IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

ML-LGBM: A Machine Learning Model based on Light Gradient Boosting Machine for the Detection of Version Number Attacks in RPL-Based Networks

MUSA OSMAN[1],JINGSHA HE[1], FAWAZ MAHIUOB MOHAMMED MOKBAL[1,2],NAFEI ZHU[1], AND SIRAJUDDIN QURESHI[1]

[1]Department of Software Engineering, Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China.
[2]Faculty of Computer Science, ILMA University, Karachi, Pakistan.

Corresponding author: Nafei Zhu (e-mail: znf@bjut.edu.cn).

ABSTRACT Internet of Things (IoT) has caused significant digital disruption to the future of the digital world. With the emergence of the 5G technology, IoT would shift rapidly from aspirational vision to real-world applications. However, one of the most pressing issues in IoT is security. Routing protocols of the IoT, such as the Routing Protocol for Low-power and lossy network protocol (RPL), are vulnerable to both insider and outsider attacks with the insider ones being more challenging because they are more difficult to detect and mitigate. Among the most concerning insider attacks to RPL in IoT applications is the Version Number Attacks (VNAs) that target the global repair mechanisms by consuming resources of IoT devices, such as power, memory, and processing power, to eventually cause the IoT ecosystem to collapse. In this paper, a lightweight VNA detection model named ML-LGBM is proposed. The work on the ML-LGBM model includes the development of a large VNA dataset, a feature extraction method, an LGBM algorithm and maximum parameter optimization. Results of extensive experiments demonstrate the advantages of the proposed ML-LGBM model based on several metrics, such as accuracy, precision, F-score, true negative rate and false-positive rate of 99.6%, 99%, 99.6%, 99.3% and 0.0093, respectively. Moreover, the proposed ML-LGBM model has slower execution time and less memory resource requirement of 140.217 seconds and 347,530 bytes, making it suitable for resource-constrained IoT devices.

INDEX TERMS IoT, RPL Protocol, 6LowPAN, Machine Learning, RPL Attacks.

I. INTRODUCTION

WITH the rapid advancement of our modern society, the Internet of things (IoT) technology is poised to be the future major digital revolution, particularly with the recent 5G revolution, which makes this technology move quickly from aspirational dreams to practical applications. Internet of Things (IoT) allows the modern world to be more effective, saving time and resources for businesses and individuals. Nowadays, such cutting-edge technologies significantly impact our lives, such as smart homes, smart cities, smart healthcare, and even wearable devices. Furthermore, it has a significant impact on saving time and resources in industry fields [1], [2], [3].

By allowing people and intelligent devices to communicate at anytime, anywhere, the billions of physical devices around the world that are now and in the future can communicate via the Internet without human intervention and controlled through actuators [4], [5], [6]. The sensors capture critical data from homes (smart home devices), industrial units and smart cities, etc., and can be shared [7]. This precious treasure trove of big data transmitted over the Internet generates rapacity and hacking to seize it. Therefore, the main challenge in IoT is cybersecurity. Many IoT devices don't think of security fundamentals like encrypting data in transit and at rest due to their resource limitations. IoT software flaws are also discovered regularly. Consequently, researchers seek to devise alternative defiance mechanisms that are suitable for IoT to try to protect the network protocol vulnerable to attacks.

The primary routing protocol used by IoT devices to provide communication between sensors and actuators is the distance vector (DV) routing protocol, aka. IPv6 Routing Protocol

for Low Power and Lossy network (RPL) [6], [8]. The RPL protocol organizes the low power and lossy network nodes as a Destination Oriented Directed Acyclic Graph (DODAG) with a unique DODAGID. Two or more DODAGs form together an instance of RPL, which possesses a unique RPLinstanceID. An RPL network node can be a member of multiple instances but must only belong to one DODAG at any instance [6], [9].

The RPL protocol has distinct features like auto-configuration, self-healing, loop-avoidance, and detection in addition to the control messages used to construct the DODAG, such as ICMPv6 control messages. There are four types of control messages including (i) DODAG Information Object (DIO) responsible for building the upward route from the leaves to the sink node, (ii) Destination Advertisement Object (DAO), which constructs the reverse path from the sink node to the leaves, (iii) DODAG Information Solicitation (DIS) that is used to solicit the DIO when it has not been received for a while, and (iv) Destination Advertisement Object Acknowledgment (DAO-ACK) which is sent as a result of receiving a DAO message. The DIO messages contain many fields, including RPLinstanceID, the version number of the DODAG, the node's rank, and the unique DODAGID [6], [10].

Version Number Attacks (VNA) are among the most threatening attacks targeting the RPL network availability by maliciously increasing the version number. In such attacks, the malicious node increases the version number when it receives a DIO message in the version number field maliciously, leading to an inconsistency in the DODAG. As the result, the global repair mechanism is triggered by the root to rebuild the DODAG properties from scratch [11], [12]. Thus, the attack drains the network resources, affecting network availability, quality-of-service (QoS), and the life of network.

One of the most significant challenges in the real world of IoT is a secure connection that is challenging due to the heterogeneity of IoT devices. To the best of our knowledge, only a few studies have targeted at the security of VNA attacks in the literature. However, most of such studies have not yet been evaluated, while some are resource-consuming with respect to power, memory, and processing capability with a high false-positive rate [13], [14].

Machine-learning (ML) techniques have demonstrated their efficiency in the cybersecurity domain, which can be applied to detecting anomalies with a high positive rate. Moreover, ML can handle a tremendous amount of data, making it a potentially suitable method for massive data provided by the network of sensors. Therefore, this study proposes an ML method for IoT networks with the aim of performing the function of an Intrusion Detection System (IDS) to detect VNA in the RPL-based network using the ensemble learning technique. The proposed IDS (ML- LGBM) adopts the Light Gradient Boosting Machine with extreme parameter optimization. An extensive and unique VNA dataset constructed depends on RPL properties and a proposed feature extraction technique.

In addition, an extensive performance evaluation and comparison of the proposed IDS (ML-LGBM) to different well-known machine learning algorithms were performed. The experimental results demonstrate that the presented IDS results are impressive and outperform previous results with lower false-positive rate, shorter training and testing time, smaller model size, and higher accuracy, precision, and F1-score. The main contributions of this paper can be summarized as follows:

- A novel model is proposed using a light gradient boosting machine trained on a unique dataset with extreme parameter optimization characterized by high accuracy, high detection rate, and low computational complexity.
- An extensive VNA dataset that consists of 1,050,861 records have been constructed. The dataset is uniquely built depending on RPL properties, such as the version number and rank fields.
- A feature extraction technique is proposed to extract and provide training and testing datasets to feed the ML-LGBM model dynamically.
- Feature engineering analysis using a step forward feature selection approach is applied to deriving the optimal subset of features, resulting in reducing the dimensionality of the dataset and the size of the model simultaneously and making the proposed model very efficient and suitable for IoT.
- An in-depth evaluation of ML-LGBM has been performed based on various performance evaluation metrics, including training time, testing time, model size, etc.

The rest of the paper is organized as follows. Section 2 reviews some related work. Section 3 describes the proposed method in details. Section 4 presents the experimental results and analysis. Section 5 concludes this paper.

II. RELATED WORK

Research on securing IoT devices from internal attacks is still in its infancy despite the number of studies on detecting and mitigating these kinds of attacks. However, RPL-based networks are still suffering from VNA attacks, which consumes the RPL-based network resources and threatens its availability.

As a preliminary work, Mayzaud et al. [15] studied the effect of VNA in an RPL-based network using a simulation with 20 nodes and discovered that control overhead could increase up to 18 times while the delivery ratio of packets would decrease by 30%. Furthermore, the location of the attacker affects the consistency of the network. Another study by Aris et al. [12] provided a lightweight technique to mitigate the impact of VNA in an RPL-based network. The proposed method used two types of procedures to reduce the effects of version number attacks. The first is named elimination in which a node would eliminate any updates coming from a leaf node. The second is called shield in which a node changes the version number depending on its neighbors with a better rank. The authors claimed that the proposed model could

reduce the delay, control message overhead and data packet delivery ratio up to 87%, 63%, 71% and 86%, respectively. Yavuz et al. [16] generated a dataset using Cooja emulation based on the Contiki operating system. The dataset contains three types of RPL attacks, i.e., hello flood, decreased rank, and VNA. The authors also applied a deep neural network model to detect the attacks mentioned above that can achieve accuracy of 94.9% for the decreased rank attack, 99.5% for the hello flood attack, and 95.2% for the VNA. The main drawback of the proposed model is in the long training time, however. In a further study by Kfoury et al. [17], the authors use the Self-Organizing Map (SOM) technique to detect VNA and other attacks. There was no clear indications on the placement of the IDS as well as its power consumption in this study. Dvir et al. [18] presented an IDS for combating the VNA and Rank Attack (RA) named VeRA (Version number and Rank Attack) using a hash function. The main idea is that the DODAG root is responsible for generating a hash chain based on random numbers. The main drawback of the proposed IDS is that it consumes network resources. Sahay et al. [19] investigated the VNA and its impact on the RPL-based network and then presented a centralized machine learning technique to analyze the data captured from the network. The performance of the proposed model, as the authors claimed, was 98%. Table 1 presents a summary of some related work along with the limitations.

III. THE PROPOSED METHODOLOGY

This paper proposes a machine learning model called ML-LGBM to detect VNA in an RPL-based network. The model consists of four modules that work together to achieve high accuracy and detection rate. These four modules are data collection, data preprocessing, feature selection, and machine learning, which is depicted in Figure 1.

A. MODEL SETUP

To carry out the experiment, we used the Cooja simulator bundled with the Contiki operating system. Cooja simulator is a Java-based simulator that permits the emulation of real hardware platforms with its core written using the C language [25]. We used Contiki instance 3.0 on a virtual machine with 4 GB of RAM and 40 GB of hard disk to collect the raw data. Each node is created as a Zolertia mote consisting of 8KB RAM and 92KB flash memory in the experiment. The attacker node starts after 30 seconds of running the emulator. Table 2 shows the dataset generated from the simulation and Figure 2 shows one scenario of the simulation. Furthermore, to collect the raw data, we used the 6Lowpan Package Analyzer already built into the simulator. This tool is used to sniff the radio traffic and save it as a PCAP file.

B. NETWORK SCENARIO

1) Network parameters

In the experiment, the RPL-UDP example was used for all the scenarios which contained two nodes, one udp-server and udp-client, and in each design, the udp-server is the sink node. One sink node was used in the network in our experiment along with (10, 20) client nodes and (1, 2, 3) malicious nodes. The memory of Zolertia (Z1) node was very small, so we selected 10 minutes as our standard time to eliminate the simulation hanging. Table 3 shows the Cooja parameters used in the simulation phase.

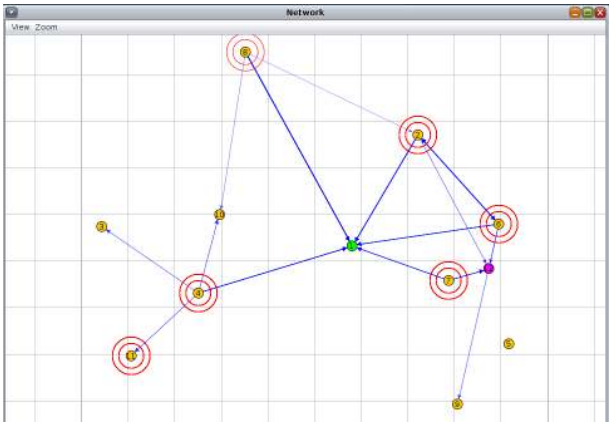


FIGURE 2. **One Cooja simulation scenario.**

TABLE 3. **Cooja simulation parameters.**

Parameters	Values
Operating system	Contiki 3.0
Emulator/simulator	Cooja
Node type	Zolertia (Z1)
Routing protocol	RPL
Radio environment	Unit Disk Graph Medium (UDGM)
Simulation duration	10 minutes
Node transmission rate	50m

2) Simulation of VN attacks

As mentioned above, the Cooja core is written in the C language. In a VNA, when the malicious node receives a DIO message in the VN attack, it increments the version number field or attribute and sends it to its neighbors. As the result, loops may occur, causing the network to become inconsistent and resulting in the root node or the sink node to trigger the global repair mechanism to restore the network properties. To originate the VNA, we edited the Cooja file system to make the malicious node increment the version number in our case by one each time it receives a DIO message. Algorithm 1 declares the VNA.

Algorithm 1: The VNA Scenario

Input: DIO message

Output: DIO message (with increased VN)

1: **for** each DIO broadcasting message **do**
2: **if** (node-id = attacker-node-id) **then**
3: $received\ the\ DIO\ message$
4: $extract\ the\ DAG\ information$
5: $Version\ Number = Version\ Number + 1$

TABLE 1. Summary of some related work and limitations.

Ref	Dataset	Methodology	Results	Limitations
[16]	Self-generated dataset	A deep learning method for detecting RPL routing attacks	Precision is 0.94%, recall is 0.94% and F1 Score is 0.95%	Deep learning methods are not suitable for IoT
[20]	Self-generated dataset	Statistical and machine learning techniques	Decision trees are 94.07% and Artificial neural network accuracy is 93.99%	The authors just evaluated the IDS performance
[21]	Self-generated dataset	Random forest for detecting RPL attacks	Classification accuracy is 99.33%	Random forest is resource extensive
[22]	Self-generated dataset	An ensemble learning technique for detecting seven RPL attacks	Classification accuracy is 94.5%	Ensemble learning techniques are resource extensive
[23]	-	Gated recurrent unit network model-based deep learning is proposed	Accuracy is 99.96%	Scalability problem when the number of nodes increases
[12]	-	A lightweight technique to mitigate the impact of VNA	It reduced delay by 87%, control message overhead by 63%, and improved data packet delivery ratio up to 71%	It doesn't incorporate mobility
[18]	-	Trust-based IDS	VeRA prevents attack from occurring	It consumes network resources
[15]	-	In-depth study of VNA	Control overhead is increased while delivery ratio of packets is decreased	
[24]	-	Self-Organizing Map	It clusters the attacks and normal traffic	Deployment overhead is not considered

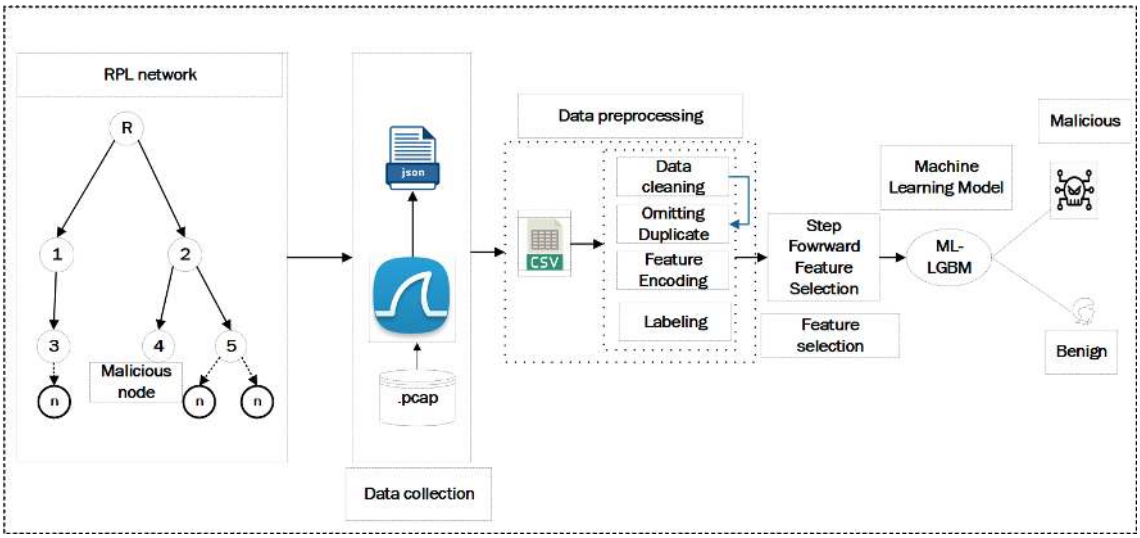


FIGURE 1. The proposed model for detecting VNA.

TABLE 2. The datasets from different scenarios.

Scenarios	No of Benign nodes	No of malicious node(s)	Benign	Malicious	The total
One malicious	10	1	223243	89296	312539
Two malicious	10	2	286965	27422	314387
Three malicious	20	3	374653	374653	374653

6: *send the updated DIO to the neighbors node* 7: **else if** node-id $\neq$ attacker-node-id **then**

8: $regular\ DAG\ version$
9: **end if**
10: **end for**

3) Raw data collection

After implementing our scenario, the radio message tool was used to sniff and collect the radio messages transmitted between the nodes that would be analyzed with a 6Lowpan analyzer with a PCAP feature incorporated into the tool. Figure 3 presents a sample of the collected data. Subsequently, the collected data was processed by the Wireshark software and saved as JavaScript Object Notation (JSON).

4) Dataset creation

We developed a Python feature extraction model to extract the JSON file features, which was introduced to get the proper feature vectors. The main goal of this model is to dynamically extract the essential characteristics of both benign and malicious values from raw data to be saved in a CSV file. Algorithm 2 declares the parsing procedure. As the result, the total number of extracted features are 113. We then performed the data cleaning process to delete duplicate features, resulting in the total number of features to be reduced to 59. Subsequently, the fixed features (with values not changing like WPAN security) were removed from the dataset. We also cleaned up the other features with hexadecimal values, e.g., frame checksum. As the result, the remaining features after this process were only 17. Then, categorical features were encoded, e.g., source IP address and destination IP address. Moreover, we encoded the target label to [0, 1] for regular and malicious traffic, respectively. The dataset contained 1,050,861 instances of which 884.861 were benign and 166.000 were malicious. Figure 4 shows an example of the candidate features.

Algorithm 2: Parsing JSON codes

Input: JSON file
Output: CSV file

Initialisation :
$processed\_data \leftarrow [\,]$
$header \leftarrow [\,]$
$reduced\_item \leftarrow \{\,\}$
$data\_to\_be\_processed \leftarrow [\,]$
//Reading arguments
$raw\_data \leftarrow json.loads(json\_value)$
//LOOP Process
1: **for** item in data_to_be_ processed **do**
$header+ = reduced\_item.keys()$
$reduce\_item(node, item)$
$processed\_data.append(reduced\_item)$
2: **end for**
//Writing arguments
Open csv file
header $\leftarrow list(set(header))$
write $\leftarrow csv.DictWriter(header)$
LOOP Process
3: **for** row in processed_data: **do**
$writer.writerow(row)$
4: **end for**
5: **return** $CSVFile$

5) Feature normalization

Feature normalization or feature scaling is the process of normalizing the distribution of the independent variables. The collected data have different mean and variance, which decreases the performance of the machine learning model. We thus performed min-max scaling or normalization to normalize the collected data, which scales the feature in a new dimension between [0, 1] or [1, -1]. The min-max scaler is defined using Equation 1.

$$X^{'} = \frac{X - min(x)}{max(x) - min(x)}. \tag{1}$$

C. FORWARD FEATURE SELECTION

Also known as Sequential Forward Selection (SFS), forward feature selection selects a subset of features based on an iterative method from a complete set of features to minimize the classification error. Starting from an empty feature set S, it iteratively evaluates the features and selects the one with the best performance depending on some estimation function to minimize the mean square error (MSE) [26]. At each iteration, it tests all potential combinations of the selected feature with the residual features and holds over the pair that produces the best performance among them. A major advantage of SFS is that it assesses the utility of a subset of features by training a model on it. It also uses cross-validation for the evaluation of a subset of features. It can always provide the best subset of features [27], [28]. The SFS is defined through Equation 2. Table 4 and Figure 5. offer the selected features through using the SFS algorithm.

$$X^{+} = argmax_{x^{+} \in Y_k}[J(Y_k + X^{+})]. \tag{2}$$

Where Y_k = $\emptyset$,$k = 0$ is an empty set, and X^{+} represents the selected features.

TABLE 4. The selected features (SFS).

NO	Selected features	Feature name
1	$6lowpan.src$	Source IP
2	$6lowpan.Dst$	Destination IP
3	$iphc.m$	IP Header Compression
4	$frame.len$	Frame length
5	$dio.dtsn$	DSTN number
6	$dio.rank$	DIO rank
7	$dio.version$	DIO version
8	$dao.sequence$	DAO sequence
9	$ipv6.hlim$	IPv6 hope limit
10	$ipv6.plen$	IPv6 packet length
11	$wpan.seq_no$	6Lowpan sequence number

D. THE LIGHT GRADIENT BOOSTING MACHINE MODEL

Developed by Microsoft in 2016, LightGBM is a fast, distributed, open-source and high-performance gradient boost-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3087175, IEEE Access

No.	Time	Source	Destination	Protocol	Length	Info
	0.000000	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.003288	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.006552	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.009815	fe80::c30c:0:0:5	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.009848	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.013103	fe80::c30c:0:0:5	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.013121	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.016367	fe80::c30c:0:0:5	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.016380	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.019663	fe80::c30c:0:0:5	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)
	0.019674	fe80::c30c:0:0:2	ff02::1a	ICMPv6	64	RPL Control (DODAG Information Solicitation)

FIGURE 3. **A sample of the collected data.**

Index	time_delta	6lowpan.src	6lowpan. Dst	iphc.m	iphc.sac	frame.len
0	0.001	2	2	1	1	50
1	0.001	12	12	1	1	50
2	0	12	999	1	0	97
3	0	13	999	1	0	97
4	0.001	11	6	0	0	76
5	0.001	10	999	1	0	97
6	0.001	8	10	0	0	102

FIGURE 4. **A candidate feature as a result of feature extraction.**

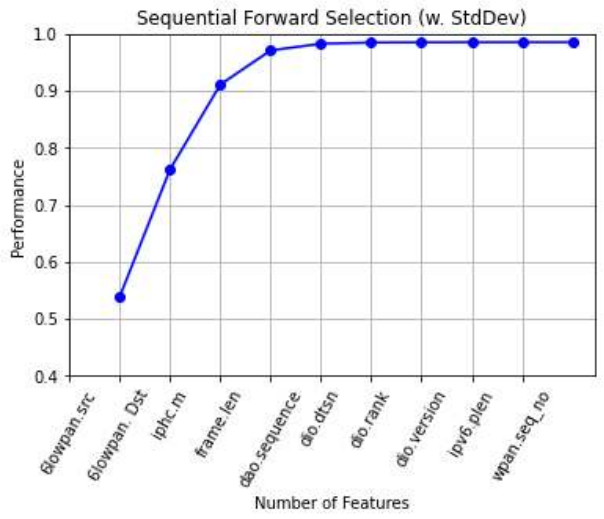


FIGURE 5. **Selected features and their performance.**

ing machine learning algorithm. It uses histogram-based algorithms to speed up training and to reduce memory usage. LightGBM is highly efficient and accurate, supports parallel learning, and is convenient with large datasets, making it suitable for Low power Lossy Network (LLN) [28] . Additionally, LightGBM contains two techniques that work together: Gradient-based One Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), which overcome the shortcomings of the histogram-based algorithm used in all GBDT (Gradient Boosting Decision Tree) frameworks [27]. In GOSS, different data instances play different roles in calculating information gain in which an instance with greater gradients can add more to the information gain. GOSS keeps those instances with high gradients and drops those with limited gradients at random to maintain the precision of information gain estimation [29]. The mathematical analysis in GOSS is shown in Equation 3.

$$\hat{V}_j(d) = \frac{1}{n}\left(\frac{(\sum_{x_i\in A_l} g_i + \frac{1-a}{b}\sum_{x_i\in B_l} g_i)^2}{n_l^j(d)} + \frac{(\sum_{x_i\in A_r} g_i + \frac{1-a}{b}\sum_{x_i\in B_r} g_i)^2}{n_r^j(d)}\right) \tag{3}$$

Where $\hat{V}_j(d)$ estimated variance gain over the subset $A \cup B$, $A_l = \{x_i \in A : x_{ij} \leq \text{d}\}$, $A_r = \{x_i \in A : x_{ij} > \text{d}\}$, $B_l = \{x_i \in B : x_{ij} \leq \text{d}\}$, $B_r = \{x_i \in B : x_{ij} > \text{d}\}$, and the coefficient $\frac{1-a}{b}$ is used to normalize the sum of the gradients over B back to the size of A^c. Thus, the estimated $\hat{V}_j(d)$ is used over a smaller instance subset, instead of the accurate $V_j(d)$ over all the instances to determine the split point. At the same time, The EFB technique is used by LightGBM to minimize the model complexity by bundle the exclusive features into a single feature.

The loss function in our model is the logistic regression calculated as Equation 4. This function is considered to be an optimal calibration statistic function for training our detector as it penalizes the discrepancy between true and expected odds. Besides, it estimates the relative uncertainty between the class that our system forecasts and the real classes.

$$Logloss = -\frac{1}{N}\sum_{i=i}^{N} y_i log(\hat{y}) + (1-y_i) log(1-\hat{y}). \quad (4)$$

Where i denote the given observation/record, y_i denotes the actual/true value, and $\hat{y}$ indicates the probability of prediction.

IV. EXPERIMENT AND ANALYSIS

A. PERFORMANCE EVALUATION METRICS

In this study, we use different evaluation metrics to evaluate the performance of ML-LGBM to detect VNA in an RPL-based network. These metrics are based on the confusion metrics results. The evaluation metrics include accuracy, precision, false negative, false positive, F-score, misclassification rate (error rate), and detection rate (DR) [30], [31], as shown in Equations 5-10.

$$Accuracy\ overall = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (5)$$

$$Precision = \frac{TP}{(TP+FP)} \quad (6)$$

$$False\ negative\ rate\,(FNR) = \frac{FN}{(FN+FP)} \quad (7)$$

$$Detection\ Rate\,(DR)\ or Recall = \frac{TP}{(TP+FN)} \quad (8)$$

$$FP\ Rate\ or\ Fall-out = \frac{FP}{(TN+FP)} \quad (9)$$

$$F-score = 2\left(\frac{(Recall \times Precision)}{(Recall+Precision)}\right) \quad (10)$$

B. RESULTS AND ANALYSIS

For the ML-LGBM model that is proposed to detect the VNA in an RPL-based network, we have performed a considerable analysis using Python version 3.7. The model was trained on the training dataset and validated using 10-fold cross-validation. Moreover, we performed extensive parameter tuning for the model to get the best results. Also, other machine learning classifiers are compared to the proposed model.

The results obtained from the experiment showed great promises in both performance and complexity, for the model achieved accuracy, precision, F-score of 99.6%, 99.0% and 99.3%, respectively. In terms of complexity, the model needed less training, resulting in short testing time and smaller model size of 140.217s and 347530 bytes, respectively, indicating that the proposed model has less complexity. Table 5 shows the overall results obtained from the experiment compared to other machine learning algorithms, such as Gradient Boosting (GB), Extra tree classifier (EXT), Random forest (RF), k-nearest neighbors (K-NN) and eXtreme Gradient Boosting (XGBoost), demonstrating that ML-LGBM has the best performance among all these classifiers. Table 6 shows the results obtained from different dataset scenarios (one malicious, two malicious, and three malicious nodes).

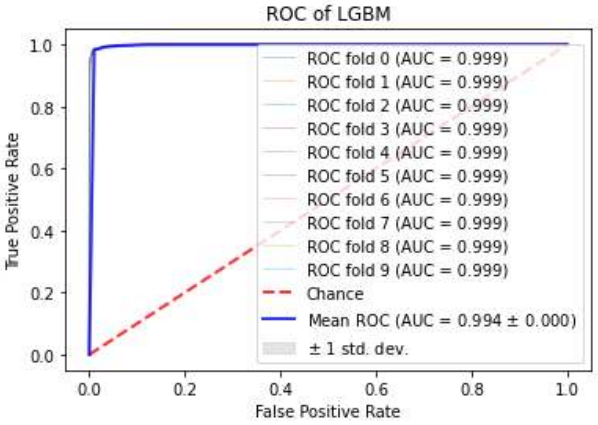


FIGURE 7. The ROC curve for ML-LGBM.

Figure 6 shows a comparison between ML-LGBM and XGBoost in which Figure 6 (a) and (b) show the performance of ML-LGBM and XGBoost. The first plot is the learning curve, which offers training vs. score. The second plot shows the model scalability depending on the time and the third plot shows the model performance and its score. Figure 7 shows the Receiver Operating Characteristic curve (ROC) for ML-LGBM. Furthermore, we calculated the training and testing time for both classifiers ML-LGBM and XGBoost, concluding that ML-LGBM has a better fitting time of 140.217s where XGBoost fitting time, i.e., the time required to train the models for each training size, of 208.752s. The results indicate that the proposed ML-LGBM has faster training speed, higher efficiency, lower memory usage and higher accuracy. Therefore, it is suitable for IoT applications.

Also, these figures indicate that LightGBM fit time scalability is less than XGBoost and can achieve better performance in less time. On the other hand, the proposed model is very efficient in true negative rate 0.990 and false-positive rate 0.009399 compared to XGBoost 0.977 for both true negative rate 0.022582 and false-positive rate. Moreover, from Table 7, ML-LGBM achieves a size of 347,530 bytes, which is adequate to fit into IoT devices, a small standard deviation of 0.00024, which indicates that the data points are close to the mean, and a log loss of 0.1289, which implies sufficient confidence (very low uncertainty) in the estimated probabilities.

Table 8 compares the effect of different estimator values on the training time, standard deviation, and accuracy acquired from parameter of the number of estimators. From the result,

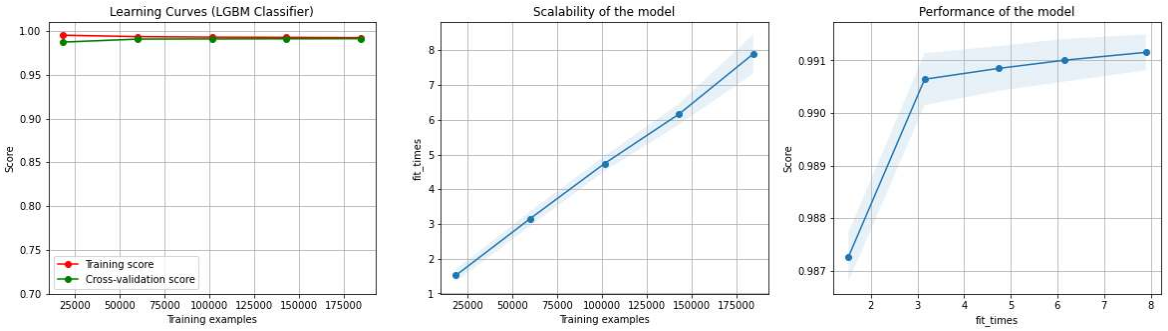
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3087175, IEEE Access

TABLE 5. Comparison of LGBM with other classifiers.

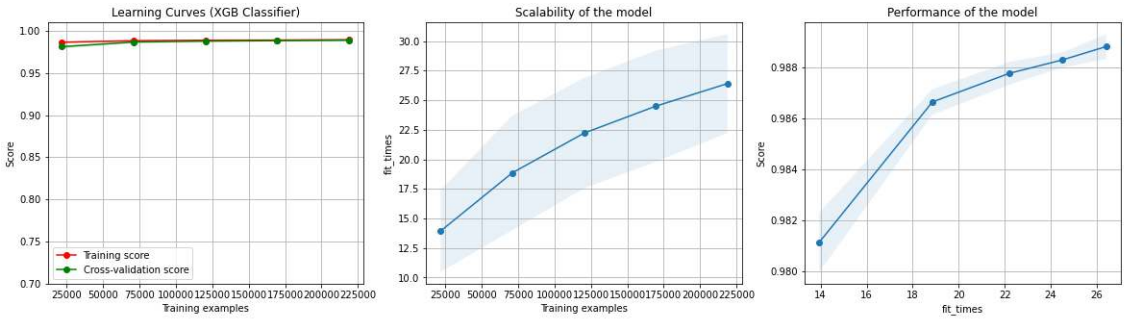
Scenario	Accuracy	Precision	Recall	$F1-score$	TNR	FPR
LGBM	0.996	0.990	0.996	0.993	0.990	0.009399
GB	0.990	0.980	0.985	0.983	0.980	0.019237
EXT	0.993	0.988	0.989	0.988	0.988	0.011307
RF	0.993	0.986	0.991	0.988	0.986	0.013933
K-NN	0.989	0.983	0.978	0.980	0.983	0.016574
XGBoost	0.988	0.977	0.983	0.980	0.977	0.022582

TABLE 6. The results obtained from the different dataset scenarios.

Model	No of malicious node(s)	Accuracy	Precision	Recall	F1_score	TNR	FPR
ML- LGBM	1	0.996	0.990	0.996	0.993	0.990	0.009399
	2	0.995	0.995	0.974	0.984	0.974	0.025823
	3	0.993	0.993	0.946	0.969	0.946	0.053489



(a) LGBM classifier performance.



(b) XGBoost classifier performance.

FIGURE 6. Results comparison of ML-LGBM with XGBoost.

TABLE 7. Comparison between ML-LGBM and other classifiers.

Model	Model size/byte	Fitting time/second	Standard deviation	Log Loss
ML- LGBM	347530	140.217	0.00024	0.1289
GB	369816	656.730	0.00060	0.3090
EXT	192835	629.326	0.00049	0.2199
RF	377721	781.503	0.00037	0.2304
K-NN	859819	1024.39	0.00151	0.3602
XGBoost	552088	208.752	0.00065	0.3919

it is clear that when the tree increases, the time also increases. The best accuracy is obtained when the number of estimators

TABLE 8. Effect of the number of estimators on accuracy.

Model	N-estimators	Time/second	Standard deviation	Accuracy
ML- LGBM	50	22.714	0.00054	0.986
	100	41.289	0.00040	0.992
	150	62.483	0.00033	0.994
	200	94.190	0.00018	0.995
	250	101.897	0.00028	0.995
	300	122.649	0.00029	0.995
	350	140.217	0.00024	0.996

is 350.

Furthermore, to emphasize the proposed detection method advantages, five experiments to compare ML-LGBM with five well-known ML techniques are carried out. The five models are as follows: Gradient boosting, Extra trees, Random forest, K-NN classifier, and XGBoost. Figure 8(a), (b),(c) and (d) presents the results obtained from the different classifiers depends on three criteria; the first criterion is the learning curve which compares the training examples with accuracy. The second criterion is the model scalability shown the number of training compared with the fitting time (training and validation time). The last criterion is the model performance shows the variation of the time taken to train each model. The results demonstrate that the ML-LGBM model outperforms the other four ML algorithms, making it light and easy to deploy. For an in-depth overview of the concept of MLLGBM, Figure 9 is presented, which depicts the model's first tree (No. 1), and Figure 10 depicts the model's last tree (No.300).

Further, to objectively evaluate the proposed scheme, the results of the ML-LGBM model performance is compared with the methods proposed by Yavuz et al. [16], Verma et al. [20], Sharma et al. [21] and Verma et al. [22] that shown in Table 9. The proposed ML-LGBM model outperforms the previously proposed schemes significantly, and this is shown in ML-LGBM outstanding performance achieved with high accuracy, precision, recall and F-score.

V. CONCLUSION

This paper proposed a machine learning model ML-LGBM for detecting VNA in an RPL-based network. For our study, we generates a dataset using the Cooja simulator for testing and validating and the generated dataset contains 1,050,861 rows where 884.861 are benign and 166.000 are malicious. The dataset has been used as a benchmark of our model. Considerable analysis has been performed to test the proposed model at various stages with the conclusion that ML-LGBM can achieve optimized results and is more advantageous over all the other machine learning models in terms of accuracy, precision, recall, F-score, true negative rate and false-positive rate. In addition, An in-depth comparison between the ML-LGBM and other well-known ML techniques demonstrated that the ML-LGBM model has a lower standard deviation, lower Log Loss, consumes fewer resources, and less training and testing time. However, ML-LGBM can be a better choice for attack detection in RPL-based networks from all the modes and techniques mentioned above. Although the proposed model showed high performance as it was trained using a large and unique data (relying on RPL properties such as version number and rank fields), it has some limitation, i.e., the data contains only one type of RPL attack. Consequently, future work is needed to add other variants to the dataset to further evaluate the model.

REFERENCES

[1] M. A. M.Sadeeq, S. R. M. Zeebaree, R. Qashi, S. H. Ahmed, and K. Jacksi, ŞInternet of Things Security: A Survey,Ť in 2018 International Conference on Advanced Science and Engineering (ICOASE), Oct. 2018, pp. 162Ű166, doi: 10.1109/ICOASE.2018.8548785.

[2] M. binti Mohamad Noor and W. H. Hassan, ŞCurrent research on Internet of Things (IoT) security: A survey,Ť Comput. Networks, vol. 148, pp. 283Ű294, Jan. 2019, doi: 10.1016/j.comnet.2018.11.025.

[3] G. Sharma and S. Kalra, ŞA Lightweight User Authentication Scheme for Cloud-IoT Based Healthcare Services,Ť Iran. J. Sci. Technol. Trans. Electr. Eng., vol. 43, no. S1, pp. 619Ű636, Jul. 2019, doi: 10.1007/s40998-018-0146-5.

[4] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, ŞInternet of Things security: A survey,Ť J. Netw. Comput. Appl., vol. 88, pp. 10Ű28, Jun. 2017, doi: 10.1016/j.jnca.2017.04.002.

[5] Z. Yan, P. Zhang, and A. V. Vasilakos, ŞA survey on trust management for Internet of Things,Ť J. Netw. Comput. Appl., vol. 42, pp. 120Ű134, Jun. 2014, doi: 10.1016/j.jnca.2014.01.014.

[6] O. Gaddour and A. Koubâa, ŞRPL in a nutshell: A survey,Ť Comput. Networks, vol. 56, no. 14, pp. 3163Ű3178, Sep. 2012, doi: 10.1016/j.comnet.2012.06.016.

[7] D. Praveen Kumar, T. Amgoth, and C. S. R. Annavarapu, ŞMachine learning algorithms for wireless sensor networks: A survey,Ť Inf. Fusion, vol. 49, pp. 1Ű25, Sep. 2019, doi: 10.1016/j.inffus.2018.09.013.

[8] P. Thubert et al., ŞRPL: IPv6 Routing Protocol for Low power and Lossy Networks,Ť IETF, vol. RFC 6550, 2012.

[9] B. Ghaleb et al., ŞA Survey of Limitations and Enhancements of the IPv6 Routing Protocol for Low-Power and Lossy Networks: A Focus on Core Operations,Ť IEEE Commun. Surv. Tutorials, vol. 21, no. 2, pp. 1607Ű1635, 2019, doi: 10.1109/COMST.2018.2874356.

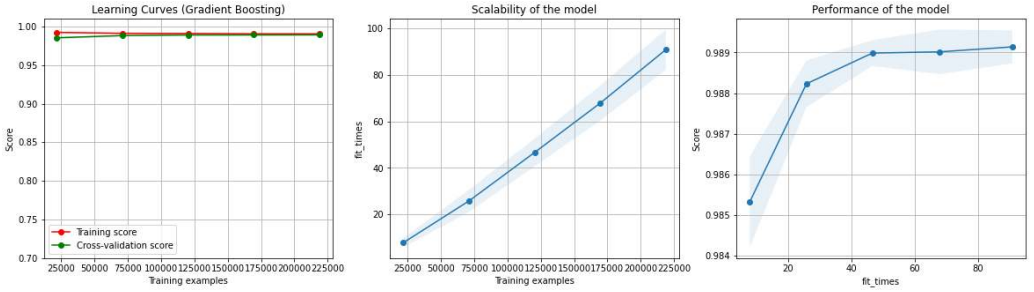
[10] A. Raoof, A. Matrawy, and C.-H. Lung, ŞRouting Attacks and Mitigation Methods for RPL-Based Internet of Things,Ť IEEE Commun. Surv. Tutorials, vol. 21, no. 2, pp. 1582Ű1606, 2019, doi: 10.1109/COMST.2018.2885894.

[11] A. Aris, S. F. Oktug, and S. Berna Ors Yalcin, ŞRPL version number attacks: In-depth study,Ť in NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, Apr. 2016, pp. 776Ű779, doi: 10.1109/NOMS.2016.7502897.

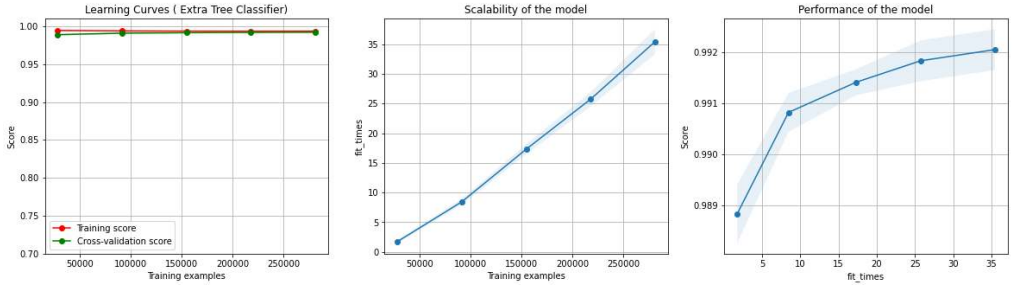
[12] A. Aris, S. B. Ö. Yalçin, and S. F. Oktug, ŞNew lightweight mitigation techniques for RPL version number attacks,Ť Ad Hoc Networks, vol. 85, pp. 81Ű91, 2019, doi: https://doi.org/10.1016/j.adhoc.2018.10.022.

[13] S. Mangelkar, S. N. Dhage, and A. V. Nimkar, ŞA comparative study on RPL attacks and security solutions,Ť in 2017 International Conference

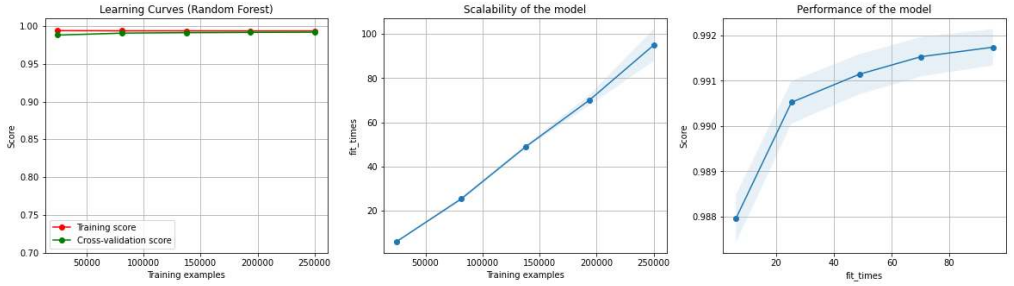
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3087175, IEEE Access



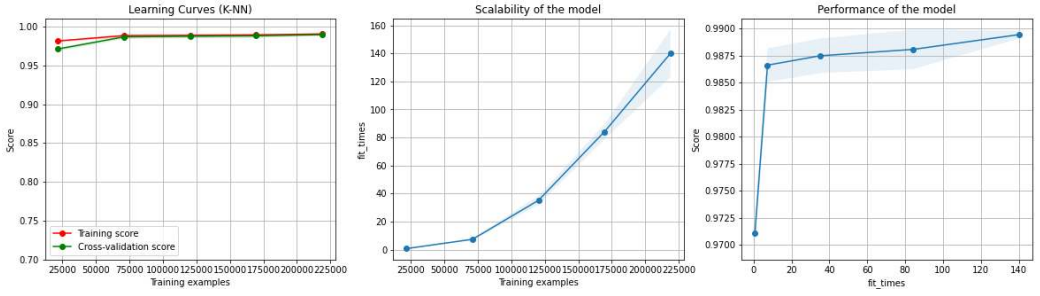
(a) Gradient boosting classifier performance.



(b) Extra trees classifier performance.



(c) Random forest classifier performance.



(d) K-NN classifier performance.

FIGURE 8. Results comparison of different classifiers.

TABLE 9. Comparison of related work.

Model	Precision	Recall	F- Score	Accuracy
Yavuz et al. [16]	0.940	0.940	0.950	94.70
Verma et al. [20] (DT results)	-	-	-	94.07
Sharma et al. [21]	0.960	0.950	-	95.33
Verma et al. [22]	-	-	-	94.50
ML- LGBM	0.990	0.996	0.993	99.60

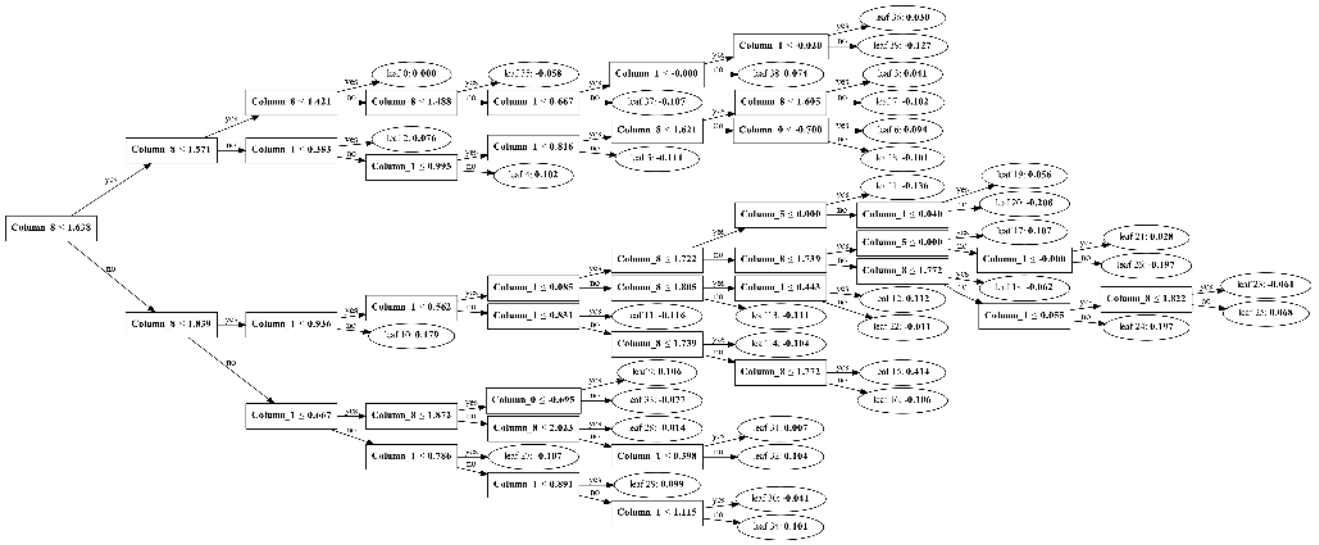


FIGURE 9. **The first tree in the ML-LGBM.**

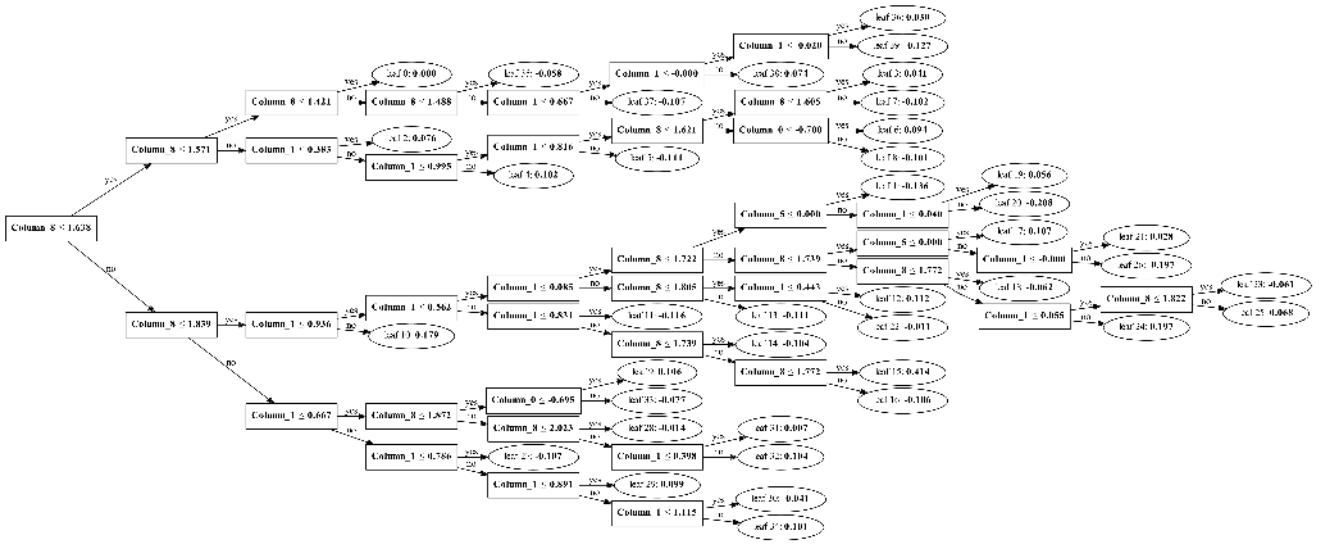


FIGURE 10. **The tree No. (300) in the ML-LGBM.**

on Intelligent Computing and Control (I2C2), Jun. 2017, pp. 1Ű6, doi: 10.1109/I2C2.2017.8321851.

[14] L. Wallgren, S. Raza, and T. Voigt, ŞRouting Attacks and Countermeasures in the RPL-Based Internet of Things,Ť Int. J. Distrib. Sens. Networks, vol. 9, no. 8, p. 794326, 2013, doi: 10.1155/2013/794326.

[15] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder, ŞA Study of RPL DODAG Version Attacks,Ť in Monitoring and Securing Virtualized Networks and Services, 2014, pp. 92Ű104.

[16] F. Y. Yavuz, D. Ünal, and E. Gül, ŞDeep learning for detection of routing attacks in the internet of things,Ť Int. J. Comput. Intell. Syst., vol. 12, no. 1, pp. 39Ű58, 2018, doi: 10.2991/ijcis.2018.25905181.

[17] E. Kfoury, J. Saab, P. Younes, and R. Achkar, ŞA Self Organizing Map Intrusion Detection System for RPL Protocol Attacks,Ť Int. J. Interdiscip. Telecommun. Netw., vol. 11, pp. 30Ű43, 2019, doi: 10.4018/IJITN.2019010103.

[18] A. Dvir, T. Holczer, and L. Buttyan, ŞVeRA - Version Number and Rank Authentication in RPL,Ť in 2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems, Oct. 2011, pp. 709Ű714, doi: 10.1109/MASS.2011.76.

[19] R. Sahay, G. Geethakumari, B. Mitra, and I. Sahoo, ŞEfficient Framework for Detection of Version Number Attack in Internet of Things,Ť 2020, pp. 480Ű492.

[20] A. Verma and V. Ranga, ŞEvaluation of Network Intrusion Detection Systems for RPL Based 6LoWPAN Networks in IoT,Ť Wirel. Pers. Commun., vol. 108, no. 3, pp. 1571Ű1594, Oct. 2019, doi: 10.1007/s11277-019-06485-w.

[21] M. Sharma, H. Elmiligi, F. Gebali, and A. Verma, ŞSimulating Attacks for RPL and Generating Multi-class Dataset for Supervised Machine

Learning,Ť in 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Oct. 2019, pp. 0020Ű0026, doi: 10.1109/IEMCON.2019.8936142.

[22] A. Verma and V. Ranga, ŞELNIDS: Ensemble Learning based Network Intrusion Detection System for RPL based Internet of Things,Ť in 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), Apr. 2019, pp. 1Ű6, doi: 10.1109/IoT-SIU.2019.8777504.

[23] S. Cakir, S. Toklu, and N. Yalcin, ŞRPL Attack Detection and Prevention in the Internet of Things Networks Using a GRU Based Deep Learning,Ť IEEE Access, vol. 8, pp. 183678Ű183689, 2020, doi: 10.1109/ACCESS.2020.3029191.

[24] E. Kfoury, J. Saab, P. Younes, and R. Achkar, ŞA Self Organizing Map Intrusion Detection System for RPL Protocol Attacks,Ť Int. J. Interdiscip. Telecommun. Netw., vol. 11, no. 1, pp. 30Ű43, Jan. 2018, doi: 10.4018/ijitn.2019010103.

[25] N. D. Patel, B. M. Mehtre, and R. Wankar, ŞSimulators, Emulators, and Test-beds for Internet of Things: A Comparison,Ť in 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Dec. 2019, pp. 139Ű145, doi: 10.1109/I-SMAC47947.2019.9032519.

[26] B. Mahapatra and S. Patnaik, ŞSelf Adaptive Intrusion Detection Technique Using Data Mining Concept in an Ad-hoc Network,Ť Procedia Comput. Sci., vol. 92, pp. 292Ű297, 2016, doi: 10.1016/j.procs.2016.07.358.

[27] A. Marcano-Cedeno, J. Quintanilla-Dominguez, M. G. Cortina-Januchs, and D. Andina, ŞFeature selection using Sequential Forward Selection and classification applying Artificial Metaplasticity Neural Network,Ť in IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society, Nov. 2010, pp. 2845Ű2850, doi: 10.1109/IECON.2010.5675075.

[28] Y. Meidan, V. Sachidananda, H. Peng, R. Sagron, Y. Elovici, and A. Shabtai, ŞA novel approach for detecting vulnerable IoT devices connected behind a home NAT,Ť Comput. Secur., vol. 97, p. 101968, Oct. 2020, doi: 10.1016/j.cose.2020.101968.

[29] G. Ke et al., ŞLightgbm: A highly efficient gradient boosting decision tree,Ť in Advances in neural information processing systems, 2017, pp. 3146Ű3154.

[30] F. M. M. Mokbal, W. Dan, A. Imran, L. Jiuchuan, F. Akhtar, and W. Xiaoxi, ŞMLPXSS: An Integrated XSS-Based Attack Detection Scheme in Web Applications Using Multilayer Perceptron Technique,Ť IEEE Access, vol. 7, pp. 100567Ű100580, 2019, doi: 10.1109/ACCESS.2019.2927417.

[31] F. M. M. Mokbal, D. Wang, X. Wang, and L. Fu, ŞData augmentation-based conditional Wasserstein generative adversarial network-gradient penalty for XSS attack detection system,Ť PeerJ Comput. Sci., vol. 6, p. e328, Dec. 2020, doi: 10.7717/peerj-cs.328.

MUSA OSMAN is a PhD student at Beijing University of Technology (BJUT), China. He received his BSc in computer science at the University of Gazira, Sudan, and MSc in Information System at Osmania University,India. His main research interests are security issues in the Internet Of Things, primarily based on RPL protocol, Machine Learning, and Artificial Neural Network.

JINGSHA HE (Member, IEEE) received the bachelorŠs degree in computer science from XiŠan Jiaotong University, China, and the masterŠs and Ph.D. degrees in computer engineering from the University of Maryland, College Park, MD, USA. He worked for several multinational companies in USA, including IBM Corp., MCI Communications Corp., and Fujitsu Laboratories. He is currently a Professor with the Faculty of Information Technology, Beijing University of Technology (BJUT), Beijing. He has published more than ten articles. He holds 12 U.S. patents. Since August 2003, he has been published over 300 papers in scholarly journals and international conferences. He also holds over 84 patents and 57 software copyrights in China and authored nine books. He was a principal investigator of more than 40 research and development projects. His research interests include information security, wireless networks, and digital forensics.

FAWAZ MAHIUOB MOHAMMED MOKBAL received his BS degree in Computer Science from Thamar University, Yemen, and MS degree in Information Technology from the University of Agriculture, Pakistan. He is currently a Ph.D. researcher in Computer Science and Technology with Beijing University of Technology, China. He also Research Associate with the Faculty of Computer Science at ILMA University, Pakistan. He served as head of the Technical Team of Information Center Project for the local Authority for 2 years, and Manager of Information Systems at the Ministry of Local Administration for 5 years. He is the author and reviewer with various SCI, EI, and Scopus indexed journals. His interest area includes Machine and Deep Learning, Medical Images, Brain-Computer Interface, Web Application Security, and IoT security issues.

NAFEI ZHU received the B.S. and M.S. degrees from Central South University, China, in 2003 and 2006, respectively, and the Ph.D. degree in computer science and technology from the Beijing University of Technology, Beijing, China, in 2012. She was a Postdoctoral Research Fellow with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, from 2015 to 2017. She is currently an Associate Professor with the Faculty of Information Technology, Beijing University of Technology. She has published over 20 research papers in scholarly journals and international conferences. Her research interests include information security and privacy, wireless communications, and network measurement.

SIRAJUDDIN QURESHI received his bachelorŠs degree in Computer Sciences from Quaid-e-Awam University of Engineering, Science Technology, Pakistan. Afterwards, he pursued his MasterŠs in Information Technology from Sindh Agricultural University Tandojam, Pakistan. Currently he is pursuing PhD in Information Technology at Beijing University of Technology, China. He has nine research publications to his credit as main author and co-author, which featured national and international journals and conferences. SirajuddinŠs research areas includes but not limited to Network Forensics Analysis, Digital Forensics, Cyber security, Computer Networks and Network Security.