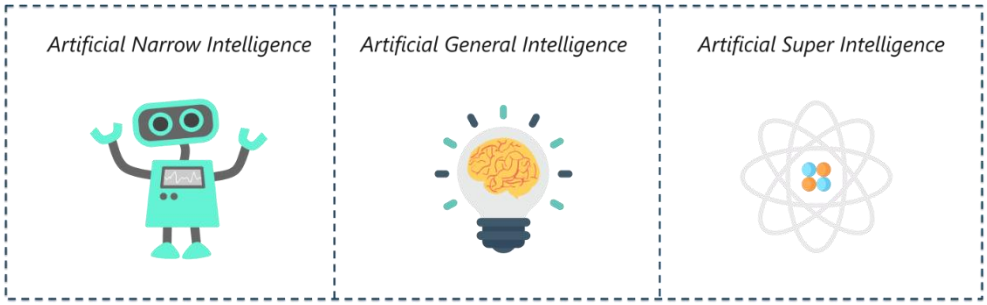


Figure II 2 Niveaux de l'intelligence artificielle.

- **Artificial Narrow Intelligence (ANI)** : Appelée aussi IA faible, est appliqué uniquement dans des tâches spécifiques. La plupart des systèmes actuels basés sur l'IA fonctionnent en fait comme une IA faible. Le moteur de recherche Google, Sophia et les voitures autonomes entrent dans la catégorie des IA faible.
- **Artificial General Intelligence (AGI)** : appelée aussi IA forte, implique des machines qui possèdent la capacité d'effectuer n'importe quelle tâche intellectuelle qu'un être humain peut le faire.
- **Artificial Super Intelligence (ASI)** : c'est un terme se référant à l'époque où la capacitédes ordinateurs dépassera les humains.

Nous n'avons actuellement atteint qu'une IA étroite ou faible (ANI). Alors que les capacités d'apprentissage automatique continuent d'évoluer et que les scientifiques se rapprochent de l'IA générale , des théories et des spéculations concernant l'avenir de l'IA circulent entre qui sont basés sur la peur d'un avenir dystopique, où des robots super intelligent sasservissant toute l'humanité , comme le décrivent de nombreux récits de science-fiction et entrequi sont plus optimistes

de l'utilisation de l'IA où les humains et les robots travailleront ensemble, les humains utilisant l'intelligence artificielle comme un outil pour améliorer leur expérience de vie.

Les outils d'intelligence artificielle ont déjà un impact significatif sur la façon dont nousmenons nos activités dans le monde entier, accomplissant des tâches avec une vitesse et une efficacité qui ne seraient pas possibles pour les humains. Cependant, l'émotion humaine et la créativité sont quelque chose d'incroyablement spécial et unique, extrêmement difficile - voireimpossible - à reproduire dans une machine.

2.5 Différence entre l'IA et ML et DL

L'intelligence artificielle est alimentée par des algorithmes, utilisant des techniques telles que l'apprentissage automatique ML, l'apprentissage profond DL et les règles. L'IA, MLet DL sont des domaines interconnectés comme illustré dans la figure II.3.

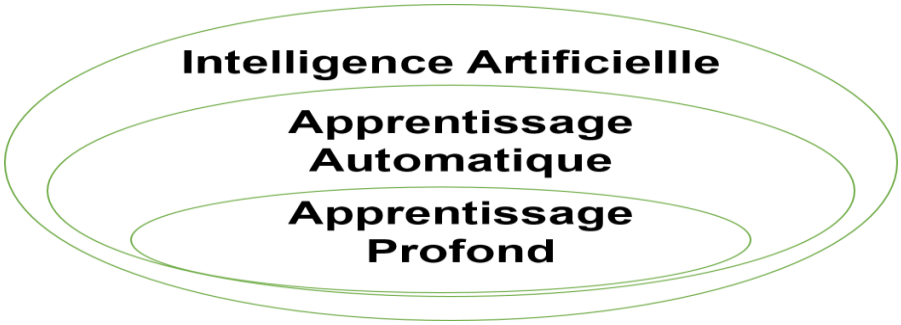


Figure II 3 Relation entre IA et ML et DL.

2.5.1 Apprentissage automatique (ML)

Le terme apprentissage automatique a été inventé pour la première fois par Arthur Samuel en 1959 qui a dit « ML est le domaine d'études qui donne aux ordinateurs la capacité d'apprendre sans être explicitement programmé » [42]. Dans cette partie, nous définissons le concept de l'apprentissage automatique et nous présentons ses types, son processus ainsi que ses limites.

a ***Définition de l'apprentissage automatique***

L'apprentissage automatique est un sous-ensemble de l'intelligence artificielle (IA) quioffre aux machines la possibilité d'apprendre automatiquement en lui fournissant des tonnes dedonnées et en lui permettant de s'améliorer grâce à l'expérience. Ainsi, l'apprentissage automatique est une pratique consistant à amener les machines à résoudre des problèmes en acquérant la capacité de penser [40].

b ***Types de l'apprentissage automatique***

Il y a trois principales catégories de l'apprentissage automatiques ; apprentissage supervisé (supervised learning) et apprentissage non supervisé (unsupervised learning) et apprentissage renforcé (Reinforcement learning). Pour bien comprendre la différence, reprenons un exemple : On suppose qu'on a une nouvelle base de photos àcatégoriser. On dispose de données d'exemple (training set) préalables pour entraîner notre modèle.

Il existe différentes façons de ML dont les machines peuvent apprendre. Une machine peut apprendre à résoudre un problème en suivant l'une des trois approches suivantes :

▪ **Apprentissage supervisé :** L'apprentissage supervisé est une technique dans laquelle on enseigne ou bien former la machine à l'aide de données bien étiquetées.

▪ **Apprentissage non supervisé :** implique une formation en utilisant des données non étiquetées et en permettant au modèle d'agir sur ces informations sans guide.

▪ **Apprentissage par renforcement :** fait partie de l'apprentissage automatique où un agent (objet connecté) est placé dans un environnement et il apprend à se comporter dans cet environnement en effectuant certaines actions et en observant les récompenses qu'il obtient de ces actions

❖ **En apprentissage supervisé**, on va récupérer des données dites

étiquetées ou annotéesde leurs sorties, c'est-à-dire qu'on leur a déjà associé un **label** ou une classe **cible** pour entraînerle modèle et on veut que l'algorithme devienne capable, une fois entraîné, de prédire cette cible sur de nouvelles données non annotées. L'apprentissage supervisé comporte deux familles ; **classification** et **régression**. On parle de classification si les sorties sont des catégoriesdiscrètes, ou de régression si la sortie est une fonction continuée [43].

- **En apprentissage non supervisé**, dit regroupement (**clustering)** les données d'entréesne sont pas annotées. Dans ce cas, l'algorithme d'entraînement s'applique à trouver seul les similarités et distinctions au sein de ces données, et à regrouper ensemble celles qui partagent des caractéristiques communes [43].

La figure II.4 illustre un exemple de l'apprentissage supervisé et non supervisé.

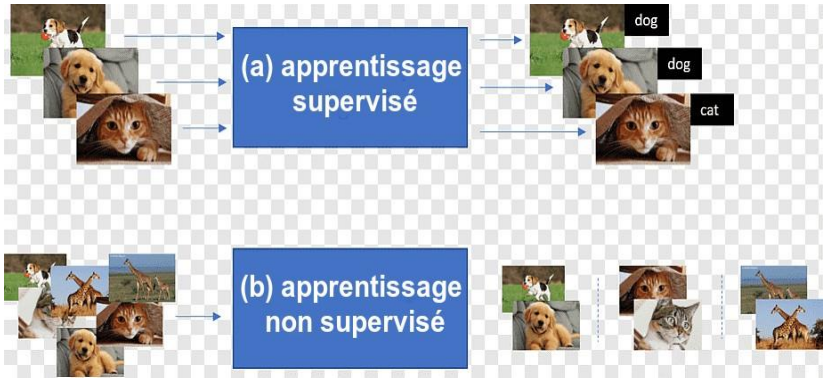


Figure II 4 Apprentissage supervisé et non supervisé

La figure II.5 illustre les principales différences entre l'apprentissage supervisé (Classification et Régression) et l'apprentissage non supervisé (clustering).

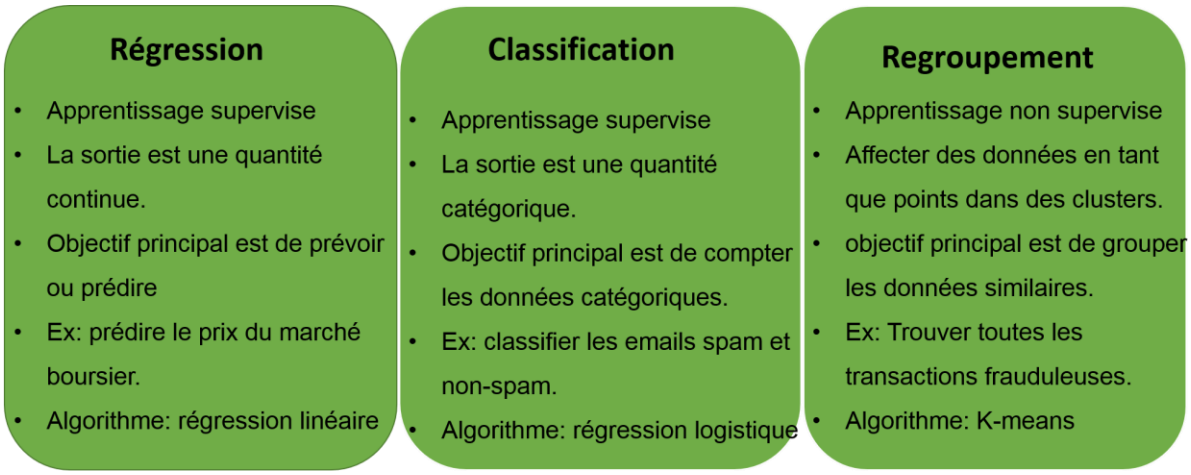


Figure II 5 Différence entre apprentissage supervisé et non supervisé

Il existe ainsi une autre catégorie ; **Apprentissage semi-supervisé** (semi-supervised learning) où on a une connaissance a priori de la classification de certains éléments (supervisé),qui aide à mieux converger des algorithmes non-supervisés [44]

La figure II.6 illustre les différents algorithmes ML répartis sur trois familles ; apprentissage supervisé, apprentissage non supervisé et apprentissage renforcé.

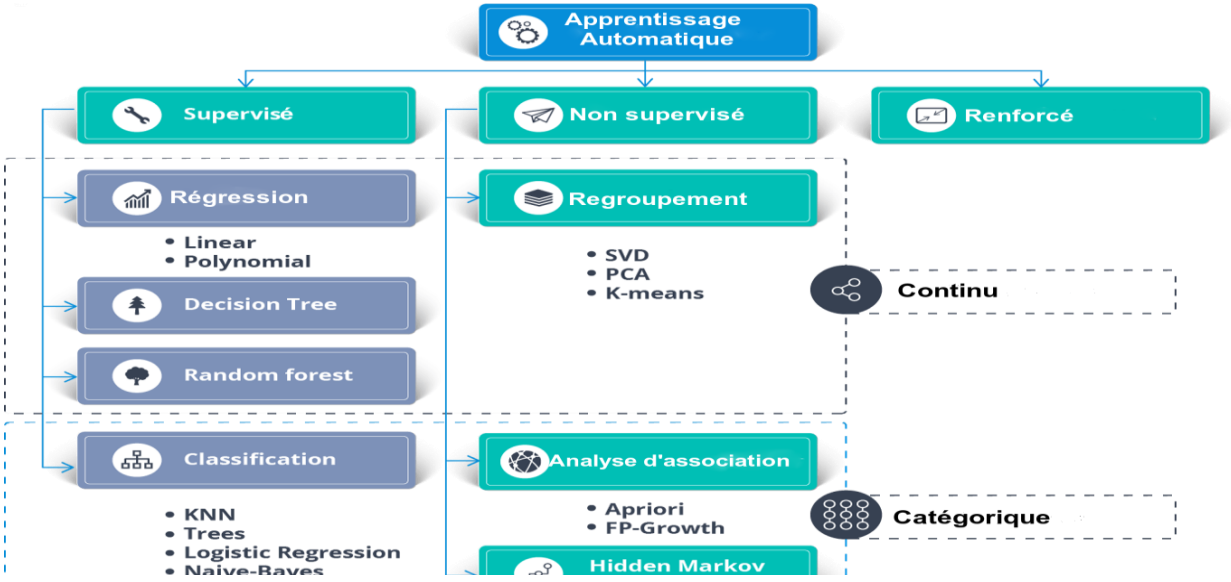


Figure II 6 Différents algorithmes de l'apprentissage automatique

c ***Processus de l'apprentissage automatique***

Le processus d'apprentissage automatique implique la construction d'un modèle prédictif qui peut être utilisé pour trouver une solution pour un problème.

La figure II.7 schématise les différentes étapes de processus ML.

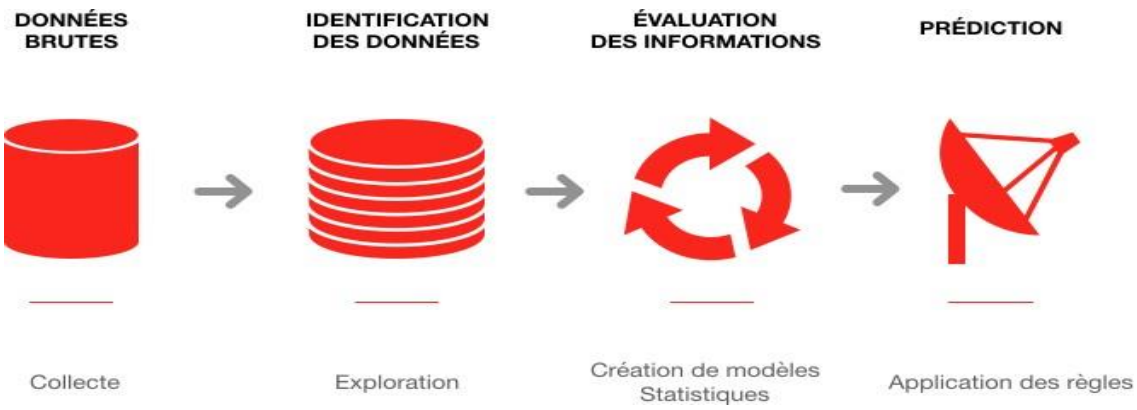


Figure II 7 Étapes de processus de l'apprentissage automatique

d ***Limitation de l'apprentissage automatique***

Malgré les performances de ML, il existe des limites à savoir :

- Machine Learning n'a pas la capacité de gérer et de traiter **des données de grandes dimensions.**
- Un autre défi majeur dans le Machine Learning est de dire à la machine quelles sont

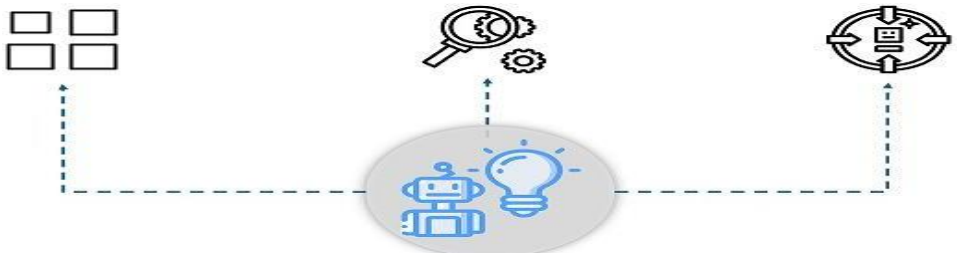


Figure II 8 Limitations de l'apprentissage automatique

les fonctionnalités importantes qu'elle doit rechercher afin de prédire avec précisionle résultat. Ce processus même est appelé **features extraction,** reste un processus manuel dans Machine Learning.

Les limitations ci-dessus peuvent être résolues en utilisant le Deep Learning. Ce derniersera présenté dans les sections suivantes.

Les limitations ci-dessus peuvent être résolues en utilisant le Deep Learning. Ce derniersera présenté dans les sections suivantes.

2.5.2 Apprentissage profond (DL)

L'apprentissage profond (DL ; Deep learning) est un sous-type de l'apprentissage automatique. DL peut traiter une plus large gamme de ressources de données, nécessite moins de prétraitement des données par les humains (par exemple l'extraction des caractéristiques), etpeut parfois produire des résultats plus précis que les approches ML traditionnelles (bien qu'il nécessite une plus grande quantité de données pour le faire). Dans cette partie, nous nous intéressons au concept de DL et ses algorithmes.

a ***Définition de l'apprentissage profond***

« Le Deep Learning est un ensemble de techniques d'apprentissage automatique statistique utilisées pour apprendre des hiérarchies de fonctionnalités basées sur le concept de réseaux neurones artificiels ».

Le DL utilise des algorithmes similaires à ceux du ML pour entraîner **les réseaux neurones profonds** afin d'obtenir une meilleure performance en matière de précision. Fondamentalement, l'apprentissage en profondeur **imite le fonctionnement denotre cerveau,** c'est-à-dire qu'il apprend par expérience.

2.6 Ensemble de données (Dataset)

Un ensemble de données est une collection d'observations connexes organisées et formatées dans un but particulier [45]. Les ensembles de données ont joué un rôle fondamental dans l'avancement de la recherche sur l'apprentissage automatique. Ils constituent la base des modèles que nous concevons et déployons, ainsi que notre principal moyen d'analyse comparative et d'évaluation [46].

2.6.1 Types de dataset :

❖ **real dataset :**

il représente les données générées à partir d'événements mondiaux réels, c'est-à-dire des données provenant d'un système de production, d'un fournisseur, d'archives publiques ou de tout autre jeu de données contenant des données opérationnelles. Par exemple, un ensemble de données qui est une sauvegarde vieille de dix ans d'un système existant et qui contient des données sur de vrais individus, sujets ou cas, serait des données réelles [47].

❖ **dataset synthétiques :**

les données synthétiques sont générées à partir de données réelles en utilisant les propriétés statistiques sous-jacentes des données réelles pour produire des ensembles de données synthétiques qui présentent ces mêmes propriétés statistiques [48]. Un excellent ensemble de données synthétiques devrait remplacer les valeurs sensibles et fournir des garanties plus solides de confidentialité et d'anonymat. Les données synthétiques peuvent être utilisées de deux manières :

– Pour augmenter la taille d'un jeu de données, pour les moments où un jeu de données est déséquilibré en raison de l'occurrence limitée d'un événement.

– Pour générer un jeu de données synthétique complet représentant le jeu de données d'origine, pour les moments où les données ne sont pas disponibles en raison de leur nature sensible.

2.7 Prétraitement des données (Data Preprocessing)

Cette phase est essentielle pour atteindre la qualité des données. Ainsi, la réduction des données est faite en sélectionnant les attributs les plus importants sans perte de qualité [45].

2.7.1 Sélection des fonctionnalités

Avant d'utiliser un algorithme d'apprentissage, la sélection des caractéristiques est souvent une étape essentielle du traitement des données. Par exemple, les algorithmes d'apprentissage automatique améliorent fréquemment leurs performances en supprimant les données non pertinentes et en double [49]. La sélection des fonctionnalités est le processus par lequel un scientifique des données sélectionne un sous-ensemble de caractéristiques pertinentes à utiliser dans le développement de modèles d'apprentissage automatique, soit automatiquement, soit manuellement. C'est en effet l'un des principes essentiels du machine learning qui a un impact significatif sur les performances de vos modèles puisqu'il est la clé pour développer des modèles de machine learning fiables. La procédure choisira le sous-ensemble optimal d'attributs à partir d'un pool de caractéristiques les plus importantes et aura une contribution élevée au moment de la prédiction.

2.8 Problèmes de sécurité du SDN

2.8.1 Niveau Communication

1. Au niveau des communications vers le nord, les principaux problèmes de sécurité sont :

• L'absence de confiance et une authentification faible entre les applications et le contrôleur peuvent usurper les messages d'API en direction nord.

• Une autorisation inappropriée sur les applications peut déclencher un accès inapproprié ou malveillant.

2. Au niveau des communications en direction sud :

• L'absence de cryptage entre le contrôleur et les commutateurs permet l'écoute clandestine et l'usurpation des communications vers le sud.

• Le manque de confiance et une authentification faible à ce niveau peuvent conduire à des attaques de type "man-in-the-middle" et à des attaques

d'usurpation d'identité, ce qui permet à un attaquant d'écouter plus facilement le flux pour voir le flux utilisé et le réseau qui autorise le trafic à travers ce flux

• Une autorisation incorrecte peut entraîner un accès non autorisé. Supposons qu'un utilisateur demande à ce que le contrôleur crée une route et fasse traverser à un paquet le service routé. Dans ce cas, nous devons nous assurer que l'utilisateur est autorisé à le faire et que l'environnement peut résoudre l'activité.

• La communication vers le sud est affectée par les attaques de règle de flux, principalement dans les déploiements intrabande.

2.8.2 Chaque niveau de composant

Chaque composant de l'architecture SDN doit être protégé.

1. Le composant applicatif

• Si les applications ne sont pas vérifiées comme provenant de sources fiables, nous risquons d'avoir des applications vulnérables ou malveillantes qui créent des vulnérabilités majeures et, par conséquent, compromettent l'ensemble du réseau SDN.

• L'API elle-même peut être vulnérable. Si l'attaquant peut utiliser l'API vulnérable, il peut contrôler le réseau SDN via le contrôleur. Si le contrôleur ne dispose d'aucune forme de sécurité de l'API nord, l'attaquant peut créer sa stratégie SDN pour prendre le contrôle de l'environnement SDN.

• Des règles de flux incohérentes peuvent résulter d'un manque d'isolation des applications. L'isolation de flux utilise des balises pour distinguer les paquets appartenant à différentes politiques, permettant des mises à jour réseau cohérentes et, par conséquent, des règles de flux cohérentes.

2. Le composant Contrôleur

• Le contrôleur est vulnérable aux attaques par déni de service, compromettant l'ensemble de l'infrastructure réseau. Les paquets usurpés utilisés

dans les différents types de messages utilisés par OpenFlow ou OpFlex (prise de contact, bonjour et autres), ainsi qu'une mauvaise logique dans le code du contrôleur, pourraient entraîner l'évitement des périphériques réseau légitimes et potentiellement provoquer un déni de service (DoS) problèmes.

• L'implémentation par l'attaquant de contrôleurs escrocs pourrait entraîner des entrées dans les tables de flux des éléments de réseau. En conséquence, l'attaquant aurait le contrôle complet du réseau.

• Le détournement de contrôleur se produit lorsqu'un pirate prend le contrôle du contrôleur et obtient le contrôle complet du réseau SDN, des flux et des politiques définies.

3. Le composant Switch

• Au niveau de la table de flux, la possibilité d'une attaque DoS sur les commutateurs existe ; l'attaquant peut inonder le commutateur avec des règles de flux massives.

• La possibilité d'attaques diverses sur l'agent installé sur un commutateur (par exemple, débordement de tampon, déni de service, etc.).

• L'instabilité et l'indisponibilité du réseau peuvent être causées par des incohérences dans les règles de flux.

• Lorsque le mode d'écoute du commutateur est activé, il permet une connexion entre le contrôleur et le commutateur sans authentification ni contrôle d'accès intégrés.

2.8.3 Journalisation et niveau d'audit

Le composant d'audit et de conformité a été identifié comme un problème de sécurité majeur dans le SDN. L'absence de journaux et de pistes d'audit peut entraîner une perte de contrôle sur le réseau SDN, sans surveillance ni vérification des modifications non autorisées ou des tentatives malveillantes de la part des administrateurs ou des attaquants.

2.9 Déjouer les attaques DDoS dans les réseaux basés sur SDN

Les attaques DDoS augmentent en taille, en fréquence, en gravité et en sophistication dans le réseau traditionnel. Cela signifie que les attaques DDoS avec les IDS actuels se limitent à résoudre le problème. Les attaquants ont développé des méthodes plus avancées pour contourner les défenses existantes. À la place, le SDN présente certaines fonctionnalités qui présentent des avantages dans une certaine mesure pour surmonter les attaques DDoS.

- **Séparation du plan de contrôle et du plan de données :** L'expérimentation dans les réseaux traditionnels est compliquée. De plus, étant donné que la logique de contrôle est intégrée dans les appareils, il est trop difficile d'examiner les algorithmes nouvellement utilisés dans les réseaux traditionnels. Chaque appareil doit être mis à jour individuellement. Le SDN, cependant, sépare deux plans ci-dessous et simplifie le test de mécanismes complets. SDN a une grande fonctionnalité de configuration dynamique, qui permet des environnements expérimentaux.

- **Global View Network :** le contrôleur surveille le trafic réseau et dispose d'une vue globale du réseau. La centralisation du contrôleur SDN permet d'isoler facilement l'hôte compromis de l'hôte légitime en utilisant les informations obtenues en demandant aux hôtes finaux [50].

- **Analyse du trafic basée sur le logiciel** : pour surveiller et configurer les périphériques réseau, divers utilitaires d'application s'exécutent dans le plan d'application SDN. De nombreux logiciels et algorithmes sont disponibles pour analyser le trafic réseau, ce qui réduit la charge des commutateurs pour analyser le trafic [50].

- **Programmabilité du réseau :** le plan d'application contient des applications qui programment le contrôleur et contrôlent davantage le comportement du réseau. La programmabilité du réseau SDN le rend plus flexible car plus d'intelligence peut être déployée à tout moment [51]. De plus, comme le

réseau est programmable, le trafic entrant peut être analysé pour identifier le trafic ou les hôtes malveillants et maintenir les performances du réseau.

• **Mise à jour dynamique de la politique réseau :** la réponse immédiate dans l'atténuation des attaques DDoS est l'ajustement dynamique des règles de flux sur les commutateurs OpenFlow. Sur la base de l'analyse du trafic, de nouveaux algorithmes innovants pour bloquer le trafic peuvent être propagés instantanément [52]. L'ajout d'une nouvelle règle à chaque appareil est brutal dans un réseau traditionnel, mais la mise à jour dynamique des commutateurs est simple avec le SDN.

2.10 Types d'attaques DDoS dans le SDN

L'architecture réseau SDN est un nouveau concept qui dissocie le plan de données du plan de contrôle pour améliorer la gérabilité et la sécurité du réseau. La séparation du plan de contrôle du plan de données, la vue globale du réseau, la programmabilité du réseau, l'analyse du trafic basée sur le logiciel et la mise à jour dynamique des politiques de réseau sont quelques caractéristiques inhérentes au SDN pour améliorer la sécurité du réseau [13]. Néanmoins, diverses attaques infectent le plan de données, le plan de contrôle et les connexions planes à plan. Ces attaques détruisent l'architecture SDN et en font une architecture réseau problématique.

Dans l'architecture SDN, la figure II.21 montre quel avion est vulnérable à une attaque spécifique. Thé

Voici quelques exemples de telles attaques.

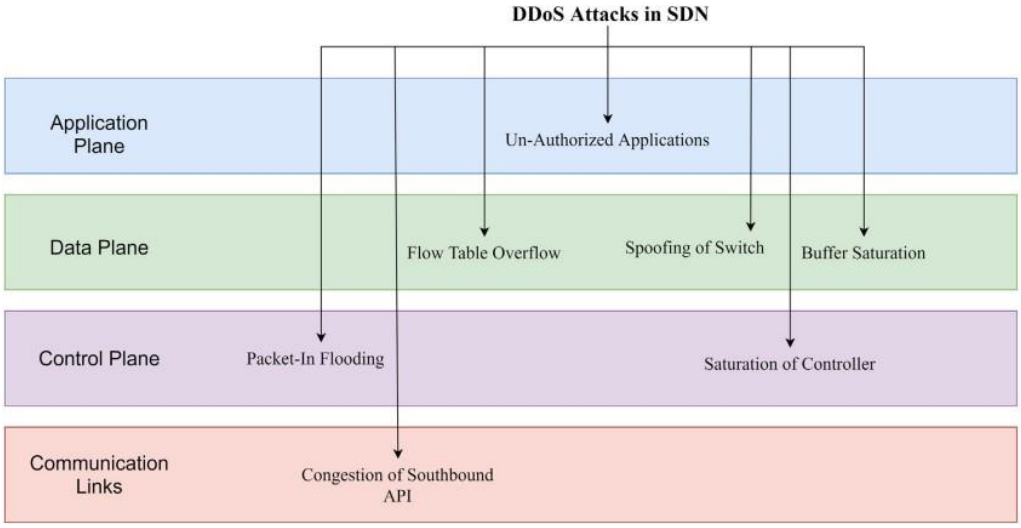


Figure II 10 Types d'attaques DDoS à différentes couches du SDN

2.10.1 Attaques DDoS de la couche application

Les attaques de la couche application utilisent le logiciel de manière malveillante, visant à épuiser les ressources pour traiter toute autre demande. Ces attaques sont généralement plus difficiles à détecter au niveau du réseau car elles ne montrent aucun écart clair par rapport au trafic légitime [33]. Les attaques DDoS sur une application auront un impact sur d'autres applications car la séparation des applications ou des ressources dans le SDN n'est pas bien résolue. Parmi les attaques de cette couche :

• **Applications non autorisées :** dans le plan d'application, de nombreuses applications ont accès aux ressources réseau pour fournir des services de contrôleur et de réseau. Certaines applications peuvent accéder aux ressources du réseau à l'exemple d'autres applications. Cependant, la validité des applications n'est ni authentifiée ni approuvée, et des applications malveillantes peuvent accéder à des instances d'autres applications et modifier le comportement du réseau et dégrader les performances du réseau.

2.10.2 Attaques DDoS de la couche de contrôle

Les contrôleurs de SDN et leurs communications peuvent être soumis à différents types d'attaques

[53]. Les attaques sur le plan de contrôle et la communication entre le contrôleur et d'autres composants du réseau, tels que l'API nord, l'API sud, l'API ouest ou l'API est, sont des exemples de menaces qui peuvent causer des dommages importants. De plus, le contrôleur peut être considéré comme un point unique de défaillance et d'évolutivité, ce qui soulève la possibilité de problèmes de performances et d'indisponibilité du plan de contrôle. Parmi les attaques de cette couche :

• **Saturation du contrôleur :** lorsque plusieurs paquets de requêtes (à cause de faux flux générés par l'attaquant) arrivent au contrôleur, le contrôleur crée des files d'attente pour gérer toutes ces requêtes. Cependant, supposons qu'il existe de nombreux faux paquets. Dans ce cas, le contrôleur restera occupé à traiter de fausses requêtes et ses performances finiront par se dégrader, ce qui constitue un obstacle pour les réseaux basés sur SDN .

• **Packet in flooding :** Chaque fois que le commutateur reçoit le flux sans correspondance, il demande (en utilisant le paquet dans le message) au contrôleur centralisé de dessiner une règle de transfert pour le nouveau flux via l'interface sud. L'attaquant envoie de nombreux paquets au vswitch en usurpant les adresses IP et force le vswitch à envoyer des paquets en masse dans des messages au contrôleur. En conséquence, cette inondation surcharge le contrôleur et le rend inaccessible aux utilisateurs légitimes car le contrôleur sera occupé à traiter uniquement les fausses requêtes de flux .

2.10.3 Attaques DDoS dans la couche de données

Les attaques DDoS de la couche de données pourraient potentiellement surcharger par deux points : les commutateurs ou en attaquant l'API vers le sud . Par exemple, un attaquant peut envoyer une grande quantité de trafic vers un nœud pour lancer une attaque DoS en établissant plusieurs flux nouveaux et inconnus dans la couche infrastructure. Parmi les attaques de cette couche :

• **Flow Table Overflow :** lorsque le commutateur OpenFlow demande la nouvelle règle de flux au contrôleur, la nouvelle règle envoyée par le contrôleur est stockée dans la table de flux du commutateur. Chaque règle de flux a une valeur de délai d'attente fixe, et après ce délai, les anciennes règles sont expulsées par le commutateur de la table de flux [4]. TCAM a une capacité très limitée pour stocker les entrées de la table de flux car il est très coûteux et gourmand en énergie . L'attaquant profite de cette fonctionnalité pour submerger le commutateur. L'attaquant envoie de nombreux nouveaux faux flux au commutateur et, par conséquent, la table de flux du commutateur manque rapidement de mémoire et ne contient que de fausses règles. Toutes les entrées valides sont supprimées de la table de flux, ce qui entraîne une dégradation des performances. [54]

• **Saturation de la mémoire tampon :** lorsqu'un commutateur envoie un paquet en message au contrôleur, il envoie une partie du paquet au contrôleur tout en stockant le reste dans la mémoire tampon. L'attaquant utilise cette fonctionnalité pour attaquer la victime ; l'attaquant envoie plusieurs faux paquets au commutateur, épuisant rapidement la mémoire tampon. De plus, lorsque la mémoire tampon interne du commutateur est épuisée, il doit envoyer le paquet entier aux contrôleurs dans le cadre de l'événement, encore un autre goulot d'étranglement pour le SDN. À ce stade, les utilisateurs légitimes ont des difficultés à traiter leurs demandes de flux et, par conséquent, l'attaquant peut réduire les performances.

• **Spoofing Switch :** Dans cette attaque, l'attaquant peut usurper l'adresse IP du commutateur et envoyer des messages de contrôle en utilisant l'adresse modifiée. Lorsqu'un commutateur crée une connexion avec le contrôleur et communique, un deuxième commutateur malveillant (un avec une adresse IP usurpée et le matériel et le nom identiques) est activé. Il démarre simultanément une connexion avec le contrôleur. Le contrôleur va mettre fin à la connexion avec un switch légitime et communiquer avec le malicieux, dégradant progressivement les performances du réseau [55]. Cette attaque peut amener le contrôleur à faire de fausses requêtes, et un mécanisme est nécessaire pour contrôler.

2.10.4 Liens de communication

entre chacune des deux couches, il y a des liens qui sont l'API vers le sud et l'API vers le nord, parmi les attaques dans ces liens :

- **Congestion des API Southbound :** lorsque le commutateur OpenFlow envoie un paquet dans une demande pour obtenir une nouvelle règle du contrôleur, il envoie une partie du paquet et une autre partie est stockée dans la mémoire tampon. Néanmoins, si le tampon est complet, alors le commutateur est susceptible d'envoyer le paquet entier au contrôleur. À ce stade, en envoyant plusieurs faux flux au commutateur, un attaquant peut facilement surcharger la bande passante unique utilisée dans les API sud et y créer une congestion pour la rendre indisponible pour les utilisateurs légitimes.

2.11 Détection des attaques DDoS dans le réseau SDN

Les attaques par déni de service distribué (DDoS) constituent une menace réelle dans de nombreux aspects des réseaux informatiques et des applications distribuées. L'objectif principal d'une attaque DDoS est de faire tomber les services d'une cible en utilisant plusieurs sources distribuées. Par exemple, les attaquants peuvent transférer des milliers de paquets à une victime pour submerger la bande passante d'accès avec un trafic illégitime, rendant les services en ligne indisponibles. De nombreuses méthodes d'attaque par déni de service (DoS) sont utilisées pour dégrader les performances ou la disponibilité de services ciblés sur Internet [33]. Ces méthodes peuvent être classées en tant que défis SDN à chaque couche : couche d'application, couche de contrôle et couche d'infrastructure.

2.11.1 Techniques de détection des attaques DDoS

Cette section fournit un examen approfondi des solutions de détection DDoS actuelles dans le SDN.

Selon le type de métrique de détection et de mécanisme de détection utilisé, ces solutions sont classées en quatre catégories : solutions de défense DDoS basées sur la théorie de l'information, solutions de défense DDoS basées sur

l'apprentissage automatique, solutions de défense basées sur le réseau de neurones artificiels et autres la défense

solution (voir Figure II.23). Pour comparer ces solutions, nous sélectionnons huit paramètres différents :

- Année de publication
- le périmètre de la solution (Détection ou Atténuation)
- métrique de détection utilisée
- paramètres utilisés dans l'algorithme de détection
- plan cible
- type de contrôleur
- le jeu de données utilisé pour la validation et les autres fonctionnalité

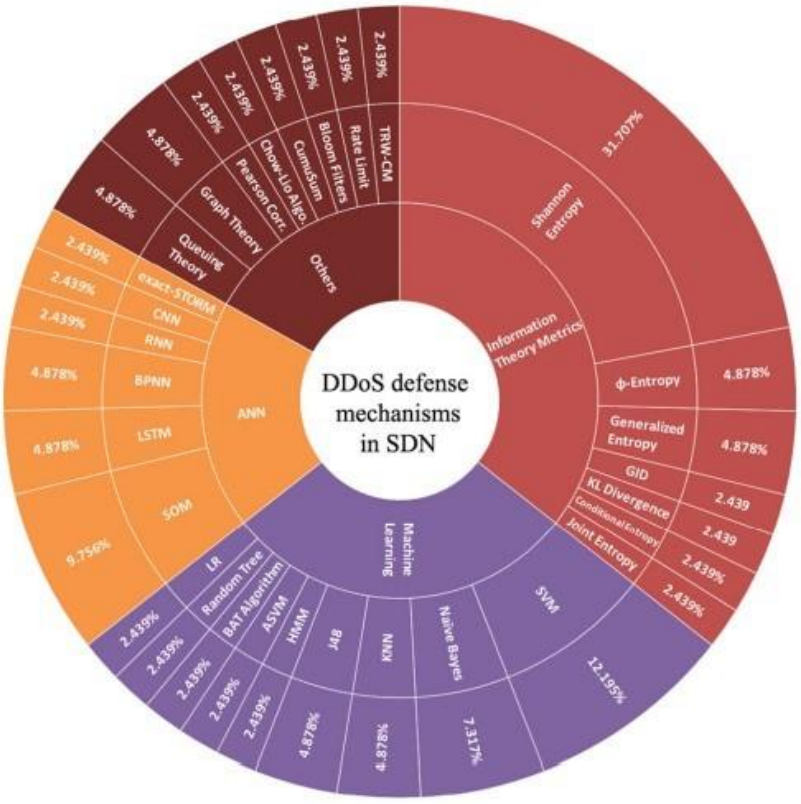


Figure II 11 Solutions de défense DDoS basées sur la théorie de l'information dans le SDN

- **Solutions de défense DDoS basées sur la théorie de l'information dans le SDN**

Les métriques d'entropie et de divergence basées sur la théorie de l'information sont couramment utilisées pour détecter les attaques DDoS. L'entropie représente le caractère aléatoire des caractéristiques du réseau, tandis qu'une métrique de divergence représente la similarité de deux distributions de probabilité . Le concept de mesure de l'incertitude est initialement inventé par Claude Shannon en 1948 [56]. La distance d'information ou la métrique de divergence, qui est calculée à l'aide de différentes distributions de probabilité des flux de trafic, est utilisée pour détecter les anomalies de trafic du réseau. En utilisant la mesure d'entropie, il est possible de voir comment le comportement actuel du réseau s'écarte du comportement normal du réseau, ce qui entraîne la détection d'une attaque DDoS. De nombreux chercheurs ont fourni des approches de défense DDoS basées sur la métrique d'entropie, comme le montre SDN sont programmables, ils permettent pour l'extraction et l'analyse des statistiques de flux du réseau. Giotis et al. étendent cette fonctionnalité en réduisant la charge du contrôleur lors de l'extraction des informations. La méthode proposée collecte et analyse des données en utilisant la méthode d'entropie pour détecter les anomalies du réseau. Le module collecteur est chargé de collecter régulièrement les données et de les envoyer au module de détection d'anomalies. Ensuite, la détection des anomalies examine toutes les entrées de flux pour chaque fenêtre temporelle afin d'identifier les flux malveillants.

Suite au module d'atténuation de la détection, des règles de flux concernant les flux malveillants ont été implémentées pour les bloquer. Ils testent leur méthode en collectant du trafic innocent sur le réseau de l'Université technique nationale d'Athènes. les programmes Tcpreplay et Scapy sont utilisés pour produire du trafic malveillant.

- **Solutions de défense DDoS basées sur l'apprentissage automatique dans le SDN**

Dans plusieurs domaines, les techniques d'apprentissage automatique sont utilisées pour résoudre des problèmes complexes, ces algorithmes ont également été utilisés pour détecter les attaques DDoS, et ils se sont avérés plus efficaces que les systèmes de détection basés sur les signatures. Ces classificateurs basés sur l'apprentissage automatique peuvent être formés pour détecter un comportement aberrant du trafic réseau avec une plus grande précision. Support Vector Machine (SVM), Arbre de décision (J48), Régression logistique, Advanced Support Vector Machine (SVM), Naive Bayes, Random Trees, Binary Bat algorithm, Random forest, Hidden Markov Model (HMM) et K-plus proche voisin (KNN) sont parmi les classificateurs d'apprentissage automatique les plus couramment utilisés, comme indiqué et résumé dans le tableau II.2. [57].

Propriétés	**Contrôleurs SDN**								
	Beacon	DISCO	IRIS	Maestro	OpenDaylight	NOX	POX	Floodlight	Ryu
Support du Parallélisme	Oui	Oui	Oui	Oui	Oui	Non	Non	Oui	Oui
Protocole	OpenFlow	OpenFlow	OpenFlow OVSDB	OpenFlow	OpenFlow OVSDB	OpenFlow	OpenFlow	OpenFlow	OpenFlow OVSDB
L'architecture	Centralisée	Distribuée	Centralisée	Centralisée	Distribuée	Centralisée	Centralisée	Centralisée	Centralisée
Langage de programmation	Java	Java	Java	Java	Java	C++	Python	Java	Python
API supportés	Ad-hoc API	REST API	REST API	Ad-hoc API	REST API	Ad-hoc API	Ad-hoc API	REST API	Ad-hoc API
Plateformes supportées	Linux Windows	Linux Windows	Linux Windows	Linux Windows Mac OS	Linux	Linux Windows Mac OS	Linux Windows Mac OS	Linux Windows Mac OS	Linux

Tableau 4 et discuté brièvement ci-dessous. Parce que les réseaux

2.12 **Discussion**

Notre étude comparative des solutions de détection d'attaques DDoS dans les réseaux SDN que nous pensons nous a permis de créer le tableau II.10.1. À partir de là, nous avons pu sélectionner un ensemble de résultats. Le tableau ci-dessus montre que les approches ont été utilisées différemment pour le même

but. En d'autres termes, alors que les mécanismes basés sur la théorie de l'information collectent et analysent les informations des commutateurs pour créer un module de détection DDoS, les mécanismes basés sur ML identifient les attaques DDoS en capturant le trafic et en identifiant les fonctionnalités appropriées.

À partir du tableau présenté précédemment, nous avons pu voir que les travaux étudiés basés sur des approches de théorie de l'information sont efficaces pour estimer le caractère aléatoire d'un ensemble de données. Par conséquent, des niveaux d'entropie faibles indiquent une distribution de probabilité plus concentrée pour identifier les anomalies probables du réseau. En revanche, des valeurs d'entropie élevées indiquent une distribution de probabilité plus distribuée. Cependant, nous soutenons que le plein potentiel de la détection des attaques DDoS basées sur l'entropie n'est actuellement pas exploité en raison de son utilisation inefficace. L'approche d'apprentissage automatique augmente la précision de la détection DDoS et offre une indépendance vis-à-vis du matériel. De plus, cela réduit la charge de travail et le temps. Nous laissons l'algorithme effectuer l'analyse des processus en notre nom en automatisant les tâches. De même, de nombreux éléments affectent l'efficacité de l'apprentissage automatique. L'informatique en est une. Il est capable de traiter tout type de données.

2.13 **Conclusion**

L'émergence des environnements SDN a fait que de nombreux chercheurs et industriels voient d'un bon œil ce nouveau concept. Cependant, cela apporte également de nouveaux problèmes à traiter, tels que de nouveaux types d'attaques DDoS. Par exemple, l'attaque DDoS dans l'environnement SDN pourrait attaquer le contrôleur centralisé et arrêter tous les réseaux si le contrôleur tombe en panne. En conséquence, nous fournissons un examen complet des stratégies de pointe de détection des attaques DDoS présentées par les technologies SDN. Dans ce travail, nous proposons les ML-Algorithmes pour détecter les attaques DDoS.

Chapitre 3 Application, tests et résultats

3.1 Introduction

Ce chapitre vise à présenter notre approche utilisée pour notre mécanisme, qui détecte les attaques DDoS dans les réseaux SDN, et décrit les outils utilisés pour mettre en œuvre et simuler l'expérimentation. Le module développé pour la détection est implémenté comme une application de contrôleur Ryu. La simulation de la topologie se fait avec le réseau virtuel Mininet. Pour le trafic DDoS, nous utilisons l'outil Hping3.

3.2 Approche proposée

Dans notre approche, nous proposons de mettre en œuvre des algorithmes d'apprentissage automatique pour détecter le trafic DDoS, en raison de ses avantages. Nous pouvons expliquer notre approche avec ces étapes résumées :

3.2.1 Phase de construction du modèle :

Le but de cette phase est de construire notre modèle et de s'assurer que nous avons pris les meilleurs choix et décisions lors de sa construction. Surtout quand nous avons beaucoup de choix concernant l'ensemble de données et les algorithmes d'apprentissage automatique. Comme le motre l'organigramme de la figure III.1.

La première étape de cette phase consiste à collecter les données car elles jouent un rôle principal dans les performances du modèle d'apprentissage automatique. Dans cette étape, nous devons faire des recherches sur les données disponibles, qui sont liées à notre problème et comparer les données basées sur les articles disponibles et les chercheurs qui les ont utilisés et en ont parlé. Après la collecte de données, nous devons entraîner ces données à l'aide d'un des nombreux algorithmes ML qui existent, pour ce faire, nous devons savoir quels sont les algorithmes les plus utilisés dans ce cas de

problème et parmi ces algorithmes, lequel est le meilleur pour notre cas. Selon une étude réalisée en 2020, nous avons constaté que les algorithmes ML les plus couramment utilisés pour l'IDS sont l'arbre de décision, le K-Nearest Neighbor (KNN), le réseau de neurones artificiels (ANN), la machine vectorielle de support (SVM), le clustering K-Mean, Fast Réseau d'apprentissage et méthodes d'ensemble ,Ainsi, en raison de cette étude, nous avons décidé d'utiliser six algorithmes ML nommés (LR, KNN, SVM, NB, DT, RF). LR et NB ont été sélectionnés comme l'un des algorithmes ML les plus linéaires utilisés et pour examiner et comparer leurs performances avec les autres algorithmes les plus utilisés dans le cas des systèmes de détection d'intrusion.La troisième étape de cette phase consiste à sélectionner le meilleur algorithme parmi les sixièmes algorithmes à l'aide de métriques d'évaluation. Une fois que nous avons le meilleur modèle, nous devons utiliser l'une des techniques de sélection de caractéristiques dont nous avons discuté dans le chapitre précédent afin de réduire le temps de prédiction et d'éviter le surajustement s'il existe.

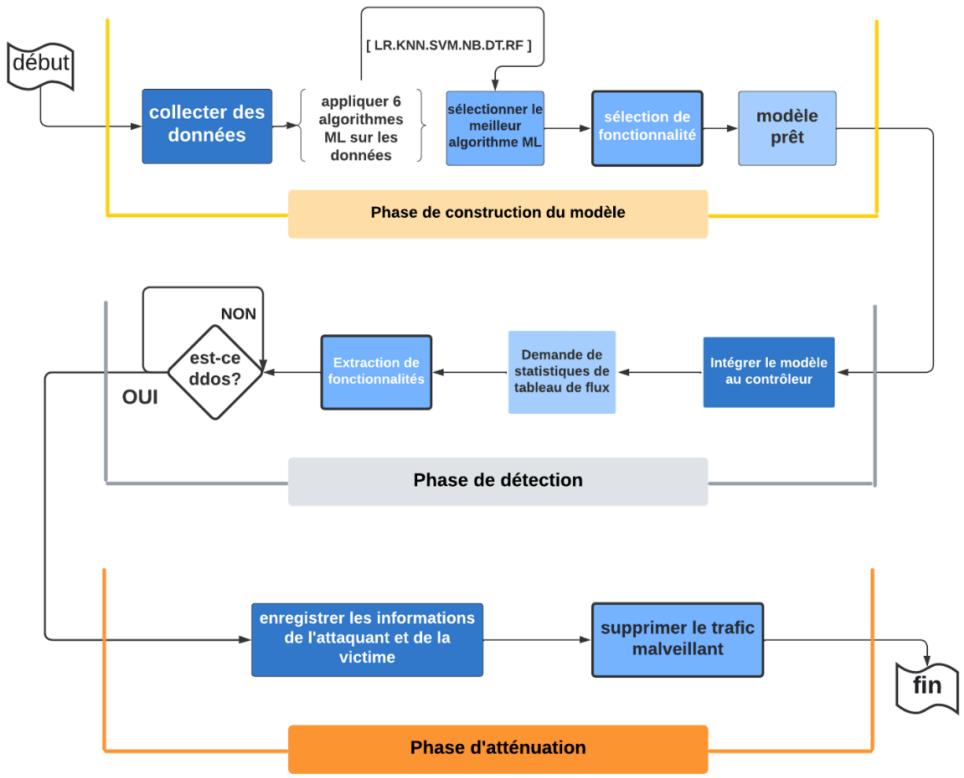


Figure III 1 L'approche proposée

3.2.2 **Phase de détection :**

Une fois que nous avons notre modèle, tout ce dont nous avons besoin est de l'intégrer dans le contrôleur afin que nous puissions faire la prédiction et détecter toute anomalie sur le réseau. Ensuite, nous configurons le contrôleur pour demander des statistiques de table de flux toutes les 5 secondes comme utilisé par qui a déterminé que cette fenêtre est la plus appropriée pour permettre la détection des attaques suffisamment tôt pour protéger le contrôleur. Donc, ici, le contrôleur fait une extraction de fonctionnalités pour prédire si nous sommes attaqués ou non.

3.2.3 Phase d'atténuation :

Une fois que le modèle détecte une attaque DDoS, il enregistre les informations de l'attaquant et de la victime afin d'enregistrer l'adresse IP de la victime et l'adresse MAC et l'ID du chemin de données (l'ID du commutateur) de l'attaquant. Ensuite, nous installons une nouvelle règle sur le commutateur dont nous avons déjà enregistré l'identifiant afin de supprimer tous les paquets provenant de l'adresse MAC de l'attaquant. De cette façon, nous empêchons tout trafic provenant de l'attaquant, nous visons à comparer six algorithmes supervisés différents afin de choisir le meilleur d'entre eux.

Dans cette approche, nous configurons le contrôleur pour surveiller le trafic et l'analyser en extrayant les informations de chaque flux puis utiliser le modèle que nous avons formé pour faire la prédiction et informer si nous sommes attaqués ou s'il s'agit d'un trafic légitime. Une fois qu'il détecte une attaque DDoS, il ajoute une nouvelle règle de flux au commutateur qui supprime tous les paquets provenant du nœud attaquant pour atténuer et empêcher cette attaque.

3.3 Fonctionnement de l'approche proposée

Le système fonctionne à chaque fois que le contrôleur demande aux commutateurs de renvoyer les statistiques des tables de flux toutes les cinq secondes. Ensuite, les commutateurs renvoient ces statistiques au contrôleur. Le contrôleur utilise ces statistiques pour prédire si les flux sont légitimes ou s'il s'agit de trafic DDoS. Si le trafic est prédit comme DDoS, le contrôleur indique la source de l'attaque et la victime.

Ensuite, il commence la phase d'atténuation en ajoutant une entrée de flux qui correspond aux informations de l'attaquant afin de supprimer tous ces paquets.

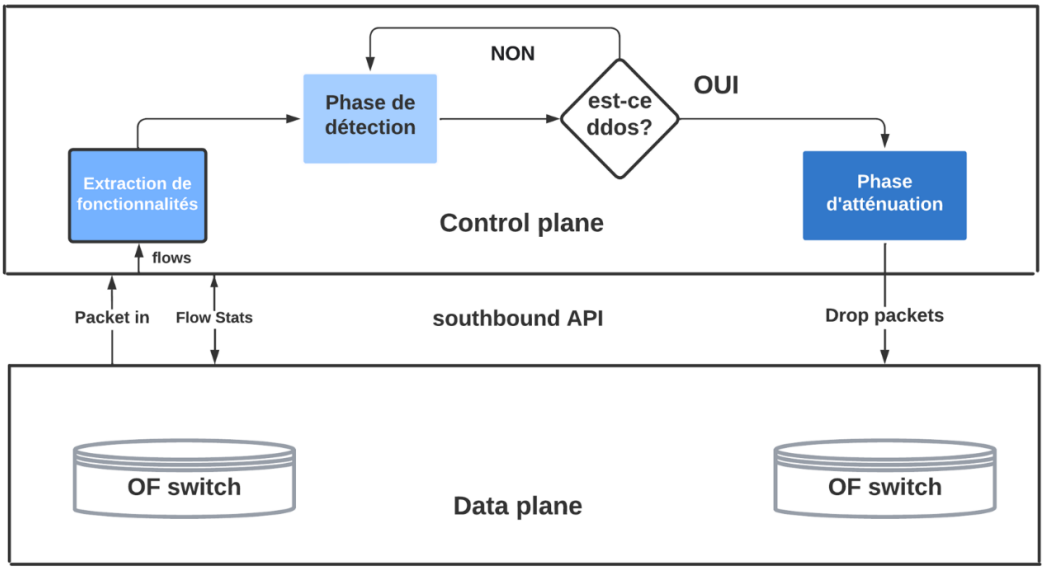


Figure III 2 fonctionnement de l'approche prposée

La figure explique le fonctionnement de ce mécanisme. Il montre comment les paquets entrants sont envoyés au contrôleur une fois qu'ils arrivent au commutateur. Donc ici, le contrôleur extrait toutes les fonctionnalités de ce paquet et le sauvegarde. Ensuite, avant de l'envoyer au modèle, il sélectionne les fonctionnalités pour opter pour les caractéristiques les plus importantes pour faire la prédiction afin de réduire le temps de prédiction. Dans la phase de détection, le contrôleur utilise le modèle pour effectuer la prédiction à l'aide des caractéristiques sélectionnées. Selon le résultat de la prédiction, le contrôleur prend la décision en informant comme dans les deux cas et en cas d'attaque, le contrôleur ajoute une nouvelle règle de flux au commutateur en utilisant le protocole OpenFlow pour échanger ces messages afin que le commutateur abandonne tous les paquets. Provenant de l'attaquant pour atténuer et empêcher ce trafic DDoS.

3.4 Moyens matériels

Il est important de mentionner que ce travail a été émulé en utilisant :

• Processeur : processeur Intel (R) Core(TM) i7-8565U CPU @ 1,80 GHz 1,99 GHz

• Mémoire installée (RAM) : 16 Go

• Type de système : Système d'exploitation 64 bits, processeur x64

• Windows 10 Professionnel

3.5 Environnement de développement

Cette section présente les outils utilisés dans ce projet et les raisons de l'utilisation de chaque outil.

3.5.1 Langage Python

Python est un langage de programmation interprété, orienté objet et de haut niveau avec une sémantique dynamique. Ses structures de données intégrées de haut niveau combinées au typage dynamique et à la liaison dynamique ; le rendent très attrayant pour le développement rapide d'applications. De plus, python est une syntaxe simple et facile à apprendre, met l'accent sur la lisibilité et réduit donc le coût de maintenance du programme .

Puisque nous travaillerons avec l'apprentissage automatique et les données, nous avons constaté que python est largement utilisé dans ce domaine en raison de ses différents packages open source.

3.5.2 Mininet

Mininet est un émulateur de réseau qui crée un réseau d'hôtes virtuels, de commutateurs, de contrôleurs et de liens. Les hôtes Mininet exécutent un logiciel réseau Linux standard et ses commutateurs prennent en charge OpenFlow pour un routage personnalisé extrêmement flexible et un réseau défini par logiciel. Mininet prend en charge la recherche, le développement, l'apprentissage, les tests, et toute autre tâche pouvant bénéficier d'un réseau expérimental complet sur un ordinateur portable ou un autre PC.

- Grâce à la commande `$ sudo mn` et la multitude d'options offertes par l'émulateur, il est possible de réaliser maintes formes de topologies et de réseaux virtuels.
- L'option `$ --topo=` permet de créer 3 types de topologies de base : **tree** (Arbre), **linear** (linéaire) et **single** (topologie centralisée d'un seul switch).
- Pour connecter notre topologie à un contrôleur l'option adéquate est `--controller`, dans l'exemple d'un contrôleur distant on ajoute `=remote,ip=192.168.0.101` , L'adresse que nous utilisons est l'adresse locale « «localhost addr».
- D'autres options sont disponibles sur Mininet pour n'en citer que certaines : `--switch` offre la possibilité de choisir le modèle de switch parmi ceux supportés par l'émulateur, et `$ --mac` qui simplifie l'adresse mac des nœuds.
- L'option –x ouvre un terminal xterm pour chaque élément du réseau généré, et `--link tc` pour pouvoir personnaliser la bande passante et la latence des liaisons.

Après avoir inséré la commande mn suivie des options voulues et créé la topologie nous accédons à la ligne de commande de Mininet identifiable par l'expression `mininet>` au début de chaque ligne (Hors CLI les commandes sont précédées du symbole **$** comme toute plateforme sous linux). Les commandes principales de la CLI sont :

- `mininet> net` Affiche toute les liaisons du réseau, `mininet> nodes` qui cite tout les nœuds réseau et `mininet> dump` qui affiche les informations liées aux nœuds du réseau.
- Un simple test de connectivité peut se faire grâce au Ping et la commande `mininet> iperf` permet de mesurer la bande passante entre deux hôtes.
- Un terminal xterm peut être ouvert pour chaque hôte en utilisant `xterm h18` ou h18 est le nom de l'hôte. On peut ainsi exécuter tout type de programme sur les différents hôtes (Tout ce qui est disponible sous linux) et ainsi tester le réseau à l'aide d'outils identiques à ceux de l'installation physique.

- Pour ce qui est la connectique avec OpenFlow `mininet> dpctl dump-flows` nous permet d'afficher les différents flux installés sur les nœuds à partir de la ligne de commande.

- **Avantages de Mininet**

  • C'est rapide - il faut quelques secondes pour démarrer un réseau.

  • Il est facile de créer des topologies personnalisées allant d'un commutateur unique aux topologies de centre de données.

  • Peut fonctionner sur un ordinateur portable comme dans ce cas.

  • Il est open source

- **Inconvénients de Mininet**

  • Prise en charge d'une seule plate-forme (noyau Linux)

  • Les hôtes Mininet partagent des systèmes de fichiers hôtes et des espaces d'ID de processus. Ils nécessitent donc une attention particulière lors de l'exécution de démons nécessitant une configuration.

La raison du choix de Mininet parmi les simulateurs/émulateurs disponibles n'est pas seulement due à ses avantages mais aussi selon une comparaison entre ces outils mentionnée dans [94].

3.5.3 OpenVSwitch

C'est un composant essentiel dans notre émulation, OpenVswitch est un outil très utilisé dans le monde SDN. C'est un commutateur virtuel sous licence open source Apache 2.0 que l'on utilise pour interconnecter des machines virtuelles provenant d'un hôte ou de machines distantes. Il présente plusieurs fonctionnalités d'un commutateur couche 2 et même d'un commutateur couche 3

3.5.4 Contrôleur Ryu

Ryu est un contrôleur Open Flow basé sur python avec une API robuste qui permet aux développeurs de créer leurs applications pour gérer le réseau. Il est open-source et

disponible sous la licence Apache. De plus, il peut être utilisé pour collecter des informations statistiques à partir des commutateurs. Ainsi, il peut être configuré comme un moniteur de trafic, un pare-feu ou un commutateur . La raison pour laquelle nous avons choisi Ryu parmi une liste de contrôleurs pour simuler notre travail est que :

- Nous allons utiliser le langage Python pour écrire nos algorithmes.
- Ryu est bien documenté contrairement à la plupart des autres contrôleurs.
- selon [64] Ryu est très rapide par rapport à POX et Pyretic.

Proprieties	Ryu controller
Language	Python
OF version	1.0 , 1.1 , 1.2 , **1.3**
Rest API support	yes
Platform support	Linux

Tableau 5 Caractéristiques du contrôleur Ryu

3.5.5 Hping l'outil d'attaque DDoS

Hping peut gérer la taille aléatoire des paquets et les fragmentations. En outre, hping effectue les tests de règles de pare-feu, l'analyse des ports et les tests de performances réseau basés sur le protocole. Son langage d'implémentation est TCL et possède une interface en ligne de commande .

La sélection d'un outil d'attaque DDoS a été difficile car nous avons trouvé de nombreux outils utilisés à cette fin. Une comparaison entre différents outils DDoS est fournie dans [nous n'avons pas trouvé de différence énorme entre eux, nous avons donc

opté pour Hping3 au hasard car il fonctionne bien, et puisqu'il utilise les attaques d'inondation ICMP, TCP et UDP que nous utilisons dans ce travail.

- **Syntaxe de la commande hping3 :**

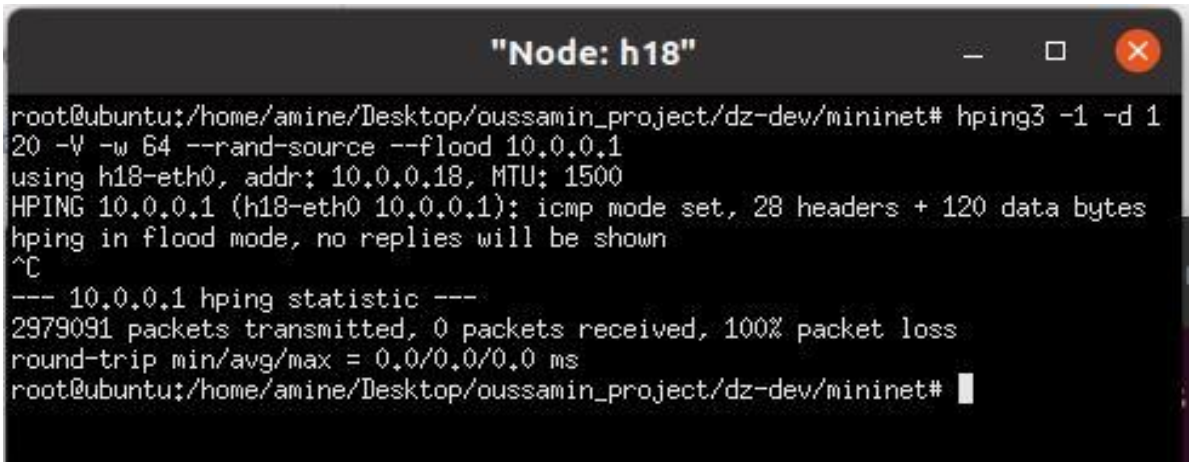


Figure III 3 Syntaxe de la commande hping3

hping3 = Nom du binaire de l'application.

-c 100000 = Nombre de paquets à envoyer.

-**d 120** = Taille de chaque paquet envoyé à la machine cible.

-S = J'envoie uniquement des paquets SYN.

-w 64 = taille de la fenêtre TCP.

-p 21 = port de destination (21 étant le port FTP). Vous pouvez utiliser n'importe quel port ici.

--flood = Envoi de paquets le plus rapidement possible, sans prendre soin d'afficher les réponses entrantes. Mode inondation.

--rand-source = Utilisation d'adresses IP sources aléatoires. Vous pouvez également utiliser -a ou –spoof pour masquer les noms d'hôte. Voir la page MAN ci-dessous.

www.gbhackers.com = Adresse IP de destination ou adresse IP des machines cibles. Vous pouvez également utiliser un nom de site Web ici. Dans mon cas, se résout en 10.0.0.18 (comme indiqué dans le fichier /etc/hosts)

- **Test des règles de pare-feu**

Hping3 par défaut (sans option) envoie un paquet nul avec un en-tête TCP au port 0. Vous pouvez choisir d'utiliser un protocole différent en utilisant l'option numérique disponible pour chacun :

- **- 1** (ICMP mode)
- **- 2** (UDP mode)
- **- 8** (Scan mode)
- **- 9** (Listen mode)

3.6 Fichiers et fonctions utilisés dans la mise en œuvre

Chemin d'accès	**Files**	**Description**
/Desktop/oussamin_project/ dz-dev/controller	collect_benign_trafic.py	Ce fichier fonctionne comme un moniteur et collecteur du trafic normal. Nous l'avons utilisé pour collecter le trafic normal et le mettre dans un fichier CSV.
	collect_ddos_trafic..py	Ce fichier fonctionne comme un moniteur et un collecteur de trafic ddos
	controller.py	Ce fichier est utilisé pour surveiller et collecter le trafic DDoS puis le mettre dans un fichier CSV.

	Mitigate.py	Ceci est notre application pour détection et atténuation des attaques DDoS à l'aide de l'algorithme RF pour la classification et l'installation de règles de flux sur le commutateur pour éliminer les paquets malveillants.
/Desktop/oussamin_project/ dz-dev/mininet	Generate ddos trafic.py	Ce fichier est utilisé pour générer Trafic DDoS (ICMP flood, TCP flood, UDP flood, SYN flood) à l'aide de l'outil Hping3.
	Generate_normal_trafic.py	Ce fichier est utilisé pour générer trafic normal (ping, TCP et UDP).
	topology.py	Ce fichier représente notre scénario de topologie.
/Desktop/oussamin_project/ dz-dev/ml	(LR KNN SVM NB DT RF).py	nous visons à comparer six algorithmes supervisés différents afin de choisir le meilleur d'entre eux.

Tableau 6 Fichiers et fonctions utilisés dans la mise en œuvre

Dans cette section, nous donnons un aperçu des fichiers et des fonctions utilisées pour mettre en œuvre ce système avant d'entrer dans les détails. Toutes les informations sont résumées dans le tableau III.2

Il est important de mentionner que la plupart de ces fonctions sont préexistantes dans les applications Ryu. La seule chose que nous ayons faite est de

comprendre la fonctionnalité de ces fonctions, de les modifier et d'en ajouter de nouvelles.

3.7 Installation et configuration

Voici les étapes nécessaires pour installer les logiciels requis pour émuler un réseau multi-domaines, comme décrit ci-dessus :

3.7.1 Mininet

Il existe trois façons d'installer mininet sur Ubuntu 20.04. Nous pouvons utiliser **apt-get**, **apt** et **aptitude**. dans notre cas, nous utilisons la méthode apt-get mettez à jour la base de données apt avec apt-get en utilisant la commande suivante.

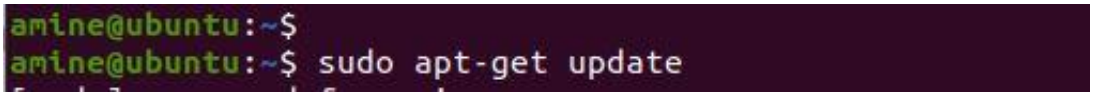


Figure III 4 utiliser apt-get, apt

Après la mise à jour de la base de données apt, nous pouvons installer mininet en utilisant apt-get en exécutant la commande suivante :

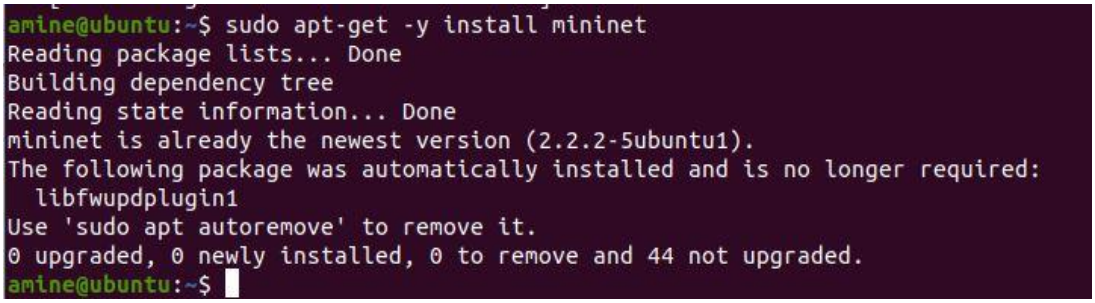


Figure III 5 installer mininet

Mininet est déjà installé

3.7.2 ryu

L'installation de Ryu est assez simple :

Figure III 6 L'installation de Ryu

Si vous préférez installer Ryu depuis le code source :

Figure III 7 installer Ryu depuis le code source

RYU est déjà installé

3.7.3 **hping3**

Cette commande montrent comment installer hping3 sur Ubuntu 20.04.

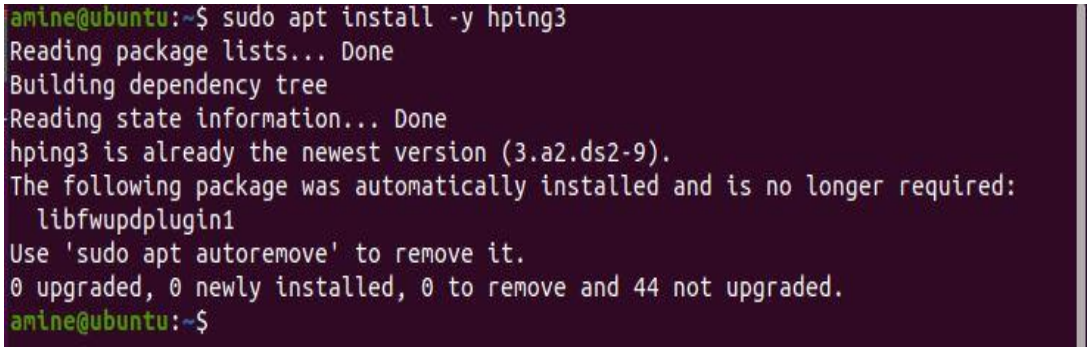


Figure III 8 montrent comment installer hping3 sur Ubuntu 20.04**.**

Hping3 est déjà installé

3.7.4 **création d'une topologie personnalisée via l'API Mininet :**

L'émulateur Mininet fournit une interface de programmation Python, l'une de ces utilisations consiste en la création de topologies customisées en y définissant toutes les caractéristiques voulues des éléments du réseau, une approche pareille nous offre l'avantage d'exploiter le temps et les ressources en main de la meilleure façon possible.

Alors on illustre avec un exemple la structure générale d'un script python « aminetopo.py » comprenant les informations du réseau voulu.

Le script python doit être sauvegardé dans le répertoire :~/mininet/custom$

```
amine@ubuntu:~/mininet/custom$ ls
aminetopo.py  aminetopo.py.save  aminetopo.py.save.1  README  topo-2sw-2host.py
```

Figure III 9 ls

de plus pour pouvoir appeler la topologie par l'option --custom la dernière ligne du script est primordiale, en gros nous y avons définis la topologie par le nom « aminetopo ». La commande complète pour générer cette topologie devra comporter le chemin d'accès du script et le nom de la topologie en question

```
amine@ubuntu:~/mininet/custom$ sudo mn --custom=aminetopo.py --topo=mytopo -
-controller=remote,ip=192.168.0.101 --mac
```

Figure III 10 appelé la topologie

Celle-ci générera les différents éléments de la couche physique qui sera connectée au contrôleur RYU à distance.

```
amine@ubuntu:~/mininet/custom$ sudo mn --custom=aminetopo.py --topo=mytopo -
-controller=remote,ip=192.168.0.101 --mac
*** Creating network
*** Adding controller
Connecting to remote controller at 192.168.0.101:6653
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h16 h17 h18
*** Adding switches:
s1 s2 s3 s4 s5 s6
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s2) (h5, s2) (h6, s2) (h7, s3) (h8, s3) (h9,
 s3) (h10, s4) (h11, s4) (h12, s4) (h13, s5) (h14, s5) (h16, s5) (h16, s6) (
h17, s6) (h18, s6) (s1, s2) (s2, s3) (s3, s4) (s4, s5) (s5, s6)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h16 h17 h18
*** Starting controller
c0
*** Starting 6 switches
s1 s2 s3 s4 s5 s6 ...
*** Starting CLI:
```

Figure III 11 Création des éléments du réseau dans Mininet

Le contrôleur détecte les 6 switchs Openflow, et après un ping global détecte les hôtes et parvient à afficher la structure du réseau sous-jacent comme l'atteste la figure suivant ;

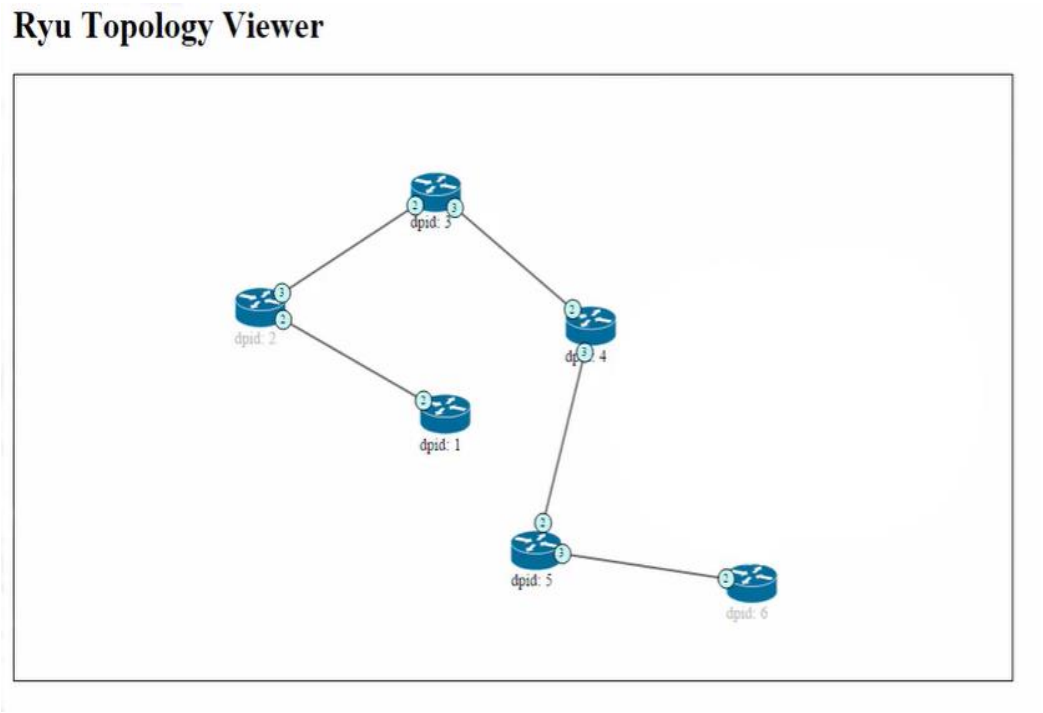


Figure III 12 notre topologie

Dans ce cas, nous avons généré 6 switchs, et 18 hôtes auxquelles nous avons alloué différents identifiants.

Figure III 13 généré 6 switchs

3.8 Ensemble de données et algorithmes ML (Dataset and ML algorithms)

Cette section représente la construction du processus de détection. Il comprend la sélection de l'ensemble de données et de l'algorithme d'apprentissage automatique pour effectuer la prédiction.

3.8.1 Sélection du Dataset

Les performances des modèles d'apprentissage automatique dépendent largement de la sélection des fonctionnalités, de la qualité et de la quantité des données d'apprentissage .

❖ Générer un trafic normal

Alors maintenant, nous allons générer du trafic normal à partir de la machine virtuelle mininet et collecter le trafic généré là où le contrôleur préparerait notre ensemble de données, comme vous pouvez le voir sur **la figure** Error! No text of specified style in document.**-15.**

Figure III 15 générer du trafic normal

Le code source pour collecter le trafic régulier se trouve dans le script python **collect_benign_trafic.py** .

Et nous avons écrit une application python pour collecter le trafic généré depuis le mininet qui est Montré dans la figure III **-1**

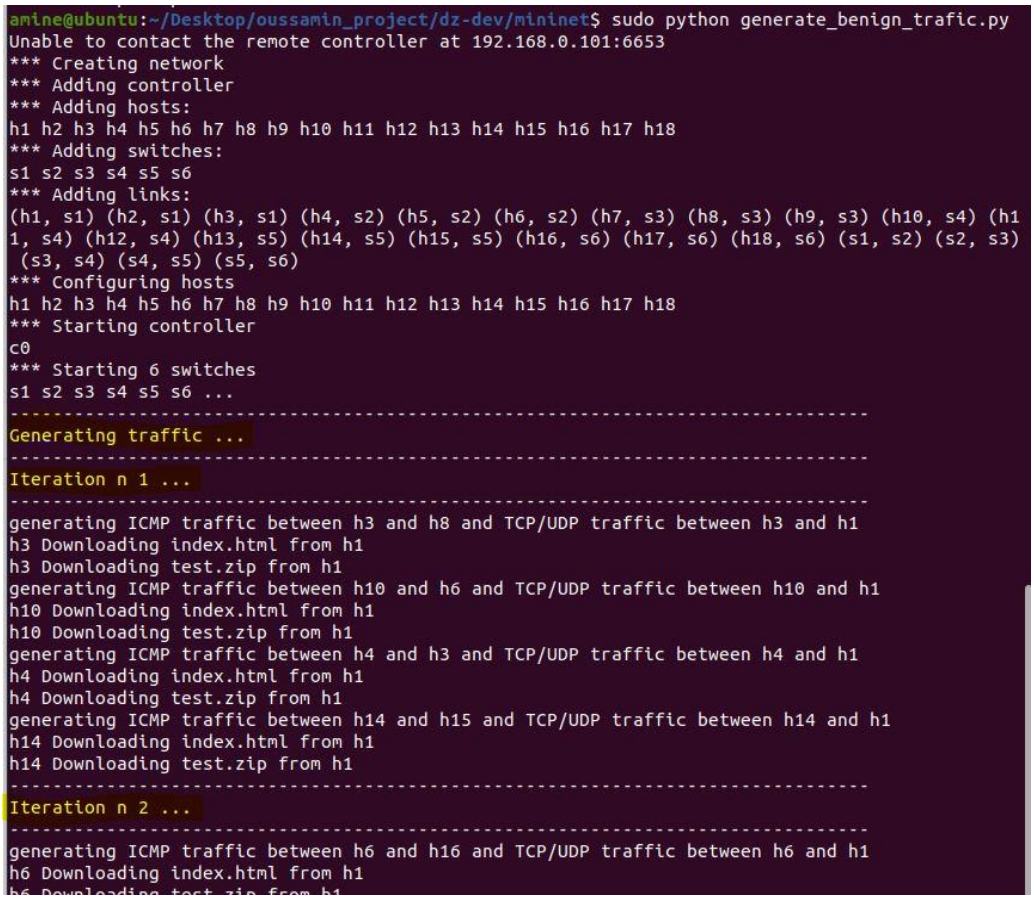


Figure III 16 collecter le trafic généré

Donc ce qui s'est passé ici, c'est que nous avons un simple serveur http dans l'hôte 1 et tous les autres hôtes téléchargent la page Web par défaut à partir du serveur Web pendant que tous les autres hôtes se déchirent et que tout ce trafiquant a collecté le contrôle de révision Et c'est ainsi que nous créons un trafic normal.

Et pour votre information, nous avons presque mis environ trois jours pour terminer le processus de génération de données. ainsi, nous pouvons rassembler les informations et les enregistrer dans un FlowStatsfile.csv.

Le code source pour générer du trafic normal se trouve dans le script python **generate_benign_trafic.py** .

❖ Générer un trafic DDos

De même, nous générons le trafic d'attaque dans la machine virtuelle mininet à l'aide du script python **generate_ddos_trafic.py**, ce trafic sera calculé par le contrôle

de révision à l'aide de l'application python **collec_ddos_trafic.py** en cours d'exécution, comme nous le voyons sur la figure *III* -17 et *III* -18)

Figure III-17 Générer un trafic DDos

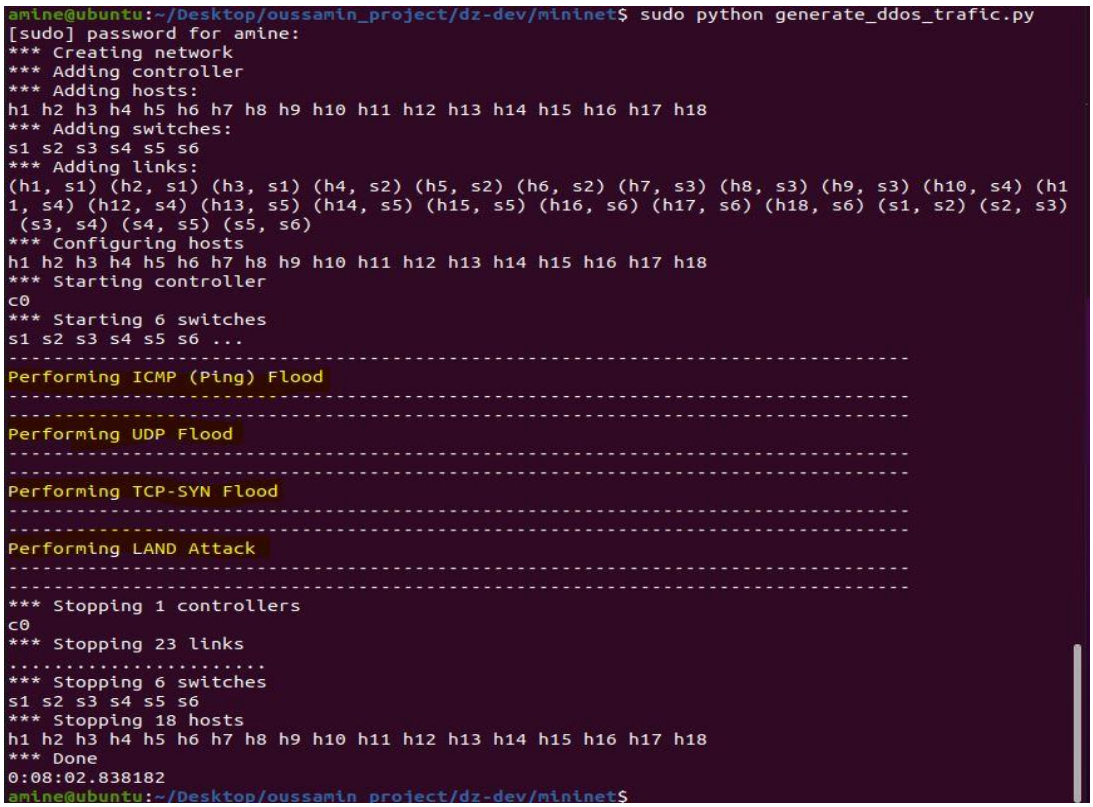


Figure III 19 Générer un trafic DDos

Le code source pour collecter et générer le trafic ddos se trouve dans le script python **collect_ddos_trafic.py** et **generate_ddos_trafic.py** .

3.8.2 Comparaison entre les algorithmes ML

Comme cela a été discuté dans le chapitre précédent, il existe une variété d'algorithmes d'apprentissage automatique qui peuvent être utilisés pour faire la

classification des flux. Il était donc essentiel de comparer ce sous-ensemble d'algorithmes (LR, K-NN, SVM, NB, DT, RF) pour sélectionner la plus grande précision. Pour ce travail, nous utilisons le script **ML.py**

Le code source pour sélectionner l'algorithme la plus grande préciser se trouve dans le script python **ML**.

```
amine@ubuntu:~/Desktop/oussamin_project/dz-dev/controller$ python3 ML.py
Loading dataset ...
```

Figure III 20 Comparaison entre algorithmes

Tout ce dont nous avons besoin est d'exécuter ce script et de sélectionner l'algorithme avec la meilleure précision uniquement parce que nous savons que la précision fonctionne bien et peut être satisfaite en utilisant un nombre égal d'échantillons appartenant à chaque classe, ce qui est notre cas. Exécutons le script et voyons les résultats :

```
------------------------------------------------------------------------------
Logistic Regression ...
------------------------------------------------------------------------------
confusion matrix
[[    0   537]
 [    0 91724]]
succes accuracy = 99.42 %
fail accuracy = 0.58 %
------------------------------------------------------------------------------
LEARNING and PREDICTING Time:  0:00:01.486364
------------------------------------------------------------------------------
K-NEAREST NEIGHBORS ...
------------------------------------------------------------------------------
confusion matrix
[[    0   537]
 [    0 91724]]
succes accuracy = 99.42 %
fail accuracy = 0.58 %
------------------------------------------------------------------------------
LEARNING and PREDICTING Time:  0:00:48.279741
------------------------------------------------------------------------------
NAIVE-BAYES ...
------------------------------------------------------------------------------
confusion matrix
[[  537     0]
 [31825 59899]]
succes accuracy = 65.51 %
fail accuracy = 34.49 %
------------------------------------------------------------------------------
LEARNING and PREDICTING Time:  0:00:00.188536
------------------------------------------------------------------------------
DECISION TREE ...
------------------------------------------------------------------------------
confusion matrix
[[  537     0]
 [    0 91724]]
succes accuracy = 100.00 %
fail accuracy = 0.00 %
------------------------------------------------------------------------------
LEARNING and PREDICTING Time:  0:00:00.236434
------------------------------------------------------------------------------
RANDOM FOREST ...
------------------------------------------------------------------------------
confusion matrix
[[  537     0]
 [    0 91724]]
succes accuracy = 100.00 %
fail accuracy = 0.00 %
------------------------------------------------------------------------------
```

Figure III 21 Comparaison entre algorithmes.

Les résultats montrent que :

• NB donne une précision de 65.51%.

• LR et SVM donnent la même précision avec 99.42% de réussite.

• K-NN donne une précision de 99,42% de réussite.

• DT et RF donnent la meilleure précision avec 100% de succès.

Les résultats de l'algorithme représenté dans la figure suivante

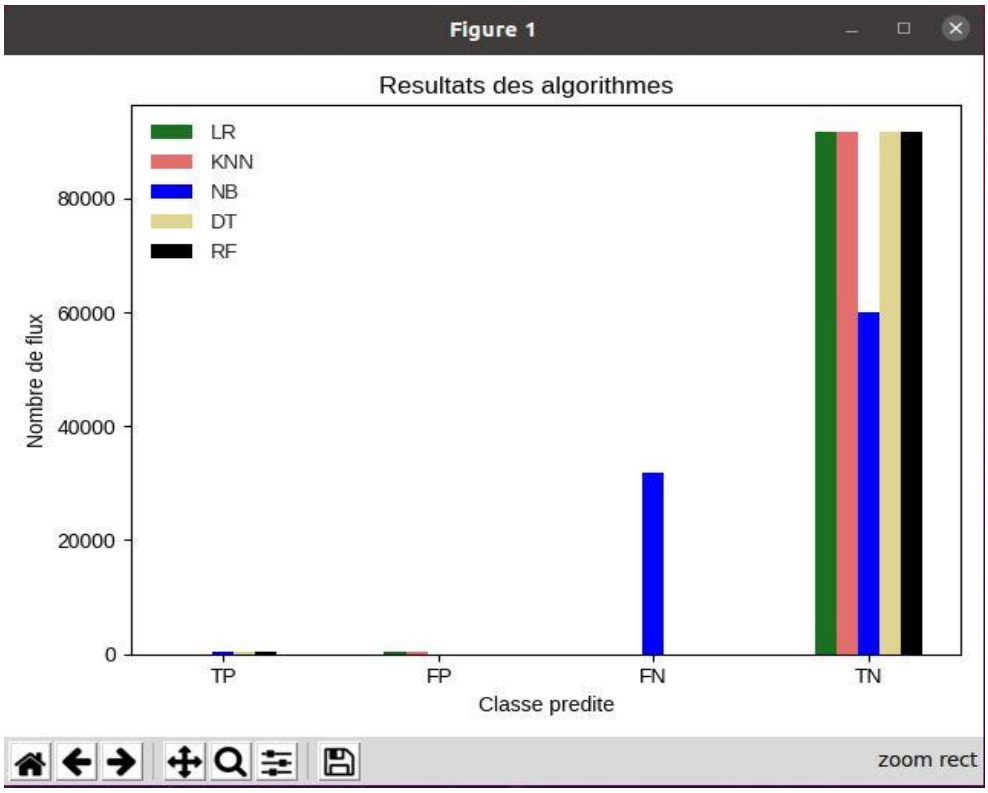


Figure III 22 résultats des algorithmes

3.8.3 Comment calculer la précision de chacun de ces algorithmes :

La précision n'est pas le seul moyen d'évaluer les performances ; parfois, nous avons besoin d'une image complète des performances du classifier. L'un de ces tableaux détaillés est la matrice de confusion.

	Predicted	
	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Tableau 7 Matrice de confusion

La précision d'un algorithme est une mesure de la façon dont il classe les occurrences invisibles, et elle peut être calculée à l'aide de la formule ci-dessous :

$$\boldsymbol{accuracy} = \frac{(\mathbf{TP} + \mathbf{TN})}{(\mathbf{TP} + \mathbf{FP} + \mathbf{FN} + \mathbf{TN})}$$

Dans le cas de l'algorithme **NB** on a :

$$\boldsymbol{accuracy} = \frac{((\mathbf{537} + \mathbf{59899})}{(\mathbf{537} + \mathbf{0} + \mathbf{31825} + \mathbf{59899})} = \mathbf{0,6550546}$$

Accuracy (NB) = 65.5%

3.9 La détection du ligitimate et ddos trafic utilisons le meilleur algorithme de machine learning

Selon le résultat précédent, nous avons décidé d'utiliser l'algorithme RF.

3.9.1 La détection du ligitimate trafic

à l'aide de l'application python **RF_controller.py** je vais executer la topolgie **topology.py** une fois que j'ai fait la topologie je vais just entrer la commande Ping pour vérifier si le réseau est opérationnel et fonctionne correctement ou non .

Tout ce dont nous avons besoin est d'exécuter le script RF_controller.py et de montrer sa précision (accuracy).

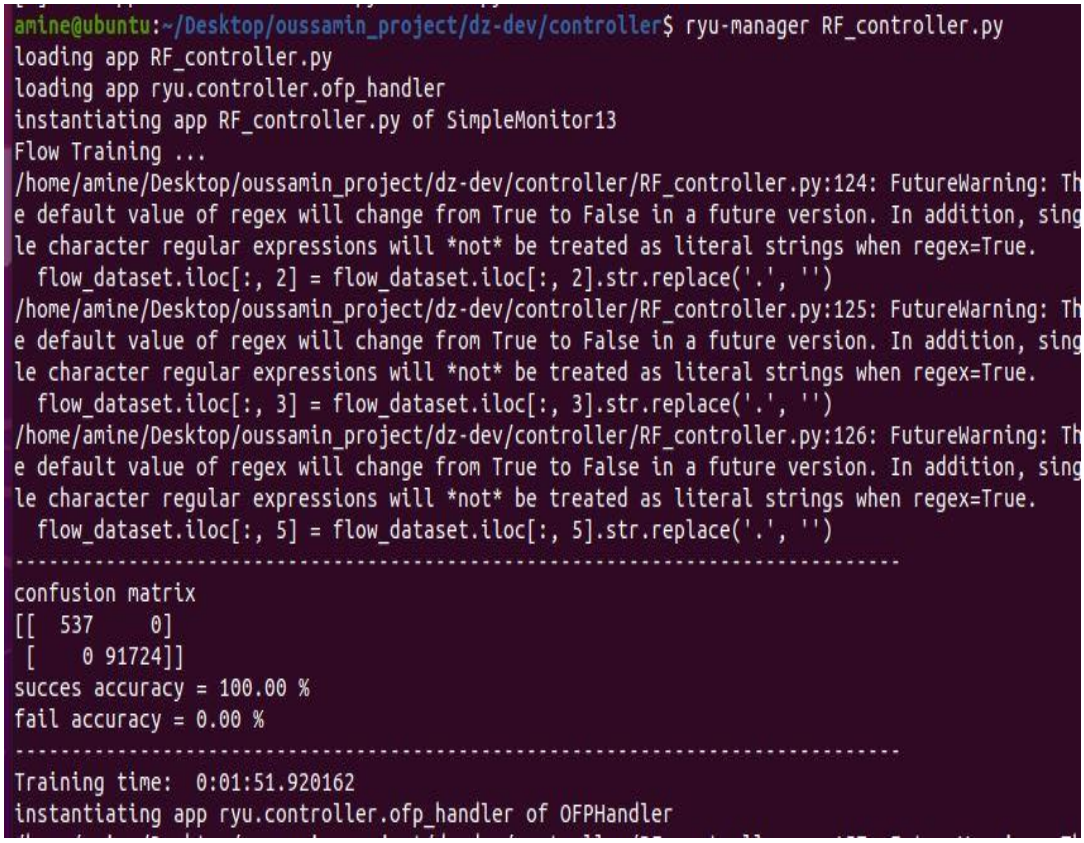


Figure III 23 La détection du ligitimate trafic

les résultats montrent que : le contrôleur donne la meilleure précision avec 100% de réussite

la topologie fonctionne bien, il suffit de faire le Pingall et on verra le résultat :

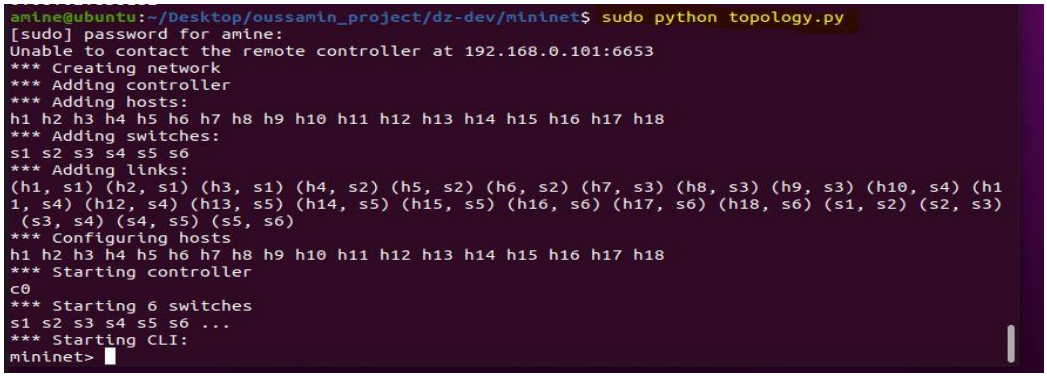


Figure III 24 pingall

le résultat nous montre que tous les paquets sont reçus et rien n'a été dropper.

Figure III 25 le résultat de pingall

nous allons ouvrir le terminal de l'hôte h1 avec la commande xterm et faire un simple ping vers l'hôte 18 (10.0.0.18) et voir si notre module de détection prédit correctement le trafic légitime dans la figure suivante :

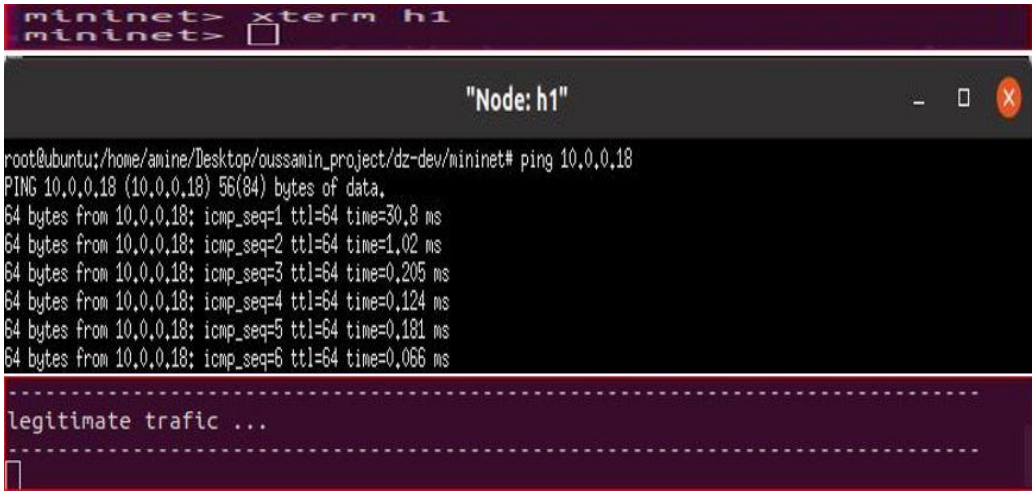


Figure III 26 faire un simple ping vers l'hôte 18

3.9.2 La détection du ddos trafic

De même façon nous allons ouvrir le terminal de l'hôte h3 avec la commande xterm et faire un hping3 vers l'hôte 9 (10.0.0.9) et voir si notre module de détection prédit correctement la propriété de trafic légitime ou ddos dans la figure suivante :

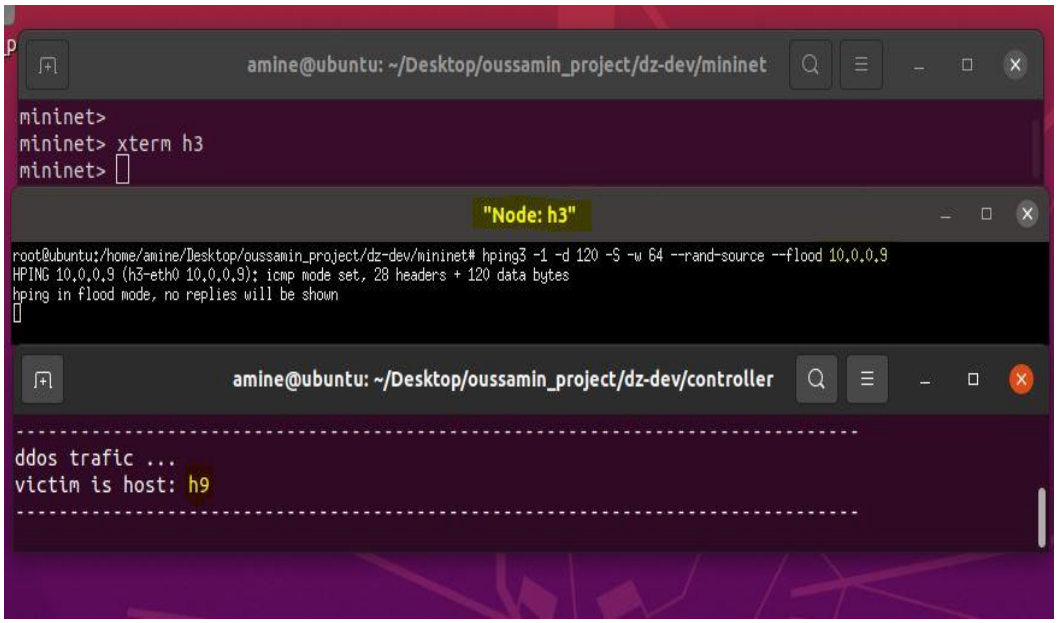


Figure III 27 La détection du ddos trafic

Et ici vous pouvez voir que notre mode de détection identifiera correctement l'attaque ainsi que le nom d'hôte h9. et nous obtenons que le système est très efficace et fonctionne comme prévu.

3.10 L'utilisation du CPU sous l'absence et la présence de l'attaque DDOS

De temps en temps, il y aura quelque chose qui ralentira un système. Il existe quelques outils qui peuvent aider à identifier quel processus est à l'origine de ce ralentissement. L'un de ces outils est htop.

Htop est une application de surveillance de processus interactive et en temps réel pour Linux qui vous montrera votre utilisation par processeur (CPU), ainsi qu'un graphique textuel significatif de votre utilisation de la mémoire et de l'échange.

Commençons à installer htop. Pour l'installer sur Ubuntu, nous exécutons la commande suivante dans Terminal

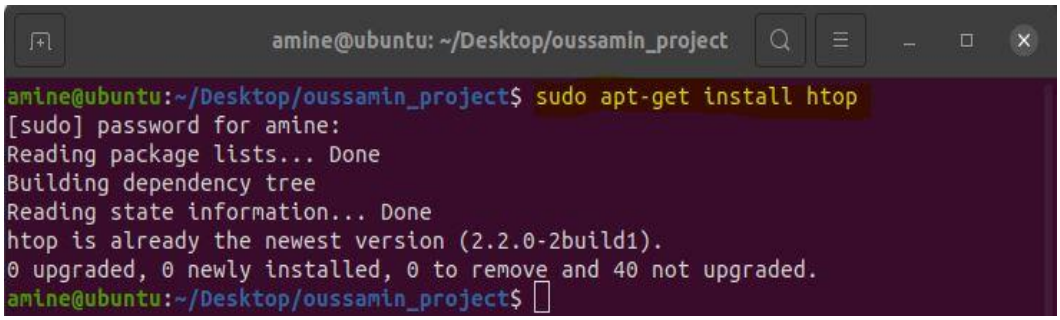
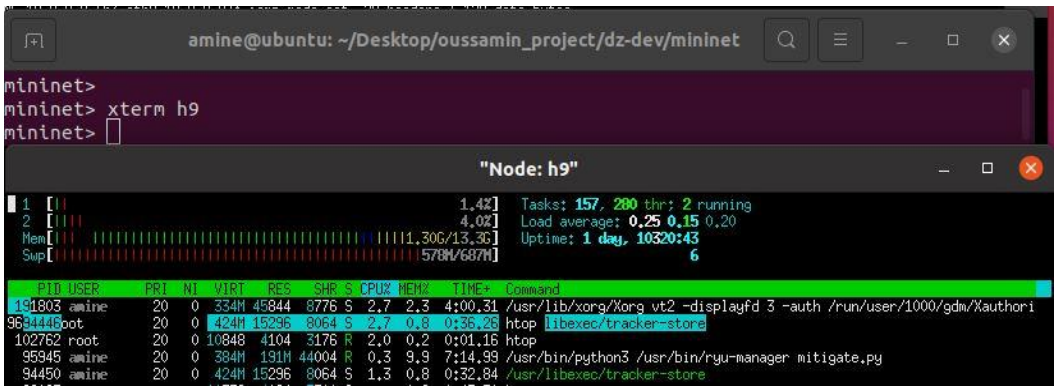


Figure III 28 installer htop

Htop est déjà installé

3.10.1 l'utilisation du processeur avant l'attaque

Après l'installation, nous tapons simplement htop dans le terminal pour l'exécuter et voici un exemple de l'hôte h9 sans l'affecter.



La figure III.29 Illustre l'utilisation du processeur avant l'attaque.

Le résultat montre que l'utilisation du processeur avant l'attaque était constante entre 1.4 % et 0.2 % donc le CPU en mode relax.

3.10.2 l'utilisation du processeur pendant l'attaque

le résultat de la figure **Error! No text of specified style in document.**-30 montre que l'utilisation du processeur pendant l'attaque a augmenté d'environ 22% en 2 secondes et est passée à 100% en 5 secondes follement, ensuite il est resté stable à 100%.

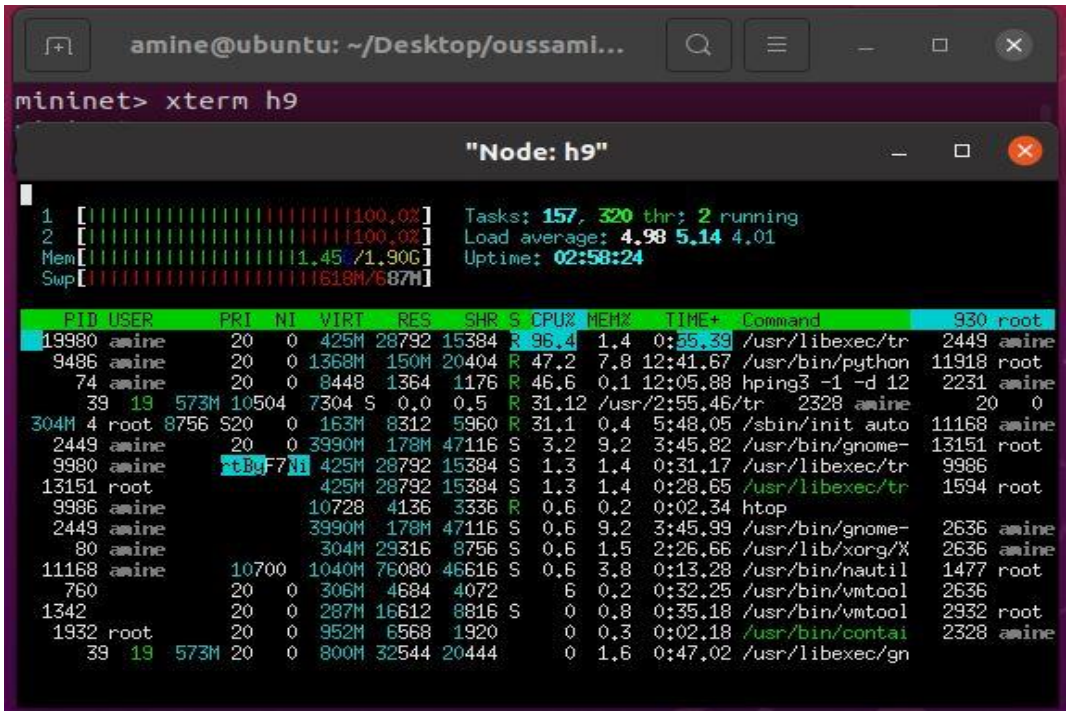


Figure III 30 l'utilisation du processeur pendant l'attaque

La figure présente les résultats sous forme de graphique linéaire et montre l'impact de l'attaque par inondation ICMP avant et pendant l'attaque

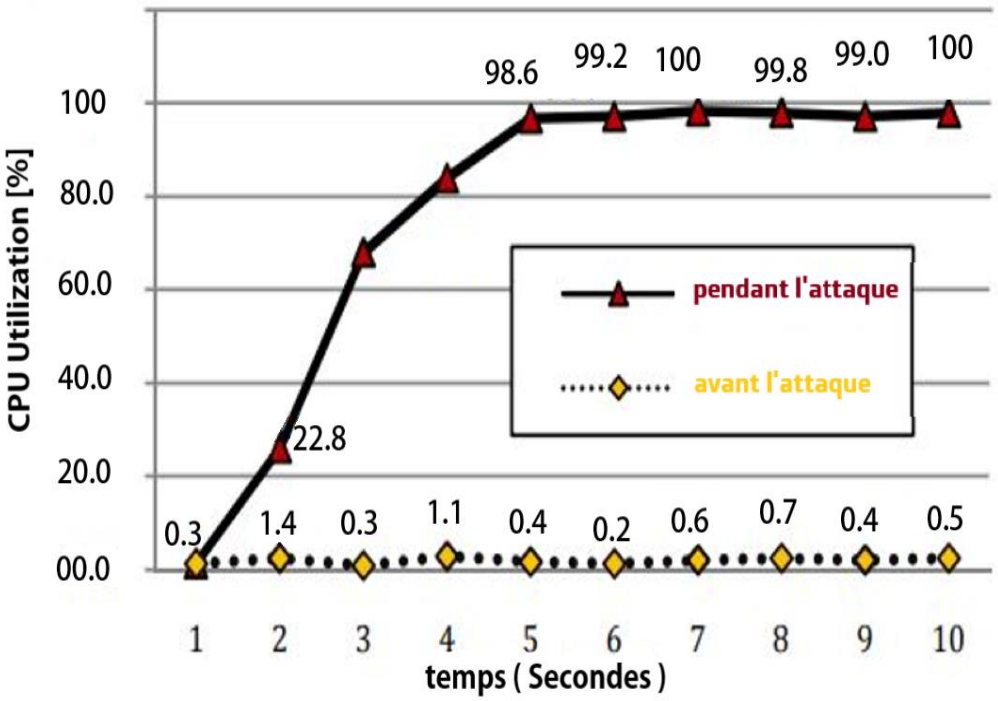


Figure III 31 L'utilisation du CPU sous l'absence et la présence de l'attaque DDOS

Htop est un outil fantastique qui est parfois négligé. Lorsque je crois qu'il y a des problèmes de performances avec un système, htop est l'un des premiers outils que j'utilise pour commencer l'enquête sur ce qui pourrait causer le problème.

3.11 L'intégration du modèle mitigate avec le contrôleur Ryu

Dans cette partie, nous intégrons notre modèle au contrôleur Ryu pour améliorer son efficacité à détecter les attaques DDoS en temps réel. Nous installons également des règles de flux sur le commutateur pour détecter le trafic anormal, de sorte que le contrôleur bloque l'hôte attaquant en supprimant ces paquets pour atténuer l'attaque. C'est l'objectif du script mitigate.py

```
194
195             self.logger.info("------------------------------------------------------------------------------")
196
197
198             if(legitimate_trafic /len( y_flow_pred )*100 ) > 80:
199                 self.logger.info ("legitimate trafic ...")
200             else:
201                 self.logger.info ("ddos trafic ...")
202                 self.logger.info ("victim is host: h{}".format(victim ))
203                 self.logger.info ("starting mitigation ...")
204
205
206
207
208             print("l'attaquant du commutateur est S"+ dpid )
209
210             cmd ="ovs_ofctladd_flows" + dpid +"dl_src="+ mac+",actions=drop"
211
212             passw ="user"
213             os.system ("echo %s|sudo-S %s" %(passw,cmdS))
214
215
216             self.logger.info ("tous les paquets de cette adresse mac"+ mac +"ont été supprimés du commutateur S"+ dpid )
217
218             self.logger.info("------------------------------------------------------------------------------")
```

Figure III 32 script mitigate.py

En exécutant le script mitigate.py en tant qu'application Ryu avec la topologie topology.py dans le mininet.

nous obtenons que le système est très efficace et fonctionne comme il était supposé.

Figure III 36 Exécution de l'application mitigate.py

Nous avons implémenté l'application d'atténuation des attaques DoS décrite sur notre algorithme et surveillé le paquet dans le message. Nous définissons le seuil à un paquet par seconde. Si le contrôleur reçoit plus d'un paquet par seconde, il est considéré comme une attaque DDoS et déconnectera le commutateur pour supprimer le trafic de l'attaquant externe. et ceci est représenté dans la figure suivante.

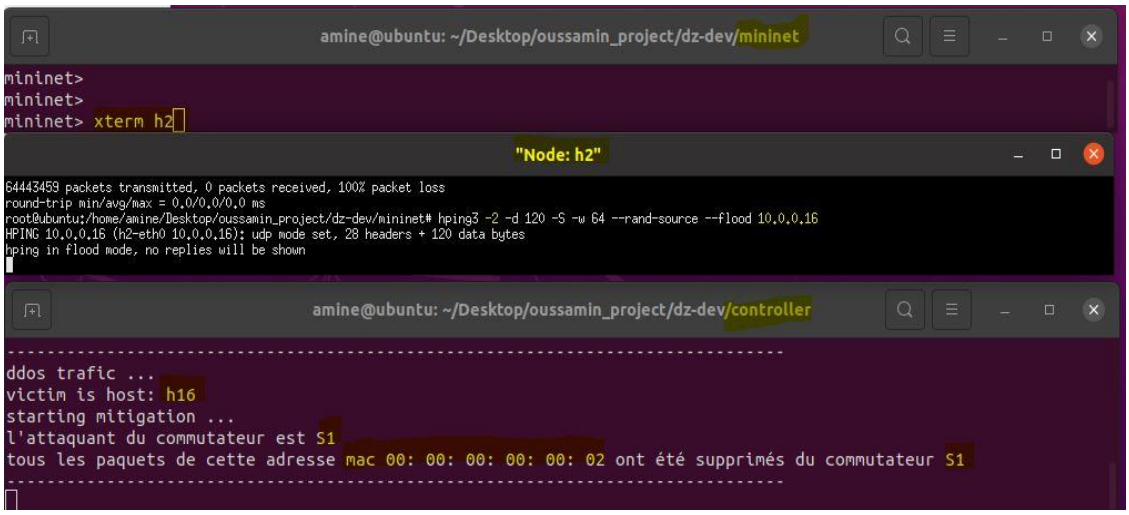


Figure III 37 Résultats indiquant la capacité du système à détecter et à atténuer les attaques DDoS.

3.12 **Conclusion**

Ce chapitre a proposé un mécanisme basé sur ML pour détecter les attaques DDoS et a démontré qu'il pouvait les détecter et les atténuer avec précision. De plus,

nous avons comparé les performances des différents classificateurs supervisés et avons trouvé que l'approche Random Forest Classifier donne un meilleur résultat.

Dans ce chapitre, nous améliorons la capacité du mécanisme proposé à détecter les attaques DDoS à l'aide d'un classificateur RF avec une précision de 100 %. Nous démontrons également l'importance de la topologie pour atténuer ces attaques dans le SDN, ce qui est une nouvelle contribution dans ce domaine. La simulation montre que la mitigation des attaques DDoS est efficace quelle que soit la topologie proposée.

Conclusion générale

Dans ce mémoire, nous avons proposé un mécanisme pour détecter et atténuer les attaques DDoS dans les réseaux SDN. Six algorithmes ML (LR, K-NN, NB, SVM, DT et RF) ont été testés et évalués à l'aide de l'ensemble de données synthétique . Les résultats montrent que NB donne la plus mauvaise précision avec 56,62%, LR et SVM donnent la même précision avec 87,61% de réussite. En second lieu, K-NN donne 99,47% de réussite, tandis que DT et RF donnent la meilleure précision avec 100% de réussite. Selon les résultats, le classificateur RF a été sélectionné en utilisant seulement cinq caractéristiques. Bien que cela vise à éviter le sur-ajustement du modèle et à réduire le temps du processus, les résultats de la réduction des fonctionnalités montrent que le temps était sept fois inférieur à l'utilisation des fonctionnalités complètes. Dans le même temps, la précision n'a pas été affectée, et cela est dû à la quantité de données utilisées.

Cette thèse démontre également que le type de topologie est important dans ce type d'attaque dans les réseaux SDN. Cela a été fourni en utilisant un réseau virtuel mininet pour configurer deux topologies : la topologie arborescente et la topologie linéaire. Ensuite, nous avons étudié l'effet de l'attaque DDoS sur les deux topologies à l'aide de l'outil Hping3. L'analyse des résultats indique que la connectivité entre commutateurs en topologie linéaire est de 66,66%, contrairement à la topologie arborescente qui donne 0% de connectivité entre commutateurs. Par conséquent, nous concluons que la topologie linéaire peut atténuer l'effet des attaques DDoS sur le SDN, contrairement à la topologie arborescente, qui représente un point de défaillance unique.

Bien que la topologie ne puisse pas satisfaire pour atténuer les attaques DDoS, nous proposons d'utiliser les informations collectées auprès de l'attaquant dans la phase de détection pour ajouter une règle de flux qui supprime tous les paquets correspondant

à l'adresse MAC de l'attaquant. Ce mécanisme prouve son efficacité pour prévenir l'attaque même avec une topologie arborescente.

Dans le cadre des futurs travaux de cette mémoire, et puisque la partie atténuation a certaines limitations, comme la capacité d'un attaquant à usurper son adresse MAC, ce qui signifie que la règle de flux sera incapable de bloquer l'attaquant. La pénalité étant sévère, l'hôte attaquant ne se connectera pas sauf si l'administrateur supprime cette règle de flux. Nous pensons étudier plus de temps pour résoudre ces problèmes dans notre système. Nous avons pensé à quelques solutions pour résoudre ces problèmes, mais le temps était l'un des défis pour y parvenir, nous le gardons donc pour de futures améliorations.

Bibliographie

[1]. **I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury, "**The dual-tree complex wavelet transform,". *IEEE signal processing magazine, vol. 22, no. 6, pp. 123–151, 2005.*

[2]. **M. V. Pawar and J. Anuradha,** "Network security and types of attacks in network," Procedia Computer Science, vol. 48, pp. 503–506, 2015.

[3]. **P. W. Dowd and J. T. McHenry,** "Network security: it's time to take it seriously," Computer, vol. 31, no. 9, pp. 24–28, 1998.

[4]. **Sotomayor and L. Childers, "**Fundamental security concepts," Globus Ttoolkit 4: Pro- gramming Java Services, pp. 257–269, 2005.

[5]. **D. Branstad, "**Encryption protection in computer data communications," in 4th Data Com- munications Symposium, October 7-9, 1975 et Quebec City, Quebec, Canada, pp. 8–1, IEEE, 1975.

[6]. **Branstad, "**Encryption protection in computer data communications," in 4th Data Com- munications Symposium, October 7-9, 1975 et Quebec City, Quebec, Canada, pp. 8–1, IEEE, **1975.**

[7]. **K. Salah, K. Elbadawi, and R. Boutaba,** "Performance modeling and analysis of network firewalls," IEEE Transactions on network and service management, vol. 9, no. 1, pp. 12–21, 2011.

[8]. **X. Liang and Y. Xiao, "**Game theory for network security," IEEE Communications Surveys & Tutorials, vol. 15, no. 1, pp. 472–486, 2012.

[9]. **H. BOUIDA,** « étude et mise en œuvre d'une solution SDN : application de gestion de vlans », mémoire de PFE master, université ABOU BAKR BELKAID– TLEMCEN, 2017.

[10]. **Documentation propre à ERICSSON, WDM fundamentals, 2011.**

[11]. **A. Voellmy, H. Kim, and N. Feamster,** "Procera: a language for high-level reactive network control," in Proceedings of the first workshop on Hot topics in software defined networks, pp. 43–48, 2012.

[12]. **N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "**Nox: to- wards an operating system for networks," ACM SIGCOMM computer communication review, vol. 38, no. 3, pp. 105–110, 2008.

[13]. **T. Ubale and A. K. Jain,** "Survey on ddos attack techniques and solutions in software-defined network," in Handbook of computer networks and cyber security, pp. 389–419, Springer, 2020.

[14]. **Y. Jarraya, T. Madi, and M. Debbabi,** "A survey and a layered taxonomy of software-defined networking," IEEE communications surveys & tutorials, vol. 16, no. 4, pp. 1955–1980, 2014.

[15]. **M. ALTUFAILI,** « intelligent network bandwidth allocation using SDN », international journal of engineering research & technology, volume. 4 – issue. july 2015.

[16]. **V. Thirupathi, C. Sandeep, N. Kumar, and P. Kumar,** "A comprehensive review on sdn archi- tecture, applications and major benifits of sdn," International Journal of Advanced Science and Technology, vol. 28, no. 20, pp. 607–614, 2019.

[17]. **S. ZNATY,** « le protocole OpenFlow dans l'architecture SDN », efort editions, 2016.

[18]. OPEN NETWORK FOUNDATION, « OpenFlow switch specification », version 1.4.0, October 2013.

[19]. **V. CLOAREC,** « le projet OpenFlow », mémoire de PFE MASTER, TELECOM LILLE 1, 2013.

[20]. **M. TURY,** « les risques d'openflow et du SDN », ANSSI, 2014.

[21]. **M. Eslahi, R. Salleh, and N. B. Anuar,** "Bots and botnets: An overview of characteristics, detection and challenges," in 2012 IEEE International Conference on Control System, Com- puting and Engineering, pp. 349–354, IEEE, 2012.

[22]. **C. Li, W. Jiang, and X. Zou,** "Botnet: Survey and case study," in 2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC), pp. 1184–1187, IEEE, 2009.

[23]. **J.-S. Lee, H. Jeong, J.-H. Park, M. Kim, and B.-N. Noh, "**The activity analysis of malicious http-based botnets using degree of periodic repeatability," in 2008 International Conference on Security Technology, pp. 83–86, IEEE, 2008.

[24]. **R. Vishwakarma and A. K. Jain,** "A survey of ddos attacking techniques and defence mech- anisms in the iot network," Telecommunication systems, vol. 73, no. 1, pp. 3–25, 2020.

[25]. **J. Mirkovic and P. Reiher,** "A taxonomy of ddos attack and ddos defense mechanisms," ACM SIGCOMM Computer Communication Review, vol. 34, no. 2, pp. 39–53, 2004.

[26]. **W. Braun and M. Menth,** "Software-defined networking using openflow: Protocols, applica- tions and architectural design choices," Future Internet, vol. 6, pp. 302–336, 05 2014.

[27]. **A. Asosheh and N. Ramezani,** "A comprehensive taxonomy of ddos attacks and defense mechanism applying in a smart classification," WSEAS Transactions on Computers, vol. 7, no. 4, pp. 281–290, 2008.

[28]. **A. Bhardwaj, G. Subrahmanyam, V. Avasthi, H. Sastry, and S. Goundar**, "Ddos attacks, new ddos taxonomy and mitigation solutions—a survey," in 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), pp. 793– 79.

[29]. **S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen,.**

[30]. **S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali,** "On scalability of software-defined net- working," IEEE Communications Magazine, vol. 51, no. 2, pp. 136–141, 2013.

[31]. **M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia,** "Modeling and performance evaluation of an openflow architecture," in 2011 23rd International Teletraffic Congress (ITC), pp. 1–7, IEEE, 2011.

[32]. **Y. Jarraya, T. Madi, and M. Debbabi,** "A survey and a layered taxonomy of software-defined networking," IEEE communications surveys & tutorials, vol. 16, no. 4, pp. 1955–1980, 2014.

[33]. **S. Scott-Hayward, G. O'Callaghan, and S. Sezer,** "Sdn security: A survey," in 2013 IEEE SDN For Future Networks and Services (SDN4FNS), pp. 1–7, IEEE, 2013.

[34]. **Y.-W. Chen, J.-P. Sheu, Y.-C. Kuo, and N. Van Cuong,** "Design and implementation of iot ddos attacks detection system based on machine learning," in 2020 European Conference on Networks and Communications (EuCNC), pp. 122–127, IEEE, 2020.

[35]. **M. Eslahi, R. Salleh, and N. B. Anuar,** "Bots and botnets: An overview of characteristics, detection and challenges," in 2012 IEEE International Conference on Control System, Com- puting and Engineering, pp. 349–354, IEEE, 2012.

[36]. **I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani,** "Developing realistic dis- tributed denial of service (ddos) attack dataset and taxonomy," in 2019 International Car- nahan Conference on Security Technology (ICCST), pp. 1–8, IEEE, 2019.

[37]. **M. Xia, W. Lu, J. Yang, Y. Ma, W. Yao, and Z. Zheng,** "A hybrid method based on extreme learning machine and k-nearest neighbor for cloud classification of ground-based visible cloud image," Neurocomputing, vol. 160, pp. 238–249, 2015.

[38]. **M. A. M. Yusof, F. H. M. Ali, and M. Y. Darus,** "Detection and defense algorithms of different types of ddos attacks," International Journal of Engineering and Technology, vol. 9, no. 5, p. 410, 2017.

[39]. **R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno,** "Machine learning algorithms to detect ddos attacks in sdn," Concurrency and Computation: Practice and Experience, vol. 32, no. 16, p. e5402, 2020.

[40]. **HABIB, MAKI K,** "Revolutionizing Education in the Age of AI and Machine Learning",.

[41]. RAKESH D, "Artificial Intelligence for Managers: For Individuals Aspiring to Get into the AI Domain", Edition Notion Press, pp 204, 2020.

[42]. **JUSTUS W,** "Machine Learning Asset Valuation", GRIN Verlag, pp 148, 2017.

[43]. **https://medium.com/@parthvadhadiya424/hello-world-program-with-ai-artificial.**

[44]. **(2020.03.20),** https://www.linkedin.com/pulse/briefing-machine-learning-algorithms-aditya-bikram- dash/.

[45]. **I. A. A. Amra and A. Y. Maghari,** "Students performance prediction using knn and na ¨ıve bayesian," in 2017 8th International Conference on Information Technology (ICIT), pp. 909– 913, IEEE, 2017.

[46]. **A. Chapman, E. Simperl, L. Koesten, G. Konstantinidis, L.-D. Ib´an˜ez, E. Kacprzak, and.**

[47]. **https://www.lawinsider.com/dictionary/real-data., Real data definition.".**

[48]. **]R. Heyburn, R. R. Bond, M. Black, M. Mulvenna, J. Wallace, D. Rankin, and B. Cleland,** "Machine learning using synthetic and real data: similarity of evaluation metrics for different healthcare datasets and for different algorithms," in Data Science and K.

[49]. **M. A. Hall and L. A. Smith,** “Feature selection for machine learning: comparing a correlation- based filter approach to the wrapper.,” in FLAIRS conference, vol. 1999, pp. 235–239, 1999.

[50]. **T. Ubale and A. K. Jain,** “Survey on ddos attack techniques and solutions in software-defined network,” in Handbook of computer networks and cyber security, pp. 389–419, Springer, 2020.

[51]. **Q. Yan, F. R. Yu, Q. Gong, and J. Li,** “Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges,” IEEE communications surveys & tutorials, vol. 18, n.

[52]. **T. Ubale and A. K. Jain,** “Taxonomy of ddos attacks in software-defined networking envi- ronment,” in International Conference on Futuristic Trends in Network and Communication Technologies, pp. 278–291, Springer, 2018.

[53]. **B. Wang, Y. Zheng, W. Lou, and Y. T. Hou,** “Ddos attack protection in the era of cloud computing and software-defined networking,” Computer Networks, vol. 81, pp. 308–319, 2015.

[54]. **Q. Yan and F. R. Yu,** “Distributed denial of service attacks in software-defined networking with cloud computing,” IEEE Communications Magazine, vol. 53, no. 4, pp. 52–59, 2015.

[55]. **J. M. Dover,** “A denial of service attack against the open floodlight sdn controller,” Dover Networks LCC, Edgewater, MD, USA, 2013.

[56]. **C. E. Shannon,** “A mathematical theory of communication,” The Bell system technical jour- nal, vol. 27, no. 3, pp. 379–423, 1948.

[57]. **M. P. Singh and A. Bhandari,** “New-flow based ddos attacks in sdn: Taxonomy, rationales, and research challenges,” Computer Communications, vol. 154, pp. 509–527, 2020.