Received: 27 January 2020 | Revised: 13 August 2020 | Accepted: 7 September 2020 | CAAI Transactions on Intelligence Technology

DOI: 10.1049/cit2.12003

ORIGINAL RESEARCH PAPER

# SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL)

**Azka Wani**[1] | **Revathi S**[2] | **Rubeena Khaliq**[3]

[1]Department of Computer Applications, Crescent B S Abdur Rahman Institute of Science and Technology, Chennai, India

[2]Department of Computer Science and Engineering, Crescent B S Abdur Rahman Institute of Science and Technology, Chennai, India

[3]Department of Mathematics, Crescent B S Abdur Rahman Institute of Science and Technology, Chennai, India

**Correspondence**

Azka Wani, Department of Computer Applications, Crescent B S Abdur Rahman Institute of Science and Technology, Vandalur, Chennai – 600048, India.
Email: graceazka@gmail.com

**Funding information**

MANF UGC, Government of India, Grant/Award Number: MANF-2015-17-JAM-60506

**Abstract**

The participation of ordinary devices in networking has created a world of connected devices rapidly. The Internet of Things (IoT) includes heterogeneous devices from every field. There are no definite protocols or standards for IoT communication, and most of the IoT devices have limited resources. Enabling a complete security measure for such devices is a challenging task, yet necessary. Many lightweight security solutions have surfaced lately for IoT. The lightweight security protocols are unable to provide an optimum protection against prevailing powerful threats in cyber world. It is also hard to deploy any traditional security protocol on resource-constrained IoT devices. Software-defined networking introduces a centralized control in computer networks. SDN has a programmable approach towards networking that decouples control and data planes. An SDN-based intrusion detection system is proposed which uses deep learning classifier for detection of anomalies in IoT. The proposed intrusion detection system does not burden the IoT devices with security profiles. The proposed work is executed on the simulated environment. The results of the simulation test are evaluated using various matrices and compared with other relevant methods.

## 1 | INTRODUCTION

The concept of Internet of Things (IoT) allows interconnection of ordinary objects through Internet. IoT joins devices of varying strengths and produces huge amount of data. With the expansion of IoT and increased automation, 5G network is expected to improve IoT security and other challenges faced by IoT [1]. IoT devices are resource constrained, hence security and protection are hard to enforce [2]. The analysis of traffic in a network and detection of any abnormal behaviour are resource-heavy. Many lightweight methods for security enhancement in IoT have been developed over the last few years [3, 4], but such mechanisms cannot upfront the massive security threats that have been identified lately. An intrusion detection system (IDS) protects the networks proactively against any anomaly. IDS needs resources to operate and it is difficult to protect constrained IoT system against cyberattacks [5]. The state-of-the-art IDS is based on machine learning and deep learning techniques. Such IDS does not need human intervention for operation and does not require the system to be altered [6]. Deploying such an IDS for a system with limited capabilities is quite suitable

An software-defined networking (SDN)-based intrusion detection and prevention system for IoT is introduced as an enhancement to our previous work [7]. The mechanism uses the features of SDN to design a proactive system for intrusion detection in the IoT network. The SDN-based system allows programming of networks. It separates the control and data planes, and provides a global view of the network. Based on its programmability feature and global view, SDN is considered as a better alternative to overcome challenges faced in the smooth operation of IoT [8, 9]. The proposed method solves the security problem in IoT by comparing the actions and data flows within the network.

The contributions of the paper are as follows:

- The security issues prevalent in the IoT have been highlighted.
- An intrusion detection system based on SDN framework for the IoT has been introduced.

- The intrusion detection system uses deep learning classifier to detect the abnormalities in a network.
- The Long Short-Term Memory (LSTM) classifier is trained using the latest dataset that includes most recent attacks of common platform as well as IoT.
- The proposed algorithm is executed to detect anomalies in IoT and identify attacks with accuracy.

An SDN-based IDS for IoT network is proposed. The proposed IDS detects the anomalies using the northbound interface of SDN framework. The proposed method makes use of deep learning algorithm for efficient classification of normal and malicious traffic. The paper is organized as follows: the security issues are highlighted in Section 2. In Section 3, the IDSs and SDN are briefly introduced. The work related to IDSs in IoT is presented in Section 4. Section 5 discusses the proposed solution in detail. The dataset used in the proposed method is discussed briefly in Section 6. The experimental results and performance of the proposed method are given in Section 7.

## 2 | SECURITY ISSUES IN IoT

IoT is an important invention of contemporary times. IoT has been of great benefit to industries and it resuled in automation of maintenance and management [10]. IoT has found its way in every aspect of human lives. The invention of IoT has been a striking discovery, but it is also vulnerable to a number of security threats. The vulnerabilities in IoT devices can cause severe consequences, if left unattended. The adversaries can take advantage of the vulnerabilities of IoT, alter data and create havoc by exploitation of the global IoT network. If IoT security issues are not addressed in timely manner, these can surpass the advantages of IoT [11]. The attacks that pose a major threat to the security of IoT devices have been listed as follows [12]:

- Denial of service (DoS)
- Distributed denial of service (DDoS)
- Man in the middle
- Transmission Control Protocol attacks
- Firmware auto-upgrading attacks
- Heartbleed
- Botnet
- Data/SQL injection
- Password attacks

The limited resource capacity and battery operation of the IoT devices prevent enforcement of a complete security mechanism for each device. The filtering of traffic at the network layer is a better way of ensuring security in IoT [13]. The network-based security mechanism is applied at the IoT gateway to monitor the incoming and outgoing traffic. Based on the normal behaviour of network traffic, a template is formed against which the incoming and outgoing packets are compared. The intruders are restricted from accessing the IoT devices that are registered with the IoT gateway. If the network traffic, at some point, does not match the template, the network is suspected to be under attack and an alarm is raised.

## 3 | RELATED WORK

The huge influx of data and heterogeneity of devices makes management of IoT devices a tedious task. The security of IoT is one of the biggest challenges in cyber world. IDSs have been built using different methods and techniques to protect and safeguard the IoT networks [11]. Recently, SDN-based IoT framework has come up as a solution to many concerns in IoT [14]. In a typical SDN-based IoT set-up, the devices are handled with the software applications that are deployed on an SDN controller [15–17]. The applications are designed to carry out control and management-related tasks in the underlying IoT network. Some notable intrusion detection models that have been introduced lately to combat the security issues in IoT are highlighted as follows:

Deng et al. [18] have first defined the various types of intrusions that are exploiting the IoT system and have discussed the detection and prevention mechanism that can counter such intrusions. The authors have studied the security in IoT based on confidentiality, integrity and availability. The working of different intrusion detection technologies has been compared and based on the evaluation an outlook for future research is presented.

Liu et al. [3] have introduced an intrusion detection for IoT-suppressed fuzzy logic and principal component analysis. The traffic is divided into two categories. The simulation experiments are carried out to understand the factors that influence the algorithms used in the proposed method. The results of the experiment depict that increasing data volume reduces the efficiency and accuracy of the proposed IDS. However, the results are better as compared to two other algorithms shown in the paper.

Nobakth et al. [19] have introduced a model called IoT-IDM that uses SDN and machine learning to design a network IDS. A working model of the proposed method is developed and used on a real environment. IoT-IDM notifies the network devices about the attack. IoT-IDM is developed as a module for Floodlight controller of SDN. IoT-IDM has been designed for smart home application of the IoT network.

Hossien et al. [20] have designed a lightweight Artificial Immune System (AIS)-based IDS for the IoT network. The authors have presented the IoT in three segments of cloud, fog and edge. The training for detection is done in cloud layer. The fog segment works on intrusion notifications using the concept of smart data. The detectors are implemented on edge devices.

Midi et al. [21] designed knowledge-driven, self-adapted intrusion detection for IoT called Kalis. Kalis collects the features of the network and then automatically adapts the detection technique for the network. The suggested model can also be deployed on a stand-alone device to support resource-constrained IoT devices. Internet Control Message Protocol

(ICMP) flood attack and Smurf attack are considered for evaluation.

Hassan et al. [22] have used Naïve Bayes algorithm for classifying the traffic for intrusion detection in their proposed IDS. The authors have used Weka tool for processing attack data and relevant information. The notification of the attack is send to the device or the administrator.

Kumar et al. [23] proposed an IDS for IoT which includes a newer dataset and tries to detect the recent attacks. The work states that the proposed solution performs better than many recent IDS and prevents from internal malicious attacks as well.

In our previous work [7], we have introduced an IDS for IoT based on SDN and machine learning. The backward propagation algorithm has been used in the classifier. The ICMP flood attack has been simulated to evaluate the working of the model.

## 4 | PROPOSED WORK

Based on characteristics of IoT privacy and security, a SDN-based IDS has been proposed to monitor the traffic flow. To design an efficient security model for IoT, the experiments are performed on a dataset containing normal and malicious data relevant to an IoT scenario. Any IDS system placed in a network gets the network statistics and classifies the traffic as malicious or normal.

An IDS called IDSIoT-SDL that identifies security breaches in an IoT environment is presented in Figure 1.

The system is able to detect the set of attacks that threaten the normal functioning of the IoT. The proposed system detects anomalies in IoT using features of the SDN. The classical approaches of network intrusion detection have not proved to be beneficial for an IoT setup [24]. The IoT devices have limited resources and operate on battery. Any network device should have sufficient resources for inclusion of a complete IDS. Many lightweight IDS have been tailor made for IoT devices, but such systems may not always counter the voluminous attacks that have surfaced recently. IDSIoT-SDL is deployed in a SDN-based IoT setup for smart working of the IDS. The use of SDN makes it possible to include functionality of an entire IDS for a resource constrained IoT environment. The IDSIoT-SDL uses the traffic-sniffing tool and statistics recording feature of the OpenFlow switches to obtain the flow features and aggregate those for intrusion detection. In IDSIoT-SDL, deep learning approach has been used for detection of any abnormality in the IoT network. The classical methods for intrusion and detection have been replaced by machine learning techniques recently [25, 26]. The machine learning methods of securing the networks have proved to be finer than the traditional approaches. The concept of deep learning has emerged lately and is proving to be beneficial in almost every field. The application of deep learning in the area of cyber security is also quite impactful [27]. The deep learning approach for detection of attacks has ability to overcome the shortcomings of the traditional approaches. Machine learning solutions are precise and can detect maximum anomalies. There is an option of sharing diagnostic data with a database and input the database to a deep learning engine to find vulnerabilities [28].

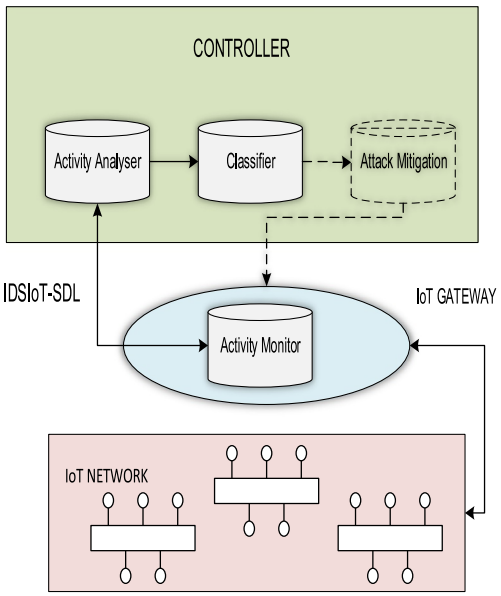



**FIGURE 1** Working of IDSIoT-SDL

The deep learning algorithms can accommodate the heterogeneity of the IoT devices. The data gathered from various users and devices are sent to a machine/deep learning program, which can intelligently rectify the malicious behaviour. A firmware update or patch can be loaded to the devices separately to avoid any threats.

IDSIoT-SDL uses deep learning approach for analysing the network and detecting the intrusion. The network packets meant for each IoT device pass through Openflow switch that acts as a gateway to the IoT domain. The SDN controller manages all the OpenFlow switches in a network. The global view of network in a SDN helps IDS to detect the anomalies in the IoT network. The intrusion detection is performed with the help ofLSTM deep learning algorithm.

### 4.1 | LSTM overview

LSTM neural networks are an improvement over the recurrent neural networks (RNNs). The LSTM networks contain different memory blocks called *cells*. In LSTM, two entities are forwarded to the next cell. Those two entities are the *cell state* and the *hidden state*. The cells are required to remember the information, that is, the cell state and hidden state. The actions are performed to these cells or memory units with the help of three functions called *gates*, namely *input gate*, *output gate* and *forget gate*. These three gates (*ig*, *fg*, *og*) maintain the flow of information. These gates control the flow of information and do not allow any invalid manipulations to be done on the cells, and hence make long-term storage possible for LSTM networks [29].

The *input gate* is responsible for adding any latest information. The input gate adds information to the cell state in three steps:

(a) A sigmoid function is used to regulate the values that are to be added to the cell state.
(b) Set of values that can be added to cell state are represented by a vector using tan*h* function.
(c) The sigmoid function value is multiplied by the created vector (the tanh function) and the result is added to the cell state.

The *fg* enhances the performance of a LSTM network by removing unnecessary information from the cell. The information of lesser priority is removed using a filter. There are two inputs of an *fg*: $h_{t-1}$ and $x_t$, where $t$ is the time, $h_{t-1}$ can be previous hidden state or previous cell output, and $x_t$ is the input at that particular time step. The given inputs are multiplied by the weight matrices and a bias is added. The sigmoid function is applied to the result of the previous step. The output of sigmoid function is a vector ranging from 0 to 1, corresponding to each number in the cell state. The output of the sigmoid function is multiplied to the cell state. When the output of the sigmoid function is '0', the *fg* has to forget the information and if output is '1', then the *fg* has to remember the information.

The *og* selects the useful information from the current cell state and sends it out as an output. The output gate does following job: (a) tanh function is applied to the cell state and creates a vector to range the values from −1 to +1. (b) A filter is used with values $h_{t-1}$ and $x_t$ to control the output from the vector created in the above step. The sigmoid function is again used. (c) The values of filter and vector from the first step are multiplied, which forms the output [30].

The operations of *ig, fg and og* are given in following equations:

$$ig_c = \sigma\left(W_{ig}.[C_{c-1}, h_{c-1}, i_c] + b_{ig}\right)$$

$$fg_c = \sigma\left(W_{fg}.[C_{c-1}, h_{c-1}, i_c] + b_{fg}\right)$$

$$og_c = \sigma\left(W_{og}.[C_c, h_{c-1}, i_c] + b_{og}\right)$$

where $ig_c$, $fg_c$ and $og_c$ represent the operations *ig*, *fg* and *og*, respectively. $W_{ig}$ $W_{fg}$ $W_{og}$ are the weight matrices for the *ig*, *fg* and *og*, respectively. $b_{ig}$ $b_{fg}$ $b_{og}$ are the biases for the *ig*, *fg* and *og*, respectively. $C_{c-1}$ is the previous cell state, $C_c$ is current cell memory, $h_{c-1}$ is the previous hidden state or previous output, and $i_c$ is the current input.

## 5 | SDN-BASED INTRUSION DETECTION SYSTEM FOR IoT (IDSIoT-SDL)

The IDSIoT-SDL consists of three major components, namely:

- Activity monitor,
- Activity analyser
- Classifier

These components carry out the following tasks:

- traffic capturing
- traffic parsing
- feature extraction
- classifier learning
- anomaly detection

The controller gets the statistical information from the OpenFlow switches and forwards it to another component of IDSIoT-SDL, the activity analyser. Algorithms 1 describes the working of IDSIoT-SDL. The working of proposed method is also shown in the flowchart (Figure 2).

**Algorithm 1** *Working of IDSIoT-SDL*

Input: Flow statistics
Output: Anomaly Detection
*Step 1*: determine suspicious packets from filter
*Step 2*: parse traffic
*Step 3*: determine $n$
*Step 4*: extract $x \in n$
*Step 5*: input $\rightarrow \mathbb{C}$
*Step 6*: train C
*Step 7*: execute C
*Step 8*: start timer $t = 0$
*Step 9:* ***while*** $t \leq \mu$
*Step 10*: $\beta++$
*Step 11*: ***if*** $\beta \geq \varphi$ *then*
*Step 12*: suspend hosts in $\beta$
*Step 13:* if anomaly == attack
*Step 14:* raise attack alert
*Step 15:* identify the attack type
*Step 16:* invoke mitigation
*Step 17*: **end if**
*Step 18*: **else**
*Step 19*: raise False Flag
*Step 20*: **end if**
*Step 21*: **end while**
*Step 22*: reset timer $t$
*Step 23*: go to *Step 9*

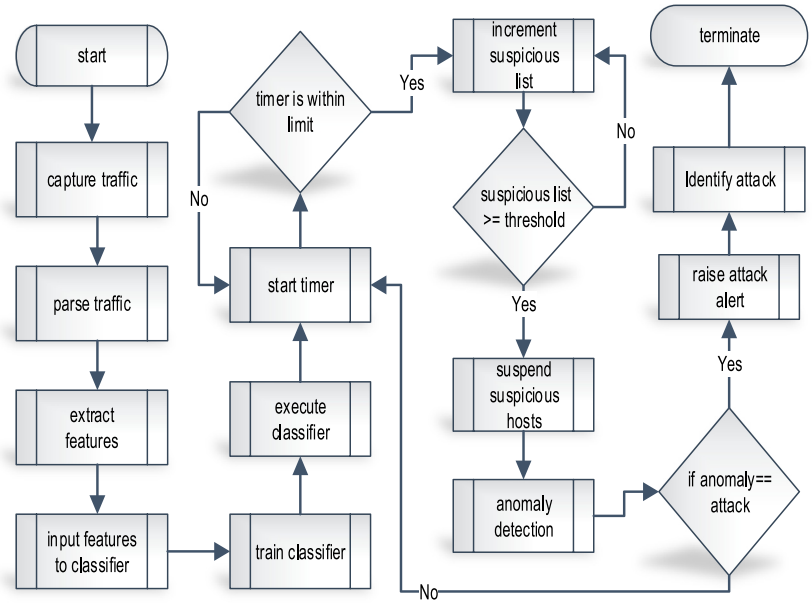
where $n$ represents the features of traffic,

$x$ = subset of $n$, that is, the reduced set of features used for learning classifier,

C = the classifier,
$t$ = the time elapsed,
$\mu$ = static time frame,
$\beta$ = the list of suspicious hosts,
$\varphi$ = the predetermined threshold for maximum number of malicious hosts detected in a static time frame $\mu$.

SDN controllers send request messages to the OpenFlow switches seeking the network statistics [31]. The Openflow switches with help of Wireshark tool [32] collects the network traffic. The IDSIoT-SDL processes the information from the network statistic to detect any intrusion in the network. Upon sensing some intrusion in the IoT the OpenFlow switches can

**FIGURE 2** Workflow of IDSIoT-SDL




mitigate the attack by dropping the traffic from the malicious source or by changing the security policies of the network. The mitigation of the attacks has not been discussed, since different procedure needs to be followed for specific attack type. The detailed working of IDSIoT-SDL has been explained in following subsections:

## 5.1 | Activity monitor

Activity monitor is the component responsible for observing the traffic flow of the IoT domain. The activity monitor gathers information needed for finding any abnormal behaviour within the IoT network. This component uses the pre-existing feature of an OpenFlow switch and Wireshark tool to record network traffic statistics. SDN allows flow-based detection of malicious traffic. The SDN controller obtains the statistical information about the IoT devices using OpenFlow switch as gateway. The controller identifies each OpenFlow switch using DataPath ID (DPID). The OpenFlow switches contain the flow tables with flow entries. Each flow entry includes fields like *match fields, priority counters, actions/instructions* and *cookie*. On receiving a flow of packets, its match is searched in the flow table. Upon finding a match, the count of incoming packets is increased and the message is forwarded or dropped as per the *actions* field. If the flow does not match with any flow entry, a *packet_in* message is created and forwarded to the SDN controller. The *packet_in* holds the header of the first packet of the flow. The SDN controller sends a new flow rule to the OpenFlow switch, after parsing the header information. The activity monitor uses the northbound interface to obtain the statistics of every flow from the OpenFlow switches by sending a request. The tasks of IDSIoT-SDL included in Activity Monitor are:

1. Traffic capturing: This is the process of recording the traffic on a network. The traffic is captured using *wireshark* tool. The *dumpcap* program of *wireshark* acquires the raw network traffic.
2. Traffic parsing: This involves analysis of raw traffic to acquire the meta data needed in selection of features. The program *tshark* is used for traffic parsing.

The information from the Activity Monitor is sent back to the controller. This information is used by the next component of IDSIoT-SDL for correct intrusion detection [33].

## 5.2 | Activity analyser

The important task of IDSIoT-SDL is to detect any suspicious behaviour in the network and Activity Analyser plays a major role in that. Based on the statistics information collected by Activity Monitor, any abnormal behaviour of the network is detected. The type of network attack is identified, if attack takes place. LSTM deep learning algorithm is used for the purpose of anomaly and attack detection in IDSIoT-SDL. The task carried out by Activity Analyser are:

### 5.2.1 | Feature extraction

The deep learning method needs the features of the captured traffic for learning. The attributes of the data packets (*fl_dur*: Flow duration, *fw_pkt_l_max*: Maximum size of packet in forward direction, *fw_pkt_l_min*: Minimum size of packet in forward direction, *pkt_len_min*: Minimum length of a flow, *pkt_len_max*: Maximum length of a flow, *pkt_len_avg*: Mean length of a flow, *pkt_len_std*: Standard deviation length of a

flow, *pkt_len_va*: Minimum inter-arrival time of packet, etc.) that are extracted first are denoted by $'n'$. The feature extraction task is responsible for distilling meaningful features $'x'$ from the captured and parsed network traffic. Feature selection [34] has key importance in determining the accuracy of an intrusion detection system. IDSIoT-SDL uses deep learning feature extraction from the captured data. Deep learning algorithms are capable of extracting features from limited samples of data [28]. LSTM can abstract features of the data packets intelligently.

## 5.3 | Classifier and alert mechanism

To protect the IoT network from any possible intruders, the classifier distinguishes legitimate traffic from attack traffic. The information taken from the activity monitor is classified as malicious or benign. It also identifies the specific attack that might have attacked the network. Once the classification of information is done and it detects any attack, the alert is raised and control is shifted towards mitigation strategy. The tasks carried out in classifier component are as follows:

### 5.3.1 | Classifier learning

*The classifier learning is the most essential task of any IDS. First a reduced set of features called 'x' is created from 'n' known as mapping. The reduced set 'x' is input to the anomaly detector. This task works simultaneously along with the anomaly detection. The LSTM based classifier is first trained using the dataset* CSE-CIC-IDS2018 [35]. The signature analysis is also carried out to understand the possible nature of known IoT attacks, before supplying the data to classifier $\mathbb{C}$. The accuracy of detection by algorithm is evaluated and improved by changing hyper-parameters [36]. Once the classifier is trained, it has to be executed for anomaly detection of the IoT. The process begins with initializing a timer. On detecting any sort of malicious behaviour by a host, the host is added to a list of suspicious hosts called $\beta$. The list is expanded until its count reaches a predetermined threshold value $\varphi$ within a given time frame $\mu$. The hosts in the list are then suspended and suspicious hosts are further evaluated for exact attack detection.

### 5.3.2 | Anomaly detection

This task is responsible to detect abnormal behaviour in the network and figure out if abnormality is because of some attack in the IoT network. The hosts in the suspicious list $\beta$ are evaluated to detect whether the anomaly is because of attack. If there is an attack then the alert is raised and type of the attack is identified. The attack mitigation measures are then taken to tackle the type of adversary detected. If the suspicious behaviour does not confirm any attack, then the hosts in $\beta$ are allowed to participate in the network again.

## 5.4 | Dataset for IDSIoT-SDL

The dataset for training machine/deep learning algorithms for IDS mainly include KDD Cup'99, NSD-KDD, UNB ISCX, DARPA KDD [34]. The CSE-CIC-IDS2018 is used as dataset for training in IDSIoT-SDL [35]. This dataset includes normal traffic and latest attack types. The data in CSE-CIC-IDS2018 includes data similar to actual IoT network traffic or *pcap* (packet capture data). It also contains the traffic flows with time stamps, origin and destination IPs, origin and receiving ports, network protocols or *csv*(comma-separated value) files. In each traffic record, there are distinct features that depict the nature of traffic flow and each feature is associated with either attack traffic or normal traffic. The features of each network connection vector can be of following types: basic features, content-related, time-related traffic features or host-based traffic features.

The major attack classes included in the CSE-CIC-IDS2018 dataset are: (a) DoS: Denial of Service attack that makes the resources of victim unavailable for legitimate requests. The attributes associated with DoS attack are 'source bytes' and 'number of malign packets'. (b) Brute Force FTP: An attacker sniffs the data packets communicated between server and client to gain credential information and uses it for unauthorized access. (c) Brute Force SSH: These are the Brute force attacks on the Secure Shell (SSH). These attacks are launched for compromising the accounts with weaker passwords. These were first identified more than a decade ago, but are still prevalent. (d): Heartbleed: The Heartbleed is a bug that exploits OpenSSL. This vulnerability opens a way for theft of data encrypted using SSL/TLS protection. (e): Botnet and DDoS: DDoS stands for distributed denial of service. A DDoS keeps a server or a network resource unavailable for legitimate users. It constantly floods the server or network resource, which suspends the server. A botnet is a malware infected network device that is used by attacker to launch DDoS attack. An attacker from remote location controls such devices. A DDoS attack uses multiple connected botnets to suspend a target. CSE-CIC-IDS2018 dataset consists of 80 features. In IDSIoT-SDL some features are selected from the 80 features set of dataset for training and testing of the classifier. The features selected need to be easily attainable in an SDN setup and must help in creating realistic IDS. The features are selected using the deep learning LSTM approach

# 6 | PERFORMANCE ANALYSIS

This section presents the experimental details that exhibit the performance of IDSIoT-SDL framework. The proposed mechanism has been implemented using Mininet WiFi [37]. The controller used for the setup is the Floodlight controller, customized to incorporate IDSIoT-SDL. An OpenFlow enabled switch is required as gateway to IoT network. OpenFlow version 1.5 [38] has been used in the experiment. IDSIoT-SDL is implemented in the control layer as well as OpenFlow switch. It receives communication from the

gateway and sends information back to it. The attacks simulations show ability of system against various attacks in IoT environment. The IDSIoT-SDL considers normal and malign class of traffic for classification. The performance of any deep learning-based model depends on the optimization of hyper-parameters. In case of the proposed method, tuning of hyper-parameters is done using Bayesian optimization with Hyperbot library of Python. The number hidden layers used is three, since more number of hidden layers would make the detection process time consuming. The metrics for evaluation and comparison with other related models are discussed next.

## 7 | RESULTS AND DISCUSSION

The results of IDSIoT-SDL classifier are presented in this section and the results are compared with relevant models to determine the effectiveness of the system. The system is evaluated for the testing data. The simulation is tested on one gateway switch, which is the network edge of the IoT domain. The normal traffic flow consists of requests like transfer of files, notifications, alerts and voice over IP. The total number of instances for normal traffic used are 17,899 in number. The attack traffic includes instances of ICMP Flood, Heartbleed, UDP DoS, Port Scan, Brute Force and Network SYN Scan. The total number of malign requests constituting the attack traffic are 35,731. The distribution of normal traffic flows and different attack traffic flows are shown in Table 1. The instances of the attack traffic like SYN Scan, Port Scan can be taken under same class of attacks, whereas ICMP Flood, UDP DoS and Brute Force come under one category of attacks.

As per the analysis of these attacks, the simulation is carried out on four classes of traffic: ICMP Flood, HeartBleed, Brute force and normal traffic.

The dataset used contains normal traffic samples as well as samples originating from different attack classes. The samples were divided into five batches with each batch representing 20% of the dataset, the four batches were used for training and one batch for testing. The number of samples chosen was a smaller set (2006 in this case) resembling the smallest from the traffic class of four elements so that the dataset is uniform for training and testing. This uniform dataset has 1605 samples for training and remaining 401 samples for testing.

Once the system is trained, the hyper-parameters are optimized for different values. The hyper-parameters of LSTM are varied to achieve the best results. The model is trained and tested for the time step range as shown in Figure 3. The time-step set for the proposed mechanism is 5, 15, 25, 35, 45, 55, 65, 75, 85 and 95. The accuracy of the model is checked for each time-step. The results in Figure 3 clearly show that performance of the system is optimized when the input is submitted with '65' time-steps. This time step is selected for the proposed method.

In addition to that the optimization was checked by varying the learning rate through values 0.1 to 0.001. During training, the decrease in learning rate showed increase in accuracy of the detection process. The learning rate of the model is optimized and set to 0.001.

Any network intrusion detection checks a network for activities by a malicious individual that compromise the security of network.

The values that are required to create a confusion matrix are defined as; the true positive (TP) happen when illegitimate requests are correctly identified as malicious by the IDS, a true negative (TN) is the number of requests that are illegitimate but are not reported as malicious.

The false positive (FP) happens when legitimate network traffic is identified as malicious and a false negative (FN) occurs when illegitimate traffic goes undetected. The simulation results of IDSIoT-SDL have 1775 TPs, 212 TNs, 12 FPs and 7 FNs. The confusion matrix given in Tables 2 and 3) summarizes this information.

In order to evaluate the IDSIoT-SDL comparatively, the performance is shown in terms of *Precision* and *Sensitivity*. Nobakht et al. [19] have used the terms *Precision* and *Recall* to refer to Positive Productive Power*(PPP)* and *Sensitivity*, respectively. As can be seen from table (Table 4) *Precision* is given by $TP/(TP+FP)$, and *Sensitivity/Recall/Detection Rate (DR)*, given by $TP/(TP+FN)$ are valued at 0.9932 and 0.9961, respectively. The work in Ref. [19] on the other hand

**TABLE 1** Distribution of normal traffic and different attack types

| Traffic | Count |
|---|---|
| UDP DoS | 940 |
| HTTP Credential Brute Force | 5907 |
| Heartbleed | 5898 |
| Port Scan | 16,500 |
| ICMP Flood | 4480 |
| SSH Brute Force | 2006 |
| Normal | 17,899 |

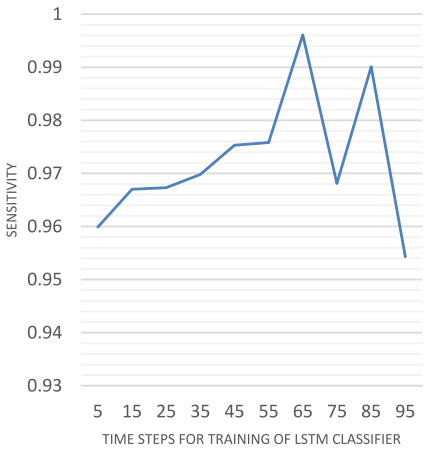



**FIGURE 3** Sensitivity of IDSIoT-SDL for various time steps

**TABLE 2** Accuracy measures for performance of IDSIoT-SDL

| | Positive | Negative |
|---|---|---|
| True | 1775 | 212 |
| False | 12 | 7 |
| $N = 2006$ | | |

Abbreviations: IDS, intrusion detection system; IoT, Internet of Things.

have a *PPP/Precision* of 98.32%, which is little lesser as compared to our system, that is, 99.32%. In [19], the *Sensitivity/Recall/DR* has value of 95.94%, which is also lesser as compared to 99.61% of our system. The comparison of the two systems is given in Table 4.

Additionally, the performance of the proposed system has also been evaluated in terms of *Sensitivity or Recall or DR* and *False Positive Rate/False Alarm Rate (FPR/FAR)*, for the purpose of comparison with [20] which has used *FPR/FAR* as an evaluation metric. For a binary classifier, FPR/FAR == (1 −Specificity) == (false detections)/(all detections)

And is given by $FPR/FAR = FP/(TN + FP)$

The performance of IDSIoT-SDL in terms of Sensitivity/Recall/DR and False Alarm Rate FAR is given as:

Sensitivity/Recall/DR = 99.61% and FAR = 0.53%

For evaluating the performance of the proposed model, the proposed detection module was compared with Ref. [20]. The results of this comparison are presented in Table 5. As seen in Table 5, IDSIoT-SDL offers better DR and a better FAR.

The authors in [21] have proposed a method called KALIS and have used *DR* and *Accuracy* metrics. The DR of IDSIoT-SDL is 99.61% which is much better than the DR of work proposed by authors in KALIS [21], that is, 91%; however, the accuracy of our system is 99.05% and that of method proposed in Ref. [21] is perfect 100%. The comparison is shown in Table 6

IDSIoT-SDL results are also compared with the IDS for IoT introduced in [3]. The authors have used suppressed fuzzy clustering (SFC) and principal component analysis (PCA) to devise a lightweight IDS for IoT. The comparison of with IDSIoT-SDL is shown in table (Table 7)

The results of the proposed mechanism are also compared with work done in Ref. [18] which introduces an lightweight IDS for IoT, that is, lightweight intrusion detection method combined with Fuzzy C Means (FCM) algorithm and PCA algorithm. The comparison between IDSIoT-SDL and FCM-PCA is shown in Table 8.

The comparison of accuracy of IDSIoT-SDL against the above discussed related algorithms is described graphically in Figures 4 and 5 and represented in Table 9.

**TABLE 3** Confusion matrix (performance of IDSIoT-SDL)

| Measure | Calculation | Values |
|---|---|---|
| Prevalence | $\frac{TP+FN}{N}$ | 0.8883 |
| Overall diagnostic power | $\frac{FP+TN}{N}$ | 0.1116 |
| Correct classification rate | $\frac{TP+TN}{N}$ | 0.9905 |
| Sensitivity(TPR) | $\frac{TP}{TP+FN}$ | 0.9961 |
| Specificity(TNR) | $\frac{TN}{FP+TN}$ | 0.9464 |
| False positive rate | $\frac{FP}{FP+TN}$ | 0.0536 |
| False negative rate | $\frac{FN}{TP+FN}$ | 0.0039 |
| Positive predictive power (Precision) | $\frac{TP}{TP+FP}$ | 0.9932 |
| Negative predictive power/value | $\frac{TN}{FN+TN}$ | 0.9680 |
| Misclassification rate | $\frac{FP+FN}{N}$ | 0.0094 |
| Odds-ratio | $\frac{TP*TN}{FN*FP}$ | 4437 |
| False discovery rate | FP/(FP + TP) | 0.0067 |
| Accuracy | (TP + TN) / N | 0.9905 |
| F1 score | 2TP / (2TP + FP + FN) | 0.9947 |
| Mathews correlation coefficient | TP * TN − FP * FN / sqrt((TP + FP) * (TP + FN) *(TN + FP) * (TN + FN)) | 0.9519 |

Abbreviations: FN, false negative; FP, false positive; IDS, intrusion detection system; IoT, Internet of Things; TN, true negative; TP, true positive; TNR, true negative rate; TPR, true positive rate.

**TABLE 4** Comparison of IDSIoT-SDL with [19]

| | PPP/Precision | Sensitivity/Recall/DR |
|---|---|---|
| IoT-IDM | 0.9853 | 0.9594 |
| IDSIoT-SDL | 0.9932 | 0.9961 |

Abbreviations: DR, detection rate; IDS, intrusion detection system; IoT, Internet of Things; PPP, Positive Productive Power.

**TABLE 5** Comparison of IDSIoT-SDL with AIS [20]

| Detection Mechanism | DR (%) | FAR (%) |
|---|---|---|
| AIS | 96.49 | 3.51 |
| IDSIoT-SDL | 99.61 | 0.53 |

Abbreviations: AIS, artificial immune system; DR, detection rate; FAR; false alarm rate; IDS, intrusion detection system; Iot, Internet of Things.

**TABLE 6** Comparison of IDSIoT-SDL with KALIS [21]

| Detection Mechanism | DR (%) | Acc (%) |
|---|---|---|
| KALIS | 91 | 100 |
| IDSIoT-SDL | 99.61 | 99.05 |

Abbreviations: IDS, intrusion detection system; IoT, Internet of Things

**TABLE 7** Comparison of IDSIoT-SDL with SFC-PCA [3]

| Detection Mechanism | DR (%) | FAR |
|---|---|---|
| SFC-PCA | 97.4 | 1.5 |
| IDSIoT-SDL | 99.61 | 0.53 |

Abbreviations: DR, detection rate; FAR; false alarm rate; IDS, intrusion detection system; Iot, Internet of Things; PCA, principal component analysis; SFC, suppressed fuzzy clustering.

**TABLE 8** Comparison of iDSIoT-SDL with FCM-PCA [18]

| Detection Mechanism | DR (%) | FPR (%) |
|---|---|---|
| FCM-PCA | 96.8 | 1.6 |
| IDSIoT-SDL | 99.61 | 0.53 |

Abbreviations: DR, detection rate; FCM, Fuzzy C Means; FPR; false positive rate; IDS, intrusion detection system; Iot, Internet of Things; PCA, principal component analysis.

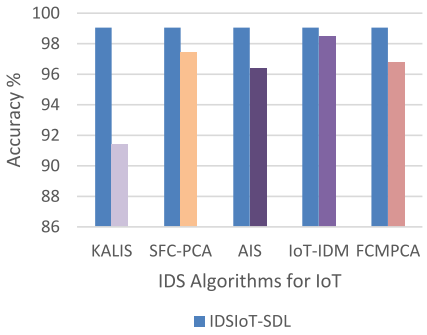



**FIGURE 4** Comparison of IDSIoT-SDL with other IoT-IDS algorithms based on accuracy or detection rate

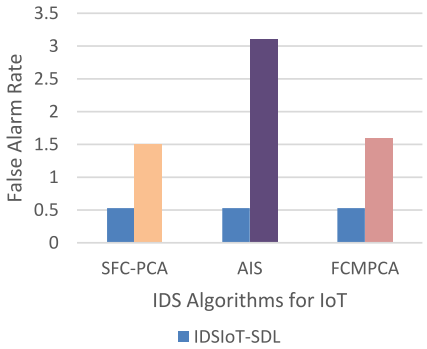



**FIGURE 5** Comparison of IDSIoT-SDL with other IoT-IDS algorithms based on false alarm rate

**TABLE 9** Comparison of accuracy of IDSIoT-SDL(99.05) with other IoT-IDS

| KALIS | 91.4 |
|---|---|
| SFC-PCA | 97.4 |
| AIS | 96.4 |
| IoT-IDM | 98.5 |
| FCM-PCA | 96.8 |

Abbreviations: AIS, artificial immune system; FCM, Fuzzy C Means; FPR; false positive rate; IDS, intrusion detection system; IoT, Internet of Things; PCA, principal component analysis; SFC, suppressed fuzzy clustering.

## 8 | CONCLUSION AND FUTURE WORK

This work is novel because it uses SDN and deep learning to design the intrusion detection for IoT traffic. SDN provides a smart management of networks by decoupling of control and data planes. The deep learning has proved to be a better approach in almost all areas. The deep learning based classifiers are providing better results in current IDS as compared to traditional classifiers. The proposed model detects any intrusion in networking systems, in particular IoT networks. The performance of the proposed model is evaluated using F1, Precision, Recall, Accuracy and other metrics. The results for the proposed model have shown a notable improvement over other intrusion detection models for IoT. Considering future work, the other deep learning classifiers can be explored for improvisation. The simulation work of the proposed model can be tested on a real environment with increased attack and normal traffic.

## ACKNOWLEDGEMENT

The authors are grateful to MANF UGC, Government of India, for providing financial support under MANF-UGC (MANF-2015-17-JAM-60,506) programme to carry out this work.

## CONFLICT OF INTEREST

There is no conflict of interest regarding the publication of this paper.

## REFERENCES


1. Li, S., Da Xu, L., Zhao, S.: 5G Internet of Things: a survey. J. Ind. Inf. Integr. 10, 1–9 (2018)
2. Russell, B., Van Duren, D.: Practical Internet of Things Security., Packt Publishing (2016). https://www.oreilly.com/library/view/practical-internet-of/9781785889639/
3. Liu, L., et al.: An intrusion detection method for internet of things based on suppressed fuzzy clustering: EURASIP J. on Wirel Comm. and Netw. 2018(1), (2018). http://doi.org/10.1186/s13638-018-1128-z
4. Qushtom, H.: Enhancing the QoS of IoT networks with lightweight security protocol using Contiki OS. Int J Comput Netw Infor Secur. 9(11), 27–35 (2017)
5. Khan, Z.A., Herrmann, P.: Recent advancements in intrusion detection systems for the internet of things. Secur. Commun. Netw. 2019 (2019)
6. Sultana, N., et al.: Survey on SDN based network intrusion detection system using machine learning approaches. Peer-to-Peer Netw. and Appl. 12(2), 493–501 (2019). https://doi.org/10.1007/s12083-017-0630-0
7. Wani, A., Revathi, S.: Analyzing threats of IoT networks using SDN based intrusion detection system (SDIoT-IDS). Commun. Comput. Inf. Sci. 828, 536–542 (2018)
8. Valdivieso Caraguay, ÁL., et al.: SDN: Evolution and Opportunities in the Development IoT Applications. Int. J. Distrib. Sens. Netw. 10(5), 735142(2014). https://doi.org/10.1155/2014/735142
9. Kiani, F.: A survey on management frameworks and open challenges in IoT. Wirel. Commun. Mob. Comput. 1–33 (2018)
10. Da Xu, L., He, W., Li., S.: Internet of things in industries: a survey, IEEE Trans. Ind. Informat. 10(4), 2233–2243 (2014)
11. Lu, Y., Da Xu, L.: Internet of things (IoT) cybersecurity research: a review of current research topics. IEEE Internet Things J. 6(2), 2103–2115 (2019)
12. Khan, M.A., Salah, K.: IoT security: review, blockchain solutions, and open challenges. Futur. Gener. Comput. Syst. 82, 395–411 (2018)
13. Azka, W., Revathi, S.: Protocols for secure Internet of Things. Int. J. Educ. Manag. Eng., 7(2), 20–29 (2017)
14. Azka., Revathi, S., Geetha.: A survey of applications and security issues in software defined networking. Int J of Comput Netw and Inform Secur. 9(3), 21–28 (2017)
15. Li, J., Altman, E., Touati, C.: A General SDN-based IoT Framework with NVF Implementation. ZTE Communications, ZTE Corporation, 13, pp. 342–45. HAL (2015)
16. Vilalta, R., et al.: End-to-End SDN Orchestration of IoT Services Using an SDN/NFV-enabled edge node. In: Optical fiber Conference and Exhibition (OFC), pp. 7–9. (2016)
17. Kalkan, K., Zeadally, S.: Securing internet of things with software defined networking. IEEE Communications Magazine, vol. 56(9), pp. 186–192 (2018). https://doi.org/10.1109/mcom.2017.1700714
18. Deng, L., et al.: Mobile network intrusion detection for IoT system based on transfer learning algorithm. Cluster Comput. 1–16. (2018)
19. Nobakht, M., Sivaraman, V., Boreli, R.: A host-based intrusion detection and mitigation framework for smart home IoT using openflow. 2016 11th International Conference on Availability, Reliability and Security (ARES), Salzburg, pp. 47–156 (2016)
20. Hosseinpour, A., et al.: An intrusion detection System for fog computing and IoT based logistic systems using a smart data approach an intrusion. Int J Dig Content Technol Appl. 10(5), 34–46 (2016)
21. Midi, D., et al.: Kalis – A system for knowledge-driven adaptable intrusion detection for the internet of things. 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, pp. 656–666. (2017)
22. Hassan, S., Houbing, A., Malik, K.M.: NBC-MAIDS: Naïve Bayesian classification technique in multi-agent system-enriched IDS for securing IoT against DDoS attacks. J Supercomput, 74(10), 5156–5170 (2018)
23. Kumar, V., Das, A.K., Sinha, D.: UIDS: A unified intrusion detection system for IoT environment. Evol. Intell. 0123456789 (2019)
24. Deogirikar, J.: Vidhate, A.: Security Attacks in IoT: A Survey. 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam 32–37 (2017)
25. Mishra, P. et al.: A detailed investigation and analysis of using machine learning techniques for intrusion detection. IEEE Commun. Surv. Tutor. 21(1), 686–728 (2019). https://doi.org/10.1109/comst.2018.2847722
26. Yuxin, D., Siyi, Z.: Malware detection based on deep learning algorithm. Neural Comput. Appl. 1 (2017)
27. Kwon, D., et al.: A survey of deep learning-based network anomaly detection. Cluster Comput. 22(S1), 949–961 (2019). http://doi.org/10.1007/s10586-017-1117-8
28. Cakir, B., Dogdu, E.: Malware Classification Using Deep Learning Methods. Association for Computing Machinery, New York, NY, USA (ACMSE '18) (10), 1–5 (2018)
29. Srivastava, P.: Essentials of deep learning Introduction to long short term memory. (2017) https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/. Accessed 19 Sept 2018
30. Olah, C.: Understanding LSTM Networks -- colah's blog. (2015) http://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed 19 Sept 2018
31. Goransson, P., Black, C., Culver, T.: Software Defined Networks: A Comprehensive Approach. Elsevier Science, (2016)
32. Wireshark · Go Deep. [Online]. https://www.wireshark.org/ (2019). Accessed 22 Jan 2019
33. Ajaeiya, G.A., Ids, A.B. F.: Flow-Based Intrusion Detection System for SDN. (2017)
34. Hindy, H., et al.: A taxonomy and Survey of Intrusion detection System design Techniques. Netw. Threats Datasets. 8, 104650–104675 (2020). https://doi.org/10.1109/access.2020.3000179
35. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new Intrusion detection dataset and Intrusion Traffic characterization. 108–116 (2018)
36. Berman D., et al.: A survey of deep learning methods for cyber security. Information. 10(4), 122 (2019). https://doi.org/10.3390/info10040122
37. Fontes, R.R., Afzal, S., Brito, S. H., Santos, M. A., Rothenberg, C.E.: Mininet-WiFi: Emulating software-defined wireless networks. In: Proceedings of 11th International Conference on Network and Service Management CNSM 2015, pp. 384–389, IEEE, Barcelona, 9 November (2015)
38. Heller, B.: OpenFlow Switch specification 1.0.0. Current. 1–36 (2009)


**How to cite this article:** Wani, A., Revathi, S., Khaliq, R.: SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL). CAAI Trans. Intell. Technol. 6(3), 281–290 (2021). https://doi.org/10.1049/cit2.12003