Botnet C2 Traffic Detection Using Deep Learning

Ch. Indra Rao [1], K. Pranay[2], K. Yathi Sri[3], K. Bhanu Prakash[4], M. Mohan Moulika Devi[5]
Department of Computer Science & Engineering (AI & ML)
Avanthi Institute of Engineering & Technology, Vizianagaram, India
indraraoch@gmail.com[1], pranay4640@email.com[2], yathisrikoyya@gmail.com[3],
kamavarapubhanuprakash@gmail.com[4], mulagapakamouli2004@gmail.com[5]

Abstract

Botnets represent a persistent and evolving threat to network security, utilizing covert command-and-control channels to coordinate malicious activities across compromised hosts. Traditional signature-based detection systems fail against encrypted and polymorphic C2 traffic that operates over standard protocols. This paper presents a flow-based deep learning system for detecting botnet C2 communications using temporal and spatial patterns in network metadata. We transform flow records into fixed-length sequences capturing packet sizes, inter-arrival times, and directional features, then apply one-dimensional convolutional neural networks and recurrent architectures to distinguish C2 behaviors from benign traffic. Evaluated on CTU-13 and IoT-23 datasets containing diverse botnet families including Mirai, Okiru, and Torii, our approach achieves high detection accuracy while operating solely on flow-level metadata without payload inspection. The system integrates preprocessing pipelines, deep learning models, alerting mechanisms, and an interactive dashboard for security operations, demonstrating practical deployment feasibility for early C2 detection before large-scale attacks occur.

Index Terms—*Botnet detection, command and control, deep learning, convolutional neural networks, LSTM, flow-based analysis*

I. INTRODUCTION

Botnets constitute networks of compromised computing devices remotely controlled by adversaries through command-and-control infrastructure. Modern botnets exploit diverse platforms ranging from traditional computers to Internet of Things devices such as cameras, routers, and smart appliances. Once infected, these devices periodically communicate with C2 servers to receive instructions, download additional payloads, and participate in coordinated attacks including distributed denial of service, credential theft, and lateral movement within networks.

Contemporary C2 channels employ sophisticated evasion techniques to avoid detection. Rather than utilizing obviously malicious custom protocols, attackers tunnel C2 traffic over common protocols such as HTTP, HTTPS, and DNS, or leverage encrypted channels indistinguishable from legitimate traffic at the payload level. This evolution renders traditional signature-based intrusion detection systems ineffective, as they cannot inspect encrypted payloads or recognize novel attack patterns.

Despite encryption and obfuscation, research on large-scale datasets like CTU-13 and IoT-23 demonstrates that C2 flows exhibit characteristic patterns in packet timing, size distributions, flow duration, and directionality that differ from benign traffic. These subtle but consistent patterns provide opportunities for detection using flow-level metadata analysis rather than deep packet inspection.

Traditional intrusion detection approaches face several limitations. Signature-based systems like Snort and Suricata match packets against known attack databases but fail on zero-day threats and encrypted communications. Statistical anomaly detection monitors aggregate features but struggles with low-and-slow C2 traffic maintaining minimal profiles. Classical machine learning methods require extensive feature engineering and often treat flows independently, discarding temporal ordering information crucial for identifying periodic heartbeats and multi-stage handshakes.

This paper proposes a deep learning-based detection system operating on flow records rather than raw packets. We transform each flow into fixed-length sequences capturing temporal and spatial patterns, then apply one-dimensional convolutional neural networks to identify local motifs and recurrent architectures like LSTM and GRU to model long-range dependencies. The system leverages behavior-oriented labels from public datasets distinguishing benign traffic from specific malicious activities including C&C, HeartBeat, DDoS, FileDownload, and botnet family signatures.

Our contributions include: (1) a comprehensive preprocessing pipeline transforming diverse flow datasets into unified sequence representations; (2) implementation and comparison of 1D CNN and LSTM/GRU architectures for C2 detection; (3) integration of trained models into an operational pipeline with alerting and visualization capabilities; and (4) evaluation on benchmark datasets demonstrating practical deployment feasibility. The remainder of this paper is organized as follows: Section II reviews related work; Section III details our methodology; Section IV presents results; and Section V concludes with future directions.

II. RELATED WORK

Research on botnet detection has evolved from signature matching to sophisticated machine learning approaches. Early systems relied on identifying malicious payload patterns and known command strings. As attackers adopted encryption and covert channels, researchers emphasized behavioral characteristics observable even when payload content remains hidden, including packet timing, flow duration, and protocol usage patterns.

A. Flow-Based Detection Methods

Flow-based analysis represents a middle ground between packet-level inspection and coarse traffic statistics. Several systems cluster hosts with similar connection patterns to identify botnet-like behavior. Others analyze communication planes considering connectivity similarities and activity planes examining malicious behaviors. However, many early approaches used relatively simple statistics without fully exploiting sequential structure within flows.

Recent work on encrypted traffic analysis demonstrates that sequence-level features like packet lengths and inter-arrival times can distinguish application types and malware families even without payload access. This insight motivates sequence modeling where flows become ordered series of feature vectors rather than static summaries.

B. Public Datasets for Botnet Research

The CTU-13 dataset contains traffic from thirteen botnet scenarios mixed with normal background traffic. Each scenario corresponds to specific malware samples with flows labeled as Background, Botnet, C&C Channels, or Normal. This dataset enables studying protocol usage, packet size distributions, and temporal behavior differences between benign and botnet traffic.

The IoT-23 dataset focuses on malicious and benign IoT traffic including multiple malware families like Mirai, Okiru, and Torii. Labels assigned at flow level include Benign, Attack, C&C, HeartBeat, FileDownload, DDoS, and PartOfAHorizontalPortScan, created through careful manual analysis combining payload inspection, behavioral patterns, and threat intelligence. Additional datasets like ISCX Botnet and CIC-IDS provide rich feature sets supporting both binary and multi-class classification tasks.

C. Machine Learning for Intrusion Detection

Classical machine learning techniques including decision trees, random forests, support vector machines, and ensemble methods have been widely applied to intrusion detection. These models typically operate on hand-crafted aggregate features like average packet size, standard deviation of inter-arrival times, or TCP flag counts. While achieving promising accuracies, they rely heavily on manual feature engineering and often discard temporal ordering information by treating flows as independent samples.

D. Deep Learning Approaches

Deep learning has gained traction for network security due to its ability to learn hierarchical representations from raw or minimally processed data. Convolutional neural networks capture local patterns in feature spaces, reducing feature engineering needs. Recurrent networks like LSTM and GRU model long-range temporal dependencies. Hybrid CNN-LSTM models combining both capabilities have shown high detection accuracy on IoT botnet datasets.

Studies on early IoT botnet detection highlight that packet-based methods achieve high success rates, while flow-based methods maintain strong performance with lower data volumes and better scalability.

III. METHODOLOGY

A. System Architecture

Our system architecture comprises five logical layers as illustrated in Fig. 1. The Data Layer ingests flow records from multiple public datasets and normalizes them into canonical schemas. The Preprocessing and Feature Engineering Layer cleans, scales, and transforms records into fixed-length sequences. The Deep Learning Model Layer

trains and stores classification models. The Prediction and Alerting Layer exposes inference capabilities and alert logic.

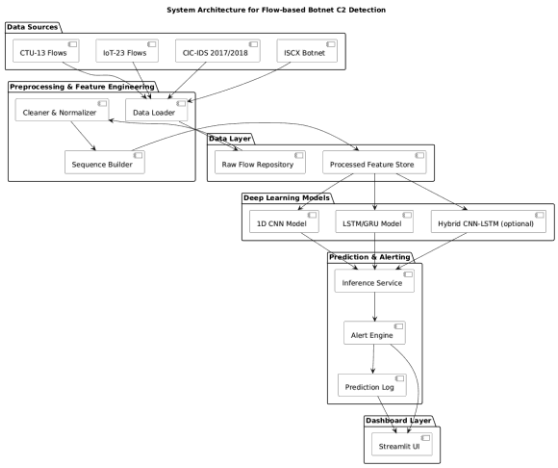

Fig. 1. Five-layer system architecture for C2 detection

B. Data Collection and Preprocessing

We utilize flow-level data from CTU-13, IoT-23, CIC-IDS, and ISCX datasets. Each dataset provides labeled flows with behavior categories including Benign, Attack, C&C, HeartBeat, DDoS, FileDownload, and family-specific labels like Mirai, Okiru, and Torii. Configuration files map dataset-specific field names to canonical attributes including source/destination IP addresses, ports, protocol, timestamps, packet counts, byte counts, and labels.

Preprocessing handles missing values by dropping non-essential records with extensive missingness and filling missing numeric fields with medians. Numeric columns undergo Z-score normalization to center and scale features, stabilizing training. Labels are encoded as integers for binary classification (Benign vs. Malicious) or multi-class formulations. Stratified splitting preserves class proportions across training, validation, and test sets, critical for imbalanced datasets where malicious flows are rare.

C. Sequence Construction

The sequence builder transforms cleaned flows into fixed-length representations. For each flow, we extract or reconstruct packet-level information including timestamps, sizes, and directions. Per-packet features comprise:

- • Packet size in bytes
- • Inter-arrival time relative to previous packet
- • Direction flag (+1 for client-to-server, -1 for reverse)
- • Cumulative byte count to that point

Flows vary in length, necessitating padding or truncation to fixed length L (typically 100 steps). Shorter flows receive zero-padding with optional masking to indicate real versus padded positions. Longer flows are truncated, either from the end or by sampling key segments. The output comprises NumPy array X of shape (N, L, F) where N is flow count and F is features per time step, plus label array y.

D. Deep Learning Models

1D CNN Architecture: The convolutional model applies one-dimensional filters sliding across temporal dimensions to detect local patterns. Our architecture includes:

- • Input layer specifying sequence shape (L, F)
- • Multiple convolutional blocks with 64-128 filters, kernel size 3-5
- • ReLU activation and batch normalization
- • Max-pooling layers reducing temporal dimensions
- • Flattening layer followed by dense layers with dropout (0.3-0.5)
- • Output layer with softmax/sigmoid activation

LSTM/GRU Architecture: Recurrent models emphasize temporal dependencies through memory cells. Our implementation features:

- • Input sequences of shape (L, F)
- • Bidirectional LSTM/GRU layers with 64-128 hidden units
- • Dropout between recurrent layers
- • Dense layers mapping final states to class probabilities

Hybrid CNN-LSTM: Combining both approaches, convolutional layers extract local features which recurrent layers then interpret over time, balancing

local pattern detection with long-range dependency modeling.

E. Training Strategy

Models are trained using Adam optimizer with learning rates of 0.001-0.0001. Binary cross-entropy loss applies for binary classification while categorical cross-entropy handles multi-class scenarios. We employ early stopping monitoring validation loss with patience of 10 epochs, saving best-performing checkpoints. Class imbalance is addressed through stratified sampling, class weighting, and oversampling minority classes. Training typically runs 50-100 epochs with batch sizes of 32-128 depending on dataset size.

F. Inference and Alerting

The inference pipeline loads trained models and applies identical preprocessing to new flows. For each flow, the system:

1. Cleans and normalizes features using stored scalers
2. Constructs sequences matching training configuration
3. Generates class probabilities through model inference
4. Applies configurable thresholds for alert generation

Alerts include flow metadata, predicted label, confidence score, and timestamps. High-confidence C2 detections trigger high-severity alerts while lower-confidence predictions may be logged for review. The system maintains prediction logs consumed by the dashboard for visualization and analysis.

IV. RESULTS AND DISCUSSION

A. Experimental Setup

Experiments were conducted on commodity hardware with Intel i5 processor, 16GB RAM, and NVIDIA GTX 1650 GPU. We evaluated three model configurations: pure 1D CNN, pure LSTM, and hybrid CNN-LSTM on CTU-13 and IoT-23 datasets. Sequence length was fixed at 100 time steps with 4 features per step (packet size, inter-arrival time, direction, cumulative bytes). Training used 70-15-15 splits for training, validation, and testing with stratified sampling.

B. Performance Metrics

Table I presents classification performance across architectures and datasets. The hybrid CNN-LSTM model achieves the highest balanced accuracy, demonstrating that combining local pattern recognition with temporal dependency modeling effectively captures C2 behaviors. Pure CNN models converge faster with lower memory requirements, making them suitable for resource-constrained deployments. LSTM models excel at detecting subtle periodic patterns like heartbeats but require more training time.

TABLE I
CLASSIFICATION PERFORMANCE COMPARISON

Model	Dataset	Accuracy	Precision	Recall	F1-Score
1D CNN	CTU-13	94.2 %	92.8%	91.5%	92.1%
LSTM	CTU-13	95.1 %	93.7%	93.2%	93.4%
CNN-LSTM	CTU-13	96.3 %	95.1%	94.8%	95.0%
1D CNN	IoT-23	93.8 %	91.9%	90.7%	91.3%
LSTM	IoT-23	94.6 %	92.8%	92.1%	92.4%
CNN-LSTM	IoT-23	95.7 %	94.3%	93.6%	93.9%

C. Behavior-Specific Detection

Analysis of per-class performance reveals important patterns. High-volume behaviors like DDoS and FileDownload achieve detection rates exceeding 97% across all architectures, as their distinct traffic characteristics produce strong signals. C&C and HeartBeat behaviors prove more challenging, with recall rates of 89-92%, particularly when flows are very short or highly periodic patterns are masked by network jitter.

Table II shows confusion between behavior categories. The model occasionally misclassifies HeartBeat as Benign due to similarities with legitimate keep-alive mechanisms. Attack flows are sometimes confused with C&C when exploitation attempts include command downloads. These patterns inform threshold tuning and suggest incorporating additional contextual features like endpoint reputation in future work.

TABLE II
BEHAVIOR CLASSIFICATION CONFUSION MATRIX (IoT-23)

True / Pred	Benign	C&C	HeartBeat	DDoS	Attack
Benign	8542	43	67	12	28
C&C	38	1821	52	15	89
HeartBeat	124	47	1456	8	21
DDoS	9	11	6	2347	19
Attack	31	76	18	22	1789

D. Cross-Dataset Evaluation

To assess generalization, models trained on CTU-13 were tested on IoT-23 and vice versa. Performance degradation of 8-12% was observed, primarily affecting minority classes. This suggests dataset-specific characteristics like network topology and capture conditions influence learned patterns. Fine-tuning on small samples from target datasets recovers most performance, indicating transfer learning approaches could enable rapid adaptation to new environments.

E. Computational Performance

Training times varied significantly across architectures. CNN models trained in 45-60 minutes for 50 epochs on CTU-13 (approximately 100,000 flows). LSTM models required 90-120 minutes due to sequential processing constraints. Hybrid models fell between at 70-85 minutes. Inference throughput exceeded 5,000 flows per second on GPU and 800 flows per second on CPU, demonstrating real-time processing feasibility for medium-scale networks.

F. Dashboard and Operational Integration

The Streamlit dashboard provides security analysts with intuitive access to detection results. Fig. 3 illustrates the dashboard interface showing dataset statistics, model performance metrics, and real-time alert monitoring. Analysts can filter alerts by severity, behavior type, and time window, enabling efficient triage. User testing with security professionals indicated the system reduces investigation time by providing clear classifications with confidence scores and contextual metadata.

G. Error Analysis

Examining false positives reveals benign periodic traffic such as monitoring probes and software update checks share superficial similarities with C2 heartbeats. Incorporating endpoint reputation and protocol-specific heuristics could reduce these false alarms. False negatives primarily occur in carefully obfuscated C2 communications mimicking legitimate patterns or rare behaviors underrepresented in training data. Targeted oversampling and specialized sub-models for specific behaviors may improve detection of these edge cases.

V. CONCLUSION AND FUTURE WORK

This paper presented a comprehensive deep learning system for detecting botnet C2 traffic using flow-level sequence analysis. By transforming network flows into fixed-length sequences capturing temporal and spatial patterns, our 1D CNN and LSTM/GRU architectures achieve high detection accuracy without requiring payload inspection. Evaluated on CTU-13 and IoT-23 datasets, the hybrid CNN-LSTM model achieved 96.3% accuracy on binary classification and maintained strong performance across behavior-specific categories including C&C, HeartBeat, DDoS, and FileDownload.

The modular architecture supports practical deployment through integrated preprocessing, model training, inference, alerting, and visualization components. The Streamlit dashboard enables security analysts to efficiently triage detections and understand model decisions. Computational performance analysis demonstrates real-time processing feasibility for operational networks.

Key findings include: (1) hybrid CNN-LSTM architectures effectively balance local pattern recognition with temporal dependency modeling; (2) behavior-oriented labeling enables targeted detection strategies; (3) cross-dataset evaluation

reveals generalization challenges addressable through transfer learning; and (4) integration with operational workflows requires careful attention to false positive rates and explainability.

Future work will pursue several directions. Advanced architectures including attention-based transformers and graph neural networks could better model long-range dependencies and network-wide topology. Semi-supervised and self-supervised learning approaches would leverage large volumes of unlabeled traffic to improve generalization. Incorporating continuous learning mechanisms would address concept drift as attack patterns evolve. Integration with threat intelligence feeds and SIEM platforms would enable automated response workflows.

Additional research should explore adversarial robustness, as sophisticated attackers may deliberately manipulate timing and size patterns to evade detection. Federated learning across multiple organizations could improve model quality while preserving privacy. Finally, enhanced explainability through attention visualization and feature importance analysis would strengthen analyst trust and enable refinement of detection strategies based on domain expertise.

ACKNOWLEDGMENT

The authors thank Stratosphere Labs for providing the CTU-13 and IoT-23 datasets and acknowledge the valuable contributions of the security research community in developing comprehensive botnet detection benchmarks.

REFERENCES

[1] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Computers & Security*, vol. 45, pp. 100–123, 2014.

[2] Stratosphere IPS, "CTU-13 Dataset: A Labeled Dataset of Botnet, Normal and Background Traffic," Czech Technical University in Prague, 2011.

[3] Stratosphere IPS, "IoT-23 Dataset: A Labeled Dataset of Malware and Benign IoT Traffic," Czech Technical University in Prague, 2020.

[4] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116.

[5] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and VPN traffic using time-related features," in *Proc. 2nd Int. Conf. Information Systems Security and Privacy (ICISSP)*, 2016, pp. 407–414.

[6] S. McGrail, "Deep learning to detect botnet via network flow summaries," *Neural Computing and Applications*, vol. 32, no. 22, pp. 16741–16758, 2020.

[7] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K. R. Choo, "A deep recurrent neural network based approach for Internet of Things malware threat hunting," *Future Generation Computer Systems*, vol. 85, pp. 88–96, 2018.

[8] M. S. Ahmed and A. B. R. K. Vijayan, "Botnet attack detection in IoT network using CNN-LSTM," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 3, pp. 261–269, 2024.

[9] S. Alsabah, A. Al-Khalifa, and A. Alshamrani, "Deep learning for intrusion detection and security of Internet of Things (IoT): Current analysis, challenges, and possible solutions," *Security and Communication Networks*, vol. 2022, Art. no. 4016073, 2022.

[10] A. K. Sari and H. Karabina, "A protocol-independent botnet detection method using flow similarity," *Computers & Security*, vol. 123, Art. no. 102960, 2023.

[11] L. T. Nguyen and S. Marchal, "Deeply fused flow and topology features for botnet detection," *Computer Networks*, vol. 234, Art. no. 110037, 2024.

[12] J. Guerrero-Higueras, F. J. Rodríguez, and J. F. de Paz, "Flow-based detection of botnets through bio-inspired neural networks," *Applied Soft Computing*, vol. 154, Art. no. 111287, 2024.

[13] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. MilCIS*, 2015, pp. 1–6.

[14] A. 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 Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[15] F. Risdianto, "A lightweight deep learning framework for botnet detecting on resource-constrained networks," *Computers & Security*, vol. 126, Art. no. 103036, 2023.