PROACTIVE MITIGATION OF DDoS IoT-RELATED ATTACK USING MACHINE LEARNING AND SOFTWARE DEFINED NETWORKING TECHNIQUES

Emmanuel J. Ebong[1], Samuel N. John[1], Dominic S. Nyitamen[1] and Samuel F. Kolawole[1]

[1] Department of Electrical/Electronic Engineering, Nigerian Defence Academy, Kaduna, Nigeria

Abstract

The number of Internet of Things (IoT) connected to the Internet have increased globally. The insecure nature of IoT have made attackers to capitalize on the devices to launch Distributed Denial of Service (DDoS) attacks on networks, thus causing massive destruction to network resources. The setting of the research work is an enterprise organization wide area network (WAN) that is structured into 3 LANs topology in Software Defined Networking (SDN) environment. The WAN is emulated, and includes a single RYU SDN controller, three routers, three OpenFlow switches with three simulated IoT devices connected to each switch, to form the 3 LANs topology. Both normal and DDoS IoT-related attack data traffics are generated every 5 seconds, from Transport Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP) and Hypertext Transfer Protocol (HTTP). The packets capture (pcap) files from Wireshark are exported as comma-separated values (csv) files. The datasets are preprocessed to extract relevant features using Python libraries. The large dataset was scaled down using Min Max Scaler before the Machine Learning (ML) classification stage. Four (4) ML algorithms namely, Support Vector Machine (SVM), Logistic Regression (LR), Decision Tree (DT) and Random Forest (RF) were used to classify the models. The performances of SVM and LR recorded higher percent accuracy of 99.474 each while the DT and RF recorded 99.123 percent accuracy each in detecting the DDoS-IoT data traffic from the normal data. The flow table entries (FTE) rules of the OpenFlow switches together with the RYU controller mitigated the DDoS attack traffic, to drop the attack data packets and block the affected ports within 10 seconds and less with more tries.

Keywords: DDoS attacks, IoT devices, ML classification, SDN environment, RYU Controller, FTE rules of OpenFlow switches

Corresponding author's e-mail: ejebong@nda.edu.ng

1.0 INTRODUCTION

There were reports of surges of DDoS IoT-botnet related attacks on the Internet infrastructure in 2016 (Garba et al., 2024). The reports included the Mirai malware attack against Dyn Domain Name System (DNS) provider by over 150,000 IoT devices in October 2016 (Garba et al., 2024; Bhayo J., 2023). The attacks rendered many websites such as GitHub, Amazon, Netflix, Twitter, CNN, Spotify, and PayPal inaccessible for several hours. The attacks highlighted the risks due to inadequate security mechanisms in IoT devices and the devastating effects on the Internet. Other reports also indicated sharp increases in DDoS attacks targeting the healthcare industry during the COVID-19 global pandemic in 2020 (Singh & Jain., 2024). The attacks were particularly higher in the second quarter of 2020 compared to the first quarter of the same year. According to Neustar, IoT devices made up about 15 percent of all the DDoS attacks, which was higher than the 10 percent involvement recorded in 2019. Further attacks have also been recorded. For instance, Amazon Web Services was attacked by a massive DDoS in 2020 measuring 2.3Tbps while Yandex, a Russian Internet giant, was confronted by a huge DDoS attempt with 21.8 million requests per second (Kumari & Jain., 2023). The Yandex attack which lasted a few weeks was recorded from 7th August 2021 to 15th September 2021.

The most targeted industry for DDoS attacks has remained the financial services sector, which have accounted for 25 percent of all attacks in the first quarter of 2021 (Singh & Jain., 2023). Also, according to Akamai Technologies, DDoS attacks propagated through IoT devices increased by 62 percent in 2022 compared to 2021 (Singh & Jain, 2024; Singh & Jain, 2023). Further report by NetScout (Gelgi et al., 2024), indicates that the frequency and intensity of the attacks have increased in 2023, rising from an average of 144 daily attacks at the start of the year to 611 by the end of June, an increase of approximately 353 percent. Meanwhile, the number of IoT devices connected globally continue to increase while the worldwide spending on security of IoT has also been on the increase.

A DDoS attacker requires means such as a script, hack or tool to reach the target system such as server facility, and to force it into failure condition, thereby denying its services to other users (Alshammari & Alserhani, 2022). Other DoS attacks exploit vulnerabilities in the computer networks technology, involving the TCP/Internet Protocol (IP) and Operating Systems (OS). Further attacks could be due to system configuration that are often unsecured, and lack of written security policy for the organization.

IoT devices such as wireless sensors, software and actuators, can be assigned IP addresses, connect wirelessly to conventional or SDN networks, and transmit data. However, the biggest worry with IoT devices lies in security and their vulnerabilities that can be exploited for DDoS attacks. IoT vulnerabilities include insecure web interfaces, insufficient authentication and authorization, insecure software and network services, lack of transport encryption, weak physical security, and privacy concerns (Singh et al., 2024).

In conventional networks, DDoS attacks can be classified as Network/Transport and Application-levels DDoS Attacks, based on the protocol level or layer of the Open System Interconnection (OSI) model targeted (Anusuya et al., 2023). The Network/Transport-level DDoS attacks are flooding in design to disrupt a legitimate user's connectivity by exhausting bandwidth, router processing capacity or network resources. They are launched using TCP, UDP, ICMP, and DNS protocol packets. The Application-level DDoS

attacks disrupt legitimate user's services by exhausting the server resources such as Sockets, Central Processing Unit (CPU) memory, database and input/output (I/O) bandwidths. The attacks target HTTP, DNS etc. The greatest threat of DDoS attacks is TCP SYN flooding, which accounts for about 85 percent of attacks. Thus, networks must prepare to detect and mitigate the assaults.

SDN makes the network more flexible and easier to manage by decoupling the control plane from data plane (Alashhab et al., 2024). Thus, SDN control and manage the whole network from a centralized remote controller location. However, SDN is also susceptible to DDoS botnet attacks, which can exhaust the SDN component resources. The main attacks against the SDN networks can be Data Plane, Control Plane Communication, SDN Controller and Application Plane Attacks.

The problem of DDoS attack is a significant one as such attack in SDN environment may target any host IoT device in the data plane to exhaust the resources (Wang., et al 2024). The attack could extend to disrupt the hosts server (victim) by exhausting its resources and rendering it incapable to provide service to legitimate users for hours. There is therefore justification and motivation to do this research work, which aim to investigate further into the techniques of using ML algorithms to detect DDoS attack on any host IoT device or hosts server and SDN flow rules to proactively mitigate the attack in enterprise organization WAN. The techniques used are based on features of the attack data packets and flow characteristics.

The research methodology was limited to using only the necessary hardware and software tools to emulate the required network devices and simulation of the IoT devices. The attack data traffics were originated from any of the hosts IoT devices within the WAN with exception of the host target (victim), and comprised only floods of TCP, UDP, ICMP and HTTP packets.

The literature review forms Section 2, which consist mainly the theoretical background. Section 3 is the related works and includes gaps in knowledge. Section 4 is the methodology which highlights the research methods, research materials and implementation of the DDoS IoT-related attack, including the detection using ML models and mitigation using SDN techniques. Section 5 highlights the results and discussion of the work while Section 6 is the conclusion.

2.0 LITERATURE REVIEW

The DDoS attack uses a collection of compromised machines (zombies) forming a botnet, under a command-and-control (C&C) infrastructure, to attack a target simultaneously from multiple locations on the Internet (Abhishek., 2023). DDoS botnet consists of 4 elements, namely attacker, C&C (handlers), zombies (bots) and victims (targets) as in Fig 2.1. The attacker communicates with the handlers to identify the bots, which include IoT devices (Abhishek., 2023; Wani et al., 2021). DDoS attacks utilize IP spoofing to conceal their own IP address.

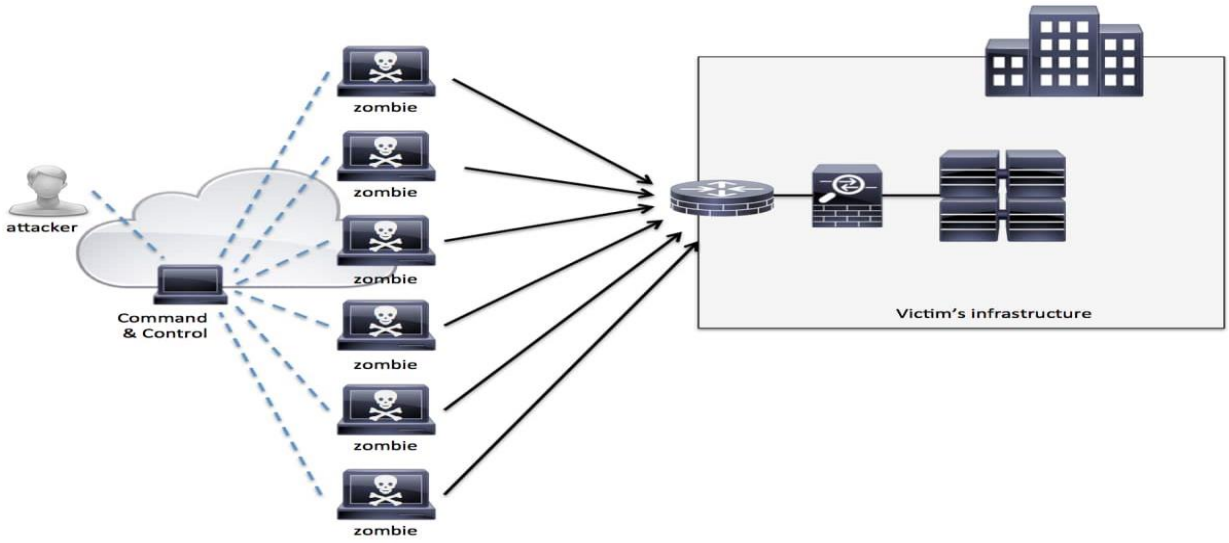

Fig 2.1: Botnet-based DDoS Attack Architecture (Wani et al., 2021).

IoT Devices

Using the IoT, physical objects can be empowered to create, receive, and exchange data in a seamless manner, without any human intervention (Karmous et al., 2024). The connectivity, networking and communication protocols used depend on the specific IoT deployment (Negera et al., 2022). Deployment of IoT devices can be categorized into consumer, enterprise and industrial.

i IoT Key Technologies. Among the key technologies of IoT are Radio frequency identification (RFID), Bluetooth, Wi-Fi, ZigBee, Nanotechnology, Tagging technologies like Near Field Communication (NFC), Actuators and Wireless Sensor Networks (WSN) (Pakmehr et al., 2024). RFID technology is a foundational for IoT, which allows microchips to transmit the identification information to a reader through wireless communication.

ii IoT Architecture**.** An IoT architecture which addresses scalability, reliability and interoperability is proposed in (Jenny & Sugirtham, 2023). The architecture consists of the first four common layers, namely Perception, Network, Support and Application and the fifth layer, which is Business layer.

iii IoT Elements. The IoT six main elements for functionality are identification, sensing, communication, computation, services, and semantics, according to (Almadhor et al., 2024).

iv IoT Common Protocols. The IoT four common protocols are application, service deliver, infrastructure and other influential protocols as related to DoS attacks (Almadhor et al., 2024).

v IoT Security Vulnerabilities. Various IoT security threats that involve DoS/DDoS attacks are highlighted in (Jenny & Sugirtham, 2023; Aslam et al., 2022) and includes: (1) IoT devices have default software configuration, irregular updates of software installed, and default login credentials' vulnerability. (2) IoT devices lack conventional interfaces such as keyboards, mice, and touchscreen, hence cannot authenticate and authorize users in familiar ways. (3) IoT devices have low power, less storage capacity, low memory and less processing capability, which do not allow for intricate security strategy. (4) Intricacies of IoT systems involving many

devices, apps, service, communication protocols.

vi IoT Traffic Features. Some of the IoT traffic features include: (1) IoT traffic is often distinct from that of other Internet connected devices (Alatram et al., 2023). (2) IoT devices have repetitive network traffic patterns, with small packets at fixed time intervals for logging purposes. (3) IoT devices tend to have a fixed number of states, so their network activity is more predictable. (4) IoT devices use wireless medium to broadcast data which makes them an easier target for an attack (Kumar & Arul, 2023). (5) IoT traffic features must be lightweight, to enable routers handle high bandwidth and process packets from TCP, UDP and HTTP protocols. (6) IoT device connected to the router or switch can send both normal and DoS/DDoS traffics within same time period.

ML Techniques

ML is intended towards data computation and needs large amount of data for training, that includes repetitive training to refine the ability of learning, decision-taking (testing) and prediction (Butt et al., 2024). The techniques to select are supervised, unsupervised, semi-supervised, and reinforcement learnings (Butt et al., 2024). The ML supervised learning algorithm will create a prediction function using the training data that will generalize for unseen training vectors to classify them correctly as normal or DDoS attack data (Hassan et al., 2024; Liu et al., 2023).

The ML algorithms used for the data analysis are as highlighted.

i Support Vector Machine. To generalize in SVM, the data will have two classifications, positive and negative groups, as separated by a hyperplane (Alwabisi et al., 2022).

ii Decision Tree. In DT, the source data is reduced into a predictor tree (Praba & Sridaran, 2022).

iii Random Forest. RF predicts by averaging the predictions of each component tree (Sanjeetha et al., 2022).

iv Logistic Regression. LR finds a relationship between features (X) and probability of outcome (response variable Y = 0 (normal) and Y = 1 (DDoS anomaly) (Altamemi et al., 2022).

Metrics such as Confusion Matrix, Accuracy, Precision, Recall, F1 Score, ROC and AUC were used to evaluate the performances of the developed system models.

SDN Techniques

The SDN basic principle involves decoupling of control and data planes, programmability of network application services, and logically centralized control (Clinton et al., 2024; Hill et al., 2024). DDoS attacks in SDN can be classified into four (**1**,**2**,**3** and **4**) vectors (Elsayed et al., 2020) as in Fig 2.2.

i Data Plane Attacks. The DDoS attacker (**1**) can attack hosts in the data plane. The attacker (**1**) can also manipulate the FTE rules of OpenFlow switch in the data plane to reroute legitimate traffic.

ii Control Plane Communication Attacks. Spoofed flooding DDoS attack (**2**) can cause congestion in the Southbound OpenFlow communication links, which may result in breakdown between the SDN controller and data plane elements. DDoS attack (**2**) can also cause congestion in the Northbound RESTful communication links between the SDN controller and the application plane elements.

iii SDN Controller Attacks. DDoS attack (**3**) can bring down the SDN controller, resulting in whole system disruption.

iv Application Plane Attacks. The DDoS attack (**4**) can violate the security policy, or bypass any installed firewall and intrusion detection system (IDS) apps.

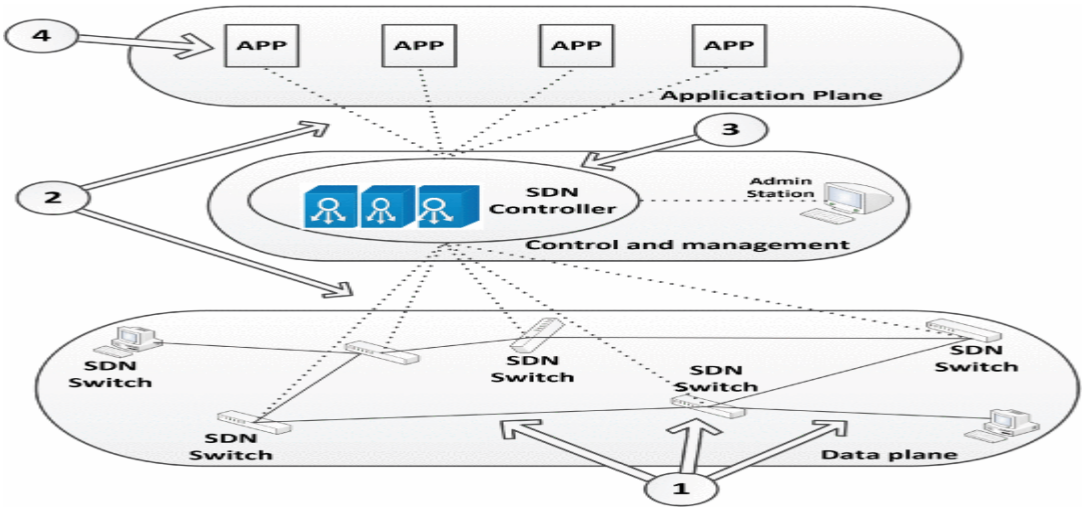

Fig 2.2: Main Attacks Targeting the SDN Networks (Elsayed et al., 2020).

Attacks (**1**) and (**4**) in Fig 2.2 are common in both SDN and conventional networks. Attacks (**2**) and (**3**) are specific to SDN.

The SDN controller typically runs on a server and uses protocols to tell switches where to send packets (Sudar & Deepalakshmi, 2020). An SDN model must capture three phases as in Fig 2.3. Phase 1, the first packet of a flow arrives at the switch and there is no matching FTE for the packet. Phase 2, the packet without a matching FTE is forwarded to the controller or a packet with matching FTE is serviced by the switch and forwarded to the destination. Phase 3, the controller feeds the forwarding information back to the switch and updates the flow table in the switch.

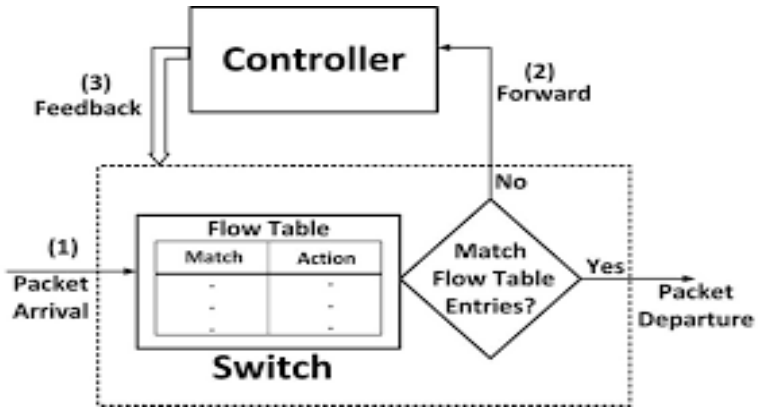

Fig 2.3: Generic Block Diagram of a SDN Controller using Protocols to tell Switches where to Send Packets.

3.0 RELATED WORKS

This section reviewed DDoS attack detection and mitigation in related research works in SDN environment, without and with ML techniques. The related literatures reviewed that used only SDN techniques with various methods to detect and mitigate DDoS attacks

are analyzed in Table 3.1. The algorithms utilized for the researches in (Wang et al., 2021; Valizadeh & Taghinezhad-Nair, 2022; Galeano-Brajones et al., 2020) are based on features of SDN OpenFlow Virtual Switches (ovs), their respective Raspberry Pi 3 single controller, Floodlight multi-controller, and single RYU controller topologies, using python scripts for the simulations. The mitigation in (Wang et al., 2021) is implemented based on entropy for detection and on FTE rules of the SDN switches and controller for mitigation. The research work in (Wang et al., 2021) also utilize attack protocols that comprise TCP and UDP. Also, (Valizadeh & Taghinezhad-Nair, 2022) achieved high detection for both single and multi-victim (controller) attack scenarios. For (Galeano-Brajones et al., 2020) which also comprise TCP and UDP attack protocols, it was detected by correlation of entropy values of different features while the mitigation also utilized switch FTEs. (Galeano-Brajones et al., 2020) in its future work plans to introduce ML technique.

Table 3.1: Analysis of Reviewed Literatures on DDoS Detection and Mitigation using SDN Techniques only.

Author and Year	Objective	Model Type	Algorithms Type /Simulation	Attack Types /Protocols	Application/Result/ Limitation/Future work
Wang et al, 2021	**Mitigation** SECOD plane algorithm to resist DDoS attacks on real SDN-based IoT testbed.	Classification	SECOD algorithm. **Simulation-** Ubuntu OS, Raspberry Pi 3 Controller, 60 hosts, 30 OpenFlow switches.	TCP, UDP	Predefined policies converted to flow entries. **Limitation**: No Application-level attack. **Future**: Study the behavior of IoT subscribers.
Valizadeh, & Taghinezhad-Niar, 2022	**Detection** Entropy framework for detecting DDoS attacks and a fault-tolerant method to replace faulty leader controller in distributed multi-controller SDN.	Classification	**Simulation** – Mininet Emulator, 3 Floodlight Controllers, Ubuntu 16.04 OS.	Not indicated	**Result**: **Single victim** attack detection rate is **98.34** percent. **Multi-victim** attack detection rate is **95** percent. **Future**: To investigate several protection methods during attack detection at an SDN-enabled wide area measurement system.
Galeano-Brajones et al., 2020	**Detection** and **Mitigation** An Entropy-based solution to detect and mitigate DoS/DDoS attacks in IoT scenarios.	Classification 3 testbeds: Bandwidth consumption, DoS, DDoS attacks.	Bandwidth, Window Size **Simulation:** RYU controller, 3 hosts, 1 ovs switch, Mininet.	TCP, UDP	**Application**: Using the correlation of the entropy values of different features to detect the attack. Mitigation by FTEs. **Future:** Introducing the ML technique.

Table 3.2 presents the analysis on related literatures reviewed that used combined techniques of ML and SDN to detect and/or mitigate DDoS IoT-related attacks in networks. In the experimental testbeds, the SDN single controller topologies are simulated using Mininet to create virtual machines (VMs), hosts, switches, routers and controllers (Singh, 2021; Sanjeetha et al., 2021; Ravi & Shalinie, 2020; Singh, 2020; Mohsin & Hamad, 2022). The research works in (Ravi & Shalinie, 2020) and (Singh, 2020) use mostly features of incoming packets to detect and mitigate DDoS attacks in the combined ML and SDN techniques while (Al-Fayoumi & Al-Haija, 2023) in addition use MQTT protocol for easy detection and capture of the low-rate (LR) DDoS. Also, (Singh, 2021) and (Sanjeetha et al., 2021) in their works employed FTE rules to obtain higher detection accuracies and mitigation of the DDoS IoT-related attacks in SDN networks. However (Al-Fayoumi & Al-Haija, 2023) in its proposed future work hope to use FTE rules to detect attack in the SDN plane.

Except (Sanjeetha et al., 2021; Ravi & Shalinie, 2020; Al-Fayoumi & Al-Haija, 2023; Mohsin & Hamad 2022), the researches use SVM and other ML algorithms to classify and detect DDoS attacks. The consistency in the use of SVM proves the algorithm provides high-level performance in detection accuracy. The use of DT and RF algorithms were next in detection accuracy as evident in (Singh, 2021) and (Sanjeetha et al., 2021). Furthermore, all the researches use attack protocols that comprise any number of TCP, UDP, ICMP or HTTP floods except (Al-Fayoumi & Al-Haija, 2023), which use MQTT protocol. In particular, (Sanjeetha et al., 2021) use HTTP attack type. The use of RYU controller was common where more ML algorithms were employed in the testbeds to achieve higher accuracies as in (Singh, 2021; Sanjeetha et al., 2021; Singh, 2020; Mohsin & Hamad 2022) introduced linear and multi-controller topologies in its testbeds.

Table 3.2: Analysis of Reviewed Literatures on DDoS Detection and Mitigation using both ML and SDN Techniques.

Author and Year	Objective	ML Type or Model Used	Algorithms Used and Simulation	Attack Types /Protocols	Application/Result/ Limitation/Future work
Singh, 2021	**Detection** DDoS dataset in SDN that utilized the packet's library and port statistic request to extract 25 features in all.	Classification	ML - SVM (Linear and K-RBF), KNN, DT, RF, MLP, GNB. **Simulation** - Mininet, topology with 3 switches, 1 RYU Controller, 6 hosts.	ICMP, TCP UDP floods.	SVM - most efficient for identifying DDoS attack. **Accuracy**, precision, and recall approx **100** percent. **Limitation**: No switch flow rules. **Future**, Analyze the significance of DL method to detect similar attacks.
Sanjeetha et al., 2021	**Detection and Mitigation** SDN application written in python to detect DDoS attack using ML model and mitigate the attack using flow rules.	Supervised Classification	6 ML models: Catboost, XGBoost, DT, LR, GNB, and KNN. **Simulation** – Mininet, 1 RYU controller, hosts, switches and links.	HTTP, UDP, smurf.	Catboost - has less prediction time, less training time, detect DDoS with accuracy of **98** percent. **Limitation**: SVM not used. **Future** Test the detector-mitigator module for other types of attacks such as HTTP floods, Ping floods.
Ravi & Shalinie, 2020	**Detection and Mitigation** LEDEM leverages the cloud and SDN to mitigate DDoS attack on IoT servers based on the feature of incoming packets.	Semi-supervised ML algorithm.	LEDEM has a hierarchical control plane. Divides IoT devices into two categories: fixed (fIoT), moving (mIoT).	Not indicated	Local controller - manages a small part of the network. Universal controller -manages all the local controllers. **Accuracy** rate of **96.28** percent in detecting DDoS.
Singh, 2020	**Detection and Mitigation** Statistical and ML methods to detect and mitigate DDOS attacks in SDN.	Classification	ML – SVM **Simulation** - Mininet, 1 RYU Controller, 1 OF switch, OpenFlow protocol, 25 hosts/nodes.	TCP/IP traffic	**Accuracy** of **99.26** percent. **Detection** rate **100** percent. **Limitation**: Few data feature. **Future** Have multiple switches and controllers in network packet analyzer.
Al-Fayoumi & Al-Haija, 2023	**Detection** A lightweight detection scheme that can capture LR-DDoS attacks based on MQTT protocol in SD-IoT environment.	Classification	ML- DT, MLP, ANN and NB OpenFlow tool for generating real-time DDoS attacks in SD-IoT.	MQTT	**Highest accuracy** of **99.5** percent with DT. **Future**: Mitigation or prevention technique using FTE can be investigated as an extension for the presented detection system.
Mohsin & Hamad, 2022	**Detection and Mitigation** Performance Evaluation of SDN DDoS Attack Detection and Mitigation based on RF, KNN ML Classification **Single**: 1 controller, 1 switch, 64 hosts. **Multi-** 2 controllers, 8 switches, 64 hosts.		ML-RF, KNN, LR, NB. **Simulation**: Mininet, RYU controller, OV Switches, OpenFlow protocol, hosts. **Linear**: 1 controller, 8 switches, 64 hosts.	ICMP flood	Block port with timeout. **Limitation:** One attack type. **Future**: Implementing in real network instead of virtual and adopting more SDN controllers in the multi-controller topology.

The gaps in knowledge were sufficiently brought out by leveraging the limitations and future works of the related literatures reviewed in this section. The gaps form the bases of the techniques employed for the proactive mitigation of DDoS attack in the enterprise IoT-based WAN.

4.0 METHODOLOGY

The methodology centers on the objectives of the research work, which include the following:

i Develop an SDN emulated environment for the simulated IoT-based WAN.

ii Develop a DDoS IoT-related dataset from the SDN emulated environment.

iii Develop an ML model to detect the DDoS IoT-related attack data packets.

iv Develop SDN FTE security rules to mitigate the detected DDoS IoT-related attack packets.

v Evaluate the model performance metrics for the DDoS detection, such as accuracy, precision, recall, F1 score, receiver operating characteristics (ROC) curve, area under ROC curve (AUC), mitigation time, data packets dropped and switch ports blocked.

The methodology highlights the research methods and materials for the proactive mitigation of DDoS IoT-related attack in the enterprise WAN. It also highlights the implementation which requires downloads and installations of various software for the ML models in SDN environment.

The research methods involved launching the DDoS IoT-related attack using TCP, UDP, ICMP and HTTP multi-protocols. The models to detect the DDoS attack from normal traffic utilize four supervised ML algorithms, namely SVM, DT, RF and LR, for classification. The method utilized a single RYU controller and three ovs switches. The research methods are based on the functional block diagram of Fig 4.1. The research materials and applications required to set up the testbed to achieve the objectives are grouped by functions in Tables 4.1- 4.4. These include software and hardware for the enterprise IoT WAN set up and simulation with VMs, Mininet emulation of the network topology in SDN environment, and frameworks for ML data preprocessing and data analysis.

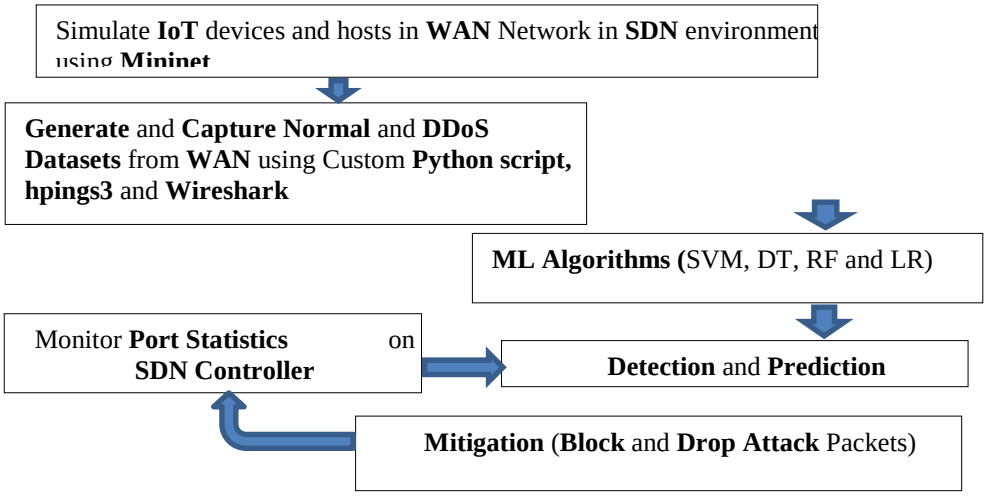

Fig 4.1: Functional Block Diagram for Detection and Mitigation of DDoS IoT Attack Traffic

Table 4.1: WAN (3 LAN Nodes), Devices, Hardware and Software for IoT Enterprise Network

WAN Materials	Type	Description	Remark
3 LAN Nodes	1 HP Laptop/ Desktop Computer (PC)	a. OS- Windows 10 Pro. b. Processor-Intel (R) Core i7 2.8GHz, 64 bits. c. Memory–12-16 GB RAM.	Hardware/ Software
Other network tools	a. Mininet VM. b. Apache Web Server.	a. Run Mininet VM on Ubuntu OS, which in turn runs on ORACLE VM VirtualBox 6.1 on Desktop comp. b. Run Apache Web Server 2.0 at DDoS victim.	Software for Emulation of Enterprise IoT WAN
Normal traffic Generation DoS/DDoS attacks Generation	9 IoT devices. Hping3 Utility tool.	Normal traffic. Attack floods of TCP SYN, ICMP, UDP and HTTP GET.	Simulated
Capturing Data	Wireshark 3.2.3	Running on single controller console to capture data at victim's end.	Software

Table 4.2: IoT Devices Deployable in Enterprise Organization WAN

IoT Devices (virtual objects)	Remark
1.Enhance security: Motion Sensor, Smart Camera, Smart Thermostat, Air Cooler, Alarm and Smoke Detector. 2. Save energy: Smart socket, Switch, Battery Bank, Smart Meter and LED bulb.	Simulated 9 IoT devices for WAN testbed set up.

Table 4.3: Software for Simulation and Mininet Emulation in SDN Environment

Software/ Frameworks	Processes	Remark
Mininet 2.3.0	A network emulator that creates virtual hosts, switches, links, controllers, IoT devices and	Mininet switches support OpenFlow for

	routers, simulating the SDN network.	custom routing and SDN.
Linux network software	Mininet hosts run standard Linux network software.	
OpenFlow Protocol 1.3	Defines how the switch will behave as it encounters different types of data packets.	
Oracle VM VirtualBox 6.1 software	A virtual environment for deployment of Ubuntu and Mininet.	
Ubuntu 20.04 OS	Ubuntu 20.04 is run from Oracle VM Virtual Box 6.1.	
Mininet VM	Mininet VM would run on the Ubuntu OS via VMware.	Mininet VM comes with Mininet.

Table 4.4: Frameworks for ML Data Preprocessing and Analysis

Processes	Frameworks	Remark
Data Cleansing	**Pandas Library**	
Data Visualization	**Matplotlib and Seaborn frameworks**	
Extracting, Selecting Features	**Pandas and NumPy frameworks**	
Data Analysis (Classification) **a. Analysis of the algorithms for supervised learning.** **b. Computing.**	**Scikit-learn framework for predictive data analysis.** **Pandas Library for data analysis.** **SciPy open-source software.**	Python- based (Python3) Scientific Python.

Designing and setting up the Virtual Enterprise IoT-based WAN Testbed

The Mininet emulate the 3 LANs in linear topology to form the enterprise IoT WAN in SDN environment. Each LAN has a minimum of 4 Hosts per Subnet, the Classless Inter-domain Routing (CIDR) of /29, and a Mask of 248. LAN 1 is a Class C network; LAN 2 is a Class B network while LAN 3 is a Class A network.

i Installations of Softwares. Installations and configurations of the various software and tools are done. The design employs a nested VMs setup. Mininet VM 2.3.0 runs within Ubuntu 20.04 OS, which in turn runs from Oracle VM VirtualBox 6.1 that has been installed in a laptop using Windows 10 Pro as in Fig 4.2. With the Mininet, the 3 LANs are set up with the various simulated IoT devices, hosts, Apache server 2.0, and connected to their respective switches, routers. These in turn connect to a single RYU 4.3.4 controller to form a WAN in the SDN environment (Ali et al., 2020).

ii Configurations and Running the Softwares. On starting Oracle VM VirtualBox 6.1, Ubuntu 20.04 virtual disk file is downloaded to the disk and installed. The Ubuntu 20.04 VM is setup with 12GB of RAM and 100GB of storage to accommodate the nested Mininet VM 2.3.0. With the Ubuntu VM runing, the other softwares, namely python3-pip, Mininet, RYU controller, x-term, hping3, Apache server 2.0 are installed in Ubuntu environment using terminal command line interface (CLI). A custom python script using the mininet framework was utilized to design and implement the IoT WAN testbed, which was built using Pycharm community edition Integrated Development Environment (IDE).

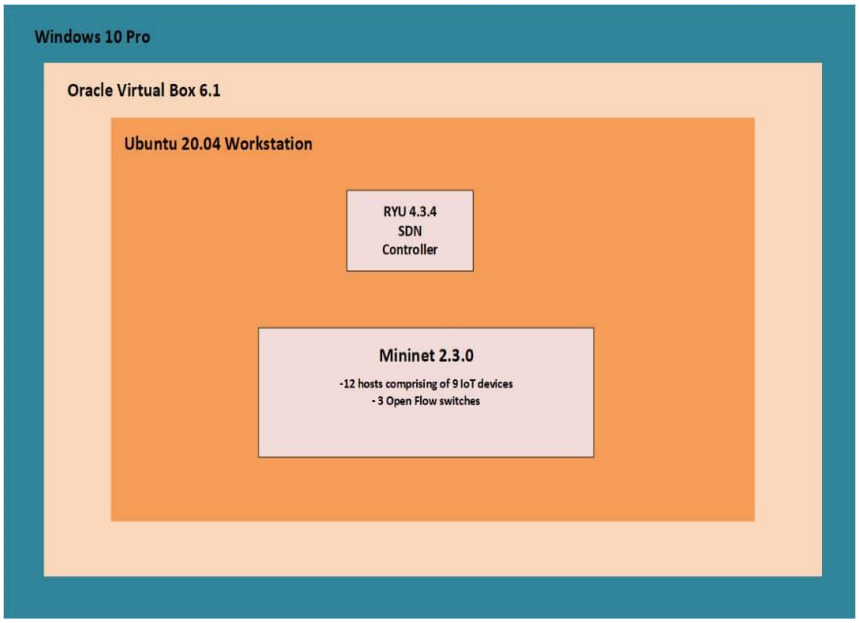

Fig 4.2: A Nested VMs Setup Strategy.

iii Designing WAN Topology. Routers r1, r2 and r3 were created. Next was to create ovs switches s1, s2 and s3 and links, which include router-switch and router-router links. The remote controller c0 was also created. The 3 LANs as in Table 4.5 were configured thus: LAN 1 (192.168.0.0/29), LAN 2 (172.16.0.0/29) and LAN 3 (10.0.0.0/29). Upon creating the hosts (h1-h12), h1-h4 were connected to LAN 1, h5-h8 were connected to LAN 2, and h9-h12 were connected to LAN 3. Host h12 in LAN 3 is used for the Apache web server and can be accessed by all the hosts from LAN1 (h1-h4) and LAN2 (h5-h8) through the routers r1, r2 and r3. The switches s1, s2 and s3 connect to their respective routers via the interfaces r1-eth1, r2-eth1, and r3-eth1. These interfaces are assigned as the gateway for each of the 3 LANs.

Table 4.5: WAN Topology Design for Simulated IoT Testbed

LAN	Network Class	Network Address	CIDR	Subnet Mask	Gateway	Broadcast	Interfaces (router-switch links for r1, r2, r3)	Interfaces (router-router links for r1, r2, r3)
1	C	192.168.0.0	/29	255.255.255.248 (/29)	192.168.0.1	192.168.0.7	r1-eth1	r1-eth2, r1-eth3
2	B	172.16.0.0	/29	255.255.248.0 (/29)	172.16.0.1	172.16.0.7	r2-eth1	r2-eth2, r2-eth3
3	A	10.0.0.0	/29	255.248.0.0 (/29)	10.0.0.1	10.0.0.7	r3-eth1	r3-eth2, r3-eth3

iv Starting the Custom WAN Topology and Testing Reachability. Once the topology is programmed, it is run with a simple python command from the terminal as below. Next is testing the reachability of all the simulated IoT devices in the 3 LANs that form the WAN.

Sudo python wantopoFinal.py

v Starting RYU Controller. The RYU controller, which controls the operation of the ovs switches, is run with the command from a new terminal, while Mininet is still in operation. The RYU controller begins with handling packet-in messages to register the Datapath id of each switch, namely 000000000000001, 0000000000000002, 0000000000000003.

ryu-manager trafficMonitor_completed_perfected_edited.py

ryu-manager trafficMonitor_completed_perfected_edited.py

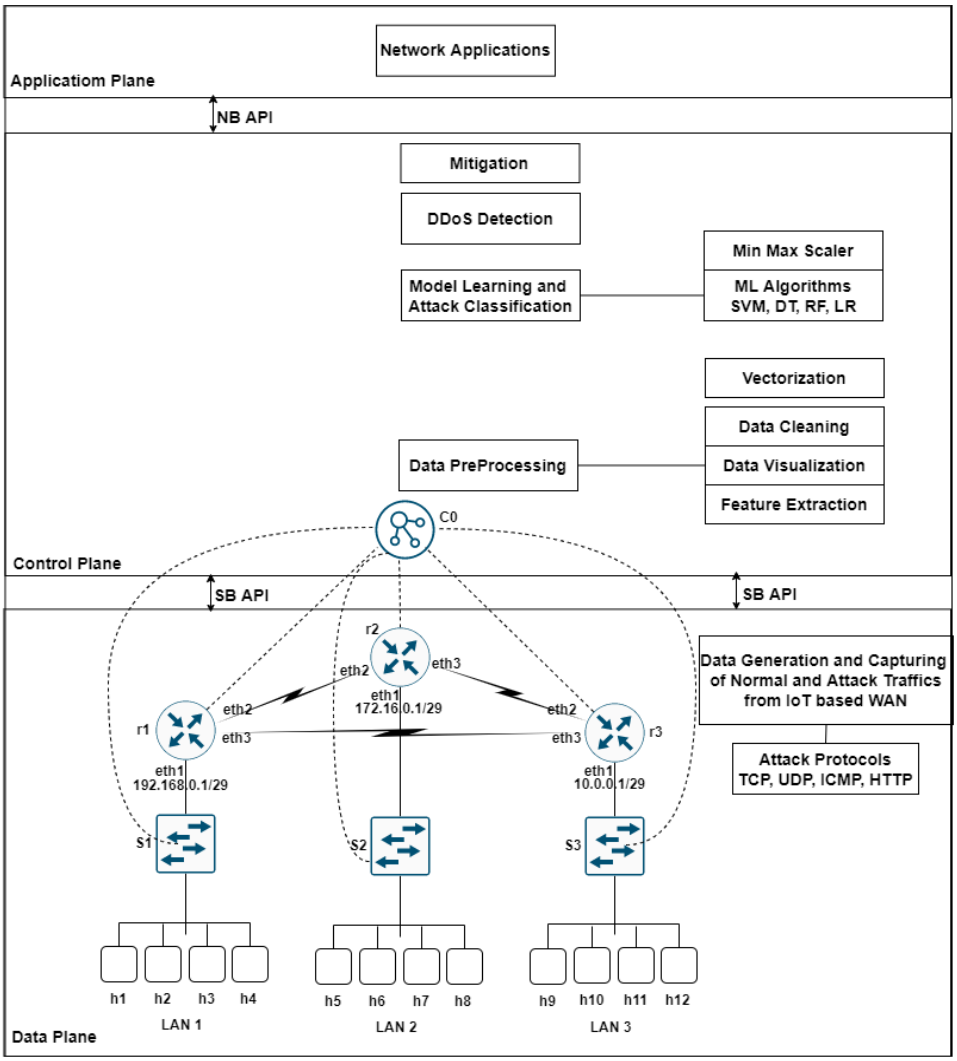

Fig 4.3: Framework of Systems for DDoS IoT-related Generation, Detection and Mitigation in SDN Environment.

Generating Normal and DDoS IoT Attack Datasets, Capturing, Preprocessing, and Extracting Relevant Features from the Datasets

The ML and SDN systems stages for the DDoS IoT- related attack data generation and capturing are highlighted in the framework at Fig 4.3.

i Generating Normal Traffic and Capture. Terminals of the simulated IoT devices are opened with the command,

xterm h1 h2 h3 … h8

`and the custom normal traffic generation script is run from their consoles using the command

python3 ~/datageneration.py

The HTTP server is started with the command,

python –m SimpleHTTPServer

which enables h12 to serve HTML documents to simulated IoT devices. Wireshark 3.2.3 is opened on the controller with the command from the Mininet terminal

xterm c0

Capture process is started with the wireshark. Fig 4.4 shows the Wireshark interface, with all the nodes on the SDN network. Capture of pcap files is done on the loopback interface. OpenFlow is selected as a filter to display aggregate statistics of the network every 5 seconds. Fig 4.5 shows the normal data capture using the wireshark with the various features stored by the hosts. The features are also indicated in Table 4.6. The 237th frame shows the total transmit packets for broadcast port and hosts h1 - h12. The Wireshark pcap dataset captured is saved and exported as a csv file.

ii Generating Attack Traffic and Capture. Similar process is used to generate attack traffic, except for the use of hpings3 from hosts, say h2, h4, h6 console by command

hpings3 –flood 10.0.0.5 -2 –flood.

This floods the host h12 (victim) with requests. Wireshark is used to capture traffic on the loopback interface of the controller (Silva et al., 2020). The observable difference is the spike in transmit and receive data on the attackers (h2, h4, h6) and victim (h12). Wireshark uses this system to filter attack pcap data traffic captured, which are converted to csv file like the normal traffic.

iii Pre-Processing the Datasets. The normal traffic of about 1,000 transmitted packets per second dataset is imported as csv with the Pandas python library into a Jupyter notebook. Other supporting python libraries like Matplotlib and NumPy are also imported (Sumadi et al., 2022). The Tx bytes and Tx packets, being the only varying dataset features, were selected and converted into numbers. Much more attack traffic is transmitted every 5 seconds, reaching a maximum of 24,000 transmitted packets per second. The phase involves adding the target, either a 0 or 1 to represent normal or attack traffic (Najafimehr et al., 2022). The normal and attack preprocessed datasets were combined to form a single dataset of 25,000 packets per second with a script written in Jupyter notebook.

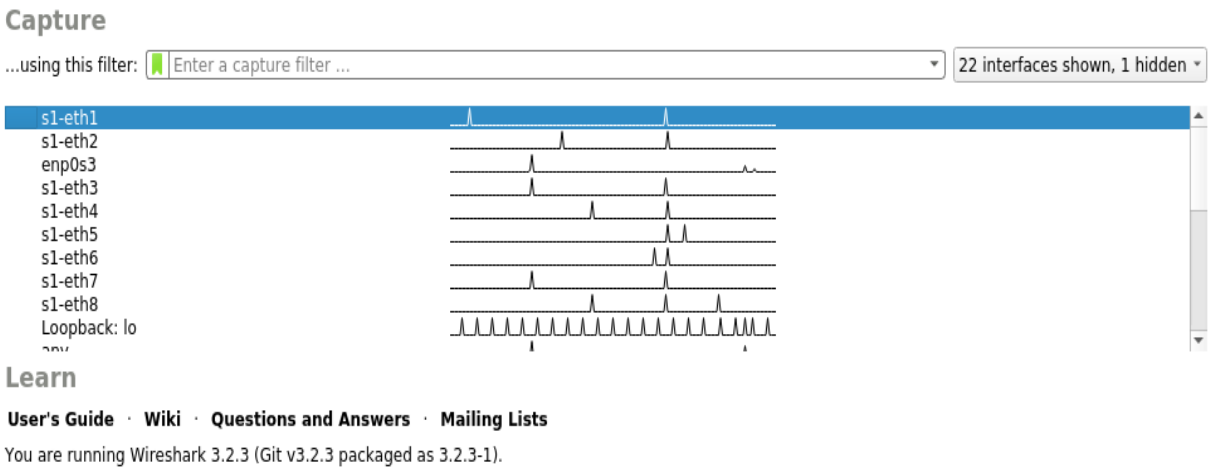

Fig 4.4: Wireshark on the Controller for Data Capturing.

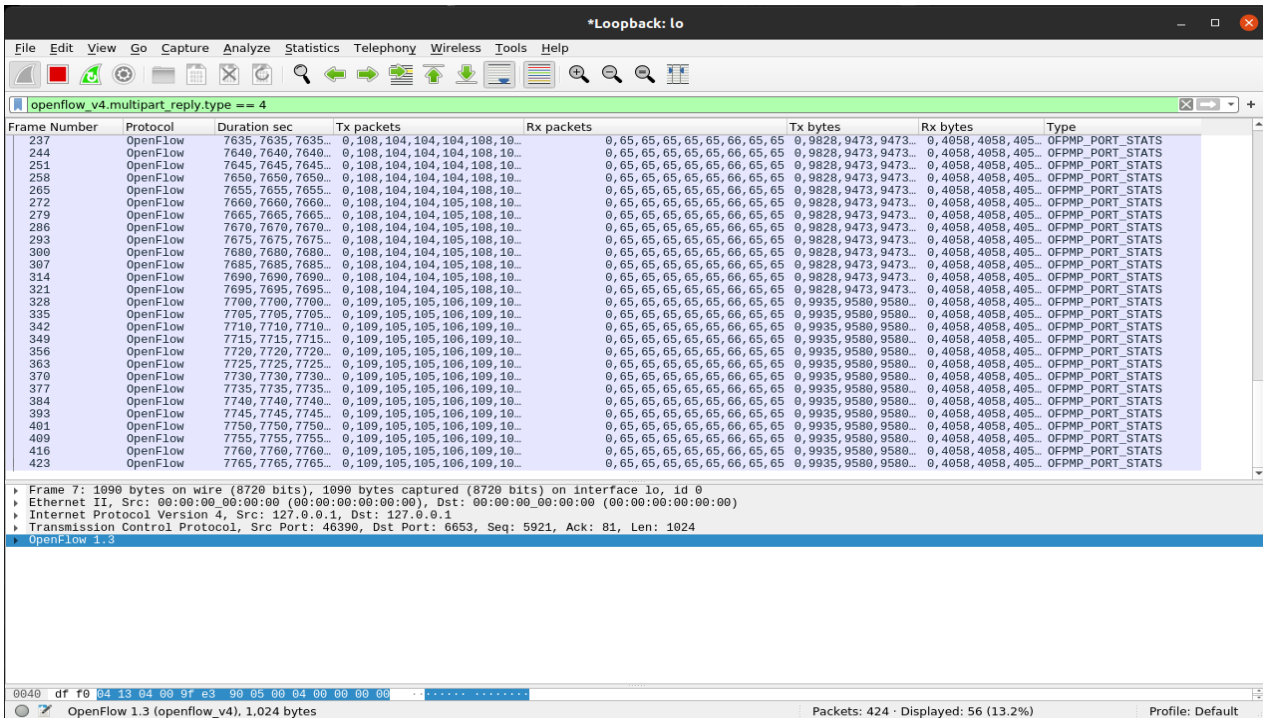

Fig 4.5: OpenFlow Normal Data Capture with Wire shark.

Table 4.6: Features of Normal and Attack Datasets Captures.

5 Features	Remark
Tx packets, Rx packets, Tx bytes, Rx bytes, Type	

Training the ML Models for Classification and Detection of Attack Traffic

The features are processed with a RF classifier and Tx packets difference was selected as the better. The selected features are then split into train and test data, with the test data constituting about 25 percent in size. The features are scaled down with the Min Max Scaler from Scikit-learn to shrink the range to between 0 and 1 using formula below.

```
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min
```

With the train and test datasets ready, the four ML algorithms train the models and then evaluated for accuracy.

i Logistic Regression Classifier**.** The LR was trained with default hyper parameter settings and the score is shown in Fig 4.6.

```
In [155]: logReg.fit(X_train_scalled,y_train)
          scoreTest  = logReg.score(X_test_scalled,y_test)
          scoreTrain = logReg.score(X_train_scalled,y_train)

          print("the score on training data is {:.16f}".format(scoreTrain))
          print("the score on testing data is {:.16f}".format(scoreTest))



          import seaborn as sns
          sns.regplot(x='Tx packets differnce', y='status', data=data, logistic=True)

          y_predict_logistic = logReg.decision_function(X_test_scalled)
          0.9947368421052631

          the score on training data is 0.9977443609022556
          the score on testing data is 0.9947368421052631
```

Fig 4.6: LR Classifier Model.

ii Random Forest Classifier**.** The RF classifier was also trained with the result in Fig 4.7.

```
In [101]: forest = RandomForestClassifier( random_state=47, n_estimators=100, max_features="sqrt")
          forest.fit(X_train_scalled,y_train)

          forestScoreTest   = forest.score(X_test_scalled, y_test)
          forestScroreTrain = forest.score(X_train_scalled, y_train)

          print("the score on the training data is {:.16f}".format(forestScroreTrain))
          print("the score on the test data is {:.16f}".format(forestScoreTest))


          the score on the training data is 0.9992481203007518
          the score on the test data is 0.9912280701754386
```

Fig 4.7: RF Classifier Model.

iii Support Vector Machine Classifier**.** The SVM classifier was also trained, and the score is at Fig 4.8.

```
In [143]: svm = SVC()

In [144]: svm.fit(X_train_scalled,y_train)
          svmTestScore  = svm.score(X_test_scalled,  y_test)
          svmTrainScore = svm.score(X_train_scalled, y_train)



          print(f'the test accuracy is {svmTestScore} and the train accuracy is {svmTrainScore}')


          y_predict_svm = svm.decision_function(X_test_scalled)

the test accuracy is 0.9947368421052631 and the train accuracy is 0.9977443609022556
```

Fig 4.8: SVM Classifier Model.

iv Decision Tree Classifier**.** The final model trained is the DT classifier; the method is shown in Fig 4.9.

```
treeDess = DecisionTreeClassifier()
treeDess.fit(X_train_scalled,y_train)


treeTestScore  = treeDess.score(X_test_scalled,  y_test)
treeTrainScore = treeDess.score(X_train_scalled, y_train)

print(f'the test accuracy is {treeTestScore} and the train accuracy is {treeTrainScore}')


# Draw graph
dot_data = export_graphviz(treeDess)
dot_data
graph = graphviz.Source(dot_data, format="png")
# graph
y_predict = treeDess.predict(X_test_scalled)


the test accuracy is 0.9912280701754386 and the train accuracy is 0.9992481203007518
```

Fig 4.9: DT Classifier Model.

The LR and SVM classifiers each had test dataset accuracy detection of 99.474 percent, which is higher than the RF and DT with test dataset accuracy detection of 99.123 percent each. LR and SVM models are more suitable classifiers of attack and normal traffics. Both models are pickled and exported for usage by the RYU controller.

Mitigating the DDoS IoT-related Attack Packets using SDN Controller

When a layer 2 switch is started with the RYU-manager command, it listens on 127.0.0.1:6633 for *Packet-In* messages from the hosts setup in Mininet. Like typical network that uses Address Resolution Protocol (ARP) to discover the presence and state of hosts on the network, *Packet-In* messages are sent to the controller by the switch to determine the action for hosts that are not registered on its forwarding FTE.

This behaviour had to be modified so the custom RYU controller could reject *Packet-In* messages from suspected hosts which could potentially overwhelm and bring down the server. To start the detection and

mitigation process, the default *simple_monitor.py* script was modified by:

i Deactivating the *Packet-In* Message. *Packet-In* message is sent only when an attacker is not found on the network.

ii Modifying the Statistics (monitor) Function to make Port and Flow Statistics Request every 5 seconds. This gives enough time for the controller to function properly without being over tasked.

iii Extracting the relevant Features required by the Trained Model from Port Statistics Messages. By computing the Packet and Byte Differences every 5 seconds.

iv Using the Pickled ML Model to Predict Status of the Traffic. Since the data used to train the model was scaled down by the Min Max scaler, the value of 4.1209e-5 has to be multiplied by each data point as gotten from the port statistics to make it suitable for classification.

v Blocking the Switch Port if an Attack is Detected. If an attack is detected, the result variable will be 1 else 0. The mitigation function is used to block the port from which an attack is detected.

5.0 RESULTS AND DISCUSSIONS

This section discussed the results of the ML Classification models for DDoS IoT-related attack detection and mitigation in SDN WAN environment. The performance of the RYU custom controller is also discussed.

Running the RYU Program

On execution of the custom RYU controller from the terminal, it greets the admin with a message, indicating the total number of ports in the network, that is port statistics. The next phase was to test the operation of the network with a single attack i.e., DoS and then attacks from multiple IoT devices i.e., DDoS. Thus, Hping3 carried out DDoS attacks from h2, h4 and h6 hosts (virtual terminals) of the attacker IoTs, which were launched to flood h12 (victim). The RYU controller was able to detect and mitigate attack for the first time in 10 seconds, and less time in subsequent tries. Thus, Fig 5.1 shows the detection and blocking of attacker h2 from port 3 on switch 1. It can be observed that rx-packets and rx-bytes features have spiked values of 407231 and 17105730 respectively. These indicate arrival of attack data traffic on port 3 switch 1 as compared to other ports. Further on Fig 5.1, the tx-packets and tx-bytes features are observed to have spiked values of 407330 and 17115274 respectively due to attack data traffic leaving port 1 on switch 1 which is the *gateway*.

This command below is used to block h2. The command is copied and executed from the Mininet terminal.

sh ovs-ofcl add flow s1 priority=65535, in port = 3, actions drop

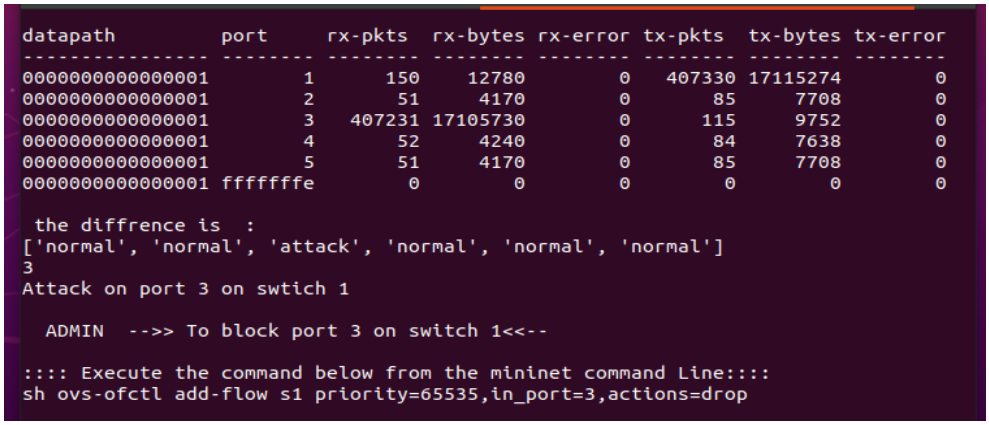

Fig 5.1: Detection and Blocking of Attacker h2 (Port 3 on Switch 1).

Results and Analysis of ML Classification of the Data Traffic

In Fig 5.2, the router can no longer reach host h2. This shows that the custom controller is actively detecting, mitigating and monitoring attacks on the WAN.

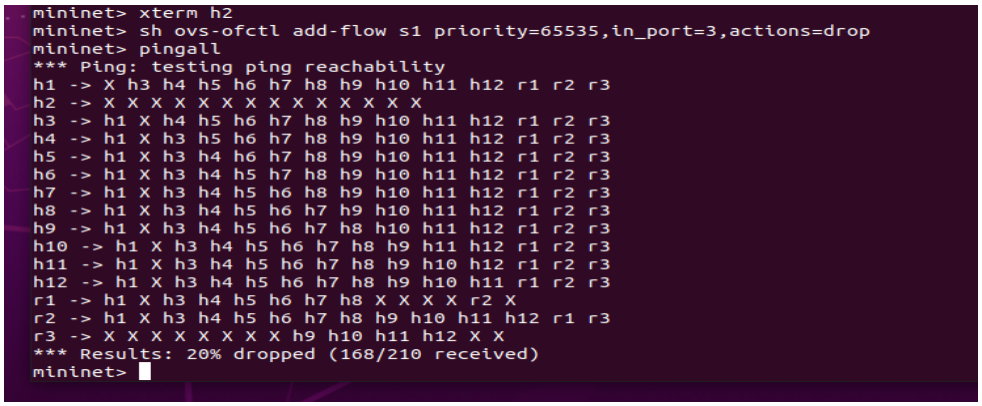

Fig 5.2: Ping Reachability during Mitigation of Attacker h2 (from Port 3 on Switch 1).

In Table 5.1, the accuracy on the test datasets for the four ML algorithms indicate that LR and SVM models have higher accuracy of 99.474 percent each while DT and RF models come next with 99.123 percent each. The ROC curve in Table 5.2 indicates LR model with AUC of 0.998, which is greater than SVM model with AUC of 0.995. Both ROC curves are shown in Fig 5.3 with AUC scores closer to 1, meaning that the models have the ability to separate the two classes, DDoS and normal, and both curves come closer to the top left corner of the graph. Hence the best model to use for this task is the LR model. This model was pickled and used for detection by the RYU controller. Furthermore, the SkLearn metrics for accuracy score and Confusion Matrix results in Fig 5.4 provide more analysis as in Table 5.3. This shows other performance metrics in addition to accuracy, which include precision, recall, F1 score, ROC, sensitivity and specificity.

Table 5.1: Accuracy Results from ML Classification of Models.

Evaluation Metric	ML Algorithms			
	LR	DT	RF	SVM
Accuracy on Test data	99.474 percent	99.123 percent	99.123 percent	99.474 percent

Table 5.2: ROC Curve Results from ML Classification with Algorithms.

Evaluation Metric	ML Algorithms	
	LR	SVM
ROC Curve	AUC= 0.998	AUC = 0.995

```
plt.figure(figsize=(5,5),dpi = 100)
plt.plot(svm_fpr, svm_tpr, linestyle= '-', label='SVM(auc = %0.15f)'%auc_svm)
plt.plot(logistic_fpr, logistic_tpr, marker='.', label='LR (auc = %0.15f)'%auc_logistic)
plt.xlabel('False positive Rate')
plt.ylabel('True Positve Rate')
plt.legend()

<matplotlib.legend.Legend at 0x2abd572beb0>
```

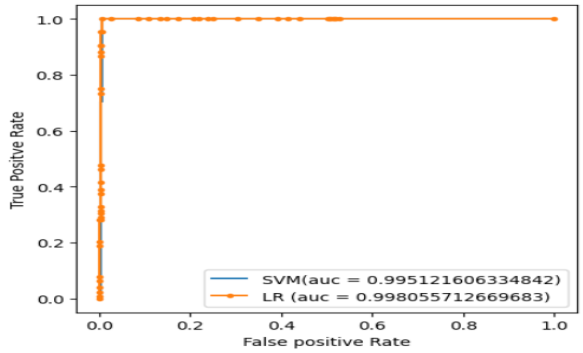

Fig 5.3: ROC Curves for SVM and LR Models.

```
In [34]:
from sklearn.metrics import accuracy_score
print(accuracy_score(y_test, y_predict))
pd.crosstab(y_test,y_predict)

0.9912280701754386

Out[34]:
col_0    0    1
row_0
0      439    3
1        2  126
```

Fig 5.4: SkLearn Metrics Accuracy Score and Confusion Matrix Result.

Table 5.3: Performance Metrics by Substituting the SkLearn Accuracy Scores into the Confusion Matrix Formular and Other Metric Equations

Other Performance Metrics for the ML Classification	Confusion Matrix Formular	SkLearn Metrics Accuracy Score	Remark
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	$\frac{439 + 126}{439 + 126 + 3 + 2}$	$\frac{565}{570} = 0.99123$
Precision	$\frac{TP}{TP + FP}$	$\frac{439}{439 + 3}$	$\frac{439}{442} = 0.99321$
Recall	$\frac{TP}{TP + FN}$	$\frac{439}{439 + 2}$	$\frac{439}{441} = 0.99546$
F1 Score	$\frac{2 * TP}{2 * TP + FP + FN}$	$\frac{2 * 439}{2 * 439 + 3 + 2}$	$\frac{878}{883} = 0.99434$
ROC	FP Rate = $\frac{Number\ of\ FP\ Samples}{Total\ Number\ of\ Samples}$	$\frac{3}{439 + 126 + 3 + 2}$	$\frac{3}{570} = 0.00526$
	TP Rate = $\frac{Number\ of\ TP\ Samples}{Total\ Number\ of\ Samples}$	$\frac{439}{439 + 126 + 3 + 2}$	$\frac{439}{570} = 0.77018$
Sensitivity (True Positive Rate)	$\frac{TP}{TP + FN}$	$\frac{439}{439 + 2}$	0.99546
Specificity	$\frac{TN}{FP + TN}$	$\frac{126}{3 + 126}$	0.97674

Another ML classification result is the LR visualization in Fig 5.5 which shows the plot of status with range 0 to 1 against Tx packet difference with range from 0 to 25000. The plot's shape has semblance comparable with any unique and standard LR shape. The status (0 to 1) indicate the outcome of classifying the features as normal or DDoS attack.

```
import seaborn as sns
sns.regplot(x='Tx packets differnce', y='status', data=data, logistic=True)

y_predict_logistic = logReg.decision_function(X_test_scalled)
0.9947368421052631

the score on training data is 0.9977443609022556
the score on testing data is 0.9947368421052631

Out[19]: 0.9947368421052631
```

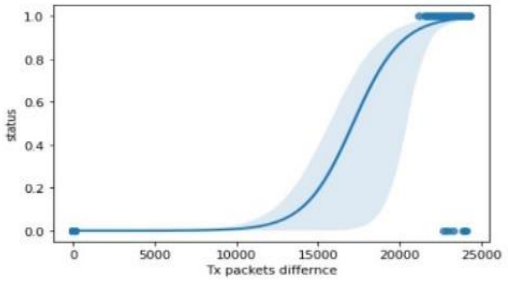

Fig 5.5: LR Plot of Status against Tx Packet Difference.

Evaluation of the DDoS IoT-related ML Detection and SDN Mitigation Techniques

Evaluation of the project was achieved in two-fold as follows:

i Evaluation of Detection using ML Technique. A subset of the training data, called test data, forming about 25 percent, was held back from the ML algorithms until the very end. After selecting and tuning the ML algorithms on the training dataset, the trained models were evaluated on the test data to get a final of how the models performed, predicted or generalised on new or unseen data (Bhandari., 2020, April 17; Ameisen., 2020; Ali et al., 2023). Table 5.4 shows the detection scores while evaluating the train data on the test data for the four ML classifiers.

Table 5.4: Evaluation of Detection Train Scores on Test Scores for the ML Classifiers

ML Classifier	Detection Scores		Remark
	Test	Train	
Logistic Regression (LR)	0.99474	0.99774	Train scores higher than Test values for all classifiers
Random Forest (RF)	0.99123	0.99925	
Support Vector Machine (SVM)	0.99474	0.99774	
Decision Tree (DT)	0.99123	0.99925	

ii Evaluation of Mitigation using SDN Techniques. The output of the detection system is run over the SDN mitigation framework through a simulated environment that comprises the RYU controller, routers, OpenFlow switches, and Mininet emulator (Jimenez et al., 2021; Ramalakshmi & Kavitha, 2024; Mahajan & Bhandari, 2020). The arriving packet that match the switch FTE was serviced and forwarded by the switch as normal traffic to the destination. The packet without a matching FTE was forwarded to the RYU controller, which dropped the packet and blocked the port (Kaur & Gupta, 2023; Ali., 2024; Surekha et al., 2023). The controller then fed the forwarding

information back to the switch and updated its FTE. Table 5.5 shows percentages of packets dropped, received and ports blocked during DDoS attack from the nodes.

Table 5.5: Evaluation of Mitigation Through Packets Dropping and Ports Blocking

DDoS Attack Nodes	**Percentage Packets**		**Ports Blocked**	**Remark**
	Dropped	**Received**		
No Attack	7 Percent	195/210 = 92.86	Normal	Normal
h2	20 percent	168/210 = 80	Port **3** on Switch **1**	Ongoing Mitigation
h2, h4	31 percent	143/210 = 68.1	Port **5** on Switch **1**	Ongoing Mitigation
h2, h4, h6	42 percent	120/210 = 57.14	Port **3** on Switch **2**	Ongoing Mitigation

Validation of ML Detection Accuracy

A comparative analysis is done in Table 5.6, where own study ML detection accuracy value of 99.474 percent with LR at serial 7 is the highest when viewed alongside the accuracy values obtained in other research literatures that were reviewed. The only exception is the work at serial 2, which recorded approximately 100 percent with SVM as the most efficient classifier among the ML algorithms that this researcher deployed. In my own study, SVM also recorded accuracy of 99.474 percent but had a slightly lower AUC value of 0.995 compared to LR, which had higher AUC value of 0.998. These results, thus validate the use of LR algorithm in my work. Also, own detection accuracy of 99.474 percent was obtained using RYU SDN controller. This score ranks excellently well when compared to other detection accuracies obtained that used RYU controller at serials 2, 3 and 5, moreso with the use of multiple ML algorithms. Therefore, the use of RYU controller further validates my research work.

Table 5.6: Comparative Analysis of DDoS IoT Attack Detection Accuracy with other Research Works

Serial	Author and Year	Objective	Model Type	ML Algorithms Type	SDN Controller Type	Evaluation Metrics (Accuracy)
1	Valizadeh, & Taghinezhad-Niar, 2022	**Detection**	Classification	**Simulation** –		**Single victim** attack detection rate is **98.34** percent. **Multi-victim** attack detection rate is **95** percent.
2	Singh, 2021	Detection	Classification (binary)	ML - SVM (Linear and K-RBF), KNN, DT, RF, MLP, GNB.	RYU	SVM - most efficient. Approx **100** percent.
3	Sanjeetha et al., 2021	Detection and Mitigation	Classification (binary)	ML - Catboost, XGBoost, DT, LR, GNB, and KNN.	RYU	Catboost– Best **98** percent
4	Ravi & Shalinie, 2020	**Detection and Mitigation**	Semi-supervised ML algorithm.	Divides IoT devices into two categories: fixed (fIoT), moving (mIoT).	Local and Universal Controllers	**Accuracy** rate of **96.28** percent in detecting DDoS.
5	Singh, 2020	Detection and Mitigation	Classification (binary)	ML – SVM	RYU	**99.26** percent.
6	Al-Fayoumi & Al-Haija, 2023,	**Detection**	Classification	ML- DT, MLP, ANN	MQTT	**Highest accuracy** of **99.5** percent with DT.
7	**Own Study**	Detection and Mitigation	Classification (binary)	ML – LR, SVM, RF, DT.	RYU	LR – Best **99.474** percent

The SDN RYU Controller determines if the flow is normal or DDoS IoT-related, based on the behaviour of the traffic in terms of features and ML classification. Then, it dynamically sets security FTE rules for switches.

6.0 CONCLUSION

The application of ML techniques to provide actionable insights into flooding datasets and SDN tools to mitigate DDoS IoT-related

attacks in the enterprise WAN, has been the set goal of the research work.

The IoT WAN, comprising 3 LANs in SDN environment was designed and simulated using Mininet emulator. The testbed generated both normal and DDoS attack datasets from TCP, UDP, ICMP and HTTP protocols every 5 seconds interval, which were captured as pcap files by Wireshark. The captured normal and DDoS attack datasets were imported to csv files, and into Jupyter Notebook, preprocessed using Pandas, Matplotlib and NumPy Python libraries to extract the relevant features.

The initial five features namely, ports on the switches, received packets, received bytes, transmit packets and transmit bytes were reduced to two features, that is, transmit packets and transmit bytes. These were evaluated during the 5 seconds intervals, and the RF classifier selected transmit packets difference as the best feature. With a combined normal and DDoS dataset value of about 25,000, the selected feature was scaled down to 1 using the Min Max scaler, then split into train and test datasets for the ML classification.

Four ML models using SVM, LR, DT and RF algorithms, classified the trained datasets to detect the DDoS IoT- related attack packets. SVM and LR recorded higher percent accuracy of 99.474 each, while DT and RF recorded 99.123 percent accuracy each. The ROC curve for LR model showed AUC of 99.8 percent, which is greater than SVM with 99.5 percent, indicating that the LR model is the best of the four ML models for detection. For a legitimate prediction, a value of 4.1209e-5 was multiplied to each data point to scale it up after classification.

For mitigation of the DDoS attack, packet-in messages were sent to the SDN RYU controller by the OpenFlow switch to determine the action for simulated hosts (IoTs) that are not registered on its forwarding table FTE. If an attack is detected, the result variable will be 1 else 0, which was used to determine the mitigation function. Thus, the mitigation function is used to block the port and drop the packets from which an attack is detected on the host IoT. The RYU controller detected and mitigated the attack within 10 seconds, and less time in subsequent tries.

The limitations of using simulated IoT devices and emulated WAN SDN environment to generate the datasets (normal and DDoS attack) for the testbed meant that the results (findings) may differ if physical IoT devices were used to generate real datasets. In practice, enterprises would adopt real IoT deployments instead of simulation and emulation of the SDN environment.

The future work could require changing the RYU SDN controller to another python-based controller such as POX and noting the results.

REFERENCES

Abhishek G. S. (2023). Review on DDoS Attacks on IoT Devices. *International Journal of Research Publication and Reviews, 4* (12), 4771-4776.

Al-Fayoumi M., & Al-Haija Q. A. (2023). Capturing Low-Rate DDoS Attack based on MQTT Protocol in Software Defined-IoT Environment. *SSRN Electronic Journal*, ResearchGate

Alashhab A. A., Zahid M. S., Isyaku B., Elnour A. A., Nagmeldin W., Abdelmaboud A, Abdullah T. A. A., & Maiwada U. M. (2024). Enhancing DDoS Attack Detection and Mitigation in SDN using an Ensemble Online Machine Learning Model. *IEEE Access, 12*, 51630-51649.

Alatram A., Sikos L. F., Johnstone M., Szewczyk P., & Kang J. J. (2023). DoS/DDoS-MQTT-IoT: A Dataset for Evaluating Intrusions in IoT Networks using MQTT Protocol. *Computer Networks, 231* (109809), Elsevier.

Ali A. F. A. (2024). DDoS (Distributed Denial of Service) Attack Detection and Mitigation using Statistical and Machine Learning Methods in SDN (Software-Defined Networking). *International Journal of Formal Sciences: Current and Future Research Trends (IJFSCFRT), 21*(1), 1-13.

Ali J., Lee G., Roh B., Ryu D., & Park G. (2020). Software- Defined Networking Approaches for Link Failure Recovery: A Survey. *Journals/Sustainability/ 12* (10) /10.3390/su12104255, MDPI https://doi.org/10.3390/su12104255

Ali T. E., Chong Y., & Manickam S. (2023). Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review. *Applied Sciences, 13* (3183), MDPI.

Almadhor A., Altalbe A., Bouazzi I., Hejaili A. A, & Kryyinska N. (2024). Strengthening Network DDoS Attack Detection in Heterogeneous IoT Environment with Federated XAI Learning Approach. *Scientific Reports, 14*, 24322, Natureportfolio.

Alshammari T. M., & Alserhani F. M. (2022). Scalable and Robust Intrusion Detection System to Secure the IoT Environments using Software Defined Networks (SDN) Enabled Architecture. *International Journal of Computer Networks and Applications (IJCNA), 9* (6), EverScience Publications, 678-688.

Altamemi A. J., Abdulhassan A., & Obeis N. T. (2022). DDoS Attack Detection in Software Defined Networking Controller using Machine Learning Techniques. *Bulletin of Electrical Engineering and Informatics (BEEI), 11* 5, 2836-2844.

Alwabisi S., Ouni R., & Saleem K. (2022). Using Machine Learning and Software-Defined Networking to Detect and Mitigate DDoS Attacks in Fiber-Optic Networks. *Electronics, 11* (4065), MDPI.

Ameisen E. (2020). Building Machine Learning Powered Applications: Going from Idea to Product. *O'Reilly Media Inc*., USA.

Anusuya R., Prabhu M. R, Prathima Ch., & Kumar J. R. A. (2023). Detection of TCP, UDP and ICMP DDoS Attacks in SDN using Machine Learning Approach. *Journal of Survey in Fisheries Sciences*, 964-971.

Aslam M., Ye D., Tariq A., Asad M., Hanif M., Ndzi D., Chelloug S. A., Elaziz M. A., Al-Qaness M. A. A., & Jilani S. F. (2022). Adaptive Machine Learning Based Distributed Denial-of-Service

Attacks Detection and Mitigation System for SDN-Enabled IoT. *Sensors, 22* (2697), MDPI.

Bhandari A., (accessed 2020, April 17). Everything you should know about Confusion Matrix for Machine Learning. Analytics Vidhya, ,https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/

Bhayo J., Shah S. A, Hameed S, Ahmed A., Nasir J., & Draheim D. (2023). Towards a Machine Learning-based Framework for DDoS Attack Detection in Software-Defined IoT (SD-IoT) Network. *Engineering Applications of Artificial Intelligence*, Elsevier.

Butt H. A., Harthy K. S. A., Shah M. A., Hussain M., Amin R., & Rehman M. U. (2024). Enhanced DDoS Detection using Advanced Machine Learning and Ensemble Techniques in Software Defined Networking. *Computers, Materials & Continua (CMC), 81*(2), Tech Science Press, 3003-3031.

Clinton U. B., Hoque N., & Singh K. R. (2024). Classification of DDoS Attack Traffic on SDN Network Environment using Deep Learning. *Cybersecurity, 7* (23), Springer, 1-28.

Elsayed M. S., Le-khac N., & Jurcut A. D. (2020). InSDN: A Novel SDN Intrusion Dataset. *IEEE Access, 8*, School of Computer Science, University College Dublin, Dublin 4, Ireland, 165263-165284.

Galeano-Brajones J., Carmona-Murillo J., Valenzuela-Valdés J. F., & Luna-Valero F. (2020). Detection and Mitigation of DoS and DDoS Attacks in IoT-Based Stateful SDN: An Experimental Approach. *Sensors, 20* (816), MDPI, 1-18.

Garba U. H., Toosi A. N., Pasha M. F., & Khan S. (2024). SDN-based Detection and Mitigation of DDoS Attacks on Smart Homes. *Computer Communications, 221*, Elsevier, 29-41.

Gelgi M., Guan Y., Arunachala S., Rao M. S. S. & Dragoni N., (2024). Systematic Literature Review of IoT Botnet DDoS Attacks and Evaluation of Detection Techniques. *Sensors, 24* (3571), MDPI.

Hassan A. I., Reheem E. A. E., & Guirguis S. K. (2024). An Entropy and Machine Learning based Approach for DDoS Attacks Detection in Software Defined Networks. *Scientific Reports, 14* (18159).

Hill W., Acquaah Y. T., Mason J., Limbrick D., Teixeira-Poit S., Coates C., & Roy K. (2024). *DDoS in SDN:* A Review of Open Datasets, Attack Vectors and Mitigation Strategies. *Discover Applied Sciences*, 6 (472), Discover

Jenny R. S., & Sugirtham N. (2023). SDN-based Security for Smart Devices against Denial-of-Service Attacks. *Indian Journal of Science and Technology, 6* (3), 181-189.

Jimenez M. B., Fernandez D., Rivadeneira J. E., Bellido L., & Cardenas A. (2021). A Survey of the Main Security Issues and Solutions for the SDN Architecture. *IEEE Access, 9*, 122016-122037.

Karmous N., Dhjab Y. B., Aoueileyine M. Q., Youssef N., Bouallegue R., & Yazidi A. (2024). Deep Learning Approaches for Protecting IoT Devices in Smart Homes from MitM Attacks. *Frontiers in Computer Science*.

Kaur G., & Gupta P. (2023). Detection of Distributed Denial of Service Attacks for IoT-Based Healthcare Systems. *Computer Assisted Methods in Engineering and Science, 30* (2), 167–186.

Kumar J., & Arul L. R. (2023). Mitigate Volumetric DDoS Attack using Machine Learning Algorithm in SDN Based IoT Network Environment. *International Journal of Advanced Computer Science and Applications (IJACSA), 14* (1), 559-568.

Kumari P., & Jain A. K. (2023). A Comprehensive Study of DDoS Attacks over IoT Network and their Countermeasures. *Computers & Security*, Elsevier.

Liu Z., Wang Y., Feng F., Liu Y., Li Z., & Shan Y. (2023). A DDoS Detection Method Based on Feature Engineering and Machine Learning in Software-Defined Networks. *Sensors, 23* (6176), MDPI, 1-24.

Mahajan A., & Bhandari A. (2020). Attacks in Software-Defined Networking: A Review. *ICICCR*.

Mohsin M. A. & Hamad A. H. (2022). Performance Evaluation of SDN DDoS Attack Detection and Mitigation Based Random Forest and K-Nearest Neighbours Machine Learning Algorithms. *IIETA, 36* (2), .233-240, Journal homepage: http://iieta.org/journals/ria

Najafimehr M., Zarifzadeh S., & Mostafavi S. (2022). A Hybrid Machine Learning Approach for Detecting Unprecedented DDoS Attacks. *The Journal of Supercomputing, 78*, 8106-8136.

Negera W. G., Schwenker F., Debelee T. G., Melaku H. M., & Ayano Y. M. (2022). Review of Botnet Attack Detection in SDN-Enabled IoT using Machine Learning. *Sensors, 22* (9837), MDPI.

Pakmehr A., Aβmuth A., Tehari N., & Ghaffari A. (2024). DDoS Attack Detection Techniques in IoT Networks: A Survey. *Cluster Computing, 27*, Springer, 14637-14668.

Praba J. J., & Sridaran R. (2022). An SDN-based Decision Tree Detection (DTD) Model for Detecting DDoS Attacks in Cloud Environment. *International Journal of Advanced Computers Science and Applications (IJACSA), 13* (7), 54-64.

Ramalakshmi R., & Kavitha D. (2024). DDoS Attack Mitigation using Distributed SDN Multi Controllers for Fog Based IoT Systems. *International Journal of Intelligent Systems and Applications in Engineering (IJISAE), 12* (4s), 57–69.

Ravi N., & Shalinie S. M. (2020). Learning Driven Detection and Mitigation of DDoS Attack in IoT via SDN-Cloud Architecture. *IEEE Internet of Things Journal*.

Sanjeetha R., Kanavalli A., Gupta A., Pattanaik A., & Agarwal S. (2022). Real-time DDoS Detection and Mitigation in Software Defined Networks using Machine Learning Techniques. *International Journal of Computing, 21*(3), 353-359.

Sanjeetha R., Raj A., Saivenu K., Ahmed M. I., Sathvik B., & Kanavalli A. (2021). Detection and Mitigation of Botnet Based DDoS Attacks using Catboost Machine Learning Algorithm in SDN Environment. *International Journal of Advanced Technology and Engineering, 8* (76), ISSN 2394-7454, 445-461.

Silva F. S. D., Silva E., Neto E. P., Lemos M., Neto A. J. V., & Esposito F. (2020). A Taxonomy of DDoS Attack Mitigation Approaches Featured by SDN Technologies in IoT Scenarios. *Sensors*, www.mdpi.com/journal/sensors.

Singh A. K. (2021). Machine Learning in OpenFlow Network: Comparative Analysis of DDoS Detection Techniques. *The International Arab*

Journal of Information Technology, 18 (2), 221-226.

Singh C., & Jain A. K. (2023). Detection and Mitigation of DDoS Attacks on SDN Controller in IoT Network using Gini Impurity, Research Square.

Singh C., & Jain A. K. (2024). A Comprehensive Survey on DDoS Attacks Detection & Mitigation in SDN-IoT Networks. *e-Prime – Advances in Electrical Engineering, Electronics and Energy, 8*, 100543, Elsevier.

Singh K., Kumar B., Kumar S., Singh V. P., & Singh A. (2024). Mitigation of Cyber Attacks in SDN-Based IoT Systems using Machine Learning Techniques. *International Journal of Intelligent Systems and Applications in Engineering (IJISAE), 12* (8s), 482-492.

Singh V. K. (2020). DDoS Attack Detention and Mitigation using Statistical and Machine Learning Methods in SDN. National College of Ireland.

Sumadi F. D. S, Widagbo A. R., Reza A. F., & Syaifuddin (2022). SD-Honeypot Integration for Mitigating DDoS Attack using Machine Learning Techniques. *International Journal on Informatics Visualization (JOIV), 6* (1), 39-44.

Sudar K. M., & Deepalakshmi P. (2020). A Two-Level Security Mechanism to Detect a DDoS Flooding Attack in Software-defined Networks using Entropy-based and C4.5 Technique. *Journal of High-Speed Networks, 26* (1), SAGE Journals, 55-76.

Surekha A., Induvadana K., Krishna R. C., Harini B, Neeha B. S., & Aakash R. (2023). Detection of Distributed Denial of Service Attacks in SDN using Machine Learning Techniques. *International Research Journal of Modernization in Engineering Technology and Science (IRJMETS), 5* (3), 2527-2532.

Valizadeh P., & Taghinezhad-Nair A. (2022). A Fault Tolerant Multi-Controller Framework for SDN DDoS Attacks Detection. *International Journal of Web Research*, 5 (1), Winter-Spring.

Wang K., Fu Y., Duan X., & Liu T. (2024). Detection and Mitigation of DDoS Attacks based on Multi-dimensional Characteristics in SDN. *Scientific Reports, 14* (16421).

Wang S., Gomez K., Sithamparanathan Asghar M. R., Russello G., & Zanna P. (2021). Mitigating DDoS Attacks in SDN-Based IoT Networks Leveraging Secure Control and Data Plane Algorithm. *Applied Sciences, 11* (929). MDPI, https://doi.org/10.3390/app11030929

Wani S., Imthivas M., Almohamedh H., Alhamed K. M., Almotairi S., & Gulzar Y. (2021). Distributed Denial of Service (DDoS) Mitigation using Blockchain – A Comprehensive Insight. *Symmetry, 13* (227), MDPI.