International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 06 | June - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Smart-shield: Dual Phase ML Defense for IOT Botnet Threats

Abhinav Murkute, Tejas Fulzele, Yash Lase, Kundan Kumar, Prof. C.A.Deshpande

Department of Information Technology, Sinhgad College of Engineering, Pune, Maharashtra, India

I.ABSTRACT

Botnet attacks are one of the most prevalent and harmful threats in the Internet of Things (IoT) ecosystem. These attacks typically begin with a reconnaissance or scanning phase and eventually lead to large-scale Distributed Denial of Service (DDoS) assaults. Current detection methods often focus on identifying botnet activity only after IoT devices have already been compromised and are actively participating in DDoS attacks. Additionally, many of these techniques are restricted to specific datasets, which limits their effectiveness across different environments due to variations in attack behaviors. To overcome these limitations, this study introduces a diverse and comprehensive dataset that includes 33 types of scanning attacks and 60 types of DDoS attacks. The dataset is constructed using samples from three widely recognized public datasets, enhancing the model's ability to generalize across various attack patterns. The paper presents a novel Dual Phase Machine Learning Defense strategy for countering IoT botnet threats. In the first phase, a ResNet-18 model is trained to detect scanning behaviors at the early stage of an attack, enabling timely prevention. In the second phase, another ResNet-18 model is utilized to identify DDoS attacks, ensuring effective detection of botnet activities.

Keywords: IoT Botnet Attacks, ResNet-18, DDoS Detection, Scanning Attack Prevention, Two-Phase Machine Learning, Intrusion Detection System (IDS)

II.INTRODUCTION

The Internet of Things (IoT) has revolutionized the digital landscape by linking everyday physical objects to the internet, thereby enhancing convenience and functionality in daily life. With the increasing use of smart devices like cameras, televisions, wearables, and lighting systems, connectivity has expanded rapidly. However, many of these devices are deployed with minimal security configurations. Common issues such as default or hardcoded login credentials make them easy targets for cyber threats.

This lack of security has led to a rise in botnet-based cyber-attacks, particularly Distributed Denial of Service (DDoS) attacks, which have grown in frequency and intensity. Typically, a botnet attack begins with a reconnaissance phase, where attackers scan for devices with known vulnerabilities. Once a vulnerable device is found, malware is installed to compromise it and link it to a command-and-control network. These compromised devices can then be collectively used to execute large-scale attacks. One of the most notable examples is the Mirai botnet attack in 2016, which compromised millions of IoT devices and disrupted major internet services like GitHub and AWS.

Tools such as Shodan and Censys have made it even easier for attackers to identify and target unprotected IoT devices, increasing the scale and ease of exploitation. Research indicates that compromised IoT devices are highly susceptible to becoming part of a botnet, reinforcing the need for effective and scalable security solutions. Traditional detection methods are generally categorized into host-based and network-based techniques. Due to the limited processing power and memory of IoT devices, host-based methods are often unsuitable, making network-based approaches more practical. These are typically further divided into signature-based, anomaly-based, and specification-based detection strategies. Among them, anomaly-based detection—especially using machine learning—has shown strong potential in identifying both encrypted and previously unseen threats by monitoring unusual patterns in network activity. However, a major limitation of many machine learning approaches is their dependency on specific datasets, which can reduce their performance when applied to different environments. To address this gap, this work introduces a Dual Phase ML Defense for IoT Botnet Threats. This method focuses on early detection and prevention of attacks by using two phases: the first phase utilizes the ResNet-18 deep learning model to recognize scanning behavior, which helps intercept the attack before devices are compromised. The second phase also employs ResNet-18 to identify DDoS attacks originating from already infected devices. The core contributions of this work include the creation of a rich and diverse dataset, the design of a dual-phase machine learning strategy for both preventing and detecting botnet activity, and a demonstration of the method's strong performance across multiple datasets. This approach offers a more adaptable and effective way to secure IoT networks against evolving botnet threats.

III.EXISTING SYSTEM

Earlier research proposed a graph-based method using Printing String Information (PSI) graphs to detect IoT botnets. High-level features were extracted from function call graphs, and a Convolutional Neural Network (CNN) was trained on these graphs for botnet detection. Later researchers introduced an automated system named BotMark, which detects botnets through a combination of flow-based and graph-based network traffic analysis. Flow-based detection utilized k-means clustering to calculate similarity and stability scores, while graph-based detection employed least-square techniques and Local Outlier Factor (LOF) to measure anomalies. After certain duration a researcher proposed an approach combining frequency analysis, graph visualization, and rules generation to study Mirai botnet attacks using graph theory. However, this approach is limited to Mirai botnet attacks.

With due span of time, experimenters introduced a hybrid botnet detection method implemented at three levels: host, network, and a combination of both. The approach focused on HTTP, P2P, IRC, and DNS traffic using Naïve Bayes and decision tree algorithms for classification. Later they presented BotFP, a bot detection framework with two versions: BotFP-Clus for clustering similar traffic patterns and BotFP-ML for identifying new bots using Support Vector Machines (SVM) and Multi-Layer Perceptron (MLP). Researcher named Soe proposed a two-stage machine learning model for detecting IoT botnet attacks. The first stage involved model training and feature selection, while the second stage decoded packets, extracted features, and used Artificial Neural Networks (ANN), J48 decision trees, and Naïve Bayes for attack detection.

Disadvantages:

- Existing methods only detect botnet attacks during the scanning phase and while identifying DDoS attacks in both inbound and outbound traffic.
- IoT botnet attacks do not always begin with scanning and end with DDoS.

IV.PROPOSED SYSTEM

The proposed system analyzed the most commonly used scanning and DDoS attack techniques and created a comprehensive dataset by generating 33 types of scanning attacks and 60 types of DDoS attacks. This dataset was supplemented with partial integration of samples from three publicly available datasets to ensure maximum coverage and better training for machine learning models.

An Algorithmic ML approach is introduced to prevent and detect both inbound and outbound botnet attacks in IoT networks. The first phase of the approach detects scanning activities to prevent IoT botnet attacks before they progress. The second phase identifies IoT botnet attacks by detecting DDoS activities. To validate the performance of the proposed approach, ResNet-18 models were trained on three different datasets and compared with the proposed two phase model, demonstrating its superiority in preventing and detecting botnet attacks.

Advantages:

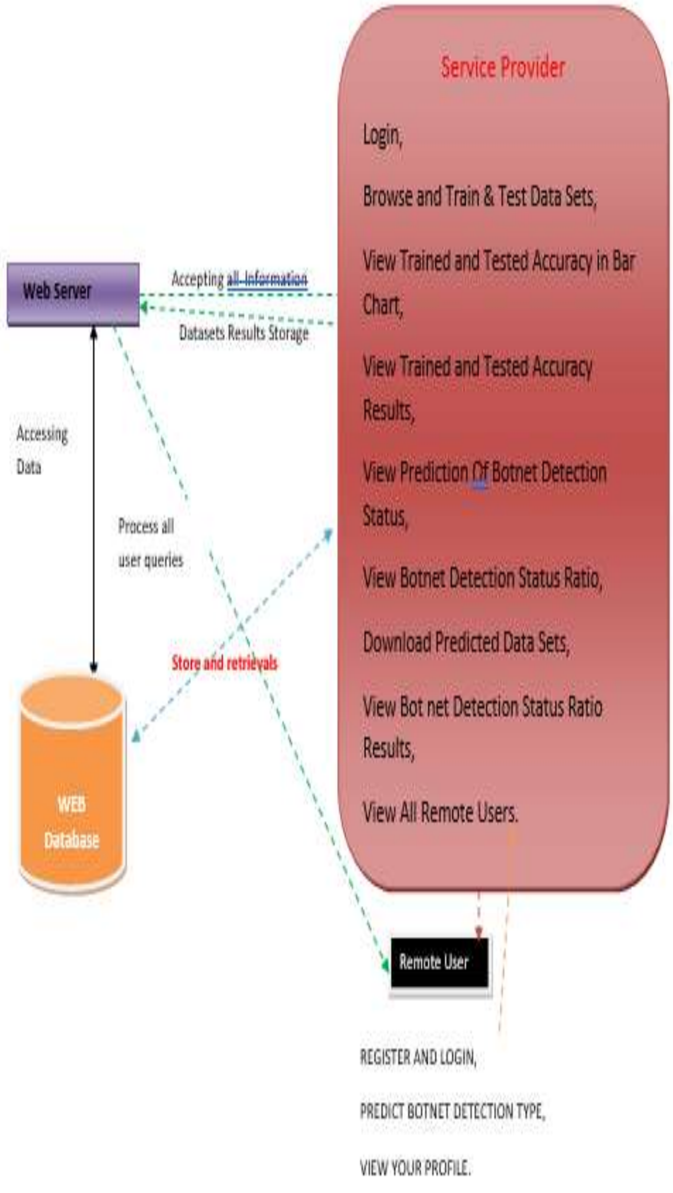
- A novel two-phase machine learning approach is proposed to prevent and detect botnet attacks in IoT environments.
- The approach stops attackers during the scanning phase, preventing further attack progression.

SYSTEM REQUIREMENTS

➢ Hardware Configuration:
- Processor: Pentium IV or higher
- RAM: 4 GB (minimum)
- Hard Disk: 20 GB
- Keyboard: Standard Windows Keyboard
- Mouse: Two or Three-Button Mouse
- Monitor: SVGA

➢ Software Requirements:
- Operating System: Windows 7 Ultimate
- Programming Language: Python
- Front-End: Python
- Back-End: Django ORM
- Design: HTML, CSS, JavaScript
- Database: MySQL (WAMP Server)

Architecture Diagram:-


Service Provider
Login,
Browse and Train & Test Data Sets,
View Trained and Tested Accuracy in Bar Chart,
View Trained and Tested Accuracy Results,
View Prediction Of Botnet Detection Status,
View Botnet Detection Status Ratio,
Download Predicted Data Sets,
View Bot net Detection Status Ratio Results,
View All Remote Users.
Web Server
Accepting all Information
Datasets Results Storage
Accessing Data
Process all user queries
Store and retrievals
WEB Database
Remote User
REGISTER AND LOGIN,
PREDICT BOTNET DETECTION TYPE,
VIEW YOUR PROFILE.

V.LITERATURE SURVEY

The Internet of Things (IoT) has significantly transformed modern life by connecting billions of devices, enabling automation, and improving efficiency. However, with the rapid adoption of IoT technology, security concerns, particularly IoT botnet attacks, have escalated. A botnet attack utilize compromised IoT devices to carry out malicious activities, such as Distributed Denial of Service (DDoS) attacks. In response to these threats, various machine learning-based techniques have been proposed to detect and mitigate botnet attacks.

1. An Intuitive Model for IoT Botnet Prediction Using Machine Learning Algorithms

This paper proposes a two-fold machine learning model designed to prevent and detect botnet attacks by employing both anomaly detection and behavioral analysis. The first fold focuses on proactive prevention, leveraging historical data to establish baseline patterns of IoT device behavior. When deviations from these baselines are detected (e.g., abnormal resource usage or communication patterns), alerts are triggered for further investigation, thus preventing botnet recruitment. The second fold applies real-time detection, continuously monitoring IoT devices for unusual behaviors that indicate botnet attacks. The use of supervised machine learning models, such as Support Vector Machines (SVM), helps differentiate between benign and malicious behaviors. This paper emphasizes the model's adaptability through regular updates to address evolving attack techniques.

Key Contributions:

- Proactive and real-time defense mechanisms.
- Emphasis on the adaptability of machine learning models to emerging threats.
- Focus on reducing false positives/negatives and integrating the model with existing security measures.

2. A Two-Fold ML Approach to Prevent and Detect IoT Botnet Attacks

This paper focuses on a similar two-fold approach, specifically designed to detect IoT botnet attacks at multiple stages. The first fold aims to detect attacks in their premature stages, using the ResNet-18 deep learning model to identify scanning activities before the IoT devices are compromised. The second fold of the approach detects DDoS attacks in real-time, also employing ResNet-18 to differentiate between normal and attack behaviors. This method highlights a significant improvement in accuracy across various datasets, reaching 98.89% accuracy for both scanning and DDoS detection.

Key Contributions:

- High accuracy and precision in detecting both scanning and DDoS attacks.
- Use of ResNet-18, a state-of-the-art deep learning model, in both folds of detection.
- Integration of multiple datasets to ensure broad coverage of diverse attack patterns, addressing the issue of dataset specificity found in other models.

3. Internet of Things Botnet Detection Approaches: Analysis and Recommendations for Future Research

This paper provides a systematic literature review (SLR) of existing IoT botnet detection approaches. The authors categorize detection techniques into anomaly-based, signature-based, and machine learning-based methods, offering a comprehensive analysis of each approach's strengths and weaknesses. The review also highlights the importance of detecting botnets during their early stages, emphasizing the need for real-time monitoring and early intervention. The paper concludes by identifying research gaps in the current literature, such as the limited generalization of models across datasets, and the need for adaptive and scalable solutions.

Key Contributions:

- Comprehensive review of existing IoT botnet detection methods.
- Identification of critical research gaps, such as limitations in dataset generalization and real-time detection challenges.
- Recommendations for future research directions, including early-stage detection and improvements in model scalability and adaptability.

4. An Intuitive Model for Prediction of IoT Botnet Attacks Efficiently by Using Machine Learning Algorithms

This paper reiterates the importance of a two-fold machine learning algorithm for efficiently predicting IoT botnet attacks. Like the other models discussed, this approach combines anomaly detection for early prevention and behavioral analysis for real-time detection. The authors propose using Support Vector Machines (SVM) for the classification of device behaviors, differentiating between benign and malicious activities. The focus is on continuously monitoring IoT devices and adapting the model based on new threats, ensuring that the system remains effective against emerging attack techniques.

Key Contributions:

- A combined anomaly detection and behavioral analysis approach.
- Use of SVM for classifying behaviors and preventing botnet recruitment.
- Focus on regular model updates to address emerging threats and attack patterns.

VI.FUNCTIONALITY

→Service Provider

The Service Provider module facilitates secure access through authentication using a valid username and password. Upon successful login, the service provider is granted access to a suite of functionalities, including:

- Accessing the system dashboard through secure login
- Uploading, training, and testing IoT-related datasets
- Displaying the performance accuracy of trained models using bar charts
- Viewing comprehensive performance metrics for trained and tested models
- Monitoring real-time prediction status for botnet detection
- Reviewing the ratio of successful and unsuccessful botnet detections

- Downloading datasets with prediction results
- Analyzing botnet detection outcomes through visual reports
- Managing and reviewing profiles of all registered remote users

→View and Authorize Users Module

The Administrator module allows the admin to monitor and manage system users. The administrator has access to:

- A complete list of all registered users
- User details including username, email address, and physical address
- The ability to authorize or restrict user access to system functionalities

→Remote User Module

The Remote User module supports multiple user registrations. Each user must register by providing the necessary details, which are then stored securely in the system's database. Upon successful registration and authorization, users can log in and access services. Functionalities include:

- Creating an account and securely logging in
- Using the system to predict the type of botnet threat
- Viewing and managing their personal profile information

VII.TESTING METHODOLOGIES

The following are the various testing methodologies used:

- Unit Testing
- Integration Testing
- User Acceptance Testing
- Output Testing
- Validation Testing

UNIT TESTING

Unit testing focuses on verifying individual units or components of the software design, typically modules. It examines specific control paths within a module to ensure thorough coverage and error detection. Each module is tested independently to ensure it functions properly and matches design specifications. All significant processing paths and error-handling mechanisms are tested to confirm they behave as expected.

INTEGRATION TESTING

Integration testing deals with combining individual software modules and testing them as a group. The goal is to identify issues related to interfacing between modules and ensure they work cohesively to create the intended system.

Types of Integration Testing:

1. Top-Down Integration

This incremental method starts by integrating modules from the top level of the program's control hierarchy. Modules are incorporated starting from the main module and moving downwards, using either a depth-first or breadth-first approach. Individual stubs are replaced as the process moves forward.

2. Bottom-Up Integration

This approach begins at the lowest level of the program hierarchy. Low-level modules are integrated and tested first, eliminating the need for stubs. Clusters of low-level modules that perform specific functions are tested together. Test drivers are used to coordinate inputs and outputs during testing, which are later removed as the integration moves upwards.\

User Acceptance Testing

User acceptance testing (UAT) ensures that the system meets user requirements and functions correctly from the user's perspective. By keeping in contact with prospective users throughout development, feedback is gathered to make any necessary changes. The system is designed to have an intuitive user interface, even for those unfamiliar with it.

Output Testing

Output testing verifies that the system produces the desired outputs in the correct format. The format of outputs is determined based on user needs, both in terms of on-screen displays and printed documents. The system is considered effective only if the outputs meet these specifications.

Validation Testing

Validation checks are conducted on various fields to ensure correct data input.

- Text Fields: Must contain a number of characters within predefined limits. Depending on the context, these fields can be alphanumeric or alphabetic. Incorrect entries trigger error messages.

- Numeric Fields: These fields only accept numbers (0-9). Any other characters will prompt an error. All modules undergo test runs using sample data to verify their correctness. The system is tested with real data to ensure it behaves as expected, and any bugs are identified and corrected.

Preparation of Test Data

Testing involves creating and using test data to uncover errors. This step is crucial for identifying problems and making corrections. The test data helps simulate real-world scenarios, and any errors found during testing are documented for future reference.

Using Live Test Data

Live test data consists of actual data extracted from the organization's files. After partial system development, users are asked to input real data, which can then be used to test the system's performance under real-world conditions. However, live data often does not cover all possible cases, meaning it might not uncover all errors, especially edge cases.

Using Artificial Test Data

Artificial test data is generated specifically for testing purposes. This data is designed to test all possible formats and combinations of inputs. Artificial data allows for comprehensive testing of all system paths and logic. Often, an independent team creates and tests the artificial data to ensure objectivity.

Testing Strategy

A well-structured testing strategy integrates test case design, test planning, execution, and evaluation. It must accommodate both low-level tests, which verify that individual code segments are correct, and high-level tests, which confirm that the overall system meets user requirements. Testing ensures that the system functions as intended and meets the established objectives.

SYSTEM TESTING

System testing integrates the software with other system elements, such as hardware, databases, and users, to ensure that everything functions together as expected. This phase identifies discrepancies between the system's actual performance and its intended objectives or specifications.

UNIT TESTING

Unit testing verifies individual modules against design specifications to ensure they function correctly. This process tests each module's internal logic and ensures that the code written during the development phase operates as expected. Any errors identified in this phase are corrected before integration with other modules.

USER TRAINING

When implementing a new system, user training is essential to help users understand its functionality and how to operate it effectively. The project was demonstrated to the end users, making it easy for them to learn the system. Since the expected users are familiar with computers, the training process is simple and straightforward.

MAINTENANCE

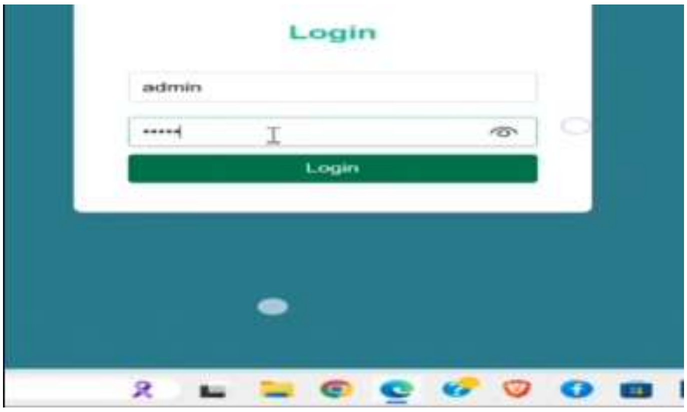
Maintenance involves fixing any errors in the code or design that may arise after deployment. By accurately defining user requirements during development, the need for extensive maintenance is minimized. The system has been designed to be flexible, allowing for future updates and new features as needed. Simple, well-documented coding practices ensure that maintenance will be easy and efficient.

VIII.OUTPUT

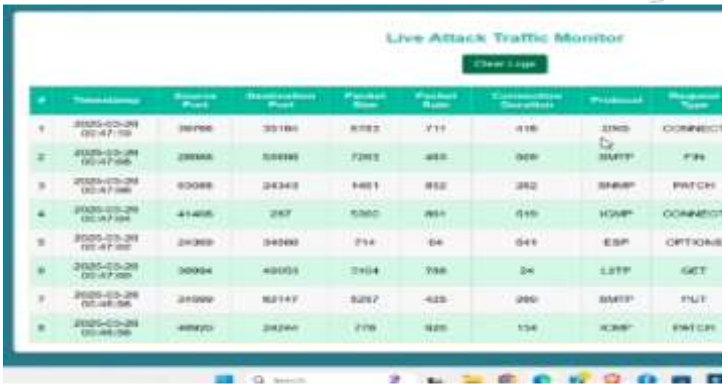
The output of our project, titled "Dual Phase Machine Learning Approach for Detecting IoT Botnet Attack", effectively showcases how machine learning can be employed to enhance cybersecurity within Internet of Things (IoT) environments. The project is designed to work in two sequential phases—data preprocessing and classification. In the initial phase, raw network traffic data is subjected to a series of preprocessing steps such as noise removal, feature encoding, normalization, and feature selection to ensure that the input data is clean, consistent, and optimized for machine learning algorithms. This step is vital as IoT traffic is often characterized by large volumes of unstructured and imbalanced data. Once the dataset is preprocessed, it is passed into the second phase where multiple machine learning models are trained and evaluated for their ability to distinguish between benign and malicious traffic.

In this phase, we implemented and tested several classification algorithms, including Decision Tree, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Random Forest. After extensive evaluation based on key performance metrics such as accuracy, precision, recall, and F1-score, Random Forest emerged as the most effective model for our dataset. It achieved the highest accuracy, indicating that it could correctly identify botnet attacks with greater reliability than the other models. Random Forest's ensemble nature, which relies on building multiple decision trees and averaging their outputs, makes it particularly robust against overfitting and well-suited for handling high-dimensional data, which is common in network traffic logs. Its ability to rank feature importance also provides valuable insights into which network behaviors are most indicative of malicious activity.

To illustrate our findings and make the results more interpretable, we generated a set of visual outputs that will be included here. These visuals include an accuracy comparison chart showing the performance of all tested models, a confusion matrix highlighting the Random Forest model's prediction distribution, a feature importance plot identifying key contributing factors, and an ROC curve evaluating the model's classification thresholds. These outputs collectively demonstrate the reliability, efficiency, and practical applicability of the proposed system in real-time IoT intrusion detection scenarios. Thus, our project not only meets its objective of detecting IoT botnet attacks with high accuracy but also provides a scalable and explainable machine learning solution that can be adopted in broader cybersecurity frameworks.



admin
•••••



#	Timestamp	Source Port	Destination Port	Packet Size	Packet Rate	Connection Duration	Protocol	Request Type
1	2025-03-28 02:47:10	39766	35180	8782	711	416	DNS	CONNECT
2	2025-03-28 02:47:06	28846	50698	7262	465	909	SMTP	FIN
3	2025-03-28 02:47:06	93088	24343	1461	852	262	SNMP	PATCH
4	2025-03-28 02:47:04	41465	287	5060	861	615	ICMP	CONNECT
5	2025-03-28 02:47:02	24369	34060	714	64	641	ESP	OPTIONS
6	2025-03-28 02:47:00	30064	40033	3104	798	24	L2TP	GET
7	2025-03-28 02:46:58	31099	62147	5257	425	269	SMTP	PUT
8	2025-03-28 02:46:56	48920	24244	778	925	134	ICMP	PATCH

IX.CONCLUSION

This study proposes a two-phase machine learning approach for the prevention and detection of IoT botnet attacks. In the first phase, a deep learning model based on ResNet-18, named ResNetScan-1, is used to detect scanning attacks at an early stage. If these attacks go undetected, the second phase activates another ResNet-18 model, ResNetDDoS-1, which identifies DDoS attacks. The models were trained using traffic data from three publicly available datasets and evaluated on separate, unseen datasets to test their generalization capability. Experimental results demonstrated that both ResNetScan-1 and ResNetDDoS-1 achieved high accuracy in detecting scanning and DDoS attacks, outperforming existing techniques. This two-phase framework currently supports detection of 33 types of scanning attacks and 60 types of DDoS attacks. For future work, the system will be enhanced by including a broader range of attack types in the training data and deploying the proposed approach within a real-time Intrusion Detection System (IDS) to evaluate its effectiveness in live network environments.

X. REFERENCES

1. I. Ali, A. I. A. Ahmed, A. Almogren, M. A. Raza, S. A. Shah, A. Khan, and A. Gani, “Systematic literature review on IoT-based botnet attack,” IEEE Access, vol. 8, pp. 212220–212232, 2020.
2. F. Hussain, S. G. Abbas, G. A. Shah, I. M. Pires, U. U. Fayyaz, F. Shahzad, N. M. Garcia, and E. Zdravevski, “A framework for malicious traffic detection in IoT healthcare environment,” Sensors, vol. 21, no. 9, p. 3025, Apr. 2021.
3. MASSCAN. Accessed: May 3, 2021. [Online]. Available: https://github. com/robertdavidgraham/masscan.
4. Kumar, K.; Kumar, N.; Shah, R. Role of IoT to avoid spreading of COVID-19. *Int. J. Intell. Netw.* **2020**, *1*, 32–35. [CrossRef].
5. CICFlowmeter. Accessed: May 3, 2021. [Online]. Available: https://github.com/ahlashkari/CICFlowMeter.
6. Network Traffic Flow Analyzer. Accessed: May 3, 2021. [Online]. Available: http://www.netflowmeter.ca/netflowmeter.html .
7. Pour, M.S.; Mangino, A.; Friday, K.; Rathbun, M.; Bou-Harb, E.; Iqbal, F.; Samtani, S.; Crichigno, J.; Ghani, N. On data-driven uration, learning, and analysis for inferring evolving internet-of-Things (IoT) botnets in the wild. *Comput. Secur.* **2020**, *91*, 101707.
8. R. Vishwakarma and A. K. Jain, “A survey of DDoS attacking techniques and defence mechanisms in the IoT network,” Telecommun. Syst., vol. 73, no. 1, pp. 3–25, Jan. 2020.
9. M. M. Salim, S. Rathore, and J. H. Park, “Distributed denial of service attacks and its defenses in IoT: A survey,” J. Supercomput., vol. 76, no. 7, pp. 5320–5363, 2020.
10. C. Yuan, J. Du, M. Yue, and T. Ma, “The design of large scale IP address and port scanning tool,” Sensors, vol. 20, no. 16, p. 4423, Aug. 2020.