Hybrid Time-Position Embedding for Provenance-Based Intrusion Detection

Seonghyeon Gong , Jake Cho * , Kyuwon Ken Choi

Posted Date: 19 November 2025

doi: 10.20944/preprints202511.1455.v1

Keywords: data provenance; intrusion detection; embedding; semantic information

 doi:10.20944/preprints202511.1455.v1

Article

Hybrid Time-Position Embedding for Provenance-Based Intrusion Detection

Seonghyeon Gong [1], Jake Cho [2,*] and Kyuwon Ken Choi [1]

[1] Illinois Institute of Technology
[2] Lewis University
* Correspondence: jcho1@lewisu.edu

Abstract

Provenance-based Intrusion Detection Systems (IDS) model the causal relationships between security events through a provenance graph and learn contextual information to detect Advanced Persistent Threats (APTs) effectively. However, existing provenance graph representation methods fail to fully reflect the characteristics of security domain data and the semantic information embedded in system logs, resulting in limitations in learning efficiency and detection accuracy. This paper proposes a provenance representation method that effectively captures security context from system log data. The proposed method improves the performance of provenance-based IDS by combining (1) a provenance graph construction technique that transforms meaningful string attributes—such as command lines, process names, and file paths—into vector representations to extract semantic information in the security context, (2) a hybrid time-position embedding technique for capturing causal relationships between events, and (3) an iterative refinement learning strategy tailored to the characteristics of system log data. Experimental results using the DARPA Transparent Computing Engagement 3 (E3) benchmark dataset for APT detection demonstrate that our method achieves improved accuracy compared to existing approaches while significantly accelerating convergence during iterative training. These results suggest that the proposed embedding technique can more effectively capture abnormal temporal patterns, such as long dwell times characteristic of APT attacks.

Keywords: data provenance; intrusion detection; embedding; semantic information

1. Introduction

As information technology advances and networks become increasingly complex, the scale and impact of cyberattacks continue to grow. As demonstrated by high-profile incidents such as Stuxnet [1], the Sony Pictures hack [2], NotPetya [3], and the SolarWinds breach [4], APTs can result in catastrophic consequences for nations, societies, and enterprises [5]. IDS are essential for responding to such incidents; however, they face significant limitations in detecting and responding to sophisticated attacks like APTs [6]. Traditional rule-based and signature-based IDS effectively identify known, isolated attack patterns. However, these systems fail to detect novel or stealthy APT attacks that exploit zero-day vulnerabilities or use evasion techniques [7]. Moreover, their lack of behavioral analysis makes detecting complex, multi-stage attack sequences difficult [8]. To overcome the limitations of traditional IDS approaches, many studies have explored machine learning-based IDS (ML-IDS), which has shown some success in detecting APTs, including zero-day attacks [9].

Nevertheless, ML-IDSs also have limitations: they struggle to capture causal relationships among security events, which is crucial for detecting the staged behaviors of APTs that often involve long, stealthy periods [10]. Furthermore, their performance heavily depends on the quality of the training dataset and the domain expertise applied during feature engineering. These challenges hinder the development of generalizable detection models and can lead to high false-positive rates and overfitting [11].

 doi:10.20944/preprints202511.1455.v1

To address the challenges posed by highly structured and stealthy APT attacks, data provenance-based approaches have recently gained attention. Data provenance represents the contextual and causal relationships between system events using a directed acyclic graph (DAG) structure [12]. This representation enables the modeling of complex system behaviors and supports learning attack patterns, making it suitable for detecting APTs. As a result, a growing body of research has explored provenance-based IDS to detect and classify sophisticated attacks.

However, provenance-based IDS still faces several limitations: (1) Neglect of Semantic Information: Security event logs often include not only universally unique identifiers (UUIDs) for processes and network sockets but also rich semantic attributes such as executable commands, timestamps, and file paths [13,14]. These attributes may contain important contextual clues depending on the attack scenario [15]. Nevertheless, many IDS approaches disregard such information due to challenges in generalization and representation. (2) Difficulty in Inferring Temporal Causality: Provenance graphs typically establish edges based on the order of event occurrence. However, this approach overlooks the actual time intervals between events, limiting the system's ability to infer precise causal relationships [16]. For instance, APT attacks often exploit long dwell times and mimic normal behavior between lateral movements. Detection strategies that rely solely on event order may fail to capture such attack patterns [17].

In this research, we propose a novel provenance graph generation method that addresses the limitation of semantic information neglect in provenance-based IDS by leveraging a hybrid time-position embedding mechanism. Our method parses system logs to extract subject-type-object triples and accordingly constructs provenance nodes and edges. We enrich node attributes by aggregating contextual information derived from each subject's behavior.

Furthermore, we incorporate time information into the embedding process to support accurate causal inference. Rather than encoding only the procedural order of events, our method also embeds the actual time intervals between them. This enables the model to reason about temporal relationships and detect abnormalities in timing patterns, which is especially effective for capturing APT behaviors characterized by long periods of inactivity.

Our approach employs a graph neural network (GNN) [18] to learn from the constructed provenance graph. To improve learning efficiency, we further refine the training data through an iterative mechanism that considers the unique characteristics of security datasets.

The main contributions of this study are as follows:

- We propose a novel provenance representation method using a hybrid time-position embedding mechanism. This method generates provenance graphs that better express system behavior by embedding procedural order and time intervals from system logs. It can be broadly applied to the embedding process of raw security data and contributes to improving dataset quality, particularly during preprocessing and dataset construction.
- We demonstrate that models trained on provenance graphs generated using our log embedding technique can more effectively capture the semantic information in system logs. Through experiments on the DARPA E3 benchmark dataset, we show that the GNN model trained on our proposed representation converges more quickly than baseline methods, indicating the effectiveness of our embedding in expressing node attributes in the provenance graph.

The rest of the paper is organized as follows—section 2 overviews existing ML-based and provenance-based IDS methods for detecting APTs. Section 3 details the mechanism and algorithm of the proposed time-aware provenance-based IDS. In Section 4, we present the experimental results on the DARPA E3 benchmark dataset to demonstrate the effectiveness of our approach. Finally, Section 5 concludes the paper.

2. Related Work

This section reviews existing approaches to provenance-based intrusion detection systems (IDSes) and the efforts made to overcome their limitations. We mainly focus on provenance graph embedding

 doi:10.20944/preprints202511.1455.v1

techniques aimed at encoding sufficient semantic information from the perspective of IDS and recent studies that attempt to embed semantic attributes into provenance representations.

2.1. Provenance- and Causality-based Anomaly Detection Approaches

ProvDetector [19] leverages Doc2Vec [20] representations to model process-level operating system logs and generate provenance graphs for detecting stealthy malware. TRACE [21] constructs provenance and causal graphs by tracing execution units induced by loop structures in program behavior, targeting real-time APT detection. ATLAS [13] identifies abstract attack strategies commonly shared among APT campaigns and builds a sequence-based model using audit logs and causality graphs to propose a generalized detection strategy. ThreaTrace [22] presents a node-level anomaly detection method based on host system logs. Unlike graph-level approaches, it focuses on fine-grained detection by classifying anomalies at the individual node level. The framework uses a multi-model architecture to enhance classification performance and applies GraphSAGE [23] to analyze the causal dependencies between nodes. However, it does not incorporate semantic information within the embedding process.

PalanTír [24] addresses the dependency explosion problem in traditional log-based detection by analyzing causality at the instruction level within system call traces. Although this approach enables detailed tracking of attack stages at the command level, it still lacks explicit consideration of semantic information. Paradise [25] proposes a scalable, distributed provenance-based IDS capable of real-time analysis of large-scale logs. It integrates a Kafka-based infrastructure [15] to support high-throughput log ingestion and real-time intrusion detection. DEPCOMM [26] introduces a method for graph summarization by identifying process-centric communities from causal dependency graphs and pruning redundant edges caused by repetitive benign behaviors. This method reduces subgraph size while improving detection performance. Angus [27] proposes an active learning framework for provenance-based IDS to enhance learning efficiency, even in scenarios with extremely limited labeled data. The study demonstrates that high detection performance can still be achieved under label-scarce conditions.

While these studies have improved provenance-based IDS performance in terms of scalability, learning efficiency, and detection accuracy, they largely overlook the unique characteristics of security domain data. In particular, few approaches focus on mechanisms to extract and encode rich semantic and contextual information embedded within raw system logs into the provenance representation.

2.2. Semantic Representation for IDS

Attack2Vec [28] proposed a temporal word embedding technique to capture semantic information across different stages of sophisticated attacks. This approach treats each event as a word and sequences of events as sentences, enabling the modeling of vector representations along the temporal axis. While this work provides a foundation for handling sequences of events, it does not capture causal relationships between attack events. Watson [29] introduced a semantic inference and behavior abstraction mechanism to automate the extraction of high-level activity information. By clustering inferred behaviors, the system identifies a small number of representative audit logs from large-scale security data, thereby facilitating more efficient analysis. DEEPCASE [30] presented a semi-supervised approach for learning contextual representations and identifying correlations among security events. This method does not rely heavily on labeled data and achieves high detection accuracy through contextual embeddings based on surrounding events. FLASH [31] proposed a provenance graph representation and learning framework for APT detection by combining Word2Vec-based semantic encoding with GNN-based structural encoding. To overcome the limitations of Word2Vec in ignoring sequential information, FLASH incorporates Transformer-style positional embedding to enhance APT detection performance. Most existing efforts to extract semantic information from security data focus on log embedding techniques. However, due to the nature of APTs—such as long dwell times and anomalous execution delays—more sophisticated causal reasoning over security data nodes is necessary.

 doi:10.20944/preprints202511.1455.v1

2.3. Temporal and Positional Encoding for Provenance Graphs

Incorporating temporal and positional information into semantic and contextual representations is critical for accurately modeling APT characteristics in the security domain. UNICORN [32] models long-term behavior through graph sketching techniques to detect slow-acting APT attacks. While this approach effectively summarizes entire event flows and improves detection, it still leaves room for improvement in accurately modeling event causality based on timestamp data. PSSID [14] addresses evasive behavior in provenance-based IDSes by using the longest common subsequence (LCS) metric to measure similarity between provenance sequences and rule-based reference sequences. Although this method is robust against reordered execution and noise injection, it lacks semantic and temporal encoding capabilities. ProGrapher [33] combines time-series-based snapshot generation with whole-graph embedding to detect APTs. By incorporating event ordering, it provides temporal awareness and generates key indicators to assist analysts, thereby enhancing the practical utility of the system. PG-AID [34] generates provenance graphs using meta-path Doc2Vec [20] embeddings and Transformer-based attention mechanisms. It leverages attention vectors to introduce positional encoding into provenance representations. While many prior works incorporate positional embeddings into security context modeling, relying solely on node order in provenance graphs is insufficient for fully capturing the semantic characteristics of APTs.

3. Method

In this section, we present our hybrid time-position embedding method designed to improve the learning efficiency of semantic information in provenance-based intrusion detection systems and enhance their detection performance against APTs. As illustrated in Figure 1, our approach consists of a five-stage pipeline. The Provenance Graph Parsing module takes system log data as input and generates a provenance graph that incorporates a wide range of semantic information relevant to the security domain. The Node Attribute Embedding and Hybrid Time-Position Embedding stages perform vectorization of this semantic information, embedding contextual and temporal characteristics into node representations. The resulting embedded graph data is then used to train a GNN, which learns both structural and semantic patterns within the graph. Finally, in the Attack Detection phase, the trained GNN performs node-level anomaly detection to identify potential attack behaviors.

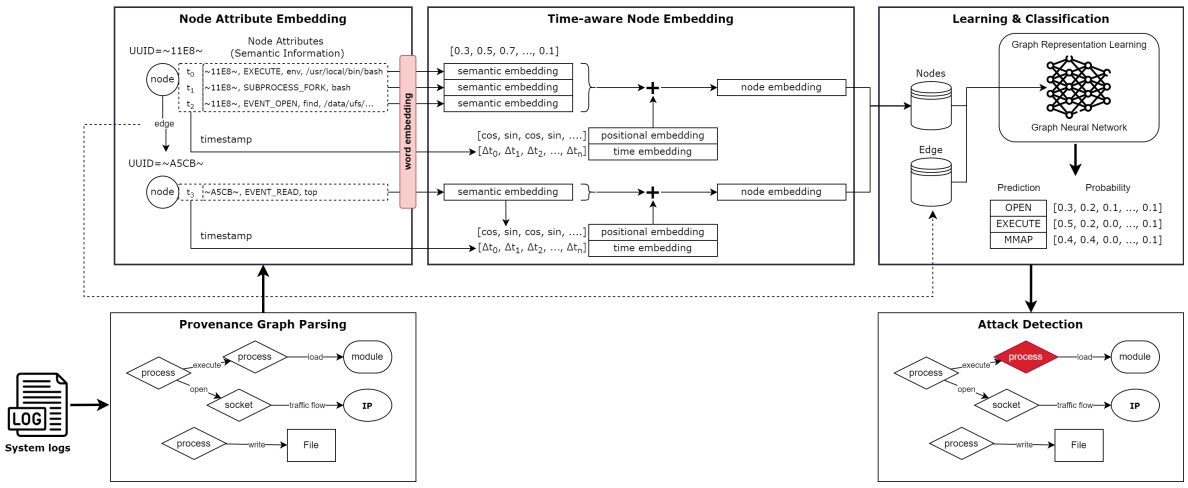

Figure 1. Overall architecture of the proposed provenance-based intrusion detection system

3.1. Provenance Graph Parsing

Our method constructs a provenance graph from system log data, including Windows Event Logs and Linux system audit logs. These logs are time-ordered by their timestamps and contain critical fields such as universal unique identifiers (UUIDs), event types, actions performed, and related file or process paths. Inspired by prior work on provenance-based IDS [13,31,33], we model each log entry as a subject-action-object triplet and use UUIDs to define the nodes in the provenance graph.

 doi:10.20944/preprints202511.1455.v1

Edges between nodes are determined by event types, while each node is enriched with semantic information extracted from the log entry, including event types, executed commands, process names, and file paths—attributes highly relevant to the security domain. At this stage, each node in the provenance graph is associated with a text sequence composed of meaningful tokens from the original system log, effectively forming a sentence representation for that node's attribute.

3.2. Semantic Information and Hybrid Time-Position Embedding

Identifiers such as UUIDs, process names, file names, and IP addresses are crucial indicators in security contexts. However, as these elements appear in string format, they must be converted into numerical vectors for machine learning-based analysis. Inspired by prior log embedding approaches such as Log2Vec [35] and FLASH [31], we apply the Word2Vec model [36] to embed the semantic attributes of each node into a dense vector space.

In our method, if multiple events are associated with a single node (e.g., multiple edges connected to the same entity), we aggregate the corresponding log entries into a single sequence to represent the node's attribute. As a result, each node's attribute is represented by a variable-length set of log tokens, which is then embedded into a fixed-length vector using Word2Vec. While this approach effectively captures semantic attributes, it discards the sequential order and timing information of the log entries—information that is critical for detecting APTs.

To mitigate the loss of temporal context, we propose a hybrid time-position embedding technique. While several existing provenance embedding approaches use positional encoding to address the loss of sequential information [31,34], we argue that such methods are insufficient for capturing the behavior of stealthy attacks, particularly APTs. APTs are often characterized by seemingly benign actions and prolonged dwell times, which cannot be captured by sequence order alone.

To address this, we extend the standard positional embedding mechanism used in self-attention models [37] by incorporating a temporal embedding that reflects the actual time intervals between events. Specifically, for each node, we normalize the time gaps between individual log entries and the starting timestamp of the node. This temporal information is embedded as a supplementary vector and added to the semantic embedding output to form the final node representation.

This approach enables the model to distinguish patterns such as abnormal delays between process invocations or prolonged inactivity—both of which are typical indicators of APT behavior. By embedding such time-aware features, our method enhances the graph's capacity to model temporal causality and improves its effectiveness in detecting stealthy and persistent threats.

Algorithm 1 and Equation 1 illustrate the embedding process of our proposed method. Given a sequence of tokens associated with a node attribute, denoted as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, we first generate the initial word embeddings $\mathbf{x}_i \in \mathbb{R}^d$ using a pretrained Word2Vec model. Let t_i denote the timestamp corresponding to token $\mathbf{x}_i$. Our hybrid time-position embedding for each token is defined as:

$$\mathbf{E}_i = \mathbf{x}_i + \mathbf{p}_i + \mathbf{t}_i, \tag{1}$$

$\mathbf{p}_i$ is the standard sinusoidal positional embedding, defined as:

$$\mathbf{p}_i^{(2k)} = \sin\left(\frac{i}{10000^{2k/d}}\right), \quad \mathbf{p}_i^{(2k+1)} = \cos\left(\frac{i}{10000^{2k/d}}\right), \tag{2}$$

and $\mathbf{t}_i$ is the temporal embedding derived from the relative timestamp:

$$\mathbf{t}_i = W_t \cdot \Delta t_i, \tag{3}$$

where $\Delta t_i = t_i - t_1$ is the normalized time difference from the first event timestamp, and $W_t \in \mathbb{R}^{d\times 1}$ is a learnable linear projection matrix.

The complete time-position embedded sequence is represented as:

$$\mathbf{H} = [\mathbf{E}_1, \mathbf{E}_2, \ldots, \mathbf{E}_n] \in \mathbb{R}^{n\times d}. \tag{4}$$

 doi:10.20944/preprints202511.1455.v1

We apply mean pooling over the sequence to generate a fixed-size node embedding vector $\mathbf{z}_{node}$:

$$\mathbf{z}_{node} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{E}_i. \tag{5}$$

Algorithm 1 Hybrid Time-Position Embedding

Require: Word list $W = [w_1, w_2, \ldots, w_n]$, timestamp list $T = [t_1, t_2, \ldots, t_n]$, word2vec model $\mathcal{M}$, embedding dimension d
Ensure: Hybrid embedding vector $\mathbf{z}_{node} \in \mathbb{R}^d$

$\mathbf{P} \leftarrow \mathbb{R}^{n \times d}$ ▷ Initialize positional embedding matrix $\mathbf{P}$
$\Delta T \leftarrow [t_1 - t_1, t_2 - t_1, \ldots, t_n - t_1]$ ▷ Normalize timestamps
$\mathbf{X} \leftarrow [\mathcal{M}(w_1), \ldots, \mathcal{M}(w_n)]$ ▷ Convert words to embeddings
$\mathbf{T} \leftarrow \text{Linear}(\Delta T)$ ▷ Compute temporal embeddings
$\mathbf{E}_i \leftarrow \mathbf{X}_i + \mathbf{P}_i + \mathbf{T}_i \quad \forall i \in [1, n]$ ▷ Compute hybrid embeddings
$\mathbf{z}_{node} \leftarrow \frac{1}{n} \sum_{i=1}^{n} \mathbf{E}_i$ ▷ Aggregate by mean
$\mathbf{z}_{node}$

3.3. Graph Representation Learning

To detect abnormal behaviors and malicious nodes in the provenance graph, it is essential to learn structural representations that incorporate the contextual information of neighboring nodes. To this end, we adopt a graph embedding approach based on Graph Neural Networks (GNNs).

GNNs enable each node to update its embedding by aggregating information from its neighbors, effectively capturing the underlying structure of the graph. In our framework, we construct a 2-layer GNN model based on GraphSAGE [23], using the node feature vectors obtained from the hybrid time-position embedding as input. The node representation at layer k is computed as follows:

$$\mathbf{h}_v^{(k)} = \sigma\left(W^{(k)} \cdot \text{AGG}^{(k)}\left(\left\{\mathbf{h}_v^{(k-1)}\right\} \cup \left\{\mathbf{h}_u^{(k-1)}, \forall u \in \mathcal{N}(v)\right\}\right)\right) \tag{6}$$

where $\mathbf{h}_v^{(0)} = \mathbf{X}_v$ is the initial node embedding, σ is the ReLU activation function, $W^{(k)}$ is the trainable weight matrix, and $\text{AGG}^{(k)}$ is a mean-based neighborhood aggregation function. This architecture effectively reflects structural patterns in the provenance graph.

Attack detection using provenance graphs is computationally expensive due to the large-scale graph structure and the sparse nature of attack behaviors. Most system logs originate from repetitive benign activities, resulting in redundant and semantically neutral patterns in the provenance graph. Consequently, a naïve node-level classification approach using GNNs is not only inefficient in terms of computational cost but also ineffective in focusing on critical anomalous behaviors.

To address this issue, we adopt a confidence-based iterative refinement strategy inspired by FLASH [31]. After performing predictions on all nodes using the trained GNN, we compute the confidence score of each node based on the softmax probabilities of the top two classes:

$$\text{Confidence}(v) = \frac{\hat{y}_1 - \hat{y}_2}{\hat{y}_1} \tag{7}$$

where $\hat{y}_1$ and $\hat{y}_2$ are the highest and second-highest softmax scores, respectively. Nodes that are correctly predicted with high confidence are removed from the training pool, while the model continues training only on the remaining uncertain (potentially suspicious) nodes.

This iterative refinement strategy enables the model to focus on more challenging samples, enhancing detection precision while saving computational resources. Ultimately, this process allows for more accurate identification of suspicious nodes, which can be used to reconstruct the attack flow and support post-hoc analysis.

 doi:10.20944/preprints202511.1455.v1

3.4. Attack Detection Phase

We utilize a model trained on the provenance graph to detect abnormal behavior at the node level. One of the node attributes, 'type', serves as the classification label for the classifier. This approach is based on the assumption that behaviors not observed in the general category are considered abnormal. We employ multiple GNN models based on GraphSAGE and use an ensemble method for the final judgment. The node type with the highest probability value among the probability vectors derived from several models is considered the final judgment. If a node is misjudged by all GNN models, it is classified as a malicious node. This method allows for node-level detection, thereby reducing the analysis cost for analysts.

4. Experiment and Evaluation

In this section, we present the experimental setup and results to evaluate the effectiveness of the proposed Hybrid Time-Position Embedding. We designed our experiments to address the following two key research questions (RQs). All experiments were conducted on a machine with 48 Intel vCPUs, 256 GB of RAM, and Debian 12 OS.

- RQ1. Does Hybrid Time-Position Embedding improve node classification performance over traditional embedding methods?
- RQ2. Does Hybrid Time-Position Embedding achieve better convergence during iterative refinement than positional-only embedding?

4.1. Dataset

We utilize the E3 dataset released by DARPA as part of its Transparent Computing (TC) program. This dataset was developed during Engagement 3 (E3) to support research on real-time and forensic-based APT detection and traceability techniques. The E3 dataset contains fine-grained system event logs, which can be structured into provenance graphs representing interactions between system entities.

Crucially, the dataset includes both normal system operations and sophisticated APT scenarios executed by a red team, making it highly suitable for intrusion detection research. For our experiments, we evaluated our proposed method using data collected by four teams—Cadets, Trace, Theia, and FiveDirections—to ensure consistent benchmarking and comparative analysis with prior work.

4.2. Baselines

To evaluate the effectiveness of the proposed Hybrid Time-Position Embedding, we compare our method against two representative provenance-based intrusion detection systems: ThreaTrace [22] and FLASH [31].

- **ThreaTrace**: ThreaTrace identifies abnormal behavior by modeling the distributional patterns of system calls (syscall distributions). It characterizes process behavior through multi-model frameworks to maximize detection performance.
- **FLASH**: FLASH leverages Word2Vec-based semantic embeddings combined with positional encoding to construct meaningful node representations. It integrates a GNN with a lightweight classifier and applies iterative refinement to boost detection accuracy. FLASH also demonstrated superior scalability through its efficient architecture.

4.3. Implementation Details

Log sequences contained in node attributes are embedded into 30-dimensional vectors using Word2Vec. The proposed Hybrid Time-Position Embedding augments these vectors by incorporating sinusoidal positional encoding and a linear temporal embedding derived from timestamp intervals. We utilize a 2-layer GraphSAGE model [23] with 32 hidden units per layer. The training objective is based on cross-entropy loss, with class imbalance mitigated via class-weighted loss. Iterative refinement is

 doi:10.20944/preprints202511.1455.v1

performed over 22 epochs; after each round, nodes that are correctly predicted with high confidence are removed from the training set.

4.4. RQ1: Classification Performance by Embedding Strategy

Table 1 presents the classification results comparing different embedding strategies. All models were trained on the same dataset and evaluated using precision, recall, and F1-score. Our proposed embedding method outperforms existing approaches across all metrics. The improvement is particularly notable for nodes rich in semantic information (e.g., processes and network flows), indicating that incorporating timestamp intervals allows the model to capture both sequential and contextual aspects of the logs better.

Table 1. Comparison of detection performance (Precision, Recall, and F1-score) across different methods on the DARPA E3 dataset (Cadets, Trace, Theia, Fivedirections).

Dataset (E3)	ThreaTrace			Flash			Proposed		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Cadets	0.90	0.99	0.95	0.95	0.99	0.97	0.928	0.999	0.962
Trace	0.72	0.99	0.83	0.95	0.99	0.97	0.943	0.999	0.970
Theia	0.87	0.99	0.93	0.93	0.99	0.96	0.927	0.998	0.961
Fivedirections	0.67	0.92	0.78	0.70	0.93	0.80	0.736	0.941	0.826

4.5. RQ2: Convergence Properties in Iterative Refinement

This experiment investigates whether the proposed Hybrid Embedding enables better convergence behavior compared to positional-only embedding. Each model starts from the same initial GNN configuration, and the number of misclassified samples is tracked after each refinement round. This metric reflects how efficiently the model stabilizes during training.

As shown in Figure 2 and 3, the proposed Hybrid Time-Position Embedding demonstrates faster and more stable convergence compared to the baseline FLASH (which uses positional-only embeddings). Our method significantly reduces the number of misclassified nodes in the early rounds (especially within rounds 0 to 7), while the baseline shows a slower, more gradual decrease. Moreover, our model reaches convergence earlier and maintains a lower misclassification rate in later rounds. These results confirm that incorporating timestamp information improves model decision-making during iterative refinement and yields more favorable convergence behavior than simple positional encoding alone.

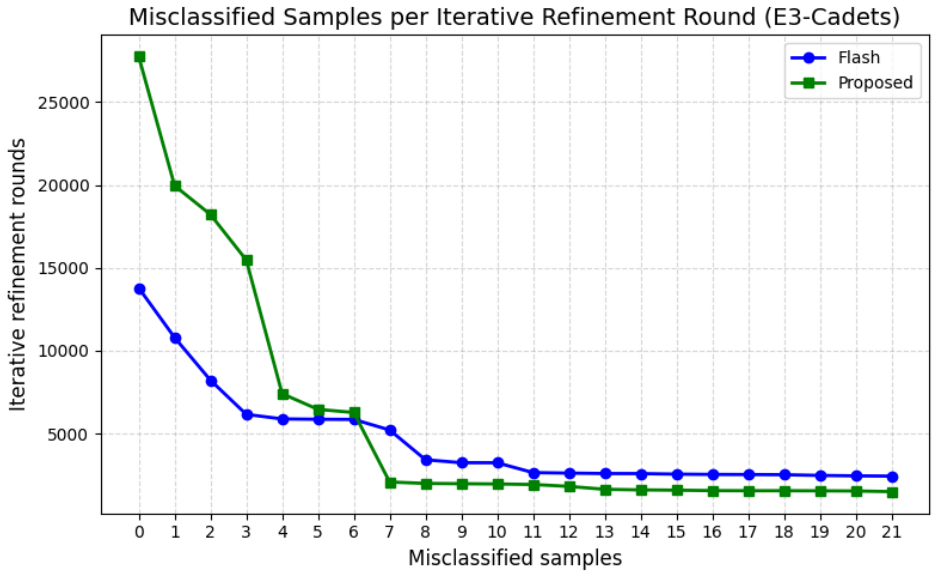

Figure 2. Convergence comparison of iterative refinement between baseline(Flash) and the proposed method on the Cadets dataset.

 doi:10.20944/preprints202511.1455.v1

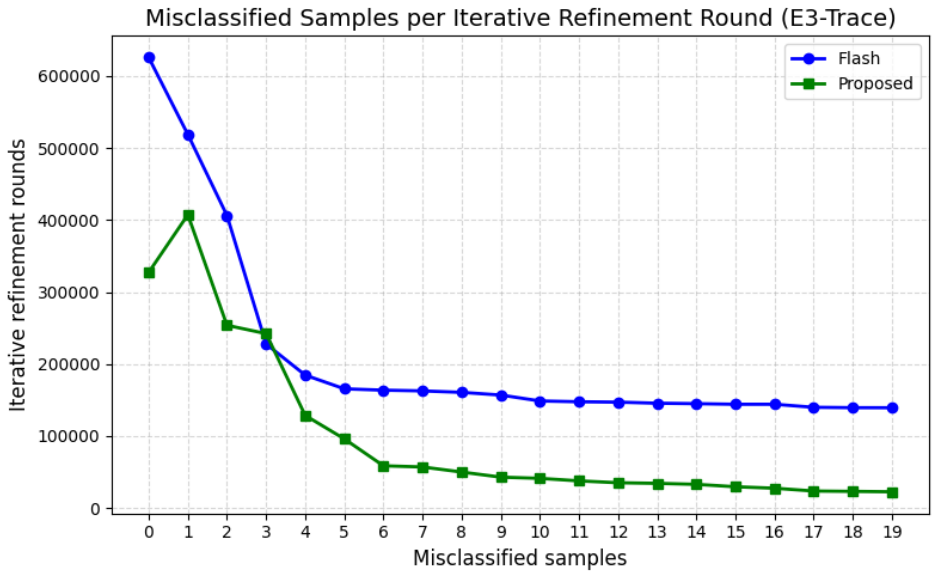

Figure 3. Convergence comparison of iterative refinement between baseline(Flash) and the proposed method on the Trace dataset.

4.6. Discussion

While the proposed Hybrid Time-Position Embedding method demonstrated slightly higher detection accuracy than the existing baseline approach, the precision, recall, and F1-score performance gain was marginal. In most datasets, the difference between the baseline and our method remained within 1–2%.

However, a more noteworthy outcome was observed during the model training process. Specifically, our method significantly outperformed the baseline regarding convergence speed during iterative refinement. As shown in Figure 2, the number of misclassified nodes rapidly decreased within our method's early refinement rounds (particularly rounds 0 to 7), whereas the baseline showed a slower and more gradual improvement. Moreover, our model achieved convergence in fewer epochs and maintained a stable misclassification rate thereafter. This indicates that the proposed embedding scheme provides more informative and structured representations for each node, allowing the GNN to learn discriminative features more efficiently.

This improvement in convergence behavior is particularly valuable in practical deployment scenarios. In provenance-based intrusion detection systems, reducing training time and computational overhead can directly contribute to better scalability and faster response. Since GNN training over large-scale provenance graphs can be computationally expensive, enhancing convergence efficiency without sacrificing detection accuracy presents a meaningful contribution.

5. Conclusions

In this paper, we proposed a novel Hybrid Time-Position Embedding technique to improve semantic representation in provenance-based intrusion detection systems. By integrating both positional encoding and relative timestamp intervals, our approach aims to capture the sequential order of events and their temporal gaps, which are critical in understanding stealthy behaviors like those found in APTs.

While our empirical results show only a slight improvement in detection accuracy compared to existing methods, we observe a significant enhancement in convergence speed during GNN training. This implies that our representation method enables the model to learn more effectively from limited iterations, reducing computational cost without degrading performance.

Our findings suggest that time-aware semantic embedding can be a practical design choice for provenance-based IDSs, particularly in environments requiring rapid model updates and efficient resource usage. Future work will explore additional temporal modeling strategies, such as attention-

 doi:10.20944/preprints202511.1455.v1

based time series embeddings, and evaluate our method's robustness against advanced evasion techniques and real-world APT datasets.

Author Contributions: Conceptualization, S.G. and J.C.; methodology, S.G.; oftware, S.G.; validation, S.G.; formal analysis, S.G.; investigation, S.G.; resources, S.G. and J.C.; data curation, S.G.; writing—original draft preparation, S.G.; writing—review and editing, J.C.; visualization, S.G.; supervision, J.C. and K.C.; project administration, J.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this study are publicly available as part of the DARPA Transparent Computing (TC) program. Specifically, the Engagement 3 (E3) dataset used for experiments can be accessed from the DARPA Transparent Computing repository at https://github.com/darpa-i2o/Transparent-Computing/blob/master/README-E3.md. No new data were created or analyzed in this study.

Acknowledgments: During the preparation of this manuscript/study, the authors used ChatGPT (OpenAI, GPT-5, 2025 version) for the purposes of English language editing. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IDS	Intrusion Detection System
APT	Advanced Persistent Threat
ML	Machine Learning
DAG	Directed Acyclic Graph
UUID	Universally Unique Identifier
GNN	Graph Neural Network
TC	Transparent Computing
E3	Engagement 3 (DARPA Transparent Computing Dataset)
ReLU	Rectified Linear Unit
OS	Operating System
LCS	Longest Common Subsequence
IP	Internet Protocol
CPU	Central Processing Unit
vCPU	Virtual Central Processing Unit
RAM	Random Access Memory

References

1. Kushner, D. The real story of stuxnet. *ieee Spectrum* **2013**, *50*, 48–53.
2. Haggard, S.; Lindsay, J.R. North Korea and the Sony hack: Exporting instability through cyberspace **2015**.
3. Krasznay, C. Case study: The notpetya campaign. *Információés kiberbiztonság* **2020**, pp. 485–499.
4. Alkhadra, R.; Abuzaid, J.; AlShammari, M.; Mohammad, N. Solar winds hack: In-depth analysis and countermeasures. In Proceedings of the 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE, 2021, pp. 1–7.
5. Alshamrani, A.; Myneni, S.; Chowdhary, A.; Huang, D. A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys & Tutorials* **2019**, *21*, 1851–1877.
6. Talib, M.A.; Nasir, Q.; Nassif, A.B.; Mokhamed, T.; Ahmed, N.; Mahfood, B. APT beaconing detection: A systematic review. *Computers & Security* **2022**, *122*, 102875.

 doi:10.20944/preprints202511.1455.v1

7. Li, Z.; Cheng, X.; Sun, L.; Zhang, J.; Chen, B. A hierarchical approach for advanced persistent threat detection with attention-based graph neural networks. *Security and Communication Networks* **2021**, *2021*, 9961342.
8. Ahmad, R.; Alsmadi, I.; Alhamdani, W.; Tawalbeh, L. Zero-day attack detection: a systematic literature review. *Artificial Intelligence Review* **2023**, *56*, 10733–10811.
9. Ali, S.; Rehman, S.U.; Imran, A.; Adeem, G.; Iqbal, Z.; Kim, K.I. Comparative evaluation of ai-based techniques for zero-day attacks detection. *Electronics* **2022**, *11*, 3934.
10. Santhosh Kumar, S.; Selvi, M.; Kannan, A. A comprehensive survey on machine learning-based intrusion detection systems for secure communication in internet of things. *Computational Intelligence and Neuroscience* **2023**, *2023*, 8981988.
11. Thakkar, A.; Lohiya, R. A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges. *Archives of Computational Methods in Engineering* **2021**, *28*, 3211–3243.
12. Zipperle, M.; Gottwalt, F.; Chang, E.; Dillon, T. Provenance-based intrusion detection systems: A survey. *ACM Computing Surveys* **2022**, *55*, 1–36.
13. Alsaheel, A.; Nan, Y.; Ma, S.; Yu, L.; Walkup, G.; Celik, Z.B.; Zhang, X.; Xu, D. {ATLAS}: A sequence-based learning approach for attack investigation. In Proceedings of the 30th USENIX security symposium (USENIX security 21), 2021, pp. 3005–3022.
14. Wu, L.; Xie, Y.; Wu, Y.; Liang, J.; Li, X. Provenance Based Intrusion Detection via Measuring Provenance Sequence Similarity. In Proceedings of the 2022 International Conference on Blockchain Technology and Information Security (ICBCTIS). IEEE, 2022, pp. 198–201.
15. Liu, M.; Xue, Z.; Xu, X.; Zhong, C.; Chen, J. Host-based intrusion detection system with system calls: Review and future trends. *ACM computing surveys (CSUR)* **2018**, *51*, 1–36.
16. Goyal, A.; Han, X.; Wang, G.; Bates, A. Sometimes, you aren't what you do: Mimicry attacks against provenance graph host intrusion detection systems. In Proceedings of the 30th Network and Distributed System Security Symposium, 2023.
17. Inam, M.A.; Chen, Y.; Goyal, A.; Liu, J.; Mink, J.; Michael, N.; Gaur, S.; Bates, A.; Hassan, W.U. Sok: History is a vast early warning system: Auditing the provenance of system intrusions. In Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP). IEEE, 2023, pp. 2620–2638.
18. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE transactions on neural networks* **2008**, *20*, 61–80.
19. Wang, Q.; Hassan, W.U.; Li, D.; Jee, K.; Yu, X.; Zou, K.; Rhee, J.; Chen, Z.; Cheng, W.; Gunter, C.A.; et al. You Are What You Do: Hunting Stealthy Malware via Data Provenance Analysis. In Proceedings of the NDSS, 2020.
20. Le, Q.; Mikolov, T. Distributed representations of sentences and documents. In Proceedings of the International conference on machine learning. PMLR, 2014, pp. 1188–1196.
21. Irshad, H.; Ciocarlie, G.; Gehani, A.; Yegneswaran, V.; Lee, K.H.; Patel, J.; Jha, S.; Kwon, Y.; Xu, D.; Zhang, X. Trace: Enterprise-wide provenance tracking for real-time apt detection. *IEEE Transactions on Information Forensics and Security* **2021**, *16*, 4363–4376.
22. Wang, S.; Wang, Z.; Zhou, T.; Sun, H.; Yin, X.; Han, D.; Zhang, H.; Shi, X.; Yang, J. Threatrace: Detecting and tracing host-based threats in node level through provenance graph learning. *IEEE Transactions on Information Forensics and Security* **2022**, *17*, 3972–3987.
23. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems* **2017**, *30*.
24. Zeng, J.; Zhang, C.; Liang, Z. Palantír: Optimizing attack provenance with hardware-enhanced system observability. In Proceedings of the Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022, pp. 3135–3149.
25. Wu, Y.; Xie, Y.; Liao, X.; Zhou, P.; Feng, D.; Wu, L.; Li, X.; Wildani, A.; Long, D. Paradise: real-time, generalized, and distributed provenance-based intrusion detection. *IEEE Transactions on Dependable and Secure Computing* **2022**, *20*, 1624–1640.
26. Xu, Z.; Fang, P.; Liu, C.; Xiao, X.; Wen, Y.; Meng, D. Depcomm: Graph summarization on system audit logs for attack investigation. In Proceedings of the 2022 IEEE symposium on security and privacy (SP). IEEE, 2022, pp. 540–557.
27. Wu, L.; Xie, Y.; Li, J.; Feng, D.; Liang, J.; Wu, Y. Angus: efficient active learning strategies for provenance based intrusion detection. *Cybersecurity* **2025**, *8*, 6.

 doi:10.20944/preprints202511.1455.v1

28. Shen, Y.; Stringhini, G. {ATTACK2VEC}: Leveraging temporal word embeddings to understand the evolution of cyberattacks. In Proceedings of the 28th USENIX Security Symposium (USENIX Security 19), 2019, pp. 905–921.
29. Zeng, J.; Chua, Z.L.; Chen, Y.; Ji, K.; Liang, Z.; Mao, J. WATSON: Abstracting Behaviors from Audit Logs via Aggregation of Contextual Semantics. In Proceedings of the NDSS, 2021.
30. Van Ede, T.; Aghakhani, H.; Spahn, N.; Bortolameotti, R.; Cova, M.; Continella, A.; Van Steen, M.; Peter, A.; Kruegel, C.; Vigna, G. Deepcase: Semi-supervised contextual analysis of security events. In Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP). IEEE, 2022, pp. 522–539.
31. Rehman, M.U.; Ahmadi, H.; Hassan, W.U. Flash: A comprehensive approach to intrusion detection via provenance graph representation learning. In Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP). IEEE, 2024, pp. 3552–3570.
32. Han, X.; Pasquier, T.; Bates, A.; Mickens, J.; Seltzer, M. Unicorn: Runtime provenance-based detector for advanced persistent threats. *arXiv preprint arXiv:2001.01525* **2020**.
33. Yang, F.; Xu, J.; Xiong, C.; Li, Z.; Zhang, K. {PROGRAPHER}: An anomaly detection system based on provenance graph embedding. In Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23), 2023, pp. 4355–4372.
34. Meng, L.; Xi, R.; Li, Z.; Zhu, H. PG-AID: An Anomaly-based Intrusion Detection Method Using Provenance Graph. In Proceedings of the 2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD). IEEE, 2024, pp. 2522–2527.
35. Liu, F.; Wen, Y.; Zhang, D.; Jiang, X.; Xing, X.; Meng, D. Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise. In Proceedings of the Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, 2019, pp. 1777–1794.
36. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* **2013**.
37. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Advances in neural information processing systems* **2017**, *30*.