Ingénierie des Systèmes d'Information
Vol. 27, No. 6, December, 2022, pp. 1029-1038
Journal homepage: http://iieta.org/journals/isi

Implementation of Blockchain with Machine Learning Intrusion Detection System for Defending IoT Botnet and Cloud Networks

Swapna Siddamsetti[1,2*], Muktevi Srivenkatesh[1]

[1] Department of Computer Science, GITAM School of Science, GITAM Deemed to be University, Vishakapatnam 530045, India
[2] Department of Computer Science and Engineering, Neil Gogte Institute of Technology, Hyderabad, Telanagana 500039, India

Corresponding Author Email: swapnangit2021@gmail.com

https://doi.org/10.18280/isi.270620

Received: 8 August 2022
Accepted: 4 November 2022

Keywords:
blockchain, cloud systems, machine learning, IoT botnets, intrusion detection system

ABSTRACT

Significant research has been done on combining intrusion detection and blockchain to increase data privacy and find both current and future threats. This research suggests a machine blockchain framework (MBF) in order to provide distributed intrusion detection with security and blockchain with privacy with the help of smart contracts in IoT networks. An XGBoost algorithm was implemented to work with sequential network data and the intrusion detection approach is explored using the N-BaIoT dataset. In order to protect the network against known malware threats (Mirai, Gafgyt, or Bashlite), the IoT malware attack prediction model created in this study offers a deterrent strategy based on the network traffic statistics. On the other hand, the models need to be taught to recognize new varieties of malware. In this work, we observe how different machine learning models, like Random Forest algorithm and proposed XGBoost algorithm, can accurately predict the infected malware in certain traffic instance. However, we provide a honeypot-based strategy that employs machine learning techniques for the detection of malware in this study. Using data from an IoT Botnet as a dataset helps train a machine learning model in a way that is effective and changes over time.

1. INTRODUCTION

Internet users have been dynamically attacked over the past 10 years by worms and viruses that are distributed over email. This is not because the Internet is significantly more secure; rather, it is more likely a result of attackers' focus shifting to infecting and controlling victim PCs, a type of threat that offers greater opportunity for individual gain and offensive capabilities. As per a Gartner report [1], by 2030, the number of Internet of Things (IoT) devices linked over the internet is predicted to reach 50 billion. These IoT devices are not just used in electronic gadgets; they have also made their way into other industries, including smart agriculture, hospitals, and homes [2]. Several strategies are used to attack threats and give users access to a secure network. In particular, blockchain, machine learning (ML), and deep learning (DL) technologies are studied in this paper as intrusion detection strategies. By using either the signature-based or anomaly-based types of cyber-analysis, machine learning may be used to identify attacks. Some publications in the literature [3] employ signature-based approaches to identify attacks. The four different machine learning algorithms are used to understand the features of some frequent attacks in the paper. The detection of botnet patterns in the network traffic was employed by using signature-dependent methods [4]. These methods were also employed to detect infected workstations by analysis of botnet network traffic patterns. The two fundamental weaknesses of signature-dependent techniques are that they cannot identify previously undetected phishing attacks and that their effective usage necessitates regular human updating of attack traffic signatures. Anomaly-based detection is the second class of detection techniques. This class simulates typical network behavior, and any unusual behavior is regarded as an attack. Many IoT gadgets, including security cameras with IP-enabled systems, printers with wireless connectivity, baby monitors, etc., have weak security configurations that Mirai took advantage of to turn the above-mentioned devices into harmful bots. The above harmful bots joined together to form a botnet that launched many DDoS attacks against a DNS provider. Later, the Mirai botnet's source code was made public on Hackers Forum, and it was looked into how these IoT devices came to be a target of the Mirai botnet [5]. Traditional DDoS attacks are usually stopped by installing expensive, complex equipment or using outside service providers who charge a lot for their services [6, 7]. To overcome the issue with centralized DDoS mitigation methods, Blockchain introduces a database which is distributed and depends on the network with peer-to-peer connectivity, which presents a high intensity of trust which also includes dependability. A node broadcasts a new block to the rest of the network after receiving it. After verifying the block, each node that has received it distributes it to other nodes. The block can only be added to the blockchain by the miners.

Several industries are using blockchain to facilitate trust and data privacy. This is because it supports parties to make different types of transactions in order to transfer information by retaining trust of the users. Intrusion Detection System with blockchain technology can be used together to detect different kinds of threats in order to protect personal and sensitive data of the clients stored on cloud technology and also IoT networks. The Collaborative Intrusion Detection Systems (CIDSs) are assumed as economical and scalable for inspecting different cloud nodes. The major challenges in the cloud technologies will be the storage capacity to maintain the privacy of data and provide assurance for trust management amongst multi-cloud service providers [8].

Privacy-preserving mechanisms are frequently implemented to convert, alter, or hide original material in order to secure it from unwanted access, along with the process of finding attack events employing CIDSs in the cloud services [9]. The blockchain structure and smart contract technologies are popular forms for preserving privacy, which provides cloud elements with authentication and integrity. Numerous security weaknesses in blockchain attacks and related technologies, like bitcoin and Ethereum, have been recently brought to light [10]. As this article says, it is important to use machine learning to create an IDS-based blockchain system in the cloud. This will help find cyberattacks and protect the data of IDS engines that are installed at different cloud nodes.

We can summarize our contributions through this article as:

- On the basis of a machine blockchain framework (MBF), a system for detecting intrusions has been proposed. This system keeps data secure and private in cloud networks.
- The detection of cyberattacks in IoT networks can be made better by looking at how efficiently machine learning algorithms will be able to work on a recent type of IoT dataset.
- Get the current features from the datasets and choose the most useful ones to improve the performance of a machine learning algorithm.

The proposed system and prescribed methods have been evaluated with the help of data sets of UNSW-BN15 and N-BaIoT of the network, the performance of the system is compared with different intrusion detection techniques to determine its effectiveness while deploying it to the cloud.

2. RELATED WORKS

The implementation of machine learning algorithms has been the subject of in-depth study in the past [11], and a number of academic articles on the applications of data mining and artificial intelligence for intrusion detection have been published [12]. Several studies have used blockchain technology to boost CIDS trust in networks and cloud platforms. As an illustration, Dawit et al. [13] examined several approaches for grouping CIDSs with blockchains. The significance of CIDSs has been increased by employing the blockchain technology as proposed by the authors in different articles. It was emphasized that the benefits of blockchain technology for CIDSs include the capacity to trust one another as well as techniques for accountability and consensus. Saldamli et al. [14] also talked about how important it is to use blockchain and how its theoretical techniques could be used to secure CIDSs.

Li et al. [15] fully implemented these issues and argued for the benefits of fusing trust management with blockchain technology. Blockchain technology enables communication in the networks without any trusted party to maintain the integrity of shared data, and trust management can assist in assessing each node's trustworthiness. Then a simple model for cooperative intrusion detection with blockchain in SDN was presented. In this study, they considered challenge-based CIDS to asses basic performance of our framework against external and internal threats. The outcomes demonstrate the framework's feasibility and efficiency. Putra et al. [16] proposed a CIDS with a decentralized environment and emphasized the value of developing trust among CIDS nodes. To implement the suggested fix, every CIDS node communicates detection rules with other nodes to aid in the identification of novel intrusion types. The design uses a store with decentralized structures to handle the shared detection rules for trustworthy mechanisms, guaranteeing scalability and offloading the trust computation to the blockchain. Implementing the approach in a lab-scale testbed showed that it works and meets the expected benchmarks for the Ethereum platform [17].

By combining these two, Li et al. [18] expected to offer a blockchain-based challenge-based CIDN system. As per the evaluation criteria, blockchain technology has the capability to improve the trust management (i.e., enhancing the detection of malware in the insider nodes) and alarm aggregation aspects of challenge-based CIDNs (i.e., identifying untruthful inputs). With the help of traffic fusion and aggregation to manage and eliminate harmful traffic, Meng et al. [19] focused on blockchain-based SDN and created a filter called the BSDNFilter, an IDS-based security mechanism. The simulated tests in a real blockchain-based SDN system shown that BSDNFilter can filter threats of flooding better than models that were made in collaboration with an IT organization. Li et al. [20] created a framework known as BlockCSDN, for collaborative blockchain-based intrusion detection in SDN, and used the CIDS of challenge-based as a case study. The investigated results under both internal and external threats prove that CIDSs and SDNs can benefit from employing blockchain technology for avoiding vulnerability and security.

In the smart grid, for MMG systems, a new type of collaborative intrusion detection (CID) solution utilizing blockchain was explained in this study by Hu et al. [21]. The technique is created without the necessity for a central server or trusted authority due to the message exchange of blockchain, which also allows for a collaborative improvement in intrusion detection accuracy. The proposal was designed with the generating mechanism which integrates both periodic patterns and trigger patterns which produce detection target for the CID, i.e., a proposal. Using consensus procedure, a CID is produced from the generated proposals and the MMG correlation model. The final detection results from CID are successively collected. A single microgrid is motivated to take part in consensus by the application of an incentive mechanism. A case study on an MMG system is used to show the efficacy of the suggested strategy.

3. PROPOSED WORK

An MBF is suggested to recognize cyberattacks and safeguard cloud-based data. The suggested framework's systematic architecture is made up of four key parts: As mentioned below and shown in Figure 1, the four components are: 1) Cloud provider; 2) Blockchain and smart contracts that protect privacy; 3) Central Coordinator Unit (CCU); and 4) Intrusion Detection System (IDS). Every component has a role to play. Cloud service providers often exchange data or notify intrusion occurrences involving data privacy and confidentiality concerns.

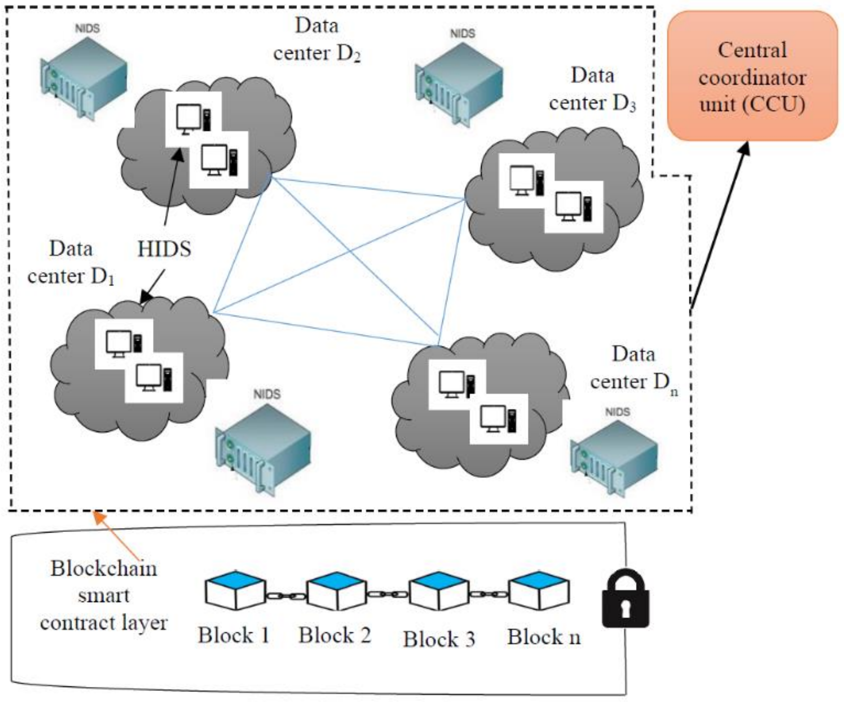

Figure 1. Proposed cloud-based system architecture for cloud data centers using the MBF installed at NIDS

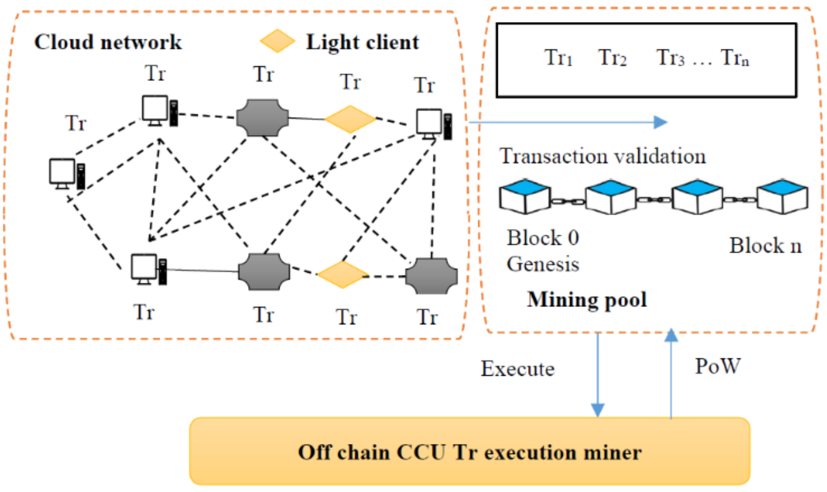

Figure 2. A DBF transaction's period of security event aggregation

The CCU entity is a mechanism for storing IDS audit logs and warnings, acting as an isolated environment operating in parallel with the blockchain system. The blockchain and smart contract layer follows, which is intended to provide integrity and authenticity to data and alarms created by NIDS and HIDS. Finally, the IDS entity facilitates the validation of sessions operating on the cloud transaction network and guarantees whether they follow the rules set.

There are several cloud companies and data centers available which are dented as the data centers D1, D2, D3, …, Dn. They are detected as being part of a blockchain network's cloud network. These organizations should have adequate cloud resources to give their customers' entities to them. In particular, a distributed digital ledger that contains all transactions cloud-based was included in the block chain with privacy-preservation and smart contract systems by means of a consortium blockchain. This object is duplicated and placed on every node of the multi-cloud network. It may be a CCU, a data center, or a single host. The proposed MBF is built in a data format similar to Bitcoin's. However, mining new blocks should be suitably rewarded during a method of adding a block to the blockchain. Miners earn from new coins minted with each new block and transaction fees from any transactions contained in the block. The proof of work mechanism, contained in the new block and functions as proof to gain reward and the privilege to register bitcoin transactions on the blockchain, is the solution to implement and serves as the basis for bitcoin's security framework. IDS audit logs and alarms are stored in the CCU, which serves as a SIEM tool. Utilizing the CCU's capabilities allows for the analysis, filtering, and correlation of incoming data from various sources (such as cloud data centers) in order to distinguish between typical and unusual events. With smart contract platforms that facilitate distributed data interchange and migration across multi-cloud services, network managers could quickly reduce threats and raise security awareness among blockchain cloud network participants. IDS based on an MBF guarantee privacy and data security in cloud networks. The cloud transaction network's frames are verified by the IDS entity, which also makes sure they follow the established standards. They are made up of several IDSs which are placed with in the wide-spread networks or hosts of individuals and interact with each another in order to predict coordinated cyberattacks and avoid potential illegal activities. The CIDS, which requires cooperation between different nodes, would make it easier to find complex intrusions like DoS, DDoS, and malicious insiders.

3.1 Intrusion detection system-based on machine learning

The proposed IDS's heterogeneous model and virtualized technologies need to be constructed and deployed on a computer infrastructure. Transaction logs and related notification data on harmful software activity may be exchanged among several cloud suppliers. However, the usefulness of the shared data was limited in this case of IDS systems which are not trusted and properly combined. When creating a cloud-based IDS, the specific concept of cloud computing creates a number of difficulties. The idea of cloud computing itself makes it hard to make an IDS that is based on the cloud. These characteristic features include efficient insider and outsider attack detection with a reduced risk of false negative values (FNs) and false positive values (FPs). IDS monitors' raw warning data is saved in a blockchain transaction that is distributed across participating network nodes.

As a result, they can be kept in the CCU for a long time within the prescribed directory for the forensic investigation and compliance. Before permanently adding the transactions into the ledger blocks, each and every participant (i.e., datacenter in the cloud) uses the consensus procedure in order to guarantee the integrity of the transactions. By ensuring that the CCU database only contains legitimate warnings, this technique offers resistance to tampered data and transparency so that the providers of cloud services can see where their migrated data is located on the blockchain ledger. In order to better supervise identity, basically in an open public access blockchain, participating nodes of the datacenter of the IDS have to build a smart contract which will be linked to other nodes with the help of a registry-based type. The characteristics of any smart contract are copied and disseminated by all network nodes. These can only do the acts they were created and only when the specified criteria are satisfied. They can no longer be edited after they have been published to the network. A high-level language known as solidity is required to develop the most prominent and successful language for creating a smart contract. Formal approaches, programming language semantics, and cryptography are some of the other cases.

The smart contract is sorted into the cloud-based blockchain of each IDS. Based on stated regulations inside the smart contract, like country code, the area, their zone, and type of organization, each and every participating IDS node can be able to select other partner nodes with whom they want to transfer the information. A new datacenter or cloud vendor must first register with the CCU to get an identification number before they can use the CID system (private and public keys). Figure 2 presents our DBF's architectural layout. MBF offers complete knowledge of reported security incidents, rapid data interchange across IDS, alarm correlation, and predictive analytics. MBF is the data protection mechanism that accepts a blockchain transaction as input, initializes the genesis block, verifies for legitimate blockchain transactions, processes off-chain, computes hash using SHA-256, and thus generates newhashBlock. Furthermore, only the resource-constrained device nodes need to keep a ledgers tiny amount which has to be processed by the distributed CCU in off-chain mode according to the blockchain’s built-in capabilities for smart contract-based systems. As a result, IoT devices may function as simple users to confirm and share the accuracy of alert events throughout the blockchain network. The CCU with distributed property, on the other hand, serves as the miner node and keeps the whole blockchain ledger. It requires a hash value of high rate to execute all the incoming transactions. Lastly, the proposed MBF results are made unchangeable, reliable by getting them and putting them on the blockchain in the correct order.

3.2 Machine learning models for intrusion detection systems

In this part the proposed MBF framework is presented. For the purpose of identifying the blockchain-based cloud network attacks, LR, RF, and XGBoost are used as IDS. It may be observed as a strong artificial neural network that processes sequence input by passing internally stored information. Figure 3 provides an illustration of the suggested IDS methodology details. There are four primary phases for training and validating the LR, RF, and algorithm: the partitioning of training and testing sets; data standardization; model construction; and dataset inputs. Data is uploaded for the first stage. This contains datasets like the N-BaIoT dataset, for instance. These were picked because they cover a wide range of security incidents and are reputed and reliable cloud network observations. In the second phase, the complete data sets are partitioned into two parts called training datasets and testing datasets to establish how well the ML algorithms can discriminate between attacks and regular observations. In order to successfully fit data using ML models, the training datasets and testing datasets are normalized according to certain range, like [0, 1], in the third step. It was made so that the proposed XGBoost model could be trained and tested to see how well it could classify attack events. The Python-based Keras deep learning package was used to create the XGBoost model. Every data collection is split into three categories: 1) training phase, 2) validation phase, and 3) testing phase, with respective percentages of 60%, 20%, and 20%. This results in the highest accuracy detection model during implementation. The trained model was put to the test by working on each row of the testing dataset. Rows that follow are then classified as either regular records or attack record datasets. Nine commercial IoT devices that were attacked by the Mirai and BASHLITE botnets provided the data for this dataset, which contains genuine network traffic information.

A. Dataset:

The dataset is produced by separating harmful from benign data [16]. The dataset displays actual traffic data taken from seven Internet of Things (IoT) devices that were infected with the Mirai virus. Each IoT device has a unique value assigned to it in the dataset. Although each dataset shares 115 features, the total size of the collection differs depending on the device. There are 23 features in each of the five-time window that make up the 115 features. The "N-BaIoT" dataset, which records typical network traffic patterns, was used to train seven IoT devices. For example, the MI dir L5 function of weight displays traffic from MAC packets of the host and IP at a five-second interval. Similar to how HH dir L3 weight specifies traffic from the source host to the destination host at a three-second interval, HpHp dir L3 weight specifies the traffic from the source host port to the destination host port at a three-second interval. It contains 115 traffic features, eight separate attack classes, two botnets, and one benign botnet. The attributes used for training include DoS HTTP, DoS UDP, DoS TCP, Backdoor, Exploits, Analysis, Reconnaissance, Worms, Shellcode, Ports canning, OS fingerprinting, DDoS HTTP, DDoS UDP, DDoS TCP, Keylogging, and Datatheft. All the properties are independent.

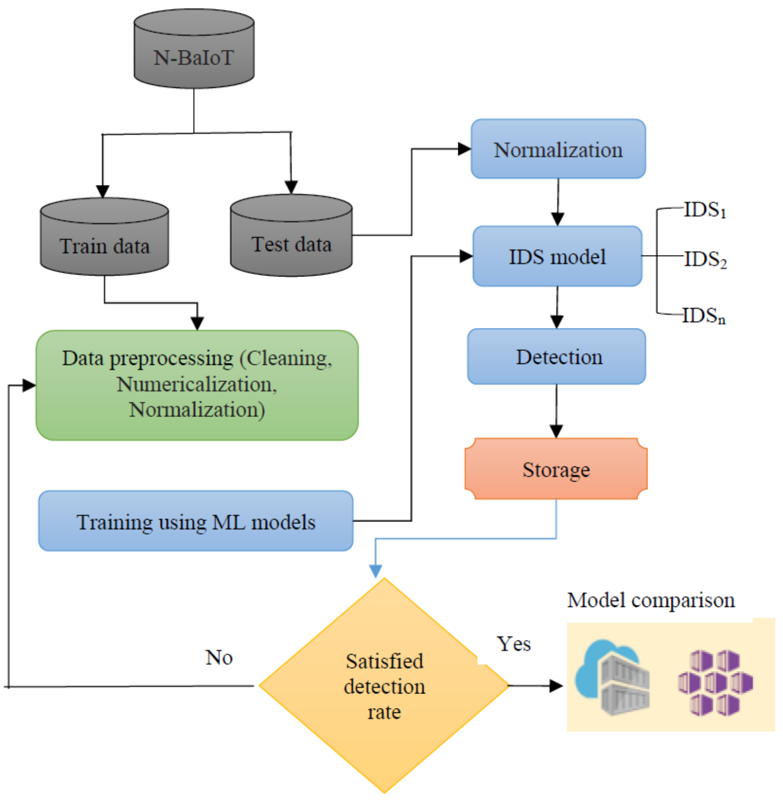

Figure 3. Proposed XGBoost model for intrusion detection

B. Data preprocessing:

We employed the max-min normalization, in which the input data are to be mapped according to the range of the values between 0 and 1 [22], as the preprocessing step. For the system's implementation, we employed various Python libraries, including Scikit-learn, Keras, and TensorFlow. Scikit-learn is a supporting tool for effectively implementing a variety of machine learning methods. Additionally, it offers the ability to divide datasets into several subsets, including the ability to divide training and test datasets. We have partitioned the chosen dataset into training and testing datasets using this package. Additionally, we experimented with the help of a tree-based approach and naive bayes using this library. In this case, we use TensorFlow, which is a more complicated framework designed for distributed numerical computation with the help of data flow graphs. It can run on top of Keras, the high-level efficient neural network of the Python-API.

Data cleaning: The IoT Botnet dataset contains properties of a comparable type and quantity with somewhat different attack types. Before normalizing this dataset, the null values are taken out, the redundant indices are removed, and the data types are changed to the right type (float). This is done to avoid problems.

Numericalization: Each dataset's feature data types must be identified in advance for numericalization. As the N BaIoT Data Set dataset lacks a category value. By using label encoding, these category variables are transformed into numeric data. Label encoding uses little memory and produces the same dimension of the dataset as previously, while giving repeated labels the same values. So, label encoding is used to turn the category-type features in all four of these datasets into numbers. For label encoding in this study, we employed the scikit-learn library. The steps are as follows: (i) Create a LabelEncoder; (ii) Instance and save it in the LabelEncoder variable/object; (iii) Then, fit and transform are used to assign numerical values to categorical values, and the results are kept in a new column named "State N."

Normalizing: Following numericalization, the five relevant dataset's continuous numerical data, particularly those with a high range, are subjected to the StandardScaler technique for normalization. StandardScalar scales the data using a normal distribution by dividing by the standard deviation and subtracting the mean of 0.

$$\frac{x_i - mean(x)}{std.dev(x)} \tag{1}$$

The mean and standard deviation of the *x* features are determined for feature x_i, and xi is scaled using Eq. (1) above.

C. Classification:

The network detection mechanism classifies the specified features as either normal or attacks using classification algorithms. Three classifiers are used to analyse the data:

Logistic Regression (LR), Random Forest (RF), and XGBoost (a proposed technique), and the data is classified by botnet, attack, and device. Below, we have a detailed explanation of each.

Logistic Regression (LR):

A machine learning classifier called logistic regression (LR) is used to simulate the probability of a given class value. Although LR may be expanded to classify more than one class, in its most basic form, LR models a binary dependent variable using a logistic function. It is possible to employ a weighting of the factors to lower the level of the penalties from a complete penalty to a very small penalty. The L2 penalty is employed by the LogisticRegression class by default with a weighting of coefficients set to 1.0. Although not all solvers support all penalty types, the type of penalty may be chosen through the "penalty" parameter with values of "l1", "l2", or "elasticnet" (e.g., both). The "C" option allows the penalty's coefficient weighting to be modified [23].

Random Forest (RF):

The Random Forest algorithm belongs to the category of supervised learning. Random Forest is unique in that it may be used with both classification and regression methods. A random forest, to put it simply, is a collection of decision trees that use the bulk of their outputs to enhance prediction and outcomes. In this study, the Random Forest model is used to predict the accuracy of the Mirai or Gafgyt malware in the dataset. The LR and XGBoost models are also used to predict and compare the precision value, recall value, and F-1 score.

Algorithm XGBoost (Proposed model):

Test and train data were included in the dataset. To combine the data into one file, the two sets were first concatenated. The Python environment is used to run the combined data. The dataset has been opened on the Python platform, and the XGBoost algorithm for the N-BotIoT dataset was run by configuring different XGBoost-related settings. XGBoost was basically created by utilizing gradient-boosted decision trees to improve the speed and performance. It represents a method for applying boost to machines, or machine boosting. A general architecture of the XGBoost algorithm is shown in Figure 4. For tree boosting algorithms, XGBoost, also known as extreme gradient boosting, which helps in making use of all the available within the specified hardware and memory resources. It offers the advantages of algorithm improvement, model tuning process, and deployment in computing with different environments. The three main gradient boosting methods—Gradient Boosting, Regularized Boosting, and Stochastic Boosting—can all be carried out by XGBoost. In contrast to other libraries, it also enables the insertion and fine-tuning of regularization parameters.

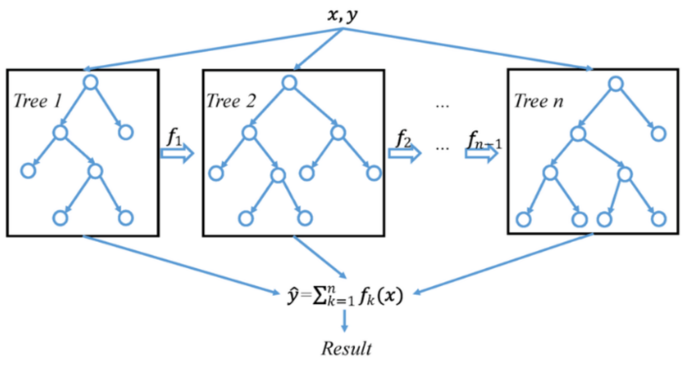

Figure 4. A general XGBoost architecture

The technique makes the best use of memory resources while being very successful at decreasing computation time. It has the special ability to execute boosting on additional data along with the trained model, which can be sparse-aware or can efficiently handle the missing values. It also provides a parallel structure for the tree construction along with several other notable features. The objective function has two parts: a training loss and a regularization. The training loss is the first portion of the objective function.

$$obj(\varphi) = TL(\varphi) + R(\varphi) \tag{2}$$

In Eq. (2), R stands for the regularization term, while TL represents training loss. Simply said, the TL is a measurement of the proposed model's predictive power. Regularization helps maintain the model's complexity within desirable bounds by removing issues like data overstacking or overfitting, which may result in the model having lower accuracy. All of the trees created from the dataset are simply added together by XGBoost, who then optimizes the outcome. At each stage, XGBoost adds a tree while attempting to optimize the learned tree (training). Taylor's theorem says that, at step t, the new objective function usually has up to the second order and looks like an expansion.

$$obj^{(t)} = \sum_{i=1}^{n} \left[m_i f_t(p_i) + \frac{1}{2} c_i f_t^{2}(p_i) \right] \tag{3}$$

where, m_i and c_i are taken as inputs.

The new tree that decides to join the model, the outcome reflects the intended optimization. In order to handle loss functions like logistic regression, XGBoost does it in this manner. To continue, regularization is crucial in determining the tree complexity defined in terms of *R*. Tree f(p) can be more precisely defined as:

$$f_t(p) = w_q(p), w \epsilon R^L, q: R^d \rightarrow \{1,2,3 \dots., L\} \tag{4}$$

In the above equation, the function *q* assigns leaves to the relevant data points, and *w* represents a vector of leaf scores (same score for data points utilizing the same leaf). The number of leaves is *L*. Given the intricacy of XGBoost,

$$R(f) = \alpha L + \frac{1}{2} \beta \sum_{j=1}^{L} w_j^2 \tag{5}$$

By using the normalization equation in the theorem, it is possible to get the proposed control objective function at the step *t*, also known as the t^{th} tree. The model will represent recently upgraded tree models and also provide an assessment of the quality (*p*) of the tree structures. Since it's hard to compute all tree options at the same time, the regularization, leaf scores, and objective function are all calculated at each level to define the tree structure. The gain of the tree structure is calculated at each and every level by splitting the leaf into a left leaf and a right leaf. The gain value of the current leaf is calculated by applying the regularization to all potential subsequent leaves. If the benefit is less than the additional regularization value, that particular branch is severed (by using the concept also called "tree pruning"). XGBoost uses this method to classify data and explore trees in depth. The extracted features are fed into the XGBoost model. Then, during every round, we employ the weighted column method to subsample columns, allowing XGBoost to concentrate on features that can better classify attacks. As a result, this strategy may lower the impact of redundancy features while improving classification accuracy and computational resource savings. As a result, accuracy values and other parameters can be calculated.

4. RESULTS AND DISCUSSION

The data collected was classified by type of device, by type of botnet, and then by type of attack, since we are categorizing each attempt in each botnet for each device.

Table 1. Provision_PT_737E Security Camera device results for Gafgyt attacks

Device-Name	Botnet-Name	Attack-Type	Alg.	Acc	Pre	Recall	F1 score
Provision_ PT_737E Security Cameras	Gafgyt	Junk, combo, scan, tcp, udp	LR	0.86	0.85	0.89	0.82
			RF	0.93	0.93	0.95	0.95
			XGBoost	0.97	0.97	0.96	0.96

Table 2. Provision_PT_737E Security Camera device results for Mirai attacks

Device-Name	Botnet-Name	Attack-Type	Alg.	Acc	Pre	Recall	F1 score
Provision_ PT_737E Security Camera	Mirai	Junk, combo, scan, tcp, udp	LR	0.85	0.84	0.88	0.81
			RF	0.92	0.92	0.94	0.94
			XGBoost	0.97	0.97	0.96	0.96

Table 3. Philips_B120N10 Baby Monitor results for Mirai attacks

Device-Name	Botnet-Name	Attack-Type	Alg.	Acc	Pre	Recall	F1 score
Philips_B120 N10 Baby Monitor	Mirai	Junk, combo, scan, tcp, udp	LR	0.86	0.85	0.89	0.82
			RF	0.94	0.96	0.97	0.90
			XGBoost	0.98	0.96	0.97	0.95

Table 4. SamsungSNH1011NWebcam device results for Gafgyt attacks

Device-Name	Botnet-Name	Attack-Type	Alg.	Acc	Pre	Recall	F1 score
SamsungSNH1 011NWebcam	Gafgyt	Junk, combo, scan, tcp, udp	LR	0.83	0.85	0.83	0.80
			RF	0.94	0.93	0.95	0.96
			XGBoost	0.98	0.96	0.97	0.98

Our early findings with the three classifiers—LR, RF, and XGBoost—did not perform well, mostly because of how severely unbalanced the data was. We employed nearly equal amounts of malicious and benign (normal) data to address this problem. Before executing the algorithms, a random sample of roughly half of the benign data was taken from the collection of malicious data and added to the malicious dataset.

The data was then normalised using z-scores as part of the preprocessing. The normalised data was then utilised to train and predict each of the classifiers (LR, RF, and XGBoost) as binary classifiers. 20% of the data was utilised for testing, while the remaining 80% was used for training. The classifiers were run using Scikit Learn. In this experiment, the commonly used multi-class performance measure "accuracy" is assessed in order to analyse the performances of the implemented models. Additionally, the precision values, recall values, and F1-score have been calculated. The qualitative model quality indices that are used to figure out these metrics are the true positive values, the true negative values, the false positive values, and the false negative values.

Accuracy is defined as the ratio of the model’s correct data (TP+TN) to the total data, given by

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FP} + \text{TN} + \text{FN})} \tag{6}$$

Attack Detection Rate (ADR): It often known as recall value, sensitivity, or: The number of positive instances (TP+FN) in the given dataset for which the model correctly identified a positive case as a true positive (TP) indicates how good the model is at recognising an attack.

$$\text{Recall} = \text{sensitivity} = \text{ADR} = \text{TPR} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{7}$$

Precision: This value represents the percentage of attack occurrences which are correctly categorised as attacks, or the number of real cases for all the positive cases (TP+FP) determined by the model (TP).

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{8}$$

F1-score: The F1-score value represents the relationship between precision_value and recall_value, given by:

$$\text{F1} - \text{score} = \frac{1}{\text{Precision}} + \frac{1}{\text{Recall}} \tag{9}$$

if the F1-score is maximum, then the proposed classification model will be more robust [24].

UCI collected data from 9 IoT devices in the three categories indicated below. Data is grouped by attack type for each and every device (Benign, Mirai, Bashlite). Benign traffic represents the normal traffic which is not being attacked by any botnet software. Malware that targets these IoT devices includes Bashlite and Mirai. The botnets that are used in this research are called Gafgyt and Bashlite. The information shown below is gathered for each device:

1) Benign (Normal)
2) Mirai malware attack
3) Gafgyt Malware attack

Figure 5 comparison of the Provision_PT_737E Security Camera IoT device's performance evaluation findings may be understandable given that it is based on Gafgyt botnet attacks which are predicted based on above Table 1 to Table 4. We have discussed these findings on the Gafgyt botnet and its attacks using the three classifiers Logistic regression (LR) [25], Random Forest (RF) [26], and proposed XGBoost. In this bar graph, each pair of colors represents a combination of the several performance indices that were used in this study. Although LR has attained an accuracy value of 0.86, lower than our proposed XGBoost model's 0.97, we assess three ML approaches in this analysis. These results also reveal a very high attack detection rate of over 0.96 for XGBoost as related to the 0.89 of LR, and 0.95 of RF models. Also, the proposed XGBoost model precision is 0.97 and has an F1-score of 0.96 in almost all Gafgyt botnet attacks, which is very close to one.

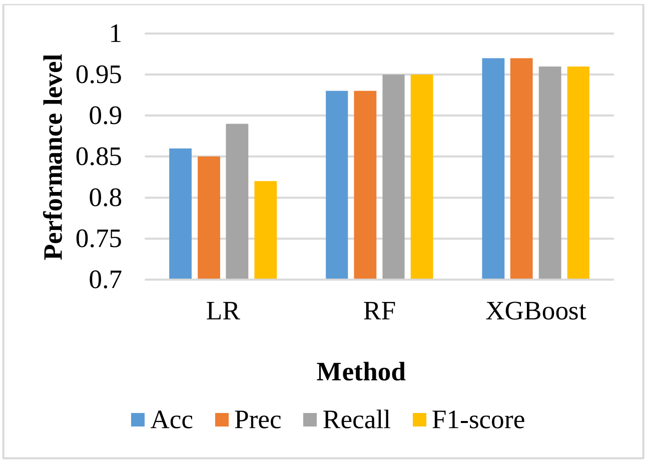

Figure 5. Performance comparison of Provision_PT_737E Security Camera device for Gafgyt attacks

Figure 6 comparison of the Provision_PT_737E Security Camera IoT device's performance evaluation findings may be understandable given that it is based on Miria botnet attacks. From the findings on the Miria botnet and its attacks using the three classifiers LR, RF, and XGBoost. Although LR has attained an accuracy value of 0.85, lower than our trained XGBoost model's 0.97, we assess three ML approaches in this analysis. These results also reveal a very high attack detection rate of over 0.96 for XGBoost as related to the 0.88 of LR, and 0.94 of RF models. Also, the proposed XGBoost model precision is 0.97 and has an F1-score of 0.96 in almost all Miria botnet attacks, which is very close to one.

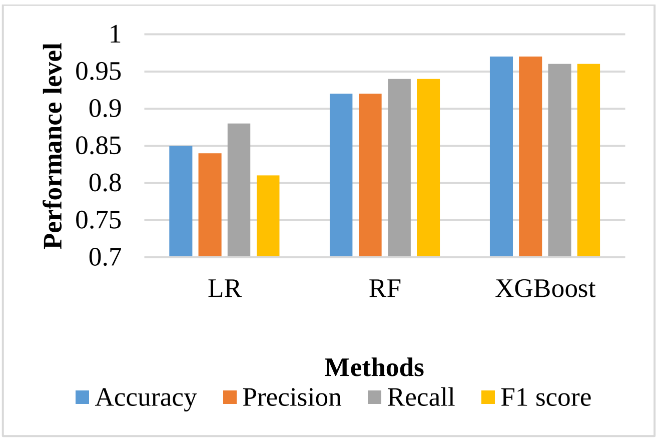

Figure 6. Performance comparison of Provision_PT_737E Security Camera device for Mirai attacks

Figure 7 shows the performance comparison of the Philips_B120N10 Baby Monitor device for Mirai attacks, in which the proposed XGBoost outperforms in all four cases of evaluation indices as related to the LR and RF models.

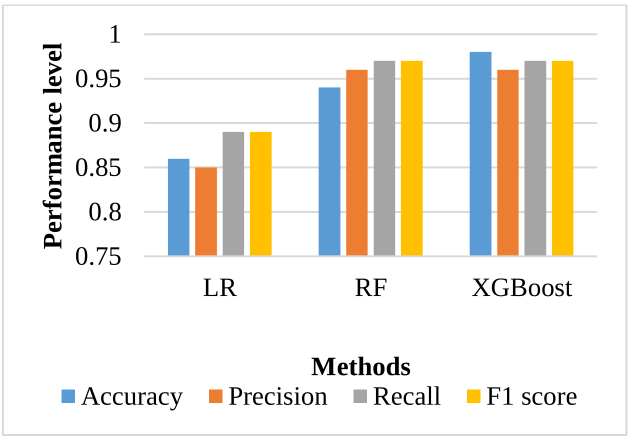

Figure 7. Performance comparison of Philips_B120N10 Baby Monitor device for Mirai attacks

Figure 8 shows the performance comparison of the SamsungSNH1011NWebcam device for Gafgyt attacks, in which the proposed XGBoost outperforms in all four cases of evaluation indices as related to the LR and RF models.

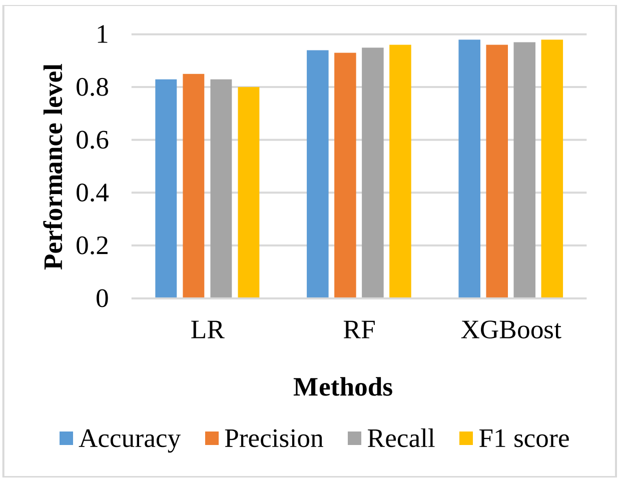

Figure 8. Performance comparison of SamsungSNH1011NWebcam device for Gafgyt attacks

In the testing phase, the proposed XGBoost model obtains high DR vs the epoch's times, as shown in Figure 9.

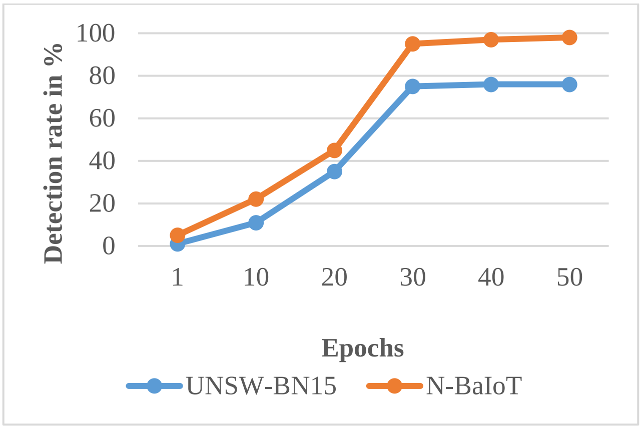

Figure 9. Attacks that were correctly identified (DR) were measured against both datasets epoch periods

In Figure 10, the detection ratio (DR) under multiclass classification is compared between the XGBoost model as an IDS and other well-known machine learning algorithms for classification, such as RF and LR. The findings show that the proposed IDS through XGBoost records lower false alarm rates for each and every attack than the RF, LR, and false alarm rates. This demonstrates that, when compared to other models, the model obtains the best DR [27, 28].

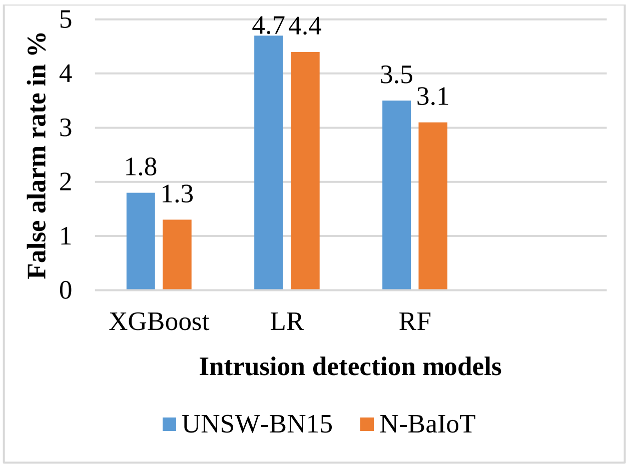

Figure 10. Contrasting the proposed IDS-based XGBoost model on both data sets with alternative machine learning methods

5. CONCLUSIONS

To identify cyberattacks, we developed an intrusion detection system based on MBF. It is intended to accomplish cloud environment privacy preservation as well. The XGBoost algorithm, which was tested on the UNSW-NB15 dataset and the N-BaIoT dataset for identifying different attack events that leverage cloud networks, is the foundation of the suggested approach to intrusion detection. The results are quite precise, and the error rate is very small. On average, the XGBoost algorithm's performance was best and the LR algorithm's performance was the worst of the three algorithms when detecting whether an IoT device was attacked by any specific botnet. The results by botnet type, for each attack on each device, for all three classifiers show very high ADRs and classification accuracy values (over 98%). In XGBoost approaches, which essentially provide a new type of classifier to the already existing trained ensemble, the re-training component of RF is omitted. Future research studies will examine the typical types of traffic patterns on various IoT devices to expand the anomaly-sub-capabilities engine's ability to efficiently identify unidentified attacks.

REFERENCES

[1] Sivaraman, V., Gharakheili, H.H., Fernandes, C., Clark, N., Karliychuk, T. (2018). Smart IoT devices in the home: Security and privacy implications. IEEE Technology and Society Magazine, 37(2): 71-79. https://doi.org/10.1109/MTS.2018.2826079
[2] Fotsing, P.T., Lim, S.Y., Musa, O., Almasri, A. (2020). Authchain blockchain-based authentication system. International Journal of Engineering Trends and Technology (IJETT), 70-74. https://doi.org/10.14445/22315381/CATI1P212
[3] Arnaldo, I., Cuesta-Infante, A., Arun, A., Lam, M., Bassias, C., Veeramachaneni, K. (2017). Learning representations for log data in cybersecurity. CSCML, 250-268. https://doi.org/10.1007/978-3-319-60080-2_19
[4] Stevanovic, M., Pedersen, J.M. (2016). Detecting bots using multi-level traffic analysis. International Journal on Cyber Situational Awareness (IJCSA), 1(1): 182-209. https://doi.org/10.22619/IJCSA.2016.100109
[5] Idriss, H.K. (2020). Mirai botnet in Lebanon. 2020 8th International Symposium on Digital Forensics and Security (ISDFS), 1-6. https://doi.org/10.1109/ISDFS49300.2020.9116456
[6] Zalte, S., Kamat, R.K., Ghorpade, V. (2020). Mitigation of DDoS attack in MANET. International Journal of Engineering and Advanced Technology, 9(6): 410-413. http://dx.doi.org/10.35940/ijeat.E95400.089620
[7] Abdulkarem, H.S., Alethawy, A.D. (2021). DDoS attack detection and mitigation at SDN environment. Journal of Information & Communications Technology, 4(1): 1-9. https://doi.org/10.31987/ijict.4.1.115
[8] Li, W., Meng, W., Kwok, L., IP, H. (2017). Enhancing collaborative intrusion detection networks against insider attacks using supervised intrusion sensitivity-based trust management model. Journal of Network and Computer Applications, 77: 135-145. https://doi.org/10.1016/j.jnca.2016.09.014
[9] Bernal Bernabe, J., Canovas, J.L., Hernández-Ramos, J.L., Torres Moreno, R., Skarmeta, A. (2019). Privacy-preserving solutions for blockchain: Review and challenges. IEEE Access, 7: 164908-164940. https://doi.org/10.1109/ACCESS.2019.2950872
[10] Liang, X., Shetty, S.S., Tosh, D.K., Kamhoua, C.A., Kwiat, K.A., Njilla, L.L. (2017). ProvChain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), pp. 468-477. https://doi.org/10.1109/CCGRID.2017.8
[11] Zhang, J., Pan, L., Han, Q., Chen, C., Wen, S., Xiang, Y. (2022). Deep learning based attack detection for cyber-physical system cybersecurity: A survey. IEEE/CAA

Journal of Automatica Sinica, 9(3): 377-391. https://doi.org/10.1109/jas.2021.1004261

[12] Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A. (2018). Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. ArXiv, abs/1802.09089. https://doi.org/10.14722/NDSS.2018.23204

[13] Dawit, N.A., Mathew, S.S., Hayawi, K. (2020). Suitability of blockchain for collaborative intrusion detection systems. 2020 12th Annual Undergraduate Research Conference on Applied Computing (URC), pp. 1-6. https://doi.org/10.1109/URC49805.2020.9099189

[14] Saldamli, G., Ramesh, P.H., Nair, K.M.S., Munegowda, R., Venkataramana, J., Tawalbeh, L. (2020). When Healthcare Services Meet Blockchain Technology. https://doi.org/10.1201/9780429324932-12

[15] Li, W., Tan, J.Q., Wang, Y. (2020). A Framework of Blockchain-Based Collaborative Intrusion Detection in Software Defined Networking. NSS, 261-276. https://doi.org/10.1007/978-3-030-65745-1_15

[16] Putra, G.D., Dedeoglu, V., Pathak, A., Kanhere S.S., Jurdak, R. (2021). Decentralised trustworthy collaborative intrusion detection system for IoT. 2021 IEEE International Conference on Blockchain (Blockchain), Melbourne, Australia, pp. 306-313. https://doi.org/10.1109/Blockchain53845.2021.00048

[17] Chen, L., Su, S. (2022). Optimization of the trust propagation on supply chain network based on blockchain plus. J. Intell. Manag. Decis., 1(1): 17-27. https://doi.org/10.56578/jimd010103

[18] Li, W., Wang, Y., Li, J., Au, M.H. (2019). Towards Blockchained Challenge-based Collaborative Intrusion Detection. Acns workshops, 122-139. https://doi.org/10.1007/978-3-030-29729-9_7

[19] Meng, W., Li, W., Zhou, J. (2021). Enhancing the security of blockchain-based software defined networking through trust-based traffic fusion and filtration. Information Fusion, 70: 60-71. https://doi.org/10.1016/j.inffus.2020.12.006

[20] Li, W., WANG, Y., Meng, W., LI, J., SU, C. (2022). BlockCSDN: Towards blockchain-based collaborative intrusion detection in software defined networking. IEICE Transactions on Information and Systems, E105.D(2): 272-279. https://doi.org/10.1587/transinf.2021BCP0013

[21] Hu, B., Zhou, C., Tian, Y., Qin, Y., Junping, X. (2019). A collaborative intrusion detection approach using blockchain for multimicrogrid systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 49(8): 1720-1730. https://doi.org/10.1109/TSMC.2019.2911548

[22] Kumar, R., Zhang, X., Khan, R., Ahad, I., Kumar, J. (2018). Malicious code detection based on image processing using deep learning. ICCAI 2018 Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, pp. 81-85. https://doi.org/10.1145/3194452.3194459

[23] Basysyar, F.M. Dwilestari, G. (2022). House price prediction using exploratory data analysis and machine learning with feature selection. Acadlore Trans. Mach. Learn., 1(1): 11-21. https://doi.org/10.56578/ataiml010103

[24] Zhang, F., Wu, D., Liu, P., Zhu, S. (2014). Program logic based software plagiarism detection. 2014 IEEE 25th International Symposium on Software Reliability Engineering, Naples, Italy, pp. 66-77. https://doi.org/10.1109/ISSRE.2014.18

[25] Singh, H., Bijalwan, A. (2017). Botnet detection using logistic regression technique. International Journal of Computer Science and Information Security, 15(7): 306-313.

[26] Moubayed, A., Injadat, M., Shami, A. (2020). Optimized random forest model for botnet detection based on DNS queries. 2020 32nd International Conference on Microelectronics (ICM), Aqaba, Jordan, pp. 1-4. https://doi.org/10.1109/ICM50269.2020.933181

[27] Yemelyanov, V., Nikonenko, U., Sytnyk, Y., Okhrimenko, I., Shulga, A. (2022). A model for countering the information and technical threats of intellectual capital management of innovation-oriented systems in the engineering sector. Ingénierie des Systèmes d'Information, 27(5): 799-806. https://doi.org/10.18280/isi.270513

[28] Saeed, S.H., Hadi, S.M., Hamad, A.H. (2022). Iraqi paradigm E-voting system based on hyperledger fabric blockchain platform. Ingénierie des Systèmes d'Information, 27(5): 737-745. https://doi.org/10.18280/isi.270506