International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-6, April 2020

Robust Intelligent Malware Detection using Light GBM Algorithm

Mohammad A. Abbadi, Ahmed M. Al-Bustanji, Mouhammd Al-kasassbeh

Abstract: ***Attackers take advantage of every second that the anti- vendor delays identifying the attacking malware signature and to provide notifications. In addition, the longer the detection period delayed, the greater the damage to the host device. To put it another way, the lack of ability to detect attacks early complicates the problem and rises serious harm. Consequently, this research intends to develop a knowledgeable anti-malware system capable of immediately detecting and terminating malware actions, rather than waiting for anti-malware updates. The research concentrates in its scope on the detection of malware on the Internet of Things (IoT), based on Machine Learning (ML) techniques. A latest open source ML algorithm called the Light Gradient Boosting Algorithm (LightGBM) has been used to develop our instant host and network layer anti-malware approach without any human intervention. For examination reasons, the suggested approach serves the LightGBM machine learning algorithm to adopt datasets obtained from real IoT devices using the LightGBM machine learning algorithm. The results indicate a successful method to detecting and classifying high accuracy malware at both network and host levels based on the Holdout method of cross-validation. Additionally, this result is better than many prior related studies which used different algorithms of Machine Learning and Deep Learning. Though, an old study which used the same dataset was the best among the literature. However, it still slightly less than what this study achieved, besides the complexity which deep learning adds. Lastly, the results show the ability of the proposed approach to detect IoT botnet attacks fast, which is a vital feature to end botnet activity before spreading to any new network device.***

Keywords: ***Malware, Classification, Machine Learning, Botnet, Detection, Internet of Things, Gradient Boosting, LightGBM.***

I. INTRODUCTION

Competition between attacks and security defenses will never end. With each security enhancement, new attacking tools are developed to overcome security defense. Malware or malicious software is the most common type of cybersecurity threats that can perform either active attacks, passive attacks or both together [1]. Traditional virus scanning solutions rely on manually created malware signatures and statistics analysis, which never be able to practically satisfy the increasing demand for security defense solutions against malware. Off-the-shelf antivirus software products require to be updated frequently with the newly detected malware signatures.

Revised Manuscript Received on April 30, 2020.
***** Correspondence Author
Mohammad A. Abbadi*, College of Information Technology, Mutah University, Karak, Jordan. abbadi@mutah.edu.jo
Ahmed M. Al-Bustanji, College of Information Technology, Mu'tah University, Karak, Jordan. Bustanji@hotmail.com
Mouhammd Al-kasassbeh, College of Information Technology, Mutah University, Karak, Jordan. Malkasasbeh@gmail.com

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

Therefore, traditional virus software unable to detect malware in real time of the zero-day attack. However, after new malware's first attack and classified as wild, companies analysis the malware and create their signature then release definition updates to their products so it can recognize the new malware [2] [3].

Before the release of definition updates, several terabytes of data may be lost or stolen, and millions of dollars might get lost because of these attacks. Governments, companies, and individuals are potential victims of malware attacks. With every zero-day malware attacks, there will be a massive and unrecoverable financial and data loss. The number of the

victims grow as well as the loss until vendors of anti-malware update the client's software.

Malware challenge is continually evolving along with the dramatic increase in the number of victims due to the increased number of cyberspace users. In 2015, Panda Security Company announced that 230,000 new malware attacks produced in a daily base [6]. Thus, mitigating its impact has raised the demand to find a new approach for real-time detection and identification of malware attacks.

IoT is an evolutionary technology that recently emerged, and soon it will become the key to smart-life. In 2016, 9 billion IoT devices used and the number still growing [7]. The good thing about IoT that it makes life easier, safer and more fun. Though, like anything else, advantages come along with disadvantages. Many researchers start to name IoT as the Internet of Vulnerabilities (IoV) or Internet of troubles, due to functionality mutation by cyber attackers. These vulnerabilities attracted attackers who start using the popular unsecured devices exposed to the internet to host their attacks [7], [9].

Researchers who dealt with malware detection used various machine learning algorithms for classification; some

achieved better results than others. On the other hand, the literature focused on malicious software for computer operating systems, while others focused on IoT malware. The LightGBM algorithm used in two types of studies, one type was not related to malware, such as Fonseca et al. [12] which dealt with the classification of the acoustic scene classification. Though, it achieved significant results. The other type is closed to our study, such as Su et al. [13] who used LightGBM based on image recognition to classify IoT botnets. However, the results of the study were not good enough compared to other algorithms.

This research used machine learning to solve malware detection problem, by applying malware classification using LightGBM on IoT botnet. However, in this research, Machine Learning (ML) techniques and its applications to manage malware attacks are exploited based on LightGBM, which is one of the most powerful and high-performance machine learning algorithms recently developed by

Microsoft [13] [14] on IoT botnet.

A predefined dataset [15] related to IoT heterogeneous devices connected to a network used for evaluating our proposed approach. The target dataset includes (115) features obtained from different IoT devices. Considering benign traffic and botnet traffic collected from Distributed Denial of Service (DDoS) attack initiated using two families of IoT malware (Mirai, and Gafgyt).

Our approach will present efficient and accurate malware classifier to solve a challenging issue related to malware attacks in IoT. Our experiments, procedures, and results provide the flowing contributions:

- Our solution classifies the IoT botnets in both network and host layers, which provide more accurate results that maintained the same precision for classifying both IoT botnets tested in the experiment.
- Our solution can detect IoT botnet in the early phase of attack without losing its efficiency, regardless of the various devices and their operating systems.
- This research evaluates LightGBM ML algorithm for IoT botnets high-efficiency classification. Thus, this study will add new literature for scholars and future work.

This paper organized into seven sections. The next section review the IoT security and vulnerabilities. The third section discusses the Botnet attacks and its effect. Then, the fourth section discusses some of the most recent and related works to malware classification. Precisely, we discuss the used methods and algorithms for malware detection and classification, besides the achieved results for each literature. The review includes studies for computers malware classification but mainly focused on IoT botnets. The fifth section discusses the model design and the methodology for our experiment. Then, it describes experiment settings, regarding both data, classifier, and experiment phases.

The sixth section discusses the evaluation criteria, then analyze the experiment results and findings to evaluate them properly. At the end of the first part, it compares the output results based on the experiment phases, as well as the devices and botnets to understand their relation. Then it compares our method with the most related works. Finally, the last section concludes this study, along with our recommendations for future work.

II. INTERNET OF THINGS (IOT)

Researchers introduced several definitions for the IoT; however, all of them have the same concept. [16] described IoT as “the Internet of Things (IoT) is composed of inexpensive, low-power, always-on, always-networked, inconspicuous devices scattered throughout the home, office, and other environments.

These devices provide a variety of services, such as home automation, industrial process monitoring and control, entertainment, and safety” [16]. Similarly, [17] raised and answered a crucial question about the definition of “Things.” Their answer was: “we define a "Thing" on IoT that indicates a physical or virtual object which connects to the Internet and able to communicate with human users or other objects” [17].

The Internet of Things (IoT), Industrial Internet, or Internet of Everything [18] can be defined as any device, machine or appliance connected to the internet and interacts with each other without any human intervention, exclude any type of computers, laptops, tablets, smartphones and fixed phone. IoT is the future of the technology that will make smart-life, smart home, and smart cities [19], [18]. Using IoT means to configure, monitor and control the devices (Things) connected to the internet. The connected devices can communicate and send feedback or warning to the user.

The number of IoT connected objects have been growing in the last few years with broader applications domains. [8] expected that the number of IoT devices would be around 30.7 billion in 2020 and it will be 75.4 billion devices in 2025. However, [20] were less optimistic; they argued through IoT analytics website that the number of active devices would evolve to 21.5 billion in 2025. However, the number of non-IoT devices will be almost half. Also, Forbes expected a very high IoT market size in a few years.

III. IOT SECURITY

Despite IoT advantages, its rapid growth and increasing demand, critical security issues have emerged. IoT devices have many vulnerabilities due to the weakness of their operating system (OS), the complexity of hardware,

and the limited resources which are inapplicable to host existing computer security software [13].

IoT security standards introduced by different standard organizations [22]. However, the most well-known IoT vulnerabilities exposed to hackers [7], [23] are:

1) Wireless connections, Web Interface, Cloud Interface, Firmware, and Network Services are insecure.

2) Application domains diversity of the Internet, and IoT.

3) The diversity of hardware, and vendors.

4) IoT devises physical security.

5) Cloud storage save data.

6) Everything could be identified.

All these vulnerabilities make IoT devices very attractive to hackers and cyber-attacks. Traditional endpoint security solutions are not capable of protecting IoT devices. Therefore, IoT devices need a smart, efficient and light-weight security solution, which is not possible without using artificial intelligence. Furthermore, any proposed solution must be valid on IoT devices as well as the network.

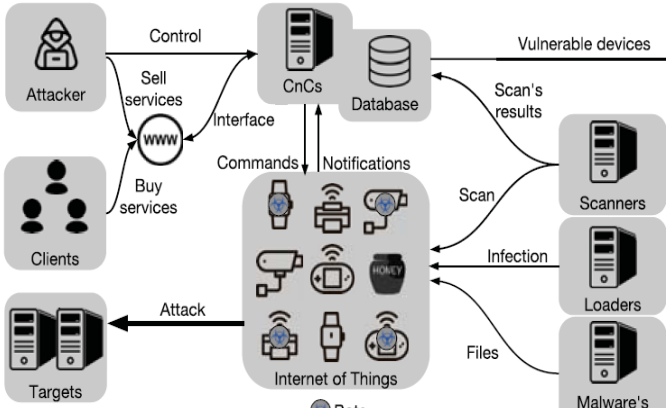
Malware develops in conjunction with IoT to the point where a specialized type of such software so-called “Botnet” has recently emerged. There are some differences between computers malware and botnet. Table I presents the main IoT weak points and summarizes the differences between the PC and IoT malware [24].

TABLE I. WHAT IS SO UNIQUE ABOUT IOT MALWARE? [24]

	PC	IoT
Platform heterogeneity	low	high
Malware family plurality	high	low
Detection on the system	easy	hard
In-vivo analysis	easy	very hard
Sandbox execution	easy	hard
Removal	medium	hard to impossible
Vulnerability assessment	medium	very hard

IV. BOTNET ATTACKS

The botnet term is an abbreviation for "robots network," a network that connects IoT devices that hosting a (robots) together with victims' computers. Attackers are taking advantage of IoT vulnerabilities to inject botnet malware into the IoT devices to initiate a wide range of attacks to one or several computers. A botnet can remotely control IoT devices as a group using unauthorized remote access [26]. Thus, IoT devices are suitable to host robots to commit a cybercrime while the criminal is safe and sound somewhere else in the world. Fig. 1. demonstrates how the IoT botnet attack works.



Overview of an IoT botnet [11].

There is a long list of IoT malware capable of making active or/and passive attacks. The DDoS attack is the most famous attack executed by IoT botnets related to many families besides credential theft, phishing and other types of attacks [7], [27]. The most common families Among IoT botnets are:

- Mirai: which is one of the most used IoT botnets for DDoS attacks; it can infect in one hour about 4000 IoT devices [15].
- Bashlite: which also called Gafgyt, Torlus, or Liz kebab, targets IoT devices that have firmware based on Linux OS.

The botnet attack divided into four phases [29]: training, command and control, attack, and post-attack. These phases used by [27] to classify botnets as centralized and decentralized. Centralized means that malware communicates with command and control (C&C) Server which propagate it to all connected IoT devices using various protocols such as Hyper Text Transfer Protocol (HTTP) and Internet Relay Chat (IRC). Decentralized botnets force the infected bot, which received the malware from the server, to start propagating to other connected IoT devices.

V. MACHINE LEARNING APPLICATION IN CYBER SECURITY

Artificial Intelligence (AI) defined from different perspectives; however, the definitions share the same concepts. Artificial means any self-reproducing system that able to evolve independently without any human impact. Intelligence is a human characteristic, which means to learn, think, behave rationally, make decisions and reasonably solve problems. AI can be viewed as the discipline in which machines can learn, think and perform some tasks in a way that resembles human to some extent [30].

Machine learning is a fundamental approach to implement AI applications by developing algorithms to analyze data and make predictions to construct systems based on the gained information [31]. ML approach is instrumental in training the machine or the system by applying one of its algorithms to the available data to create a predictive and automatic classification capability. Several algorithms, have been developed for ML [32] as follows:

- Supervised ML algorithms: Artificial Neural Network (ANN), K-Nearest Neighbors (KNN), Decision Tree (DT), Bayesian, Gaussian Mixture Models (GMM), Gradient Boosting Machine (GBM), and Support Vector Machine (SVM).
- Unsupervised ML algorithms: Dimensionality reduction, nearest-neighbor, farthest neighbor, K-means clustering, and hierarchical clustering.
- Semi-supervised ML methods: Co-training, generative models, Low-density separation, and graph-based method. In addition to heuristic approaches.
- Reinforcement ML algorithms: Criterion of optimality, brute force, value function, and Direct policy search.

VI. MALWARE DETECTION

Classification is a supervised learning method that predicts a specific behavior based on static or dynamic features learned from a set of training data and then uses what has been learned to classify a new data set. However, rating accuracy depends on features along with the accuracy of the data provided [33], [34]. Literature has introduced many classifiers (classification algorithms) such as linear classifiers (logistic regression, naive Bayes classifier), support vector machines, neural networks, random forests (RF), decision tree, boosted tree and nearest neighbor. Data used for training or classification could be bi-class or it could multiclass based on the study topic and the expected results. There are many applications for classification such as recognition of handwriting and speech, biometrics, malware, …etc. [35].

VII. DETECTION METHODS AND ALGORITHMS

Accurate and sufficient number of features and cases in the dataset are very critical for accurate classification results. Hence, detecting malware must be automatic, efficient, effective and accurate [36].

Malware can be detected and analyzed by both static or dynamic analysis using two techniques: a)code analysis without executing the software (signature based); b) behavioral analysis (anomaly based) Researchers used a diversity of techniques for detecting malware despite how they handled the results [41]. Fig. 2 illustrates some of these techniques.

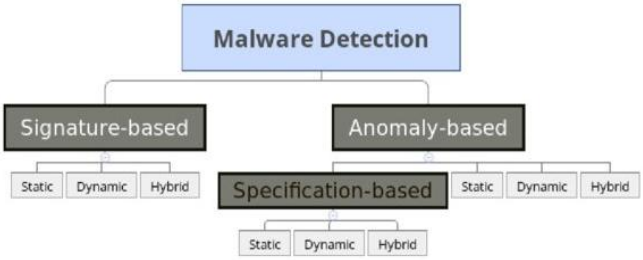


Fig. 1.Types of Malware Detection Techniques [41].

A signature-based technology has been used for a while by security companies to protect their customers. Unfortunately, it failed to prevent malicious software on the zero-day or invisible malware of causing severe damage in the first attack [40].

Hence, researchers decided to develop the techniques in both directions. However, behavioral analysis takes more attention. The most important behavior of malware is the interaction with the operating system. With every version of windows, Microsoft releases an Application Programming Interface (API) index, which has a list of references for windows APIs [42]. These APIs facilitate the application to run and take advantage of all OS capabilities and features. Malware, as well, uses APIs of OS to do what it designed to do. This behavior of malware (API calling) follows a sort of pattern, for example, APIs been called, objects been targeted in the OS and so on.

Several approaches used the analysis of API calls [38], [39], [40]. Some researchers used the Dynamic Link Library (DLL) injection techniques besides API calls [25]. Although it may be easy to create sufficient data to be analyzed, a significant effort and time may be consumed to obtain accurate datasets with the required number of instances. For instance, [38] spent six months collecting data using VirusTotal tool to obtain 32475 malware samples that represent only nine malware families. Another example, [39] used SQLite to create a particular transactional Structured Query Language database (SQL DB). A complex script to generate API and save and index them in the DB. API list Downloaded from Microsoft Developer Network (MSDN) to be compared using Python and matched with the generated list in the DB. A more complex process used by [40]. They gathered a considerable number of Portable Executable (PE) files. All files loaded to windows. They recorded the activity of API calls for each loaded PE file in Imported Address Table (IAT) where each PE has a reserved spot written by Windows loader. Finally, PE analyzer used to extract 44605 distinct API out of over two million API procedure had been recorded.

Since all types of threats analysis should be conducted in a secure and isolated environment, selecting a suitable sandbox for the experiment makes a big difference. Cuckoo Sandbox is the most popular for malware classification. For example, It used by [37], [38], [39], and [24]. It is an open source and supports many operating systems for the virtual environment. It has many features that make it appropriate for this type of analysis, such as the ability to trace API calls and analyze the general behavior of the files.

Moreover, it can analyze and dump even encrypted network traffic [24], explained how they collected, categorized and selected the features. Directly, they started to discuss the experiment results of classification; they never mentioned any tool used for classification. While others used other tools; for example, [25] used CWSandbox. However, the sandbox is not accurate as it will be in a real-world environment. The real environment could have many effective changes that could not be reflected in the sandbox environment [33], for example:

1) Physical vs. Virtual machine.
2) Operating system cumulative updates.
3) Security patches.
4) Customization made by users.
5) Applications and software installed.

Additional secondary tools used by researchers to support the experiment; however, researchers did not give attention to discuss it. [38] used INetSim to emulate internet service instead of real internet connection to ensure isolation. They used Yara as well, which describes malware in the role-based approach [44].

VIII. RELATED WORKS

Gandotra et al. [5] presented an extensive literature review on the related works (before 2013). They concluded that statistical analysis alone was not sufficient because it would not instantly detect the zero-day malware. Dynamic analysis is flawed as well because it requires and consumes time and resources. Their recommendation is to adopt a hybrid technique combining static and dynamic analysis.

Su et al. [13] claimed that their study was the first classification that tested on real IoT botnet samples. Also, they claimed that the introduced classification system could be easily deployed to any real IoT device. Based on the detection schema for the proposed light-weight solution they argued that the lightweight classification system using CNN does not need training data based on compared to all other studies that used SVM, or *k*-nearest neighbors.

Based on Neural Network (NN), they implemented their experiment for image recognition. The dataset has the most samples from the following three families (Mirai, Gafgyt, and Linux.Fgt) the rest of the samples from other different families. However, the best result they could accomplish using CNN is 94% accuracy with 94.67% True Positive Rate (TPR) and 93.33% True Negative Rate (TNR).

Meidan et al. [15] proposed a network-based technique using deep learning to perform anomaly detection. They extracted behavior static features using IoT benign traffic from devices infected with real botnets (Mirai, and Bashlite which known as well as Gafgyt) and used it to train Deep AutoEncoder (DAE), which is a deep learning neural network architecture.

The results of their experiment were promising. Although, the training time for benign traffic of each device was relatively high, for example, training on 19,528 benign instances with 172kB size took 190 seconds. Moreover, their research was limited to the node layer. The experiment tested each device in separate of other devices excluding network layer, while including network layer could introduce a wider solution for malware detection.

Costin et al. [24] introduced an open source framework for the analysis of IoT malware. They claimed that their work would fill the gap between studies in IoT malware field and their framework would help researchers in the future to better understand IoT malware and better defense them. The study estimated 90 days advantage of automated IoT malware detection before samples analyzed for its signature. They found that almost 60% of IoT malware families had two instances, which could be an interesting finding that explains a lack of accuracy in static analysis and can be an essential error factor. In the other hand, Although the paper published in Aug 2018, it was not published in Research Gate and has zero citations in google scholar to the time when this thesis wrote. They recommended the researchers to improve cyber-security by improving classification performance and quality.

Alejandre et al. [27] emphasized the importance of detecting the botnet during the early phase of its life cycle. In their study, they focused on detection process during the C&C phase according to Leonard et al. [29]. Authors proposed Genetic Algorithm (GA) integrated with the C4.5 algorithm for classification and evaluating the generated features.

For the experiment, the authors developed a Java program to implement GA and applied based on two datasets that represent centralized botnets in one dataset and decentralized botnets in the other one. The best result they could achieve was 99.58% after ten iterations. They recommended testing the proposed solution on massive datasets.

Fonseca et al. [45] combined LightGBM and Convolutional Neural Network (CNN) for acoustic scene classification using the diagram in Fig. 3. Although, this study subject is not related to cyber-security, it worth to review the classification algorithm used. The study experiment proved that LightGBM algorithm is more accurate and perform classification training faster than its predecessor eXtreme Gradient Boosting (XGBoost). Furthermore, the experiment results were better than the previous related study which used the same dataset.

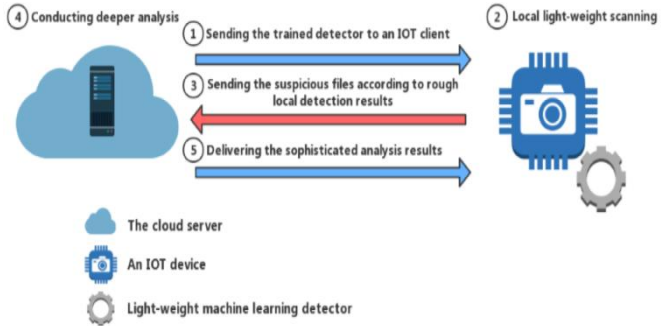


Fig. 2.Detection Schema for the proposed light-weight solution [13].

The experiment results divided into two stages. In the first stage, LightGBM achieved 80.8% accuracy improving the Multi-Layer Perceptron (MLP) of the previous study by 6% using the same dataset. While in the second stage, LightGBM combined with CNN to achieve 83% of accuracy, which means 8.2% improvement on the MLP of the previous study.

Islam et al. [36] investigated the efficiency of classification using data mining and machine learning. They argued that after making the classification public, the attacker would obfuscate the extracted features for that classification, which substantially reduce its accuracy. They added that the classification based on a given set of malware would fail or at least will not perform well with zero-day malware. They extracted static features required for their experiment, then used extracted features with their classification system.

The authors concluded that ML using Random Forests is the best classifier for their dataset. The accuracy of the experiment was 97% with the conclusion that the age of malware impacts the results.

Meng et al. [46] extracted features from behavioral analysis using API calls called Static malware Gene Sequences (SGS). They defined software genes as a fragment of the code extracted from programs that have functional information. In their experiment, they used a recursive descent algorithm using Interactive Disassembler Python (IDA Python) to extract the genes arranged in a two-dimensional matrix. They proposed a neural network module called "Static Malware Gene Sequences-Convolution Neural Network" (SMGS_CNN) for classification. In the experiment, they applied CNN on a dataset chosen from "VX-Heavens" web site [47]. The accuracy increased slightly in each iteration to achieve 98% accuracy after more than 3500 iterations as shown in Figure 18. They concluded that using CNN is better than traditional SVM which achieved 94.7% accuracy using the same dataset.

IX. DESIGN AND METHODOLOGY

A. LightGBM Classifier

Gradient boosting is one of machine learning algorithms used for classification and regression. It combines models from different algorithms to produce new iterative one. Gradient boosting is one of the most wildly used machine learning algorithms due to its accuracy and efficiency [14] [48] [49]. It started with the Adaptive Boosting (AdaBoost) then developed into many algorithms and techniques such as GBM and Model-Based Boosting (MBoost), then to CatBoost, XGBoost and LightGBM [50].

IoT networks made of heterogeneous devices that have limited resources. Such that the Lightweight classification algorithm is the best choice for any good performance security system for IoT networks. LightGBM machine learning algorithm inspired by Su et al. [13] and Fonseca et al. [45] who used it in similar studies.

LightGBM is a new gradient boosting framework based on decision tree algorithms, which introduced by Microsoft. It supports many algorithms like GBM, GBDT, Gradient Boosted Regression Tree (GBRT), and Multiple Additive Regression Tree (MART); as a result, it is scalable, accurate, and efficient [48]. The decision tree, in this algorithm, grows in leaf-wise [51], Fig. 4 demonstrates a model for prediction based on leaf-wise growth which optimize the loss which generates branches, Hince, this algorithm faster and less complexity than level wise growth which extend the tree depth [**14**]. According to Meidan et al. [15], the time complexity for the lightGBM calculated as O(#Data x #Features).

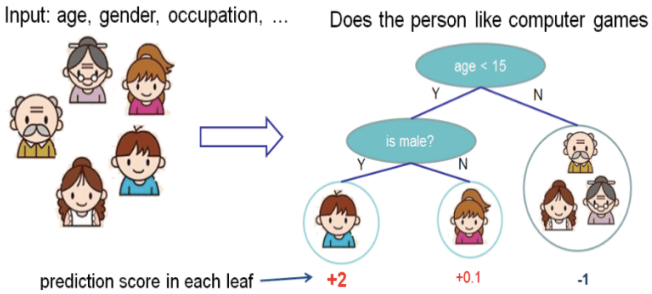


Fig. 3.Example of prediction based on leaf-wise growth [52].

According to [15], the time complexity for the lightGBM calculated as O(#Data x #Features). It contains two techniques, which are the GOSS and the EFB [53]. Given the supervised training set $X = \{(x_i, y_i)\}_{i=1}^{n}$, LightGBM algorithm aims to find an approximation $\hat{f}(x) = (x)$ to a certain function $f^*(x)$ that minimizes the expected value of a specific loss function $L(y, f(x))$ as follows:

$$\hat{f} = \arg\min_{f} E_{y,X} L(y, f(x)) \quad (1)$$

LightGBM integrates a number of T regression trees $\sum_{t=1}^{T} f_t(X)$ to approximate the final model, which is

$$f_T(X) = \sum_{t=1}^{T} f_t(X) \quad (2)$$

The regression trees could be expressed as $w_{q(x)}$, $q \in \{1, 2, \ldots, J\}$, where J denotes the number of leaves, q stands for the decision rules of the tree and w is a vector that denotes the sample weight of leaf nodes. Hence, LightGBM would be trained in an additive form at step t as follows:

$$\Gamma_t = \sum_{i=1}^{n} L(y_i, F_{t-1}(x_i) + f_t(x_i)) \quad (3)$$

In LightGBM, the objective function is rapidly approximated with Newton's method. After removing the constant term in the last equation for simplicity, the formulation can be transformed as follows:

$$\Gamma_t \cong \sum_{i=1}^{n} (g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)) \quad (4)$$

where g_i and h_i denote the first- and second-order gradient statistics of the loss function. Let I_j denote the sample set of leaf j, and the last equation could be transformed as follows:

$$\Gamma_t = \sum_{j=1}^{j} ((\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2)) \quad (5)$$

For a certain tree structure $q(x)$, the optimal leaf weight scores of each leaf node w_j^* and the extreme value of Γ_K could be solved as follows:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \quad (6)$$

$$\Gamma_T^* = -\frac{1}{2} \sum_{j=1}^{J} \frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \lambda} \quad (7)$$

Where Γ_T^* could be viewed as the scoring function that measures the quality of the tree structure q. Finally, the objective function after adding the split is:

$$G = \frac{1}{2} \left(\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda} \right) \quad (8)$$

where I_L and I_R are the sample sets of the left and right branches, respectively, unlike the traditional GBDT-based techniques, such as XGBoost and GBDT, LightGBM would grow the tree vertically whereas other algorithms grow trees horizontally, which makes LightGBM an effective method in processing large-scale data and features.

XGBoost is a simple solution that uses presort based algorithm for decision tree learning. Although, it is not smooth and a little bit complex to optimize. However, LightGBM increases training performance and reduce the usage of memory by using algorithms based on the histogram.

LightGBM can be described as the following [57]:

- Increases performance by using histogram subtraction and feature parallel which find and perform the best split.
- Significantly reduce memory usage by using discrete bins as replacement of continuous values to store training data.
- Optimizes network communication performance and reduce its cost in parallel learning by utilizing some collective communication algorithms.
- Suitable for large datasets more than XGBoost which consumes more training time.
- Supports parallel learning, besides its capability of launching multi-instances which allow using of multi Graphics Processing Unit (GPU).
- Supports weighted training by storing the weight of the data.

B. Datasets

IoT devices use the same internet protocols, which is the main common thing that can describe IoT devices' similarities. Traffic analysis is the best choice in the IoT network to detect and classify cyber-attacks. In any experiment, to get accurate results, accurate data must be provided as the experiment input. Hence, a dataset collected from real IoT devices' traffic is better to develop an applicable and reliable system in the real world IoT devices. Most previous experiments used datasets collected using a sandbox, which is not accurate as it would be in the case of a real-world environment. In this approach, we adopted real data introduced in [15] In their study; they set up their lab using real IoT devices for DDoS attacks initiated with two botnet families and nine IoT devices [15].

Each device has a single benign traffic dataset; besides, there are ten other datasets for each attack for most devices. Table II summarizes attacks datasets for IoT devices.

TABLE II. ATTACKS DATASETS FOR IOT DEVICES

Device Name	Device Type	Bashlite attack phases	Mirai attack phases
Danmini	Doorbell	5	5
Ennio	Doorbell	5	0
Ecobee	Thermostat	5	5
Philips B120N/10	Baby Monitor	5	5
Provision PT-737E	Security Camera	5	5
Provision PT-838	Security Camera	5	5
SimpleHome XCS7-1002-WHT	Security Camera	5	5
SimpleHome XCS7-1003-WHT	Security Camera	5	5
Samsung SNH 1011 N	Webcam	5	0

The files in a ".csv" format for selected datasets, uploaded by the authors to the repository system in the University of California, Irvine. The datasets are freely accessed online source, and consist of (115) features described in [15].

C. *Proposed Solution*

The proposed approach aims to develop a real-time system for detecting and classifying IoT botnets based on LightGBM ML techniques. The approach architecture and the model of the experiment illustrated in Fig. 5. First of all, we carefully selected the dataset which collected from a real IoT network, and then we implemented machine learning classifier to classify botnets attacks. Later in this chapter, architecture and experiment settings will clarify our approach.

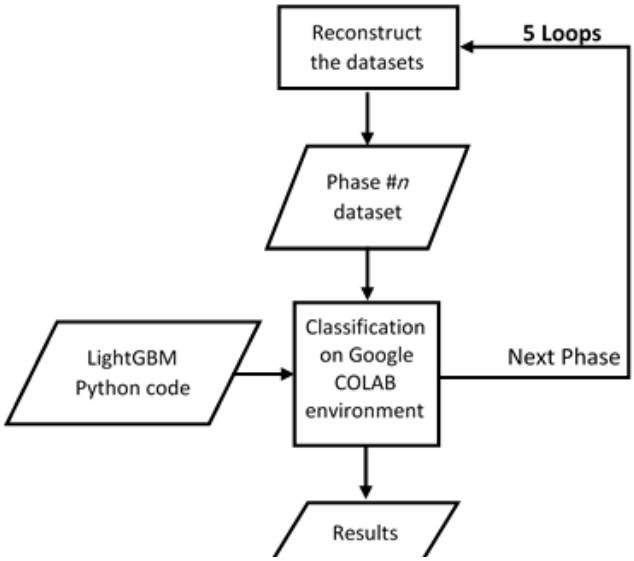


Fig. 4.Approach Architecture.

D. Architecture

Our approach implemented as a distributed architecture that covers multi-layers. Our approach based on implementing the selected algorithm for training and classification on the host and the network layers, such that, each layer has two stages, see Fig. 6.

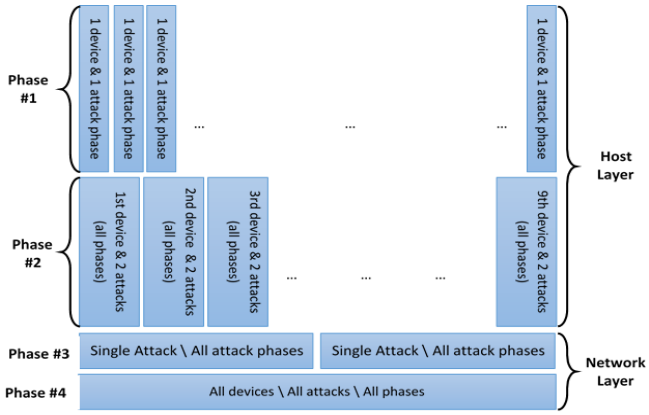


Fig. 5.Experiment's architecture.

The experiment is carried out in four stages; while results organized into five categories. The following list connects each results category to the related phase of the experiment:

1) ***Phases of attack\device****:* The most detailed results for the 1st phase.

2) ***Attack phases****:* It has driven out of the first category to evaluate the detection during an early phase of the attack.

3) ***Host-related results****:* for the host layer in the 2nd phase.

4) ***Malware****:* The results from the 3rd phase were based on a whole single IoT botnet repeated for the two tested IoT botnets.

5) ***Network layer****:* The results form the 4th phase of the experiment.

E. Experimental Settings

First of all, we reconstructed the selected dataset to align with the designed architecture, each dataset balanced to consist of fifty-fifty benign traffic and botnet attack chosen based on the aim of the experimental phase. Our experiment implemented using Google COLAB ordered as the following:

- **Phase #1: Phase of attack\device**: we separately performed the training and classification for each IoT device's benign traffic combined by one botnet attack (Mirai and Gafgyt) separately. Botnet attack selected from one attack phase on the same IoT device.
- Similarly, the training repeated for each phase for each botnet per device using 80 datasets, contains between 26,000 to 60,000 instances each.
- **Phase #2: Host layer**: the same procedures in the 1st phase applied; however, we involved both botnet attacks (Mirai and Gafgyt). Botnet attack selected from all attack phases on the same IoT device. Similarly, the training repeated for all botnets attacks for each device using nine datasets, each one contains 26,000 instances.
- **Phase #3: Network layer per malware**: again, we separately performed the training and classification for benign traffic form all IoT devices combined by single botnet attack (Mirai and Gafgyt). Botnet attack selected from all attack phases on all IoT devices. The training repeated twice for attacks for all devices using two datasets contains 182,000 instances each.
- **Phase #4: Network layer**: the same procedures in phase #3 applied; however, we involved both botnet attacks (Mirai and Gafgyt). The botnet attack selected from all attack phases on all IoT devices and contains 234,000 instances. In the first stage of this phase, we implemented the classifier on the binary-class dataset using *k*-fold cross-validation. While in the second stage, we used the Holdout method to handle multi-class dataset in the second stage.

Furthermore, we developed two codes for LightGBM algorithm using Python. One code used Holdout method to handle multi-class dataset in the second stage of the last phase. The other code designed to use *k*-fold, where k=10, to handle all other datasets used for training in all phases.

X. FINDINGS, AND DISCUSSIONS

A. Evaluation Criteria

k-Fold cross-validation is a statistical procedure to evaluate the skill of supervised machine learning models. According to Fig 7, Cross-validation delivers predictions accuracy and avoiding the overfitting where the model repeats the labels to get a perfect score but fails to get predictions [58].

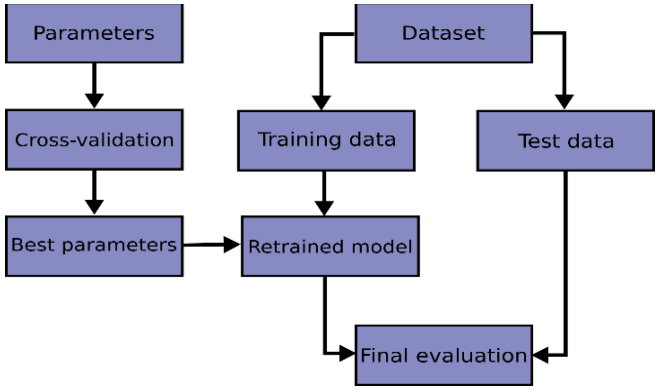


Fig. 6.Cross-validation workflow [59].

The *k*-fold in cross-validation splits the datasets to *k* equal sets, then uses (*k* – 1) sets for training. The validation results in each loop come from testings the last set. Eventually, it calculates the average of results collected from all *k* loops (folds) [58]. On the other hand, Holdout does not perform cross-validation. Though, it is another simple *k*-fold cross-validation method. This method split the data into two sets (training set, and testing set); commonly test set is smaller than training one 1:5.

To evaluate the experiment results, we considered a set of classification metrics such as (Accuracy, Area Under Curve (Auc), True Positive Rate (TPR), False Positive Rate (FPR), Mean Squared Error (MSE), Matthews Correlation Coefficient (MCC), Logarithmic Loss, and F1 Score). Also, to compare the results with related works, we used precision and recall as well.

B. Results Evaluation

The performance evaluation of the experiment was applied using Python code for binary datasets, where the class (1) represents benign traffic, and class (0) represents the attacks. Another Python code is used for Multi-class dataset, where benign traffic has the class (0), while attacks have classes (1 and 2).

Recalling the experiment setting, the proposed approach implemented in different four layers. The 1st phase used datasets that represent each device vs. one phase of a botnet attack. The metrics were very much similar for all devices, where the accuracy percentage is ranging between (0.9994) and (1.0000), with standard deviation range between (±0.0000) to (±0.0010).

Then, detailed results reanalyzed to conclude the results for each phase for both botnet attacks, which summarized in Table III. The results in Table III are the averages of each metric for each phase.

TABLE III. METRICS FOR EACH ATTACK PHASE.

Attack phase	Accuracy	AUC	TPR	FPR	MSE	MCC	Log. Loss	F1 Score
Phase #1	100%	100 %	100 %	0%	0%	100 %	0%	100%
Phase #2	100%	100 %	100 %	0%	1%	100 %	0%	100%
Phase #3	100%	100 %	100 %	0%	0%	100 %	0%	100%
Phase #4	100%	100 %	100 %	0%	0%	100 %	0%	100%
Phase #5	100%	100 %	100 %	0%	1%	100 %	0%	100%

In the second phase which represents the host layer, which used nine balanced datasets that represent the nine devices for all phases of both botnet attacks on the same device, All metrics measurements in this layer are better than the previous one.

Their averages are ranging between (0.9997) and (1.0000), See Table IV. In addition, standard deviation range between (±0.0003) and (±0.0006).

TABLE IV. METRICS FOR EACH DEVICE (HOST).

Device Name	Accuracy	TPR	MSE	MCC	Log. Loss	F1 Score
SimpleHome XCS7 1003 WHT Security Camera	0.9999	0.9999	0.0002	0.9998	0.0040	0.9999
SimpleHome XCS7 1002 WHT Security Camera	0.9998	0.9998	0.0002	0.9996	0.0066	0.9998
Samsung SNH 1011 N Webcam	0.9997	0.9997	0.0109	0.9994	0.0106	0.9997
Provision PT 737E Security Camera	0.9998	0.9998	0.0001	0.9997	0.0053	0.9998
Philips B120N10 Baby Monitor	0.9998	0.9999	0.0001	0.9997	0.0053	0.9998
Ennio Doorbell	0.9999	0.9998	0.0001	0.9998	0.0040	0.9999
Ecobee Thermostat	0.9999	1.0000	0.0001	0.9998	0.0027	0.9999
Danmini Doorbell	0.9999	0.9999	0.0025	0.9998	0.0027	0.9999
Provision PT 838 Security Camera	0.9998	0.9999	0.0001	0.9997	0.0053	0.9998
Average	**0.9998**	**0.9999**	**0.0016**	**0.9997**	**0.0052**	**0.9998**

Malware phase uses two balanced datasets, one dataset for each botnet. The accuracy for both botnets is 100%, and the standard deviation for both is 0%. Similarly, all other metrics indicate that the detection precision for both is the same as Tavle V emphasizes.

TABLE V. METRICS FOR THE NETWORK LAYER.

Device Name	Accuracy	TPR	MSE	MCC	Log. Loss	F1 Score
Gafgyt attack	1.0000	1.0000	0.0000	0.0000	1.0000	0.0008
Mirai attack	1.0000	1.0000	0.0000	0.0012	1.0000	0.0008
Average	1.0000	1.0000	0.0000	0.0006	1.0000	0.0008

The last phase which represents the network layer, used only one dataset, but with two cross-validation methods: *k*-fold and holdout. Ultimately, accuracy from both methods is 100% as represented in Table VI.

TABLE VI. METRICS FOR THE NETWORK LAYER.

Metric	Value
Accuracy	100%
AUC	100%
TPR Fold	100%
FPR Fold	0.00%
MSE	0.00%
MCC Fold	100%
Log. Loss	0.06%
F1 Score	100%

C. Comparison of the experiment results

1) Attack phases:

The comparison result between each attack phase shows a slight difference between them. See Table III.

2) Layers:

In this test, the average values of classification accuracy, AUC and F1 score are considered for each layer to be compared with averages for the host layer. Table VII emphasizes the superiority of the network layer, which has 100% accuracy compared with the host layer with minor differences.

The results are undeniable as values for MSE and Loss dropped for the network layer from 0.0060 to 0.0000 for the MSE, and from 0.0035 to 0.0006 for the Log. Loss.

TABLE VII. COMPARISON OF RESULTS FOR LAYERS.

Layer	Accuracy	AUC	F1 Score	MSE	Log. Loss
Host layer	100%	100 %	100%	0%	1%
Network layer	100%	100 %	100%	0%	0%

3) Methods:

Both k-fold cross-validation and Holdout methods used in the last experiment on the network layer. Table VIII shows the results. These results indicate the same accuracy level for both method despite the class.

This similarity indicates no differences in classification and detection accuracy as well as predictions of both used botnets (Mirai, and Gafgyt). On the other hand, We found that precision is (100% and 99.99%) and recall is (100% and 100%), respectively.

TABLE VIII. COMPARISON OF NETWORK LAYER METHODS.

		Mean			Std. Dev.		
Method	**Class**	**Accuracy**	**F1 Score**	**MSE**	**Accuracy**	**MSE**	**F1 Score**
K-fold	Binary	100%	100 %	0%	0%	0%	0%
Holdout	Multi	100%	100 %	0%	0%	0%	0%

D. Comparison with related works

Table IX compares our method with three similar approaches for IoT malware classification. The results for all other approaches maintain a lower accuracy and TPR than our approach does in the network layer. Moreover, none of the previous approaches achieves similar overall results in such a way that our approach does.

TABLE IX. COMPARISON WITH RESULTS FROM RELATED WORKS.

	Our Method		Meidan et al. [15]	Su et al. [13]	Alejandre et al. [27]
Devices	IoT		IoT	IoT	IoT
Algorithm	ML		DL	DL	ML
	LightGBM		DAE	CNN	GA
Class	Binary	Multi-class	Binary	Binary	--
Accuracy	100%	100%	--	94.00 %	--
F1 Score	100%	100%	--	--	--
MSE	0.00 %	0.00%	--	--	--
Precision	100%	99.99 %	--	93.42 %	--
Recall	100%	100%	--	94.67 %	--
TPR	100%	100%	100%	94.67 %	99.46%
FPR	0.00 %	0.00%	0.70%	5.33%	0.57%

XI. CONCLUSION AND FUTURE WORK

The results in this study prove that the advanced ML algorithms and DL do not necessarily result in better solutions. Conversely, it may increase the problem's complexity.

For example, the LightGBM algorithm achieved nearly 100 percent accuracy, which proves this ML algorithm's efficiency over DL strategies. Practically anyone who has studied AI knows that classical ML algorithms are less complicated than deep learning. In addition, the improved accuracy of using classical ML alternates with anti-malware producers to use deep learning.

The results show our approach's ability to detect botnets attacks with the same high accuracy regardless of its family. Our approach also provides an instant, accurate and straightforward method for early detection of IoT botnets at the network level before any more IoT devices are infected. Network level means anti-malware can analyze and detect malware from the network's sniffed traffic despite the traffic source, destination host, device type, device brand, or device operating system.

In particular, LightGBM accomplishes convincing results in the network layer using a multi-class dataset which is more precise than in the context of the host layer. Hence, in both host or network layers, the suggested algorithm won the race to be the light, efficient and accurate classifier for malware machine learning.

For the best of our knowledge, Anti-malware vendors can implement the system presented on all actual IoT devices. Using these results in the anti- system will allow for real-identification and termination of the threat before the damage occurred.

Based on the promising results of the proposed IoT botnets cyber-attacks algorithm, researchers are recommended to pay more attention to involve computer malware as well. Despite the high-performance levels achieved through the proposed approach, the metrics discussed in the previous sections did not take time performance into consideration. The system proposed will have higher potential if several studies improve its performance over time.

At last, we advise using computer malware dataset to conduct the same experiment and consider improving the performance of the time.

REFERENCE

1. CybeRisk, "Passive Attacks vs Active Attacks," CybeRisk, 2018. [Online]. Available: https://www.cyberisk.biz/passive-attacks-vs-active-attacks/.
2. M. Al-Kasassbeh, S. Mohammed, M. Alauthman and A. Almomani, "Feature Selection Using a Machine Learning to Classify a Malware," in Handbook of Computer Networks and Cyber Security, Springer, Cham, 2020, pp. 889-904.
3. C. Hoffman, "How Antivirus Software Works," How-To Geek, 26 September 2016. [Online]. Available: https://www.howtogeek.com/125650/htg-explains-how-antivirus-software-works/.
4. Comodo Security Solutions, Inc, "How Antivirus Works?," Comodo Security Solutions, Inc, [Online]. Available: https://antivirus.comodo.com/how-antivirus-software-works.php.
5. E. Gandotra, D. Bansal and S. Sofat, "Malware Analysis and Classification: A Survey," Journal of Information Security, vol. 5, no. 2, p. 9, 28 Mar 2014.
6. P. Security, "27% of all recorded malware appeared in 2015," Panda Security, 25 January 2016. [Online]. Available: https://www.pandasecurity.com/mediacenter/press-releases/all-recorded-malware-appeared-in-2015/. [Accessed 2018].
7. K. Angrishi, "Turning Internet of Things (IoT) into Internet of Vulnerabilities (IoV): IoT Botnets," vol. 1, 13 Feb 2017.
8. A. Spognardi, M. D. Donno, N. Dragoni and A. Giaretta, "Analysis of DDoS-Capable IoT Malwares," in 2017 Federated Conference on Computer Science and Information Systems (FedCSIS), Prague, Czech Republic, 2017.
9. M. Al-kasassbeh, M. Almseidin, K. Alrfo and S. Kovacs, "Detection of IoT-Botnet Attacks Using Fuzzy Rule Interpolation," Journal of Intelligent & Fuzzy Systems, vol. 38, no. 1, 2020.
10. G. Kambourakis, C. Kolias and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," in MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, USA, 2017.
11. A. Marzano, D. Alexander, O. Fonseca, E. Fazzion, C. Hoepers, K. Steding-Jessen, M. H. P. C. Chaves, I. Cunha, D. Guedes and W. M. Jr., "The Evolution of Bashlite and Mirai IoT Botnets," in 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 2018.
12. C. Kolias, G. Kambourakis, A. Stavrou and J. Voas, "DDoS in the IoT: Mirai and other botnets," Computer, vol. 50, no. 7, pp. 80-84, 07 Jul 2017.
13. J. Su, D. V. Vargas, S. Prasad, D. Sgandurra, Y. Feng and K. Sakurai, "Lightweight Classification of IoT Malware Based on Image Recognition," Proceedings - International Computer Software and Applications Conference, vol. 2, pp. 664-669, 11 Feb 2018.
14. Read the Docs, Inc & contributors, "LightGBM's documentation!," Read the Docs, Inc & contributors, 7 Feb 2019. [Online]. Available: https://media.readthedocs.org/pdf/lightgbm/latest/lightgbm.pdf.
15. Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher and Y. Elovici, "N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders," IEEE Pervasive Computing, vol. 17, no. 3, pp. 12-22, 11 Oct 2018.
16. S. S. Chawathe, "Monitoring IoT Networks for Botnet Activity," in 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), 2018.
17. Z. K. Zhang, M. C. Y. Cho, C. W. Wang, C. W. Hsu, C. K. Chen and S. Shieh, "IoT security: Ongoing challenges and research opportunities," in IEEE 7th International Conference on Service-Oriented Computing and Applications, SOCA 2014, Matsue, Japan, 2014.
18. I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," Business Horizons, vol. 58, no. 4, pp. 431-440, 2015.
19. M. Shobana and S. Rathi, "IoT Malware: An Analysis of IOT Device Hijacking," International Journal of Scientific Research in Computer Science, Computer Engineering, and Information Technology, vol. 3, no. 5, pp. 2456-3307, 2018.
20. K. L. Lueth, "State of the IoT 2018: Number of IoT devices now at 7B – Market accelerating," IoT Analytics, 8 Aug 2018. [Online]. Available: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/. [Accessed 28 Apr 2019].
21. L. Columbus, "2017 Roundup Of Internet Of Things Forecasts," Forbes, 10 Dec 2017. [Online]. Available: https://www.forbes.com/sites/louiscolumbus/2017/12/10/2017-roundup-of-internet-of-things-forecasts/#2aa5e28c1480. [Accessed 28 Apr 2019].
22. D. M. Mena, I. Papapanagiotou and B. Yang, "Internet of Things: Survey on Security and Privacy," Information Security Journal: A Global Perspective, vol. 27, no. 3, pp. 162-182, 04 Apr. 2018.
23. A. Pal, "The Internet of Things (IoT) – Threats and Countermeasures," CSO Online, 20 May 2018. [Online]. Available: https://www.cso.com.au/article/575407/internet-things-iot-threats-countermeasures/. [Accessed 28 Apr 2019].
24. A. Costin and J. Zaddach, "IoT Malware: Comprehensive Survey, Analysis Framework and Case Studies," in Black Hat Conference, Las Vegas, 2018.
25. C. Willems, F. Freiling and T. Holz, "Toward Automated Dynamic Malware Analysis Using CWSandbox," IEEE Security & Privacy, vol. 5, no. 2, pp. 32-39, 2 Apr 2007.
26. E. Bertino and N. Islam, "Botnets and Internet of Things Security," Computer, vol. 50, no. 2, pp. 76-79, 06 Feb 2017.
27. F. V. Alejandre, N. C. Cortés and E. A. Anaya, "Feature selection to detect botnets using machine learning algorithms," in International Conference on Electronics, Communications and Computers, Cholula, Mexico, 2017.
28. M. De Donno, N. Dragoni, A. Giaretta and A. Spognardi, "DDoS-Capable IoT Malwares: Comparative Analysis and Mirai Investigation," Security and Communication Networks, vol. 2018, p. 30, 18 Feb 2018.
29. J. Leonard, S. Xu and R. Sandhu, "A Framework for Understanding Botnets," in International Conference on Availability, Reliability and Security, Fukuoka, Japan, 2009.
30. M. K. Alsedrah, "Artificial Intelligence," Dec 2017.
31. M. Copeland, "What's the Difference Between Artificial Intelligence, Machine Learning, and Deep Learning?," NVIDIA, 29 Jul 2016. [Online]. Available: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/. [Accessed 28 Apr 2019].
32. T. e. System, "What is Machine Learning? A definition," Expert System, [Online]. Available: https://www.expertsystem.com/machine-learning-definition/. [Accessed 28 Apr 2019].
33. E. Gandotra, D. Bansal and S. Sofat, "Malware Analysis and Classification: A Survey," Journal of Information Security, vol. 5, no. 2, p. 9, 28 Mar 2014.
34. H. Zhao, M. Xu, N. Zheng, J. Yao and Q. Hou, "Malicious Executables Classification Based on Behavioral Factor Analysis.," in International Conference on e-Education, e-Business, e-Management, and e-Learning, Sanya, China , 2010.
35. M. Sidana, Technology Consultant and Sifium Technologies, "Types of classification algorithms in Machine Learning," Medium Co., 28 Feb 2017. [Online]. Available: https://medium.com/@Mandysidana/machine-learning-types-of-classification-9497bd4f2e14. [Accessed 28 Apr 2019].
36. R. Islam, R. Tian, L. M. Batten and S. Versteeg, "Classification of malware based on integrated static and dynamic features," Journal of Network and Computer Applications, vol. 36, no. 2, pp. 646-656, Mar 2013.
37. S. L. Darshan, M. A. Kumara and C. Jaidhar, "Windows Malware Detection Based on Cuckoo Sandbox Generated Report Using Machine Learning Algorithm," in 2016 11th International Conference on Industrial and Information Systems (ICIIS), Roorkee, India, 2016.
38. N. Aman, Y. Saleem, F. H. Abbasi and F. Shahzad, "A Hybrid Approach for Malware Family Classification," in International Conference on Applications and Techniques in Information Security, Nanning, China, 2017.
39. M. Alazab, S. Venkataraman and P. Watters, "Towards Understanding Malware Behaviour by the Extraction of API Calls," in 2010 Second Cybercrime and Trustworthy Computing Workshop (CTC), Ballarat, VIC, Australia, 2010.
40. A. Sami, B. Yadegari, H. Rahimi, N. Peiravian, S. Hashemi and A. Hamze, "Malware Detection Based on Mining API Calls," in SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing, Sierre, Switzerland, 2010.

41. A. Kumar, "A Framework for Malware Detection with Static Features using Machine Learning Algorithms," Research Gate, Aug 2017. [Online]. Available: https://www.researchgate.net/publication/329450675_A_Framework_for_Malware_Detection_with_Static_Features_using_Machine_Learning_Algorithms?amp%3BenrichSource=Y292ZXJQYWdlOzMyOTQ1MDY3NTtBUzo3MDA4NDY3OTQ4ODMwNzJAMTU0NDEwNjMwNTEzMQ%3D%3D&amp%3Bel=1_x_2&amp%. [Accessed 28 Apr 2019].
42. Microsoft (MSDN), "Windows API Index," Microsoft, [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/ff818516(v=vs.85).aspx. [Accessed 28 Apr 2019].
43. C. Sandbox, "Cuckoo Sandbox," Cuckoo Sandbox, 2010-2017. [Online]. Available: https://cuckoosandbox.org/. [Accessed 28 Apr 2019].
44. V. M. Alvarez, "YARA's documentation," Victor M. Alvarez, [Online]. Available: https://yara.readthedocs.io/en/v3.4.0/index.html#. [Accessed 28 Apr 2019].
45. E. Fonseca, D. Bogdanov, R. Gong, E. Gomez, O. Slizovskaia and X. Serra, "Acoustic Scene Classification by Ensembling Gradient Boosting Machine and Convolutional Neural Networks," in Workshop on Detection and Classification of Acoustic Scenes and Events, Munich, Germany, 2017.
46. X. Meng, Z. Shan, F. Liu, B. Zhao, J. Han, H. Wang and J. Wang, "MCSMGS: Malware Classification Model Based on Deep Learning," in 2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Nanjing, China, 2018.
47. "VX Heaven," vxheaven.org, 2016. [Online]. Available: http://83.133.184.251/virensimulation.org/.
48. G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T.-Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in Advances in Neural Information Processing Systems 30, 2017.
49. Ramnathv and Gdequeiroz, "Gradient Boosting Machines," 25 Sep 2017. [Online]. Available: https://github.com/ledell/useR-machine-learning-tutorial/blob/master/gradient-boosting-machines.Rmd.
50. P. Khandelwal, "Which algorithm takes the crown: Light GBM vs XGBOOST?," Analytics Vidhya, 12 June 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/. [Accessed 2018].
51. xgboost, "Introduction to Boosted Trees," xgboost.readthedocs.io, [Online]. Available: https://xgboost.readthedocs.io/en/latest/tutorials/model.html.
52. xgboost, "Introduction to Boosted Trees," xgboost.readthedocs.io, [Online]. Available: https://xgboost.readthedocs.io/en/latest/tutorials/model.html. [Accessed 28 Apr 2019].
53. S. Xiaolei, L. Mingxi and S. Zeqian, "A novel cryptocurrency price trend forecasting model based on LightGBM," Science Direct Elsevier Inc., vol. In press, 27 Dec. 2018.
54. E. Fonseca, R. Gong, D. Bogdanov, O. Slizovskaia, E. Gomez and X. Serra, "Acoustic Scene Classification by Ensembling Gradient Boosting Machine and Convolutional Neural Networks," in Workshop on Detection and Classification of Acoustic Scenes and Events, Munich, Germany, 2017.
55. Readthedocs.org, "LightGBM's documentation!," Read the Docs, Inc & contributors, 7 Feb 2019. [Online]. Available: https://media.readthedocs.org/pdf/lightgbm/latest/lightgbm.pdf. [Accessed 28 Apr 2019].
56. P. Khandelwal, "Which algorithm takes the crown: Light GBM vs XGBOOST?," Analytics Vidhya, 12 June 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/. [Accessed 28 Apr 2019].
H. Zhang, S. Si and C.-J. Hsieh, "GPU-acceleration for Large-scale Tree Boosting," arXic.org, 26 Jun 2017.
57. "Scikit-learn: Machine Learning in Python," INRIA and others., [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation.
[59] Scikit-Learn.org, "Scikit-learn: Machine Learning in Python," INRIA and others., [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation. [Accessed 28 Apr 2019].

AUTHORS PROFILE

Dr Mohammad Abbadi is an associate Professor in Department of Computer Science, Faculty of Information Technology, Mutah University. He got his Ph.D. in computer science from The George Washington University in 2000. His research interest is multimedia, networks, blended and e-learning, data compression, and image processing.

Ahmed M. Al-Bustaji graduated McS from Faculty of Information, Mutah University, Jordan in 2019. He is currently the Manager of Network an Systems Administartion in Computer Center at Mutah University. His experience include network implementation and administration, security administration and tuning and all well known systems administration.

Mouhammd Al-kasassbeh gradu-ated from the school of computing, Portsmouth University, UK in 2008. He is currently a full professor in the Computer Science Department at Princess Sumaya University for Technology. His research interests include network traffic analysis, net-work fault detection, network fault and abnormality classification and machine learning in the area of computer networking and network security.