Improved Attacks Detection/Prevention Scheme for Secure Cloud Computing Infrastructure based on Software Defined Perimeter

Gharam Nuwaysir Alruwaili[†], Saloua Hendaoui*

Department of computer Science, College of Computer and Information Sciences, Jouf University, Jouf, Skaka, Saudi Arabia

Summary
Cloud computing is becoming a well-known buzzword in which business titans, such as Microsoft, Amazon, and Google, are at the forefront of developing and providing sophisticated cloud computing systems to their users in a cost-effective manner. Security is the biggest concern for cloud computing and is a major obstacle to users adopting cloud computing systems. Maintaining the security of cloud computing is crucial, especially for the infrastructure. Several research works have been conducted in the cloud infrastructure security area; However, some gaps have not been completely addressed, and new challenges continue to arise. In this paper, we study the challenge of protecting the cloud infrastructure from denial of service (DoS) attacks. We propose a detection and protection scheme based on deep learning and profiting from the novel software-defined perimeter technology.

Keywords:
Denial of Service (DoS), Software Defined Perimeter (SDP), machine learning, deep learning, network virtualization, classification

1. Introduction

Cloud computing is based on the notion of providing all feasible services, such as software, IT infrastructure, and services, to clients over the internet. Cloud computing systems are heterogeneous, large-scale groupings of autonomous computers with a flexible computational architecture. This technology is on the rise since it is seen as the best option for firms that don't want to deal with system maintenance and development in-house [1]. Many companies, including Amazon Web Services (AWS), Google, IBM, Sun Microsystems, Microsoft, and others, are creating effective cloud products and technologies. Customers and the enterprise share data via virtual data centers with cloud technology [2]. Cloud computing has become a widely accepted and ubiquitous paradigm for service-oriented computing, in which computer infrastructure and solutions are provided as a service. Because of its advantages (e.g., self-service on-demand, broad network access, resource pooling, and so on), the cloud has revolutionized the abstraction and use of computer infrastructure, making cloud computing popular [3]. However, security is the biggest challenge, and concerns regarding cloud computing continue to arise as we witness an increasing number of new developments in cloud computing platforms [4]. Several research activities have been presented to deal with security threats. Nonetheless, new approaches to making the cloud safer have yet to be discovered. The majority of existing cloud security approaches do not address the new sorts of security concerns that cloud computing infrastructure may encounter. As a result, they are unable to identify attacks or vulnerabilities that may originate from either the cloud service provider or the consumer. Furthermore, very few existing works have examined the different levels of cloud infrastructure altogether. This paper did an exhaustive assessment of the challenges that the cloud computing infrastructure encounters at various levels due to the relevance of examining such concerns (application, host, network, and data level). It also discusses the existing solutions that have been employed to address these concerns.

In this paper, the promising software defined perimeter will be adopted by executing the devices classification algorithm in the centralized controller. The aim of the classification model is to categorize devices requesting connection to the cloud server into three categories: attackers (reject their request), suspicious devices (control their behaviors), and trusted devices (grant their requests). This classification will protect the network resources from consumption by the DoS attackers.

The remaining of this paper is organized as follows: In section 2, we outline the main problems that raised this contribution. Section 3 gives a detailed related work. Section 5 presents the proposed solution. In section 6 we evaluate the proposed solution as well as discuss the obtained results. Section 7 concludes this work.

2. Problems statement

About network security, dealing with DDoS assaults in cloud computing has been a difficulty, especially with the

Manuscript received June 5, 2022
Manuscript revised June 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.6.63

introduction of SDN and cloud computing views. Cloud computing is vulnerable to DDoS assaults due to the massive amount of data kept in the cloud. The measure of information is rapidly evolving in the age of cloud computing and massive data [18]. The existing safety measures are insufficient to fulfill the distributed computing security needs. In distributed computing environments, Gartner expects application-layer DDoS assaults to expand three times per year, according to Gartner. DDoS assaults are expected to account for 25% of all application-layer attacks, according to the prediction. In the cloud computing setting, traditional protection components have a lot of trouble detecting DDoS attacks. Despite the growing study on recognizing and anticipating DDoS assaults, security breaches have been increasing at an alarming rate, both in projects and distributed computing systems. In this paper, we discuss the reasons behind the rise of DDoS assaults in cloud computing environments. On-demand self-service, broad network access, resource pooling, quick flexibility, and measurable service are among the essential aspects of cloud computing that we examine. In addition, we look at DDoS assaults and cloud computing discovery plots.

3. Related Work

Several contributions were developed with the aim of reducing the impact of DoS in cloud computing.

3.1. Defense mechanisms of network/transport layer

In this part, we first categorize and discuss each of the four groups of protection methods against DDoS attacks that are relevant at the network/transport layer. The offered techniques, which are based on UDP, TCP, and ICMP protocols, are then introduced and discussed in each group.

3.1.2. Source-based mechanisms

Abdelsayed et al. [6] presented Tabulated Online Packet Statistics (TOPS), a monitoring system that uses heuristic algorithms to analyze traffic to detect and filter bandwidth DoS assaults. A predefined set of compact tables is utilized in this method to monitor space and IP address domain, which is useful for identifying packet flow imbalance. notwithstanding the presence of a traditional firewall, which filters outgoing packets This strategy is efficient in reducing the consequences of DDoS attacks planned inside the network because it limits the transmitter engine's (attacker's) packet transmission speed.
Wang et al. [7] established a technique based on quantitative measurements to detect DDoS attacks, obtaining two proportion factors to compromised servers, which have a substantial effect on traffic feature deviation. A multi-phase detection approach consisting of Network Traffic State (NTS), fine-grained singularity detection, and a malicious address extraction engine has been developed to identify subtle DDoS anomalies in monitors near the assault source. The deviation rate of the network state in a monitoring point is determined using NTS prediction in the suggested method.

Table 1 outlines some of the most prominent source-based techniques covered thus far, as well as their benefits and drawbacks.

3.1.2. Network-based (core) mechanisms

Mechanism	Main idea	Type	Advantages	Disadvantages
Abdelsayed et al.	Using the compact table to detect flow imbalances and heuristic methods to filter bandwidth attacks	Detection and prevention	High precision Only a little amount of computing power is required.	Inefficiency when there is a lot of traffic.
Wang et al.	Predicting traffic status and extracting malicious addresses to detect an attack.	Detection	Detecting attacks with a low incidence of occurrence. Extracting a malicious address from which to respond.	Storage necessitates memory.

Table 1: **Popular source-based defense mechanisms and their characteristics**

Gil and Poletto [8] suggested a Multi-Level Tree for Online Packet Statistics (MULTOPS) data structure as well as a heuristic attack detection approach. The network's computers and routers use MULTOPS to collect packet rate information in this manner, and the heuristic algorithm detects the intended assault. As a heuristic method in attack detection, the MULTOPS method uses rates of the host transmitting/receiving disproportional packets.
Tao and Yu [9] suggested a DDoS attack detection approach in local networks that monitors network traffic using the flow entropy of local network routers and sends out an alarm if the flow entropy drops. Information distance has also been utilized to distinguish between DDoS attacks and Flash crowds. Based on information theory, the suggested technique is a two-phase procedure. In the first step, flow entropy in local network routers is evaluated, and if flow entropy drops significantly over time, an attack notice is sent out. Flows that cause a decline in router flow entropy are classified as suspicious flows in the following step.
Zargar and Joshi [10] suggested a collaborative distributed defense system that detects and responds to DDoS flooding assaults in the neighborhood of the attack source. There are four primary components to the suggested defensive mechanism. The task assignment server is the initial component, which is used to allocate tasks. The DiCoTraM mechanism, which is a traffic monitoring component that monitors traffic flow, is the second section. The DiCoDet mechanism is the third component, and it is a detection mechanism that distributes detection jobs across the autonomous system's routers. The final section is about the DiCoRes mechanism, which handles response actions in each autonomous system's router.
Table 2 summarizes some of the popular network-based (core) mechanisms and describes the details of each mechanism.

3.1.3. Destination-based (victim) mechanisms

Zhang et al. [11] developed an instant detection and prevention technique for DDoS assaults that are deployed in the router closest to the target and perform attack detection and prevention using IP-based traffic attributes analysis. The system takes periodic samples of each user's transmission and reception traffic and determines whether it is normal or not.
Wu et al. [12] suggested a decision tree-based DDoS assault detection system. In the proposed method's attack detection

• **Mechanism**	• **Main idea**	• **Type**	• **Advantages**	• **Disadvantages**
• **Gil and Poletto**	• Using the router's data structure to collect statistics and identify attacks.	• Detection and prevention	• In the event of a large-scale attack, securing legitimate traffic flow.	• Studying packet transmission only in IPV4. • There is a storage deficit. • Possible failures in the random address. •
• **Tao and Yu**	• Attack detection by routers’ flow entropy.	• Detection	• At the moment of the assault, an alert is being exported. • • Real-time simulation software.	• There have been some restrictive assumptions made.
• **Zargar et al**	• Based on monitoring responsibilities, traffic is being monitored in a coordinated manner.	• Detection and monitoring	• Supporting • detection mechanism. • • Overhead reduction. • • Covering attack flows. •	• Restrictive assumptions. • • Complexity of computation. • • Storage space. •

Table 2: Popular network-based defense mechanisms and their properties

phase, a basic, normal traffic profile is built, and network traffic that deviates from the basic traffic profile is identified as an attack.
Table 3 describes the main characteristics of popular destination-based (victim) mechanisms.

3.1.4. Hybrid (distributed) mechanisms

Mirkovic et al. [13] presented Defensive Cooperative Overlay Mesh (DefCOM), a distributed approach for distributing defensive nodes in the internet core. In this technique, all nodes form a peer-to-peer overlay for the secure transmission of attack-related messages. When an attack occurs, nodes close to the victim detect it and alert other DefCOM overlays.

Rahmani et al. [14] proposed a two-phase approach based on break detection in distributing the size of connections. It uses the total variation distance to evaluate the similarity between flows for DDoS attack detection, and generates alarms based on abnormal variations in the size of connections.

Table 4 describes some of the popular hybrid mechanisms and their advantages and disadvantages.

.3.2. Application layer defense mechanisms

The defense techniques offered to combat DDoS at the application layer are divided into two groups: destination-based (server side) and hybrid mechanisms. These

Mechanism	Main idea	Type	Advantages	Disadvantages
Zhang et al.	The analysis of IP-based traffic behavior is used to detect and prevent attacks.	Detection and Preventio n	Low computational overhead. Attack prevention. Detection of flooding attacks at the attack beginning stage.	Storage consumption to maintain information.
Wu et al	Attacker traceback is based on traffic pattern matching and is based on a decision tree.	Detection and Identificat ion	Secure communication channel between the components. Accuracy in attack detection. Attack detection with proper false positive and false negative rate. The ability of responding in short time intervals.	Storage overhead. It's possible that a mistake was made in distinguishing typical traffic from an assault. Discussion of complexity is lacking.

Table 3: Popular destination-based defense mechanisms and their properties

Mechanism	Main idea	Type	Advantages	Disadvantages
Mirkovic et al.	Using diverse points to collaborate in attack detection based on message exchange.	Detection and prevention	Defense by cooperation. Attack neutralization without classifier nodes.	Cooperation based on the sharing of messages. If no classifier nodes are present, the client will be harmed. The existence of rate limiter nodes is required.
Rahmani et al.	Total variation distance is used for attack discrimination and detection.	Detection	High accuracy and efficiency compared to methods based on entropy and volume. Decreasing false negative and false positive.	Some attacks may not be completely detected. Each packet's IP header must be accessed.

Table 4: popular hybrid defense mechanisms and their properties.

techniques given in each category are then explained and described.

3.2.1. Destination-based (server side) mechanisms

Jun et al. [15] suggested an entropy-based detection approach for ensuring regular traffic transmission and preventing anomalous traffic flooding. Entropy is employed in detection and is determined based on packet information over a specified time period.
Liu and Chang [16] have proposed a Defense scheme Against Tilt DDoS attacks called DAT, which detects if a person is suspicious or not based on the characteristic of monitoring each user.
Table 5 summarizes the main properties of popular destination-based (server side) mechanisms and describes the details of each mechanism.

3.2.2. Hybrid mechanisms

Yu et al. [17] presented a flow correlation coefficient as a similarity metric between suspicious flows in their detection system. The correlation coefficient is effective because the similarity between DDoS assault flows is significantly greater than that of the Flash crowd.
Table 6 summarizes the main properties of popular hybrid mechanisms.

4 Proposed Solution

The main contribution of this paper consists of the detection and prevention of DoS attacks using deep learning. The proposed algorithm will be executed by the software-defined network controller which allows full control of the network with ultra-reliability, low complexity, and low cost of deployment.

4.1 Software Defined Network (SDN)

The control plane and data plane are separated in the software-defined networking (SDN) paradigm, with an SDN controller receiving requests from linked switches and managing the operation of the switches under its control. To balance the load between SDN controllers, dynamic re-assignments between switches and their controllers are conducted. Most dynamic assignment systems employ a central element to collect information requests for switch reassignment in order to execute load balancing. When one super controller is utilized for all controllers and gets information from all switches, increasing the number of controllers poses a scalability difficulty. The distances

Mechanism	**Main idea**	**Type**	**Advantages**	**Disadvantages**
Jun et al	Attack detection by entropy.	Detection	Ensures normal traffic flow while screening questionable traffic.	Lack of comparison and accurate study of the proposed method with other detection methods, based on quality-of-service factors.
Liu and Chang	To guard against assaults, client properties and scheduling requests are used.	Response and Protection	Proper response time and accuracy in detection. Efficient service provided for legitimate users.	Dependence of defense on scheduling policies.

Table 5. Popular destination-based defense mechanisms and their properties

• **Mechanism**	• **Main idea**	• **Type**	• **Advantages**	• **Disadvantages**
• **Yu et al.**	• Using flow correlation coefficient to discriminate attack from Flash Crowd.	• Detection	• Using real data to survey the suggested strategy. • • Efficiency in the face of unforeseen threats. • • Efficiency versus current Botnet's size.	• Dependence of analysis on assumptions. • • Storage space to record information. • • Not surveying computational complexity.

Table 6: Popular hybrid defense mechanisms and their properties**.**

between controllers might be a limitation when allocating switches in a big network.

SDN, which provides centralized, decoupled, and programmable network switching methods, is gaining traction in academia and business. Traditional networking devices decide how an incoming packet should be handled based on its IP destination address, whereas SDN uses a flow-based forwarding method in which many header fields determine how the approaching bundle should be handled. SDN uses the concept of a centralized network control plane and adds programmability, which can simplify network administration and allow security policies to be organized in real-time. SDN can thus respond quickly to network irregularities and malicious traffic.

The logical separation between the control plane and the data plane, in the architecture and functional behavior of network nodes, is dissociated in a software-defined network (SDN) architecture [18], allowing for the centralization of all logic-related to control plane procedures in a so-called SDN Controller. As a result, network nodes can be reduced and streamlined for data plane performance. This design enables developers to create new algorithms that may be co-located at the SDN controller and used to govern and alter the network's functionality [19,20]. Even though only one controller can manage traffic for a small network [21], this is not feasible when dealing with a wide network at the internet size since each controller has limited processing capacity, thereafter large networks require several controllers to react to all requests [22]. The usage of numerous identical instantiations of a single controller, where each instantiation of the SDN controller accomplishes essentially the same work as the others, and each switch is coupled to one SDN controller, is one way to achieve this aim. Multiple controllers raised several key challenges, including where to locate them and how to link each control to a switch. These concerns should be evaluated not only during network deployment based on static data [23] but also frequently owing to the dynamic nature of the network [24].

4.2. Software Defined Perimeter (SDP)

In recent years there has been a boom in the development and spread of technology in our daily lives. The number of smartphones, wireless devices connected to mobile phone, social networks, and sensors in use has grown exponentially due to the advent of the concept of the Internet of Things (IoT). The rapid expansion and acceptance of new technologies, architectures, and models such as cloud computing, SDN, and NFV have created a new set of security and privacy concerns. Authentication, access control, data privacy, and data integrity are just a few of the challenges/concerns. SDP was introduced as a security model/framework to dynamically protect modern networks. This framework follows a need-to-know model where device identity is first verified and authenticated before accessing the application infrastructure [25].A promising solution is a software-defined perimeter (SDP), the concept proposed by the Cloud Security Alliance (CSA) as a security model/framework that has the potential to protect networks in a dynamic manner. This proposal was based on the Defense Information Systems Agency's (DISA) proposed Global Information Grid (GIG) Black Core network initiative. This architecture is based on the need-to-know principle, in which the device's/identity application's is validated and authenticated before access to the application infrastructure is permitted. Essentially, the infrastructure becomes “black,” meaning, it is undetectable by infrastructures unauthorized to see it. As a result, numerous network-based attacks such as server scanning, denial of service, password cracking, man-in-the-middle attacks, and others can be mitigated [25].

SDP is based on the idea of giving application/service owners the ability to add perimeter capabilities as needed to safeguard their servers. This is accomplished by substituting logical components for any physical appliances. These components are managed by the application/service owner(s) and act as a safety net. After a client's identity has been verified and authenticated, the SDP architecture grants access to the device. Multiple entities within the Department of Defense have implemented an architecture in which classified network servers are masked behind an access gateway.

The client must first authenticate to this gateway before gaining visibility and access to the server and its applications/services. The aim is to incorporate the logical model adopted in classified networks into the standard workflow. Hence, the SDP architecture leverages the benefits of the need-to-know model while simultaneously eliminating the need for a physical access gateway. The general concept is that the client’s devices/applications are first authenticated and authorized before creating encrypted connections in real time to the requested servers.

We exploit this feature by installing the classification model in the SDN controller which will exclude attacks from the network before connecting to the cloud server.

The SDP architecture is composed of and relies on five separate security layers:

Single Packet Authentication (SPA) - Device authentication is built on the foundation of SPA. This SPA is used by the SDP to reject traffic from untrusted devices. The initial packet is encrypted and forwarded from the client's device to the SDP controller, which verifies the device's authorization before granting access. The device then sends the SPA to the gateway again to assist it in determining the authorized device's traffic and rejecting all other traffic.

Mutual Transport Layer Security (mTLS) - Transport layer security (TLS) was originally designed to enable device authentication and confidential communication over the Internet. Even though the standard offers mutual device authentication, it has typically only been used to authenticate servers to clients. However, the SDP utilizes the full power of the TLS standard to enable mutual two-way cryptographic authentication.

Device Validation (DV) - since mTLS simply proves that the key hasn't expired or been revoked, device validation adds an extra degree of security by guaranteeing that the cryptographic key used is held by the correct device. It cannot, however, prove that it has not been stolen. As a result, DV confirms that the device belongs to a legitimate user and is running trusted software.

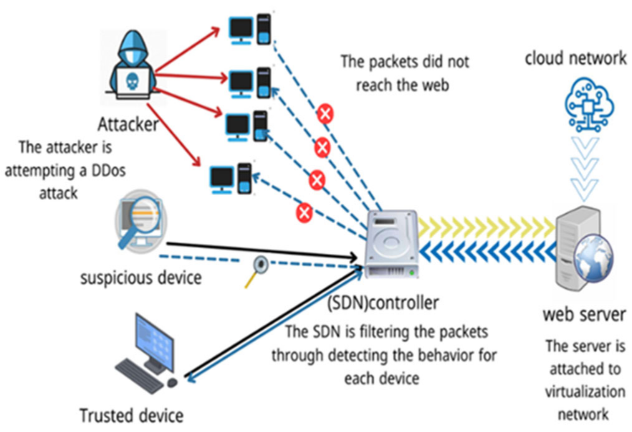
4.3. Proposed architecture

The SDP is made up of three basic components: client, gateway, and controller, as previously stated. The client gateway design is the SDP architecture developed and implemented for this work, and it can accurately characterize an internal company network scenario. In this situation, the gateway also serves as an application server, hosting the requested service. The three components' functions are outlined below.

1. Controller – SDP's major component is the controller. It stores information about authorized clients and servers, provides the gateway with information about rules, and manages the authentication of each component. For all of the aforementioned functions, the controller described in this paper makes use of a database. All of the hosts participating are listed in the database, which is subsequently forwarded to the gateway. It uses certificates to verify the authenticity of these hosts. The algorithm of DoS attack detection and prevention is executed by the SDP controller.

2. Gateway – The gateway enforces the rules which prevent any unauthorized access to the service hidden behind it. By default, the gateway blocks all traffic. However, once the controller provides the list of authorized initiating (clients) and accepting hosts (servers) and the list of services, it sets up rules which allow a connection to be established between the two while preventing all other traffic. This includes any attempt by the authorized hosts to establish a connection to a service they are not authorized to access. In this work, these rules are set up using the iptables.

3. Client – The client is the machine that is attempting to use a service. The client connects to the controller first in SDP and informs it of the service it wants to use. It then tries to connect to the service concealed behind the gateway once the verification is complete. The connection request will be accepted by the gateway, allowing the client-server data flow to proceed. Once a connection has been established, it should not be reset unless explicitly asked [25].



Attacker
The attacker is attempting a DDos attack
The packets did not reach the web
cloud network
suspicious device
(SDN)controller
The SDN is filtering the packets through detecting the behavior for each device
web server
The server is attached to virtualization network
Trusted device

Fig.1. Proposed DoS attacks detection/prevention architecture.

Using deep learning, a classification algorithm will be executed by the SDP controller. Whenever a device is categorized as an attacker, the SDP controller will block the device by adding it to the blacklist. Hence, as the proposed architecture presents, the attacker will not be directly connected to the cloud server. This will protect the network resources by rejecting all connection requests coming from devices classified as attackers. Additionally, a list will contain the suspicious devices for which the controller will control their behaviors. Trusted devices will be allowed to connect to the network, and send and receive data from the cloud server.

4.4 Detection/Prevention Model

Based on an existing dataset and machine learning approaches, we exploit a framework for DDoS attack classification and prediction in this study. The main steps in this framework are as follows [26].

- **First step**: Choose the dataset that will be used.
- **Second step**: Entails tool and language selection.
- **Third step**: Involves data pre-processing techniques to handle irrelevant data from the dataset. In the fourth step feature extraction and label.
- **Fourth step**: Encoding is performed to convert symbolical data into numerical data.
- **Fifth step**: The data partitioning is done in the model's train and test sets. We develop and train our proposed model in this step. To increase model efficiency, model optimization is performed on the trained model in terms of kernel scaling and kernel hyperparameter adjustment. We will generate output results from the model once it has been optimized.

The main contribution is to develop the best data usage model and model optimization, as well as the best model learning model. In this research work, we used two popular supervised learning models which are: (i) Random Forest Classifier; and (ii) the XGBoost workbook. Figure 2 shows the architecture diagram and data flow diagram of the proposed method.

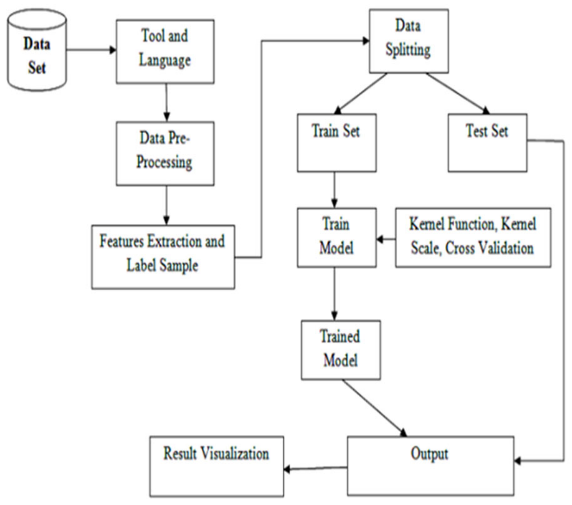


Fig.2. Data Flow Chart for the exploited machine learning based DDoD attack prediction technique [26].

5.1 Performance Evaluation

5.1 NSL-KDD Dataset Description

The dataset availability for intrusion detection is rare because most datasets cannot be shared due to various security and privacy concerns. The NSL-KDD dataset, on the other hand, allows free access to the whole dataset and was created to address the issues with the KDD99 dataset, which was created using data from DARPA'98 [25]. Even though KDD99 has been used in many research studies, there are several advantages when using the NSL-KDD dataset. The ML classifier will not be biased towards classes with frequent records due to the elimination of duplicate data.

The classification rates of various ML algorithms vary because the selected record count from each difficulty-level group is inversely proportional to the percentage of records in the KDD99 dataset, allowing the accuracy of multiple learning approaches to be adequately assessed. Furthermore, duplicate records in the test set were completely deleted, even though NSL-KDD is substantially smaller than KDD99, the number of records in the training sets is adequate to train an ML algorithm.

Table 7 shows the number of records in each data set, as well as the number of records associated with each attack type.

5.2 NSL-KDD Dataset for Training Model

Dataset	**Total**	**Normal**	**attack**
Train	125,973	67,343	58630
Test	22,544	9711	12833

Table 7. Details of normal and attack data

NSL-KDD is an improved version of the KDD, where the number of duplicated examples is significantly reduced. To improve the quality of classification models, duplicated records and null values were removed, reducing the complexity of the MDL models. Thus, the Training phase allows the generation of computer network traffic prediction data structures based on legitimate and four malicious classes.

5.3 Deep Neural Network

The artificial neural network (ANN) is a concept that was developed based on human brain biology. Neural networks (NN) can act as nonlinear discriminating functions because they can generate any decision boundary classification in feature space. The use of DNN in the domain of intrusion detection has become a prominent research focus in recent years, and it is an effective method that emerged from the shallow neural network paradigm (SNN). DNN excels at modeling and abstract representations, and it can simulate even the most complex models.

DNN has a huge potential for efficient data representation to build useful solutions. Facts and comparative analysis performed between ML algorithms, categorized under supervised and semi-supervised, and unsupervised learning led to the use of DNN for the proposed method. A DNN produces output based on weights applied to connections and related activation functions of neurons and is composed of many processing layers. The proposed

- Categorical data encoding: For categorical data encoding, one-hot encoding (OHE) was used because ML algorithms perform best when numerical values are used. Because the nominal categorical data were encoded using integer encoding, an ordered numerical list would be created, which would mislead the ML algorithms by assigning irrelevant importance to the values based on their magnitude, OHE was used instead. Because the OHE creates a new column for each category, it suffers from the "curse of dimensionality," and the categories with the lowest frequency were merged into a single category.

After the category reduction phase is completed, OHE is undertaken for categorical features using the 'OneHotEncoder' function in the Scikit-Learn library.

- Feature scaling: feature scaling concludes the data pre-processing, and it is employed to transform the numerical values of the complete dataset to a standard scale. Furthermore, the Standardization method is a scaling mechanism capable of rescaling the attributes to zero means and the distribution with unit standard deviation [29].

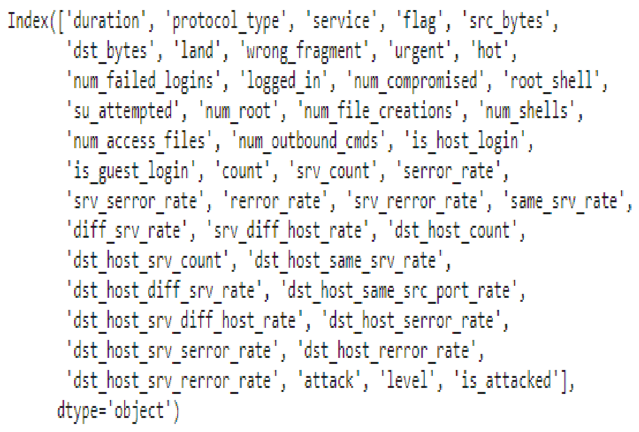
5.4 Machine Learning Pipeline

For real-time commercial applications, manual data transformation before training an ML algorithm is ineffective and impractical. ML pipelines can automate the data transformation and correlation into the model in the ML workflow. ML pipelines will improve the efficiency and simplicity of building ML models by eliminating redundant tasks in the workflow. The ML pipeline is a collection of five key tasks in the ML workflow: data ingestion, cleaning, pre-processing, model validation, and deployment. Because ML pipelines are not one-way and have iterative behavioral capabilities, they improve ML algorithm performance [28].

5.5 Data Pre-processing

Data pre-processing is the initial step that should be performed before feeding the data into the ML model. The tasks are feature selection, categorical data encoding, and feature scaling.

- Feature selection: The NSL-KDD dataset's 41 attributes are classified into three categories. They are basic, content, and traffic. The basic features can be deduced from the packet headers without inspecting the payload. To calculate traffic features, the time interval is used. However, domain knowledge is required to assess the packet's payload and derive content features [29]. Furthermore, the authors of the NSL-KDD dataset have not stated how the content features were derived from the packets. The attributes chosen for training the ML algorithm are listed in the figure below (fig 3).



```
Index(['duration', 'protocol_type', 'service', 'flag', 'src_bytes',
       'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot',
       'num_failed_logins', 'logged_in', 'num_compromised', 'root_shell',
       'su_attempted', 'num_root', 'num_file_creations', 'num_shells',
       'num_access_files', 'num_outbound_cmds', 'is_host_login',
       'is_guest_login', 'count', 'srv_count', 'serror_rate',
       'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate',
       'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count',
       'dst_host_srv_count', 'dst_host_same_srv_rate',
       'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
       'dst_host_srv_diff_host_rate', 'dst_host_serror_rate',
       'dst_host_srv_serror_rate', 'dst_host_rerror_rate',
       'dst_host_srv_rerror_rate', 'attack', 'level', 'is_attacked'],
      dtype='object')
```

Fig 3. Selected attributes for training the DNN

Categorical data encoding: For categorical data encoding, one-hot encoding (OHE) was used because ML algorithms perform best when numerical values are used. Because the nominal categorical data were encoded using integer encoding, an ordered numerical list would be created, which would mislead the ML algorithms by assigning irrelevant importance to the values based on their magnitude, OHE was used instead. Because the OHE creates a new column for each category, it suffers from the "curse of dimensionality," and the categories with the lowest frequency were merged into a single category.
After the category reduction phase is completed, OHE is undertaken for categorical features using the 'OneHotEncoder' function in the Scikit-Learn library.

- Feature scaling: feature scaling concludes the data pre-processing, and it is employed to transform the numerical values of the complete dataset to a standard scale. Furthermore, the Standardization method is a scaling mechanism capable of rescaling the attributes to zero means and the distribution with unit standard deviation [29].

5.6 Real-Time Feature Extraction

Network Configuration

The network configuration is the first step in implementing real-time feature extraction. To sniff the inbound and outbound data packets, a Linux workstation with two network interfaces was installed between the gateway router and the LAN inline to the data connection. Furthermore, the two network interfaces were virtually bridged using Linux machine configurations, and it will capture inbound and outbound data packets that flow through the Linux workstation using the packet sniffing mechanism developed in the C++ programming language.

Cost Analysis

As the number of data grows, the performance of DL algorithms improves. However, as the amount of data grows, most traditional machine learning methods perform worse. The advantage of performance improvement can be used for complex problems when using DL. However, outperforming many ML algorithms requires a large amount of data (>100,000). Furthermore, when compared to other traditional ML algorithms, the proposed DL

CPU	**Intel(R) Core(TM) i7 CPU @ 1.8 GHz 2.3 GHz**
Memory	8 GB
Platform	kaggle
Libraries	Keras (from tensorflow)
Language	Python

Table 8. Hardware specifications and utilized libraries

approach requires a significant amount of processing power, and the training time is significantly longer.
The hardware and software used to train the machine learning algorithms are shown in the table above (Table 8).

5.7 Simulations Results

The DNN model's performance was evaluated using the accuracy, loss, precision, recall, f1-score, and confusion matrix (CM) together with the curves illustrated below. The below Figs. 4, 5, 6, and 7 were obtained using the TensorBoard, the TensorFlow visualization toolkit to demonstrate the behavior of the accuracy, loss, precision, and recall of the training and cross-validation sets with the number of epochs

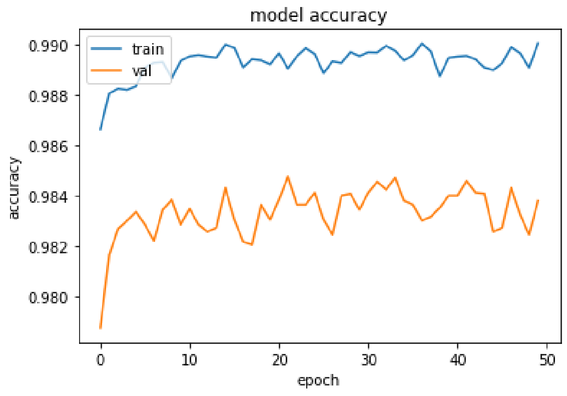


Fig. 4 Accuracy curves of training and validation set

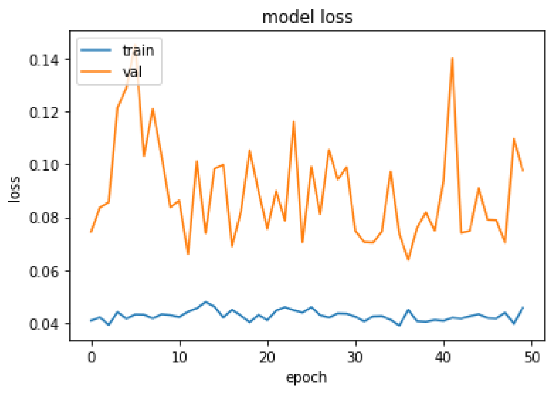


Fig. 5. Loss curves of training and validation set

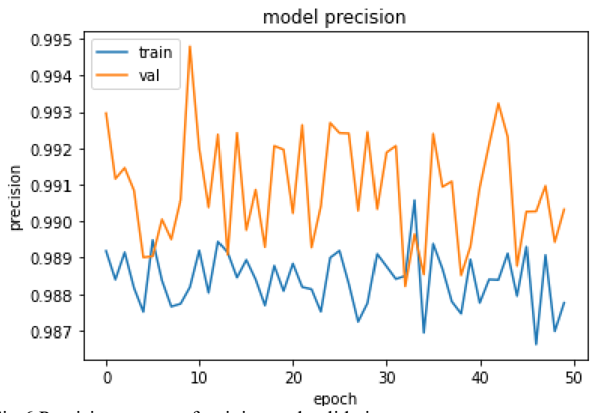
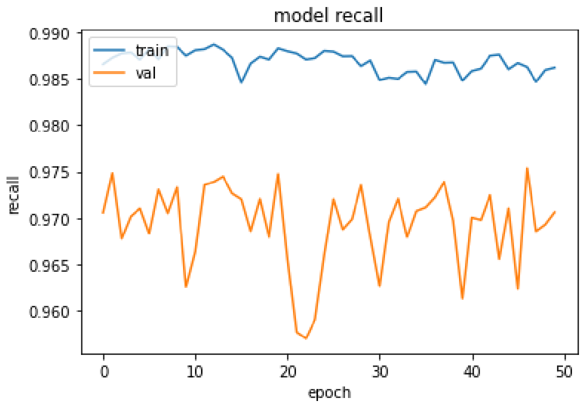


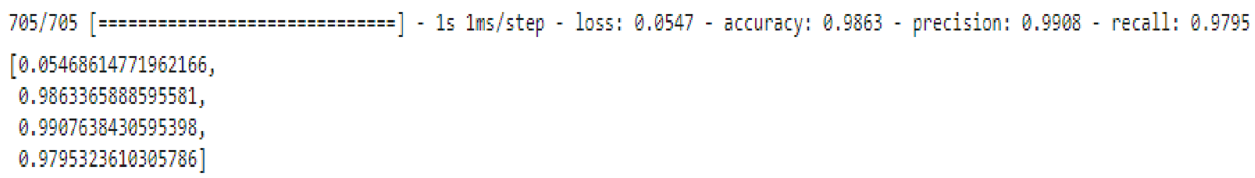
Fig.6 Precision curves of training and validation set



S
Fig. 7. Recall curves of training and validation set

Precision, recall, and F1-score depend on the number of predicted true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). These performance indicators are particularly effective when analyzing the performance when the class distribution is skewed.

- TP: predicts 1 and the actual class is 1.
- FP: predicts 1 and the actual class is 0.
- TN: predicts 0 and the actual class is 0.
- FN: predicts 0 and the actual class is 1.



```
705/705 [==============================] - 1s 1ms/step - loss: 0.0547 - accuracy: 0.9863 - precision: 0.9908 - recall: 0.9795
[0.05468614771962166,
 0.9863365888595581,
 0.9907638430595398,
 0.9795323610305786]
```

Fig. 8 **Training set performance results**

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{total samples}},$$

$$\text{Precision } (P) = \frac{\text{TP}}{\text{TP+FP}},$$

$$\text{Recall } (R) = \frac{\text{TP}}{\text{TP+FN}},$$

$$\text{F1 - score} = 2 \times \frac{P \times R}{P+R}.$$

Precisions determine the number of positive predictions that are truly positive. Moreover, recall calculates the proportion of positive predictions created employing the positive instances in the dataset. The F1-score is derived by computing the weighted average between precision and recall, and it may be used to seek a balance between precision and recall. The performance of the model is exceptional when the F1 score is greater. fig 8 shows the training and test results for accuracy, precision, recall, and f1-score obtained using the NSL-KDD test dataset .

Conclusion

Although the exploited data set still has some problems and may not be an ideal representative of current real networks, due to the lack of general data sets for network-based IDS systems, we believe that it can still be applied as an effective data set standard to help researchers compare methods of the detection of Various intrusion.

The prediction algorithm was executed by the SDP controller which will allow full control of the whole network with low cost and high reliability.

The rejection of attackers' devices is expected to save the cloud infrastructure thanks to the SDP technology that requires authentication before connection to the cloud server.

Acknowledgment

The authors would like to thank the Deanship of Graduate Studies at Jouf University for funding and supporting this research through the initiative of DGS, Graduate Students Research Support (GSR) at Jouf University, Saudi Arabia.

References

[1] K. Dubey and S.C. Sharma," An extended intelligent water drop approach for efficient VM allocation in secure cloud computing framework", Journal of King Saud University – Computer and Information Sciences, Nov.2020.
[2] J. Hanen, Z. Kechaou and M. B. Ayed, "An enhanced healthcare system in mobile cloud computing environment", 2016.
[3] Y. Alghofaili, A. Albattah, N. Alrajeh, M.A. Rassam and B.A.S Al-rimy," Secure Cloud Infrastructure: A Survey on Issues, Current Solutions, and Open Challenges", 2021.
[4] S.V. Hatwar and R. K. Chavan, "Cloud Computing Security Aspects, Vulnerabilities and Countermeasures", International Journal of Computer Applications (0975 – 8887), June.2015.
[5] T. Diaby and B. Bashari Rad, "Cloud Computing: A review of the Concepts and Deployment Models", I.J. Information Technology and Computer Science, 2017.

[6] S. Abdelsayed, D. Glimsholt, C. Leckie, S. Ryan and S. Shami, "An efficient filter for denial-of-service bandwidth attacks", 2003.
[7] F. Wang, H. Wang, X. Wang and J. Su, "A new multistage approach to detect subtle DDoS attacks", 2012.
[8] T.M. Gil and M. Poletto,"MULTOPS: A Data-Structure for Bandwidth Attack Detection", 2001.
[9] Y. Tao and S. Yu, "DDoS Attack Detection at Local Area Networks Using Information Theoretical Metrics" 2013.
[10] S. T. Zargar and J. Joshi,"DiCoDefense: Distributed Collaborative Defense against DDoS Flooding attacks", 2013.
[11] Y. Zhang, Q. Liu and G. Zhao, "A real-time DDoS attack detection and prevention system based on per-IP traffic behavioral analysis", 2010.
[12] Y. C. Wu, H. R. Tseng, W. Yang and R. H. Jan, "DDoS detection and traceback with decision tree and grey relational analysis", 2011.
[13] J. Mirkovic, M. Robinson, P. Reiher and G. Oikonomou, "Distributed Defense Against DDOS Attacks", 2005.
[14] H. Rahmani, N. Sahli and F. Kamoun,"DDoS flooding attack detection scheme based on F-divergence", 2012.
[15] J. H. Jun, H. Oh and S. H. Kim, "DDoS flooding attack detection through a step-by-step investigation", 2011.
[16] H. I. Liu and K. C. Chang, "Defending systems Against Tilt DDoS attacks", 2011.
[17] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang and F. Tang, "Discriminating DDoS Attacks from Flash Crowds Using Flow Correlation Coefficient", 2012.

[18] S. S- Hayward, G. O'Callaghan and S. Sezer, "SDN Security: A Survey", 2013.
[19] T. Hu, Z. Guo, P. Yi, T. Baker and J. Lan, "Multi-controller Based Software-Defined Networking: A Survey", 2018.
[20] Y. Zhanga, L. Cuiab, W. Wangc, Y. Zhanga, "A survey on software defined networking with multiple controllers", 2018.

[21] B. Heller, R. Sherwood and N. McKeown, "The Controller Placement Problem", 2012.
[22] J. Lu, Z. Zhang, T. Hu, P. Yi and J. Lan, "A Survey of Controller Placement Problem in Software-Defined Networking", 2019.
[23] G. Saadonab, Y. Haddada and N. Simonib, "A survey of application orchestration and OSS in next-generation network management", 2019.
[24] S. Auroux, M. Draxler, A. Morelli and V. Mancuso, "Dynamic Network Reconfiguration in Wireless DenseNets
[25] A. Moubayed, A. Refaey, and A. Shami, "Software-Defined Perimeter (SDP): State of the Art Secure Solution for Modern Networks", 2019.
[26] M. Tavallaee, E. Bagheri, W. Lu, A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set" 2009.
[27] S. P. Thirimanne, L. Jayawardana, L. Yasakethu, P. Liyanaarachchi, Chaminda Hewage, "Deep Neural Network Based Real-Time Intrusion Detection System" 2022.
[28] B. M. Serinellia, A. Collena, N. A. Nijdam, "Training Guidance with KDD Cup 1999 and NSL-KDD Data Sets of ANIDINR: Anomaly-Based Network Intrusion Detection System" 2020.
[29] Algorithmia. What an ML pipeline is and why it's important. 2020. https://www.datarobot.com/blog/what-a-machine-learning-pipeline-is-and-why-its-important/