ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

IoT Malware Detection and Mitigation in AMQP Simulated Environment

Ajeet Kumar Sharma[1] and Rakesh Kumar[2]

ABSTRACT

Internet of Things(IoT) devices are increasing rapidly and providing an infrastructure to connect and share information. Attackers are targeting the IoT network and sending malicious Traffic to IoT devices. Devices share large volumes of data, so malware is a security issue. IoT devices work in resource-constrained environments, and malware changes the traffic pattern. Due to that, it is a challenging situation to detect malware. An optimized ensemble learning-based approach is applied to detect the malware by analyzing the behavior. Training models employ the NF-BoT-IoT dataset to understand benign and attack traffic patterns. Multiple machine learning algorithms were evaluated on a given dataset, and observed that optimized ensemble learning performs best with an outstanding accuracy of 0.9915 and a precision of 0.9937. Another phase of the model mitigates malware by blocking attack IPs of Traffic flagged as malicious by the detection phase. The model's performance is evaluated in a simulated environment of advanced message queuing protocol(AMQP) using RabbitMQ broker. Future research directions will assist in further research work in enhancing the security of IoT infrastructure.

Article information:

Keywords: AMQP, Internet of Things, Malware Analysis, Machine Learning, Security

Article history:
Received: June 4, 2024
Revised: August 15, 2024
Accepted: September 6, 2024
Published: September 21, 2024 (Online)

DOI: 10.37936/ecti-cit.2024184.257016

1. INTRODUCTION

The growth of smart devices like computational intelligence and the IoT revolutionized various sectors like innovative automobiles, industries, and cities. On the other side, these devices face various security risks, i.e., malware attacks, ransomware, DDoS, device hijacking, and botnets. The massive development of IoT devices to complete the requirement may cause vulnerabilities, raising the risk of attacks. The insecurity of such devices leads to data breaches and threats to the entire network. The limitation of security design rules and the embedded nature of IoT devices allow cyber criminals to identify the vulnerabilities in the system and target attacks [1]. Maintaining IoT devices' security is challenging due to their diverse nature, as attackers might intercept numerous devices to exchange information and data. It is further required to maintain the confidentiality and integrity of sensitive data [2]. IoT is a collection of objects and services embedded with various components, such as sensors, communication channels, and protocols. Cyber attackers target smart homes, businesses, and healthcare by victimizing devices, networks, and communication channels. It is required to understand the variants of threats and analyze root causes to develop a framework that detects and prevents the attack [3]. The security of IoT devices is a serious issue due to diverse categories of attacks on IoT networks.

DoS attack occurs frequently, as devices are more prone to connectivity-related issues. Attackers target a device or network by sending malicious Traffic and preventing normal users from accessing the service.

A wormhole attack is an internal attack that violates the system policies related to access and disconnects communication between nodes.

A man-in-the-middle attack intercepts messages between the sender and receiver and changes the original message.

Sybil attacks are widespread in an IoT environment. The attacker pretends to be a normal node and is not easily identifiable due to integrating multiple devices in an IoT system. Buffer overflow attack causes insufficient space error, even if memory is

[1,2]The authors are with the Department of Computer Engineering & Applications, GLA University, Mathura, India, E-mail: ajeetsharma1989@gmail.com and rakesh.kumar@gla.ac.in
[1]Corresponding author: ajeetsharma1989@gmail.com

available [4,5,6]. Figure 1 shows the significant challenges to IoT systems, from data integrity, use of the common framework, and privacy issues to encryption handled carefully.

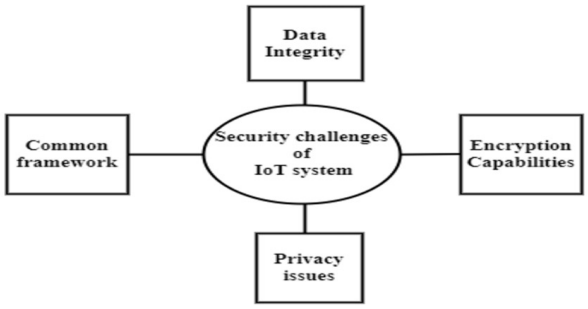


Fig.1: *Security Challenges to IoT System [1].*

Several IoT devices are connected and transmit the data and meet with the integrity of data, which is a significant challenge for security. Developing an integrated standard security framework in an IoT system puts an additional layer of protection [8]. Various platforms are used in IoT devices to share or exchange data; it is essential to ensure end-point security. Adequate encryption standards are vital to securing confidential data [9]. IoT networks are studied extensively to detect malware, but these studies face challenges regarding performance stability, false-positive rates, and high resource consumption. [10]. Ensemble learning will propose the IoT malware detection model, and the NF-BoT-IoT dataset, made up of eight basic net flows, will be used to perform training and testing. Norah et al. [11] developed a framework to detect malicious Traffic in Android-based systems by analyzing the behavior. [12] proposed a defense system against diverse IoT malware using deep learning and achieves an accuracy of 97.99% and an F1 score of 97%. RabbitMQ is a message broker used to implement AMQP and provides a reliable mode of communication. AMQP sets rules for the exchange of messages between applications. AMQP works on the application layer and establishes communication between the client and broker. In our study, we used RabbitMQ as a broker to ensure the exchange of messages from publisher to client and to simulate IoT devices for real-time model performance evaluation by passing the messages to the model. AMQP is reliable and secure and supports client-broker and client-server architecture. It stores and forwards data and ensures reliability during the network's troubleshooting and the message's delivery. The message is sent at most once, whether received or not, at least once and exactly once. Figure 2 shows the publish/subscribe model of AMQP; the broker works in two parts of exchange, and the queues hold the communication. Exchange is used to receive the messages of publishers, put them into the Queue, and forward them to subscribers. AMQP is based on TCP and relies on TLS/SASL to ensure security. It supports message brokers for messaging infrastructure. [13,14].

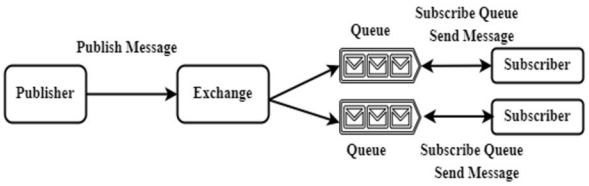


Fig.2: *Working of AMQP.*

The exchange plays a role in publishing/subscribing; the Queue is created automatically and bound to the exchange with the topic name, which is the same as the queue name. A message broker is a middleware for both sender and receiver to exchange the message. In the context of IoT, it consists of a central notification service with a fixed IP known to all devices. The server is responsible for receiving messages from publishers and distributing them to subscribers. In this scenario, the publisher is unaware of data processing, and the subscriber does not know the origin of the data [15].

AMQP protocol works on two cluster modes, normal and mirror, whereas the queue concept performs queue mirroring, as shown in Figure 3. Every mirror queue contains one master and one or more mirrors, with each broker consisting of the data in a queue. Every Queue holds a controller node, and an operation is applied to the controller node and then to mirrors. In case of failure, the subscriber is forwarded to another broker to digest data from the mirror. Exchanges assign messages to Queue on a binding rule and exchange basis. Memory and disk nodes can be in RabbitMQ, and configuration information can be on the disk node. RabbitMQ provides a web-based user interface to manage the Queue and supports several standard protocols, functionalities, and plugins [16,17].

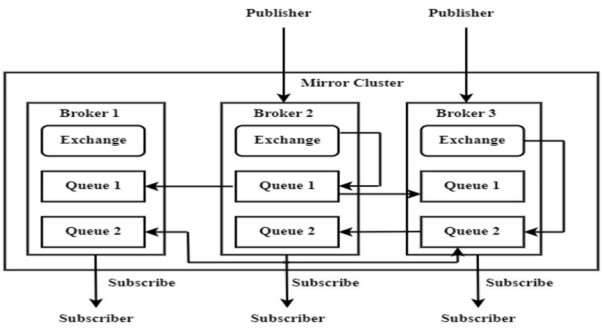


Fig.3: *RabbitMQ architecture [16].*

1.1 Contribution

In the presented work, an IoT malware detection and mitigation model detects IoT malware at an early

stage utilizing an optimized ensemble learning approach. Multiple algorithms are applied to evaluate the performance; the findings show that the proposed model outperforms all performance indicators. The proposed model analyzes the behavior of different malicious traffic categories, which enhances its capability to trace the pattern of unseen malware as well. The proposed firewall mitigates attack traffic by blocking suspicious IP addresses and allowing Traffic from benign IP addresses.

RabbitMQ broker simulates IoT devices to pass attacks and regular Traffic to evaluate the model's performance. The high throughput, less burden on the CPU, and shorter computation time show the model's efficiency in dealing with malicious Traffic in real-time. The Synthetic Minority Over-Sampling Technique(SMOTE) handles class imbalances and prevents biases.

1.2 Organization of Paper

This section gives the structure of the remaining part of the paper. Section 2 presents an overview of related works. Section 3 discusses the methodology used to develop the model; Section 4 details the result, comparison with another study, and experimental setup. Section 5 gives the overall summary of the paper and future research directions for researchers and security professionals.

2. RELATED WORK

Although several studies perform better in detecting IoT malware, The most recent studies present the most effective methods for detecting IoT malware. Nguyen, Huy-Trung, *et al.* [18] proposed a model that utilizes deep learning and machine learning-based techniques to detect IoT botnets. The model achieves an accuracy of 97% and an F-score of around 98%; compared with several machine learning algorithms, the presented approach is more efficient and robust. Karanja, Evanson Mwangi, *et al.* [19] presented a method to analyze and classify IoT malware using machine learning utilizing image texture features and applying multiple algorithms like k-nearest neighbors, naïve Bayes, and random forest. The binary file forms a grayscale image, and a gray-level co-occurrence distribution computes every mined image and uses features to detect malware. The model achieves an accuracy of 95% with random trees,89% with Naïve Bayes, and 80% with k-nearest neighbors.

Authors Jung, Woosub, et al. [20] proposed a convolutional neural network (CNN)-based deep learning model having 8-layer CNN. The model performs the task of categorizing data into four different classes, including botnet. The model achieves an accuracy of 96.5% and cross-tests around 90% approximately. Ma, Zhuo, *et al.* [21] proposed a model to detect Android malware utilizing a machine learning algorithm. A control flow graph shows information about API, and then Boolean frequency and time series data sets are constructed. Based on these datasets, three detection models for API calls, API frequency, and API sequence work. Finally, an ensemble model experiments of 10010 benign and 10683 malicious applications. The accuracy of 98.98% shows the efficiency and stability of the model in malware detection. Ren, Zhongru, *et al.* [22] presented a technique to detect malware by reducing time and effort using CNN and achieving an accuracy of 95.5%. R. M. Arif *et al.* [23] proposed an approach utilizing Generative Adversarial Network (GAN) and Deep Reinforcement Learning (DRL). The neural network of GAN takes unfiltered features of the dataset. It selects useful features using minimax objective functions used by the DRL agent—an increment of 35% in detection accuracy on retraining of machine learning models using adversarial samples. Ali, Mudasir, *et al.* [24] proposed a hybrid machine learning-based model to detect botnets in an IoT environment. UNSW-NB15 dataset contains nine different categories of attack, including benign, which is developing the model using artificial neural network(ANN), convolutional neural network(CNN), long short-term memory(LSTM), and recurrent neural network(RNN). The model achieves an accuracy of 96.98% in botnet detection and a cross-validation accuracy of 97.49%.

Authors T. G. Palla and S. Tayeb [25] proposed a model to detect Mirai malware in IoT nodes and use the n-BaIoT dataset with Mirai malware-infected features. A machine learning approach performs evaluation, and the accuracy of 92.8%, false negative rate of 0.3%, and F1 score of 0.99 proves it reliable and accurate. The study, El-Ghamry, Amir, et al. [26] performed a study to detect malicious network traffic. The optimized machine learning algorithm is applied and utilizes an ant colony optimizer(ACO) to select important features. The PSO algorithm tunes the support vector machine(SVM) classifier. The model produces an accuracy of 95.56%, recall of 96.43%, precision of 94.12%, and F1_score of 95.26%.

Bhayo, Jalal, *et al.* [27] performed a study to detect DDoS in software-defined IoT networks. They are using machine learning. The accuracy produced by naïve Bayes, support vector machine, and decision tree is 97.4%,96.1%, and 98.1%, respectively. In another study, Ravi, Vinayakumar, *et al.* [28] presented a multidimensional, deep learning-based model to detect and classify malware and CPU architecture classification in the Internet of Medical thing(IoMT) environment employing CNN and BLSTMT. The Proposed model uses an attention mechanism to target the most essential features to improve detection accuracy. Another author, Khan et al. [29], employed an ensemble learning-based approach utilizing an LSTM and decision tree classifier to form an intrusion detection model in an IoMT environment.

Authors Saravanan S and Balasubramanian UM [30] proposed a data pipeline using Apache Kafka and MongoDB to adapt the system while recognizing attacks in an IoT environment. On detection of the concept, the drift system restarts the classifier training along with the instances that cause drift. The model performs well on the IoT 23 dataset and achieves good accuracy. The training time of the random forest algorithm is in the Apache Spark environment.

In another study, Tabassum, Aliya, et al. [31] proposed an intrusion detection system to enhance training and detection efficiency. The incremental learning approach retrains the classifier on emerging features to detect new attack types. Chaganti et al. [32] designed an intrusion detection system using deep learning for the IoMT environment. They used Particle swarm optimization(PSO) to enhance the performance and detection of a combined dataset of network traffic and patient sensing feature selection.

3. METHODOLOGY

This section gives an overview of the methodology applied to develop and evaluate the model's performance and the Environment to simulate the behavior of IoT devices. The work aims to propose a methodology to classify benign and attack instances by analyzing traffic behavior using ensemble learning. The second phase of the model uses a script to block the IP addresses of Traffic flagged as malicious by the detection phase.

3.1 Dataset

The development of the model relies on the NF-BoT-IoT dataset[33,34] available at The University of Queensland web portal, which contains 600,100 instances of different malware and benign families, as shown in Figure 4.

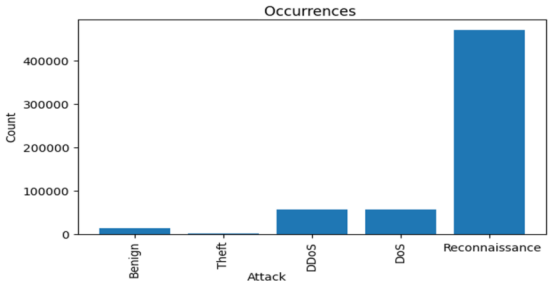


Fig.4: *Labels in Dataset.*

The dataset contains 586,241 attacks and 13,859 benign samples. Table 1, presented below, shows the distribution of flows.

Malware applications perform activities such as Reconnaissance to gather information about a network, DOS/DDoS using Mirai malware, and theft activities by applications such as Emotet and Stuxnet, which attack IoT devices. Rather than targeting individual malware, we analyze various malicious activities to understand the nature of different types of malware.

Table 1: *Distribution of Flows.*

Family	Count	Description
Benign	13859	Normal flow
Reconnaissance	470655	Used to insights the information about the host of the network.
DoS	56833	Denial of service prevents access to a service by sending multiple requests and overloading the system.
DDoS	56844	Distributed Denial of Service is similar to DoS but has multiple sources.
Theft	1909	Capture sensitive data or information such as theft or keylogging.

3.2 Model Architecture

Technology innovations are taking the world forward to an intelligent and embedded environment. IoT-empowered systems like automated smart homes, smart cities, and advanced health care systems are at risk of cyber threats. As shown in Figure 5, the proposed model effectively identifies the malware [35]. The preprocessing phase is vital in constructing datasets compatible with machine learning classifiers. It includes removing less or no contributing features, handling missing or uncaptured values, and converting categorical values into numerical ones. In this work, a Label encoder converts categorical values into numerical values, and IP addresses are converted into integers to make operations more convenient. The network traffic contains benign and malicious (reconnaissance, DoS, DDoS, and theft) traffic. The dataset is highly imbalanced at this label as a significant difference exists between benign (13,859) and attack instances (586,241). The Synthetic Minority Over-Sampling Technique (SMOTE) is applied to generate samples for the minority class, and benign samples are increased to 586,241 to overcome data imbalance and prevent the model from biased results; this balanced dataset will not only produce unbiased results but also reduce the false positives.

The random forest classifier computes and extracts the importance of features. It performs feature analysis and arranges them in decreasing order of importance, as in Figure 6. This analysis supports the model in gaining insights into the most contributing features to the result. As in Eq. (1), the Gini index identifies the best features to split data at every node of the tree, p_i, which is the probability of a thing with a specific class.

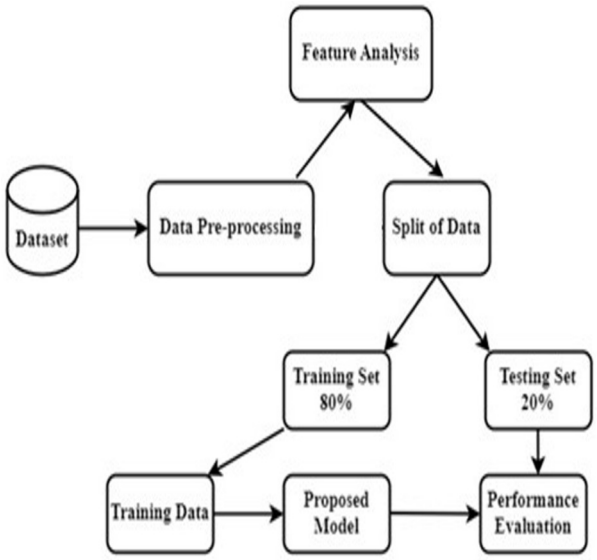


Fig.5: *Detection Phase of the model.*

$$Gini = 1 - \sum_{i=1}^{n} (p_i)^2 \quad (1)$$

In our dataset, two classes, benign and attacks, are taken. If the chance of probability for a class is p, then for others, it is $1-p$. In binary classification, the Gini index goes from 0 to 0.5. 0 shows a pure node, and 0.5 shows an impure node. Feature importance is the sum of decreasing impurity in all tree nodes.

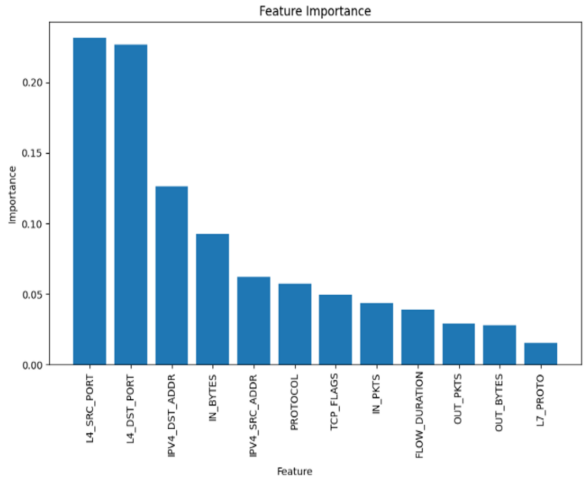


Fig.6: *Feature Analysis.*

Table 2 shows the datasets containing twelve different features to train and develop the model.

3.3 Ensemble Learning

Ensemble learning combines multiple machine learning models to enhance the model's predictive performance. The main objective is to develop a robust and more efficient model.

1) Bagging

Table 2: *Features of dataset.*

Features	Description
L4_SRC_PORT	Port number of source
L4_DST_PORT	Port number of destination
IPV4_DST_ADDR	Address of destination
IN_BYTES	Incoming bytes
IPV4_SRC_ADDR	Address of source
PROTOCOL	Identifier byte for IP
TCP_FLAGS	TCP flags
IN_PKTS	Incoming packets
FLOW_DURATION	Duration of flow
OUT_PKTS	Outgoing packets
OUT_BYTES	Outgoing bytes
L7_PROTO	Protocol at layer 7

The bootstrap aggregation technique uses multiple dataset samples and trains the base model on each sample. Then, the base models vote for the final prediction.

2) Boosting

Base model training sequentially corrects previous errors; the final prediction is an average of predictions made by all base models.

3) Stacking

Predictions made by base learners are stacked together and used as input to train meta-learners for more robust predictions. The meta-learner makes the final prediction.

4) Blending

The data's hold-out structure from base models, and then a meta-learner is trained based on these predictions.

5) Voting

Multiple trained models make a final combined prediction using various voting approaches such as Soft, Hard, or Weighted.

3.3.1 Hyperparameter

The selection of hyperparameters contributes a lot to optimizing the classifier to enhance the model's performance.

1) Estimator

The estimator defines the number of trees the algorithm builds, each correcting the previous one's errors. Developing a balanced model depends on the correct selection of estimators because many estimators may enhance the model's performance but increase the risk of overfitting. In our study, cross-validation finds a balance estimator.

2) Learning Rate

In the final stage of the model, the learning rate monitors and controls the contribution paid by the individual tree. A lower learning rate supports the learning process more gradually but can increase the number of estimators. The proposed study uses a grid search to select the optimal eta.

3) Max Depth

This parameter controls the maximum depth of the tree. A smaller depth prevents overfitting, while deeper trees can lead to overfitting. In the proposed model, cross-validation selects the optimal depth.

4) Gamma

The minimum loss reduction partitions the tree node, and a suitable value must be selected to avoid overfitting and provide the most accurate result. Cross-validation is applied to select a perfect gamma value.

3.4 Malware Detection

The model's training on various malicious Traffic enables it to detect the malware that changes the behavior. Data division is 80% for training and 20% for testing. Grid search, along with cross-validation, uses distinct combinations of hyperparameters. Grid search is a process that looks for a specified subset of hyperparameter space and evaluates the cost function on the generated Hyperparameter. Cross-validation combines the average fitness measures in a prediction to predict more accurate performance. Consider the k trees, then the model(M), a collection of decision trees, can be defined as in Eq (2).

$$M = \sum_{k=1}^{k} f_k \tag{2}$$

Where f_k is the prediction from the decision tree after utilizing all the prediction trees, computed using Eq (3)

$$\hat{y}_i = \sum_{k=1}^{k} f_k(x_i) \tag{3}$$

Here, x_i is the feature vector for the ith data point. As in Eq (4), the optimized log loss function trains the model and log loss for binary classification.

$$L = -\frac{1}{N}\sum_{i}^{N} (y_i \log(p_i)) + (1 - y_i)\log(1 - p_i) \tag{4}$$

Regularization prevents overfitting and complexity control.

$$\Omega = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{5}$$

T is the number of leaves, and w_j^2 is the score of the jth leaf. To get the model's objective, loss function and regularization are combined. Less function controls prediction power, and gradient descent optimizes the objective. The objective function $Obj = L + \Omega$ is an iterative procedure objective function, as in Eq(6).

$$\begin{aligned} obj^{(t)} &= \sum_{t=1}^{N} L(y_i, \hat{y}_t^{(t)}) + \sum_{t=1}^{t} \Omega(f_t) = \\ &\sum_{t=1}^{N} L(y_i, \hat{y}_t^{(t-1)} + f_t(x_t)) + \sum_{t=1}^{t} \Omega(f_t) \end{aligned} \tag{6}$$

First-order and second-order gradients optimize and improve performance $\partial \hat{y}_t^{(t)}\ (obj)^{(t)} \partial^2 \hat{y}_t^{(t)}\ (obj)^{(t)}$, and Eq(7) calculates the second-order Taylor approximation.

$$\text{obj}^t = \sum_{i=1}^{N} \left[L(y_i, \hat{y}^{t-1}) + g_t f_t(x_t) + \frac{1}{2} h_t f_t^2(x_t) \right] + \sum_{i=1}^{t} \Omega(f_t) \tag{7}$$

In the testing phase, the trained model is used to classify benign and attack Traffic. In the dataset, benign Traffic is labeled 0, and all four types of malware are labeled. The proposed algorithm1 performs binary classification and achieves a high accuracy of 99.15% and a precision of 99.37%.

Algorithm:1 Detection

1: Initialization

$D = \{[x_t y_t]\}_{i=1}^{n} \rightarrow$ Dataset, where x_i is a set of features and y_i is the target label

Split dataset into training and testing

2: Grid Search

Split training data into a set of training and validation

Training of model with set hyperparameters and compute validation loss

Average validation loss = $\frac{1}{k}\sum_{k=1}^{k} l_k(\theta)$

Optimize Hyperparameter to reduce validation loss

3: Training of model

Model training using optimized Hyperparameter

4: Testing

Evaluation of trained model on test dataset

Compute loss and performance metrics

3.5 Malware Mitigation

Network traffic is passed to each simulated device (Thermostat et al.), as shown in Figure 7. Attack traffic IP addresses are blocked and allow only regular Traffic, as discussed in algorithm 2. The Python script is designed to simulate network traffic and includes various input parameters, such as device name, traffic data, dictionary for traffic statistics, and traffic generation rate. A JSON dictionary extracts the traffic information and device name and then publishes it to a message queue. Further, it enlightens the data

sent from a device, and there is a delay between 1 to 5 seconds for intervals between transmissions. A Function is defined to consume the message after that inner function performs the processing of each message received. Message dictionary checks the label of Traffic as benign (label 0) or malicious (label 1). If the message is benign, then an acknowledgment is sent back to the Queue about the successful processing of the message and further removal from the Queue. In case of an attack, traffic acknowledgment disappears, and the message is not removed from the Queue to perform further analysis and mitigation.

Algorithm:2 Mitigation

1: Initialization
Statistics for benign, S_B and attack, S_A traffic
Set of benign(Allowed_IPs) and attack IP(Attack_IPs) addresses
2: Simulation of Traffic for each IoT device
For each benign instance
Update $S_B = S_B + [f_i$[In bytes], f_i[Out bytes], f_i[In packets], f_i[Out packets]
Publish f_i data to RabbitMQ
For each attack instances
Update $S_A = S_A + [fj$[In bytes], f_j[Out bytes], f_j[In packets], f_j[Out packets]
Publish f_j data to RabbitMQ
Add f_j to set of Attack_IPs
3: Mitigation of attack IP.
For each IP ∈ Attack_IPs
Block unique attack IP addresses
For each IP ∈ Allowed_IPs
Add f_i[source address]to Allowed_IPs
4: Consumption of message
If label = 0(benign)
Acknowledge the message
else
Do not acknowledge the message

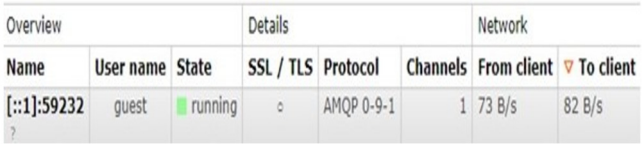
3.6 Simulation

RabbitMQ, a message broker, is configured with exchange to route the messages and Queue to hold incoming messages. RabbitMQ implements AMQP, which provides a framework for communicating applications. JSON for encoding and decoding data and Pika library to communicate with RabbittMQ are used to connect simulated devices to the ML model. The dataset's traffic instances (benign and attack) pass to simulated IoT devices. All the messages received from RabbitMQ are consumed and processed by the model. If the message is suspicious, it is blocked immediately, as shown in Figure 7.

As shown in Figure 8, sending and receiving the message is executed with RabbitMQ using the host, port, username, and password.

The queue parameter ensures the message's existence before consumption and publishing. Figure 9

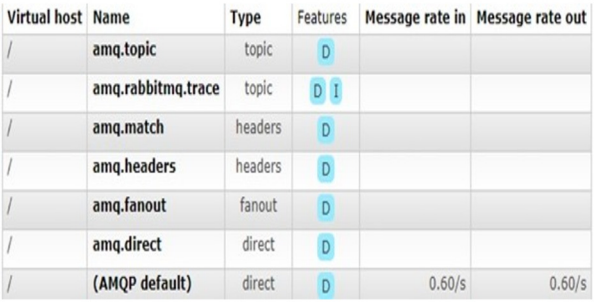
Fig. 7: *Simulation of Traffic.*



Overview			Details			Network	
Name	User name	State	SSL / TLS	Protocol	Channels	From client	To client
[::1]:59232	guest	running	○	AMQP 0-9-1	1	73 B/s	82 B/s

Fig. 8: *Connection Establishment.*

shows the exchanges where the virtual host provides a route for messages to Queue based on the routing key and a route based on header attributes, default route messages through a key released by Queue. In feature space, 'D' indicates that the exchange will be durable, and 'I' shows the internal, which means it is not to be published directly to the client.



Virtual host	Name	Type	Features	Message rate in	Message rate out
/	amq.topic	topic	D		
/	amq.rabbitmq.trace	topic	D I		
/	amq.match	headers	D		
/	amq.headers	headers	D		
/	amq.fanout	fanout	D		
/	amq.direct	direct	D		
/	(AMQP default)	direct	D	0.60/s	0.60/s

Fig. 9: *Message Exchanges.*

Figure 10 below shows the listening port used to monitor the connections (incoming) and port number to each port. These ports play a vital role in establishing connections between brokers and clients. Proper port configuration is required to keep communication secure and efficient; otherwise, it may lead to security and vulnerability flaws.

In the simulated Environment, the data is passed and consumed by the model and flagged as either standard or attack Traffic based on the behavior of instances. If the Traffic is benign (label 0), the model acknowledges or blocks it. This simulation presents the model's effectiveness in real scenarios and the areas of improvement. It contributes to adapting best

Protocol	Bound to	Port
amqp	0.0.0.0	5672
amqp	::	5672
clustering	::	25672
http	::	15672

Fig.10: *Listening Ports.*

security practices to handle malicious actors.

3.7 Tools Used

An ideal selection of tools is required to develop a sophisticated model. In the proposed main tools, we utilized Google Colab for Python script, RabbitMQ broker to implement AMQP, and a system running on a Windows-based operating system with 12 GB RAM and 1 TB Hard disk.

4. RESULT ANALYSIS

This section summarizes the outcomes of various algorithms and performance comparisons with existing studies.

4.1 Performance Matric

The model is evaluated utilizing the NF-BoT-IoT dataset on four different categories of malicious Traffic and achieves an accuracy of 99.15%. The model's efficiency is measured using various performance metrics to show the correct predictions. The confusion matrix in Figure 11 shows that 1127 correctly predicted Negative classes(TN),893 positive classes were mispredicted (FP),297 Negative classes were predictive as positive, and 117,703 classes were positive.

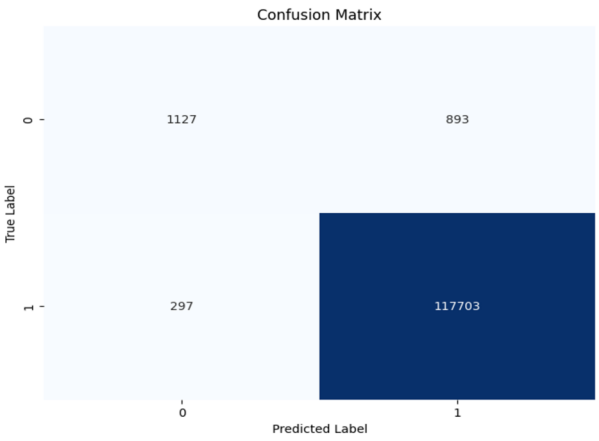


Fig.11: *Confusion Matrix.*

Table 3 shows the performance of the optimized XG Boost model used to perform binary classification.

Accuracy measures correct classifications, and the model classifies the instances with 99.15% accuracy computed using Eq (8).

Table 3: *Performance Metrics.*

Accuracy	Precision	Recall	F1 Score
99.15	99.37	99.75	99.56

$$\text{Accuracy} = \frac{\text{True Positive+Ttue Negative}}{\text{Total Instances}} \quad (8)$$

Precision indicates the Positive predictions out of the total made by the model. A precision of 99.38%, calculated using Eq (9), indicates that the model's prediction about Positive instances is correct up to 99.38%; a high value shows a low false positive rate.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives+False Positive}} \quad (9)$$

Recall is a ratio of true positives and total Positive classes. A value of 99.76%, calculated with Eq (10), shows that the correctly identified instances indicate a low rate of False Negatives.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives+False Negative}} \quad (10)$$

The F1 score combines precision and recall. The F1 score of 0.9957, calculated using Eq (11), shows the perfect balance of precision and recall.

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (11)$$

The training of five machine learning models and comparing their results show our model's outstanding performance. The comparisons establish that the proposed algorithm's performance is superior across all the metrics. Table 4 shows a summary of performance obtained through various classifiers.

Table 4: *Performance of different classifiers.*

Classifier	Accuracy	Precision	Recall	F1 Score
SVM	0.9800	0.9800	1.0	0.9899
Logistic Regression	0.9815	0.9818	0.9947	0.9908
Naive Bayes	0.9814	0.9814	0.9815	0.9906
MLP Neural Network	0.9800	0.9799	1.0	0.9898
LDA	0.9858	0.9863	0.9923	0.9927
Proposed Algorithm	0.9915	0.9937	0.9975	0.9956

Support Vector Machine (SVM), a supervised machine learning algorithm, performs the task of classification and regression. The main motive is to find a hyperplane for the separation of classes. This algorithm is not very suitable for large datasets. Logistic regression is the probability of success and failure of

events suitable for binary classification. It makes the assumption of linearity between dependent and independent variables a significant limitation. Naive Bayes is a simple and easy-to-use algorithm based on the Bayes theorem to classify the classes. One of the significant limitations is that it assumes all the features are independent. Multilayer Perceptron (LP) is one of the types of neural networks characterized by multiple layers (input layer, hidden layer, output layer) of interconnected nodes. It solves complex patterns and overcomes the limitations of a single layer. It consumes significant resources in training, and performance is highly dependent on parameter tuning. Linear Discriminant Analysis (LDA) is a supervised machine learning algorithm for classification. It finds directions in feature space used to separate various classes. Generally, LDA assumes the data is linearly separable and has a Gaussian distribution form, but that is not the case in every case. The proposed algorithm enhances the model's performance and iteratively minimizes errors, and the regularization feature prevents the issue of overfitting. The facility to use atomized loss functions improves the algorithm's applicability while performing classification. Cross-validation and parameter tuning ensure the model can classify known and known data.

Numerous studies perform malware detection and prevention, and results obtained from those studies have been compared with the proposed model, as shown in Table 5. The comparison demonstrates our model's outstanding performance, making it unique.

Table 5: *Comparison with existing studies.*

Research Work	Methodology	Accuracy	F1-Score
Huy-Trung, *et al.* [18]	DL & ML	97%	98%
Woosub, *et al.* [20]	CNN	96.5%	85%
Ma, Zhuo, *et al.* [21]	DL	98.98%	95.22%
Ali, Mudasir, *et al.* [24]	Hybrid ML	94.40%	94.39%
T. G. Palla, *et al.* [25]	ML	92.8%	99%
Amir, *et al.* [26]	ML	95.56%	95.6%
Chaganti *et al.* [32]	PSO-DNN	96%	95%
Proposed Work	Optimized ML	0.9915	0.9956

The combined use of Deep Learning (DL) and Machine Learning (ML) approaches have achieved an accuracy of 97% and F1 value of 98%, which show consistency in performance. The Convolutional Neural Network (CNN) model presents an accuracy of 96.5% and an F1 score of 85%. Performance shows the variability in evaluation. DL models achieve a high classification accuracy of 98.98% and an F1-value of 95.22%, indicating the model's performance consistency. The hybrid ML-based app approach achieves an ideal output of accuracy and F1 values of 94.40% and 94.39%, respectively. Another ML-based model achieves an accuracy of 92.8% and measures the F1 at 99%. Another ML model shows consistency in results with an accuracy of 95.56% and an F1-score of 95.26%, showing the model's reliability. Particle Swarm Optimiza ion with Deep Neural Network (PSO-DNN) achieves an accuracy of 96% and an F1-score of 95%, indicating the robustness of the model. Although the performance of existing models is good, our model performance is excellent, with an accuracy of 99.15% and an F1 value of 99.56%.

4.2 Statistics Evaluation

The model takes around 2.38 seconds to complete and handles approximately 50,518 samples per second. This high throughput indicates the algorithm's efficiency in processing large amounts of data. The CPU utilization is an indicator of the capacity consumption during the execution, and it consumes 4.5% of the CPU's capability. Simulated devices publish data in the RabbitMQ queue, process the traffic message, and perform blocking(attack) and allowing(benign) Traffic, as shown in Figure 12. A function distinguishes between attack and benign Traffic from randomly selected datasets. Specific instances of regular and attack Traffic pass through each device for valuable insight into behavior. A list of attack IPs is maintained to block these IPs further to mitigate attacks and prevention from future attacks. Messages from benign Traffic are only acknowledged to maintain the system's integrity and mitigate the impact of malware; malicious traffic messages are not acknowledged and prevented from being processed in the system.

```
Malicious traffic received. Not acknowledging message.
Blocking attack traffic for Thermostat...
Blocking traffic from attacker IP: 192.168.100.148
Blocking traffic from attacker IP: 192.168.100.147
Blocking traffic from attacker IP: 192.168.100.149
Blocking traffic from attacker IP: 192.168.100.150
Allowing benign traffic for Thermostat...
Allowing traffic from benign IP: 192.168.100.150
```

Fig.12: *Blocking of Attack Traffic.*

Once a traffic event simulation starts, the statistics are updated, and data is published; blocking and no acknowledgment of attack traffic show significant contributions and provide valuable insight into forthcoming security strategies. Statistics, as in Figur 13, show the behavior of various traffic patterns. High incoming and outgoing benign Traffic indicates the utilization of the network by legitimate services. High packets of malicious Traffic indicate the attempt of malicious entities to connect with a system for cyber-attack. The traffic differences between both labels (benign and attack) will help create malware profiles.

Analysis of packets can help security professionals to understand the risk and further advance mitigation strategies.

```
Benign Traffic Statistics:
{'in_bytes': 5717, 'out_bytes': 6164, 'in_pkts': 54, 'out_pkts': 39}

Malicious Traffic Statistics:
{'in_bytes': 7960, 'out_bytes': 8999, 'in_pkts': 83, 'out_pkts': 63}
```

Fig.13: *Traffic Statistics.*

5. CONCLUSION AND FUTURE WORK

This study proposed a model to detect and mitigate malware that generates malicious Traffic in an IoT environment. The model produces high throughput, less turnaround time, and low CPU utilization, making it a perfect fit for IoT devices.NF-BoT-IoT dataset, having instances of regular and attack Traffic, is used to analyze the behavior and understand the pattern of malware generating the Traffic. Another phase of the model is mitigating attack traffic by blocking unique attack IPs to prevent future attacks. The model is reliable and more realistic. To make the model more reliable and realistic, RabbitMQ, a message broker, simulates the network traffic based on the dataset. This simulation helps analyze the model's performance and gives insights into optimization and deployment in IoT infrastructure. Employing ensemble learning is a wise decision for environments where the network's security is a concern. The proposed model performs well and provides an additional layer of security to the IoT environment with a high accuracy of 0.9799 and a precision of 0.9937. Future work could include using diverse datasets to categorize and detect malware. Further research will be working on developing a model for some specific categories of IoT devices by understanding the severity of the attack.

ACKNOWLEDGEMENT

The authors have no conflicts of interest to declare relevant to this article's content. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial or non-financial interest in the subject matter or materials discussed in this manuscript.

AUTHOR CONTRIBUTIONS

Conceptualization, Ajeet Kumar Sharma; Methodology, Ajeet Kumar Sharma and Rakesh Kumar; Formal analysis, Ajeet Kumar Sharma and Rakesh Kumar; Writing—original draft preparation, Ajeet Kumar Sharma, and Rakesh Kumar; Writing— Ajeet Kumar Sharma, and Rakesh Kumar; Supervision, Rakesh Kumar. All authors have read and agreed to the published version of the manuscript.

References

[1] M. Asam *et al.*, "IoT malware detection architecture using a novel channel boosted and squeezed CNN," *Scientific Reports*, vol. 12, no. 15498, 2022.

[2] S. Sasikala and S. Janakiraman, "A Review on Machine Learning-based Malware Detection Techniques for Internet of Things (IoT) Environments," *Wireless Personal Communications*, vol. 132, pp. 1961-1974, Aug. 2023.

[3] D. Rani, N. S. Gill and P. Gulia, "Classification of Security Issues and Cyber Attacks in Layered Internet of Things," *Journal of Theoretical and Applied Information Technology*, vol. 100, no. 13, pp. 4895–4913, 2022.

[4] R. Praveen and P. Pabitha, "ASK-RAM-IMOT: Autonomous Shared Keys based Remote Authentication Method for Internet of Medical Things Applications," *Wireless Personal Communications*, vol. 131, pp. 273-293, 2023.

[5] R. Praveen and P. Pabitha, "Improved Gentry–Halevi's fully homomorphic encryption-based lightweight privacy-preserving scheme for securing medical Internet of Things," *Transactions on Emerging TelecommunicationsTechnologies*, vol. 34, no. 4:e4732, 2023.

[6] R. Praveen and P. Pabitha, "A secure, lightweight, uzzy Embedder-based user authentication scheme for Internet of Medical Things applications," *Journal of Intelligent & Fuzzy Systems*, vol. 44, no. 5, pp. 7523–7542, 2023.

[7] U. Khadam, M. M. Iqbal, M. Alruily, M. A. A. Ghamdi, M. Ramzan and S. H. Almotiri, "Text Data Security and Privacy in the Internet of Things: Threats, Challenges, and Future Directions," *Wireless Communications and Mobile Computing*, vol. 2020, no. 7105625, 2020.

[8] M. Nobakht, R. Javidan and A. Pourebrahimi, "DEMD-IoT: a deep ensemble model for IoT malware detection using CNNs and network traffic," *Evolving Systems*, vol. 14, pp. 46 –477, 2023.

[9] C. Haar and E. Buchmann, "FANE: A Firewall Appliance for the Smart Home," *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Leipzig, Germany, pp. 449-458, 2019.

[10] N. Dandotiya, P. Khatri, M. Kumar and S. K. Kachhap, "Sensor-Based Secure Framework for IoT-Based Smart Homes," in *Proceedings of International Conference on Computational Intelligence. Algorithms for Intelligent Systems,* Springer, Singapore, pp. 283-290, 2022.

[11] N. Abanmi, H. Kurdi and M. Alzamel, "Dynamic IoT malware detection in Android systems using profile hidden Markov models," *Applied Sciences*, vol. 13, no. 1:557, 2022.

[12] A. A. Almazroi and N. Ayub, "Deep learning hybridization for improved malware detection in

smart Internet of Things," *Scientific reports*, vol. 14, no. 7838, 2024.

[13] A. A. O. Bahashwan and S. Manickam, "A Brief Review of Messaging Protocol Standards for Internet of Things (IoT), vol. 8, no. 1, pp. 1-14, 2018.

[14] F. Iqbal, M. Gohar, H. Alquhayz, S.-J. Koh and J.-G. Choi, "Performance evaluation of AMQP over QUIC in the internet-of-thing networks," *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 4, 1-9, 2023.

[15] R. López-Gómez, L. Panizo and M.-D.-M. Gallardo, "Flextory: Flexible Sof ware Factory of IoT Data Consumers," *Sensors*, vol. 24, no. 8:2550, 2024.

[16] G Fu, Y. Zhang and G. Yu, "A Fair Comparison of Message Queuing Systems," *IEEE Access.*, vol. 9, pp. 421-432, 2021.

[17] S. Dixit and M. Madhu, "Distributing messages using Rabbitmq with advanced message exchanges," *International Journal of Research Studies in Computer Science and Engineering (IJRSCSE)*, vol. 6, no. 2, pp. 24–28, 2019.

[18] H.-T. Nguyen, D.-H. Nguyen, Q.-D. Ngo, V.-H. Tran and V.-H. Le, "Towards a rooted subgraph classifier for IoT botnet detection," in *Proceedings of the 7th International Conference on Computer and Communications Management*, pp. 247-251, 2019.

[19] E. M. Karanja, , S. Masupe and M. G. Jeffrey, "Analysis of Internet of Things malware using image texture features and machine learning techniques," *Internet of Things*, vol. 9, p.100153, 2020.

[20] W. Jung, H. Zhao, M. Sun and G. Zhou, "IoT botnet detection via power consumption modeling," *Smart Health*, vol. 15, p. 100103, 2020.

[21] Z. Ma, H. Ge, Y. Liu, M. Zhao and J. Ma, "A Combination Method for Android Malware Detection Based on Control Flow Graphs and Machine Learning Algorithms," in *IEEE Access*, vol. 7, pp. 21235-21245, 2019.

[22] Z. Ren, H. Wu, Qian Ning, I. Hussain and B. Chen, "End-to-end malware detection for android IoT devices using deep learning," *Ad Hoc Networks*, vol. 101, p. 102098, 2020.

[23] R. M. Arif *et al.*, "A Deep Reinforcement Learning Framework to Evade Black-Box Machine Learning Based IoT Malware Detectors Using GAN-Generated Influential Features," in *IEEE Access*, vol. 11, pp. 133717-133729, 2023.

[24] M. Ali, M. Shahroz, M. F. Mushtaq, S. Alfarhood, M. Safran and I. Ashraf, "Hybrid Machine Learning Model for Efficient Botnet Attack Detection in IoT Environment," in *IEEE Access*, vol. 12, pp. 40682-40699, 2024.

[25] T. G. Palla and S. Tayeb, "Intelligent Mirai Malware Detection for IoT Nodes," *Electronics*, vol. 10, no. 11:1241, May 2021.

[26] A. El-Ghamry, T. Gaber, K. K. Mohammed and A. E. Hassanien, "Optimized and Efficient Image-Based IoT Malware Detection Method," *Electronics*. vol. 12, no. 3:708, Jan. 2023.

[27] J. Bhayo, S. A. Shah, S. Hameed, A. Ahmed, J. Nasir and D. Draheim, "Towards a machine learning-based framework for DDOS attack detection in software-defined IoT (SD-IoT) networks," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106432, Aug. 2023.

[28] V. Ravi, T. D. Pham and M. Alazab, "Attention-Based Multidimensional Deep Learning Approach for Cross-Architecture IoMT Malware Detection and Classification in Healthcare Cyber-Physical Systems," in *IEEE Transactions on Computational Social Systems*, vol. 10, no. 4, pp. 1597-1606, Aug. 2023.

[29] F. Khan, M. A. Jan, R. Alturki, M. D. Alshehri, S. T. Shah and A. u. Rehman, "A Secure Ensemble Learning-Based Fog-Cloud Approach for Cyberattack Detection in IoMT," in *IEEE Transactions on Industrial Informatics*, vol. 19, no. 10, pp. 10125-10132, Oct. 2023

[30] S. Saravanan and U. M. Balasubramanian, "An Adaptive Scalable Data Pipeline for Multiclass Attack Classification in Large-Scale IoT Networks," in *Big Data Mining and Analytics*, vol. 7, no. 2, pp. 500–511, Jun. 2024.

[31] A. Tabassum, A. Erbad, A. Mohamed and M. Guizani, "Privacy-Preserving Distributed IDS Using Incremental Learning for IoT Health Systems," in *IEEE Access*, vol. 9, pp. 14271-14283, 2021.

[32] R. Chaganti, A. Mourade, V. Ravi, N. Vemprala, A. Dua and B. Bhushan, "A particle swarm optimization and deep learning approach for intrusion detection system in Internet of Medical Things," *Sustainability*, vol.14, no.19, p.12828, Oct. 2022.

[33] M. Sarhan, S. Layeghy and M. Portmann, "Towards a Standard Feature Set for Network Intrusion Detection System Datasets," *Mobile Networks and Applications*, vol. 27, no. 1, pp.357-370, 2022.

[34] M. Sarhan, S. Layeghy and M. Portmann, "Evaluating Standard Feature Sets Towards Increased Generalisability and Explainability of ML-Based Network Intrusion Detection," *Big Data Research*, vol. 30, p.100359, 2022.

[35] K. A.P. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz and V. H. C. de Albuquerque, "Internet of Things: A survey on machine learning-based intrusion detection approaches," *Computer Networks*, vol. 151, pp. 147–157, 2019.

Ajeet Kumar Sharma is pursuing Ph.d in computer science from GLA University, Mathura, India. He has more than 08 years of experience in academia, and his areas of interest are IoT, Cyber Security, Digital Forensics, and Machine learning.

Rakesh Kumar is working as an Associate Professor at the Institute of Engineering & Technology, GLA University, Mathura, India. He has 15+ years of Teaching/Research experience in the field of Computer Science, and his areas of interest are Wireless Sensor Networks, Cyber Security, and Machine Learning.