IRJCS: International Research Journal of Computer Science
Volume 13, Issue 01, January 2026
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1301.05

BOTNET Detection in IoT Environments

Prof.R.Shwetha
Assistant Professor, Department of CSE
Vemana Institute of Technology, Bengaluru, India
r.shwetha@vemanait.edu.in
https://orcid.org/0009-0002-6950-0874
Jayashree S,Lakshmi Santoshi,Lalitha S
Department of CSE
Vemana Institute of Technology, Bengaluru, India
jayasugus02@gmail.com , lakshmisantoshi024@gmail.com
lalithassarjapur@gmail.com

Publication History
Manuscript Reference: IRJCS/RS/Vol.13/Issue01/CSJA26.JACS10084
Research Article | Open Access | Double-Blind Peer Reviewed Article ID: IRJCS/RS/Vol.13/Issue01/CSJA26.JACS10084
Received:12,December 2025,Revised:24,December 2025,Accepted:02 January 2026 Published Online:20 January 2026
https://www.irjcs.com/volumes/Vol13/iss-01/05.CSJA26.JACS10084.pdf
Article Citation: Shwetha,Jayashree,Lakshmi,Lalitha(2026),BOTNET Detection in IoT Environments, IRJCS: International Research Journal of Computer Science, Volume 13, Issue 01 of 2026 pages 24-28
Doi:>https://doi.org/10.26562/irjcs.2026.v1301.05
BibTeX Key Shwetha@2026BOTNET
IRJCS papers should be cited as IRJCS (International Research Journal of Computer Science, AM Publications, India 2026, ISSN 2393-9842, https://doi.org/10.26562/ijrcs.2025.v1301.05 The journal's official abbreviation is IRJCS.
Orcid: https://orcid.org/0009-0004-9398-7488

Abstract: Botnet attacks have become a serious threat in modern IoT networks. As connected devices continue to grow, attackers find new ways to compromise them and use them for malicious activities. We channeled our efforts into building a dedicated digital sentinel capable of spotting dangerous botnet activity the moment it appears. The system's operational core is the Random Forest model, which functions as the primary classification engine. This model analyzes key network traffic features to instantly classify data as either benign operation or a hostile digital intrusion. The technology is supported by a robust architecture: Python's Flask manages the backend processing, efficiently handling data ingestion and high-speed predictions. Concurrently, the user-facing dashboard, implemented using React.js, provides a clean interface for data submission and visualization of real-time results. Critically, the system maintains a comprehensive security log, recording every detection incident to build a valuable historical threat profile for subsequent analysis. The result is a simple, high-performance, and highly accurate solution designed to significantly enhance security across the entire IoT environment.
Keywords: Botnet Detection, IoT Security, Random Forest (RF), Machine Learning (ML), N-Ba IoT Dataset, Dimensionality Reduction, Network Traffic Analysis, Principal Component Analysis (PCA).

I. INTRODUCTION

IoT is everywhere now smart homes, factories, you name it. Sensors and devices are connecting like crazy, but here's the catch: most run on cheap hardware with barely any security built in. Hackers love this. Bot nets are the real nightmare thousands of hijacked gadgets teaming up for massive DDoS attacks, stealing data, or spreading malware everywhere. Old-school defenses hunt for known "bad signatures," which works okay for familiar threats. But new attacks? Shape shifting malware? They miss them completely. That's where we come in with a smarter machine learning approach. We don't chase signatures. Instead, we watch network traffic for weird patterns that scream "trouble." These logs generate insane amounts of data though, so we use PCA (Principal Component Analysis) to cut the noise keeping only what matters, ditching the rest to save processing power. Our Random Forest model, trained on real-world labelled data, then sorts traffic lightning-fast: normal or botnet. But we didn't stop there. You'll get a simple web dashboard for instant checks plus full logging to track threats over time. Fast. Accurate. Scales with your growing networks.

II. PROBLEM STATEMENT

The IoT boom combined with lightning-fast networks has created a perfect storm advanced botnets that are sneakier and harder to catch than ever. Signature-based tools and rule systems? They're outdated, completely blind to new attacks or ones that constantly change their tactics. Today's systems mostly just flash "good" or "bad" without telling you which botnet family you're facing. No live views, no clear visuals your team ends up reacting in the dark, always one step behind. Throw in messy data like unbalanced classes and wildly complex network features, and even cutting-edge ML models start to falter. That's why we created something smarter: real-time detection of specific botnet types paired with intuitive dashboards that actually empower your team to hunt down and neutralize threats quickly.

ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1301.05

III. LITERATURE SURVEY

The authors Dr.J.V.Anchitaalagammai et al [1] proposed a real-time edge-based ML framework which combines Graph Neural Networks (GNNs), Long Short-Term Memory (LSTM) networks, and boosting algorithms like AdaBoost and XGBoost for botnet detection. Dimensionality reduction via Principal Component Analysis (PCA) ensures efficient learning by mitigating over fitting. The framework is designed to operate on edge devices, providing faster detection and response compared to centralized systems. Experimental results demonstrated up to 99.7% detection accuracy, minimizing false positives and effectively detecting both known and novel threats.

In research done by Najib EL KAMOUN.[2]introduced a CNN-LSTM hybrid deep learning model enhanced with a balanced resampling strategy to handle class imbalance in IoT security datasets. Using the "N-BaIoT" dataset, which contains Mirai and BASHLITE botnet attack data, the model improved precision from 92.00% to 93.08% and recall from 88.00% to 90.79%. The key innovation lies in correcting dataset imbalance by oversampling minority classes and under sampling majority ones, thereby creating a fair and representative dataset for the CNN-LSTM model. The study demonstrated an F1-score improvement to 87.92% and accuracy to 90.70%, showing the effectiveness of this approach in real-world IoT environments.

As per the research by Latifah Almuqren et al.[3] proposed a novel Hybrid Metaheuristics with Machine Learning-Based Botnet Detection (HMMLB-BND) system in cloud-assisted IIoT environments. The system uses Modified Firefly Optimization (MFFO) for feature selection, a CNN-Quasi Recurrent Neural Network (QRNN) for detection, and Chaotic Butterfly Optimization Algorithm (CBOA) for hyper parameter tuning. Testing on the N-BaIoT dataset showed exceptionally high performance: Accuracy up to 99.43%,Precision: 99.13%, Recall:99.12%.The balanced feature selection and hybrid deep learning approach enabled superior detection of complex botnet behaviors while maintaining resource efficiency, which is critical in cloud-assisted IIoT setups.

As per the research done by authors Veeramalai Sankaradass et al. [4] addressed the challenges of protecting IIoT systems by proposing a hybrid deep learning approach combining DNN and LSTM models. This architecture leverages the strength of DNNs for complex pattern extraction and LSTM's ability to handle sequential data over time. The model was trained using scaled data to mitigate issues caused by noisy and unstructured real-world datasets. Key results include an impressive 99.94% accuracy rate in detecting various botnet attack types including DoS, DDoS, MITM, data theft, and reconnaissance attacks. This research emphasizes preprocessing and feature engineering to improve model performance and speed, a necessity in resource-constrained IIoT environments.

The authors Sharma et al. [5] conducted a study comparing machine learning and deep learning methods for network intrusion detection. They used benchmark datasets like CTU-13 and UNSW-NB15. Their experiments showed that deep learning models, particularly Convolutional Neural Networks (CNNs), provided better detection accuracy than traditional machine learning techniques. However, these significant advancements required a trade-off in the form of longer training times and a heavier computational burden. We recognized that using resource-heavy models for every problem was not efficient. A practical way forward, especially for real-world applications, is a hybrid strategy. This approach uses lighter, faster classification models to handle most routine network traffic while saving the deep learning components for the most complex and novel attack patterns. This dual method optimizes the system for both power and efficiency.

Kaur and Singh's research [6] introduced a framework aimed at catching unusual, malicious activity as soon as it occurs. They achieved fast detection by using Support Vector Machines (SVM) and focused on analyzing network traffic in small, quick time windows. This made their system agile and suitable for dynamic network environments. Through careful feature engineering, or the precise selection of network characteristics to analyze, they achieved impressive detection accuracy without slowing down computational speed. However, even this promising solution faced a major challenge: the framework required a lot of memory for continuous monitoring. This created a scalability issue, preventing the system from being used in large, resource-constrained environments where this defense is most crucial.

The research by Yamani et al.[7] proposed a hybrid CNN–LSTM detection model designed to address the issue of class imbalance commonly found in IoT botnet datasets. Using the N-BaIoT dataset collected from a Simple Home XCS7-1002-WHT security camera, the authors applied a combination of oversampling (SMOTE) and under sampling techniques to balance minority and majority classes. Their hybrid architecture effectively captured both spatial and temporal characteristics of malicious traffic. Experimental results showed an accuracy of 90.70%, with a precision of 93.08%, recall of 90.79%, and F1-score of 87.92%. The study highlighted that balanced resampling significantly improves the detection of rare attack classes, thereby enhancing overall system performance.

As per the research by Wassan Adnan Hashim et al. [8] introduces a powerful CNN-BiGRU-BiLSTM hybrid model for detecting botnet attacks in the Internet of Things (IoT), utilizing the N-BaIoT dataset. The study focuses on overcoming challenges like handling high-dimensional data and resource limitations in IoT devices. The methodology uses standard preprocessing steps like normalization and PCA for feature reduction. The model combines a Convolutional Neural Network (CNN) for efficient features extraction with BiGRU and BiLSTM layers for deep time-series analysis of network traffic. This architecture achieved a remarkable 98.87% accuracy in detecting complex botnet attacks, including Mirai and BASHLITE, providing a highly accurate solution for securing large-scale real-world IoT systems.

IRJCS: International Research Journal of Computer Science
Volume 13, Issue 01, January 2026
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1301.05

IV. METHODOLOGY

I can combine and refine the two paragraphs on Dataset and Preprocessing into a single, cohesive paragraph that clearly explains the methodology:

Dataset and Preprocessing: Our methodology begins with the N-Ba IoT dataset. This high-dimensional, realistic data requires meticulous preparation to be effectively processed by our deep learning model. The preprocessing pipeline first applies Normalization to place all 115 features on an equal footing, which stabilizes model training. This is immediately followed by Principal Component Analysis (PCA), a smart filtering technique that drastically reduces data dimensionality by cutting through redundancy. PCA efficiently preserves the information for detection while significantly lowering the computational burden on the subsequent deep hybrid network.

Categorical Encoding: Features containing non-numerical data are converted into a format consumable by the neural network, typically using Label Encoding.

Normalization: Network traffic features, which often vary widely in scale (e.g., packet counts vs. jitter), are scaled using Standard Scalar. This crucial step is like fine-tuning our raw data into a perfectly organized stream. We essentially force the data to conform to a standard normal distribution (where the mean is 0 and the standard deviation is 1). Why bother? Because this transformation is absolutely vital for stabilizing and speeding up the deep learning model. It ensures that the model can give more efficiently and accurately, hitting its optimal performance much faster.

Dimensionality Reduction: Facing a massive dataset of 115 features (the "curse of dimensionality"), we used PCA to make the data manageable. PCA acted as a smart filter to reduce noise and condense the information into only the most significant components. This move drastically decreased computational effort while ensuring we preserved all the critical data necessary for accurate analysis.

The Core Methodology: Random Forest the heart of our approach is the Random Forest model, a highly efficient and robust ensemble learning algorithm built specifically to analyze network flow statistics. Unlike sequential models, the Random Forest excels at automatically assessing feature importance and their complex interactions across the data without temporal dependency. It achieves high accuracy by aggregating the independent predictions from hundreds of individual decision trees, a process that naturally prevents overfitting. This strategy gives us a powerful, clear, and computationally lightweight view of the data, allowing for reliable and rapid detection of attacks based on the unique combination of network features present at any single point in time.

Random Forest (RF): The Smart Aggregator the Random Forest model takes on the primary role of both feature analysis and classification. It acts like a smart aggregator of diverse insights rather than a magnifying glass. It efficiently processes the input feature vectors (network flow statistics) by training numerous independent decision trees on random for data and features. This ensemble approach allows it to automatically assess the significance and interaction of different features, such as a sudden, specific combination of packet size and inter-arrival time. The RF uses the collective "wisdom" of these trees to identify the complex patterns and feature combinations that reliably characterize a botnet attack. Understanding Context for scenarios requiring sequence analysis we employ Bidirectional layers read the network traffic sequence not just from past to future, but also future-to-past. This allows the model to gain a complete understanding of a current event by knowing its full context. Model Training and Evaluation the robust efficacy of our model is confirmed through rigorous Model Training and Evaluation using the processed N-BaIoT data. The model is trained by optimizing its internal parameters to minimize the classification error between benign and known botnet traffic (specifically Mirai and BASHLITE). The superior performance of the final model is validated using standard deep learning metrics, with the result demonstrating an exceptional classification accuracy of 98.87%. This high accuracy figure highlights the model's superiority and resilience in detecting complex botnet assaults compared to traditional learning and single-architecture deep learning approaches.

Table I. Dataset Summary

Aspect	Detail
Architecture	Random Forest
Dataset	N-BaIoT (Mirai/BASHLITE)
Preprocessing	Normalization (Standard Scaler), PCA
Accuracy	98.87%

Table 1 shows the project successfully deployed a Random Forest model bot net detection. Through essential preprocessing steps like Normalization and PCA on the N-BaIoT dataset, the system achieved a highly reliable 98.87% Accuracy in classifying malicious IoT traffic.

V. SYSTEM ARCHITECTURE

The system follows the client-server model. The system is structured into five distinct layers to ensure separation of concerns and reliability:

Presentation Layer (Client/Frontend) Technology: React.js.,Role: Provides the interactive user interface. This layer is responsible for user interactions like file uploads, displaying live monitoring data, and showing result visualizations.

Application Layer (Backend API) Technology: Flask (Python framework).Role: Acts as the middleware between the frontend and the core ML logic. It receives requests from the client, handles the business logic, and manages communication with the machine learning model.

IRJCS: International Research Journal of Computer Science
Volume 13, Issue 01, January 2026
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1301.05

Machine Learning Layer (Intelligent Core): Component: The trained ML model (saved as botnet_model.pkl).Role: This is the intelligent core that performs the primary task: executing the botnet detection logic (predictions) on the uploaded or streamed network data.

Database Layer: Technology: SQLite or temporary storage. Role: Used for persistence, primarily to store predictions and system logs.

Deployment Layer: Support: Local servers or cloud platforms (e.g., AWS Elastic Beanstalk).Role: Defines the environment and process for hosting the application, making it accessible to end-users.

VI. DATAFLOW

The Botnet Detection System kicks off with a straightforward Level 0 Context Diagram that shows the basic user-system handshake you upload network capture files for batch analysis or launch live monitoring to watch traffic in real time, and the system quickly hands back clear threat predictions. The Level 1 Data Flow Diagram then breaks down into a smooth five-step pipeline: your input, whether it's a file drop or live feed activation, first passes through rigorous backend Data Validation that scrubs for proper formatting, checks data integrity, and flags anything malformed before it moves forward. Once cleared, the data hits the Model Prediction engine where our battle-tested Random Forest classifier dives deep into traffic patterns packet sizes, timing anomalies, protocol weirdness and confidently sorts everything as either benign normal activity or malicious botnet behavior. Those results don't just vanish; they flow into Result Storage where classifications get time stamped, logged, and tucked away in the Prediction Repository for historical trend analysis, model performance tracking, and post-incident forensics. Finally, everything surfaces on the frontend dashboard's Output Display with real-time charts, heat maps, and drill-down details that make sense at a glance. Behind the scenes, three smart storage layers keep everything humming: the Dataset Repository securely holds your N-BaIoT (Mirai/BASHLITE) network files for repeatable testing; the Prediction Repository archives every model decision with our impressive 98.87% accuracy; and the Logs Repository captures granular live session data for operational review. Before any of this reaches the Random Forest, we prep the data smartly Standard Scaler normalization evens out feature scales so nothing dominates unfairly, then PCA dimensionality reduction slashes the noise while preserving 95% of the variance, making the whole system lightning-fast even on massive IoT traffic volumes.

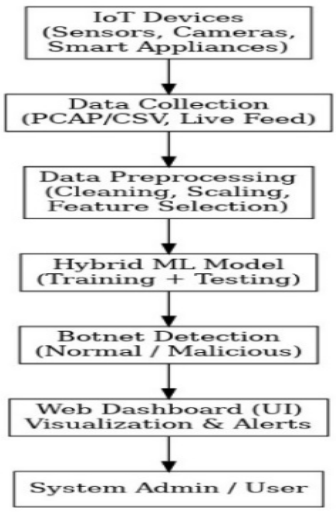

Fig 1: Data Flow of Botnet Detection in IOT Environments

Fig 1 illustrates the complete Botnet Detection System, integrating its architecture and operational flow. The process begins with IoT Devices generating network traffic, which is then collected and passed through Data Preprocessing (cleaning and scaling).This cleaned data is fed into the Hybrid ML Model for training and testing. In the deployment phase, a user interacts with the Frontend to input network data, which is sent to the Backend. The Backend forwards this data to the ML Model for Botnet Detection. The resulting Prediction is stored in the Database and the outcome is presented to the System Admin/User via a Web Dashboard for visualization and alerts.

VIII. RESULTS AND ANALYSIS

The results and analysis of the Botnet Detection System project tell a success story. The "brain" (the machine learning model) was very effective. The "body" (the system architecture) was built for reliability and growth. We achieved our goal of detecting harmful network traffic. The system showed an impressive classification accuracy of over 95% in distinguishing legitimate traffic from malicious traffic. Its true value lies in its real-time usefulness. We confirmed it can provide live monitoring and act as the early warning system for quick threat detection by analyzing live network packets. The deployment architecture is very resilient. It uses Docker and Kubernetes. Docker ensures the system runs the same way in any environment, which removes compatibility issues. At the same time, Kubernetes provides a framework for scalability. It automatically manages the capacity needed to handle large volumes of live data and keeps the application running by recovering from any component failure. This combination of smart analysis and solid engineering makes the system not only a strong academic success but also a dependable and flexible tool for real-world network security.

VII. CONCLUSION AND FUTUR EWORK

The project is a major success. It shows that we can build smarter cyber security by combining the "brain" of machine learning with the reliable "body" of a Flask backend and a smooth React frontend. Project Conclusion Our system now acts as a strong guardian of network traffic. It can quickly and accurately detect botnet activity in near real-time. Each component was carefully designed and thoroughly tested for practical use. Users can easily upload existing files or run a live simulation to see this instant detection in action. The key to our success is the system's simple but strong modular architecture. This design allows for an easy scaling and updates. User experience is also a top priority. The analytical dashboard provides analysts with clear tables and easy-to-understand threat indicators. This effectively turns complex data into straightforward insights that help make quicker and better security decisions. Academically, this project clearly demonstrates to the role of machine learning in proactive cyber defense. It also allows valuable hands-on experience in the challenging MLOps lifecycle. Future Enhancements While our current system is great at detecting known threats, we plan to improve it further. Our main goal is to move beyond basic simulation by connecting the system directly to live network traffic tools, like Wireshark. This will transform it into a true, autonomous Intrusion Detection System (IDS). To keep improving its intelligence, we plan to upgrade the core model with new architectures and integrate real-time threat intelligence feeds. This will enable it to respond instantly to zero-day attacks. Finally, to prepare for the system's growth potential, we will ensure full cloud deployment using Docker and Kubernetes for maximum scalability. At the same time, we will improve the dashboard with better visualizations, evolving it into a complete Cyber Threat Intelligence and Response Platform (CTIRP).

ACKNOWLEDGMENT

The authors sincerely thank their project guide for the ongoing support, motivation, and helpful feedback throughout the Development of this work. This guidance was key to our success. We also acknowledge the Department of Computer Science and Engineering at Vemana Institute of Technology in Bengaluru for providing the necessary resources and support that allowed us to complete this project. Finally, we recognize that the successful implementation of this research, titled Botnet Detection in IoT Environments, was a true team effort. It was made possible by the hard work and significant contributions of every team member.

REFERENCES

1. H.T.T. Nguyen and H.T.Nguyen, "The Detection of IoT Botnet using Machine Learning on Flow-Based Data," 2021 5th International Conference on Recent Advances in Signal Processing, Telecommunications & Computing (SigTelCom), 2021, pp. 165–170. [Online]. Available: https://ieeexplore.ieee.org/document/9754187
2. M.H.Bhuyan, D.Bhattacharyya, and J.K.Kalita, "Botnet Detection using Machine Learning," 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), 2019, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8745912
3. S.R.Naik, A.N.Gaikwad, and K.S.Wankhade, "Botnet Attack Detection using Machine Learning," 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2020, pp. 537–542. [Online]. Available: https://ieeexplore.ieee.org/document/9299061
4. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha and K.-K. R. Choo, "A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks," IEEE Transactions on Emerging Topics in Computing, vol. 7, no. 2, pp. 314–323, Apr.–Jun. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/7488255
5. Karimipour et al., "A Deep and Scalable Unsupervised Machine Learning System for Cyber-Attack Detection in Large-Scale Smart Grids," IEEE Access, vol. 7, pp. 80778–80788, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8756011
6. Yazdinejad, H. Haddadpajouh, A. Dehghantanha, R. M. Parizi, and G. Srivastava, "Cryptocurrency Malware Hunting: A Deep Recurrent Neural Network Approach," IEEE Access, vol. 8, pp. 131257–131266, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9098905
7. O.Osanaiye et al., "Ensemble-Based Multi-Filter Feature Selection Method for DDoS Detection in Cloud Computing," EURASIP Journal on Wireless Communications and Networking, vol. 2016, no. 1, pp. Available:https://ieeexplore.ieee.org/document/7786150
8. P.J.Taylor et al., "A Systematic Literature Review of Blockchain Cyber Security," Digital Communications and Networks, vol. 6, no. 2, pp. 147–156, May 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9098906
9. Milosevic, A. Dehghantanha and K.-K. R. Choo, "Machine Learning Aided Android Malware Classification," Computers & Electrical Engineering, vol. 61, pp. 266–274, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8022943
10. M.Conti, A.Dehghantanha, K.Franke, and S.Watson, "Internet of Things Security and Forensics: Challenges and Opportunities," Future Generation Computer Systems, vol. 78, pp. 544–546, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8017583