Research Article

Efficient Intrusion Detection System for SDN Orchestrated Internet of Things

Esubalew M. Zeleke¹,¹ Henock M. Melaku,² and Fikreselam G. Mengistu³

¹Department of Electrical and Computer Engineering, Debre Berhan University, Debre Berhan, Ethiopia

²School of Electrical and Computer Engineering, Addis Ababa Institute of Technology, Addis Ababa, Ethiopia

³Faculty of Electrical and Computer Engineering, Bahir Dar Institute of Technology, Bahir Dar, Ethiopia

Correspondence should be addressed to Esubalew M. Zeleke; zesubalewmlat@gmail.com

Received 22 January 2021; Revised 7 October 2021; Accepted 26 October 2021; Published 29 November 2021

Academic Editor: Ihsan Ali

Copyright © 2021 Esubalew M. Zeleke et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Internet of Things (IoT) can simply be defined as an extension of the current Internet system. It extends the human to human interconnection and intercommunication scenario of the Internet by including things, to bring anytime, anywhere, and anything communication. A discipline in networking evolving in parallel with IoT is Software Defined Networking (SDN). It is an important technology that is aimed to solve the different problems existing in the traditional network systems. It provides a new convenient home to address the different challenges existing in different network-based systems including IoT. One important security challenge prevailing in such SDN-based IoT (SDIoT) systems is guarantying service availability. The ever-increasing denial of service (DoS) attacks are responsible for such service denials. A centralized signature-based intrusion detection system (IDS) is proposed and developed in this work. Random Forest (RF) classifier is used for training the model. A very popular and recent benchmark dataset, CICIDS2017, has been used for training and validating the machine learning (ML) models. An accuracy result of 99.968% has been achieved by using only 12 features on Wednesday's release of the dataset. This result is higher than the achieved accuracy results of related works considering the original CICIDS2017 dataset. A maximum cross-validated accuracy result of 99.713% has been achieved on the same release of the dataset. These developed models meet the basic requirement of a supervised IDS system developed for smart environments and can effectively be used in different IoT service scenarios.

1. Introduction

Internet of Things (IoT) is an extension of the Internet. People use the Internet to get different kinds of benefits. If human beings use the Internet to get several benefits in life, then why not objects are permitted to the communication world to add extra other benefits for human beings? IoT aims to answer this question by extending the scenario on the Internet, the interconnection, and intercommunication of people to people, by including things. Thus, in IoT, an interconnection and intercommunication of things among themselves and with people will be possible. By things, it is meant any physical object in the world, equipped with sensors and/or actuators, communication capability, and processing units. Things can also be virtual objects like objects in object-oriented programming, processes,

database, and other related entities found in the computer science world [1].

IoT is envisioned to make the life of human beings smarter and better than have been ever before. Even if it is in its infancy, IoT has already seen some initial deployments in different sectors. Health care, mobile asset tracking, intelligent fleet management, smart grid, environmental hazard detection and protection, home automation, smart agriculture, and smart industrial services are only some of the applications expected to be acquired from the provision of IoT systems [2,3]. The potential applicability of IoT is immense, and it is difficult to anticipate what it can bring in the future.

An equally evolving discipline in networking is Software Defined Networking (SDN). It is an intelligent networking paradigm which is characterized by its focus on the

separation of the control plane and the data plane. The underlying network will then be made programmable using the software. In the traditional networking paradigm, routing, network management, and other network-related decisions are undertaken by routers and switches which at the same time are responsible for forwarding data to the intended interface. But such network configurations impose problems on the scalability, network management, flexibility, interoperability, and other issues on the underlying network system [4,5]. Thus, the traditional network paradigm hinders proper service functioning and guarantying of various Quality of Service (QoS) requirements.

SDN is a promising paradigm in solving the aforementioned problems in which the traditional networking paradigm can hardly do. It is found promising for different technologies like cloud computing, data center, and future technologies (Information-Centric Network (ICN), 5G, IoT, and others) [4]. It is highly catching the eye of industries, academia, and governmental organizations for the enormous advantages it can bring to different systems. Cox et al. in [4] added that SDN has already been deployed in over 100 famous companies starting from its infancy including Google, China Mobile, AT&T, T-Mobil, and Telefonica. The general consensus is that SDN is the heart of future networks.

The architecture of the SDN system is composed of three layers. This three-layered architecture is shown in Figure 1. The bottom layer is the data plane layer. OpenFlow (OF) switches are the main components here in the data plane. The switches forward data as per the instruction received from the control unit.

The second layer is the control plane. It is the intermediate layer of the SDN system. In this layer, the SDN controller is found, which is also known as the “the network operating system”. The controller is responsible to formulate various control rules and pass these flow rules to the underlying switches in the data plane. The controller may request flow statistics information from the switches or in the other direction switches might request the controller for a flow control decision to be made and may send statistical information when certain conditions prevail. This inter-communication between the two layers is accomplished via certain protocols or application program interfaces (APIs). The OF protocol is the dominant protocol utilized for data plane and control plane communications.

On top of the control plane layer is the application layer. The application layer is responsible to render services or customer requirements. These applications are communicated with the controller using some kind of North-bound interfaces like the RESTfull API. These applications are just normal software programs which can be developed using any of the languages supported by the controller.

Though the SDN system is assumed to be a centralized system, such a centralized approach is practically less feasible for large-scale networks as such a system lacks scalability and is prone to a single point of failure. As far as large-scale networks are meant to run and render a quality and secure service, the SDN network on hand shall be kept only logically centralized and shall use physically distributed controllers.

FIGURE 1: Architecture of SDN system.

One challenge associated with SDN systems is to arrange the controllers in a distributed SDN network to function in the most efficient way. Various approaches are used to cluster distributed controllers. Even if several clustering schemes have been developed, this issue shall remain a focus of future works as far as SDN systems are meant to function as per the envision set.

Several load balancing techniques have been devised in SDN systems aiming to lessen the reduction of efficiency and other network performance degradation problems that exist due to poor traffic distribution. Though the load balancing solution is effective in an SDN environment, the flexibility in customization remains a challenge and an open issue since a single solution cannot fit the interests of different service providers.

Among the enormous sets that SDN is aimed to benefit, IoT is among the leading ones. A significant role is expected to be rendered by SDN in different aspects of IoT systems, and it is one of the prominent technologies, along with network virtualization, network function virtualization (NFV), and others, believed to be key enablers of IoT. Some ambiguity or uncertainty issues arise on the kind and level of contribution SDN provides to different technologies including IoT. Yet, in general, the interest of companies and organizations for SDN lies in earning one or more of the following advantages [4]: flexibility, interoperability, easier network configuration and management, scalability, reduction of CAPEX and OPEX costs, quality of services, and security.

The traditional network paradigm is not capable of addressing the problems incurred in IoT implementation. This is mainly due to the heterogeneous, constrained and highly scalable nature of the IoT systems. All these and other factors make the realization of IoT systems much difficult. The dynamic and agile nature of SDN, however, can address the aforementioned challenges in IoT paving the way for more secure, highly scalable, and interoperable IoT systems. SDN is considered revolutionary network technology, and it is supporting heterogeneous networking with rapid

evolution and dynamism using programmable planes. Tayyaba et al. in [6] stated that the SDN and IoT integration can meet the expectations of control and management in diverse scenarios.

1.1. Motivations. The human being has benefited and has also lost from the use of different computer technologies. There have often been intense arguments between the supporters and the opponents on the adoption of various technologies. Issues like loss of memory, depression, radio-frequency radiations, and other health problems are often mentioned to be caused or intensified by the use of digital computing and communication devices. These issues can exist in IoT in a more complex fashion if several issues are not handled well.

It is up to future researchers including the researcher of this work to have a firm understanding of the impacts of technology at first and then determine, in a better way, to bring the best of technology. In addition, leaving out the possible negative impacts, IoT is really an awesome technology to be part of. This is the main aspiration of the researcher behind his focus on IoT systems. The focus on security work mainly accounts for the inclination of the researcher to the cybersecurity competitions and the seriousness of the issue of security in IoT and other network-based systems.

1.2. Statement of the Problem. Even if IoT is envisioned to provide immense opportunities for human beings, its vision will not come with ease and comfort. Rather, a milliard of challenges has to be addressed and lots of vulnerabilities shall be filled up. One prime challenge that faces these systems is the issue of security. Several security works are out there to defend these systems against different security breaches. Yet, the issue of security is at its infancy and its lowest levels for these systems. As it is stated in [7], recent security attacks have revealed the ubiquitousness of security loopholes in IoT.

Moreover, even if SDN is hoped for providing a conducive environment for novel security works in IoT, there is still a lot to do since SDN cannot address the security issues by itself. Moreover, there are extra additional unique vulnerabilities imposed by SDN itself. Accounting to the extended vulnerability space in SDN-based IoT (SDIoT) systems, enormous types of attacks are launched toward them. One class of such attacks targets the availability of network systems and devices. Availability is mentioned to be one of the three main requirements of IoT alongside confidentiality and integrity [8]. Denial of Service (DoS) attacks are responsible for such denial of normal service availability.

DoS attacks, which are a collection of attacks that target denial of the normal functioning of network devices or systems, are ever-increasing. As it is put in [9], these attacks are widely used cyberattacks. The rise is mainly accounted for the enormous integration of poorly secured IoT devices to the Internet, which in turn are recruited and used for botnet attacks. The vulnerabilities found within devices,

communication links, and communication protocols can be used by these attacks to achieve their goals.

The motive behind the launch of these DoS attacks varies a lot. And based on these motives, the consequences of successful DoS attacks also vary. They might end up causing only minor inconveniences to users. But, in many circumstances, they are very much costly and have devastating effects. This catastrophe is expected to further escalate in IoT systems. This is due to the fact that many IoT systems are deployed in environments like health systems and vehicular traffic services, where minimal service unavailability might result in disastrous consequences which can include loss of money and even human life [7, 8]. Mendez et al. in [10] stated that DoS attacks are listed as one of the key challenges to be addressed in the IoT. The stealth nature of DoS attacks, the significant loss it incurs with a minor attack operation, the very constrained environment provided by IoT systems, and additional vulnerable space granted by SDN systems, among others, make SDIoT systems a conducive area for DoS attacks. IoT systems must be defended against these attacks as far as they are meant to meet their objectives.

This work aimed to develop a supervised Intrusion Detection (IDS) System. Several techniques have been used in the development of an IDS system. One technique used in IDS development is the use of machine learning (ML). Different datasets are out there for use for ML-based IDS systems.

Considering the insufficiency of the legacy dataset for developing IDS systems, CIC has developed new datasets that fill this gap. One of the datasets developed by the institute, CICIDS2017, has attracted many recent security researchers [11]. This research work focuses on this dataset for its attractive characteristics. The dataset expresses a more realistic network scenario, which includes normal traffic mixed with high-volume and low-volume malicious traffic with sneaky behavior, such as slow application layer attacks [12]. This dataset contains relatively new attack types [13]. Several IDS works are implemented using this dataset.

However, even if works that use the CICIDS2017 dataset exist, many of them are not specially designed to provide effective and efficient IDS systems against DoS attacks. Most of the research works conducted using this dataset are driven by other motivations which can be providing online detection [14], enhancing the performance of some ML classifiers [15–17], checking the efficiency of some feature selection algorithms, to provide anomaly detection [18], or meeting other objectives [11, 13]. Some of them focus on providing an IDS system to detect all the attacks found in the datasets. Yet, the efficiency and effectiveness of these systems on detecting DoS attacks are lower and need enhancement.

The main problem that necessitates the undertaking of this research work is the less focus of researches on using the dataset for security purposes. And more importantly, it is the less focus of researches on enhancing the efficiency and effectiveness of IDS systems in detecting DoS attacks that make the researchers undertake this work. In this work, an IDS model that has a high detection accuracy and a lower number of features has been developed.

1.3. Contributions of This Work. The following contributions are provided by this research work:

it provides an efficient IDS system with a high detection performance particularly suited to defend the tremendously increasing DoS attacks.

the work provides an efficient system in terms of memory use and processing time which is essential in real-time IoT application scenarios where real-time processing is badly needed.

unlike many other related works, this work provides in-depth parameter tuning and feature selection works and shows how effective performance improvement can be provided using parameter tuning and feature selection.

This research paper is organized as follows. Section 2 presents a review of related works. Section 3 presents the required resources and procedures used in the development of the detection system. Section 4 presented the results acquired from the experiments and their corresponding discussions. Finally, the conclusion of this research work and tips pointing some research gaps related to this work are presented in Section 5.

2. Related Works

Several security works are out there to defend systems from such attacks. The broad set of DoS defense mechanisms can be categorized as protection measures and mitigation measures [19]. The protection measures provide a first-line defense and aim to protect systems from attacks altogether. Like many other engineering protection works, it is difficult and costly to provide successful protection mechanisms. That is why mitigation mechanisms are out there with the aim of lessening the effects of the attacks after their successful launch. IDS systems are one kind of mitigation solutions that are used to detect the presence of an intrusion and if any, to identify the type of attack on hand.

Several techniques can be used for implementing detection systems. ML, statistical approaches, or other knowledge-based techniques were utilized throughout the literature to implement such systems [20]. The statistical approaches are based on statistical measures of various packet and flow parameters. The mean, median, mode, and standard deviation measures can be used for such a purpose [20]. A univariate approach which focuses on a single feature, a multivariate approach which focuses on statistical measures of a combination of features, or a time-series approach which makes observations in a given time interval can be used.

ML-based works are being widely used in IDS system development [16]. Both supervised and unsupervised approaches can be used. One ML-based work is found in [21]. In this work, a two-stage Artificial Intelligence- (AI-) based IDS system empowered by the global view of SDN technology was proposed and implemented targeting IoT networks. The selection of important features was carried out by leveraging the Bat algorithm with Swarm division and

differential mutation algorithms. The work has achieved a higher convergence over other Swarm intelligence algorithms. In addition, the work has got a higher detection performance in different attack classes reaching a DR performance of 100% for DoS attacks. However, it is based on the KDD'99 dataset which is outdated and expired for use as an evaluation dataset for IDS works [12]. In addition, testing did not consider cross-validation (CV) and other related principles that are significant in avoiding overfitting.

Several researchers have used different datasets for building their IDS system models. KDD'99 and its revised version NSL-KDD, CAIDA, DARPA, LBNL, ICSI, MNIST, CIFAR-10, and other datasets have been used throughout the literature for DoS experimentation [21,22]. Yet, these datasets are outdated and expired for use for IDS [12,22]. And, even if high detection results are achieved using these datasets, the detection solutions are ineffective when deployed for real cases. Consistent performance evaluation challenges are encountered by IDS systems caused by these unreliable datasets. This unreliability of the datasets emanates from the fact that the datasets do not represent the current actual attack traffic behavior. The developed works that used these datasets have achieved high detection results reaching a DR of 100% in some circumstances [21]. However, even if the results achieved are very high, the solutions are ineffective when deployed for real cases since most of the datasets mentioned above are outdated and unreliable for use [12].

One very important work which used the CICIDS2017 dataset was provided by [12]. The work designed and implemented an online DoS/DDoS attack detection using RF, AdaBoost, DTree, stochastic gradient, and other ML algorithms. The detection work used the three well-known datasets CICIDS2017, CIC-DoS, and CSE-CIC-IDS2018 and also had prepared its own customized dataset. It achieved results reaching up to 99.93% accuracy for a customized dataset. Its Detection Ratio (DR), and precision values for the CICIDS2017 dataset are 80% and 99.2%, respectively. The work also had succeeded in reducing the number of features to 28 with a reasonable accuracy using RF and then succeeded in reducing the features further to 20 features using its own new algorithm for feature selection. Several additional experiments were carried out in this work to calibrate and evaluate the system by adjusting the sampling rate, minimum flow table length, and maximum flow table length parameters. However, even if a very high result was achieved and CV was considered, several techniques could have been leveraged and experiments are performed to improve the accuracy even more. In addition, for IoT-related works, it is very important to decrease the number of features to the smallest possible. A lower number of features than the achieved number of 20 could be selected with reasonable accuracy.

Ahmin et al. in [23] proposed and implemented a hierarchical IDS system combining DTree and various rule-based algorithms using the CICIDS2017 dataset. It used the entire CICIDS2017 dataset and had collected DR values for five attack types, namely, DoS Hulk 96.782%, DoS Slowloris with 97.758%, DoS Slowhttptest with 93.841%, DoS GoldenEye with 67.571%, and Heartbleed 100%. Even if it had used a new hierarchical approach, its DR was not that high.

A new detection model based on the LeNet-5 and LSTM neural network algorithms using the CICIDS2017 and the CTU dataset was proposed and implemented in [13]. The LeNet algorithm is widely used in image processing, and it was used in this work to extract spatial features while LSTM is widely used in sound processing and was used in this work to extract temporal features. The work has applied these algorithms for network detection systems and has succeeded in having estimators having an accuracy of 99.91%. The authors in [17,18,24–28] have also developed an IDS system using this dataset.

Conventional IDS systems might be functional in IoT environments. More specifically, in ML-based IDS systems, similar techniques and datasets can be used for the development of IDS systems for IoT [22]. However, conventional systems are not suited for such smart environments. Somewhat strict requirements are expected of conventional IDS systems to be fully suitable for use in IoT. Elrawy et al. in [8] have emphasized that higher detection accuracy, low false-positive rate, low energy consumption, fast processing, and low-performance overheads are badly desired to be satisfied by IDS systems proposed for these smart environments.

Accounting for the advantages earned from SDN, several detection works were carried out to defend SDIoT systems. Yet, security works considering the integration of SDN and IoT seem missing in many cases. Kalkan et al. in [29] stated that only a few works have leveraged SDN for strengthening IoT security. Kalkan et al. in [29] added that providing security is an important priority for the heterogeneous SDIoT environment.

One work focusing on SDN systems was discussed under [30]. In this work, a lightweight flow-based IDS system was developed which resulted in a high DR of 98% and relatively low False Alarm Rate (FAR). A periodic collection of statistical information about the flows was made using SDN switches. Afterward, traffic classification was made using feature extraction and aggregation techniques. The work prepared and used its own dataset using the advantage of flow statistics collection by the OpenFlow (OF) switches. However, a very simple network was used to collect attack traffic which might result in certain deviations from the ground real traffic flow.

Moreover, although many research works were implemented using ML algorithms, most of them used outdated and unreliable datasets. This work focuses on filling some of the existing gaps, more specifically on parameter tuning, dimensionality reduction, and cross-validation using an up-to-date and popular dataset, CICIDS2017.

3. System Development

In order to develop the proposed IDS solution, several resources have been used and several procedures have been followed. The description of the dataset, the software resources used, the development steps followed, the proposed IDS architecture, and the evaluation technique and metrics used in the work are presented in this section.

3.1. Dataset Selection. Eleven metrics have been set for evaluating the validity of datasets. These measures are complete network configuration, complete traffic, labeled dataset, complete interaction, complete capture, available protocols, anonymity, attack diversity, heterogeneity, feature set, and availability of metadata. The CICIDS2017 meets all of them [31]. In addition to its conformity to the evaluation standards, the CICIDS2017 dataset is believed to have the characteristics expected of real-time network traffic [16] matching the current network traffic scenario.

Twelve attacks of various types are supported within the dataset. DoS attack traffic collection has been mainly carried out on Wednesday's traffic capture session. Four DoS attack families, namely, DoS Hulk, DoS Slowhttptest, Slowloris, and DoS GoldenEye attacks, have been supported in this session. This session has also a traffic capture for the Heartbleed attack. Additional DDoS attacks are collected on the traffic capture of Friday's release. Table 1 shows the class distribution of DoS/DDoS attacks prevailing in the CICIDS2017 dataset. Underneath is presented a brief description of these attacks.

DoS Hulk: HULK is short for HTTP Unbearable Load King. It is a DoS attack that targets web servers and achieves its objectives by flooding the servers with uniquely designed HTTP requests. A single attacker can launch this attack to disrupt a less secured web server entirely in just a couple of minutes. DoS Golden Eye: it is an application-level DoS attack tool used to bring websites down. Socket smashing is done until all the available sockets are consumed [32].

DoS Slowloris: it is a simple yet very effective low-volume DoS attack. It opens and maintains multiple open connections to eventually break normal connections to the target.

DoS Slowhttptest: it is a DoS attack tool used to generate low bandwidth application-level DoS attacks. It achieves its goal by using partial connections and slow requests. Slowloris, Slow HTTP Post, and Slow Read attacks are supported by this tool.

Heartbleed: it is an attack that uses the advantage of the vulnerability found in the implementation of the OpenSSL library while implementing the Heartbeat protocol. Though the Heartbleed attack is not a DoS attack; it is a vulnerability that can be comprised to render DoS attacks in the future [33,34]. Accounting for this issue, detection of this attack has been considered in this work. Moreover, Filho et al. in [12] stated that this attack can be assumed to be a DDoS. It has been considered and labeled as a DoS attack in both [12,23].

DDoS Loit: it is another DDoS attack used in the dataset. Two customized datasets are prepared from the dataset. These customized datasets are labeled CICIDS2017-Wed1 and CICIDS2017-Wed2 datasets.

The CICIDS2017-Wed1 contains the entire Wednesday release of the CICIDS2017 dataset. Even if the CICIDS2017-Wed1 customized dataset has considered detection of Heartbleed attack, no known cases which make the Heartbeat vulnerability, upon which the Heartbleed attack depends on and is a source for DoS attacks, have been found yet. The CICIDS2017-Wed2 customized dataset contains the

TABLE 1: Total number of DoS/DDoS attacks in CICIDS2017 dataset.

No	Type	Total number
1	Benign	440031
2	DoS Golden Eye	10293
3	DoS Hulk	231073
4	DoS Slowhttptest	5499
5	DoS Slowloris	5796
6	Heartbleed	11
7	DDoS Loit	128027

traffic samples of Wednesday's session without the samples containing Heartbleed attack. On top of this part of Wednesday's release, the DDoS Loit attack release from Friday's traffic capture is added to make the CICIDS2017-Wed2 customized dataset. Hereafter, whenever the term DoS is used, it refers to the five attacks collected in Wednesday's release, and whenever a DDoS is used, it refers to the DDoS Loit attack launched and collected in Friday's release. This labeling is as per what is put in the original research paper generating the dataset [31], which is also used in different other works.

3.2. Software Requirements. Python is the most commonly used Programming Language (PL) for ML works, and it is selected in this work for the development of the detection system. Jupyter Notebook is a client-server-based web application supporting interactive data science program development and presentation. It has been selected as a development IDE. Scikit-learn is used as an ML Development Framework. It is a Python library built upon Numpy and SciPy. It provides an implementation of a number of well-prepared ML algorithms used for classification, regression, clustering, parameter tuning, dimensionality reduction, and other related tasks. The Numpy library is used for mathematical computing, and Pandas, an open-source data analysis and manipulation tool, is used to provide different data analysis and extraction tasks. Finally, the Matplotlib Python library is used to create and visualize publication quality plots of various types.

3.3. ML Steps Followed in IDS Development. The following tasks have been executed to develop the IDS model. The ML development steps are shown in Figure 2.

3.3.1. Data Preprocessing. Normally, a dataset on hand cannot be processed as it is by an ML algorithm. Missing values, irrelevant features, and the issue of the categorical column are present in the CICIDS2017-Wed1 and CICIDS2017-Wed2 customized datasets. The following data cleaning and preprocessing steps have been undertaken.

I- Data Cleaning. A total of 1297 and 1299 values are not available for use in the CICIDS2017-Wed1 and CICIDS2017 Wed2 datasets, respectively, for there are nonavailable and/or infinite values. Since the missing values are mostly from the benign category and we have a high number of benign

FIGURE 2: ML steps followed.

samples, the rows associated with the nan-number values (NaNs) and infinities have been just dropped. A similar operation was performed in [23] on the same dataset.

II- Removing Constant Features. Some columns contain no kind of information required for the classification of normal and attack traffic. Ten columns in both of the customized datasets have constant values. These columns are Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, CWE Flag Count, Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk Rate, Bwd Avg Bytes/Bulk, Bwd Avg Packets/Bulk, and Bwd Avg Bulk Rate. These constant columns are irrelevant for any kind of detection work. They have been removed since they, otherwise, might result in performance decrease and unwanted complexities.

III- Categorical Data Processing. In both the CICIDS2017-Wed1 and CICIDS2017-Wed2 customized datasets, the only column having categorical information is the 'target' column which contains information about the type of traffic. A categorical encoding has been carried out in this work to convert the categorical target column to a numeric one using the available module for such a purpose, Ordinal Encoder preprocessing functionality.

3.3.2. Train Test Data Split. Splitting of the dataset for training and testing is made after data cleaning and pre-processing works. These two portions of the customized datasets are required for training the estimator and then testing the performance of the corresponding model. Two common techniques are used to generate these training and test datasets. The techniques are the percent split and K-fold cross-validation.

I. Percent Split. A 70%–30% percent split scheme has been utilized in this work. In this scheme, 70% of the overall dataset is dedicated for training the estimator and the rest 30% is reserved for testing the model.

II. K-Fold Cross-Validation. One of the challenges that occur in ML system development is the issue of overfitting [11,16,35,36]. Several algorithm-specific and generic mechanisms are there to circumvent such a problem. One of the mechanisms is to use a separate validation dataset. This is efficiently achieved by using a strategy called K-fold cross-validation [36]. In K-fold cross-validation, the entire dataset is split into K folds. One of the folds is used to test the model while the rest part is used for training. In such a way, K numbers of training sessions are made. Each time a fold that has not previously been used for testing will be kept for testing while the rest (K-1) folds are used for training. The prediction performance of the model will then be the average performance of the models in each of the K experiments. In this work, a common 10-fold cross-validation is used. The data is split into 10 folds. 10 experiments will be made each time using 9 of the folds for training and the single fold left for testing.

3.3.3. Feature Selection. One very important task in ML-related works is feature selection. Not all features available within a dataset are important or equally important for the detection of attacks. In many circumstances, increasing the number of features above a certain number does not have a noticeable significance on the classification performance. It rather might add unnecessary complexity, performance degradation, and overfitting [17].

Three types of feature selection techniques are utilized for such a purpose. These techniques are filter-based feature selection, wrapper techniques, and embedded methods. In this work, the wrapper method, more specifically Recursive Feature Elimination (RFE) which is supported in Scikitlearn, is used. An extension of the RFE, Recursive Feature Elimination with Cross-Validation (RFECV), is used to select features when using cross-validation. Although this feature selection step has been described as a separate ML step, it is actually performed alongside the training phase discussed hereinafter.

3.3.4. Training. At the heart of an ML-based work is the building of the model which is used for classification or other related tasks. That is what is accomplished in the training phase. An ML algorithm is subject to train on a portion of the

overall dataset, the training dataset, which has been prepared previously in the train test data split phase. After training an algorithm, it results in a model that has been learned from the data. A number of estimators are out there for classification. In this work, RF has been used. This estimator has been selected mainly for its high performance in related works [12,26,27].

Unlike many other works, extensive parameter tuning is made in this research work. A rigorous experimentation shall be conducted to find the best hyperparameter values since it is difficult to anticipate the effects of the changes in parameter values [37]. Criterion, min_samples_leaf, min_samples_split, max_depth, max_features, n_estimators, and others can be used for tuning an RF estimator. The n_estimator attribute has been used in this work.

3.3.5. Testing. Testing and/or validation works are very crucial and probably the final steps in an ML-based work. After the model is trained, its prediction capability shall be checked before it is deployed for specific security use. This is made by providing the model to make predictions on the other set of the dataset, the test dataset.

3.4. IDS Architecture Design. Different implementations of an IDS can be made for the proposed SDIoT system. In this work, the IDS is implemented as a separate SDN application. A modular implementation of the IDS is proposed, and the whole system is split into three modules. These modules are the raw traffic data collector module, feature extractor and aggregator module, and detector module. The placement of the modules and the data flow within the modules is seen in Figure 3. The description of the tasks accomplished in the modules is given below.

3.4.1. Raw Traffic Data Collection. The first task of the IDS is the collection of traffic statistics and related information about a flow. This is what is done by the raw data collector module. Basic information about the flows can be obtained by parsing the FLOW_REMOVED and FLOW_STATS messages of an SDN network [30]. The FLOW_REMOVED message is sent to a controller when there is no match to a certain traffic flow or there is no packet flow for a specified period of time, idle_timeout. The FLOW_STATS is a periodic message sent to a controller periodically in response to statistics information requested by a controller. More specific messages can be sent to reduce the communication overhead.

In this work, the flow statistics collection messages are used to collect the required statistics information about a flow. This implementation of the statistics collection process does not require an implementation of extra novel capabilities for such a purpose beyond what can be provided by the controller. In addition, it does not depend or has no effect on the internal implementation of a controller. The features collected are then forwarded to the feature aggregation phase to derive the necessary features required for the IDS module. Table 2 shows the summary of the features required from the switches.

FIGURE 3: Architecture of the proposed IDS system.

TABLE 2: Description of features to be collected from the switches.

No	Extracted feature	Description
1	Destination port	Port number of the destination host
2	Time interval	The duration between consecutive packets of a flow
3	Packet count	Number of packets in a flow
4	Byte count	Number of bytes in a packet

3.4.2. Traffic Aggregation. The raw traffic data collected from the switches cannot be used for detection. Important features shall be selected and aggregated from the collected raw data. The features that are to be selected and aggregated are the ones that are selected by the best performing estimator mentioned in section IV-A. These feature selection and aggregation works are done upon the features collected by the raw data collector module explained above. The flow tables of SDN switches contain the required base features which can intern be used to derive the required features.

3.4.3. Traffic Classification. Traffic classification is performed by the detector module. This detector module is the third module used in the proposed IDS system. It contains an implementation of the model developed during the ML training phase which contains the rules and parameters used

for classification. This module accepts the aggregated traffic features from the feature extractor and aggregator module. It is this module that finally determines whether there are any anomalies within the traffic flow or not. And if there are any, this module determines the type of attack on hand.

3.5. Evaluation Techniques and Measures. After the design and implementation of the detection system, a performance evaluation measurement shall be made to assess how it is performing on classifying attacks. Underneath is discussed the evaluation techniques and evaluation measures used in the research work.

3.5.1. Evaluation Techniques. Starting from the most theoretical measure of mathematical modeling, different kinds of evaluation techniques have been proposed and used to

evaluate how the various security solutions are performing. Mathematical models, simulation and emulation tools, real test-beds, and benchmark datasets can be used for such a purpose. Real or simulation/emulation-based datasets are coming to be very popular evaluation mechanisms in our days. They provide a benchmark to compare the performance of one solution against the others. In this work, a performance evaluation is done using the aforementioned CICIDS2017 benchmark dataset.

3.5.2. Evaluation Measures. Four detection performance metrics and a timing performance in terms of the number of features are used in evaluating how the system is performing in relation to the other systems. The following detection performance measures are listed below. Additional measures of Negative Predictive Value (NPV), True Negative Rate (TNR), False Positive Rate (FPR), and False Detection Rate (FDR) are also used.

Accuracy measures how often instances subject to an ML classifier are told apart correctly. It is the ratio of accurately classified instances to the total number of instances subject to classification. It is the main measure used in the measure of the performance of the model in this work.

Mathematically, it is computed as

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \quad (1)$$

Sensitivity is also known as Recall, True Positive Rate (TPR), or DR [38]. Sensitivity determines the measure of the true cases identified in terms of the total true cases. In the case of this work, it refers to the ratio of the number of attacks detected by the IDS model as an attack to the total number of attack traffic cases provided to it.

$$DR = \frac{TP}{TP + FN}. \quad (2)$$

Precision refers to the ratio of positively detected attacks to the total number of attacks labeled as an attack regardless of whether they are attacks or not. Mathematically, it is computed as follows:

$$\text{Precision} = \frac{TP}{TP + FP}. \quad (3)$$

F1-score provides a harmonic average measure of two of the aforementioned metrics, the precision and sensitivity, of an estimator [12]. It is calculated as follows:

$$F1 - \text{sscore} = \frac{2 * DR * \text{Precision}}{DR + \text{Precision}}. \quad (4)$$

NPV:

$$NPV = \frac{TN}{TN + FN}. \quad (5)$$

FPR:

$$FPR = \frac{FP}{TN + FP}. \quad (6)$$

FNR:

$$FNR = \frac{FN}{TP + FN}. \quad (7)$$

FDR:

$$FDR = \frac{FP}{TP + FP}. \quad (8)$$

4. Evaluation

4.1. Parameter Tuning with RF

4.1.1. On CICIDS2017-Wed1 Dataset. A set of experiments have been performed using RF estimators having 5, 10, 15, 20, 30, and 100 ensemble trees. The corresponding accuracy results are shown in Figure 4. The estimator with 20 ensemble trees has a maximum accuracy result of 99.968% using only 12 features. The number of features can further be reduced to 11 features with a reasonably equivalent accuracy of 99.967%.

High Performing RF Estimator. Figure 5 shows the performance of the highest performing RF estimator, the one with 20 trees. The estimator achieved an accuracy result of 83.975% using only a single feature, 89.925% with two features, and 89.940% with three features. It then immediately reached a very high accuracy value of 97.125% using only four features and then achieved its maximum value with 12 features. With the increase of the number of features beyond 12, the estimator then shows certain small downs and ups but never reaches a value greater than the maximum achieved. Table 3 shows the classification report of the estimator for the main performance measures, Table 4 shows the classification in terms of TP, TN, FP, and FN, and Table 5 shows the results of other classification performance measures. Table 6 presents the 12 best features selected by the estimator.

4.1.2. On CICIDS2017-Wed2 Dataset. Similar parameter tuning and feature selection steps had been taken on the CICIDS2017-Wed2 dataset. A maximum accuracy result of 99.954% has been achieved using an estimator having 15 ensemble trees and using 14 features.

The corresponding estimators have a little less detection performance than their equivalent estimators trained upon the CICIDS2017-Wed1 dataset. In addition, an extra number of features are required to achieve the maximum accuracy result. Yet, the estimators follow a more or less similar pattern as the number of features is trained from 1 to 68. Figure 6 shows the results collected. Figure 6(b) shows the zoomed version of Figure 6(a).

4.2. Cross-Validated Experiments

4.2.1. On CICIDS2017-Wed1 Dataset. Additional cross-validated experiments have been carried out on both two datasets. Similar estimators have been used except the fact that the estimator with 100 trees is not used here accounting for its extended performance delays and its insignificant performance enhancement. RFECV is used to select the parameters. A

FIGURE 4: Accuracy score on CICIDS2017-Wed1 dataset with RFE. (a) Accuracy score above 99%. (b) Accuracy score above 99.9%.

FIGURE 5: Accuracy score of best performing estimator. (a) Accuracy score above 99%. (b) Accuracy score above 99.9%.

TABLE 3: Classification report of the best performing estimator.

Attack	Precision	Recall	F1-score	Support
Benign	0.99980	0.99983	0.99982	131861
DoS GoldenEye	0.99808	0.99680	0.99744	3128
DoS Hulk	0.99967	0.99971	0.99969	69083
DoS Slowhttptest	0.99516	0.99516	0.99516	1653
DoS Slowloris	0.99823	0.99646	0.99734	1696
Hearbleed	1.00000	1.00000	1.00000	1
Accuracy			0.99968	207422

maximum accuracy result of 99.713% has been achieved using 15 features by the estimator having 20 ensemble trees. Corresponding to its equivalent non-cross-validated model trained using the same estimator, it takes more features to reach its maximum value.

TABLE 4: Classification status of the best performing estimator.

Attack	TP	TN	FP	FN	Support
Benign	131839	75535	22	26	131861
DoS GoldenEye	3118	204288	10	6	3128
DoS Hulk	69063	138316	20	23	69083
DoS Slowhttptest	1645	205761	8	8	1653
DoS Slowloris	1690	205723	6	3	1696
Hearbleed	1	207421	0	0	1

In addition, the estimator has lower detection performance than its corresponding non-cross-validated estimator. Figure 7 shows the results of the cross-validated experiments. Figure 7(b) shows the zoomed version of Figure 7(a).

TABLE 5: Classification report of other metrics of the best performing estimator.

Attack	NPV	TNR	FPR	FDR	FNR	Accuracy
Benign	0.999709	0.999656	0.000344	0.000197	0.000167	0.999769
DoS GoldenEye	0.999951	0.999971	0.000029	0.001921	0.003197	0.999923
DoS Hulk	0.999855	0.999834	0.000166	0.000333	0.000290	0.999793
DoS Slowhttptest	0.999961	0.999961	0.000039	0.004840	0.004840	0.999923
DoS Slowloris	0.999971	0.999985	0.000015	0.001772	0.003538	0.999957
Hearbleed	1.000000	1.000000	0.000000	0.000000	0.000000	1.000000

TABLE 6: Features selected by the best performing RF estimator.

No	Feature name
1	Destination port
2	Fwd packet length mean
3	Bwd packet length mean
4	Bwd packet length std
5	Flow IAT mean
6	Fwd Header length
7	Bwd Packets/s
8	Max packet length
9	Packet length mean
10	Average packet size
11	Subflow bwd bytes
12	Init_Win_bytes_forward

FIGURE 6: Accuracy score on CICIDS2017-Wed2 dataset with RFE. (a) Accuracy score above 99%. (b) Accuracy score above 99.9%.

4.2.2. On CICIDS2017-Wed2 Dataset. Additional experiments involving cross-validation have been conducted using the CICIDS2017-Wed2 dataset. Similar estimators and feature selection techniques as those of the cross-validated experiment upon the CICIDS2017-Wed1 dataset have been used. Maximum accuracy of only 98.523% has been achieved using 14 features by the estimator with 15 ensemble trees. Figure 8 shows the result achieved.

4.2.3. Comparison with Other Related Works. Most of the research works conducted using this dataset are driven by other motivations. Due to this reason, comparing this

research work against many CICIDS2017-based IDS systems is difficult. Gu et al. in [17] stated that results for DDoS detection performance on the CICIDS2017 dataset were missing. Although it stated that no experimental results are available, there are a few of them.

Even if it is aimed neither for SDN nor IoT systems, the work undertaken by [26] is very competitive. It has even slightly higher detection accuracy results and a slightly better number of feature use than this research work in a certain case. However, the work is beaten in detection accuracy for the original unsampled CICIDS2017 dataset. But, more importantly, it is PCA that has been used for

FIGURE 7: Cross-validated accuracy result on CICIDS2017-Wed1 dataset. (a) Accuracy score above 90%. (b). Accuracy score above 99%.

FIGURE 8: Cross-validated accuracy result on CICIDS2017-Wed2 dataset.

TABLE 7: Comparison with works targeting DoS/DDoS attacks.

No	Work	Best accuracy (%)	Best precision (%)	Number of features	SDN based	Remark
1	[This]	99.968	—	12	Yes	DoS
		99.954	—	14	Yes	DoS and DDoS
2	[12]	—	99.2	—	No	DoS
3	[24]	99.844	—	8	No	DoS
		99.981	—	8	No	Only DDoS
4	[16]	99.9	—	—	No	Only DDoS
5	[17]	—	98.86 (DR)	—	No	DoS
6	[25]	—	100	—	No	Only HTTP slow header DoS

TABLE 8: Comparison with works involving the generic CICIDS2017 dataset.

No	Work	Best accuracy (%)	Precision (%)	No of features	SDN based	Remark
1	[26]	99.6	98.8 (DR)	10	No	For all attacks
		—	98.323	10	No	DoS and DDoS
		100	99.999	10	No	DoS and DDoS (class balanced dataset)
2	[27]	99.16	—	—	No	Overall result
		—	98.647	—	No	DoS
3	[28]	95.53	98.94	-	No	Result for DDoS
4	[23]	99.665	94.457 (DR)	—	No	Overall result
		—	97.972 (DR)	—	No	DoS and DDoS
5	[13]	99.912	99.85	—	No	Overall result
		99.778	99.942	—	No	Overall result

dimensionality reduction. This has a performance overhead for use in IoT systems unlike a simple feature selection mechanism used in this work. In addition, the work is prone to overfitting since no kind of cross-validation works has been conducted.

Table 7 presents a comparison of works conducted with a focus on DoS/DDoS attacks while Table 8 presents works that focus on the whole attacks prevailing in the CICIDS2017 dataset. The number of attacks mentioned in Table 1 is used for weighting the accuracy results of the corresponding attacks.

5. Conclusion

Several experiments have been undertaken and high accuracy results have been achieved. A new high accuracy result of 99.968% has been achieved on the CICIDS2017-Wed1 dataset using only 12 features. Feature selection and parameter tuning have been used to enhance the detection performance and efficiency of the estimators. The $n_{estimators}$ parameter has been used for tuning the estimators. High cross-validated accuracy results of 99.713% have been achieved using 15 features. Models trained using the CICIDS2017-Wed2 dataset have slightly lower detection performance than their corresponding estimators trained using the CICIDS2017-Wed1 dataset.

The high detection accuracy collected with the use of only 12 features accompanied by the use of a relatively faster ML algorithm makes the collected result suitable for use in smart environments. This model meets the basic requirement of supervised IDS systems developed for smart environments. It can effectively be used with other related IDS systems supporting other cyberattacks and providing anomaly detection to provide a sound detection system. Even if it has a slightly less accurate result, the high-performing cross-validated model is a competent solution since it is expected to be more immune to overfitting.

Data Availability

The data are available at the Canadian Institute for Cybersecurity (<https://www.unb.ca/cic/datasets/ids-2017.html>).

Additional Points

This research work was initially conducted as a partial fulfillment of the requirements for the degree of Master of

Science in computer engineering from Bahir Dar Institute of Technology. Among other differences, different random state values were used in the original work with the aim of increasing the likelihood of achieving a high-performing estimator. A constant random state value has been chosen for experiments in this work.

Conflicts of Interest

The authors declare that they have no Conflicts of Interest.

References

- [1] C. Vandana, "Security improvement in iot based on software defined networking (sdn)," *International Journal of Science, Engineering and Technology Research (IJSETR)*, vol. 5, no. 1, pp. 2327–4662, 2016.
- [2] T. Qiu, N. Chen, K. Li, M. Atiquzzaman, and W. Zhao, "How can heterogeneous internet of things build our future: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2011–2027, 2018.
- [3] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of iot: Applications, challenges, and opportunities with china perspective," *IEEE Internet of Things journal*, vol. 1, no. 4, pp. 349–359, 2014.
- [4] J. H. Cox, J. Chung, S. Donovan et al., "Advancing software defined networks: A survey Advancing Software-Defined Networks: A Survey," *IEEE Access*, vol. 5, pp. 25487–25526, 2017.
- [5] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A survey on softwaredefined wireless sensor networks: Challenges and design requirements A Survey on Software-Defined wireless Sensor Networks: Challenges and Design Requirements," *IEEE access*, vol. 5, pp. 1872–1899, 2017.
- [6] S. K. Tayyaba, M. A. Shah, O. A. Khan, and A. W. Ahmed, "Software defined network (sdn) based internet of things (iot) a road ahead," in *Proceedings of the International Conference on Future Networks and Distributed Systems*, vol. 3, pp. 1–8, 2017.
- [7] D. Yin, L. Zhang, and K. Yang, "A ddos attack detection and mitigation with software-defined internet of things framework," *IEEE Access*, vol. 6, pp. 24694–24705, 2018.
- [8] M. F. Elrawy, A. I. Awad, and H. F. Hamed, "Intrusion detection systems for iot-based smart environments: a survey," *Journal of Cloud Computing*, vol. 7, no. 1, p. 21, 2018.
- [9] L. Liang, K. Zheng, Q. Sheng, and X. Huang, "A denial of service attack method for an iot system," in *Proceedings of the 2016 8th International Conference on Information Technology*

- in *Medicine and Education (ITME)*, pp. 360–364, IEEE, Fuzhou, China, December 2016.
- [10] D. M. Mendez, I. Papapanagiotou, and B. Yang, “Internet of things: survey on security and privacy,” 2017, <https://arxiv.org/abs/1707.01879>.
 - [11] B. A. Tama, L. Nkenyereye, S. M. R. Islam, and K.-S. Kwak, “An enhanced anomaly detection in web traffic using a stack of classifier ensemble,” *IEEE Access*, vol. 8, pp. 24120–24134, 2020.
 - [12] F. S. D. Lima Filho, F. A. Silveira, A. de Medeiros Brito Junior, G. Vargas Solar, and L. F. Silveira, “Smart detection: an online approach for dos/ddos attack detection using machine learning,” *Security and Communication Networks*, vol. 2019, Article ID 1574749, 2019.
 - [13] Y. Zhang, X. Chen, L. Jin, X. Wang, and D. Guo, “Network intrusion detection: based on deep hierarchical network and original flow data,” *IEEE Access*, vol. 7, pp. 37004–37016, 2019.
 - [14] G. Loganathan, “Real-time intrusion detection using multi-dimensional sequence-to-sequence machine learning and adaptive stream processing,” 2018.
 - [15] A. Yulianto, P. Sukarno, and N. A. Suwastika, “Improving adaboost-based intrusion detection system (ids) performance on cic ids 2017 dataset,” *Journal of Physics: Conference Series*, vol. 1192, Article ID 012018, 2019.
 - [16] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system,” *IEEE Access*, vol. 7, pp. 41525–41550, 2019.
 - [17] Y. Gu, K. Li, Z. Guo, and Y. Wang, “Semi-supervised k-means ddos detection method using hybrid feature selection algorithm,” *IEEE Access*, vol. 7, pp. 64351–64365, 2019.
 - [18] P. A. A. Resende and A. C. Drummond, “Adaptive anomaly-based intrusion detection system using genetic algorithm and profiling,” *Security and Privacy*, vol. 1, no. 4, p. e36, 2018.
 - [19] T. Mahjabin, Y. Xiao, G. Sun, and W. Jiang, “A survey of distributed denial-of-service attack, prevention, and mitigation techniques,” *International Journal of Distributed Sensor Networks*, vol. 13, no. 12, 2017.
 - [20] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges,” *Cybersecurity*, vol. 2, no. 1, p. 20, 2019.
 - [21] J. Li, Z. Zhao, R. Li, and H. Zhang, “Ai-based two-stage intrusion detection for software defined iot networks,” *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2093–2102, 2018.
 - [22] K. A. P. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, “Internet of things: a survey on machine learning-based intrusion detection approaches,” *Computer Networks*, vol. 151, pp. 147–157, 2019.
 - [23] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour, and H. Janicke, “A novel hierarchical intrusion detection system based on decision tree and rules-based models,” in *Proceedings of the 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 228–233, IEEE, Santorini Island, Greece, May 2019.
 - [24] S. Singh Panwar, Y. Raiwani, and L. S. Panwar, “Evaluation of network intrusion detection with features selection and machine learning algorithms on cids-2017 dataset,” 2019.
 - [25] G. Loganathan, J. Samarabandu, and X. Wang, “Real-time intrusion detection in network traffic using adaptive and auto-scaling stream processor,” in *Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, UAE, December 2018.
 - [26] R. Abdulhammed, H. Musaffer, A. Alessa, M. Faezipour, and A. Abuzneid, “Features dimensionality reduction approaches for machine learning based network intrusion detection,” *Electronics*, vol. 8, no. 3, p. 322, 2019.
 - [27] V. Gustavsson, “Machine learning for a network-based intrusion detection system: an application using zeek and the cids2017 dataset,” 2019.
 - [28] A. Janagam and S. Hossen, “Analysis of network intrusion detection system with machine learning algorithms (deep reinforcement learning algorithm),” 2018.
 - [29] K. Kalkan and S. Zeadally, “Securing internet of things with software defined networking,” *IEEE Communications Magazine*, vol. 56, no. 9, pp. 186–192, 2017.
 - [30] G. A. Ajaeiy, N. Adalian, I. H. Elhadj, A. Kayssi, and A. Chehab, “Flowbased intrusion detection system for sdn,” in *Proceedings of the 2017 IEEE Symposium on Computers and Communications (ISCC)*, pp. 787–793, IEEE, Heraklion, Greece, July 2017.
 - [31] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proceedings of the 2017 International Conference on Information Systems*, pp. 108–116, Porto, Portugal, January 2018.
 - [32] T. Shorey, D. Subbaiah, A. Goyal, A. Sakxena, and A. K. Mishra, “Performance comparison and analysis of slowloris, goldeneye and xerxes ddos attack tools,” in *Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 318–322, IEEE, Bangalore, India, September 2018.
 - [33] Can Heartbleed Be Used in Ddos Attacks? | network world.” <https://www.networkworld.com/article/2176153/canheartbleed-be-used-in-ddos-attacks-.html>.
 - [34] High-severity Vulnerability in Openssl Allows Dos Attacks | securityweek.Com.” <https://www.securityweek.com/high-severity-vulnerability-openssl-allows-dos-attacks>.
 - [35] E. Anthi, L. Williams, M. Slowinska, G. Theodorakopoulos, and P. Burnap, “A supervised intrusion detection system for smart home iot devices,” *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9042–9053, 2019.
 - [36] T. M. Mitchell, “Machine learning,” 1997.
 - [37] How to Tune a Decision Tree? - towards Data Science.” <https://towardsdatascience.com/how-to-tune-a-decision-tree>.
 - [38] Precision and recall definition | deepai.” <https://deepai.org/machine-learning-glossary-and-terms/precision-and-recall>.