International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct-2026, PP 01-06

Machine Learning For Botnet Detection: A Review Of Flow-Based, Behavioural And Deep Learning Approaches For Network Security

Hulwale Aditya Jaydas[1]; Prof. C. A. Kolse[2]; Prof. M. C. Kolse[3]; Prof. S. K. Said[4]
Department Of Artificial Intelligence And Data Science Jaihind College Of Engineering Pune, Maharashtra, India

Abstract—The rapid growth of internet-connected computers and Internet of Things devices has expanded the pool of machines that attackers can compromise and organize into botnets. A botnet is a network of infected devices controlled by an attacker through command-and-control channels and used for distributed denial-of-service attacks, spam, credential theft and further intrusion. Signature-based tools struggle to detect new botnet families, encrypted communication and rapidly changing infrastructure. This paper reviews Machine Learning (ML) approaches for detecting botnet activity from network traffic. It covers botnet architectures and life cycle, traffic features derived from flows such as packet size, connection duration, destination diversity and communication periodicity, and detection methods ranging from clustering of correlated behaviour and supervised classifiers such as Random Forest, Decision Tree, Support Vector Machine and k-Nearest Neighbours to autoencoder-based anomaly detection for IoT devices and recurrent models for algorithmically generated domains. Benchmark datasets including CTU-13, N-BaIoT and Bot-IoT are discussed. A layered detection architecture with traffic collection, feature engineering, hybrid detection, correlation and response is proposed, together with research gaps, objectives and a comparison of algorithms. The review concludes that combining flow-level ML with host correlation and continuous retraining provides early and reliable botnet detection.

Index Terms—Botnet detection, machine learning, network traffic analysis, NetFlow, IoT security, random forest, anomaly detection, DDoS

I. Introduction

Botnets are among the most damaging tools in the attacker's arsenal. By controlling thousands or millions of compromised machines, an attacker can generate enormous volumes of traffic, send spam at scale or harvest credentials. The Mirai botnet, analysed by Antonakakis *et al.* [1], infected hundreds of thousands of poorly secured IoT devices such as cameras and routers by trying default passwords, and launched some of the largest denial-of-service attacks recorded at the time.

Detecting botnets is difficult because infected devices often behave normally most of the time, botmasters change domains and servers frequently, and communication is increasingly encrypted or hidden in common protocols. Machine learning can recognize the subtle behavioural patterns that distinguish bot traffic, such as regular check-ins with a server or synchronized activity across many hosts.

This paper reviews ML-based botnet detection, proposes a layered detection architecture and discusses the challenges of applying these methods in real networks.

II. Background and Related Work

2.1 Botnet Architectures and Life Cycle

Botnets are classified by their command-and-control (C&C) structure, as shown in Fig. 1. Centralized botnets use IRC or HTTP servers; peer-to-peer botnets distribute commands among bots, removing single points of failure; and hybrid designs combine both. The life cycle includes infection, joining the C&C channel, receiving commands, executing attacks and updating the bot software.

International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct-2026, PP 01-06

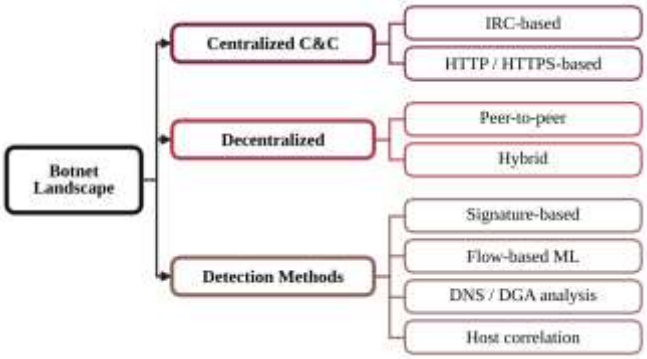


Fig. 1. Classification of botnets by command-and-control structure and detection approach.

2.2 Network Flow Data

A flow summarizes packets sharing the same source and destination addresses, ports and protocol over a period. Flow records exported in formats such as NetFlow [2] include start time, duration, packet and byte counts and TCP flags. Flows are compact enough to collect on large networks and do not require inspecting encrypted payloads.

2.3 Review of Related Studies

Gu *et al.* proposed two influential systems. BotSniffer [3] detected centralized C&C channels by looking for spatial-temporal correlation: bots in the same botnet tend to respond to commands in similar ways at similar times. BotMiner [4] generalized this idea by clustering hosts with similar communication patterns and similar malicious activity, making detection independent of the C&C protocol and structure.

Bilge *et al.* [5] developed Disclosure, which identifies C&C servers from large-scale NetFlow data using features such as flow size patterns, client access patterns and temporal behaviour, showing that flow data alone is sufficient for useful detection. García *et al.* [6] released the CTU-13 dataset of real botnet traffic mixed with normal traffic and compared several detection methods, revealing that performance varies widely between scenarios and that evaluation methodology strongly affects reported results.

Many botnets locate their servers through domain generation algorithms (DGAs) that produce large numbers of pseudo-random domain names. Yadav *et al.* [7] detected such domains from the statistical distribution of characters, and Woodbridge *et al.* [8] used an LSTM network that learns DGA patterns directly from domain strings without hand-crafted features.

For IoT botnets, Meidan *et al.* [9] proposed N-BaIoT, training a deep autoencoder per device on benign traffic and flagging deviations, which detected Mirai and BASHLITE infections quickly with few false alarms. Koroniotis *et al.* [10] created the Bot-IoT dataset with realistic IoT traffic and attacks for training and evaluation. Wang *et al.* [11] converted traffic into images and used CNNs to classify malware traffic. Table I summarizes these works.

TABLE I Summary of Literature Survey

Ref.	Method	Key Findings	Limitations
[3]	Spatial-temporal correlation (BotSniffer)	Detected centralized C&C channels	Limited to IRC/HTTP botnets
[4]	Clustering of communication and activity (BotMiner)	Protocol- and structure-independent detection	Needs malicious activity to occur
[5]	Large-scale NetFlow analysis (Disclosure)	Identified C&C servers from flows alone	False positives on some services
[6]	CTU-13 dataset and method comparison	Showed evaluation strongly affects results	Limited botnet families
[7]	Character distribution of domains	Detected algorithmically generated domains	Dictionary-based DGAs evade
[8]	LSTM on domain strings	Feature-free DGA classification	Needs representative training DGAs

Ref.	Method	Key Findings	Limitations
[9]	Per-device deep autoencoders (N-BaIoT)	Fast IoT botnet detection, few false alarms	Model per device type
[10]	Bot-IoT dataset	Realistic IoT attack traffic for research	Testbed traffic
[1]	Measurement study of Mirai	Explained IoT botnet growth and attacks	Descriptive; no detector

III. Research Gap and Objectives

3.1 Identified Research Gaps

The following gaps are identified:

- Models trained on one dataset or network often perform poorly on others because traffic characteristics differ.
- Encrypted C&C traffic limits payload-based features and requires better flow and timing features.
- Many studies evaluate on balanced data, whereas real botnet traffic is a tiny fraction of total traffic.
- Real-time, scalable deployment on high-speed links is rarely demonstrated.
- Explaining detections to network operators remains limited.

3.2 Objectives of the Review

The objectives of this review are:

- **O1:** To study botnet architectures and their observable network behaviour.
- **O2:** To identify effective flow-based features for distinguishing bot and normal traffic.
- **O3:** To review and compare ML algorithms for botnet detection.
- **O4:** To propose a layered, real-time botnet detection architecture.
- **O5:** To discuss challenges and future research directions.

IV. Design of the Proposed System

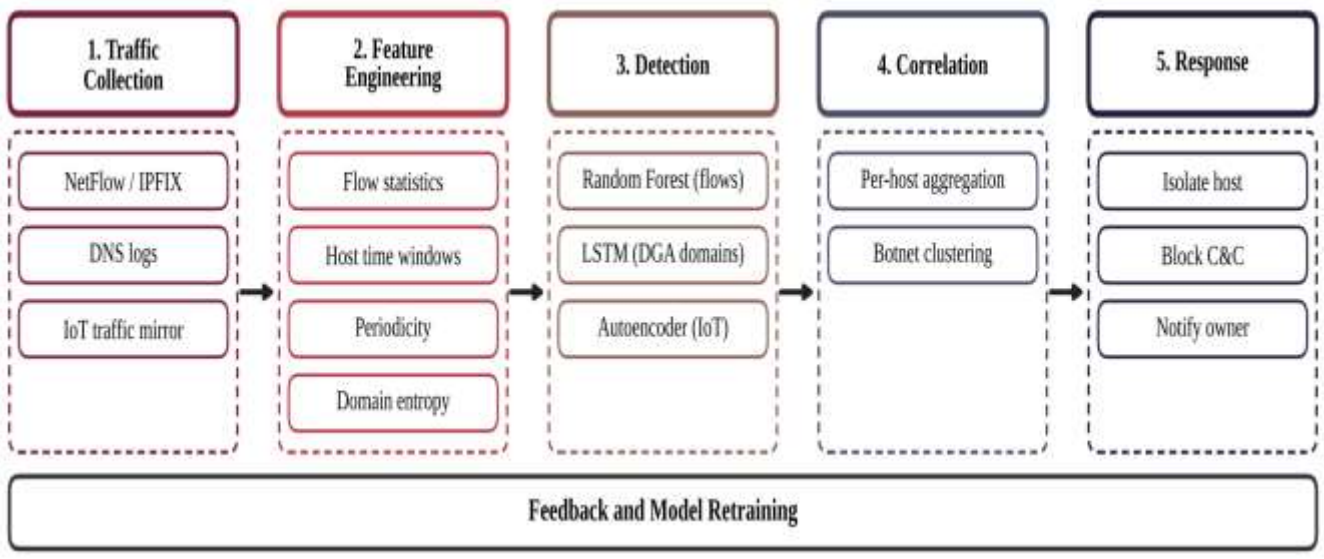


Fig. 2 shows the proposed architecture, which combines per-flow classification with host-level correlation.

Fig. 2. Proposed layered architecture for ML-based botnet detection**.**

4.1 Traffic Collection Layer

Flow records are exported from routers and switches, DNS query logs are collected from resolvers, and, for IoT segments, traffic is mirrored to a dedicated sensor.

B. Feature Engineering Layer

For each flow and each host over sliding time windows, features are computed: packet and byte statistics, connection duration, number of distinct destinations and ports, ratio of failed connections, and periodicity of connections to the same destination. Domain names are scored for randomness.

4.2 Detection Layer

A Random Forest [12] classifies flows as benign or botnet-related, an LSTM scores domain names for DGA likelihood, and an autoencoder flags anomalous behaviour of IoT devices.

4.3 Correlation Layer

Detections are aggregated per host. Hosts that share suspicious destinations, show synchronized activity or contact the same newly registered domains are clustered, identifying likely members of the same botnet.

4.4 Response Layer

Confirmed bots are isolated from the network, C&C addresses and domains are blocked, and owners are notified. Analyst verdicts are fed back for retraining.

4.5 Methodology

The workflow is presented in Fig. 3. Periodic beaconing to a C&C server is a strong indicator. For the inter-arrival times Δt_i of connections from a host to one destination, the coefficient of variation

$$CV = \sigma(\Delta t) / \mu(\Delta t) \qquad (1)$$

is close to zero for automated, regular check-ins and higher for human-driven activity. Domain randomness can be measured by character entropy

$$H = -\Sigma\, p_i \cdot \log_2 p_i \qquad (2)$$

where p_i is the frequency of character i in the domain; DGA domains typically have higher entropy than human-chosen names.

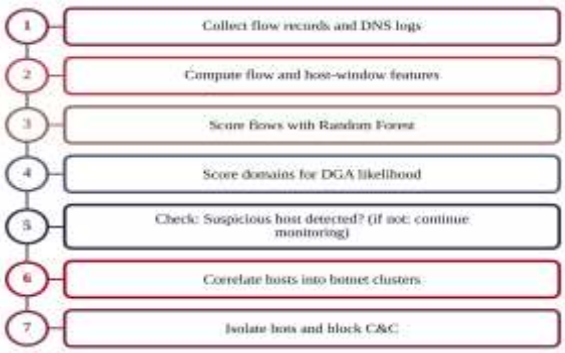


Fig. 3. Botnet detection workflow.

Models are evaluated using precision, recall, F1-score and the false-positive rate on time-separated test data, to reflect how they would perform on future traffic.

V. Discussion and Comparison

Table II compares algorithms. Random Forest provides the best overall balance of accuracy and speed; Decision Trees are interpretable; SVM performs well on smaller data; KNN is simple but slow at prediction time; and deep models excel on raw sequences and IoT anomalies.

Algorithm	Accuracy	Speed	Remarks
Decision Tree	Medium–High	Very fast	Readable rules
Random Forest	High	Fast	Robust to noise
SVM	High	Slow on big data	Kernel choice matters
KNN	Medium	Slow prediction	Simple, memory heavy
Autoencoder	High (anomalies)	Medium	Unsupervised; IoT
LSTM	High (sequences)	Medium	DGA and timing

TABLE II Comparison of ML Algorithms for Botnet Detection

VI. Challenges and Future Directions

Table III lists the main challenges. Future research directions include graph neural networks that learn from host communication graphs, federated detection across internet service providers, analysis of encrypted traffic through TLS metadata, and lightweight detectors running directly on IoT gateways to stop infections at the edge.

Challenge	Mitigation
Encrypted C&C	Timing, size and TLS metadata features
Dataset bias	Cross-dataset testing; local fine-tuning
Class imbalance	Class weights; precision–recall metrics
High-speed links	Flow sampling; streaming processing
Evolving families	Continuous retraining; anomaly models

TABLE III Challenges and Mitigation

VII. Conclusions

Botnets exploit the scale of the internet and the weak security of many connected devices. The reviewed literature shows that flow-based features capture the essential behaviour of bots, that correlation across hosts reveals coordinated activity, and that ML models from Random Forests to autoencoders and LSTMs can detect botnets with high accuracy. Realistic evaluation, attention to class imbalance and adaptation to new families are essential for operational success. A layered architecture combining flow classification, domain analysis, host correlation and automated response can strengthen network security and protect digital systems from automated attacks.

7.1 Acknowledgment

The authors express their sincere gratitude to the Department of Artificial Intelligence and Data Science, Jaihind College of Engineering, Pune, for providing the facilities, guidance and encouragement needed to carry out this work. The authors also thank the faculty members and colleagues whose valuable suggestions helped improve the quality of this review.

References

1. M. Antonakakis et al., “Understanding the Mirai botnet,” in Proc. 26th USENIX Security Symp., 2017, pp. 1093–1110.
2. B. Claise, “Cisco Systems NetFlow services export version 9,” IETF RFC 3954, 2004.
3. G. Gu, J. Zhang, and W. Lee, “BotSniffer: Detecting botnet command and control channels in network traffic,” in Proc. NDSS, 2008.
4. G. Gu, R. Perdisci, J. Zhang, and W. Lee, “BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection,” in Proc. 17th USENIX Security Symp., 2008, pp. 139–154.
5. L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, “Disclosure: Detecting botnet command and control servers through large-scale NetFlow analysis,” in Proc. 28th Annu. Comput. Security Appl. Conf. (ACSAC), 2012, pp. 129–138.
6. S. García, M. Grill, J. Stiborek, and A. Zunino, “An empirical comparison of botnet detection methods,” Comput. Secur., vol. 45, pp. 100–123, 2014.
7. S. Yadav, A. K. K. Reddy, A. L. N. Reddy, and S. Ranjan, “Detecting algorithmically generated malicious domain names,” in Proc. 10th ACM Internet Meas. Conf. (IMC), 2010, pp. 48–61.
8. J. Woodbridge, H. S. Anderson, A. Ahuja, and D. Grant, “Predicting domain generation algorithms with long short-term memory networks,” arXiv:1611.00791, 2016.
9. Y. Meidan et al., “N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders,” IEEE Pervasive Comput., vol. 17, no. 3, pp. 12–22, 2018.
10. N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, “Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset,” Future Gener. Comput. Syst., vol. 100, pp. 779–796, 2019.
11. W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, “Malware traffic classification using convolutional neural network for representation learning,” in Proc. Int. Conf. Inf. Netw. (ICOIN), 2017, pp. 712–717.
12. L. Breiman, “Random forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.