The Rise of Relational AI: A Comprehensive Survey of Graph Neural Networks in Modern Cybersecurity Defenses

Abhale B. A[1], Akshay Hole[2], Pranav Bankar[3], Shraddha Ghaytadkar[4], Diya Jejurkar[5], Mr.Pathare G N[6]
SND College of Engineering & Research Center Savitribai Phule Pune University

***Abstract*—This comprehensive survey examines and consolidates research on the utilization of Graph Neural Networks (GNNs) within the cybersecurity domain. The study traces the progression from conventional feature-extraction-based intrusion detection methodologies to contemporary structure-cognizant defensive frameworks that exploit the fundamental relational characteristics inherent in cybersecurity data. The research systematically investigates core GNN architectures, sophisticated models designed for temporal and heterogeneous datasets, and their deployment in essential security applications including lateral movement identification and premptive threat reconnaissance. Particular emphasis is accorded to the necessity of Explainable Artificial Intelligence (XAI) methodologies to reconcile the complexity of model outputs with practical security decision-making processes. The authors conduct a rigorous examination of enduring obstacles within this field, encompassing computational scalability, data integrity, real-time analytical capabilities, and resilience against adversarial attacks. Through the integration of contemporary research developments, this survey elucidates prevailing scholarly trajectories, delineates significant lacunae in existing literature, and delineates prospective avenues for advancing artificial intelligence applications in cybersecurity research and practice.**

***Index Terms*—Graph Neural Networks, Security Operation Centre, Proactive systems, Cyber Security, Temporal Heterogeneous Graph Neural Network (THGNN), Explainable AI (XAI), survey**

I. INTODUCTION

The Paradigm Shift to Graph-Based Security Analytics
The landscape of cybersecurity is defined by a continuous arms race between attackers and defenders. As digital infrastructures grow in complexity and interconnectivity, so too do the sophistication and stealth of cyber threats. For decades, the primary line of defense has relied on systems designed to identify known threats or simple deviations from a static baseline. However, the rise of advanced, multi-stage attacks, particularly Advanced Persistent Threats (APTs), has exposed the fundamental limitations of these legacy approaches. This has catalyzed a paradigm shift in security analytics, moving away from the analysis of isolated events towards a holistic, relational understanding of system behavior—a shift for which Graph Neural Networks (GNNs) have emerged as the principal enabling technology.

1.1 The Inadequacy of Legacy Systems

Traditional security infrastructures, primarily composed of Intrusion Detection Systems (IDS) and Security Information and Event Management (SIEM) platforms, form the bedrock of most enterprise security operations. These systems were designed for an era of more predictable and less coordinated threats. Their operational logic is rooted in two main methodologies: signature-based detection and rule-based correlation.

Signature-based mechanisms, the foundation of antivirus software and many IDSs, function by matching observed data such as network traffic content or file hashes against a vast database of signatures corresponding to known malware and attack patterns. While effective against previously identified threats, this approach is inherently reactive. It is fundamentally incapable of detecting novel, zero-day exploits, which by definition have no existing signature. An attacker can often bypass these defenses with minor modifications to their tools, such as recompiling malware, which is sufficient to alter the signature and render it unrecognizable. Rule-based

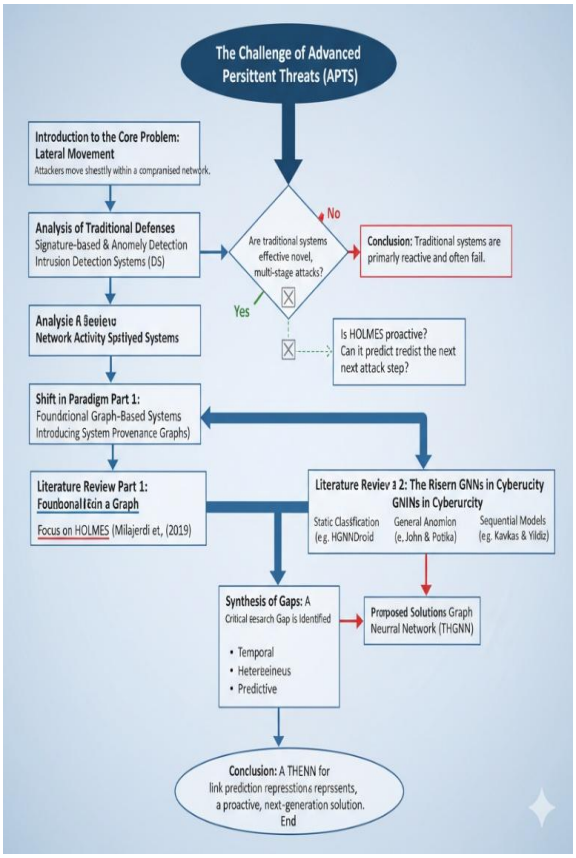
systems, common in SIEMs, attempt to overcome this by correlating events from multiple sources (e.g., firewalls, servers, applications) based on predefined logic. For instance, a rule might flag a series of failed login attempts followed by a successful one from a foreign IP address. While more flexible than static signatures, these systems struggle with the complexity and subtlety of modern attacks like APTs. APTs are characterized by their "low-and-slow" nature, unfolding over long periods and often using legitimate system tools and credentials to blend in with normal activity. The static rules required to detect such nuanced behaviors are difficult to craft, prone to generating high volumes of false positives, and can be easily bypassed by attackers who understand the logic. A deeper, more fundamental limitation of both these approaches is what can be termed the "flat data" problem. Conventional machine learning models applied in cybersecurity often process security logs as a collection of independent events or flat feature vectors. This methodology discards the rich relational context and structural dependencies that are ubiquitous in cyberattack scenarios. An intrusion is rarely a single, isolated event; it is a sequence of suspicious interactions between entities like hosts, users, and processes. By treating each log entry in isolation, these models lose the very information that connects the dots of a coordinated campaign, a vulnerability that attackers implicitly exploit. This failure of legacy systems is not merely technical; it has profound operational and economic consequences. Security Operations Centers (SOCs) are inundated with a deluge of log data from heterogeneous sources, making manual correlation an impossible task.

1.2 The Graph-Based Security Paradigm

In response to the limitations of event-centric analysis, a new paradigm has emerged, centered on the representation of security data as graphs. This approach recognizes that the state and behavior of any computer system or network are inherently relational. A graph provides a universal, abstract, and highly expressive structure for modeling these relationships. In this paradigm, system entities—such as users, devices, IP addresses, processes, and files—are represented as nodes. The interactions between them—such as network connections, authentication events, file accesses, or system calls—are represented as edges. This transformation of raw, flat log data into a graph structure preserves the critical relational information that traditional methods discard. While an adversary can easily change the features of a single event (e.g., the payload of a packet), they cannot execute a multi-stage attack without creating new interactions and altering the structure of the system graph. This shift in data representation from "what" (a specific signature or log entry) to "how" (the pattern of interactions) is a strategic move to a more durable and resilient plane of detection.

LITERATURE SURVEY

Sr no	Paper title	Author name	year
1	HOLMES: Real-time APT Detection through Correlation of Suspicious Information Flows	Sadegh M. Milajerdi, Rigel Gjomemo, Birhanu Eshete,	2019
2	Graph Neural Networks for Intrusion Detection: A Survey	TRISTAN BILOT	2023
3	Graph Neural Networks for Intrusion Detection: A Survey	Xingyu Liu, , NOUR EL MADHOUN	2024
4	Diverse GNN Encoder-Decoder for Graph Anomaly Detection	Kenneth John &Katerina Potika	2025
5	Enhancing IoMT Security with Deep Learning Based Approach for Medical IoT Threat Detection	Kavkas & Yildiz,	2025



II. FOUNDATIONAL MODELS: APPLYING GRAPH NEURAL NETWORKS TO INTRUSION DETECTION

The application of GNNs to cybersecurity begins with a fundamental re-imagining of security data. Instead of viewing logs as a temporal sequence of discrete events, the graph-based paradigm sees them as evidence of an underlying structure of interactions. This section details the principles of this transformation, reviews the foundational GNN architectures that form the building blocks of modern systems, and explains how different security problems are mapped onto formal graph-based learning tasks.

2.1 Principles of Graph Representation Learning for Security

Before any learning can occur, raw security data must be transformed into a graph. This process, far from being a simple data formatting step, is a critical form of implicit feature engineering that embeds domain knowledge into the very structure the model will analyze.

2.1.1 Graph Construction

The initial and most crucial step in any GNN-based security pipeline is graph construction. This involves defining what constitutes a node and an edge from raw, unstructured, or semi-structured data sources. Common data sources include network traffic logs (e.g., NetFlow, Zeek logs), system provenance data from host-based sensors (e.g., Linux auditd, Windows ETW), and authentication logs from services like Active Directory.

The choice of node and edge definitions is a design decision that fundamentally determines the semantic resolution of the analysis. For example, in network intrusion detection, nodes could be defined simply as IP addresses, with edges representing any communication between them. A more granular approach, however, might define nodes as unique IP-port combinations, allowing the model to distinguish between different services running on the same host. Edges could represent individual network flows, annotated with features like protocol, duration, and byte count. In a host-based context, nodes might represent processes, files, and users, while edges could represent system calls like fork, exec, read, or write.

This construction phase is where a security architect's expertise is paramount. An improperly designed graph can obscure the very patterns the GNN is meant to find. For instance, a graph that only models network connections would be blind to an attack that occurs entirely on a single host through process injection. Conversely, a graph that is too dense or includes irrelevant interactions can overwhelm the GNN with noise, degrading its performance. Therefore, constructing an appropriate graph structure is a non-trivial challenge that directly constrains and guides the model's learning potential.

2.2.2 GraphSAGE

GraphSAGE (Graph SAmple and aggreGatE) addresses this by introducing an *inductive* learning framework. Instead of learning a unique embedding for each node, GraphSAGE learns an aggregation function. To compute the embedding for a node, it samples a fixed number of its neighbors and applies the learned function to aggregate their features. Because it learns a function rather than static embeddings, GraphSAGE can generalize and generate embeddings for entirely unseen nodes, a critical capability for practical deployment in dynamic

networks. Frameworks like E-GraphSAGE have extended this approach to handle edge features, making it suitable for classifying malicious network flows in NIDS. The distinction between transductive and inductive learning is not merely an academic detail; it is arguably the most important factor determining a model's viability for real-time deployment, severely narrowing the field of applicable research to inductive architectures.

2.2.3 Graph Attention Networks (GATs)

Both GCN and GraphSAGE treat all neighbors of a node as equally important during the aggregation step. Graph Attention Networks introduce a more nuanced approach by incorporating an attention mechanism. A GAT learns to assign different *attention weights* to different neighbors, allowing the model to dynamically decide which nodes in the neighborhood are most relevant to the current task and focus on their information during aggregation.[4]This can be particularly useful in security, where an interaction with one specific suspicious host may be far more indicative of an attack than interactions with dozens of benign ones. The E-ResGAT framework is a notable example that applies this principle, integrating residual learning with attention for intrusion detection.

2.3 Task Formulation: Node, Edge, and Graph Classification

The versatility of GNNs allows different cybersecurity problems to be framed as distinct graph learning tasks. The choice of formulation depends on the specific goal of the detection system.

- Node Classification: This is one of the most common formulations. The goal is to assign a label (e.g., "benign" or "malicious") to each node in the graph. This is directly applicable to problems like identifying compromised hosts in a network, detecting malicious user accounts, or flagging malware processes based on their system call patterns.
- Edge Classification and Link Prediction: In this formulation, the task is to classify edges. This is used to identify malicious connections or interactions between entities. A closely related task is link prediction, which involves predicting whether an edge is likely to exist (or appear in the future) between two nodes. In cybersecurity, this is powerfully applied to detect anomalous connections, such as in lateral movement detection, where the model predicts if a new connection between two hosts is legitimate or part of an attack.
- Graph Classification: Here, the goal is to classify an entire graph or, more commonly, a subgraph. This is useful for identifying coordinated, multi-step attack patterns. For example, a sequence of activities captured in a provenance graph over a specific time window can be treated as a single graph instance to be classified as "attack campaign" or "benign operations".

III. CAPTURING REALITY: TEMPORAL AND HETEROGENEOUS GNNS FOR ADVANCED THREAT DETECTION

3.1 Modeling Temporal Dynamics

A static graph is a snapshot in time, incapable of capturing the sequence, duration, and evolution of events—all of which are critical for distinguishing benign activity from a malicious campaign. An APT, for instance, may involve reconnaissance, exploitation, and data exfiltration steps that occur weeks or even months apart. Detecting such threats requires models that can reason about the temporal dependencies between events.

3.1.1 Hybrid Architectures (GNN + RNN/GRU/LSTM)

The general architecture of these hybrid models involves a two-stage process. First, the evolving system activity is divided into a sequence of discrete time windows or graph snapshots. For each snapshot, a GNN is used to process the graph structure and generate embeddings for the nodes, capturing the spatial relationships at that specific moment in time. Second, the sequence of graph or node embeddings is fed into a recurrent model (like an LSTM or GRU), which is specifically designed to learn patterns and dependencies in sequential data.

3.1.2 Temporal Graph Neural Networks (TGNNs)

While hybrid models are powerful, they often treat the spatial and temporal components as separate, loosely-coupled modules. A more recent line of research focuses on creating deeply integrated architectures, known as Temporal Graph Neural Networks (TGNNs), that model spatio-temporal dynamics in a more unified manner. These models are designed to operate directly on evolving graphs, where node

features, edge features, and the graph topology itself can change over time.

Frameworks such as DURENDAL are designed for discrete-time, snapshot-based temporal heterogeneous networks. It introduces novel update schemes that aim for a "strictly intertwine between the heterogeneity and dynamicity aspects," moving beyond the simple pre- or post-processing of graph embeddings with a temporal model.

3.2 Modeling Heterogeneity

3.2.1 Heterogeneous Graphs

A heterogeneous graph explicitly models this diversity by allowing for multiple types of nodes and edges. This results in a much richer and more semantically meaningful representation of the system. For example, a heterogeneous graph could model a user node logging_into a device node, which then executes a process node that reads_from a file node. This level of semantic detail is essential for mapping observed activity to specific attacker Tactics, Techniques, and Procedures (TTPs).

3.2.2 Relational Graph Convolutional Networks (R-GCNs)

To learn from these rich structures, specialized GNNs are required. Relational Graph Convolutional Networks (R-GCNs) are a prominent architecture designed for heterogeneous graphs. The key innovation of an R-GCN is that it learns a distinct neural network transformation (i.e., a unique weight matrix) for each type of relation (edge type) in the graph. This allows the model to process information differently based on the semantic meaning of an interaction. For instance, the way it aggregates information along a DNS_query edge will be different from how it aggregates along an SSH_connection edge, enabling a far more precise understanding of system events.

However, this approach introduces a significant challenge: in a complex graph with hundreds of relation types, the number of model parameters can explode, leading to overfitting and computational inefficiency. To address this, R-GCNs employ regularization techniques like *basis decomposition*, where each relation-specific weight matrix is constructed as a linear combination of a smaller set of shared "basis" matrices. This form of weight sharing makes the model more efficient and robust, a necessary innovation for R-GCNs to be practical on complex, real-world security graphs.

IV. PROACTIVE DEFENSE: FROM REACTIVE DETECTION TO PROACTIVE THREAT HUNTING

4.1 Defining Proactive Threat Hunting

Threat hunting is not simply about looking for anomalies; it is a structured investigation that typically begins with a hypothesis. This hypothesis might be derived from external cyber threat intelligence (CTI), such as a report on a new TTP used by a particular threat actor. The hunter's goal is then to determine if that TTP is present within their own environment. Alternatively, a hunt can be triggered by known Indicators of Compromise (IOCs) or by analytics that surface subtle irregularities requiring human investigation.[45] This process combines human intuition and expertise with advanced technology to proactively uncover hidden adversaries.

4.2 GNNs as a Threat Hunting Engine

The effectiveness of threat hunting is directly proportional to the quality and comprehensiveness of the data available for analysis. The ideal data source for deep, host-level threat hunting is a system-wide *provenance graph*. A provenance graph is a detailed, directed acyclic graph that captures the complete history and causal dependencies of all events within a system. It tracks every interaction: every process created, every file accessed, and every network connection established, linking them together in a causal chain.[3] This provides an unparalleled, fine-grained record of system activity.

Within this context, GNNs enable a powerful new approach by framing threat hunting as a subgraph matching problem. The methodology is as follows:

1. An analyst or a CTI feed provides a known attack pattern, such as a sequence of TTPs from the MITRE ATT&CK framework.
2. This pattern is translated into a *query graph*, a small graph representing the structure of the malicious behaviour.
3. A GNN-based system is then tasked with searching the massive, system-wide provenance graph to find instances that match the structure of this query graph.

4.3 Review of GNN-Based Threat Hunting Frameworks

The primary challenge in applying GNNs to threat hunting is the sheer scale of provenance graphs, which can contain billions of nodes and edges in a large enterprise. Several frameworks have been proposed to address this.

- DeepHunter: This framework was designed specifically for robust graph pattern matching. It uses a GNN architecture with specialized attribute and graph embedding networks to match known attack behaviors against provenance data. Its key feature is its ability to find matches that are not completely consistent with the query graph, allowing it to detect attacker variations.
- SGMNet: To tackle the scalability problem head-on, SGMNet (Supervised Seeded Graph-Matching Network) introduces a two-stage approach. Instead of searching the entire provenance graph at once, it first uses high-confidence IOCs as "seed nodes." It then extracts smaller, more manageable subgraphs from the local neighbourhood of these seeds. A supervised graph matching network is then applied only to these compact subgraphs. This significantly reduces the search space and computational complexity, making the process more efficient and suitable for real-time environments.
- ProvG-Searcher: This approach formulates the search problem in a different way. It learns to embed entire subgraphs (both the query and candidates from the provenance graph) into a vector space. In this space, the relationship between two subgraphs (e.g., whether one is a subgraph of the other) can be evaluated directly through vector operations. This method has demonstrated extremely high accuracy in detecting query behaviors with a very low false positive rate.

V. THE CRITICAL APPLICATION: DETECTING LATERAL MOVEMENT

5.1 Framing Lateral Movement as a Graph Problem

The most effective and widely adopted approach for applying GNNs to this problem is to model lateral movement detection as **anomalous edge detection** or future link prediction in an evolving, temporal graph. In this model, network entities like hosts and user accounts are nodes, and interactions like authentications and network connections are time-stamped edges. The GNN-based system is trained on historical data to learn the normal patterns of connectivity and interaction within the network. It learns which users typically connect to which machines, using which protocols, at what times, and in what sequence.

Anomalous edge detection then works by identifying a new connection (an edge) that significantly deviates from these learned patterns. For example, a user account that normally only accesses marketing servers suddenly establishing an RDP connection to a domain controller would be a strong anomaly. Link prediction takes this a step further by forecasting the probability of future connections. A model might predict that the next likely benign action for a user is to access a shared drive. If that user instead attempts to connect to a developer's workstation, that low-probability event can be flagged as suspicious.

5.2 Review of GNN Frameworks for Lateral Movement Detection

Several specialized frameworks have been developed to implement this temporal graph-based approach to lateral movement detection.

- Euler: This is a formalized, model-agnostic framework that consists of a GNN layer stacked upon a sequence encoding layer (such as an RNN or GRU). It is designed for scalability and performance by decoupling the spatial (GNN) and temporal (RNN) components, allowing them to be distributed and computed independently. Euler models have been shown to be highly competitive with state-of-the-art methods for anomalous link prediction and can efficiently identify anomalous connections with high precision.
- LMTracker and LONGAN: These systems exemplify the use of more complex graph structures for this task. They leverage heterogeneous graphs, which can distinguish between different types of nodes (e.g., users, hosts) and edges (e.g., different authentication protocols). By incorporating these rich semantic details along with temporal features, they aim to detect more intricate and subtle lateral movement scenarios.

- Graph Foundation Models (GFMs): An emerging and promising direction is the application of large, pre-trained GNNs, often called Graph Foundation Models, to security tasks.[13] Similar to how large language models (LLMs) are pre-trained on vast text corpora and then fine-tuned for specific tasks, GFMs can be pre-trained on general graph data and then adapted for lateral movement detection. This approach holds the potential for better generalization across different network environments and could significantly reduce the amount of domain-specific labeled data required for training.

VI. OPENING THE BLACK BOX: THE IMPERATIVE OF EXPLAINABLE AI (XAI)

6.1 Review of XAI Methods and Frameworks

A variety of XAI techniques have been developed or adapted for GNNs.

- GNNExplainer: This is a seminal and widely cited model-agnostic explanation method for GNNs. For a given prediction, GNNExplainer works by learning a *mask* over the input graph's structure and features. This mask is optimized to highlight the smallest possible subgraph and the most critical node features that are most influential for the model's decision. The output is an intuitive, self-contained explanation in the form of a small, critical subgraph that an analyst can readily inspect.
- Other General-Purpose Explainers: Besides GNNExplainer, other methods have been applied, including gradient-based techniques like Integrated Gradients (IG), perturbation-based methods, and other subgraph-based approaches like PGExplainer and SubgraphX. A significant challenge in the field is that different explainers, using different underlying mechanisms, can produce markedly different explanations for the same prediction, making it difficult to determine which explanation is the most "correct" or faithful to the model's true reasoning.
- Domain-Specific Frameworks: Recognizing that generic explainers may not provide the most relevant insights for security, researchers have begun developing domain-specific frameworks. ProvExplainer is a framework tailored for provenance-based GNN intrusion detection. It generates explanations using an interpretable surrogate model and incorporates security-relevant features to provide more actionable insights for analysts. Similarly, Illuminati is a comprehensive framework designed to identify important nodes, edges, and attributes for cybersecurity applications without requiring prior knowledge of the GNN model's architecture.

VII. REAL-WORLD HURDLES: CHALLENGES, LIMITATIONS, AND ADVERSARIAL ROBUSTNESS

7.1 Scalability and Performance

The most immediate and practical challenge is scalability. Enterprise networks generate a colossal amount of security data, and the corresponding graphs can easily consist of millions of nodes and billions of edges. Applying GNNs to graphs of this magnitude presents severe computational and memory bottlenecks. The message-passing mechanism, which requires iterative aggregation over node neighborhoods, can be prohibitively expensive, making it difficult to train models and, more importantly, to perform inference in real time.

This leads to a significant gap between academic proposals and operational reality. Much of the existing research evaluates models on static, offline datasets representing several hours of traffic. However, an effective NIDS must make predictions in near real-time, on the order of seconds, to enable timely threat response. Bridging this performance gap is a critical area of research, with proposed solutions including more efficient GNN architectures, graph sampling techniques, and distributed graph processing frameworks.

7.2 Data Quality and Availability

The performance of any machine learning model is contingent on the quality of its training data, and GNNs are no exception. The field of cybersecurity faces several acute data-related challenges.

- Scarcity of Labeled Data: High-quality, publicly available datasets with accurate labels for malicious activities are notoriously scarce and difficult to obtain due to privacy and security concerns. This lack of labeled data makes it

challenging to train supervised GNN models effectively without suffering from overfitting, where the model memorizes the training data but fails to generalize to new, unseen threats. This limitation is a strong motivator for research into unsupervised, self-supervised, and few-shot learning methods that can learn from vast amounts of unlabeled data.

- Data Bias and Concept Drift: Existing datasets may not be representative of the full spectrum of modern cyber threats, leading to models with significant biases. For example, a model trained primarily on DDoS attacks may perform poorly at detecting stealthy data exfiltration. Furthermore, the baseline of "normal" behavior in a network is not static; it evolves over time as new applications are deployed, user behaviors change, and the infrastructure is updated. This phenomenon, known as *concept drift*, can cause a model's performance to degrade over time, leading to a steady increase in false positives unless it is continuously retrained or adapted.

7.3 Reproducibility and Generalization

The scientific rigor of the GNN-for-security field is facing a potential reproducibility crisis. A recent systematic study attempting to evaluate state-of-the-art GNN-based Intrusion Detection Systems (GIDS) found significant difficulties in reproducing and replicating the results reported in published papers. The study identified major performance discrepancies when models were re-implemented or tested on new datasets, citing challenges related to the lack of open-source code, unclear experimental configurations, and differences in hyperparameter settings.

This issue is closely tied to the problem of generalization. A model that achieves high accuracy on a specific, often sanitized, public dataset frequently experiences a sharp decline in performance when deployed in a new, real-world enterprise environment. This indicates that many current models are not learning generalizable patterns of malicious behavior but are instead overfitting to the specific characteristics of their training data. This poor generalization significantly increases the cost and effort of deployment, as models require extensive re-tuning for each new environment. These findings serve as a critical warning to the research community, highlighting an urgent need for more rigorous evaluation standards, the mandatory release of code and artifacts, and the development of large-scale, realistic benchmark datasets that better reflect the complexity of real-world networks.

7.4 Adversarial Robustness

As GNNs become a more common component of the defensive toolkit, they will inevitably become a target for attackers. Like other deep learning models, GNNs are vulnerable to *adversarial attacks*, where a malicious actor makes small, carefully crafted perturbations to the input data to induce a misclassification. In the context of GNNs, these attacks can target the graph's structure (by adding or deleting a few critical edges) or the nodes' features. An attacker could, for example, create a few seemingly benign connections to "poison" the graph and trick a GNN-based detector into classifying a malicious node as benign.

This marks the emergence of a new front in the cybersecurity arms race, moving from attacking endpoints and networks to attacking the AI models that defend them. The challenge is even greater for temporal graphs. A dynamic graph attack must be stealthy not only in the spatial domain but also in the temporal domain, accounting for the evolving importance of different nodes and the temporal dependencies between events to avoid detection.[33] Sophisticated attack frameworks like **HIA** (Hybrid Impact Attack) have been designed specifically to target TGNNs by strategically identifying and perturbing the most influential nodes and edges to maximize performance degradation. This evolution from defense to offense in the graph domain means that future GNN security research cannot focus solely on detection accuracy. It must prioritize the development of *robust* and *defensible* GNNs, employing techniques like adversarial training and certified defenses to buildmodels that are resilient to this new class of attacks.

VIII. CONCLUSION

The integration of Graph Neural Networks into cybersecurity represents a pivotal evolution in defensive capabilities. By shifting the analytical focus from isolated events to the intricate web of relationships that define system behavior, GNNs have unlocked the ability to detect complex, evasive, and

multi-stage threats that are beyond the reach of traditional, feature-based systems. This survey has charted the progression from foundational GNN architectures for intrusion detection to advanced temporal and heterogeneous models capable of capturing the dynamic reality of modern cyber environments. We have explored their application in critical tasks like proactive threat hunting and lateral movement detection, and underscored the necessity of Explainable AI to translate their powerful but opaque predictions into trusted and actionable intelligence.

REFERENCES

[1] X. Liu, X. Liu, K. Hao, K. Wang, X. Chen, and W. Niu, "HGNNDroid: Android Malware Detection Based on Heterogeneous Graph Neural Network," in 2024 IEEE 9th International Conference on Data Science in Cyberspace (DSC), 2024.

[2] N. C. Kavkas and K. Yildiz, "Enhancing IoMT Security with Deep Learning Based Approach for Medical IoT Threat Detection," in 2025 13th International Symposium on Digital Forensics and Security (ISDFS)

[3] T. Bilot, N. El Madhoun, K. Al Agha, and A. Zouaoui, "Graph Neural Networks for Intrusion Detection: A Survey," IEEE Access, vol. 11, pp. 49114–49139, 2023.

[4] K. John and K. Potika, "Diverse GNN Encoder-Decoder for Graph Anomaly Detection," in 2025 IEEE Conference on Artificial Intelligence (CAI),2025.

[5] S. M. Milajerdi, R. Gjomemo, B. Eshete, R. Sekar, and V. N. Venkatakrishnan, "HOLMES: Real-time APT Detection through Correlation of Suspicious Information Flows," in 2019 IEEE Symposium on Security and Privacy (SP), 2019,