Politecnico Di Torino

MASTER THESIS

ACCURATE INTRUSION DETECTION SYSTEM BASED ON DEEP LEARNING FOR SOFTWARE DEFINED NETWORKING

Student: Shikha Sharma
shikha.sharma@studenti.polito.it

Supervisors:
Dr. Anca Delia Jurcut (UCD)
anca.jurcut@ucd.ie

Prof. Ladislau Matekovits (POLITO)
ladislau.matekovits@polito.it

Acknowledgement

I would also like to thank Prof. Ladislau at POLITO. He was very supportive when I approached him to be a supervisor from POLITO for my thesis. He is always so kind to me and helpful in so many ways of guiding me throughout my thesis.

I would like to thank Dr. Anca for allowing me to work with her. It was a pleasure to work with her. She provided a lot of motivation and guidance without which this work would be impossible to achieve. She was always there to help when I'm stuck with some problems. Apart from work She helped in every possible way to help me with long distance work from Dublin.

In the end, I would like to thank all the people of my family and relatives who has been so supportive throughout my master's degree.

Abstract

In recent years, Software Defined Networking (SDN) has gained traction as an innovative approach to overcome the limitations of legacy systems. The SDN's main goal is to separate the control and data planes, making network management easier and enabling for more efficient programmability. The suggested approach would protect the SDN in a more accurate, effective, and scalable manner, overcoming current methods' shortcomings. Different businesses employ network detection algorithms to arrange and differentiate harmful traffic these days, however there may be many challenges to encounter and determine assaults in imbalanced datasets.

Examining the influence of different contemporary Deep Learning (DL) approaches, such as the Long Short-Term Memory (LSTM) based autoencoder and the Recurrent Neural Network (RNN), on the overall system functioning would increase the productivity of the anomaly-based Intrusion Detection System (IDS). DL is a relatively new topic of data security that is widely regarded as one of the most important solutions for addressing shortcomings in standard Machine Learning (ML) approaches. Deep learning can learn the nonlinear structure of data with a large number of dimensions. As a result of its ability to automatically identify connections in raw data without the need for human interference, it can increase the intrusion detection rate.

DDoSNet, an intrusion detection technique for Distributed Denial of Service (DDoS) assaults in SDN circumstances, is proposed in this thesis. It will improve the performance of the anomaly-based intrusion detection system by examining the influence of several existing DL techniques, such as the basic RNN-based autoencoder and the LSTM-based autoencoder, on the overall system functioning. We tested our model using the InSDN and CICIDDoS2019 datasets, both of which were recently published. Because our major goal is to tackle the problem of binary classification in the Intrusion Detection System, we compare both datasets and their outcomes in our suggested approach (IDS).

We use the InSDN dataset, which is an attack-specific SDN dataset, published in the year 2020. The benign and various attack categories that may occur in the various elements of the SDN platform are included in this new dataset. While CICIDDoS2019 is a collection of benign and up-to-date popular DDOS attacks that closely resembles real-world data. This dataset includes a broad range of Distributed Denial of Service attacks. As compared to different benchmarking techniques of InSDN dataset, in CICIDDoS2019 a significant enhancement was obtained in the context of attack detection. As a result, our approach gives us a lot of confidence when it comes to securing these networks.

Table of Contents

Table of Contents

Table of Contents

Table of Contents

List of Figures

List of Tables

List of Acronyms

IDS	Intrusion Detection System
SDN	Software Defined Networking
DDoS	Distributed Denial of Service
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
DL	Deep Learning
NAP	Novel Agent Program
HIDS	Host based IDS
NIDS	Network based IDS
NOS	Network Operating System
PCA	Principal Component Analysis
U2R	User-to-Root
R2L	Remote-to-Local
DoS	Denial of Service
TCP	Transmission Control Protocol

List of Figures, Tables, Acronyms

UDP	User Datagram Protocol
ICMP	Internet Control Message Protocol
TTL	Technology Time to Live
SYN	Synchronize
IP	Internet Protocol
POD	Ping of Death
DNS	Domain Name System
MCA	Multivariate Correlation Analysis
AQM	Active Queue Management
TAMs	Triangular Area MCAs
ELM	Extreme Learning Machine
DCC	Detection Control Center
MC	Mitigation Center
GRU-RNN	Gated Recurrent Neural Network
DFS	Deterministic Fair Sharing
LR	Logistic Regression
GR	Gradient Boosting
NB	Naïve Bayes
KNN	K-Nearest Neighbors
DT	Decision Tree
RF	Random Forest
SVC	Support Vector Classifier
TP	True Positive
TN	True Negative
FP	False Positive

List of Figures, Tables, Acronyms

FN	False Negative
ROC	Receiver Operating Characteristics
TPR	True Positive Rate
H-ELM	Hierarchical Extreme Learning Machine
MHBNC	Meta-Heuristic Bayesian network classification
FPR	False Positive Rate
AUC	Area Under Curve

Chapter 01

INTRODUCTION

1.1 Intrusion Detection System (IDS)

Intrusion is characterized as any illegal activity that causes harm to information. This means that any attack that poses a risk to the information's confidentiality, credibility, or availability is considered an intrusion. An Intrusion Detection System (IDS) is a software or hardware system that detects malicious activity on computer systems to preserve system protection. An IDS's aim is to detect various types of malicious network traffic and device use that a conventional firewall would miss. This is critical for achieving high levels of security against acts that jeopardize computer systems' functionality, credibility, or confidentiality [1].

1.2 General Background

The authors offered a hyper approach based totally on a Long Short-Term Memory autoencoder and an OC-SVM to find anomalies based totally assaults in an unbalanced dataset by using teaching the fashions with handiest examples of commonplace classes. [2] The version was assessed using the cutting-edge dataset InSdn. The consequences of the experiments display the excessive detection charge of 90.6 % high quality and terrible fees additionally reduces the processing time. This approach presents properly self-assurance for securing an SDN environment.

To justify the detection charge against malicious attackers within the community, Latah et al. [3] suggested a five-level hybrid classifier machine. The version includes the KNN methodology, ELM, and H-ELM, as well as three specialized gadget mastering classifiers. The offered technique's average accuracy is 84.29 percent, with precision, recall, and F1-score shares of 94.18, 77.18, and 77.18, respectively.

Prasath et al. [4] presented a NAP framework for securing a community-wide communication version of digital switches. The MHBNC approach is used to classify incoming packets as either normal or hostile guests. The proposed MHBNC version has an average accuracy of 82.99 percent. Furthermore, the precision, recall, and f-rating results were determined to be 77 percent, 74 percent, and 75 percent, respectively.

DDoSNet, an intrusion detection system for DDoS attacks in SDN environments, was proposed by Mahmoud et al. [5]. Their method is based entirely on the DL methodology, which combines the RNN with an autoencoder. They look at the version that makes use of the dataset CICDDoS2019, which includes a comprehensive set of DDoS attacks and fills in the gaps in current datasets. In comparison to other benchmarking methodologies, the results obtained as 99 percent accuracy for specific algorithms are also a progress.

The authors used the cutting-edge dataset InSdn that is publicly to be had to the researchers. In [6] the authors supplied the pleasant option to produce a complete SDN dataset to affirm the overall performance primarily based totally on diverse assault classes that manifest withinside the SDN environment. They made use of the proposed dataset with the aid of using experimenting with 8 famous strategies primarily based totally on gadget mastering for IDSs.

DDoS attack detection techniques were applied to SDN networks by the authors of [7]. On the first stage, a signature-based totally giggle detection system was used to collect community visits. At the final level, SVM and DNN approaches are used for attack class. The authors trained the two detection modules using the KDDCUP99 dataset, which consisted of seven functions out of a total of 41. The results of the tests revealed that the DNN outperforms the SVM, with accuracy rates of 92:30 and 74:30 percent, respectively.

A unique approach for identifying DDoS assaults on SDN was proposed by Mohammed and colleagues [8]. The NSL-KDD dataset was used to train NB classifiers, which included 25 pre-selected functions. To choose mixed functions from the dataset, the authors utilize a combination of three distinct selection methods (Precision, Recall, and F1-rating). Precision, Recall, and F1-rating are all regularly used values of 0:81, 0:77, and 0:77, respectively.

Within the SDN structure, Tuan et al. [9] developed a deep mastering (DL) methodology for a community intrusion detection machine (DeepIDS). The models were trained and tested on the NSL-KDD dataset, and they achieved an accuracy of 80.7 percent and 90 percent for a Fully Connected Deep Neural Network DNN and a GRU-RNN, respectively. This shows that the DL approach can deal with drift-primarily based entirely on anomaly detection within the SDN environment They also assessed the machine's overall performance in terms of throughput, latency, and aid consumption. According to the findings, DeepIDS has no longer had an impact on the OpenFlow controller's overall performance and is thus a viable technique.

Intrusion detection has also been done with DL, but only in traditional networks. With SDNs, we'll briefly outline the effects to distinguish ourselves from our artworks. Javaid et al. [10] employed a self-taught deep mastering set of rules for intrusion detection on the NSL-KDD dataset. Soft-max regression was utilized as a classifier, and the accuracy was 92.98 percent.

In conjunction with the use of recurrent neural networks, Yin et al. [11] introduced a deep mastery methodology for intrusion detection. They got an accuracy of 83.28 percent with their test on the NSL-KDD dataset. Shone et al. proposed a nonsymmetric deep autoencoder for unsupervised feature mastery in their paper [12]. They had promising results on both the KDD Cup 99 percent and the NSL-KDD datasets. However, within the context of SDN, those procedures cannot be done out without delay. The SDN structure isn't always considered in those operations, and some of the functions used in such tactics don't correspond to the drift-based completely functions.

Several intrusion detection algorithms and procedures were proposed to stable OpenFlow-primarily based totally SDN networks. For anomaly-primarily based totally detection procedures, SOM and SVM are often used due to their excessive detection accuracy. A light-weight approach for DDoS assault detection primarily based totally on visitors go with the drift functions is offered in [13] with an extraction of six-tuple functions: Average of Packets according to go with the drift, Average of Bytes according to go with the drift, Average of Duration according to go with the drift, Percentage of Pair-flows, Growth of Single-flows, and Growth of Different Ports. SOM—a neural community-primarily based totally technique—is used because the class approach.

SVM was successfully used in [14] [15] to detect DDoS attacks in SDNs. In [16], a one-magnitude SVM was trained with a malicious dataset for an extraordinarily low false alarm rate. Under daily and assault states, Trung et al. [17]combined hard detection thresholds and a fuzzy inference machine to detect the threat of DDoS attacks based entirely on actual visitor characteristics (Distribution of Inter-arrival Time, Distribution of packet amount according to drift, and Flow amount to a server).

1.3 Problem Statement

Intrusion detection has been a critical safeguard against malicious attacks on SDN infrastructure. Due to the rapid advancement of technology, it has been put in jeopardy due to the natural behavior of data, which may at any time change the data necessary via various large-scale ways. When dealing with different labelled public datasets, community anomaly detection has fallen into issues that have been tough for academics to address using DL techniques. Many difficulties have been explored in this thesis, as our goal is to develop a secure and reliable community anomaly detection system using DL approaches for our labelled datasets InSDN and CICIDDoS2019.

The concept of an SDN structure is a revolutionary and ground-breaking approach to community management. Switches in SDN do not process incoming packets. They search one of their forwarding tables for the inbound packet, and if none is identified, it is delivered to the controller for processing. SDN's running device is the controller. It examines the packets and determines whether they ought to be forwarded to the transfer or dropped. SDN separates the forwarding and processing planes the usage of this technique.

On the other side, the same function can provide a fresh new and powerful way to thwart network attacks. A DDoS attack is one of the most common types of community assault. Because of the internet's rapid growth, a wide spectrum of hosts is vulnerable to cyber-attacks. The majority of DDoS attacks are caused by malicious software that is installed without the user's awareness on compromised hosts. Deep learning techniques are employed in this investigation to uncover DDoS in SDN.

1.4 Main Objectives

The following are the key goals of this study:

- To identify a collection of network intrusion detection methods for performing attack detection in the context of SDN, with the goal of achieving high precision, at a low false positive, and low above during the detection method.
- Create a monitoring scheme in the SDN environment to identify DDoS attacks.
- Using deep learning methods such as RNN and LSTM, develop and implement a DDoS attack detection method.
- To determine the most appropriate approach and tools for evaluating the datasets collected.
  To test the efficacy of a deep learning system with data from the DDoS dataset.

Chapter 01

2.1 Need for Intrusion Detection System

Privacy, honesty, and security against rejection of service can all be given by a computer system. However, as connectivity (especially on the Internet) grows, so does the extensive array of financial prospects that are becoming available. As a result, more and more systems are becoming vulnerable to invaders. Subversion attempts aim to exploit bugs in the operational system as well as application systems, which have culminated in high-profile events such as the 1988 Internet Worm.

If a device is under threat, we want to know about it as soon as possible, so we can react appropriately. This is what an Intrusion Data Sources essentially does. When an attack is detected, an IDS typically does not take any preventative measures; it is more of a reactive than a proactive agent. Rather than becoming a police officer, it takes on the part of an informant [2].

2.1.1 Terminologies

The terminologies used in intrusion detection system are:

- Risk
- Vulnerability
- Attack
- Penetration

2.1.1.1 Risk

Details disclosure that is unintentional or unpredictable, or a breach of operations integrity caused by hardware failure or incomplete or incorrect program design [2].

2.1.1.2 Vulnerability

A proven or suspected defect in a system's hardware, software, or process that allows the system to be penetrated or its data to be accidentally disclosed [2].

2.1.1.3 Attack

A basic interpretation or execution of a threat-implementation strategy [18].

2.1.1.4 Penetration

The ability to gain illegal (undetected) access to programs and files, or the control state of a computer device, after a successful attack [18].

2.2 Classification of IDS

There are six different styles of intrusions:

- Break-in attempts detected by standard activity profiles or infringement of security restrictions.
- Masquerade attacks, which are identified by common performance patterns or security breaches.
- Security control device penetration, which is identified by looking for complex designs of operation.
- Leakage, which is identified by unusual machine resource use.
- A normal use of device resources will detect a denial of service.
- Malicious use, as shown by standard personality profiles, security constraints breaches, or the use of special privileges [18]

2.3 Intrusion Detection Techniques

Intrusion detection techniques are divided into two types:

- Anomaly detection
- Misuse detection

2.3.1 Anomaly Detection

All disruptive behaviors are assumed to be anomalous by anomaly detection techniques. This means that if we could create a "standard activity profile" for a system, we could theoretically mark all system conditions that differ by statistically significant amounts from the specified profile as intrusion attempts. However, if we assume that the group of invasive behaviors just intersects the group of unusual ventures rather than being identical, we can consider the following possibilities:

- Non-intrusive abnormal behaviors are flagged as intrusive.

- Non-abnormal invasive behaviors result in incorrect negatives (Even though incidents are invasive, they are not flagged as such). This is a risky dilemma that is much worse than the issue of false positives.

The choice of threshold thresholds such that none of the over two difficulties is overly magnified, and the selection of elements to track, become the key issues in anomaly detection systems. Because of the overhead of keeping record of, and likely reviewing, many device profile indicators, anomaly detection systems are also analytically costly [18]

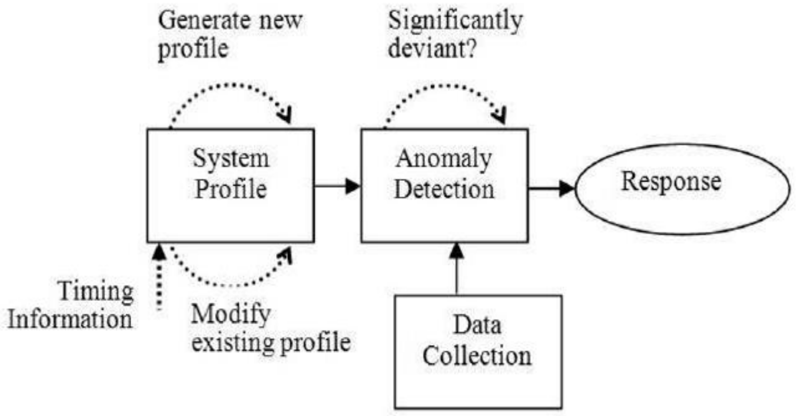

Figure 1: Anomaly detection system[18]

2.3.2 Misuse detection

Misuse detection schemes work on the principle that there are methods to indicate attacks in the form of a pattern or a signature, allowing even variants of the same attack to be detected. This means that, like virus detection systems, these systems can detect several or all established attack models, but they are useless against as-yet-undiscovered attack methods. It is worth noting that anomaly detection systems often aim to detect the opposite of "poor" behavior. Misuse detection systems attempt to identify "evil" activity that has previously been observed. The key challenges in misuse detection systems are determining how to write a signature that covers all possible variants of the relevant attack, as well as determining how to assemble a signature that does not fit non-intrusive behavior [18].

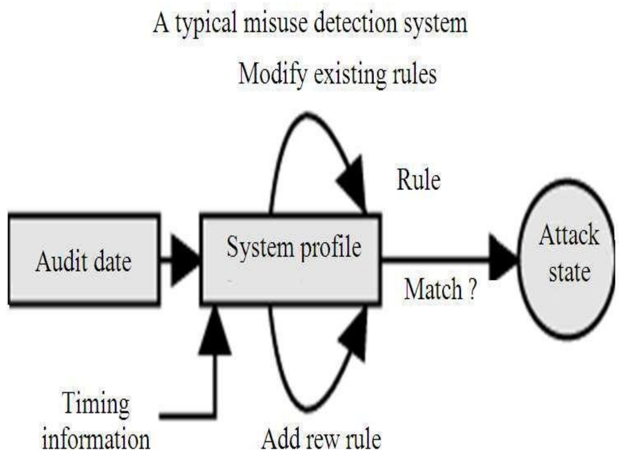

Figure 2: Misuse detection system[18]

2.4 IDS Data sources

IDS refer to the input data sources used to identify suspicious activities. In terms of data sources, there are two types of IDS technologies:

- Host-based IDS
- Network-based IDS

2.4.1 Host-based IDS

The operating system, window server logs, firewall logs, application system audits, and database logs are all examined by HIDS. HIDS [1] can identify insider assaults that do not involve network traffic. This study helps detect intrusions such as unauthorized remote login attempts to gain access to restricted data.

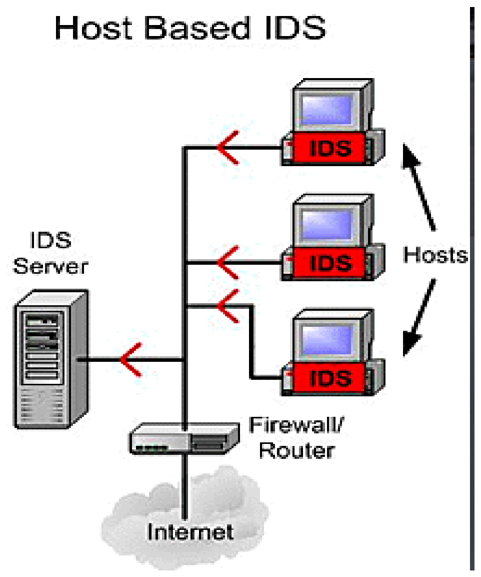

Figure 3: HIDS architecture

2.4.2 Network-based IDS

NIDS is a network traffic monitoring system that uses packet capture, NetFlow, and other network data sources to track network traffic outside of a network. IDS that is network-based can keep track on a huge number of computers on a network. NIDS can detect remote hostile actions that may be triggered by an external attack at an early stage, until the threats spread to another computer device [1].

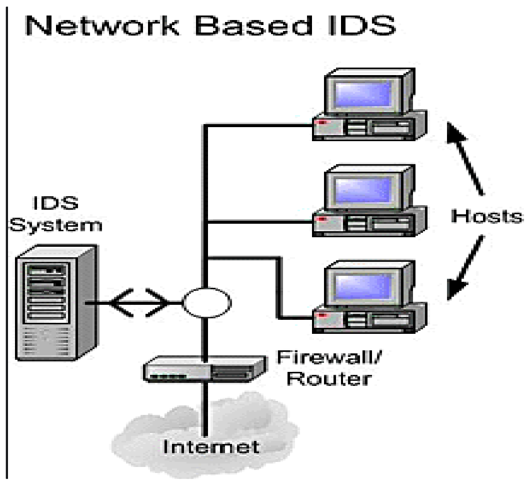

Figure 4: NIDS architecture[1]

2.5 Software-Defined Networking (SDN)

SDN is a network design in which the data plane forwarding status is controlled by a remote-control plane separate from the data plane. The networking industry has diverged from this unique definition of SDN on several occasions, referring to anything using software as SDN. It's a brand-new networking approach that can resolve the problems in today's networks infrastructures. The following four pillars can be used to describe SDN as a network architecture:

- There is a distinction between the control and data planes. Network devices that become simple (packet) forwarding components will lose their control capabilities.
- Rather of focusing on the end destination, forwarding decisions are made with the flow in mind. A flow is defined by a set of packet field values that operate as a match (filter) criterion, as well as a set of operations in general (instructions). The flow abstraction allows various network devices to coordinate their activities, such as routers, switches, firewalls, and middleboxes. Flow programming provides unrivalled versatility, restricted only by the capabilities of the flow tables that have been introduced.
- The SDN controller's Network Operating System is in the charge of the control logic . The Network Operating System (NOS) is a software framework that runs on commodity server technology and provides the tools and abstractions needed to make programming forwarding devices based on a logically centralized, abstract network view easier. As a result, it functions similarly to a traditional network.

- Software programs that interface with the fundamental data plane devices and run on top of the NOS can be used to program the network. This is a crucial aspect of SDN, as well as its main selling point. [20].

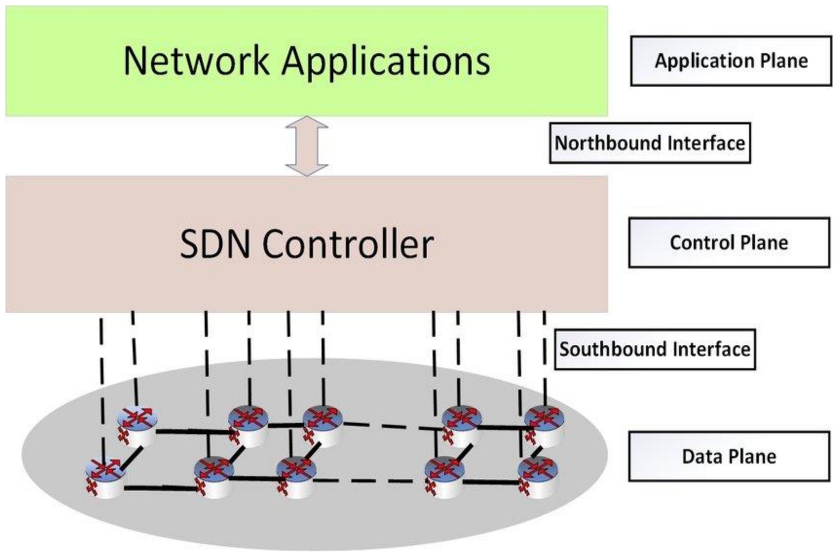

Figure 5: SDN architecture[20]

2.5.1 Basic Architecture

The basic SDN architecture is composed of following three elements:

2.5.1.1 Management plane

The management plane is a set of software that takes advantage of the northbound interface and control plane's capabilities. Using control stations, the management plane sets network strategies and controls and configures control functionalities. [20].

2.5.1.2 Control plane

The control logic of the system is implemented in the control plane. The control plane decodes the advanced network policies of the network application before informing the forwarding devices. Furthermore, control plane-based network applications have a single view of the network state. [20] .

2.5.1.3 Data plane

Forwarding devices and their connections make up the data plane. Through the southbound interface, the control plane programs the data plane elements. The most extensively used implementation of the SDN southbound interface is the OpenFlow protocol. [20].

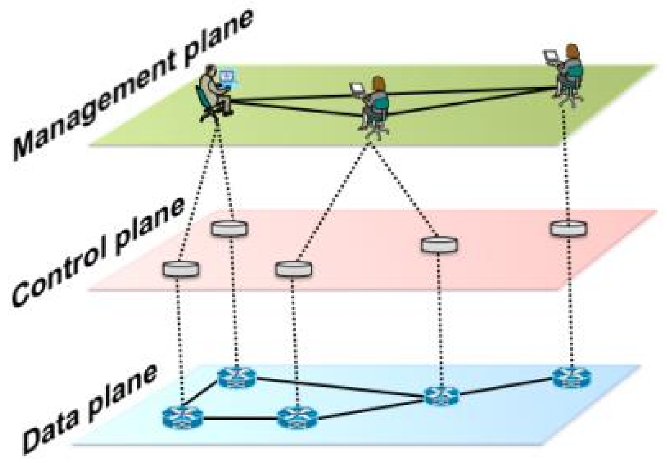

Figure 6: Basic SDN architecture

2.5.2 Benefits of SDN

The major benefit of SDN is that it separates the control and data planes, making the network more versatile and easier to manage. As a result, the new paradigm, the entire system can now be managed by a single distant computer known as the controller. For a variety of reasons, many business-related and industrialized enterprises are implementing SDN technology in their network environments that also includes:

- Network systems are easier to administer when the control plane and data plane are separated. Furthermore, any network improvement or modification becomes simpler, reducing the number of human errors. [6] IT managers may easily deploy network equipment and update network infrastructure without being bound by a single manufacturer.
- The SDN controller will have a worldwide view of the entire network since it has a centralized view of the entire network.

- Developers may use the upper layer of the SDN framework to deploy different applications that perform network services in a simulated environment.
- No programming language is needed for the devices that make up the infrastructure. As a result, relative to a traditional network, the operating costs would be considerably lower. SDN's market is constantly expanding due to its tremendous benefits.

Despite the many advantages of SDN technology, it is vulnerable to new security fears that attackers can use to perform a variety of malicious tasks. If an intruder gains access to the SDN controller, the entire device may be vulnerable to critical threats. Therefore, employing an IDS to detect anomalies in SDN network traffic is an essential part of the network architecture.

2.6 Deep Learning (DL)

DL is a branch of machinery learning that brings machine learning closer to artificial intelligence. It allows various levels of representation to be used to model complex relationships and concepts. The performance features from lower levels are used to create successively upper levels of abstraction using supervised and unsupervised learning algorithms [21]
.

2.6.1 Autoencoder

Auto-encoders, which our suggested method employs, are a popular methodology in deep learning research at this moment. A feature extraction approach based on unsupervised neural networks that learns the optimal factors to replicate its output as closely as possible to its input is known as an autoencoder. Its capacity to deliver a non-linear and more efficient generalization than Principal Component Analysis (PCA) is one of its attractive characteristics.

Backpropagation is used to do this, and the goal values are set to be equivalent to the inputs. To put it another way, it is attempting to learn a close estimate of the identity event. An auto-encoder usually has three layers: an input layer, an output layer, and a hidden layer. The hidden layer has a smaller dimension than the input, resulting in an under completion [22].

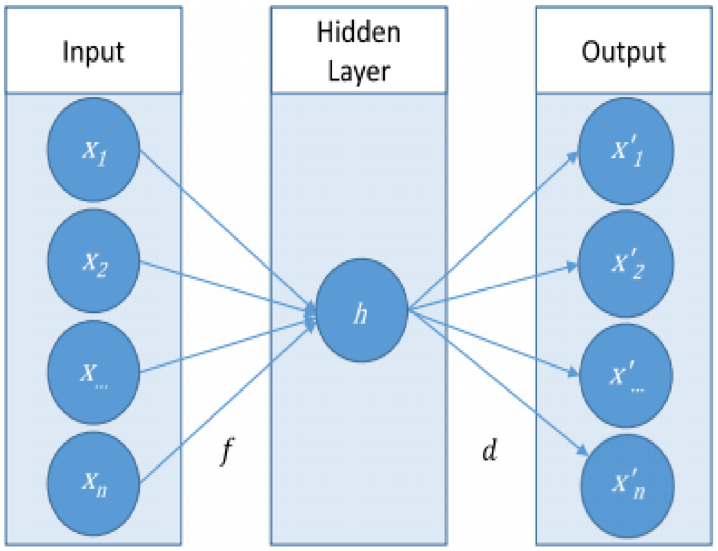

Figure 7: Single autoencoder[21]

2.6.2 Stacked Autoencoder

A deep auto-encoder, unlike a conventional auto-encoder, is made up of two regular deep-belief networks, one for encoding and one for decoding, each with four or five shallow layers. A technique known as a stacked auto-encoder can be used to apply deep learning to auto-encoders, in which the hidden layers represent basic concepts and numerous hidden layers are used to generate depth. This improved depth will lower computational costs and reduce the amount of instruction data needed, while also improving accuracy. Each hidden layer's output is used as the input for a successively greater stage. As a result, in most of the cases, the first layer of a stacked auto-encoder learns first-order features from raw input. The second layer is usually responsible for learning second-order features related to trends in the presence of first-order characteristics. Following layers teach us higher-order features. [22].

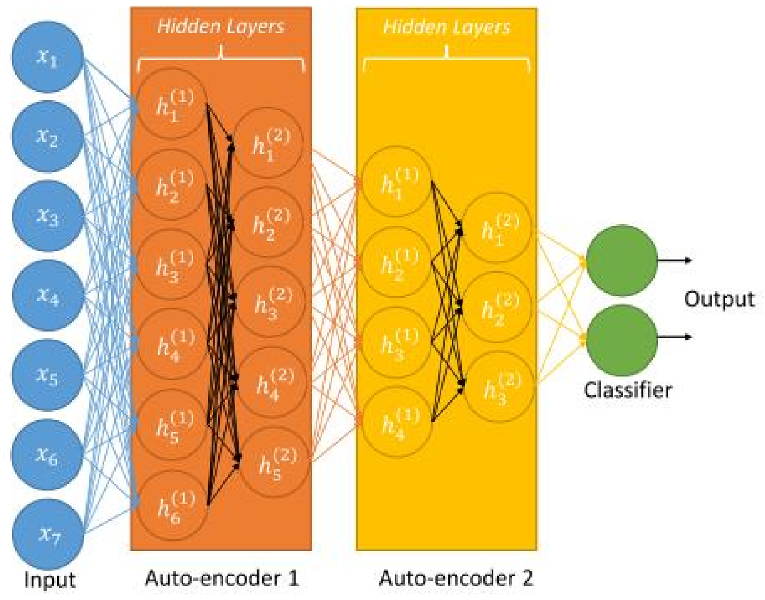

Figure 8: Stacked autoencoder

Chapter 03

LITERATURE REVIEW

3.1 Deep Learning Techniques

Deep learning techniques used in this research are:

- RNN Autoencoder
- LSTM Autoencoder

3.1.1 RNN Autoencoder

Autoencoders are a sort of artificial neural network that can perform a variety of functions including data noise filtering and image processing. In comparison to linear and kernel PCA, autoencoders can greatly enhance anomaly detection accuracy. It can discover tiny anomalies that linear PCA is unable of detecting. Furthermore, unlike kernel PCA, the autoencoder is simple to understand and does not require complex calculations. The autoencoder is made up of three layers. The input layer, which encounters the initial input Xi and encodes and decodes it using multiple hidden layers, is the first layer (encoder and decoder blocks). The encoded attributes are smaller than the input data during the encoder phase, and the encoded attributes are restored in reversed order during the decoder phase to start the final output at the last layer. The resulting output feature vector **Xi** is substantially equal to the original input data. An autoencoder is integrated with the regular RNN technique to produce a strong detection model for DDoS attacks. RNN can tackle an issue that traditional feed-forward neural networks can't. As a result, it is capable of developing models that are significantly more efficient and have better stratification precision. RNNs are widely utilized in a variety of fields, including word processing and speech recognition. The RNN's cyclic relations, in contrast to feed forward neural networks, may be used to successfully describe series [3]. RNNs excel at providing a standard way for remodeling data in a sequence with a high degree of sequence compactness and high similarity between discrete samples. It can be used to build other deep learning frameworks on top of it.

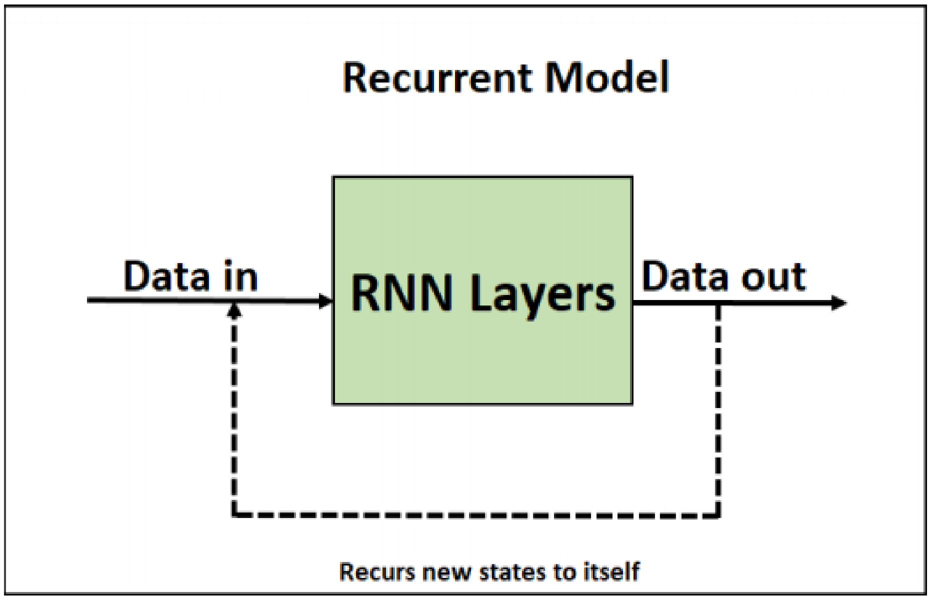

Figure 9: Recurrent Neural Network (RNN) architecture

3.1.2 LSTM Autoencoder

This is an RNN adaption in which a memory cell substitutes each neuron in addition to the conventional neuron, and each unit represents a position, with gates acting as multiplicative units for remodeling the information flow. The input gate chooses what goes to the memory cell's next device, the forget gate chooses whether the information received should be kept or forgotten within the internal state to allow data to be remembered, and the output gate chooses what goes to the memory cell's next device. LSTM excels in deciphering the context of Internet packets and extracting long and short-term dependencies, as well as trends in DDoS attack sequences [23]. LSTM has proved to be very effective at learning from experience and classifying processes like time series [24].

3.2 Types of Computer Attacks

Cyber-attacks can be classified based on the attacker's actions and goals. Each attack style can be categorized into one of four categories:

- Denial of Service (DoS) attacks
- Probing attacks
- User-to-root (U2R) attacks

- Remote-to-local (R2L) attack

3.2.1 DoS Attacks

DoS attacks are created to prevent or restrict users from accessing services provided by the network or device.

3.2.2 Probing Attacks

The aim of probing attacks is to collect data about the network or computer system.

3.2.3 U2R Attacks

The aim of a User-to-Root (U2R) attack is for a non-privileged user to gain root or admin user access on a device or machine where the attacker had user level access.

3.2.4 R2L Attacks

Sending packets to the victim machine is a remote-to-local (R2L) attack. The cybercriminal knows about the user's activities and gains access to the computer system's rights that an end user might have [1].

3.3 Distributed Denial of Service Attack (DDoS)

A distributed denial of service (DDoS) attack is an easy to execute but extremely effective method of attacking internet-based distributed networks. This form of attack has the potential to “unplug” the country's internet. One such incident occurred in Estonia in 2007, which resulted in the country's internet being disconnected. DDoS is regarded as a form of cyber warfare strategy. However, it is also used for blackmail and extraction. This DDoS attack can be carried out in connected as well as in wireless networks by flooding a server with packets in order to render it unresponsive. As a result, legal users would not be able to approach the server. A denial-of-service (DoS) attack is an effort to make a computer or network source inaccessible to its legitimate users. Over the past few years, denial-of-service attacks have become a growing epidemic, resulting in a rise in the number of victims concerned about the excellence of services. DDoS attacks are now very easy to execute with the advancement in technology. It is also possible to do it with the help of an automated platform. If an attacker discovers a device with weak protection, he can use this quick automated tool to attack it right away [25].

DDoS is a significant drain on available capital. Invaders can destroy the quality of service or disrupt the victim's connectivity in a cloud environment. The intruder first attempts to gain control of a large number of managers or systems, then employs these managers to carry out the attack. The major goal here is to prevent users from accessing the resource. The web server, CPU storage, and different network sources would be primary targets. DDoS can degrade the execution of cloud services by destroying fundamental machines in a cloud environment [25].

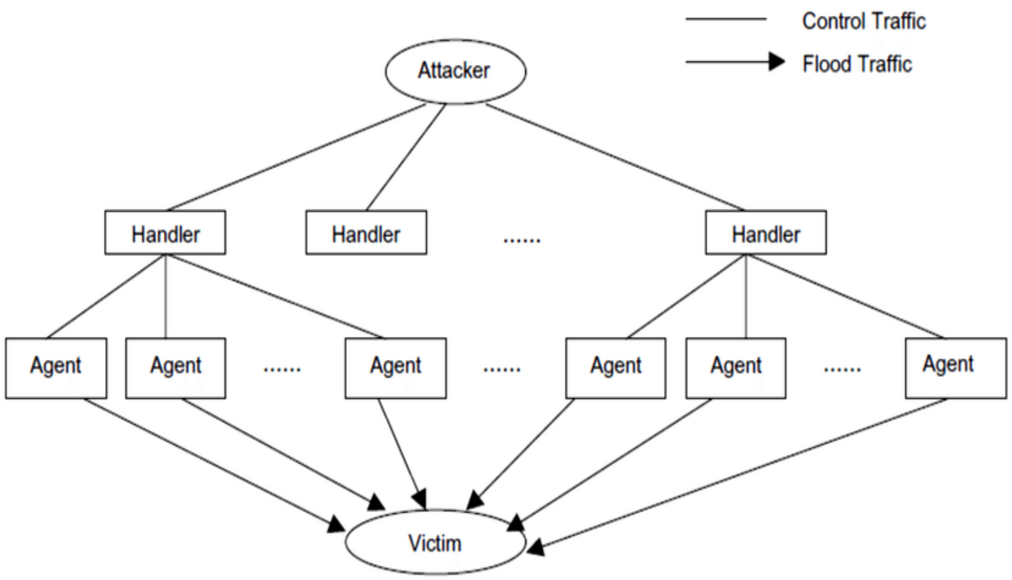

Figure 10: Architecture of DDoS Attack

3.3.1 Elements of Distributed Denial of Service (DDoS) Attack

A Distributed Denial of Service Attack is comprised of four elements [26]:

- The actual attacker; and
- The masters or trainers, who are infected hosts who run a special program that can monitor several agents.
- The assault Infected hosts that execute an application and are in charge of transmitting a stream of packets to the targeted victim are known as daemon agents, sometimes known as zombie hosts. Such machines are frequently both external to the victim's own network and external to the attacker's network, in the context to minimize culpability if the attack is traced back to the attacker.

- A victim or a target.

3.3.2 Steps while Conducting DDoS Attack

During the preparation and execution of a DDoS attack, the following steps occur:

- Selection of agents
- Compromise
- Communication
- Attack

3.3.2.1 Selection of Agents

The attacker selects the agents who will carry out the attack. An attacker must be able to exploit a flaw in these computers to get access to them. They should also have a large number of assets in order to establish a large number of effective assault streams. Initially, this process was done by hand, but scanning software rapidly automated it. [26].

3.3.2.2 Compromise

The attacker installs the attack code by taking advantage of security weaknesses and vulnerabilities on the agent machines. He also prevents the code from being deactivated and detected. The owners and users of agent systems are frequently unaware that their systems have been hacked and are about to be subjected to a DDoS attack. When engaging in a DDoS attack, each agent program consumes a small quantity of assets, resulting in limited performance degradation for computer users [26].

3.3.2.3 Communication

To establish which managers are active, when to plan incidents, and when to update managers, the attacker interacts with a variety of trainers. Depending on how the attacker structures the DDoS assault network, agents may be directed to engage with a single trainer or multiple trainers. [26]

3.3.2.4 Attack

At this point, the attacker gives the order to start the attack. The victim, the duration of the attack, and the attack's precise properties, such as shape, length, TTL, port numbers, and so on, can all be altered. In order to escape detection, the attacker will profit from the type of attributes of attack packets.[26].

3.4 Types of Distributed Denial of Service (DDoS) Attack

The five types of Distributed Denial of Service (DDoS) attack are:

- Denial of sleep attack
- User Datagram Protocol (UDP) flood attack
- Internet Control Message Protocol (ICMP) (ping) flood
- Synchronize (SYN) flood
- Ping of Death (POD)

3.4.1 Denial of sleep attack

A denial of sleep attack is launched against node power consumption. The attackers in this type of assault are knowledgeable with MAC (Medium Access Control) coating and have the ability to bypass verification and encoding protocols. The MAC layer technique was built specially for wireless sensor nodes to save battery life by putting the radio in a low-power mode. When the connection is not in use, the MAC protocol can solve the radio's main causes of power loss, such as collisions and control packet overhead [25].

3.4.2 UDP flood attack

The UDP protocol is misleading because data packets or requests can be transmitted out of order, appear to be duplicated, or be delayed. As a result, the UDP protocol enables data and requests to be sent to a server without requiring a response or acknowledgement of receipt. Because UDP protocols do not require a connection, they generate a higher bandwidth DDoS attack and are easy to set up because they don't require any authorization to pass packets. This consists of larger-than-average messages sent to the target by the rogue node, consuming network bandwidth [25].

3.4.3 ICMP (Ping) flood

ICMP is a protocol that works in the same way as UDP. Without waiting for a response, the ICMP Ping request sends out packets as rapidly as possible. As a result, approaching and extroverted bandwidth will be increased, perhaps jeopardizing the port queue size. [25].

3.4.4 SYN flood

In this situation, the attacker attempts to keep the link open by sending packets to the server. During the link time, other systems would be unable to reach the server, which is one type of DDoS assault.

In a faked SYN flood, the attacker tries to provide a higher number of Transmission Control Protocol (TCP) SYN packets with a bogus Internet Protocol (IP) address. [25]

3.4.5 Ping of Death (POD)

POD is an old attack that is no longer a threat to the machine. The IP protocol establishes a limit on the number of packets that can be sent between two devices. 65535 bytes is the maximum IPv4 allotment. The receiving server will fail if a larger volume is sent than the maximum buffer size because it will seek for packet data that is larger than the maximum buffer volume. [25]

3.5 DDoS Detection

Methods for DDoS detection are as follows:

- Detection using Fast Entropy approach
- Detection using Naïve Bayesian classification method
- Detection using Tennessee Eastman Challenge
- Detection using TCP Congestion Window Analysis

3.5.1 Detection using Fast Entropy approach

The DoS attack is detected using an adaptive threshold mechanism. This system tracks traffic flow in real time and records the flow count value at each time interval. Each channel's traffic flow is determined. When the flow count value is higher, an attack occurs, and entropy drops dramatically as one flow count dominates. However, in the event of normal flow, however, the entropy will remain the same.

The information contained in the packet headers is fully dependent on this mechanism. The header contains the IP addresses of the sender and receiver, as well as the details of the traffic flow; the data in the flow is meaningless to it. One drawback of this tactic is that attackers may attempt to send packets from multiple systems with low traffic flow, causing the fast entropy technique to miss the attack. The result of the Fast Entropy Method is evaluated using the header data, and then the attack is decided. [25]

3.5.2 Detection using Naïve Bayesian classification method

Feature selection is critical in the Naive Bayesian classification system for extracting the minimal or ambiguous data from the available datasets. Data mining would disclose a plethora of information about network traffic patterns. It's critical to prioritize the data required for identification over the data that isn't required. [25]

3.5.3 Detection using Tennessee Eastman Challenge (TEP)

The TEP-determined variables are grouped into two-dimensional clusters. The data is clustered using the Gaussian mixed model. This strategy facilitates in the differentiation of valid and harmful data. Incorporating the path coefficients and bi-dimensional graphs yields a global system. The cluster is next subjected to a method of internal evaluation. This accentuates the outlier data points produced by attackers. Clustering allows the outcomes of DoS attacks to be readily determined [25].

3.5.4 Detection using Transmission Control Protocol (TCP) Congestion Window Analysis

This method examines the TCP congestion window, which is evaluated using the maximum count, to detect TCP-based flooding attacks. This method clears the accused's transmission depending on factors such as address, terminal, or protocol at first. This method requires less memory and calculations than previous change point classification algorithm. It's utilized to detect a two-sided function that takes into the account both the positive and negative aspects of the flow. [25]

3.6 Characteristics of DDoS

A DDoS attack can't be carried out with just one or two systems. Even if the assault was limited to a single device, the consequences would be minimal. The attacker tries to make the attack more vulnerable in order to have an adverse influence on the general device. This is not possible with a single system, so the attacker attempts to manipulate multiple techniques by using their IP and then taking self-control of the systems. Zombie systems are ones that are controlled by a single machine. All other systems are controlled by the master zombies, who are the ultimate machine.

Botnets are a group of malware-infected devices that are managed and controlled remotely by a separate entity. These botnets might be turned into automated weapons that may be used in cybercrime to target a device. Their lifespans will range from one to eighteen months, and they will all be connected to the same internet subnet. Botnets may also be built using mobile nodes.

Mobile nodes can potentially be used to produce malware as a result of the growing use of mobile devices. It can only be detected via an Android application because it is a mobile node.

A Denial of Service (DoS) attack is a network-based threat that causes supercomputer memory to become too hard or full to manage resources through other means. Because of the assault, the website is unable to service its clients, and real consumers lose trust in the website or server. During a DDoS assault, the server might get thousands of requests at once, filling the device memory and rendering the server unable to process any further requests from the client. As the number of requests increases, the computer will cease answering and, in some circumstances, will be turned off for a period. Depending on the impact of the attack, the shutdown could last anywhere from a few seconds to several hours.

As technology improves, DDoS assaults have gotten more easier to carry out, and the effects are considerably harsher than previously. In wireless sensor nodes, the assault acts significantly differently. As a result, it's apparent that a DDoS assault may have a variety of effects on different systems. In the case of wireless sensor nodes, which is also known as a black hole assault, the assault must be quicker. The attack must be identified before it impacts our system to have a more identical solution [25]

3.7 Distributed Denial of Service (DDoS) Classification

It is important to provide a systematic description to be able to recognize DDoS attacks:

3.7.1 Classification by Degree of Automation

DDoS attacks can be divided into the following categories based on the degree of automation of the attack:

- Manual DDoS attacks
- Semi-automatic DDoS attacks
- Automatic DDoS attacks

3.7.1.1 Manual DDoS attacks

Manual DDoS assaults were common in the beginning. This implies that the DDoS assault involves looking for vulnerabilities in distant machines, breaking in, and installing the attack code.

All of these procedures were eventually mechanized through the use of semi-automatic and automatic DDoS assaults. [26]

3.7.1.2 Semi-automatic DDoS attacks

In semi-automatic assaults, the DDoS attack is part of the agent–handler attack paradigm. The attacker tests and deceives the handlers and agents using automated scripts. The attack type, the victims' addresses, and the assault's start time are all determined by the handler computers. [26]

3.7.1.3 Automatic DDoS Attacks

In automated DDoS assaults, interaction between the attacker and the agent computers is completely avoided. In most cases, the assault is limited to a single command. All the attack's features, such as the attack class, duration, and victim's address, are pre-programmed into the attack code. [26]

3.7.2 Classification based on protocol level

Centered on the protocol level attacked, DDoS attacks can be divided into two categories:

- Network/transport-level DDoS flooding attacks
- Application-level DDoS flooding attacks

3.7.2.1 Network/transport-level DDoS flooding attacks

TCP, UDP, ICMP, and DNS protocol packets are commonly used in these attacks, which aim to interrupt legitimate user communication by draining the victim network's bandwidth [27] .

3.7.2.2 Application-level DDoS flooding attacks

These attacks primarily aim to disrupt legitimate users' services by depleting server sources (such as plugs, CPU, memory, disk/database bandwidth, and I/O bandwidth) [27].

3.7.3 Classification by Victim type

DDoS attacks can be divided into three types based on the category of victim:

- Application
- Host

- Network

3.7.3.1 Application

They want to take advantage of flaws in software systems and/or specific versions of operational systems, as well as hardware platforms [28] .

3.7.3.2 Host

The goal in this case is a specific host. As a result, all services running on the host are impacted. The most common targets for these attacks are network access and, in some cases, power resources [28]

3.7.3.3 Network

These attacks target elements that provide interconnection, such as router interfaces or switch trunk connections, with the goal of separating large segments or even the whole network. They usually necessitate a lot of traffic [28]

3.7.4 Classification by Exploited Vulnerability

DDoS attacks can be classified into the following types based on the exploited vulnerability:

- Flood attacks
- Amplification attacks
- Protocol exploit attacks
- Malformed packet attacks

3.7.4.1 Flood attacks

In a flood attack, zombies inundate a target device with a large volume of IP traffic in attempt to jam the victim's bandwidth. The zombies' packet streams affect the target device in a number of ways, from slowing it down or crashing it to overloading network capacity. Two well-known flood attacks are UDP and ICMP. [28]

3.7.4.2 Amplification attacks

In amplification assaults, the attacker or agents use the broadcast IP address feature on most routers to amplify and depict the assault by sending messages to a broadcast IP address. To enhance the amount of assaulting movement in this form of DDoS attack, the attacker may send the broadcast message directly or through agents. [26]

3.7.4.3 Protocol exploit attacks

Protocol exploit attacks use a considerable amount of the victim's resources by exploiting a specific feature or design flaw in a protocol implemented on the target. A protocol exploit attack such as the TCP SYN assault is an example. [26]

3.7.4.4 Malformed packet attacks

In order to force the victim's machine to crash, malformed packet assaults rely on agents delivering the target improperly constructed IP packets. Malformed packet assaults can be classified into two types: Attacks on IP addresses and IP packet selection. [26]

3.8 Mitigation of Distributed Denial of Service (DDoS) Attack

The major goal of DDoS assaults is to pinpoint the source of the assault and immediately correct it. A DDoS assault is caused by an inflow of packets to the target system. Real users can try to send large quantities of packages; these users must be taken into the account, and service must be provided to them. A handful of the strategies utilized for improvement are listed below.:

- Fuzzy estimator approach
- Mitigation using Multivariate Correlation Analysis (MCA)
- Stone approach
- Active Queue Management (AQM) method

3.8.1 Fuzzy Estimator Approach

The main goal of DDoS assaults is to figure out what's causing the problem and solve it as quickly as feasible. A DDoS assault is caused by an inflow of packets to the target system. Real users may attempt to send a very large number of the packages; these users must be taken into account and services provided to them. A handful of the approaches that have been taken to improve the issue are listed below. In this scenario, the packet appearance interval and the quantity of packets delivered are utilized to determine the attack. The fuzzy estimator will be used to set the maximum number of packets that the server or device will receive. If the total number of packets gathered exceeds a certain threshold, an attack will be considered. That server's transmissions would be lost in an instant. The identification of false positives is the research's major flaw. In other words, if a lawful user tries to send the most packets, he will be classed as an attacker. [25]

3.8.2 Multivariate Correlation Analysis (MCA)

The MCA-Based analysis technique employed anomaly-based detection to detect the assault. This assists in the effective disclosure of known and unknown DDoS assaults by analyzing the patterns of normal network traffic. The suggested detection method compares a fresh traffic record to its typical characteristics. The traffic record is marked as an assault when the distinguishing value surpasses a preset threshold.

A suggested triangular area-based MCA method is employed and developed to evaluate genuine network traffic. Triangular Area MCAs then provide the quality attributes for regular profile creation (TAMs). For a more precise characterization of network traffic actions, geometrical relationships concealed in discrete sets of two different characteristics inside each network traffic record are examined, as compared to the previous techniques. Both known and unknown DoS assaults may be identified from network traffic utilizing the aforementioned methods of services. [25].

3.8.3 Stone Approach

This method employs a contemporary technology termed as stone for mitigation reasons. Vincenzo Golisano built distinct stone machines that can execute the stone algorithm and are linked to other machines to detect and mitigate DDoS assaults in this scenario. It may be used to defend both a single host and a group of hosts. The job of the stone system is to keep watch of the network, identify possible assaults, and filter traffic when it surpasses the maximum bandwidth limit. The Detection Control Center (DCC) is used to detect abnormal traffic in the system. DCC data is used by the Mitigation Center (MC) to establish attack characteristics and a system's tolerance level. The MC identifies assaults using this precise information and provides mitigation solutions. To filter away packets with a lot of data and bandwidth, the MC filtering protocol is employed. Packets will be regulated and then fall if the load becomes too much for genuine users. This is how the stone technique works in practice. [25].

3.8.4 Active Queue Management (AQM) Method

In this AQM approach, the Deterministic Fair Sharing (DFS) methodology is utilized to detect false positives. To ensure the least amount of impact to legitimate flows, the AQM method identifies harmful network flows and removes all identified malicious packets while retaining legal flows. It shouldn't confuse a valid flow with a malicious flow, and it should be able to tell the difference between the flows. This is why DFS is utilized. Before receiving or accepting packets, DFS employs the ENQUE and DEQUE functions to execute a series of actions. Attacks are detected using data from the DFS, which maintains track of a variety of characteristics depending on network activity. In some cases, the packets may appear to be a valid flow at first, However, once approved, they might create a harmful flow. As a result, the DFS has been fine-tuned to identify these types of assaults. When the DFS identifies a malicious flow, all packets connected with that flow are automatically blocked [25].

3.9 Taxonomy of Distributed Denial of Service (DDoS) Attacks

Taxonomy of DDoS attacks in words of reflection-based and exploitation-based attacks is described below:

3.9.1 Reflection-based Attacks

Reflection-based attacks conceal the attacker's identity by using a legitimate third-party variable. In order to flood the target victim with response packets, attackers send them to reflector servers with the source IP address set to the target victim's IP address. These attacks can be carried out using application layer protocols that use transport layer protocols like TCP, UDP, or a combination of both. TCP-based attacks like MSSQL and SSDP exist, while UDP-based attacks like CharGen, NTP, and TFTP exist. DNS, LDAP, NETBIOS, and SNMP are examples of attacks that can be carried out using either TCP or UDP [29].

3.9.2 Exploitation-based Attacks

Exploitation-based attacks use a valid third-party attribute to hide the attacker's identity. In order to provide flood, the target victim with response packets, attackers send packets to reflector servers with the target victim's IP address as the source IP address. Instead of using application layer protocols, these attacks can be accomplished using transport layer protocols like TCP and UDP. SYN flood is a TCP-based manipulation attack, whereas UDP flood and UDP-Lag are UDP-based attacks. A UDP flood attack is initiated by sending a very large number of UDP packets to the remote host.

These UDP packets are sent to random ports on the target device at a very fast rate. As a result, available bandwidth on the network is exhausted, the system crashes, and performance suffers. The SYN flood, on the other hand, uses the TCP three-way handshake to consume server services. Before the target device fails or malfunctions, the attack is initiated by repeatedly sending SYN packets to it. The UDP-Lag attack aims to disrupt the link between the client and the server. This technique is most often used in online gaming to outmaneuver other players by stopping or interrupting their progress. This attack can be carried out in two ways: with a hardware switch called a lag switch, or with a software program that runs on the network and absorbs the bandwidth of other users [29].

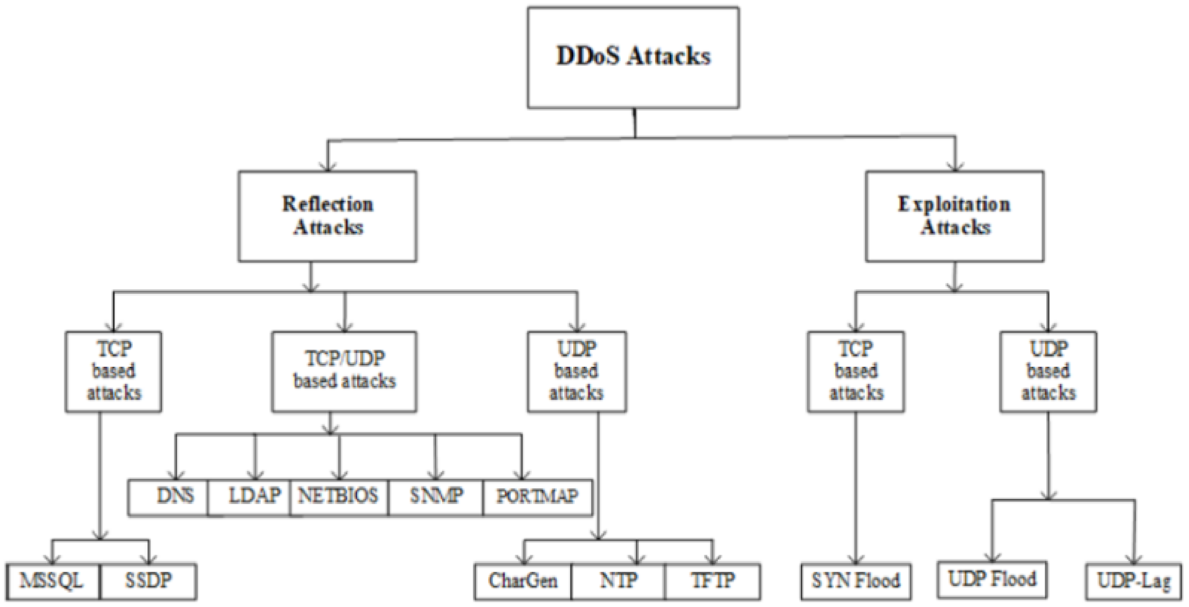

Figure 11: Taxonomy of DDoS Attacks

3.10 Dataset

Dataset availability is a prerequisite for ML/DL intrusion detection techniques. The paucity of datasets in the intrusion detection state is mostly due to privacy and regulatory concerns. The network traffic contains extremely private information, and its accessibility will reveal the secrets of consumers and organizations, as well as personal conversations. To prevent any sensitive issues, many scientists simulate their own data to fill the previous gap. Most of the datasets produced are incomplete, and the row samples used to cover the application behaviors are insufficient in these cases. KDDCUP99, NSL-KDD, Kyoto 2006+, ISCX2012, and CICIDS2019 are the most common in the public domain datasets that have been considerably utilized for intrusion detection [21].

The following two datasets are used to validate our proposed classifier in this study:

- InSDN dataset
- CICIDDoS2019 dataset

3.11 InSDN Dataset

Different attacks on the files, control, and device layers are included in the InSDN dataset. The dataset's attack sources are divided into two categories:

- Internal
- External

3.11.1 Internal

Internal users with complete entry to the SDN network are the source of these attacks. Internal attacks are uncommon in production networks, but they become more serious as time goes by, and they may trigger malicious behavior in network components. In certain cases, the attacker is unable to target network servers directly because they are protected by a high level of protection. Before initiating fresh assaults on other target servers, the attacker seeks to exploit weaknesses in the network system's individual users. The compromised hosts in the InSDN dataset are used to initiate different attacks from an internal SDN network [21].

3.11.2 External

Outside networks are often used to initiate these attacks. The attacker is primarily changing the SDN network with malevolent activities such as code vulnerabilities, refusal of service attacks, malware, and so on. We believe that the margin of the attacks in the dataset were generated from a third-party network in order to simulate real-world attack scenarios [21].

3.11.3 Limitations to InSDN Dataset

Only the ONOS SDN controller was used to construct the InSDN testbed. Other controllers' various forms of functionalities in terms of security analysis are overlooked. Different security modelling and, as a result, different countermeasures may be used by different controllers [21].

All attacks are assumed to be created by high-level ability attackers in the InSDN dataset. The dangers posed by misconfiguration or contradictory flow-tables in the switches are overlooked. A high-class disparity is one of the proposed dataset's key flaws. This issue can cause the IDS to be biased in favor of the majority class, resulting in a high number of false alarms and low evaluation accuracy.

3.12 CICDDoS2019 Dataset

CICDDoS2019 is a collection of benign and up-to-date popular DDoS attacks that closely resemble real-world data. It also contains the findings of a network traffic study applying CICFlowMeter-V3 with labelled flows based on the time stamp, source and destination IPs, source and destination ports, protocols, and attack vectors (CSV files).

The dataset includes a huge number of different DDoS attacks that can be taken out using TCP/UDP-based application layer protocols. The attacks in the dataset are classified as either exploitation-based or reflection-based attacks. The dataset was collected on two distinct days for training and research reasons. [30] The command set was recorded on January 12th, 2019, and it contains 12 different forms of DDoS assaults, each in their unique PCAP format. The attack methods addressed throughout the training day include UDP, SNMP, NetBIOS, LDAP, TFTP, NTP, SYN, WebDDoS, MSSQL, UDP-Lag, DNS, and SSDP DDoS assaults [5].

Chapter 04

RESEARCH METHODOLOGY

4.1 Deep Learning (DL) Approach

In traditional machine learning, essential input features are physically programmed, and the algorithm grasps to map the elements to an output automatically. There are various levels of attributes in deep learning. These characteristics are found spontaneously, and they are combined at different levels to create outputs. Each level reflects theoretical features that are discovered as a result of the previous level's features [31].

The InSDN and CICIDDoS2019 datasets, both recently released, were used to test our model. Since our primary goal is to solve the problem of binary classification in IDS, we compare all datasets and their outcomes in our proposed technique.

4.1.1 Classification

Classification is the method of transmission of a specific class to each and every instance of a dataset under consideration. Regular and abnormal means that the well-known structure is utilized for new examples. It is useful for both misuse detection and anomaly detection, but it is more commonly used for the former. The datasets were classified into predetermined sets through classification [32].

4.1.1.1 Binary Classification

Classification task with only two possible outcomes, such as pass or fail, etc. This type of problem is exemplified by logistic regression. Using binary classification in deep learning, we must obtain two categories of data i-e normal and malicious packets, on which the neural network will be trained. Since neural networks can only operate with mathematical data, we must mark network packets with either a 0 or a 1 to indicate whether they are natural or malicious [33].

4.2 InSDN Dataset Preparation

We focus on a binary classification issue in this paper and do not go into detail on how to distinguish different types of attacks. Anomaly traffic data refers to findings that belong to some attack class. Preparing the dataset for proper use is the first step before training the IDS model.

4.2.1 Data Pre-processing

Following are the steps taken to pre-process the entering flows [7]:

- Source IP, Destination IP, and Flow ID are included in the created dataset as a socket info. We disable all socket characteristics to prevent the problem of overfitting, which occurs when data is transferred from one network to another. Aside from the traffic group, the final dataset contains 77 different features. The magnitude of the training and testing records were slightly high, so a few samples from the whole dataset were chosen. As a result, the time spent on model training can be kept to a minimum.
- Because the features' ranges vary, they must be standardized.
- To transform the labelled string to mathematical values, one-hot encoding was used. From input data, only binary classification was used in this model to distinguish between malicious and normal traffic. As a result, the usual string was encoded as a binary value of 0 and binary value of 1 is assigned to all malicious traffic.

4.3 Simulating the Normal Traffic Data

RNN has been successfully used in the identification of anomalies in conventional networks. With such approaches, the model can be trained to reduce loss while still delivering high efficiency. The autoencoder also has a benefit in terms of the number of classification problems. The autoencoder was chosen because it tries to understand the leading variables to recreate the input at the output layer, which is why we chose it for our future model for anomaly detection. In addition, our model employs LSTM with an autoencoder to find out the network dataset's statements in a semi-supervised manner [23].

It has a number of encoder and decoder layers, each with a large number of LSTM units. Xt is transformed into Zt, a fixed range function vector, via the encoder block. Input data Xt IR77*1 is the dataset's primary encoded feature vector. The encoded data is subsequently passed to the decoder block, which produces the function vector as a result. The decoder block's input function vector is represented as X_{ct}. As the encoder block's layers, the decoder block’s layer is adjusted in the opposite sequence. The output feature vector Z_{ct} is produced by feeding the encoded features Z_{ct} through a series of LSTM blocks. After the 1st, 2nd, 3rd, and 4th layers of the decoder, the dimensions are increased to 16, 32, 64, 128, respectively. Finally, by feeding the last layer of the decoder block to a fully connected layer, output function vector Z_{ct} is produced [23].

At the top layer, with two channels a SoftMax regression layer was used. The decoder output was taken by the SoftMax layer and divides the data into two categories: regular and attack traffic. For each label class, the SoftMax function returns a value in the range (0, 1), with the total likelihood values for all classes equaling unity [30].

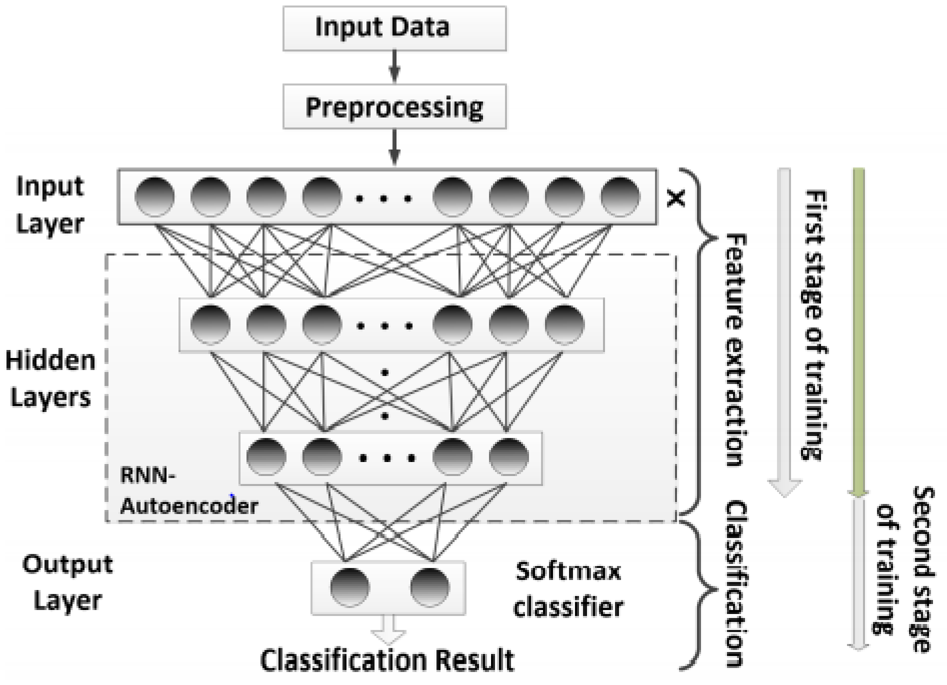

Figure 12: An attack detection model in SDN networks

4.4 CICDDoS2019 Dataset Preparation

In this research, we use the newly published CICDDoS2019 dataset to test our proposed classifier. The dataset consists of a huge number of various DDoS attacks that can be executed using TCP/UDP-based application layer protocols. The attacks in the dataset are classified as either exploitation-based or reflection-based attacks. Choosing appropriate elements in intrusion systems is a difficult project that necessitates the assistance of professionals. Attack scenarios change on a daily basis, making it impossible to pick appropriate features for a particular form of attack [32].

4.4.1 Data Pre-processing

We prepare the data in such a way that it is appropriate for the training model right away. In a flow-based format, the CICDDoS2019 dataset is available, with CICFlowMeter extracting over 80 features. The following are a couple of the steps that were taken to organize the data prior to the module training [30]:

4.4.1.1 Removing Socket Attributes

Remove Source IP, destination port, destination IP, flow ID and timestamp from the socket. Since these characteristics vary from one network to the next, the model must be trained using packet characteristics Furthermore, an intruder and a regular user can share the same IP address. As a result, preparing the DL model with socket data will lead to overfitting, as the model will be influenced against the socket data. After deleting the unnecessary features, a total of 77 features were left for the model input.

4.4.1.2 Cleaning the Data

There are a lot of missing and infinity values in the original results. Many of the data's values were deleted.

4.4.1.3 Normalizing the Input Data

Using original data to train the model would result in classification errors, and then the model takes a long time to learn. The data was normalized to have a minimum value of 0 and a maximum value of 1.

4.4.1.4 Encoding the Labeled Data

Our model was trained to identify input traffic as natural or malicious using binary classification. As a result, in addition to regular traffic, we consider all DDoS groups to be attacks. The string values for regular and attack labels are then encoded to binary values of 0 and 1, respectively.

4.5 Andrew’s Curve

In data visualization, an Andrews map, also known as an Andrews curve, is a method of visualizing structure in high-dimensional data. It is essentially a smoothed version of a similar coordinate plot or a rolled-down, non-integer version of the Kent-Kavita radar m map. It bears the name of statistician David F. Andrews [33].

4.6 Receiver Operating Characteristic (ROC)

The diagnostic capacity of a binary classifier system changes as the discrimination threshold is changed, as shown by a receiver operating characteristic (ROC) curve. Starting in 1941, the method was developed for military radar receiver operators, hence the term.

The ROC curve is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings. It was used to evaluate the proposed approach's effectiveness [34].

Chapter 05

RESULTS & DISCUSSIONS

5.1 Classification Algorithms

Deep learning is a method of predicting outcomes or defining a classification based on historical data using a mathematical model [35]. The following are some of the mathematical models/classification algorithms that have been used in this context:

5.1.1 Logistic Regression (LR)

In LR, you can check whether two variables are linearly related and calculate the linear relationship's power. It gives a straightforward and effective approach for resolving a broad range of issues [36].

5.1.2 Gradient Boosting (GB)

GB is an iterative ensemble method for supervised activities that incorporates multiple weak learners to create a solid ensemble. The basic principle behind this algorithm is to generate new base-learners that are maximally correlated with the ensemble's negative gradient of the loss function [37].

5.1.3 Naïve Bayes (NB)

The Bayes theorem and the principle of freedom was used to build a probabilistic model called a Bayes classifier. A given example defined by its characteristic vector is assigned to the most probable class by NB [38].

5.1.4 K-Nearest Neighbors (KNN)

Data can be classified and predicted using KNN. The number of nearest neighbors that the classifier can acquire and use to make a prediction is denoted by the letter k in KNN [39].

5.1.5 Decision Trees (DT)

DTs are trees that can be updated incrementally by splitting the data set into smaller data sets which are used to solve classification and regression problems. For each new element in the test set, the decision tree must be traversed from the root to one of its leaves. This means that each node in the tree must be tested and allocated to one of the sub-trees based on the value before the element reaches a leaf node [40].

5.1.6 Random Forest (RF)

It is a method that can be used for estimating as well as classification, and it is relatively simple to learn. Its high learning output and low input planning and hyperparameter tuning demands account for this choice. Essentially, it's a method for combining multiple decision trees based on database input data [41].

5.1.7 Support Vector Classifier (SVC)

SVC is a set of supervised learning techniques for analyzing and recognizing patterns in data. It's often used in classification and regression analyses. It's ideal for non-linear, high-dimensional data isolation and classification problems that don't necessitate any prior experience [42].

5.2 Results for InSDN Dataset

The performance evaluation of the InSDN dataset is discussed in detail in this section.

5.2.1 Fully-featured version

Table 1 shows the results of various classifiers when utilizing a fully functional version of our dataset. The total score metrics for the DDoS class are clearly very high for all classifiers, whereas the normal class has poor performance metrics. This is due to the fact that DDoS attacks are frequently distinct from normal traffic patterns. It was observed that Logistic Regression (LR) and Support Vector Classifier (SVC) algorithms have higher accuracy while Gradient Boosting (GB) gives low accuracy.

Table 1: InSDN Dataset Results

	Precision		**Recall**		**F1-score**		
	DDoS	**Normal**	**DDoS**	**Normal**	**DDoS**	**Normal**	**Accuracy**
LR	0.99279	0.94193	0.92400	0.99459	0.95716	0.96755	**0.96307**
GB	0.56090	0.97802	0.98955	0.37506	0.71597	0.54219	**0.64944**
NB	0.59482	0.97110	0.98304	0.45980	0.74117	0.62409	**0.69343**
KNN	0.98120	0.94228	0.92515	0.98570	0.95235	0.96350	**0.95866**
DT	0.74415	0.94620	0.94805	0.73705	0.83382	0.82863	**0.83126**
RF	0.98290	0.94256	0.92544	0.98701	0.95330	0.96427	**0.95952**
SVC	0.99474	0.94181	0.92371	0.99606	0.95791	0.96817	**0.96375**

5.2.2 Results for CICDDoS2019 Dataset

In this technique, 6 different algorithms were considered such as LR, GB, NB, KNN, DT, RF and SVC. Table 2 shows the classification metrics for our model's attack and benign classes using several classical methodologies. DDoSNet performs better as compared to other datasets. Our proposed model outscored LR and KNN, according to our findings. The NB classifier scored poorly, owing to the NB's assumption that all qualities are irrelevant to one another. As a result, its performance was degraded because the considered feature attributes are interdependent.

Table 2: CICDDoS2019 Dataset Results

	Precision		**Recall**		**F1-Score**		
	Attack	**Benign**	**Attack**	**Benign**	**Attack**	**Benign**	**Accuracy**
LR	0.95543	0.97999	0.97813	0.95913	0.96665	0.96945	**0.96811**
GB	0.99765	0.91102	0.89116	0.99812	0.94140	0.95258	**0.94758**
NB	0.98841	0.53484	0.02925	0.99969	0.05681	0.69686	**0.54118**
KNN	0.99151	0.94550	0.93610	0.99282	0.96301	0.96858	**0.96602**
DT	0.92754	0.88900	0.86907	0.93920	0.89736	0.91341	**0.90606**
RF	0.99917	0.89850	0.87396	0.99935	0.93238	0.94625	**0.94010**
SVC	0.98853	0.88340	0.85393	0.99113	0.91632	0.93417	**0.92631**

5.3 Evaluation Metrics

In general, the accuracy (AC), precision (P), recall (R), and F-measure of IDS output are evaluated (F). An IDS must have a high level of accuracy, a high-level detection rate, and a low rate of false alarms. These parameters are calculated using an uncertainty matrix [37].

In the confusion matrix:

- True Positive (TP) refers to the number of attack records that have been properly classified.
- True Negative (TN) is the number of regular documents properly categorized.
- The number of usual documents that have been classified wrongly is referred to as False Positive (FP).
- The number of attack documents classified wrongly is referred to as False Negative (FN).

5.3.1 Accuracy (AC)

The ratio of true identification over the total traffic trace is shown.

5.3.1.1 The accuracy is defined by

$$\text{AC} = \frac{TP+TN}{TP+TN+FP+FN}$$

5.3.2 Precision (P)

It demonstrates how many of an IDS's predicted intrusions are actually intrusions. The higher the P, the lower the chance of a false alarm.

5.3.2.1 The precision is defined by

$$\text{P} = \frac{TP}{TP+ FP}$$

5.3.3 Recall (R)

It compares the percentage of expected intrusions to the total number of intrusions.

5.3.3.1 The appropriate definition of Recall

$$\text{R} = \frac{TP}{TP+ FN}$$

5.3.3.4 F-measure (F)

It provides a more accurate assessment of an IDS's accuracy by considering both P and R.

5.3.3.4.1 The definition of F-score

$$\text{F} = \frac{2}{1/P+ 1/R}$$

5.5 Andrew's Curve for CICDDoS2019 Dataset

Generally, Andrew curves are used to see the distinct vision, here it is clearly visible in the space of high nonlinearity.

5.5.1 For Selected Features

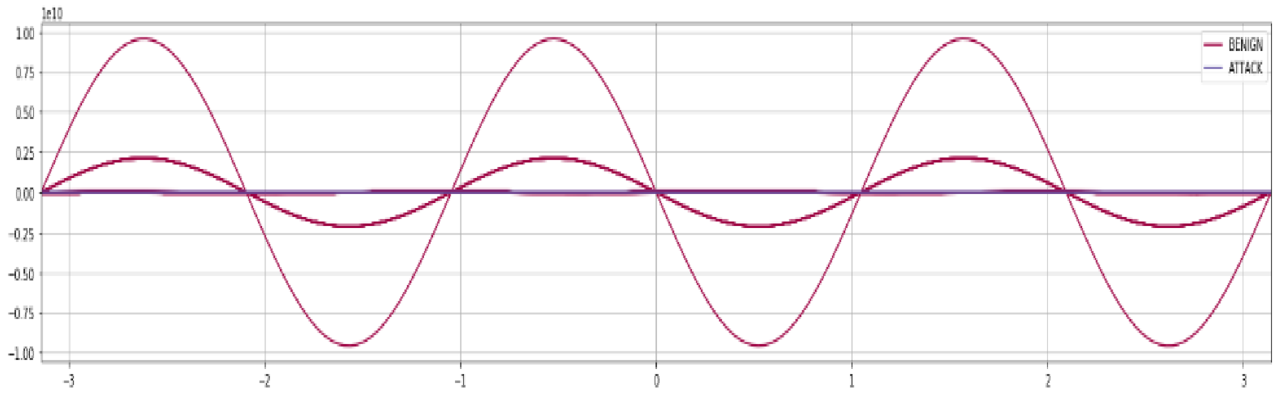

Figure 13: Andrew's curve for selected features of CICDDoS2019 dataset

5.5.2 For all Numerical Features

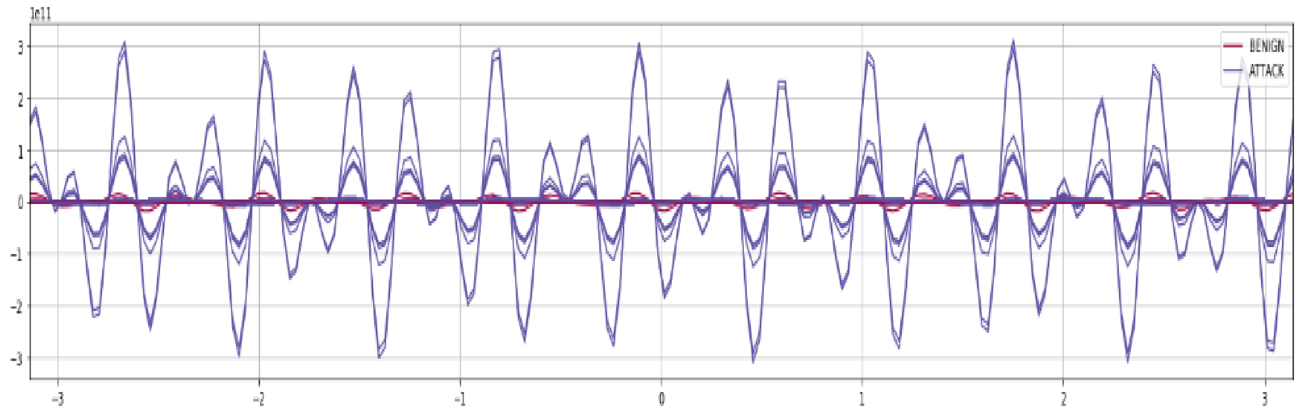

Figure 14: Andrew's curve for all numerical features of CICDDoS2019 dataset

5.6 Andrew’s Curve for InSDN Dataset

5.6.1 For Selected features

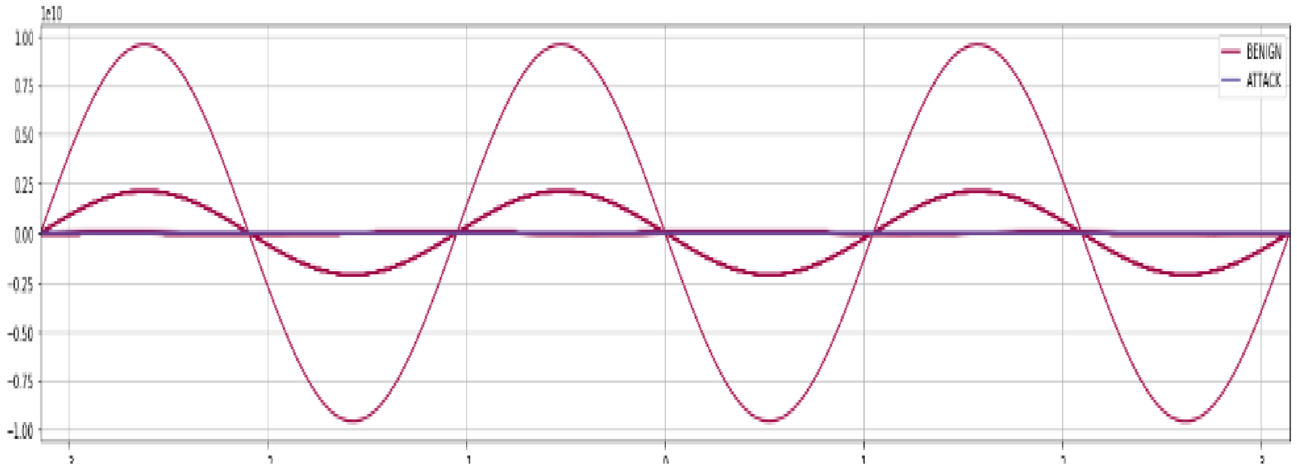

Figure 15: Andrew's curve for selected features of InSDN dataset

5.6.2 For all Numerical features

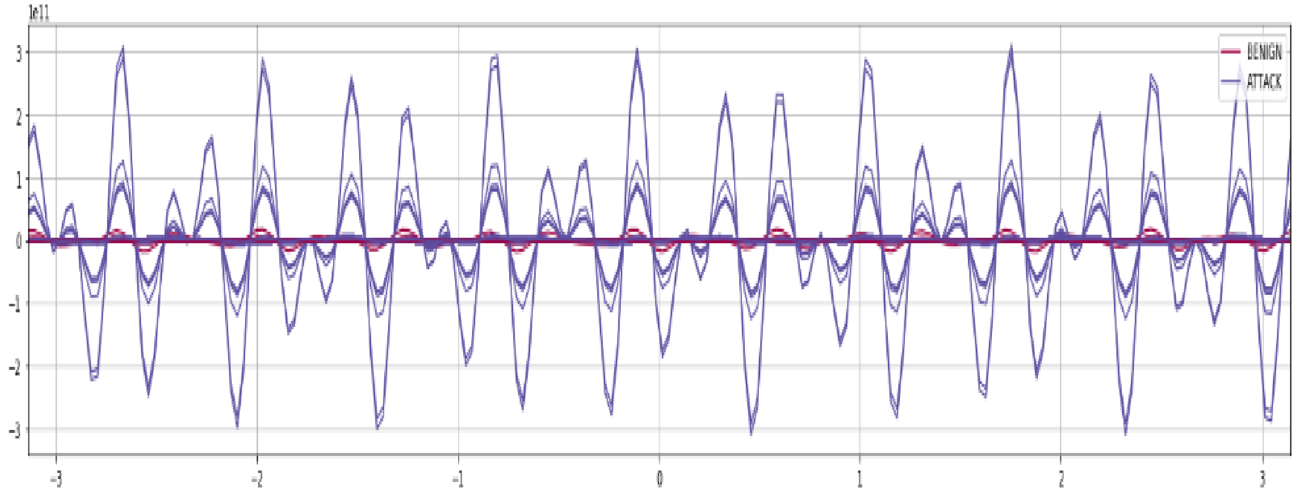

Figure 16: Andrew's curve for all numerical features of InSDN dataset

5.7 Receiver Operating Characteristics (ROC) for InSDN Dataset

The accuracy of the model can be assessed by using the Receiver Operating Characteristic (ROC) curve. It can also be used to represent the relationship between two classes i-e True and False classes. The relationship between false positive and true positive rates is represented by the ROC curve. The binary classifier's efficacy is measured by the area under the curve. When the Area Under Curve (AUC) is close to 1, the binary classifier provides perfect steps. The model, on the other hand, has the worst steps near the 0 during AUC.

5.7.1 ROC with RNN Autoencoder

The ROC of the InSDN dataset with RNN autoencoder is shown in Figure 17. Our model achieved an AUC of 0.984, indicating that it can effectively distinguish 98.4 percent of positive and negative rates.

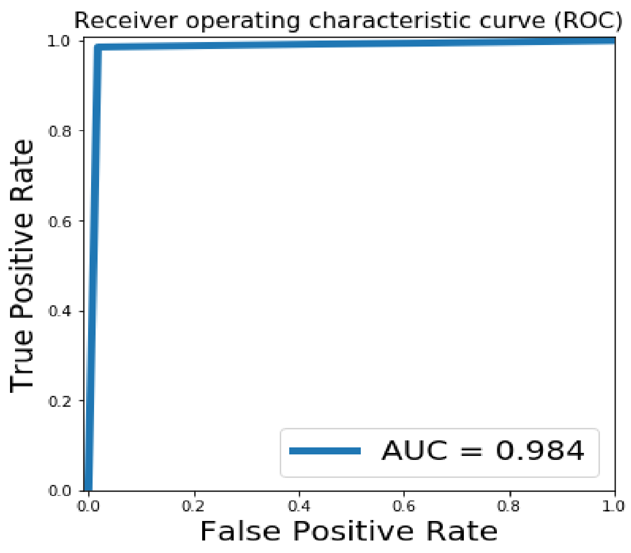

Figure 17: ROC for InSDN with RNN autoencoder

5.7.2 ROC with LSTM Autoencoder

The ROC of the InSDN dataset with the LSTM autoencoder is shown in Figure 18. Our model achieved an AUC of 0.982, indicating that it can effectively distinguish 98.2 percent of positive and negative rates.

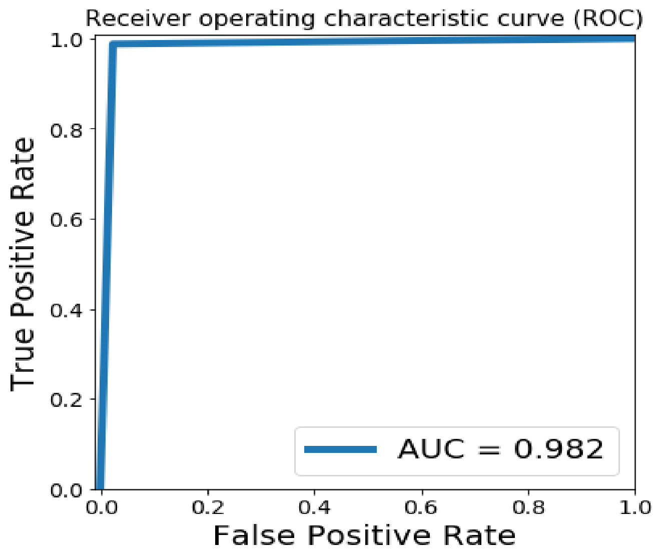

Figure 18: ROC with LSTM Autoencoder

5.8 ROC for CICDDoS2019 Dataset

5.8.1 ROC with RNN Autoencoder

The ROC of the CICDDoS2019 dataset with RNN autoencoder is shown in Figure 19. Our model achieved an AUC of 0.984, indicating that it can effectively distinguish 98.4 percent of positive and negative rates.

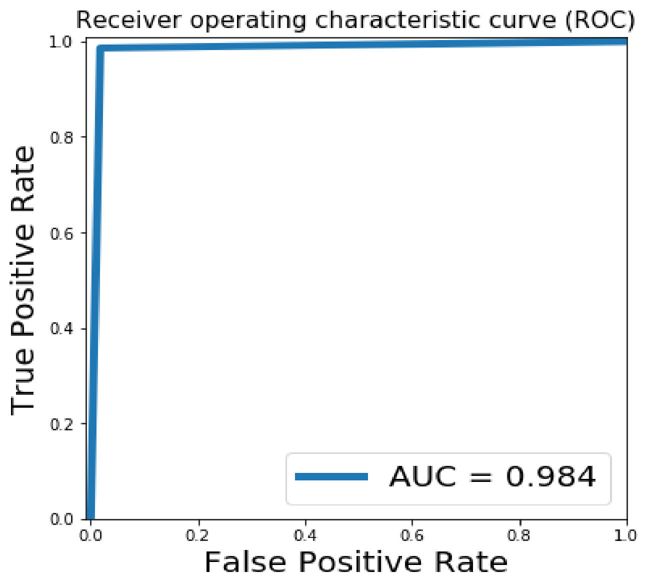

Figure 19: ROC with RNN Autoencoder

5.8.2 ROC with LSTM Autoencoder

The ROC of the CICDDoS2019 dataset with the LSTM autoencoder is shown in Figure 20. Our model achieved an AUC of 0.982, indicating that it can effectively distinguish 98.2 percent of positive and negative rates.

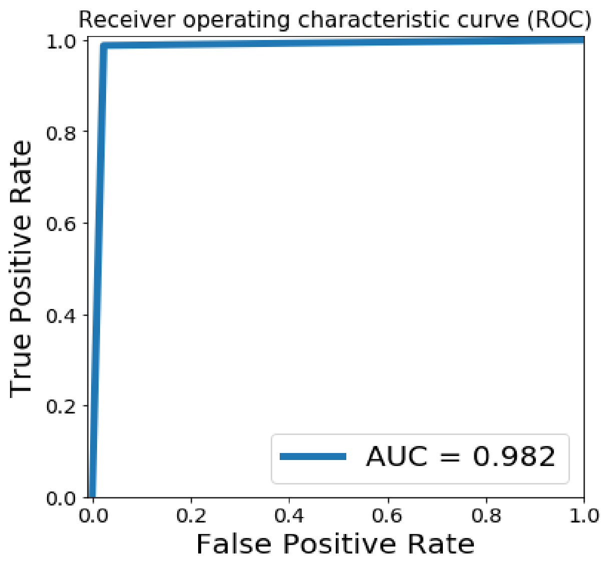

Figure 20: ROC with LSTM autoencoder

5.9 Discussion

For anomaly detection systems, this paper makes a unique contribution by illustrating the strength of Deep Learning. A unique DL technique on the basis of RNN-Autoencoder and LSTM-Autoencoder was proposed to distinguish input traffic from malicious/normal traffic. By removing functionality from input data automatically, the proposed DL model will minimize data dimensionality. DL strategies are promising for detecting network interference because of their ability to cope with a high degree of dynamic non-linear relationships. It can be used to overcome the limitations of conventional methods, which rely on domain information to identify anomalies in traffic. Our concept can be applied as part of the application layer of the SDN controller. However, as Internet traffic grows and the age of large data dawns, various new attack features emerge from previously established attacks, making it difficult to differentiate between them.

5.9.1 Experimental Setup for InSDN Dataset

5.9.1.1 Training RNN-Autoencoder and LSTM-Autoencoder

- Training, validation, and testing were the three subsets of the dataset. In both approaches, the training set is used to change the weight of the neural network. The validation set is used to fine-tune experiment parameters such as the proposed model's number of hidden layers (rather than weights). The test set is also used to assess the accuracy and performance of the model.
- In this thesis, we employed the train test split approach instead of the k-fold cross-validation approach to assess the model. A correlation plot is a visual representation of a vast amount of data that may be used to detect trends.
- In our situation, the observable pattern is that all variables that are significantly related with each other are color-coded. A strong positive relationship is represented by the color red, whereas a strong negative association is represented by the color blue. In exploratory factor analysis, correlation matrices are commonly employed as inputs.
- Cross-validation is less appropriate for these situations due to the underlying serial correlation of time series data [43]. The SoftMax layer takes the decoder output in this experiment and classifies the incoming data as regular or attack traffic. In all layers, we utilized categorical-cross entropy as a loss function, with an Adam optimizer, and the ReLU function for activation.
- Both models were trained with a batch size of 64 and the same number of epochs (5 each). The training and validation loss converge after 10 epochs and achieves its lowest value. The validation loss with the lowest model was picked. We conducted many experiments with different learning rates to get the best results.
- For this experiment, the selective learning rate is 0:001. Following that, we'll go through the implications of the learning rate in further depth.

5.9.1.2 Hyper-parameter Tuning

- The hyper-parameter values have a direct impact on the trained model's behavior, therefore picking the right ones is crucial to neural network design success.
- On the other side, choosing the optimum values for hyper-parameters is still reliant on best practice or human knowledge. We performed numerous experiments using various machine learning methods to get the optimal values of experiment hyper parameters. To assess the model's performance, we utilized a learning rate of 0:001 [44] because we know that learning

rate is a hyperparameter that regulates how much the model changes each time the model weights are changed in response to the predicted error.

- However, when the learning rate was adjusted to a lower value, both models performed better. The models exhibited an overall accuracy of up to 90%, as can be shown in the graph. As the number of hidden layers is increased, the model accuracy remains constant, but the training time increases substantially.
- As a result, our proposed structure has four layers. As a result, the four levels are more compelling in their ability to provide reasonable outcomes.

5.9.1.3 Data Partitioning

The performance of classification systems is influenced not only by the methodology employed, but also by the partitioning of training and testing data. Rasool et al. [45] looked at the effect of data partitioning techniques on the accuracy of the classifier. They agreed that gradually expanding the training data improved the performance of the classifier. When training is done with 80% of the input data, the best results are produced. After we prepared the input data, the final training data had 76 connected input characteristics. Furthermore, the sample distribution of the dataset is diversified and extremely large. We pick samples from each assault type to produce a balanced dataset in terms of different types of assaults. The training and validation set in our case have a total of 102572 and 25643 samples, respectively. We used attack records in the testing set that were not represented in the training phase to acquire a realistic detection rate. A total of 23000 samples are included in the testing set.

5.9.2 Experimental Setup for CICIDDoS2019 Dataset

5.9.2.1 Training RNN-Autoencoder and LSTM-Autoencoder

For CICIDoS2019, we used the same technique as for the InSDN Dataset, but we modified the features because we only wanted strongly correlated characteristics.

- Training, validation, and testing were the three subsets of the dataset. In both approaches, the training set is used to change the weight of the neural network. The validation set is used to fine-tune experiment parameters such as the proposed model's number of hidden layers (rather than weights). The test set is also used to assess the accuracy and performance of the model.
- Instead of using the k-fold cross-validation methodology, we employed the train test split technique to evaluate the model in this thesis.

- A correlation plot is used to summarize a huge amount of data with the purpose of seeing trends.
- In our situation, the observable pattern is that all variables that are significantly related with each other are color-coded. A strong positive relationship is represented by the color red, whereas a strong negative association is represented by the color blue. The use of correlation matrices as exploratory factor analysis inputs is common.
- Cross-validation is less suitable for these situations due to the time series data's intrinsic serial correlation [44]. The SoftMax layer in this experiment takes the decoder output and categorizes the input data as normal or attack traffic.
- We used categorical-cross entropy as a loss function in all of the layers, using Adam as the optimizer and ReLU as the activation function. With a batch size of 64 and the same number of epochs, both models were trained (5). The training and validation losses converge after 10 epochs and reach their lowest value. The validation loss with the lowest model was picked.
- To acquire the best results, we ran multiple tests with varying learning rates. For this experiment, the selective learning rate is 0:001. Following that, we'll go through the implications of the learning rate in further depth.

5.9.2.2 Hyper-parameter Tuning

- The behavior of the trained model is directly dependent on the hyper-parameter values, and choosing the appropriate values is critical to the success of neural network design. Choosing the best settings for hyper-parameters, on the other hand, is still based on best practice or human understanding.
- Using various machine learning approaches, we ran several experiments to identify the ideal values of experiment hyperparameters. We used a learning rate of 0:001 to evaluate the model's performance.
- However, both models performed better when the learning rate was set to a lower value. The models were found to have an overall accuracy of up to 96 percent. The model accuracy remains constant as the number of hidden layers is increased, but the training time grows significantly.
- As a result, our proposed structure has four layers. As a result, the four levels are more compelling in their ability to provide reasonable outcomes.

5.9.2.3 Data Partitioning

The performance of classification systems is determined not only by the technique used, but also by how training and testing data are partitioned. Rasool et al. [45] investigated the impact of data partitioning strategies on the classifier's accuracy. They agreed that increasing the training data gradually improves classifier performance. When 80 percent of the input data is used for training, the best results are produced. The final training data has ten strongly linked input features after we prepared the input data. Furthermore, the dataset's sample distribution is diverse and quite huge. To achieve a balanced dataset with respect to different sorts of attacks, we take samples from each attack type. The total number of samples for both the training and validation sets in our example is 102572 and 25643, respectively. We used attack records in the testing set that were not represented in the training phase to acquire a realistic detection rate. There are a total of 25000 samples in the testing set.

Chapter 06

CONCLUSIONS & RECOMMENDATIONS

6.1 Conclusion

SDN has the potential to provide novel network security solutions. The protection of SDN architecture, on the other hand, has received little attention. DDoS attacks against the SDN control plane may be extremely vulnerable in reactive SDN architectures.

In this article, we introduced a deep learning algorithm for detecting network interference and evaluated our IDS model. Through comparing the results of the flow-based anomaly detection system to those of other classifiers, we demonstrated the potential of using deep learning for the flow-based anomaly detection process. The deep learning approach has potential in the sense of the SDN environment. This is due to the controller's centralized design and the SDN's versatile structure. We suggested a deep learning algorithm that can effectively model regular traffic data using LSTM and RNN autoencoders. Our research demonstrates that the model we propose can detect anomalies in network traffic data. For the preparation and estimation of our suggested model, we used two datasets: InSDN and CICDDoS2019. The dataset includes the most recent and detailed DDoS attacks. In comparison to the current common classical DL techniques. DDoSNet has the leading assessment metrics in context of precision, memory, F-score, & consistency, according to our model's assessment.

Initially, the IDS was used for two-class grouping (normal and anomaly class). We tried to reduce loss while increasing accuracy when training a model. For more details, we looked at the model's precision, recall, and f-measure. The test data was used to assess the efficiency of the RNN and LSTM algorithms.

6.2 Recommendations

In the future, we would like to test the efficacy of our suggested model on different datasets and different features. We also recommend performing the experiment on different hyper parameters by utilizing different learning rates for the dataset. In this analysis, a binary classification scheme was used to separate the incoming traffic into legitimate and malicious types. It is, however, important to identify every attack type individually. We want to expand our research to include a multi-class classification system. We will also model the SDN network in numerous fields and attack traffic to provide a diverse dataset that adequately represents actual internet traffic.

References

[1] A. G. I. V. P. &. K. J. Khraisat, "Cybersecurity,," *Survey of intrusion detection systems: techniques, datasets and challenges.,* pp. 1-22, (2019).

[2] N.-A. L.-K. S. D. a. A. D. J. Mahmoud Said Elsayed, " Network Anomaly Detection Using LSTM Based Autoencoder. In The 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks," p. 16–20, November 2020.

[3] M. L. a. L. Toker., *An Efficient Flow-based Multi-level Hybrid Intrusion Detection System for Software-Defined Networks. ,* 2018.

[4] M. K. P. a. B. Perumal., " A meta-heuristic Bayesian network classification for intrusion detection.," *International Journal of Network Management ,* vol. 3 , no. (2019), p. 29, (2019).

[5] N.-A. L.-K. S. D. a. A. D. J. [. d. D. Mahmoud Said Elsayed, "A deep-learning model for detecting network attacks. In 21ST IEEE INTERNATIONAL SYMPOSIUM ON A WORLD OF WIRELESS, MOBILE AND MULTIMEDIA NETWORKS," *(IEEE WOWMOM),* jan 2019.

[6] N.-A. L.-K. a. A. D. J. Mahmoud Said Elsayed, " A Novel SDN Intrusion Dataset.," *IEEE Access,* no. 65263–165284., (2020).

[7] D. N. a. P. H. B. Karan, "“Detection of DDoS attacks in software defined networks,” in Proc. 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS)," 2018.

[8] R. H. O. S. B. B. J. L. F. H. C. A. K. E. B. a. M. Z. A. B.] S. S. Mohammed, " “A new machine learning-based collaborative ddos mitigation mechanism in software-defined networks,”," 2018 .

[9] *. M. 2. D. M. 2. S. A. R. Z. 2. M. G. 3. a. F. E. M. Tuan Anh Tang 1, "Deep Learning Approach for Intrusion Detection in Software Defined Networking," 2011.

[10] A. Javaid, Q. Niyaz, W. Sun and M. Alam, "A deep learning approach for network intrusion detection systems. In Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS).," 13-16 march 2016.

[11] C. Yin, Y. Zhu, J. Fei and X. He, " A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks.," vol. 5, p. 21954–21961., 2017.

[12] N. Shone, T. Ngoc, V. Phai and Q. Shi, "A deep learning approach to network intrusion detection.," 2018.

References

[13] R. Braga, E. Mota and Passito, "A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. In Proceedings of the 2010 IEEE 35th Local Computer Networks (LCN) Conference, Denver, CO, USA,," 10–14 October 2010.

[14] T. Phan, T. Van Toan, D. Van Tuyen, T. Huong and N. Thanh, "OpenFlowSIA: An optimized protection scheme for software-defined networks from flooding attacks. In Proceedings of the," USA, 2016.

[15] R. Kokila, S. Selvi and K. Govindarajan, "DDoS detection and analysis in an SDN-based environment using a support vector machine classifier. In Proceedings of the," in *IEEE Sixth International Conference on Advanced Computing (ICoAC),* , Chennai, India, 17–1, 2014.

[16] P. Winter, E. Hermann and M. Zeilinger, "Inductive intrusion detection in flow-based network data using one-class support vector machines. In Proceedings of the," in *IEEE 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS),*, 2011 .

[17] P. Van Trung, T. Huong, D. Van Tuyen, D. Duc, N. Thanh and A. Marshall, " A multi-criteria-based DDoS-attack prevention solution using software defined networking. In Proceedings of the," in *IEEE International Conference on Advanced Technologies* , 2015 .

[18] A. Sundaram, "An introduction to intrusion detection," vol. 2, pp. 3-7, 1996..

[19] D. A. B. a. V. M. Mane, "Comparative study and analysis of network intrusion detection tools,," *International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT,* pp. 312-315., 2015 .

[20] F. M. R. P. E. V. C. E. R. S. A. a. S. U. D. Kreutz, ""Software-defined networking: A comprehensive survey,"," vol. 103, pp. 14-76, 2014.

[21] T. N. N. V. D. P. a. Q. S. N. Shone, ""A deep learning approach to network intrusion detection," IEEE transactions on emerging topics in computational intelligence," vol. 2, pp. 41-50, 2018.

[22] R. H. O. S. B. B. J. L. F. H. C. A. K. E. B. a. M. Z. A. B. S. S. Mohammed, "A new machine learning-based collaborative ddos mitigation mechanism in software-defined networks," 2018.

[23] C. L. a. X. L. X. Yuan, " "DeepDefense: identifying DDoS attack via deep learning," in *IEEE International Conference on Smart Computing (SMARTCOMP)*, 2017.

[24] F. L. F. M. Z. T. H. Z. a. F. J. Y. Fu, ""An intelligent network attack detection method based on rnn,," *IEEE Third International Conference on Data Science in Cyberspace (DSC),* pp. 483-489., 2018.

[25] K. M. a. S. M. S. K. N. Mallikarjunan, "A survey of distributed denial of service attack," *10th International Conference on Intelligent Systems and Control (ISCO),* pp. 1-6, 2016.

References

[26] C. D. a. A. Mitrokotsa, ""DDoS attacks and defense mechanisms: classification and state-of-the-art,"," *Computer Networks,* vol. 44, pp. 643-666, 2004.

[27] F. R. Y. Q. G. a. J. L. Q. Yan, "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, 2017.

[28] C. L. a. K. R. T. Peng, ""Survey of network-based defense mechanisms countering the DoS and DDoS problems,"," *ACM Computing Surveys (CSUR),* vol. 39, 2007..

[29] D. Kshirsagar and S. Kumar, ""A feature reduction based reflected and exploited DDoS attacks detection system,," *Journal of Ambient Intelligence and Humanized Computing,,* pp. 1-13, 2021.

[30] L. M. D. M. S. A. R. Z. a. M. G. T. A. Tang, " "Deep learning approach for network intrusion detection in software defined networking,," in *international conference on wireless networks and mobile communications (WINCOM),,* 2016.

[31] R. R. C. a. S. P. Patil., ""Intrusion detection system: classification, techniques and datasets to implement," *nternational Research Journal of Engineering and Technology (IRJET),,* vol. 4, pp. 1860-1866, 2017.

[32] A. E. S. a. J.-M. F. A. Hijazi, ""A Deep Learning Approach for Intrusion Detection System in Industry Network," pp. 55-62, 2018.

[33] Y. S. Hussain, ""Network Intrusion Detection for Distributed Denial-of-Service (DDoS) Attacks using Machine Learning Classification Techniques,"," 2020..

[34] D. K. J. K. S. C. S. H. K. a. I. K. R. K. Malaiya, "An empirical evaluation of deep learning for network anomaly detection,"," *International Conference on Computing, Networking and Communications (ICNC), ,* pp. 893-898., 2018.

[35] D. L. S. a. J. Cairney, " "What's under the ROC? An introduction to receiver operating characteristics curves,"," *The Canadian Journal of Psychiatry ,* vol. 52, pp. 121-128., 2007.

[36] G. O. A. L. M. F. S. E. d. S. R. J. F. L. d. O. T. L. e. a. M. H. Lino Ferreira da Silva Barros, ""Benchmarking Machine Learning Models to Assist in the Prognosis of Tuberculosis,"," 2021.

[37] S. L. a. R. X. S. D. W. Hosmer Jr, " Applied logistic regression," vol. 393, 2013..

[38] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial,"," *Frontiers in neurorobotics,* vol. 7, p. 21, 2013.

[39] G. I. Webb., "Naïve Bayes,," *Encyclopedia of machine learning,* vol. 15, pp. 713-714, 2010.

References

[40] H. W. D. B. Y. B. a. K. G. G. Guo, "KNN model-based approach in classification,," *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems,* pp. 986-996, 2003.

[41] L. R. a. O. Maimon, "Decision trees," *Data mining and knowledge discovery handbook,* pp. 165-192, 2005.

[42] G. Biau and E. Scornet, "A random forest guided tour,"," vol. 25, pp. 197-227, 2016.

[43] D. Lee and J. Lee, "Domain described support vector classifier for multi-classification problems," *Pattern Recognition,* vol. 40, pp. 41-51, 2007..

[44] R. J. H. a. B. K. C. Bergmeir, "A note on the validity of cross-validation for evaluating autoregressive time series prediction," *Computational Statistics & Data Analysis,* vol. 120, p. 70–83, 2018.

[45] U. A. K. A. H. W. W. R. a. Z. A. R. U. Rasool, "Cyberpulse: A machine learning based link flooding attack mitigation system for software defined networks,”," *IEEE Access,* vol. 7, p. 34885–34899, 2019.

[46] P. Winter, E. Hermann and M. Zeilinger, "Inductive intrusion detection in flow-based network data using one-class support vector machines. In Proceedings of the 2011 IEEE 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS), P," 2011.