SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

July 2026, Vol. 69, Iss. 7, 172305:1–172305:17
https://doi.org/10.1007/s11432-025-4776-x

GNNFormer: capturing local interactions and long-term global dependencies for APT detection in provenance graphs

Ning YANG[2,3], Nan WANG[1*], Hairong DONG[5], Jiqiang LIU[1] & Xibin ZHAO[4]

[1]*School of Cyberspace Science and Technology, Beijing Jiaotong University, Beijing 100044, China*
[2]*State Key Laboratory of Internet Architecture, Tsinghua University, Beijing 100084, China*
[3]*School of Software Engineering, Beijing Jiaotong University, Beijing 100044, China*
[4]*School of Software, Tsinghua University, Beijing 100084, China*
[5]*School of Electronic and Information Engineering, Tongji University, Shanghai 200092, China*

Received 11 March 2025/Revised 15 September 2025/Accepted 23 January 2026/Published online 8 June 2026

Abstract Advanced persistent threats (APTs) pose a formidable challenge in cybersecurity due to their stealthy, prolonged, and multi-stage attack techniques, which enable them to bypass many existing detection approaches. Despite considerable research efforts, most methods struggle to capture the complex dependencies and extended behavioral patterns that are essential for identifying these sophisticated threats. To address these challenges, we present GNNFormer, an end-to-end framework for APT detection that leverages provenance graph representation learning. GNNFormer embeds and processes system events within a unified architecture, capturing both local interactions and long-term global dependencies within the provenance graph to provide a comprehensive understanding of system behaviors. Furthermore, by employing a hierarchical attention mechanism, the framework dynamically prioritizes critical nodes and interactions, enhancing its focus on high-risk patterns while filtering out irrelevant information. Extensive evaluations on benchmark datasets demonstrate that GNNFormer significantly improves detection capabilities, establishing it as a scalable and effective solution for APT detection in complex cyber environments.

Keywords provenance graph, graph neural network, Transformer, computer security, information security, intrusion detection

Citation Yang N, Wang N, Dong H R, et al. GNNFormer: capturing local interactions and long-term global dependencies for APT detection in provenance graphs. Sci China Inf Sci, 2026, 69(7): 172305, https://doi.org/10.1007/s11432-025-4776-x

1 Introduction

Advanced persistent threats (APTs) are characterized by their stealthy, long-term, and multi-stage nature, presenting a significant challenge to network security [1]. APT attacks exploit sophisticated tactics—including zero-day exploits and advanced evasion techniques—which enable attackers to penetrate systems and maintain an undetected presence for extended periods [2]. The complexity and multi-stage progression of APTs expose the limitations of traditional detection methods, which often prove ineffective in identifying and preventing such attacks [3]. Consequently, there is an urgent need to develop robust, adaptive detection mechanisms that can safeguard sensitive data and critical organizational infrastructure against APTs [4].

Although traditional intrusion detection systems (IDSs) play an essential role in many cybersecurity strategies, they are generally inadequate for detecting APTs [5]. IDSs that rely on anomaly- or rule-based detection methods fail to capture the nuanced dependencies and long-term behavioral patterns that characterize APTs [6]. Their dependence on static signatures or predefined rules frequently results in high false-positive rates due to the inherent variability of legitimate system behaviors. Moreover, as APTs evolve to exploit new vulnerabilities and adapt to countermeasures, rule-based systems require constant updates—a process that is both costly and time-consuming [7].

In response to these challenges, researchers have explored alternative frameworks for APT detection, with provenance graphs emerging as a particularly promising tool [8]. Provenance graphs provide a comprehensive, causal representation of system activity by capturing relationships among entities such as processes, files, and network connections. This system-wide perspective is essential for identifying the complex, multi-stage attack patterns typical of APTs. However, current provenance-based methods often emphasize short-term sequences, limiting their

* Corresponding author (email: wangnanbjtu@bjtu.edu.cn)

ability to capture the long-term dependencies critical for detecting sophisticated APTs. Prior studies have advanced provenance-based APT detection, yet many approaches decouple semantic encoding, dependency modeling, and detection, or rely on handcrafted templates and selective traversal. These choices limit their ability to comprehensively capture evolving multi-stage behaviors.

To address these limitations, we present GNNFormer, an end-to-end framework specifically designed for provenance-based APT detection. In contrast to previous approaches, GNNFormer unifies semantic representation, structural reasoning, and detection within a single architecture. It jointly models local interactions and long-term global dependencies in provenance graphs, enabling effective representation of both immediate behaviors and extended temporal correlations. Within this layered representation process, a hierarchical attention mechanism dynamically prioritizes critical nodes and interactions while down-weighting irrelevant activities. This unified design reduces reliance on external rules or templates and allows the model to directly learn task-specific representations for detecting complex, multi-stage APT campaigns.

Comprehensive evaluations on benchmark datasets demonstrate that the design consistently achieves strong performance. The results suggest that capturing both structural and temporal dependencies within a unified framework is crucial for improving robustness in noisy, real-world environments.

The principal contributions of this paper are as follows.

• We propose GNNFormer, an end-to-end framework that jointly models local interactions and long-term global dependencies in provenance graphs for APT detection.

• GNNFormer provides a comprehensive representation of system behavior by integrating semantic and structural reasoning into a unified framework, enabling the discovery of complex attack patterns without relying on handcrafted rules or selective traversal.

• A hierarchical attention mechanism embedded in the representation learning process adaptively prioritizes critical interactions, improving detection of subtle and rare APT behaviors.

The remainder of this paper is organized as follows: Section 2 reviews related work in APT detection and provenance graph learning; Section 3 details the design and methodology of GNNFormer; Section 4 presents the experimental setup and evaluation results, including comparisons with state-of-the-art techniques; and Section 5 concludes the paper by summarizing the contributions and potential impact of our work.

2 Related work

Provenance graph-based APT detection leverages rich system-level data to capture interactions among entities (e.g., files, processes) over time. Existing approaches in this area generally fall into three categories [9]: anomaly-score-based [10–12], tag-propagation-based [13–15], and graph-matching-based methods [16, 17].

Anomaly-score-based methods detect APTs by assigning a suspiciousness score to system behaviors that deviate from normal patterns. For instance, Pagoda [12] computes an anomaly score for each path within a provenance graph based on predefined rules and then multiplies the path length by its score to derive a weighted anomaly metric. If this metric exceeds a certain threshold, an alert is triggered, indicating potential malicious activity. Similarly, NoDoze [11] reduces false positives by employing path-based anomaly scores that help filter out benign events, thereby addressing the problem of "alert fatigue". In addition, Bayesian network models use conditional probabilities to evaluate the likelihood of attacks at suspicious nodes [18,19]. Although these methods can effectively highlight anomalous behaviors, their heavy reliance on predefined rules and expert-crafted heuristics limits their adaptability to novel APT tactics. To address these limitations in dynamic environments, recent research has introduced intraclass consistency enhanced variational autoencoders to identify malicious traffic even under concept drift [20]. Furthermore, hybrid frameworks integrating explainable neural networks (such as HEN) have been proposed to improve the robustness and interpretability of network intrusion detection systems [21].

Tag-propagation-based approaches monitor the diffusion of flagged entities throughout provenance graphs to reveal potential attack paths. For example, SLEUTH [15] performs backward analysis to trace attacks to their entry points and then carries out forward analysis to identify affected nodes. Enhancements, such as those introduced in MORSE [22], mitigate the issue of dependency explosion by filtering out benign nodes, while HOLMES [14] extends the provenance graph by mapping low-level data to high-level semantic chains, thereby constructing scenario graphs that facilitate advanced attack detection. However, these approaches require pre-established tagging rules and labeled data, which limits their flexibility in dynamic environments where previously unknown threats may emerge.

Graph-matching-based methods detect APTs by aligning subgraphs within the provenance graph with known attack patterns. For instance, POIROT [23] aligns query graphs that represent known attack sequences to portions of the provenance graph to identify previously observed malicious behaviors. UNICORN [17] clusters feature

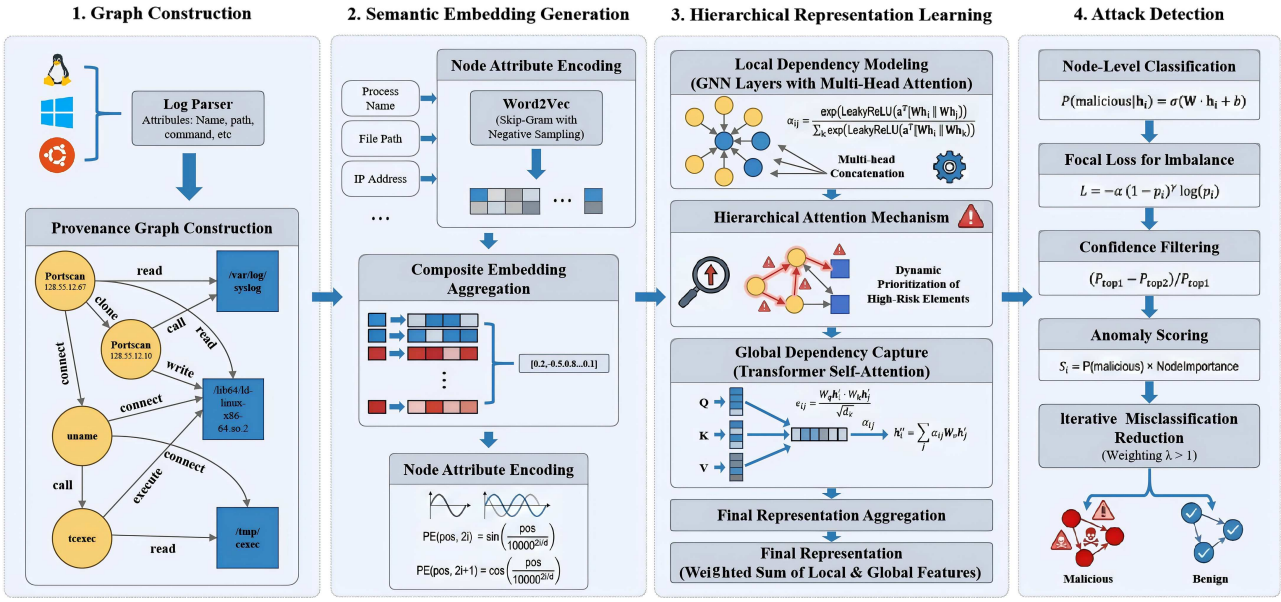


Figure 1 (Color online) Overview of GNNFormer's detection pipeline.

vectors extracted from provenance graphs to represent system states and uses machine learning techniques to track abnormal state transitions over time. Although effective for recognizing known attack patterns, these methods are computationally intensive and struggle to generalize to novel APT behaviors due to their reliance on predefined attack templates. Moreover, their focus on static graph states makes it difficult to track the sequential nature inherent in APTs.

While the aforementioned approaches have laid the foundation for provenance-based APT detection, they exhibit limitations in capturing long-term dependencies and complex multi-stage behaviors. To address these shortcomings, recent studies have explored advanced representation learning. Some approaches emphasize spatio-temporal graph modeling, such as the recurrent semantic evidence-aware graph neural network (RE-SEGNN), to capture dynamic system behaviors and forecast temporal evolution [24]. Others focus on data quality limitations, employing novel graph oversampling frameworks to rectify class imbalance in node classification tasks [25]. Additionally, the emergence of large language model-based agents offers new potential for autonomous reasoning and adaptive strategies in security operations [26]. Despite these advances, most solutions remain confined to a single modeling perspective—such as temporal alignment or lightweight anomaly scoring—without fully integrating structural and temporal dependencies into a unified framework.

Nevertheless, detecting low-frequency, persistent attacks within complex provenance graph data remains a significant challenge. The inherent complexity of managing provenance information, combined with the constraints of existing methods, often hinders effective detection. To address this, we propose GNNFormer, an end-to-end framework that jointly models local interactions and long-term global dependencies. By embedding hierarchical attention within the layered representation process, GNNFormer dynamically emphasizes high-risk behaviors while down-weighting irrelevant activities. This unified design enables the discovery of subtle and multi-stage APT patterns without predefined rules, templates, or handcrafted traversal strategies.

3 Methodology

3.1 Overview

The GNNFormer framework is designed to detect APTs by analyzing provenance graphs that capture both local interactions and long-term dependencies within system activities. Figure 1 illustrates the overall architecture of GNNFormer, which comprises four principal components forming a comprehensive detection pipeline.

In the initial phase, system logs are transformed into provenance graphs. In this process, nodes represent system entities—such as processes and files—and edges denote the interactions between them. This transformation establishes a structured representation that captures the underlying relationships and causal pathways in system activities, thereby providing a solid foundation for understanding behavioral patterns among interconnected entities.

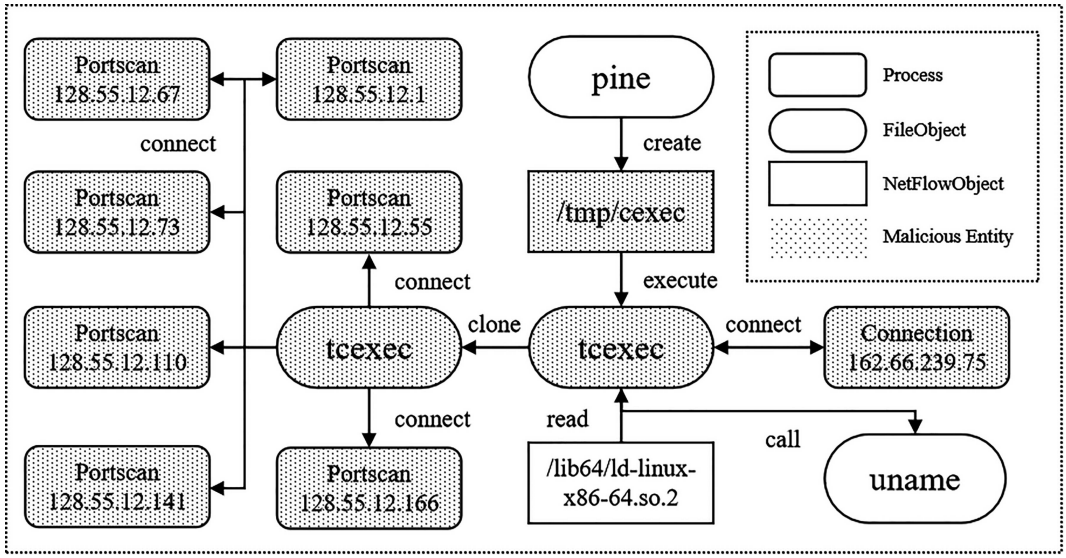


Figure 2 Example of provenance graph.

In practice, system activities are no longer treated as isolated events but as parts of a causally connected structure, which is crucial for uncovering attack chains that unfold over time.

Next, each node in the provenance graph is enriched with semantic data. Contextual attributes such as process names and file paths are encoded into dense vector representations, which offer a more nuanced depiction of each node's role within the graph. This embedding step maps categorical attributes into a continuous space, placing entities with similar roles close to each other and enabling the model to generalize beyond exact string matches. This semantic embedding enhances the overall graph by incorporating contextual relationships, thereby contributing to a deeper and more meaningful representation of node interactions.

At the core of the framework, GNNFormer's architecture incorporates specialized layers dedicated to capturing both local structural features and extended dependencies across the graph. Local dependency modeling focuses on immediate neighborhoods, while global dependency modeling captures long-range relationships across the entire graph. By embedding these perspectives into a hierarchical representation process, the framework learns fine-grained and distributed patterns simultaneously. By integrating these complementary perspectives, the model is able to analyze both immediate interactions and long-term global dependencies, constructing a holistic representation that reflects the nuanced relationships characteristic of APT behaviors.

Finally, in the analysis phase, GNNFormer leverages the learned representations to identify anomalous or high-risk patterns indicative of APT activities. Rather than applying attention as a separate step, the hierarchical attention mechanism is embedded within the representation learning process, guiding the model to emphasize critical nodes and edges while down-weighting less relevant ones. This targeted focus facilitates the precise detection of subtle threat indicators, thereby enhancing the overall effectiveness of the APT detection process.

3.2 Provenance graph construction

The construction of the provenance graph commences with interpreting system logs from diverse sources, such as Windows event logs[1)] and Linux audit logs[2)]. These logs record various system events—including process executions, file operations, and network connections—that are essential for mapping system interactions. Each event contains details such as timestamps, event types, and involved entities, thereby enabling a sequential and causal representation of system activity. By systematically mapping these events, the provenance graph captures behavioral patterns that may indicate APT activities (see Figure 2). This transformation step converts raw event streams into a structured graph representation, where relationships between entities are explicitly modeled rather than implicitly embedded in log text.

To efficiently manage the substantial volume of log data, events are processed in batches, with each batch containing a predefined number (K) of events. This batch-processing approach not only preserves the temporal sequence of events but also prevents resource overload, thereby maintaining long-term global dependencies and

1) Event tracing. https://docs.microsoft.com/en-us/windows/desktop/ETW/event-tracing-portal. 2024-11-07.
2) The Linux Audit Daemon. https://linux.die.net/man/8/auditd. 2024-11-07.

enabling the system to handle high data throughput effectively. In practice, this batching strategy enables the framework to scale to millions of events while preserving important temporal links across them.

After parsing the events, a directed provenance graph is constructed, comprising two primary categories of nodes: process nodes and object nodes. Process nodes represent active processes and capture attributes such as process names and command-line arguments, whereas object nodes denote files, network connections, or other resources with which processes interact. This distinction allows the framework to track both program execution logic and interactions with system resources.

Edges between nodes illustrate causal relationships; they are labeled by event type (e.g., system calls such as read, write, or execute) and annotated with timestamps. For example, an edge from a process node to a file node labeled "write" indicates that the process wrote data to the file. These labels and timestamps provide essential contextual information, reflecting both the sequence and the nature of interactions within the system. Thus, the provenance graph encodes causal chains of events, which are critical for reconstructing multi-stage APT scenarios.

To enrich the analytical depth of the graph, each node and edge is further augmented with additional attributes—including process names, command-line arguments, file paths, IP addresses, port numbers, and module paths. These attributes are crucial for distinguishing between benign and potentially malicious interactions, as they provide context that can uncover hidden relationships or suspicious patterns. For example, repeated access to sensitive files or unusual communication on uncommon ports can be highlighted once these attributes are integrated into the graph structure. By incorporating these details, the provenance graph becomes a comprehensive model of system behavior, enhancing its capacity to reveal intricate patterns associated with APTs.

3.3 Semantic embedding generation

The system logs provide a wealth of attributes related to various entities, such as processes, files, and network connections. Attributes—such as process names, file paths, IP addresses, and command-line arguments—must be transformed into dense vector representations to be effectively utilized by the GNNFormer model. Traditional encoding methods, like one-hot encoding or bag-of-words, yield high-dimensional, sparse vectors that lack semantic context. To overcome these limitations, we employ the Word2Vec model to map these attributes into a dense, low-dimensional vector space [27]. This enables the model to measure similarity between entities in a continuous space instead of treating them as unrelated tokens.

The Word2Vec model [28] learns vector representations by capturing contextual similarities. Specifically, we adopt the skip-gram with negative sampling (SGNS) approach [29] to train Word2Vec on sequences derived from system logs. Given a target attribute w_t within a sequence (or "sentence") of attributes, the goal of the skip-gram model [30] is to maximize the likelihood of its surrounding context attributes $w_{t-k}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+k}$, where k denotes the size of the context window. Mathematically, this objective is expressed as

$$\prod_{-k \leqslant j \leqslant k \ \ j \neq 0} P(w_{t+j} \mid w_t). \tag{1}$$

In other words, the model learns to predict attributes appearing near a target token, ensuring that attributes used in similar contexts obtain embeddings close to each other in vector space.

To compute this efficiently, negative sampling is employed to approximate the conditional probabilities. The objective function of SGNS is given by

$$\log P(w_{t+j} \mid w_t) = \log \sigma\left(\mathbf{v}_{w_{t+j}} \cdot \mathbf{v}_{w_t}\right) + \sum_{n=1}^{N} \mathbb{E}_{w_n \sim P_n(w)} \left[\log \sigma\left(-\mathbf{v}_{w_n} \cdot \mathbf{v}_{w_t}\right)\right], \tag{2}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, $\mathbf{v}_{w_t}$ is the embedding vector of the target attribute w_t, $\mathbf{v}_{w_{t+j}}$ represents the embedding vector of a context attribute, w_n are negative samples drawn from a noise distribution $P_n(w)$, and N is the number of negative samples. Put simply, the model increases similarity between a target and its true context while decreasing similarity with randomly sampled attributes outside its context.

After training, each attribute w is mapped to a dense vector $\mathbf{v}_w \in \mathbb{R}^d$, where d is the embedding dimension. For each node i in the provenance graph—which may possess multiple attributes (e.g., process name, IP address)—we generate a composite embedding $\mathbf{v}_i$ by aggregating the embeddings of its attributes. A common approach is to compute the average:

$$\mathbf{v}_i = \frac{1}{|A_i|} \sum_{w \in A_i} \mathbf{v}_w, \tag{3}$$

where A_i denotes the set of attributes associated with node i. This ensures that each node has a single dense vector capturing all its key properties, enabling comparisons across heterogeneous node types.

Although Word2Vec captures semantic similarity between attributes, it does not inherently preserve the sequential order of events—a critical aspect for APT detection. To incorporate temporal information, we adopt a positional encoding scheme inspired by Transformers. This scheme enables the model to distinguish the order of events in each node's interactions. Without positional encoding, embeddings of the same attributes would remain identical regardless of when they occurred, obscuring temporal patterns that often characterize multi-stage attacks.

The positional encoding vector for a token at position pos in a sequence with embedding dimension d is defined as

$$PE(\text{pos}, 2i) = \sin\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right), \; PE(\text{pos}, 2i+1) \quad = \cos\left(\frac{\text{pos}}{10000^{2i/d_{\text{model}}}}\right), \tag{4}$$

where i denotes the index of each dimension. This positional encoding $PE(\text{pos})$ is added element-wise to the Word2Vec embedding of each token based on its position:

$$\text{Emb}_i = \mathbf{v}_i + PE(\text{pos}). \tag{5}$$

This ensures that two identical attributes appearing at different time steps have distinct final embeddings, allowing the model to differentiate between early and late occurrences of the same action.

By combining Word2Vec embeddings with positional encodings, we obtain a representation that captures both the content and the temporal order of each node's attributes. This enriched embedding enables GNNFormer to effectively interpret the sequence of interactions, which is crucial for identifying the multi-level, temporally ordered behaviors typical of APTs.

In summary, semantic embedding generation lays the foundation for downstream representation learning. It ensures that system entities are represented not only by what they are (semantic similarity) but also by when and how they act (temporal encoding), thereby enabling GNNFormer to detect subtle, context-dependent malicious behaviors.

3.4 Hierarchical representation learning

Hierarchical representation learning is a cornerstone of GNNFormer's ability to analyze complex, layered attack behaviors within provenance graphs. By integrating layers that capture both local interactions and long-term global dependencies, GNNFormer can identify subtle patterns indicative of APTs. Unlike prior methods that model these dependencies in separate stages or through additional heuristics, GNNFormer embeds both levels of reasoning into a unified, end-to-end framework. A hierarchical attention mechanism spans this process to adaptively emphasize critical interactions while down-weighting irrelevant activities, ensuring that the learned representations remain task-specific even in noisy environments. This approach overcomes the limitations of traditional methods, which excel at local message passing but often struggle to capture broader dependencies in complex provenance graphs. The following sections detail each stage of the representation learning process, emphasizing how the integration of local and global perspectives contributes to a rich, context-aware representation of system behavior.

3.4.1 *Local dependency modeling*

Local dependency modeling in GNNFormer focuses on capturing interactions within each node's immediate neighborhood. This process leverages an attention-based mechanism that enables the model to prioritize relevant local interactions potentially indicative of malicious behavior. Specifically, each node aggregates information from its neighbors, with each neighbor's influence determined by an attention coefficient reflecting the importance of the connection.

For a given node i with feature vector $\mathbf{h}_i$ and a set of neighboring nodes $\mathcal{N}(i)$, the attention coefficient α_{ij} between node i and a neighboring node j is computed as follows:

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^{\text{T}}[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_j]))}{\sum_{k\in\mathcal{N}(i)} \exp(\text{LeakyReLU}(\mathbf{a}^{\text{T}}[\mathbf{W}\mathbf{h}_i \| \mathbf{W}\mathbf{h}_k]))}, \tag{6}$$

where $\mathbf{h}_i$ and $\mathbf{h}_j$ are the feature vectors of nodes i and j, respectively; $\mathbf{W}$ is a learnable weight matrix; $\mathbf{a}$ is an attention vector; and $\|$ denotes concatenation. The LeakyReLU function introduces non-linearity, and the attention coefficient α_{ij} reflects the relevance of node j to node i in the context of the local graph structure. Intuitively,

this equation determines which neighbors of node i should contribute more to its updated embedding, allowing the model to highlight interactions most relevant to potential attack behavior.

To further enhance expressiveness, GNNFormer employs multi-head attention, where multiple attention heads operate in parallel, each capturing different aspects of the local interactions. For each attention head m, the coefficients α_{ij}^m and the node features $\mathbf{h}_i'$ are computed independently. The outputs of all heads are then concatenated as follows:

$$\mathbf{h}_i' = \Big\|_{m=1}^{M} \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^m \mathbf{W}^m \mathbf{h}_j\right), \tag{7}$$

where M is the number of attention heads and $\|$ denotes concatenation. As part of the hierarchical attention process, this mechanism ensures that important short-range dependencies are preserved before global reasoning is applied. Multiple heads capture different interaction patterns simultaneously, producing a more expressive neighborhood representation.

3.4.2 *Global dependency capture*

While capturing local dependencies is essential for understanding node-level interactions, detecting complex APTs often requires a broader perspective that encompasses long-term and distant relationships within the graph. Localized patterns alone may not reveal distributed attack behaviors that span multiple nodes and time intervals. To address this challenge, GNNFormer incorporates a global dependency modeling component to capture the wider context within the provenance graph.

To achieve this, Transformer layers are introduced after the local dependency modeling stage. The self-attention mechanism of the Transformer allows each node to attend directly to all other nodes, regardless of their distance, thus capturing both local and long-range dependencies. Together with local attention, this forms a hierarchical process in which attention operates at multiple granularities, yielding a balanced representation of neighborhood-level details and graph-wide correlations. This global attention is crucial for identifying coordinated behaviors indicative of multi-stage attack patterns and distributed anomalies [31].

In this component, the Transformer layer computes an attention score e_{ij} between node i and node j, defined as

$$e_{ij} = \frac{(\mathbf{W}_q \mathbf{h}_i') \cdot (\mathbf{W}_k \mathbf{h}_j')}{\sqrt{d_k}}, \tag{8}$$

where $\mathbf{W}_q$ and $\mathbf{W}_k$ are learnable projection matrices for the query and key vectors, respectively, and d_k is the dimension of the key vector for scaling. This score reflects how much attention node i should assign to node j when building its global representation.

These attention scores e_{ij} are then normalized using a softmax function to produce the attention weights α_{ij}:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{j=1}^{N} \exp(e_{ij})}. \tag{9}$$

Finally, the output for each node i is computed as a weighted sum of the features of all nodes:

$$\mathbf{h}_i'' = \sum_{j=1}^{N} \alpha_{ij} (\mathbf{W}_v \mathbf{h}_j'), \tag{10}$$

where $\mathbf{W}_v$ is a learnable weight matrix for the value projections. This allows each node to update its embedding with information from the entire graph while retaining the emphasis on high-risk dependencies established earlier. The final representation thus reflects both fine-grained local interactions and broad temporal-spatial correlations.

3.4.3 *Final representation aggregation*

After the hierarchical representation learning process, each node is assigned a final embedding that encapsulates both local and long-term global dependencies, with hierarchical attention guiding their integration. These embeddings collectively form a comprehensive representation of the graph, capturing nuanced interactions and relationships across nodes. Rather than immediately classifying nodes as malicious or benign, these embeddings serve as input for the subsequent attack detection stage, where deeper analysis identifies potential threats.

This embedding-based aggregation provides GNNFormer with a context-aware view of the entire provenance graph. By unifying short- and long-range dependencies within a single end-to-end framework, GNNFormer avoids reliance on external rules or handcrafted templates and produces task-specific representations that preserve critical indicators of APT behaviors. This design establishes a solid foundation for the detection phase.

3.5 Attack detection

The attack detection phase of GNNFormer is dedicated to node-level classification within the provenance graph. Here, each node is evaluated individually to determine its likelihood of being involved in malicious activity. Instead of relying on a single classifier, GNNFormer integrates multiple complementary strategies—including focal loss, confidence filtering, anomaly scoring, and iterative correction—to better identify rare and stealthy malicious behaviors characteristic of APTs.

3.5.1 *Node-level classification*

Each node embedding $\mathbf{h}_i$, obtained from the hierarchical representation learning stage, is classified as either benign or malicious using a softmax probability score [32]. This classification is achieved by passing the embedding through a fully connected layer with a softmax activation function:

$$P(\text{malicious}|\mathbf{h}_i) = \sigma(\mathbf{W} \cdot \mathbf{h}_i + b), \tag{11}$$

where $\mathbf{W}$ is the weight matrix for the classification layer, b is the bias term, and σ is the softmax function that outputs a probability between 0 and 1. A node is flagged as malicious if its probability exceeds a predefined threshold. This provides every node with an explicit decision boundary, which forms the basis for more refined detection mechanisms.

3.5.2 *Addressing class imbalance with focal loss*

APT detection datasets are highly imbalanced: benign nodes vastly outnumber malicious ones. To prevent the model from being biased toward the majority class, we employ focal loss [33]. This function reduces the impact of well-classified benign samples and emphasizes difficult cases:

$$\mathcal{L} = -\alpha \sum_i (1 - p_i)^\gamma \log(p_i), \tag{12}$$

where α balances class contributions and γ focuses the training more strongly on misclassified nodes. In practice, a benign process performing routine file reads contributes less to the gradient than a suspicious process whose behavior is hard to classify. This shifts learning capacity toward subtle malicious behaviors that would otherwise be overshadowed by the benign majority.

3.5.3 *Confidence-based filtering*

Although focal loss improves training balance, predictions at inference time may still vary in certainty. To address this, we incorporate a confidence-based filtering step. Each node is assigned a confidence score [34], computed as

$$\text{confidence} = \frac{P_{\text{top1}} - P_{\text{top2}}}{P_{\text{top1}}}, \tag{13}$$

where P_{top1} and P_{top2} are the highest and second-highest predicted probabilities. This measure reflects the model's certainty. Low scores often indicate ambiguous cases, such as benign nodes partially resembling attack behavior. By discarding low-confidence predictions, GNNFormer avoids excessive false alarms and focuses on cases with higher certainty.

3.5.4 *Anomaly scoring with node centrality*

APT campaigns often exploit highly connected processes—such as browsers or system daemons—that serve as hubs for multiple malicious actions. Detecting such nodes requires combining classification scores with structural influence. To this end, GNNFormer computes an anomaly score:

$$S_i = P(\text{malicious}|\mathbf{h}_i) \times \text{NodeImportance}(i), \tag{14}$$

Table 1 Overview of DARPA TC E3 datasets.

Scene	System	# of benign nodes	# of abnormal nodes	# of edges
THEIA	Ubuntu	3505326	25362	102929710
Trace	Ubuntu	2416007	67383	6978024
CADETS	FreeBSD	706966	12852	8663569

where NodeImportance(i) is measured using centrality metrics such as PageRank or eigenvector centrality. This ensures that nodes classified as suspicious and structurally central in the provenance graph receive higher priority. For instance, a file repeatedly accessed by many processes or a process initiating multiple network connections would yield a higher anomaly score, making it a stronger candidate for investigation.

3.5.5 *Iterative misclassification reduction*

Even with these mechanisms, some malicious nodes may remain misclassified due to rarity or similarity to benign behaviors. To further refine detection, we adopt an iterative misclassification reduction strategy. At each training epoch, nodes misclassified in the previous epoch are collected:

$$\mathcal{M} = \{v \mid \mathbf{y}_v^{(t)} \neq \hat{\mathbf{y}}_v\}, \tag{15}$$

and their contribution to the loss is increased in the next epoch using a weighting factor $\lambda > 1$. This directs additional capacity to the most challenging cases. Over time, the set $\mathcal{M}$ evolves, meaning the model continually revisits its mistakes and gradually sharpens its decision boundary.

This mechanism improves recall on minority malicious nodes while avoiding instability from overemphasizing a fixed set. In our evaluation (Section 4), iterative refinement proved particularly effective for identifying subtle attack steps that mimic benign system activities, such as privilege escalation through legitimate processes.

Together, these components—classification, focal loss, confidence filtering, anomaly scoring, and iterative refinement—form a layered defense strategy. Each mechanism addresses a specific weakness of the detection task, and their integration enables GNNFormer to detect both obvious and subtle malicious behaviors within complex provenance graphs.

4 Experiments

4.1 Dataset

To thoroughly evaluate the effectiveness of our proposed method, we utilize the DARPA TC E3 dataset, which was generated during the third red-team versus blue-team engagement of the DARPA Transparent Computing program. This two-week-long engagement produced provenance data that capture system activities from start to finish using dedicated collection tools. Both the provenance data and the corresponding ground truth are publicly available[3)], and detailed statistics can be found in Table 1.

The dataset comprises multiple sub-datasets (e.g., THEIA, Trace, and CADETS) collected from diverse system environments. Each sub-dataset provides labeled data that carefully distinguishes between benign and malicious activities, making it especially suitable for node-level anomaly detection. In our experiments, we use the ground truth to label abnormal nodes and evaluate our framework's ability to track anomalies. To ensure reliability, we remove portions of the dataset that were generated during exceptional events. Nodes extracted from benign graphs are used for model training, while those from graphs containing threats serve as evaluation samples.

In addition to DARPA TC E3, we evaluate GNNFormer on the DARPA OpTC dataset[4)], which spans eight days of provenance logs with three dedicated evaluation days. Each evaluation day features a distinct attack type (PowerShell Empire, data exfiltration, or malware upgrades) launched on separate hosts. Compared with E3, OpTC offers larger-scale, multi-host traces and captures more diverse adversarial strategies, making it particularly valuable for assessing node-level detection in dynamic enterprise environments. A summary of its characteristics is provided in Table 2.

To further evaluate generalizability, we include the StreamSpot dataset [35], which contains 500 benign and 100 malicious streaming provenance subgraphs. As summarized in Table 3, each graph has thousands of nodes

3) DARPA Transparent Computing Program Engagement 3 Data Release. https://github.com/darpa-i2o/Transparent-Computing. 2024-11-07.

4) DARPA OpTC, https://github.com/FiveDirections/OpTC-data. 2025-09-07.

Table 2 Overview of DARPA OpTC dataset.

Attack type	# of Hosts	Duration (days)	Description
PowerShell empire	1	1	Credential theft and persistence
Data exfiltration	1	1	Sensitive data theft via network
Malware upgrade	1	1	Multi-stage malware deployment

Table 3 Overview of StreamSpot dataset.

Scene	# of graphs	Avg. # of nodes	Avg. # of edges
Benign	500	8315	173857
Attack	100	8891	28423

and edges, with benign and attack graphs showing distinct connectivity patterns. Unlike DARPA TC E3 and OpTC, which support node-level anomaly detection, StreamSpot is designed for graph-level detection. To ensure fair comparison with baselines such as StreamSpot, Unicorn, and WLSubtree, we extend GNNFormer to the graph level by aggregating node embeddings into graph representations before classification. This adaptation allows us to assess whether hierarchical representation learning generalizes beyond node-level tasks.

4.2 Evaluation metrics

To comprehensively evaluate the effectiveness of GNNFormer in detecting APTs, we employ three key metrics: Precision, Recall, and the F_1-score. These metrics are widely used in anomaly detection and classification tasks, and they provide a robust assessment of model performance—especially in scenarios involving imbalanced datasets and rare attack events.

Precision measures the proportion of true positive predictions among all instances predicted as positive. In security applications, a high Precision is critical as it indicates the model's ability to minimize false alarms, thereby preventing alert fatigue and ensuring that security analysts are notified only of genuine threats.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}. \tag{16}$$

Recall, on the other hand, calculates the proportion of true positive cases that are correctly identified out of all actual positive cases. In the context of APT detection, high Recall is essential to ensure that as many attack events as possible are captured, reducing the risk of overlooking potential threats.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}. \tag{17}$$

The F_1-score, defined as the harmonic mean of Precision and Recall, provides a balanced metric that considers both false positives and false negatives. This measure is particularly relevant for imbalanced datasets, as it reflects the trade-off between detecting all threats and minimizing false alarms.

$$F_1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{18}$$

4.3 Experimental setup

To comprehensively evaluate the performance of GNNFormer, we designed experiments to address several key research questions. All experiments were conducted on a machine equipped with an Intel Core i9-14900K CPU, 64 GB of RAM, and an NVIDIA RTX 4090 GPU, running Windows 11. In our configuration, an event batch size of 250000 was used, and a Jaccard Similarity threshold of 1 was adopted.

4.4 Comparing methods

To evaluate the effectiveness of our proposed approach, we compare GNNFormer with several state-of-the-art models, each employing unique methodologies for network threat and APT detection. These models differ in their graph-based techniques, data representation methods, and anomaly detection mechanisms, thereby providing a diverse benchmark for assessing detection capabilities in complex cyber environments.

DeepLog [36]: DeepLog utilizes an LSTM network to learn typical log patterns and identify anomalies when deviations occur, enabling effective sequence-based anomaly detection in system logs.

Table 4 Comparison between GNNFormer and state-of-the-art APT detection methods on the DARPA TC E3 datasets. The best results are in bold.

Method	Cadets (E3)			Trace (E3)			Theia (E3)		
	Precision	Recall	F_1-score	Precision	Recall	F_1-score	Precision	Recall	F_1-score
DeepLog	0.23	0.74	0.35	0.41	0.68	0.51	0.16	0.14	0.15
Log2vec	0.49	0.85	0.62	0.54	0.78	0.64	0.62	0.66	0.64
MAGIC	0.94	0.99	0.97	0.99	0.99	**0.99**	0.98	0.99	0.99
ThreaTrace	0.90	0.99	0.95	0.72	0.99	0.83	0.87	0.99	0.93
FLASH	0.94	0.99	0.96	0.95	0.99	0.97	0.92	0.99	0.95
Slot	0.96	0.99	0.97	0.99	0.97	**0.99**	0.98	0.99	0.99
TAPAS	**0.99**	0.99	0.99	0.99	**1.00**	**0.99**	0.98	0.96	0.97
STGAN	0.98	0.99	0.99	0.99	0.98	**0.99**	0.94	0.99	0.97
GNNFormer (Ours)	**0.99**	**1.00**	**1.00**	**1.00**	0.99	**0.99**	**1.00**	**1.00**	**1.00**

Log2vec [37]: Log2vec employs heuristic techniques to transform log entries into heterogeneous graphs. By clustering log entries into malicious and benign groups, it facilitates structured threat detection in enterprise environments.

MAGIC [38]: MAGIC leverages masked graph representation learning in a self-supervised manner to model benign behaviors. This approach supports the detection of APTs across varying supervision levels by effectively capturing underlying graph structures.

ThreaTrace [39]: ThreaTrace constructs provenance graphs from system audit data to trace threats at the node level. By analyzing entity interactions, it detects host-based threats with a focus on the dynamic behavior of the system.

FLASH [40]: FLASH utilizes GNN-based provenance graph learning to detect APT attacks. By integrating Word2Vec-based semantic encoding and temporal ordering, it improves anomaly detection accuracy. To enhance scalability, FLASH employs selective graph traversal and a precomputed embedding database, enabling efficient and real-time threat detection.

Slot [41]: Slot introduces a lightweight graph learning framework with streamlined attention mechanisms for APT detection. By reducing model complexity while maintaining accuracy, Slot achieves strong detection performance with lower computational overhead, making it practical for real-time monitoring environments.

TAPAS [42]: TAPAS proposes a temporal-aware provenance analysis system that leverages time-sensitive semantic embeddings to model the evolution of system behaviors. By explicitly capturing long-range temporal dependencies, TAPAS achieves robust detection of multi-stage APT campaigns and adapts effectively to diverse attack strategies.

STGAN [43]: STGAN incorporates spatio-temporal graph attention networks to jointly model structural and temporal dependencies in provenance data. Its unified attention mechanism enables fine-grained anomaly localization, making it effective for identifying coordinated attack activities across multiple entities.

Prov2Vec [44]: Prov2Vec introduces a provenance graph kernel combined with histosketching to generate compact representations of system behaviors. By profiling benign host activity and detecting deviations, it supports tasks such as graph classification and clustering, offering scalability for large enterprise networks.

Unicorn [17]: Unicorn is a runtime provenance-based detector that employs graph sketching and evolutionary modeling to capture long-running system behaviors. By summarizing provenance graphs into fixed-size sketches and clustering them into evolutionary models, Unicorn effectively detects stealthy, long-duration APT campaigns while remaining efficient in streaming environments.

4.5 Experimental results

As shown in Table 4, GNNFormer consistently achieves high Precision, Recall, and F_1-score across all datasets in the DARPA TC E3 suite. On the CADETS (E3) dataset, GNNFormer attains a Precision of 0.99, a Recall of 1.00, and an F_1-score of 1.00, clearly demonstrating its capability to effectively distinguish between benign and malicious activities. In comparison, DeepLog and Log2Vec achieve F_1-scores of only 0.35 and 0.62, respectively—indicating that approaches based on recurrent neural networks and heuristic techniques are limited in capturing the extended dependencies required for effective APT detection. Although MAGIC, ThreaTrace, and FLASH also perform well on CADETS (E3), with F_1-scores of 0.97, 0.95, and 0.96, respectively, GNNFormer's near-perfect Precision and Recall further highlight its superior detection efficacy. The newly added baselines—Slot, TAPAS, and STGAN—also perform strongly (F_1-scores = 0.96–0.99) but remain slightly below GNNFormer.

Table 5 Comparison between GNNFormer and state-of-the-art APT detection methods on the DARPA OpTC dataset. The best results are in bold.

Method	OpTC (Attack 1)			OpTC (Attack 2)			OpTC (Attack 3)		
	Precision	Recall	F_1-score	Precision	Recall	F_1-score	Precision	Recall	F_1-score
DeepLog	0.85	0.83	0.84	0.88	0.79	0.82	0.92	0.88	0.90
Log2vec	0.81	0.72	0.75	0.83	0.66	0.67	0.81	0.65	0.68
MAGIC	0.95	0.93	0.94	0.97	0.96	0.96	0.97	0.96	0.97
ThreaTrace	0.84	0.85	0.84	0.84	0.87	0.86	0.86	0.87	0.86
FLASH	0.90	0.91	0.91	0.94	0.92	0.93	0.91	0.92	0.92
Slot	0.95	0.91	0.93	0.97	0.94	0.95	0.96	0.93	0.94
TAPAS	0.97	0.96	0.97	0.97	0.96	0.96	0.97	0.96	0.97
STGAN	0.95	0.94	0.94	0.98	0.96	0.97	0.97	0.93	0.95
GNNFormer (Ours)	**0.98**	**0.99**	**0.98**	**1.00**	**1.00**	**1.00**	**0.99**	**0.99**	**0.99**

Table 6 Comparison of GNNFormer and baselines on the StreamSpot dataset.

Method	Precision	Recall	F_1-score
StreamSpot	0.73	0.91	0.81
Unicorn	0.95	0.97	0.96
FLASH	1.00	0.96	0.98
ThreaTrace	0.98	0.99	0.99
Prov2Vec	0.97	1.00	0.98
Slot	0.99	0.99	0.99
GNNFormer (Ours)	**1.00**	**1.00**	**1.00**

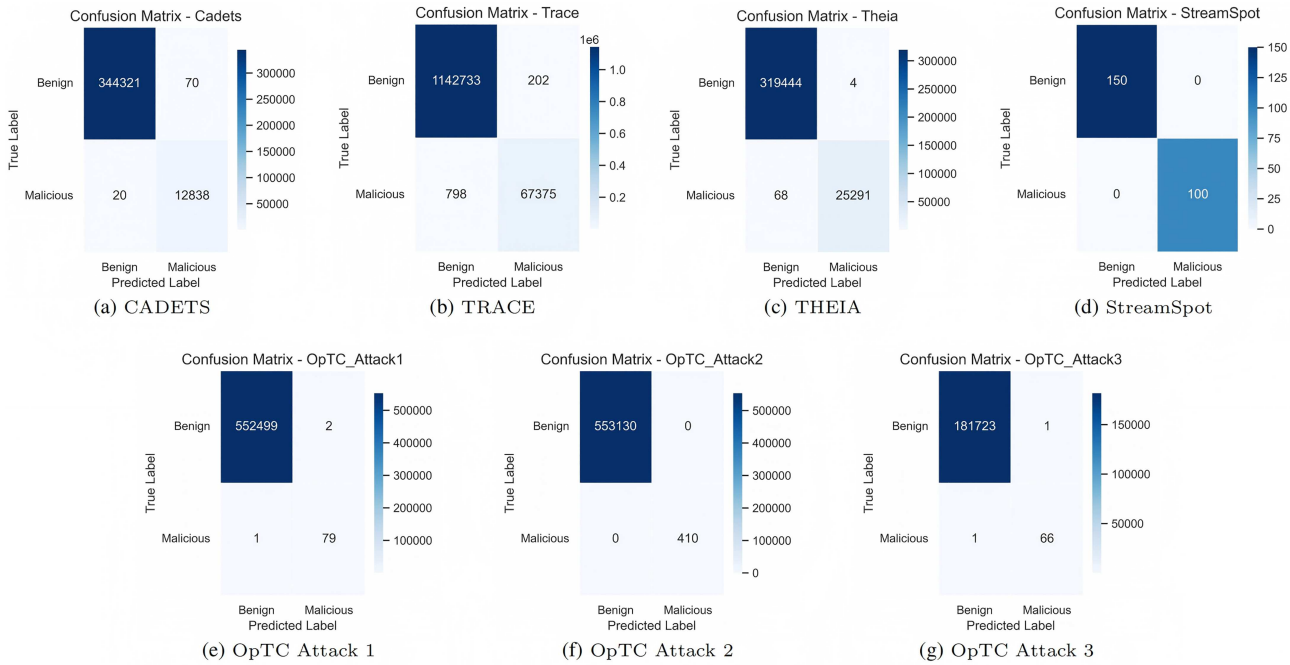
On the Trace (E3) dataset, GNNFormer maintains an F_1-score of 0.99, with Precision at 1.00 and Recall at 0.99. While MAGIC and FLASH also achieve high performance with F_1-scores of 0.99 and 0.96, respectively, and ThreaTrace records a slightly lower F_1-score of 0.83, GNNFormer's flawless Precision demonstrates its ability to reliably capture relevant patterns. Notably, unlike ThreaTrace—which employs a semi-supervised approach with the computationally intensive TransR module—GNNFormer delivers comparable or superior performance without incurring additional overhead. On this dataset, Slot, TAPAS, and STGAN are competitive, but their performance is slightly less consistent than GNNFormer's.

On the THEIA (E3) dataset, GNNFormer once again outperforms competing models, achieving perfect scores (Precision, Recall, and F_1-score all equal to 1.00). This performance exceeds that of MAGIC, ThreaTrace, and FLASH, which achieve F_1-scores of 0.99, 0.93, and 0.96, respectively. GNNFormer's ability to capture both structural and contextual dependencies within provenance graphs allows it to provide a comprehensive representation of system behavior, thereby more effectively distinguishing between benign and malicious interactions in complex, layered environments. Although Slot, TAPAS, and STGAN achieve strong F_1-scores (0.97–0.99), they still fall short of GNNFormer's perfect scores.

We further evaluate GNNFormer on the DARPA OpTC dataset, which provides node-level provenance data from three distinct attack scenarios. As shown in Table 5, GNNFormer outperforms all baselines across Attack 1, Attack 2, and Attack 3. For example, whereas TAPAS, STGAN, and Slot achieve F_1-scores of 0.94–0.97, GNNFormer attains up to 0.99, indicating better adaptability to larger-scale data and diverse attack strategies. These results show that the proposed design generalizes beyond DARPA TC E3 while maintaining high detection accuracy.

To further assess generalizability, we also evaluate GNNFormer on the StreamSpot dataset, which targets graph-level intrusion detection. Unlike the DARPA datasets that focus on node-level anomalies, StreamSpot requires classifying entire graphs as benign or malicious. To enable this comparison, we extend GNNFormer to the graph level by aggregating node embeddings into graph representations. As shown in Table 6, GNNFormer achieves Precision, Recall, and F_1-score of 1.00, outperforming strong baselines including StreamSpot, Unicorn, FLASH, ThreaTrace, Prov2Vec, and Slot. These results indicate that GNNFormer generalizes beyond node-level detection, validating the effectiveness of its end-to-end hierarchical representation learning for both localized and global intrusion patterns.

The success of GNNFormer across DARPA TC E3, OpTC, and StreamSpot datasets can be attributed to two key innovations. First, its comprehensive representation learning leverages both structural and contextual information within provenance graphs, enabling the capture of localized interactions and long-term global dependencies that are essential for identifying subtle APT behaviors. By combining layers optimized for both immediate and extended relationships, GNNFormer effectively distinguishes between normal and anomalous behaviors—a capability that



(a) CADETS (b) TRACE (c) THEIA (d) StreamSpot

(e) OpTC Attack 1 (f) OpTC Attack 2 (g) OpTC Attack 3

Figure 3 (Color online) Confusion matrices across seven datasets. The first row shows CADETS, TRACE, THEIA, and StreamSpot. The second row presents OpTC Attacks 1–3.

models such as Log2Vec and DeepLog lack due to their limited architectures. Second, GNNFormer incorporates a hierarchical attention mechanism that prioritizes critical interactions, thereby focusing computational resources on high-risk areas of the graph while filtering out less relevant data. This selective attention enhances its ability to detect rare and subtle APT patterns, especially in challenging scenarios where malicious behaviors closely resemble normal activities. In contrast, models that rely solely on feature sequence analysis may struggle to achieve this level of focused detection, as they do not incorporate mechanisms to refine attention based on interaction importance. Including the Slot, TAPAS, STGAN, and StreamSpot baselines further indicates that recent advances in graph-based detection still do not close the gap created by GNNFormer's end-to-end hierarchical representation learning.

4.6 Error analysis and explainability

To provide a deeper understanding of the performance and decision-making logic of GNNFormer, we present a visual analysis focusing on both error distribution and subgraph attribution.

4.6.1 *Error analysis visualization*

Figure 3 presents the confusion matrices across all evaluated datasets. A critical requirement for APT detection systems is minimizing false positives (FPs) to avoid overwhelming security analysts with unnecessary alerts. As shown in the figure, GNNFormer exhibits strong robustness in this regard.

In high-fidelity scenarios such as the OpTC datasets (Attacks 1, 2, and 3) and StreamSpot, the model achieves near-perfect classification results. In particular, for OpTC_Attack2 and StreamSpot, the model produces zero false positives and zero false negatives, indicating that the learned graph representations are highly discriminative and effectively separate malicious events from benign background activities.

For large-scale datasets such as THEIA and Trace, GNNFormer maintains high precision even under massive graph sizes. In the THEIA evaluation set containing approximately 340k nodes, only four benign nodes were misclassified as malicious, resulting in an exceptionally low false positive rate of 0.0012%. This demonstrates that the framework remains stable and accurate even in noisy and highly complex environments.

Despite the severe class imbalance in the CADETS dataset, the model successfully detects the majority of malicious nodes (12838) with only 20 false negatives. This shows that GNNFormer is capable of handling skewed class distributions without overfitting to the dominant benign class.

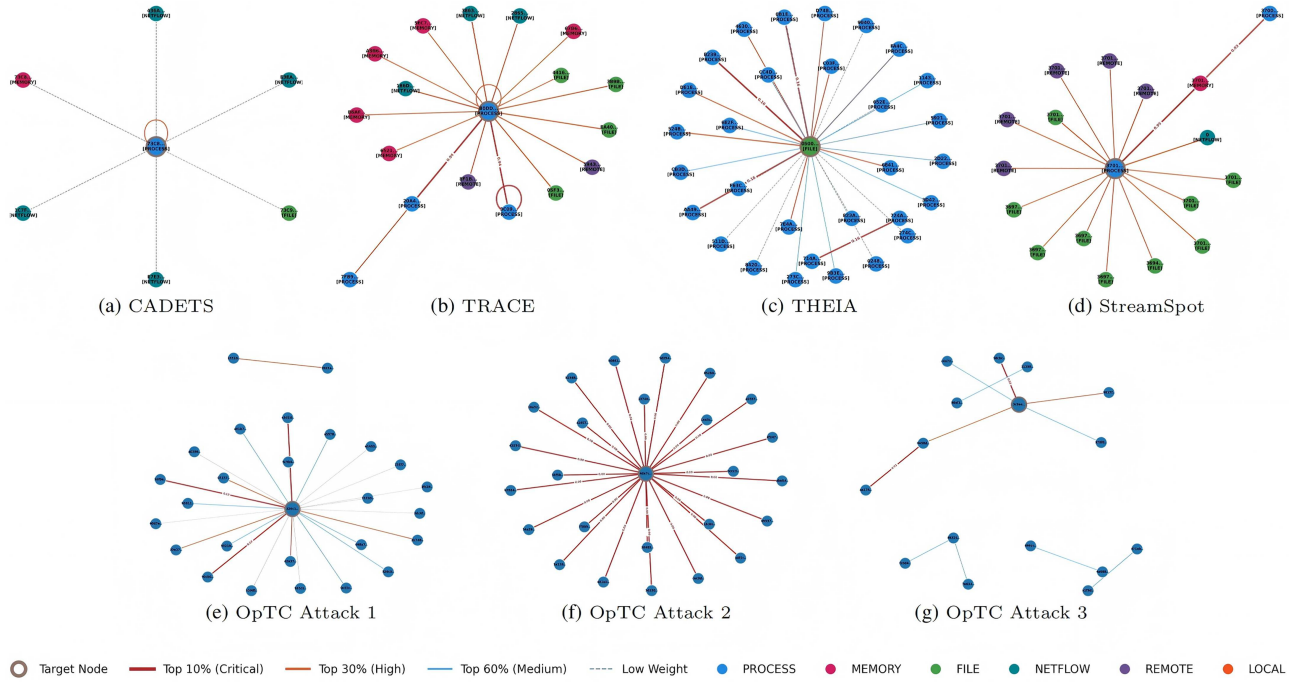


Figure 4 (Color online) GNNExplainer visualizations across seven datasets. The first row shows CADETS, TRACE, THEIA, and StreamSpot. The second row presents OpTC Attacks 1–3. The bottom strip displays the legend for edge importance and target-node highlighting.

4.6.2 *Explainability via subgraph attribution*

To verify that GNNFormer bases its predictions on semantically meaningful attack behaviors rather than spurious patterns, we adopt GNNExplainer to generate minimal subgraphs that maximize the mutual information with model outputs.

Figure 4 illustrates the explanatory subgraphs for representative malicious behaviors. Edges are color-coded according to their contribution importance: red edges correspond to the top 10% most influential dependencies, orange edges represent the top 30%, and dashed grey edges denote low-impact interactions pruned by the model during decision-making.

On datasets such as OpTC, StreamSpot, and CADETS, the explanatory results highlight process-to-process propagation chains as the dominant indicators of malicious activities. These extracted subgraphs precisely reveal parent-child process relationships associated with privilege escalation, injection, and lateral movement, confirming that the model has successfully learned the semantic structure of attack lineage rather than relying on superficial artifacts.

For resource-centric attacks appearing in THEIA and Trace, the explainer exposes star-shaped structures centered on critical file nodes, where unauthorized access, encryption, or rapid I/O bursts occur. In Trace, the influential subgraphs further span heterogeneous entity types—including processes, network flows, and memory-related events—demonstrating the model's ability to integrate multi-entity contextual cues to identify sophisticated APT techniques.

4.7 Ablation study

To comprehensively assess the contribution of each component in GNNFormer, we conducted ablation experiments on three datasets from the DARPA TC E3 suite (CADETS, Trace, and THEIA). Our objective was to evaluate the impact of removing specific components—namely, global dependency modeling, local interaction modeling, and the hierarchical attention mechanism—on the overall performance measured in terms of Precision, Recall, and F_1-score. The results in Table 7 indicate that all components contribute, though their impact varies across datasets, reflecting the diversity of APT behaviors.

4.7.1 *Effect of removing global dependency modeling*

Global dependency modeling is responsible for capturing long-term dependencies across the provenance graph. Excluding this component led to noticeable performance degradation. On CADETS (E3), the F_1-score dropped

Table 7 Ablation study: impact of different components on GNNFormer's performance across datasets. The best results are in bold.

Component	Cadets (E3)			Trace (E3)			Theia (E3)		
	Precision	Recall	F_1-score	Precision	Recall	F_1-score	Precision	Recall	F_1-score
Full model (GNNFormer)	**0.99**	**1.00**	**1.00**	**1.00**	**0.99**	**0.99**	**1.00**	**1.00**	**1.00**
Without global dependency modeling	0.95	0.97	0.96	0.94	0.96	0.95	0.96	0.97	0.97
Without local interaction modeling	0.93	0.95	0.94	0.92	0.93	0.92	0.93	0.94	0.94
Without hierarchical attention mechanism	0.97	0.98	0.98	0.98	0.98	0.98	0.98	0.99	0.98

from 1.00 to 0.96, while on Trace (E3) it decreased from 0.99 to 0.95, and on THEIA (E3) from 1.00 to 0.97. The impact is most evident in THEIA, where distributed, long-range correlations are essential for exposing multi-stage attack chains. These findings indicate that global context is important for identifying coordinated patterns that may span multiple nodes and time intervals.

4.7.2 *Effect of removing local interaction modeling*

Local interaction modeling focuses on capturing immediate relationships between nodes, which is crucial for detecting fine-grained anomalies. Removing this component consistently lowered performance: the F_1-score on CADETS (E3) dropped from 1.00 to 0.94, on Trace (E3) from 0.99 to 0.92, and on THEIA (E3) from 1.00 to 0.94. The degradation is most pronounced in CADETS, where many attack behaviors hinge on fine-grained process-file interactions. This suggests that local dependency modeling provides indispensable detail for distinguishing subtle deviations in node-level behavior.

4.7.3 *Effect of removing hierarchical attention mechanism*

The hierarchical attention mechanism refines the integration of local and global information by prioritizing high-risk nodes and interactions. Its removal caused a modest but consistent performance drop: the F_1-score decreased by about 0.02 across all datasets. Although less severe than removing local or global modeling, hierarchical attention improves detection by down-weighting irrelevant activities and highlighting interactions most indicative of malicious behavior. This shows that attention acts as a complementary refinement, enhancing precision without adding heavy computational overhead.

Overall, the ablations show complementary roles: global dependency modeling is critical for distributed attack patterns; local interaction modeling is indispensable for fine-grained anomalies; and hierarchical attention strengthens detection in noisy environments. Integrating all three within a unified framework is necessary to achieve the strong, balanced performance of the full GNNFormer model.

5 Conclusion

In this paper, we proposed GNNFormer, an APT detection framework based on provenance graph analysis. Our approach constructs provenance graphs from system log data to capture both local interactions and long-term global dependencies in system behavior. By embedding hierarchical attention directly within the layered representation process, the framework adaptively emphasizes critical interactions while down-weighting irrelevant activity, ensuring that the learned representations remain task-specific and resilient in noisy environments. This unified design enables GNNFormer to effectively detect subtle, multi-stage attack patterns within complex provenance graphs.

During training, GNNFormer is optimized using the Adam algorithm to minimize classification loss, refining its ability to distinguish between normal and anomalous sequences. In the detection phase, the model leverages the learned representations to isolate suspicious patterns through anomaly scoring and confidence-based filtering, thereby focusing on high-risk nodes within the graph. Extensive experiments on the DARPA TC E3 datasets confirm that GNNFormer consistently achieves state-of-the-art performance. Furthermore, evaluation on the StreamSpot dataset shows that the framework generalizes beyond node-level detection to graph-level intrusion classification, maintaining strong results across different detection paradigms. By incorporating recent baselines such as Slot, TAPAS, and STGAN into our comparisons, we also demonstrate that GNNFormer remains competitive with—and often surpasses—the latest provenance-based approaches, validating the advantages of its unified end-to-end architecture.

Despite these promising results, we acknowledge several limitations that highlight directions for future research. Although consistency across DARPA sub-datasets and StreamSpot suggests stability, broader evaluation on more

diverse and noisy datasets is needed to fully assess generalizability. Beyond dataset diversity, we plan to investigate adaptive model updating strategies that integrate advanced learning mechanisms to strengthen resilience against adversarial poisoning. Finally, we will explore more sophisticated decision processes beyond confidence-based thresholding to capture complex behavioral distributions more effectively. These improvements will further enhance GNNFormer's adaptability across both node-level and graph-level intrusion detection tasks in dynamic cyber environments.

Acknowledgements This work was supported in part by National Natural Science Foundation of China (Grant Nos. 62202042, 6212780016, 624B2027), State Key Laboratory of Internet Architecture, Tsinghua University (Grant No. HLW2025MS03), Fundamental Research Funds for the Central Universities (Grant No. 2024JBMC031), Aeronautical Science Foundation of China (Grant No. ASFC-2024Z0710M5002), Open Fund of the Key Laboratory of Advanced Cryptography and System Security of Sichuan Province (Grant No. SKLACSS-202312), Guangdong Science and Technology Programme (Grant No. 2024B0101030002), Ministry of Industry and Information Technology of China, National Key Research and Development Program of China (Grant No. 2023YFB3307500), Science and Technology Innovation Project of Hunan Province (Grant No. 2023RC4014), Open Project of the Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Qilu University of Technology (Shandong Academy of Sciences) (Grant No. 2024PY029), and Fundamental Research Funds for the Central Universities, Jilin University.

References

1 Ussath M, Jaeger D, Cheng F, et al. Advanced persistent threats: behind the scenes. In: Proceedings of 2016 Annual Conference on Information Science and Systems (CISS), 2016. 181–186
2 Zhang B, Gao Y, Kuang B, et al. A survey on advanced persistent threat detection: a unified framework, challenges, and countermeasures. ACM Comput Surv, 2025, 57: 1–36
3 Alshamrani A, Myneni S, Chowdhary A, et al. A survey on advanced persistent threats: techniques, solutions, challenges, and research opportunities. IEEE Commun Surv Tutorials, 2019, 21: 1851–1877
4 Che Mat N I, Jamil N, Yusoff Y, et al. A systematic literature review on advanced persistent threat behaviors and its detection strategy. J Cybersecurity, 2024, 10: tyad023
5 Kumar G S, Prakasha K K, Muniyal B. Ach reference model-a model of architecture to handle advanced cyberattacks. In: Proceedings of 2022 Second International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), 2022. 1–6
6 Adnan M, Bshara D, Awad A. Forensic analysis of APT attacks based on unsupervised machine learning. Euro J Sci Technol, 2023, doi: 10.31590/ejosat.1265586
7 Al-Yosef M Y, Arar N T. Survey on updating IDSs signatures databases. J Inform Secur Cyber Res, 2018, 1: 40–48
8 Lv Y, Qin S, Zhu Z, et al. A review of provenance graph based apt attack detection: applications and developments. In: Proceedings of 2022 7th IEEE International Conference on Data Science in Cyberspace (DSC), 2022. 498–505
9 Li Z, Chen Q A, Yang R, et al. Threat detection and investigation with system-level provenance graphs: a survey. Comput & Secur, 2021, 106: 102282
10 Hu E, Fu A, Zhang Z, et al. Actracker: a fast and efficient attack investigation method based on event causality. In: Proceedings of IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2021. 1–6
11 Hassan W U, Guo S, Li D, et al. Nodoze: combatting threat alert fatigue with automated provenance triage. In: Proceedings of Network and Distributed Systems Security Symposium, 2019
12 Xie Y, Feng D, Hu Y, et al. Pagoda: a hybrid approach to enable efficient real-time provenance based intrusion detection in big data environments. IEEE Trans Dependable Secure Comput, 2018, 17: 1283–1296
13 Li L, Chen W. ConGraph: advanced persistent threat detection method based on provenance graph combined with process context in cyber-physical system environment. Electronics, 2024, 13: 945
14 Milajerdi S M, Gjomemo R, Eshete B, et al. Holmes: real-time apt detection through correlation of suspicious information flows. In: Proceedings of 2019 IEEE Symposium on Security and Privacy (SP), 2019. 1137–1152
15 Hossain M N, Milajerdi S M, Wang J, et al. {SLEUTH}: real-time attack scenario reconstruction from {COTS} audit data. In: Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), 2017. 487–504
16 Aly A, Iqbal S, Youssef A, et al. MEGR-APT: a memory-efficient APT hunting system based on attack representation learning. IEEE Trans Inform Forensic Secur, 2024, 19: 5257–5271
17 Han X, Pasquier T F J, Bates A, et al. Unicorn: runtime provenance-based detector for advanced persistent threats. In: Proceedings of the 27th Annual Network and Distributed System Security Symposium, San Diego, 2020
18 Wu J, Yin L, Guo Y. Cyber attacks prediction model based on bayesian network. In: Proceedings of 2012 IEEE 18th International Conference on Parallel and Distributed Systems, 2012. 730–731
19 Frigault M, Wang L. Measuring network security using bayesian network-based attack graphs. In: Proceedings of 2008 32nd Annual IEEE International Computer Software and Applications Conference, 2008. 698–703
20 Luo X, Liu C, Gou G P, et al. Identifying malicious traffic under concept drift based on intraclass consistency enhanced variational autoencoder. Sci China Inf Sci, 2024, 67: 182302
21 Wei W, Chen S J, Chen C, et al. HEN: a novel hybrid explainable neural network based framework for robust network intrusion detection. Sci China Inf Sci, 2024, 67: 170304
22 Hossain M N, Sheikhi S, Sekar R. Combating dependence explosion in forensic analysis using alternative tag propagation semantics. In: Proceedings of 2020 IEEE Symposium on Security and Privacy (SP), 2020. 1139–1155
23 Milajerdi S M, Eshete B, Gjomemo R, et al. Poirot: aligning attack behavior with kernel audit records for cyber threat hunting. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019. 1795–1812
24 Cai W Y, Li M F, Shi X H, et al. RE-SEGNN: recurrent semantic evidence-aware graph neural network for temporal knowledge graph forecasting. Sci China Inf Sci, 2025, 68: 122104
25 Xia R T, Zhang C X, Zhang Y, et al. A novel graph oversampling framework for node classification in class-imbalanced graphs. Sci China Inf Sci, 2024, 67: 162101
26 Xi Z H, Chen W X, Guo X, et al. The rise and potential of large language model based agents: a survey. Sci China Inf Sci, 2025, 68: 121101
27 Ghiëtte V, Blenn N, Doerr C. Remote identification of port scan toolchains. In: Proceedings of 2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2016. 1–5
28 Rong X. word2vec parameter learning explained. 2014. ArXiv:1411.2738
29 Levy O, Goldberg Y. Neural word embedding as implicit matrix factorization. In: Proceedings of Advances in Neural Information Processing Systems, 2014
30 Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality. In: Proceedings of Advances in Neural Information Processing Systems, 2013

31 Wu Z, Jain P, Wright M, et al. Representing long-range context for graph neural networks with global attention. In: Proceedings of Advances in Neural Information Processing Systems, 2021. 34: 13266–13279
32 Hayashi H. A hybrid of generative and discriminative models based on the Gaussian-coupled softmax layer. IEEE Trans Neural Netw Learn Syst, 2025, 36: 2894–2904
33 Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection. In: Proceedings of 2017 IEEE International Conference on Computer Vision (ICCV), 2017. 2999–3007
34 Vasudevan V T, Sethy A, Ghias A R. Towards better confidence estimation for neural models. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019. 7335–7339
35 Manzoor E, Milajerdi S M, Akoglu L. Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. 1035–1044
36 Du M, Li F, Zheng G, et al. Deeplog: anomaly detection and diagnosis from system logs through deep learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017. 1285–1298
37 Liu F, Wen Y, Zhang D, et al. Log2vec: a heterogeneous graph embedding based approach for detecting cyber threats within enterprise. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019. 1777–1794
38 Jia Z, Xiong Y, Nan Y, et al. {MAGIC}: detecting advanced persistent threats via masked graph representation learning. In: Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24), 2024. 5197–5214
39 Wang S, Wang Z, Zhou T, et al. THREATRACE: detecting and tracing host-based threats in node level through provenance graph learning. IEEE Trans Inform Forensic Secur, 2022, 17: 3972–3987
40 Rehman M U, Ahmadi H, Hassan W U. Flash: a comprehensive approach to intrusion detection via provenance graph representation learning. In: Proceedings of 2024 IEEE Symposium on Security and Privacy (SP), 2024. 139–139
41 Qiao W, Feng Y, Li T, et al. Slot: provenance-driven apt detection through graph reinforcement learning. In: Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, 2025. 963–977
42 Zhang B, Gao Y, Yu C, et al. Tapas: an efficient online apt detection with task-guided process provenance graph segmentation and analysis. In: Proceedings of the 34th USENIX Conference on Security Symposium, 2025
43 Sang A, Fan X, Yang L, et al. Stgan: detecting host threats via fusion of spatial-temporal features in host provenance graphs. In: Proceedings of the ACM on Web Conference 2025, 2025. 1046–1057
44 Bhattarai B, Huang H H. Prov2vec: learning provenance graph representation for anomaly detection in computer systems. In: Proceedings of the 19th International Conference on Availability, Reliability and Security, 2024. 1–14