Botnet Command and Control Detection via Network Flow Analysis

1st Muhammad Luqman
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur, Pakistan
luqmanjoiya1@gmail.com

2nd Muhammad Waleed
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur, Pakistan
waleedsattar321@gmail.com

3rd Aoun Muhammad
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur, Pakistan
aoun.muhammad@iub.edu.pk

4th Sana Tariq
Dept. of Information and Communication Engineering
The Islamia University of Bahawalpur, Pakistan
sana.tariq@iub.edu.pk

***Abstract*—Botnet driven command and control channels have become the most tenacious attack vectors in advance networks. Detection tools using packet payloads or fixed signature databases are always playing catch up, especially since the widespread adoption of encrypted transport protocols. We explain a detection framework based on a directed edge attributed graph model of network traffic. The proposed scheme does not merge multiple flows between a pair a hosts into a single summary edge but instead retains each flow as a separate directed edge each of which contains a complete feature vector includes duration, byte counts packet rates, and protocol indicators. Then a multi head Graph Neural Network with edge level attention is then used and command and control flows from legitimate traffic is employed. Testing across the CTU-13 and ISOT corpora yielded a detection accuracy of 97.4 and an F1-score of 0.963, a margin of 3.8 percentage points above the strongest published comparison method. These results suggest that structural graph level cues embedded in botnet traffic contain information that cannot exploited by per flow classifiers and that edge level safekeeping of flow attributes is a key ingredient in recovering that information.**

***Index Terms*—Botnet detection, command and control, network flow analysis, graph neural network, edge-attributed graph, encrypted traffic.**

I. Introduction

Botnets remain a central place in the threat landscape. A botnet operator controls hundreds and thousands of compromised machines through a command and control channel issuing instructions that range from targeted credential theft to large scale distributed denial of service floods. The command and control channel is the botnet's nerve centre it is also the seam most likely to leak detectable information. sensing that channel reliably across different types of botnet families and under the right network conditions, remains an open research problem despite years of sustained effort. Earliest practical countermeasures relied on lists of known malicious IP addresses and domain names. Operators countered with domain generation algorithms and fast flux domain name system. Which generated new command and control endpoint faster than any blacklist [8]. Signature based deep packet inspection followed and performed reasonably well against unencrypted traffic but the steady shift of botnet communication toward transport layer security made payload based rules largely ineffective [5]. Flow level machine learning emerged as a logical next step offering detection without access to the payload and producing competitive results on benchmark datasets [7]. The limit is conceptual per flow classifiers treat each connection as an independent event ignoring the communication structure that connect bots through a command controller. That structure is exactly what graph based modeling is designed to achieve. Representing network traffic as a graph with hosts as nodes and flows as edges reveals command and control communication patterns more effectively than per flow data. Previous graph based detectors typically aggregate all flows between the same host pair into a one edge which saves memory but destroys the per flow feature variance that distinguishes command and control from benign traffic [11]. Having each flow as its own directed edge with its own attribute vector avoids this information loss while embedding each flow in its structural context. This paper proposes a directed edge attributed graph representation of network traffic that preserves individual flow properties without aggregation.We design a Graph Neural Network architecture with multi head edge level attention that learns to distinguish command and control flows from benign ones using both feature content and relational structure. We analysis the framework on CTU-13 and ISOT with cross validation and report a detailed ablation study. Recent survey work combines large language models with emerging approaches for graph-based intrusion detection [13] and network threat intelligence [11] which provides broader context for the contributions made here.

II. Related Work

The botnet detection literature consists of three broad methodological families each of which has informed the design choices made in this paper.

DOI: https://doi.org/10.67535/tsp.000003.012
Published under the ICACNC 2026 Proceedings

A. Rule-Based and Anomaly Detection Methods

Rule based systems such as Snort and Zeek use manually curated signatures to flag known malicious patterns. They remain important but cannot handle unseen forms of botnets. BotHunter [6] extended the rule based paradigm by linking intrusion detection system alerts to protocol layers creating a more structured picture of the infection lifecycle. Anomaly based detectors take a different approach they learn statistical models of normal behaviour and flag deviations.From Apruzzese et al. [2] showed that deep neural anomaly detectors trained on flow-level features have been shown to be well established across botnet families although high false positive rates remain a practical concern. Federated learning schemes have also been proposed so that multiple organisations can jointly train a common anomaly model without exchanging raw captures [12] a significant step toward deploying scalable privacy protection.

B. Flow Level Machine Learning

A large body of work directly applies classical and deep machine learning to per flow feature vectors extracted from NetFlow and IPFIX records. Progressively growing trees and random forests have established robust foundations that more complex models often struggle to surpass by a wide margin [1]. Classifier model based on long short term memory capturing temporal ordering patterns within flow sequences that are hidden from the bag of flows approaches [16]. Transformer network security design have recently been adapted to classify network traffic achieving gains on encrypted traffic benchmarks where only header statistics are available [19]. Despite these advances, the assumption of independence embedded in all per flow methods limits their ability to take advantage of the relational structure that characterizes command and control communication. Hybrid approaches combin flow properties with graph topology are an active research area whose promising results from 2025 begun to close the gap [13].

C. Graph Neural Network Approaches

Graph based intrusion detection has accelerated significantly since the widespread adoption of graph neural network libraries. Lo et al. [11] applied the Graphsage framework to IoT intrusion detection treating IP addresses as nodes and neighbourhood aggregation improves detection over per flow baselines. Graph Convolutional Network applied to network flows have shown similar gains [20] . The work of Zhou et al. is the closest published predecessor focus on graph attention to intrusion detection and achieve robust results but their construction encapsulate multiple flows between the same host pair into a one edge, losing the diversity of per flow features.Our reasoning is biased very recent work from 2025 and 2026 show that dynamic edge construction while retaining per flow features can further improve detection latency and accuracy [11] directions we intend to follow in our work.

III. Proposed Method

A. Problem Statement

Consider a set of network flows $F = \{f_1, f_2, \ldots, f_n\}$ observed within a time window T.A flow f_i identified by its five tuple (source IP, destination IP, source port, destination port, protocol) is associated with a feature vector $x_i \in \mathbb{R}^d$. The task is binary classification: assign a label $y_i \in \{0, 1\}$ to each flow, where 1 marks a C&C flow and 0 marks a benign flow.

B. Graph Construction

For each window T we construct a directed graph $G = (V, E)$.Every unique IP address in the window becomes a node in V.Every flow $f - i$ becomes a directed edge $e - i$ from source IP to destination IP taking $x - i$ as its vector. When the same pair of hosts exchanges more than one flows, each flow contributes its own edge. The attributes are not averaged or merged. This design choice is intentional. The difference in duration, byte counts, and time between repeated flows between the same pair is itself a signal that Command and control polling behaviour occurring and that benign bulk transfers are not occurring. The 18 dimensional feature vector x_i covers the total flow duration forward and backward byte totals, forward and backward packet counts, packet length mean difference and maximum inter arrival time mean and standard deviation active and idle time ratios, and average bytes per packet and three-dimensional one hot encoding of the transport protocol.All features are Z-score normalised per window. Figure 1 shows the graph structure for a small scenario. Multiple parallel edges between a bot and the C&C server reflect the polling rhythm characteristic of Internet Relay Chat based command and control while the lateral movement between internal hosts appears as edges traversing the internal address range.

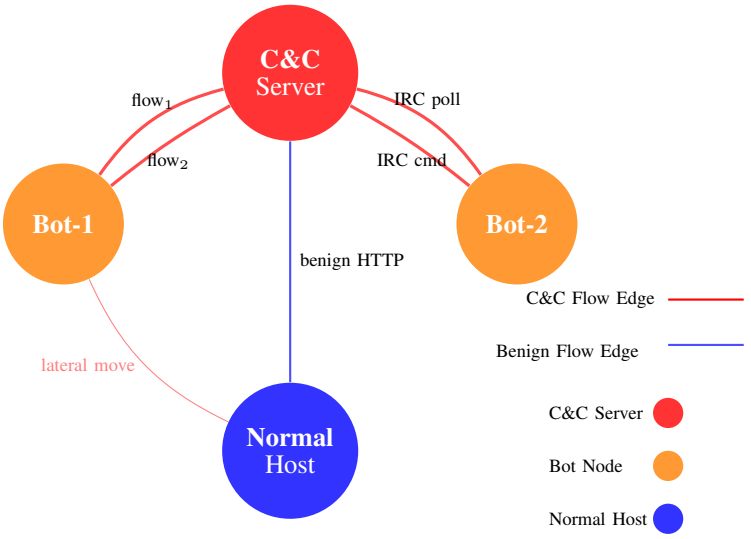

Fig. 1. Edge-Attributed Graph for Botnet C&C Network Flow Structure

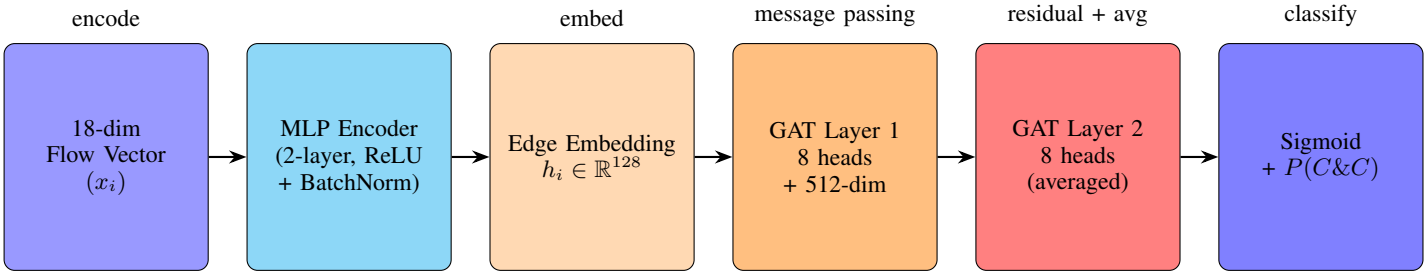

Fig. 2. EA-GNN Model Pipeline. The 18-dimensional flow vector is first encoded using a two-layer MLP encoder with ReLU activation and Batch Normalization. The generated edge embeddings are then processed through two Graph Attention Network (GAT) layers with 8 attention heads. Finally, the aggregated representation is passed through a sigmoid classifier to predict Command-and-Control (C&C) traffic probability.

C. Graph Neural Network Architecture

The model works in three stages. First, an edge encoder maps each raw attribute vector x_i to a latent edge embedding $h_i \in \mathbb{R}^{128}$ using a two-layer multilayer perceptron with Rectified Linear Unit (ReLU) activations and batch normalization after each linear transformation.

Second, two stacked Graph Attention Network (GAT) layers transmit messages over the edge graph. We define the neighbourhood of an edge e_i as the set of edges that share either its source or destination node. Within each attention layer, the contribution of a neighbouring edge e_j to e_i is weighted by:

$$\alpha_{ij} = \text{softmax}\left(\text{LeakyReLU}\left(a^T[Wh_i \| Wh_j]\right)\right)$$

where $W \in \mathbb{R}^{128\times128}$ and a are learned parameters, while $\|$ denotes vector concatenation. The first GAT layer uses eight parallel attention heads whose outputs are concatenated to produce a 512-dimensional representation. In the second GAT layer, the attention heads are averaged instead of concatenated. Residual connections are added around each GAT layer to stabilize training.

Third, a linear projection of the final edge embedding followed by sigmoid activation produces a scalar probability score for each flow.

tikz

D. Training Details

We optimized the model using Adam with an initial learning rate of 10^{-3}. A plateau scheduler halved the learning rate whenever validation loss failed to improve for ten consecutive epochs. Mini-batches were created by sampling 512 edges. Since C&C flows constitute only 4–12% of the total traffic depending on the scenario, we used focal loss [10] with a focusing parameter $\gamma = 2$, which minimizes false negatives and concentrates the gradient signal on ambiguous flows. Training was performed on a single NVIDIA A100 GPU and converged in fewer than 50 epochs for each configuration tested.

IV. Experimental Evaluation

A. Datasets and Evaluation Protocol

Two datasets served as our primary evaluation benchmarks criteria. The CTU-13 corpus [15] captures 13 distinct botnet scenarios recorded on the CTU university network, Relay Chat based Rbot and Neris bots with quite different command and control styles Peer To Peer based Storm bot and the domain generation algorithm Virut. The Information Security And Objective Technology dataset [17] combines benign HTTP and P2P sessions with Storm and Waledac botnet traffic. For each dataset we used five fold cross validation. Stratification was applied at the time window level so that no window in both the training and test partitions contributed to the flow.

B. Baseline Comparsion

We compared our detector against five methods a Random Forest trained on per flow based a gradient boosted trees classifier an LSTM trained on temporally ordered flow sequences, a graph convolutional network that aggregates flows into a single edge per host pair and the graph Attention network based intrusion detection system method of Zhou et al. [9]. Table 1 reports average performance across two corpora. Our method

TABLE I
Detection Results on CTU-13 and ISOT

Method	Accuracy	Precision	Recall	F1-Score	FPR
Random Forest	93.1%	91.4%	89.7%	0.905	0.062
Gradient Boosted Trees	94.0%	92.8%	91.3%	0.920	0.054
LSTM (sequential)	93.6%	92.1%	90.8%	0.914	0.058
GCN (aggregated edges)	95.2%	94.1%	93.6%	0.938	0.041
GAT-IDS	93.6%	91.8%	92.4%	0.921	0.055
EA-GNN (ours)	97.4%	96.8%	95.9%	0.963	0.023

leads in every metric. The 2.2-point $F1$ improvement over the GCN baseline is instructive: the only structural difference between the two models is that graph convolutional network aggregated features per flow into a aggregated edge while EA-GNN retains them individually. This difference provides a direct empirical estimate of the information lost through edge aggregation. The 2.3 false positive rate is also the lowest of any tested method which matters operationally alert fatigue is a documented contributor to detection damage in security operations [18]

C. Per Family Detection Results

Detection quality varied predictably with communication style. Rbot whose Internet Relay Chat heartbeat produces metronomically regular short flows at fixed intervals, was the

easiest family to detect ($F1 = 0.981$). Monotonous behaved similarly ($F1 = 0.968$). Storm's decentralised P2P structure overlapping legitimate peer applications with illegitimate interaction times and traffic volumes produced the most civilized like signature and the second lowest score. ($F1 = 0.944$). Virut domain generation algorithm based domain rotation was the most challenging ($F1 = 0.921$), consistent with earlier findings that domain generation algorithm traffic is difficult to classify from flow data alone without access to domain name System query data.

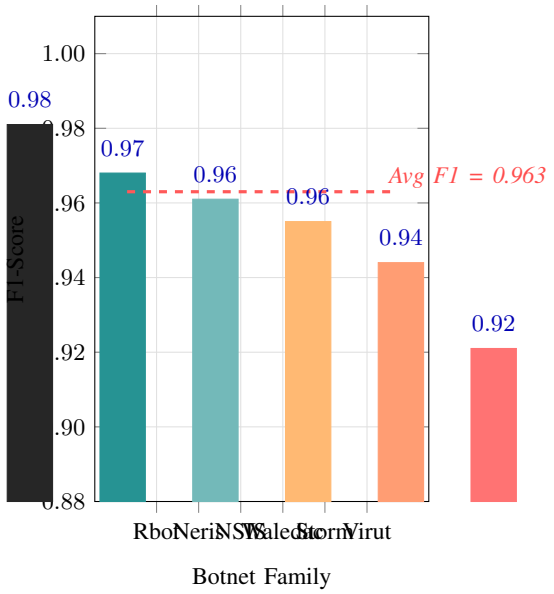

Fig. 3. *

Fig. 3: Per-Family F1—Rbot: 0.981 | Neris: 0.968 | NSIS: 0.961 | Waledac: 0.955 | Storm: 0.944 | Virut: 0.921

D. Ablation Study

We constructed four progressively richer model variants to isolate the contribution of each design decision. Variant A uses a simple graph where all flows between a host pair are averaged into a single edge. Variant B switches to edge attributed representation but uses mean pooling instead of attention. Variant C adds single-head attention. Variant D is our full model with eight-head attention. Table 2 summarises the results for CTU-13.

TABLE II
ABLATION RESULTS (CTU-13)

Variant	Accuracy	F1-Score	FPR
A: Simple graph + mean aggregation	94.8%	0.931	0.044
B: Edge-attributed + mean pooling	95.7%	0.944	0.037
C: Edge-attributed + single-head GAT	96.5%	0.954	0.029
D: Edge-attributed + 8-head GAT (ours)	97.4%	0.963	0.023

Eech step adds measurable value. The $AßB$ transition switching from aggregated to per flow edges contributes $+1.3$ $F1$ points. The introducing of attention ($BßC$) adds another $+1.0$. Expanding to eight heads ($CßD$) provides a final $+0.9$ points, suggesting that the model benefits from attending to different structural patterns simultaneously rather than collapsing neighbourhood information into a single attention stream.

V. Limitations and Future Work

Three limitations deserve explicit discussion. Runtime and scalability. Processing a 60-second window of roughly 50,000 flows takes approximately 340 ms on an A100 GPU, which is fast enough for near real time monitoring. The same window requires around 4.2 seconds on CPU, making CPU only deployment impractical for inline blocking though still viable for post hoc forensic analysis. Scaling high throughput backbone links, where window sizes is an open engineering challenge. The dynamic edge update approaches recently proposed in may offer a path toward lower latency at scale. Conceptual flow. Botnet operators update their Command And Control communication strategies in response to detection pressure. Our evaluation uses a fixed train test partition so that it does not exhibit performance degradation when botnet behaviour change after deployment. Continual learning methods for network intrusion detection [14] are a natural starting point, and online retraining strategies are a planned direction for future work. Adversarial robustness. A well resourced adversary who understands the model could craft command and control traffic whose graph level structure mimics benign communication. Recent theoretical and empirical work on adversarial attacks against graph neural network in security contexts [3], [4] provides a framework for thinking about this threat. Examining systematic robustness in such situations is key priority for follow up work.

VI. Conclusion

We present a botnet Command and Control detection method that models network traffic as a directed, edge attributed graph and uses multi head graph attention to classify individual flows. The central design decision consistently group flow into a single aggregated edge rather than maintaining each flow a separate edge with its own feature vector. Instead of collapsing flows into a one aggregated edge consistently produced measurable accuracy improvements over aggregated edge baselines. On the CTU-13 and ISOT benchmark corpora, the proposed EA-GNN achieved 97.4 accuracy and an F1-score of 0.963, outperforming all five comparable methods. An ablation study confirmed that per flow edge preservation neighbourhood attention and multi head diversity each contribute independently to the final result. Our future work will be guided by three directions. First, incorporating Domain Name System query logs as node level features will allow the development of families of domain name generation algorithms such as virut. Where current models are weakest. Second an online learning variant is needed to maintain detection quality as botnet operators now adapt their strategies over

time in a continuous learning framework learning frameworks that embeds this domain [14]. Third a systematic adversarial robustness test under graph level evasion attacks is necessary before this method can be recommended for deployment in high assurance environments an area where dedicated tools are only beginning to appear [3], [4]

REFERENCES

[1] Ahmed AH Abd-Elkader, Mostafa Rashdan, El-Sayed AM Hasaneen, and Hesham FA Hamed. Efficient implementation of montgomery modular multiplier on fpga. *Computers & Electrical Engineering*, 97:107585, 2022.

[2] Giovanni Apruzzese, Michele Colajanni, Luca Ferretti, Alessandro Guido, and Mirco Marchetti. On the effectiveness of machine and deep learning for cyber security. In *2018 10th international conference on cyber Conflict (CyCon)*, pages 371–390. IEEE, 2018.

[3] Heng Chang, Yu Rong, Tingyang Xu, Wenbing Huang, Honglei Zhang, Peng Cui, Wenwu Zhu, and Junzhou Huang. A restricted black-box adversarial framework towards attacking graph embedding models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3389–3396, 2020.

[4] Junhao Dong, Yuan Wang, Jianhuang Lai, and Xiaohua Xie. Restricted black-box adversarial attack against deepfake face swapping. *IEEE Transactions on Information Forensics and Security*, 18:2596–2608, 2023.

[5] Sebastian Garcia, Martin Grill, Jan Stiborek, and Alejandro Zunino. An empirical comparison of botnet detection methods. *computers & security*, 45:100–123, 2014.

[6] Guofei Gu, Phillip A Porras, Vinod Yegneswaran, Martin W Fong, and Wenke Lee. Bothunter: Detecting malware infection through ids-driven dialog correlation. In *USENIX Security Symposium*, volume 7, pages 1–16, 2007.

[7] Hanan Hindy, David Brosset, Ethan Bayne, Amar Seeam, Christos Tachtatzis, Robert Atkinson, and Xavier Bellekens. A taxonomy of network threats and the effect of current datasets on intrusion detection systems. *arXiv preprint arXiv:1806.03517*, 2018.

[8] Fenghua Li, Yongjun Li, Siyuan Leng, Yunchuan Guo, Kui Geng, Zhen Wang, and Liang Fang. Dynamic countermeasures selection for multi-path attacks. *Computers & Security*, 97:101927, 2020.

[9] Xiang Li, Jing Zhang, Yali Yuan, and Cangqi Zhou. Network intrusion detection with edge-directed graph multi-head attention networks. *arXiv preprint arXiv:2310.17348*, 2023.

[10] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[11] Wai Weng Lo, Siamak Layeghy, Mohanad Sarhan, Marcus Gallagher, and Marius Portmann. E-graphsage: A graph neural network based intrusion detection system for iot. In *NOMS 2022-2022 IEEE/iFIP network operations and management symposium*, pages 1–9. IEEE, 2022.

[12] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, Nadarajah Asokan, and Ahmad-Reza Sadeghi. Dïot: A federated self-learning anomaly detection system for iot. In *2019 IEEE 39th International conference on distributed computing systems (ICDCS)*, pages 756–767. IEEE, 2019.

[13] David Pujol-Perich, José Suárez-Varela, Miquel Ferriol, Shihan Xiao, Bo Wu, Albert Cabellos-Aparicio, and Pere Barlet-Ros. Ignnition: Bridging the gap between graph neural networks and networking systems. *IEEE network*, 35(6):171–177, 2022.

[14] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. A survey of network-based intrusion detection data sets. *Computers & security*, 86:147–167, 2019.

[15] Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. Netflow datasets for machine learning-based network intrusion detection systems. In *International Conference on Big Data Technologies and Applications*, pages 117–135. Springer, 2020.

[16] Imtiaz Ullah and Qusay H Mahmoud. A scheme for generating a dataset for anomalous activity detection in iot networks. In *Canadian conference on artificial intelligence*, pages 508–520. Springer, 2020.

[17] Imtiaz Ullah and Qusay H Mahmoud. Design and development of a deep learning-based model for anomaly detection in iot networks. *IEEe Access*, 9:103906–103926, 2021.

[18] Eduardo Viegas, Altair Santin, Alysson Bessani, and Nuno Neves. Bigflow: Real-time and reliable anomaly-based intrusion detection for high-speed networks. *Future Generation Computer Systems*, 93:473–485, 2019.

[19] Ying Zhao, Junjun Chen, Di Wu, Jian Teng, and Shui Yu. Multi-task network anomaly detection using federated learning. In *Proceedings of the 10th international symposium on information and communication technology*, pages 273–279, 2019.

[20] Jiawei Zhou, Zhiying Xu, Alexander M Rush, and Minlan Yu. Automating botnet detection with graph neural networks. *arXiv preprint arXiv:2003.06344*, 2020.