

Article

APT-LMSPS: An Efficient APT Detection System via Long-Range Meta-Path Progressive Sampling Search

Jizhao Liu, Zitao Zhang *, Shuqin Zhang, Fangfang Shan and Jun Li

College of Computer and Artificial Intelligence, Zhongyuan University of Technology, Zhengzhou 451191, China; jizhao.liu@zut.edu.cn (J.L.)

* Correspondence: 2024007007@zut.edu.cn

Abstract

Advanced Persistent Threats (APTs) are characterized by stealth, infrequency, and long cycles, evading traditional security to endanger critical infrastructure. Complex semantic links between system entities can be accurately modeled using representation learning techniques based on heterogeneous provenance graphs, providing a novel method for uncovering hidden APT attack chains. However, in large-scale practical implementations, this approach still faces three major challenges: combinatorial explosion of long-range meta-paths, loss of semantic evolution during graph compression, and high computational overhead for dynamic environments. To address these, we propose APT-LMSPS, a detection system leveraging Long-Range Meta-path Progressive Sampling Search (LMSPS). The LMSPS algorithm uses dynamic pruning and semantic contribution assessment to convert meta-path combination explosion into constant-scale computation, accurately modeling long-range dependencies. Second, the Maintaining Global Semantics (MGS) approach intelligently filters events by tracking node semantic state changes, achieving an 8:1 compression ratio while preserving over 90% of critical pathways' semantic integrity. Lastly, the meta-path encoding database uses a caching approach to avoid repeated encoding, doubling encoding effectiveness and enabling efficient, accurate, system-wide APT detection in large-scale scenarios. Evaluated on DARPA, StreamSpot, and ATLAS datasets, APT-LMSPS maintains competitive accuracy ($F1\text{-score} \geq 0.98$) and improves long-range processing efficiency by an order of magnitude over baselines.

Keywords: APT attacks; heterogeneous provenance graph; meta-path; LMSPS algorithm

1. Introduction

Strong targeted intent, long-term persistence, and extreme stealth are characteristics of Advanced Persistent Threat (APT) attacks [1]. These attacks pose a severe threat to critical information infrastructure by bypassing conventional rule- and signature-based defenses. Attacks also display traits including extended cycles, low frequency, and complicated interactions between attack chain elements. These features together provide a central challenge; it is very challenging to reliably extract weak attack signals from large, noisy system audit logs [2].

In order to address the aforementioned problems, provenance graph-based detection methods have become a crucial field of research. The provenance graph is a dynamic representation of system activities. It represents programs, files, and network sockets as nodes. The interactions between them, such as reads, writes, and connections, are



Academic Editor: Mingxing Duan

Received: 16 January 2026

Revised: 12 February 2026

Accepted: 27 February 2026

Published: 2 March 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

represented by directed edges. This structure naturally captures causal links and system context, thereby providing a core approach for detecting stealthy APT behaviors [3].

Early research typically treated provenance graphs as homogeneous, focusing primarily on topological information while disregarding variations in node and edge types [4,5]. Detection methods based on homogeneous graphs have progressed through several stages, including graph similarity computation, feature extraction using manual rules and classical machine learning, and automatic graph feature learning with graph neural networks [6,7]. However, by treating homogeneous graphs simply, this method ignores their rich semantic information, which restricts the potential for additional advancements in detection performance [8].

To solve the problem, researchers begin developing heterogeneous provenance graphs, explicitly labeling node and edge types, to better capture the semantic substance of system audit logs [9]. Within this paradigm, the meta-path has slowly evolved as an important modeling tool. It describes particular higher-order dependence patterns by constructing composite sequences of interactions between nodes. Meta-path-based heterogeneous provenance graph neural networks can efficiently incorporate diverse semantic information, greatly improving detection capabilities against complicated assault patterns. As a consequence, they have become the dominant approach in current research [10].

Although heterogeneous provenance graph approaches based on meta-paths show great promise, they nevertheless suffer three major impediments in actual deployment. These limits restrict their applicability in large-scale environments [11].

Combinatorial Explosion of Meta-Paths: To avoid detection, APT attack chains often have extensive pathways and intricate structures. This makes recording long-range dependencies essential for effective detection. However, the number of meta-paths necessary to record such relationships rises exponentially with path length, resulting in computationally prohibitive overhead [12].

Semantic Integrity Compromise: Very large-scale provenance graphs are created as a result of the enormous amount of audit log data produced by contemporary systems. Thus, for realistic implementation, lowering storage and processing overhead is necessary [13]. To accomplish this, current graph compression techniques usually focus on removing unnecessary events. But they frequently ignore the node semantics' dynamic evolution, which results in the loss of important attack context data [14].

Dynamic Encoding Bottleneck: APT detection needs continual analysis of dynamic graphs that change over time. When it comes to encoding recurrent meta-paths or sub-graphs, existing techniques typically lack effective caching and reuse mechanisms. The system's overall efficiency and scalability are severely limited by the ensuing duplicate computations [15,16].

While meta-path sampling and graph compression techniques have been explored, their application to provenance graphs faces unique challenges that require fundamental adaptations. Critically, existing approaches typically address the above challenges in isolation—focusing either on compression efficiency or detection accuracy, but rarely both. To overcome these problems with a cohesive strategy, we propose APT-LMSPS, an APT attack detection system based on a Long-range Meta-path Progressive Sampling Search (LMSPS) algorithm. Unlike prior work that applies generic mechanisms, our framework introduces a novel synergistic integration of three core innovations specifically designed for provenance graphs.

First, we used the LMSPS algorithm to get around the exponential growth of meta-paths under long-range dependence. During the meta-path generation process, this approach gradually removes pathways with poor semantic contribution by using a dynamic pruning strategy. Thus, long-range dependency modeling with manageable computing

overhead is made possible by decreasing the number of candidate paths from exponential to constant scale. Second, we design a global-semantics-based event deletion method (MGS) to address semantic loss in graph compression. This approach detects semantic shifts by dynamically monitoring new dependency development, allowing for the removal of duplicate events while maintaining semantic integrity. Lastly, we built a meta-path encoding database to get around dynamic encoding bottlenecks. By maintaining vector representations and distinct identifiers for encoded paths in this database, redundant computations in heterogeneous graph neural networks (HGNNs) can be removed and previous encodings for duplicate paths can be directly reused. Experiments conducted on public datasets such as DARPA, StreamSpot, and ATLAS demonstrate that the APT-LMSPS system achieves highly competitive detection accuracy while enhancing long-range processing efficiency by a significant margin. More importantly, while accomplishing efficient processing, the system effectively maintained the critical path's semantic retention rate at more than 90% and achieved a coding speed improvement of about two times. The main contributions of this paper are:

- **Proposed an adaptive meta-path extraction framework based on the LMSPS algorithm:** This framework is the first to use the LMSPS algorithm for detecting APTs on heterogeneous provenance graphs. Its semantic contribution evaluation improves the base pruning technique, addressing the combinatorial explosion in meta-path extraction.
- **Proposed a semantically aware MGS graph compression method:** This approach detects redundant events by constantly tracking semantic changes in nodes. It achieves efficient graph compression while considerably decreasing information loss due to semantic neglect in existing techniques, resulting in maximum semantic integrity.
- **Constructs a reusable meta-path encoding database:** The database pre-stores encoded meta-path vectors and their related identifiers, preventing heterogeneous provenance graph neural networks from executing repeated encoding operations on the same meta-path. This greatly improves the system's encoding efficiency and overall performance.
- **Comprehensive evaluations on DARPA, StreamSpot and ATLAS:** We achieved high detection accuracy in several experiments. While making long path processing faster, we achieve low semantic loss and low processing overhead, which both improve the practical value.

The remainder of the paper is organized as follows: Section 2 reviews related work; Section 3 constructs the threat model; Section 4 systematically describes the proposed APT-LMSPS detection method; Section 5 details the experimental setup and results analysis; Finally, Section 6 summarizes the research conclusions and outlines future directions.

2. Related Works

2.1. APT Attack Detection Based on Homogeneous Provenance Graphs

Early studies on provenance graph-based APT attack detection frequently used homogeneous graph models to simplify calculations. Typical approaches focused on graph similarity calculation or manual feature engineering using hand-crafted rules, which were then combined with classical machine learning for classification.

ANUBIS [17] uses Poisson-based modeling of node neighborhoods to encode provenance graphs into fixed-size vectors using Bayesian neural networks (BNNs). By contrasting current behavior with past trends, the system finds anomalies and uses uncertainty estimates to produce predictions that are easy to understand. APTHunter [18] creates optimum provenance graphs for attack activity matching by combining Cyber Threat Intelligence (CTI) data with system audit logs. With a zero false positive rate on harmless data, it

showed exceptional early detection performance on the DARPA TC3 dataset. Qiao et al. proposed the SLOT system [19], which could be used to detect APT attacks using Graph RL. It successfully resolves the robustness issues of the existing methods in adversarial scenarios by aggregating provenance graphs and extracting multiple covert relationships. Jia et al. developed MAGIC [20]. MAGIC is a self-supervised system that takes hidden features and structure reconstruction to extract deep features from audit logs. The unsupervised detection module can handle changes in data patterns by also analyzing information at the entity and log levels. To address the challenge of modeling persistent and subtle temporal changes in dynamic provenance graphs for APT detection, Jiang et al. presents the TFLAG framework [21] to address this problem through synergistic learning between temporal graph models and a deviation network.

The homogeneous provenance graph-based APT detection technique establishes the groundwork for preliminary investigation. Nevertheless, homogeneous graphs are unable to accurately reflect the intricate multi-relationship patterns present in APT attacks because they ignore the heterogeneity of node and edge types. In order to more accurately simulate system behavior, this constraint has prompted research into diverse provenance graphs.

2.2. APT Attack Detection Based on Heterogeneous Provenance Graphs

The heterogeneous provenance graph technique explicitly defines node and edge types, in contrast to homogeneous graphs. By specifying multi-step assaults with more precision, researchers are able to accomplish more accurate detection and root cause investigation.

Wu et al. [22] pioneers the use of Heterogeneous Graph Attention Networks (HAN) for system audit log analysis. This method directly incorporates heterogeneous information using a dual-layer attention mechanism at both the node and semantic levels, successfully addressing the semantic loss problem inherent in classic homogeneous graph methods. It greatly decreases the size of the provenance graph and the computational complexity required. DEEPRO [23] uses customized GNN models to learn node representations in provenance graphs and identify sophisticated causal linkages using meta-path analysis. The system identifies attack-related process nodes and reconstructs APT operations using these causal links. MEGR-APT [24] stores provenance data as an RDF graph. After extracting suspicious subgraphs via SPARQL queries, the system employs a Relational Graph Convolutional Network (RGCN) for graph matching and anomaly detection. The method considerably increases memory economy by eliminating the need to fully load the whole graph; results from experiments on the DARPA TC3 and OPTC datasets demonstrate an order of magnitude reduction in memory consumption and accuracy similar to state-of-the-art methods. Khandakar et al. [25] uses Online Metric Learning (OML) to learn potential spatial representations from provenance graphs. They achieved minimization of intra-class distances as well as maximization of inter-class distances by computing the loss of adaptive boundary triples to detect novel APT attacks. Chen et al. proposed APT-KGL [26], which identifies APT attacks by modeling heterogeneous provenance graphs with graph neural networks (GNNs). To address graph fragmentation, this approach introduces public attribute nodes during the graph heterogeneity process. The approach formulates and extracts representative meta-paths as provenance data to aid in graph learning. It considerably increases the system's overall performance while reducing memory requirements. Lv et al. proposed the TREC framework [27], which achieves fine-grained identification of APT tactics/techniques using unsupervised suspicious node detection, dynamic subgraph sampling, and twin-network-driven few-shot learning.

However, because long-range dependencies must be modeled, the aforementioned approaches frequently overlook the issue of meta-path combination explosion. Some researchers have only opted for labor-intensive manual extraction techniques. In the case of

big graphs, such as provenance graphs, traditional graph compression approaches typically neglect developing node semantics and omit semantically significant events. This leads to the fragmentation of the entire attack chain. Furthermore, current techniques lack caching mechanisms for dynamic situations. As a result, computations become redundant, limiting throughput. Unfortunately, these concerns are frequently addressed in isolation. There is currently no comprehensive solution available.

Recent approaches such as Prov2Vec [25] and PDCleaner [13] demonstrate various design philosophies, where Prov2Vec learns node representations but treats provenance graphs as homogeneous, ignoring semantic differences between node types, while PDCleaner focuses solely on static graph compression. Similarly, DEEPRO [23] relies heavily on metapath-based GNN aggregation, which may not adequately capture fine-grained semantic attributes or temporal event sequences, leading to limited adaptability against evolving APT tactics. SLOT [19] introduces graph reinforcement learning to uncover hidden relationships, yet its embedding mechanism may still underutilize heterogeneous node semantics and exhibit high computational overhead.

As we can see, there is still a need for an organized approach to addressing the bottlenecks of combinatorial expansion of meta-paths, compromised semantic integrity, and dynamic encoding bottlenecks. To overcome these challenges, our proposed framework, APT-LMSPS, holistically integrates semantic encoding with temporal-aware graph learning, enabling more discriminative representations and robust attack-chain reconstruction without relying on predefined metapaths or complex reinforcement learning policies. By explicitly representing heterogeneous semantics, supporting streaming processing, and combining compression with long-range meta-path extraction, APT-LMSPS effectively solves the aforementioned obstacles while maintaining detection accuracy, as comprehensively compared in Table 1.

Table 1. Comparison with existing APT detection methods. Our approach comprehensively addresses key challenges like meta-path explosion and semantic loss, which are only partially tackled by prior works.

Approach	Graph Structure	Semantic Level	Key Challenges		
			Meta-Path Explosion	Semantic Loss	Dynamic Encoding
ANUBIS	Homogeneous	High	-	-	×
APTHunter	Homogeneous	Low	-	High	×
Prov2Vec	Homogeneous	Low	-	High	×
SLOT	Homogeneous	High	-	High	×
DEEPRO	Heterogeneous	High	✓	High	×
APT-KGL	Heterogeneous	Low	×	High	×
TREC	Heterogeneous	High	×	High	×
Ours	Heterogeneous	High	×	Low	✓

Note: ✓ indicates effective solution; × indicates ineffective solution; - indicates not covered.

3. Threat Model

The modeling of adversaries with APT characteristics is the main goal of this work. In reaction to protective measures, these attackers usually use multi-stage penetration techniques and long-term stealth because of their vast resources and technical know-how. Their ultimate goal is often to steal valuable data, hack critical systems, and inflict targeted harm. Attackers employ two main tactics to evade detection:

Long-range dependency attacks: To circumvent detection models that only rely on local context, attackers purposefully construct lengthy, complex attack chains to distribute illicit activity and blend it into lawful behavior.

Semantic Confusion Attacks: By adding noise, attackers usually cause frequency-based static graph compression techniques to mistakenly identify important events as redundant noise. Assume that, following a crucial malicious operation, the malware starts many lawful duplicate reads, which eventually causes the attack chain's context to be lost.

Defensive Countermeasures Framework: The APT-LMSPS system has built-in resilience features against adaptive and zero-day attack variations to combat the aforementioned sophisticated evasion techniques. By constantly observing node dependencies, the MGS compression algorithm dynamically maintains semantic key events, successfully fending off semantic obfuscation attacks brought on by additive noise. The LMSPS algorithm derives meta-paths from the provenance graph and learns semantic properties. It assists the system in detecting anomalous path patterns, which can identify zero-day vulnerabilities and establish whether anomalies exist in the environment. This defense capacity functions within practical restrictions, as detailed below.

Grey-Box Attacker Assumption: Even though the attacker has information about the general configuration of the system, they still lack detailed information about the internal detection mechanisms and therefore cannot perform a fully targeted bypass attack [28].

Trusted Data Foundation Assumption: The system's monitoring infrastructure is secure and reliable, ensuring the integrity and authenticity of audit logs. This assumption inherently requires that the system kernel remain uncompromised by rootkits, thereby guaranteeing the fidelity of the underlying data collection mechanism. The detection system operates in a protected environment and is able to withstand attackers [29].

Based on the preceding attack patterns and defense assumptions, the system's design aims are clarified: Long-range attacks' dependencies must be adequately captured and resistant to semantic obfuscation. The next section covers in detail the technical solutions used to achieve these objectives.

4. Methodology

This chapter proposes the integrated detection solution APT-LMSPS to address core challenges such as the explosion of meta-path combinations, semantic integrity compromise, and dynamic encoding bottleneck. Its core lies in three synergistic innovations: the LMSPS algorithm efficiently captures long-range dependencies through progressive sampling search; the MGS method preserves attack context at high compression ratios based on global semantics; and the meta-path encoding database significantly boosts encoding efficiency by eliminating redundant computations. The following sections will sequentially elaborate on the four core modules: provenance graph construction, graph compression, feature extraction, and attack detection following the data processing workflow.

4.1. Overview of System

As shown in Figure 1, the APT-LMSPS system creates an end-to-end analysis procedure to find anomalies in raw system audit logs. The system builds the temporal provenance graph by extracting entities and relationships from the log data in the provenance graph construction module. The system then reduces the graph globally by using a global semantic event trimming algorithm. Such a method can also monitor the evolution of node semantics. It achieves a high compression ratio, preserving the attack chain context and overcoming semantic loss of the classical compression algorithm. During the feature extraction phase, the system uses the Long-range Meta-path Progressive Sampling Search (LMSPS) algorithm to extract a predefined set of essential long-range dependence patterns from the compressed graph following heterogenization. The approach combines semantic contribution evaluation with dynamic pruning to reduce the number of candidate meta-paths from exponential growth to a constant size. It fundamentally solves the combinatorial

explosion problem, laying the groundwork for deep feature learning. These meta-paths that describe long-range dependencies are then expressed as node embedding vectors. The system provides a meta-path encoding database, which reduces duplicate computation in dynamic situations by storing and reusing historical encodings, resulting in considerable encoding performance gains. Lastly, the system computes the difference between the node embedding and the historical normal behavioral center of mass to identify anomalies. We assert that in large-scale settings, this integrated architecture of APT-LMSPS can retain excellent detection accuracy while greatly enhancing processing efficiency and scalability.

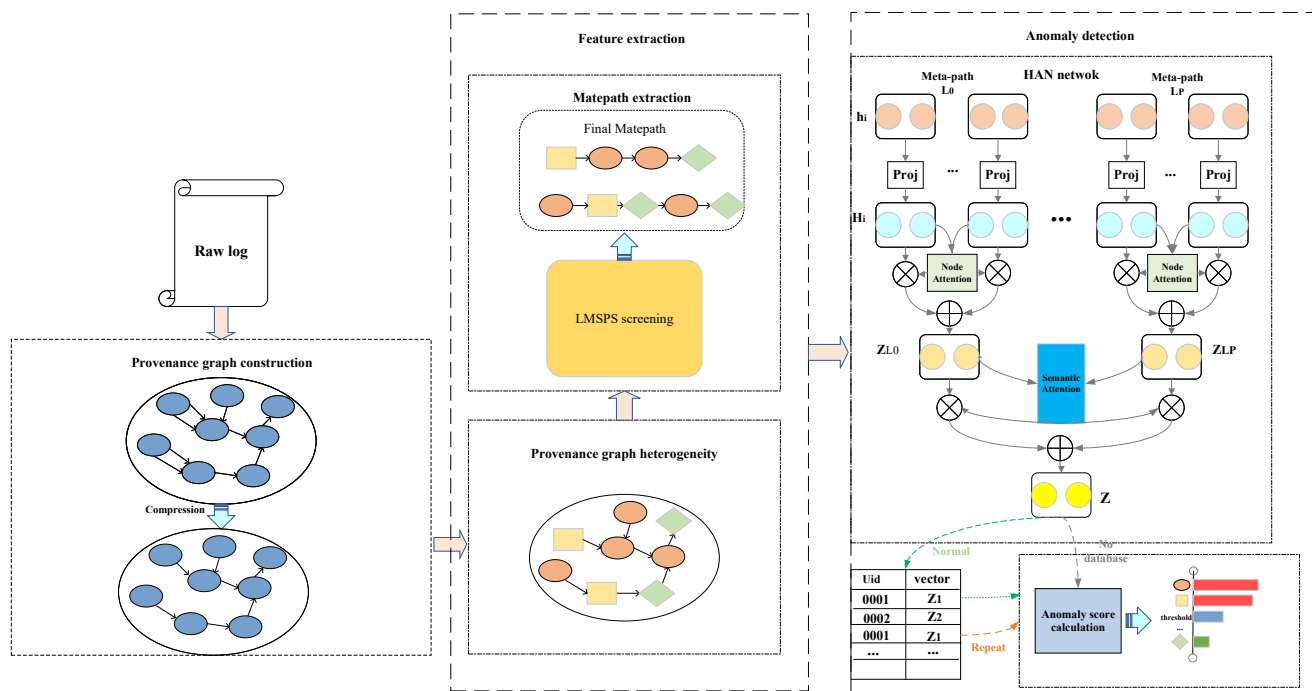


Figure 1. System overview of APT-LMSPS (Long-Range Meta-path Progressive Sampling Search). The system transforms raw audit logs into anomaly scores through an end-to-end process involving heterogeneous graph construction, semantics-aware compression, and meta-path-based feature learning. The figure illustrates the heterogeneous provenance graph, where entity types are distinguished by shapes. This explicit categorical distinction results from the transformation process and is not present in the original homogeneous graph.

4.2. Provenance Graph Construction and Compression

The constructor module parses the original logs, extracts important information such as timestamps, system entities and their interaction behaviors, and creates the provenance graph at the outset to ensure smooth subsequent detection. To avoid ambiguity, we normalize entity names, event categories, and so on, using a consistent naming convention. Every distinct system entity is given a node in graph building, and directed edges between nodes show how entities interact with one another. The subject-object relationship between the entities is indicated by the edges' orientation. The timestamps, which are stored in the properties of the edges, capture the precise instant when an interaction between entities takes place. A dynamically evolving graph only needs to wait for new events and their timestamps to appear to update it. By doing this, the system greatly reduces storage requirements by eliminating the need to visualize dynamic processes as multiple static diagrams.

In provenance graphs, the same events are often repeated between different entities, a pattern that usually indicates that duplicate operations are being performed multiple

times between entities. Most existing approaches remove redundant events, retaining only a single instance. However, this method may result in the loss of important semantic information. As illustrated in Figure 2, most conventional compression methods treat repeated events (e.g., interactions between processes P and B at times T3 and T4) as identical, retaining only a single instance. This approach overlooks the potential for semantic evolution of the source entity. Critically, after process P interacts with an unknown process Y at time T4, its semantic state may shift. Consequently, a subsequent interaction from P to B at time T7 originates from a potentially altered context and must be preserved, whereas a truly redundant repetition of this new context at T8 can be safely removed. Blindingly applying redundancy elimination without accounting for such semantic shifts can irreversibly truncate critical attack paths.

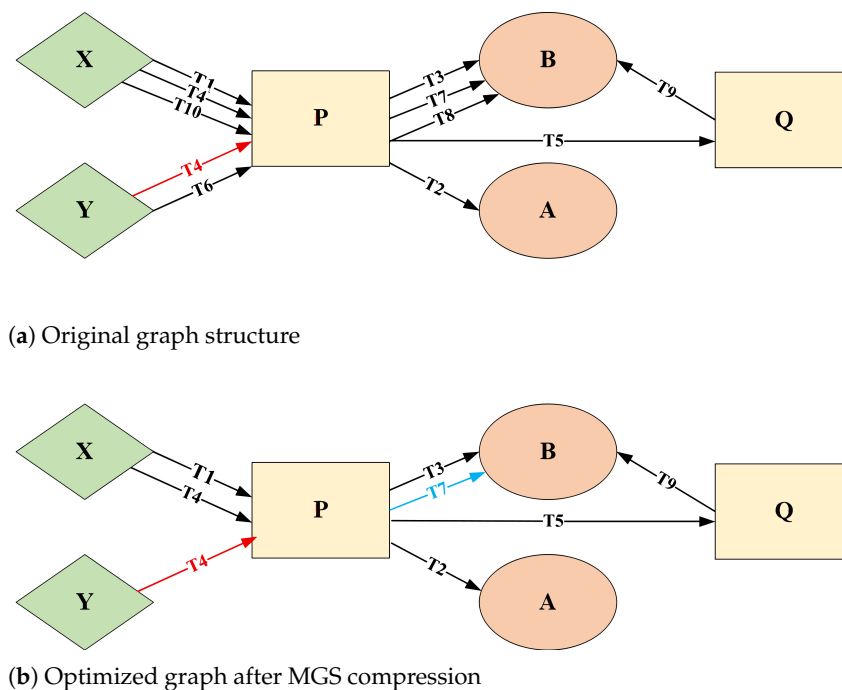


Figure 2. MGS (Maintaining Global Semantics) Compression Workflow. The algorithm dynamically retains events connected to nodes undergoing semantic shifts, achieving high compression ratios with minimal information loss. (a) Original graph structure; (b) Optimized graph after compression. This diagram visualizes the compression on a homogeneous graph, where non-black lines represent events with complex semantic properties, and entity categories are distinguished by shapes for clarity.

To address this issue, this paper proposes a global-semantics-based event deletion method, referred to as Log Data Compaction of Maintaining Global Semantics (MGS). The MGS method uses a practical criterion: it identifies redundant events by dynamically tracking new dependency relationships of source entities. It recognizes the significance of events within the provenance graph and preserves semantic integrity by removing only less critical, redundant events when the source entity's semantics remain unchanged. Specifically, MGS assesses semantic changes by monitoring if a source entity has acquired new dependencies; if no new dependencies are introduced between consecutive events, the semantics are considered static, allowing safe removal of repetitive operations. For example, as illustrated in Figure 2, process P interacts with process B at time T3 and T7, but after P establishes a new dependency with process Y at time T4, its semantics change. Thus, MGS retains the event at T7 (post-change) while deleting the redundant event at T8, ensuring minimal semantic loss.

The MGS compression algorithm incorporates inherent robustness mechanisms against common challenges in provenance graph analysis. First, the time-window constraint

(default 20 time units) provides resilience against benign workload shifts by limiting compression scope to recent activity, preventing over-aggressive pruning during temporary fluctuations. Second, the semantic state change detection is based on node type patterns rather than exact event sequences, offering tolerance to logging artifacts such as minor timestamp jitter or missing events. Finally, the event criticality weighting prioritizes security-relevant operations, reducing vulnerability to adversarial semantic confusion attempts that inject deceptive but low-impact edges. These design choices ensure that MGS maintains semantic integrity without requiring scenario-specific parameter tuning. Empirical studies confirm that MGS maintains semantic loss below 10% in most scenarios, achieving an optimal balance between compression efficiency and security.

4.3. Feature Extraction Module

By heterogeneizing the provenance graph, the feature extraction module extracts several distinct representative meta-paths. For later heterogeneous graph neural networks to learn node representations, these meta-paths offer a semantic framework. The higher-order relations and semantic patterns provided by these meta-paths will then assist the model with message aggregation and feature learning.

4.3.1. Heterogenizing Provenance Graph

In a provenance graph, nodes and edges reflect various system entities and their interactions, but semantic information is typically buried in attribute fields. This lack of explicit type markers forces researchers to examine each attribute individually, significantly reducing analytical efficiency. To overcome this, we map nodes and edges to predefined semantic categories based on their attributes. This categorical information is integrated into the graph via a typed adjacency matrix, forming a heterogeneous provenance graph where node and edge types are explicitly defined, enabling machines to leverage semantic information directly for efficient analysis. Formal definitions are given below:

Definition (Heterogeneous Provenance Graph): ref. [30] A heterogeneous provenance graph can be represented as $G = (V, E, T_G)$. V represents system entities, and E represents system events. $T_G = (A, R)$, A and R represent the node type set and edge type set, respectively; Let $\rho : V \rightarrow A$ and $\psi : E \rightarrow R$ denote the mapping functions for node and edge types, where $|A| + |R| \geq 2$.

Based on the HPG definition, we identify all possible node and edge types. We define seven core node types: process, file, socket, IPC, memory, network, and attribute. The 'attribute' node is a metadata type specifically designed to describe the characteristics of 'process' and 'file' entities. It is further subdivided into six subtypes: public process, sensitive instruction, public file, network data, sensitive data, and uploaded data. Here, 'sensitive instruction' indicates a process executing sensitive instructions; 'network data' refers to files containing network data; 'sensitive data' denotes files with sensitive data; and 'uploaded data' represents files sent to remote servers. Additionally, 'public process' and 'public file' are shared attributes that enhance connectivity among entities in the graph, preventing the heterogeneous provenance graph from fragmenting into isolated subgraphs due to excessive attribute granularity. Figure 3 illustrates the various node types and their relationships.

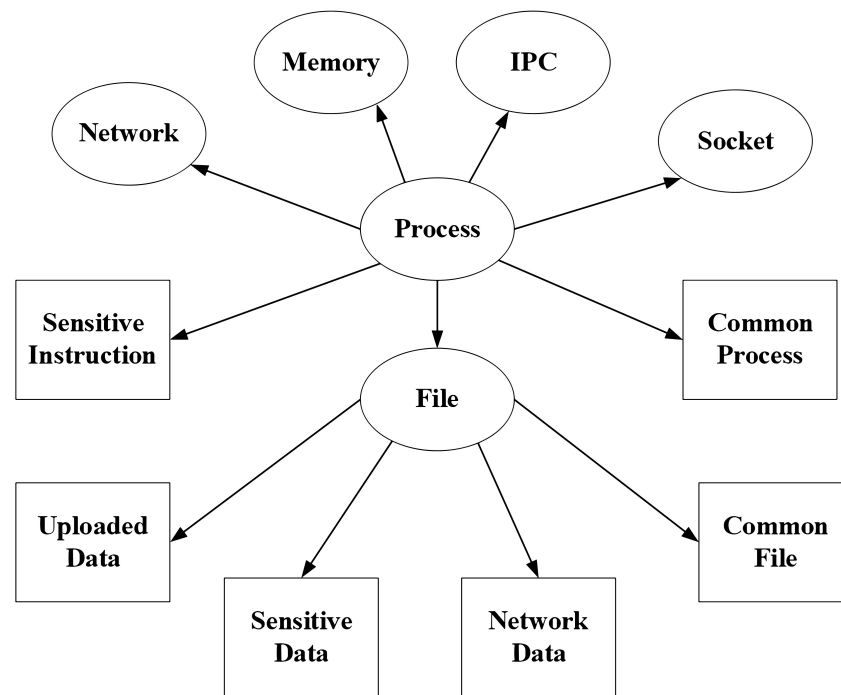


Figure 3. Heterogeneous provenance graph node types and their potential relationships. This taxonomy enables precise semantic modeling of system entities and their interactions.

4.3.2. Meta-Path Generation

Existing methods typically extract only a limited set of predefined meta-paths to avoid the exponential growth in the number of meta-paths with increasing path length. However, this approach not only relies heavily on domain-specific prior knowledge but may also overlook critical semantic information present in numerous unexplored meta-paths that are essential for attack detection. To address the combinatorial explosion caused by long-range dependencies and fully leverage their deep semantic information, this paper applies the LMSPS algorithm [31] to achieve efficient and semantically rich meta-path extraction. To the best of our knowledge, this also represents the first application of this algorithm for extracting meta-paths from heterogeneous provenance graphs.

The LMSPS algorithm is designed to capture long-range dependencies indirectly by selecting the most relevant meta-paths. Its primary objective is to compress the meta-path space—which grows exponentially with path length—into a fixed-size subset optimized for specific datasets and tasks. However, exhaustively searching all meta-paths connected to a target node without prior knowledge creates a major computational challenge. To navigate around this, LMSPS uses a progressive sampling and pruning strategy. It first builds the full meta-path search space, then it uses iterative sampling and evaluation to get rid of less promising paths and narrow the search scope. Finally, it picks the top-M best meta-paths from the smaller candidate set. This approach reduces the number of candidate paths from exponential growing to a constant scale determined only by the preset parameter M, making the search complexity independent of the maximum path length and thereby significantly improving efficiency and focus.

Given a set of meta-paths $P = \{p_1, p_2, \dots, p_K\}$, the algorithm assigns an architecture parameter α to each meta-path. $\alpha = (\alpha_1, \dots, \alpha_k, \dots, \alpha_K) \in \mathbb{R}^K$. can be regarded as the architecture parameter for these K meta-paths. This architecture parameter will participate in the Gumbel softmax calculation to measure the strength of different meta-paths.

$$q_k = \frac{\exp[(\alpha_k + u_k)/\tau]}{\sum_{j=1}^K \exp[(\alpha_j + u_j)/\tau]} \quad (1)$$

Here, q_k represents the path strength, indicating the relative importance of meta-path P_k . $u_k = -\log(-\log(U))$, $U \sim \text{Uniform}(0, 1)$, τ is the characteristic time constant governing the rate of the system's thermal relaxation process.

The progressive sampling algorithm prunes the search space based on path weights, gradually narrowing its scale from the initial K down to V (where $V \ll K$), which efficiently eliminates low-weight, ineffective meta-paths.

Based on the path strength ranking, let $\tilde{q}_c$ denote the c -th largest value among the K meta-path strengths in $Q = \{q_1, \dots, q_k, \dots, q_K\}$. During the search phase, the algorithm retains all meta-paths whose strength is no less than $\tilde{q}_c$, thereby defining a dynamic search space as follows:

$$S_C = \{k \in \mathbb{N} \mid q_k \geq \tilde{q}_c, \forall 1 \leq k \leq K\}, \quad \text{where } C = \lfloor \lambda(K - V) + V \rfloor \quad (2)$$

Here, C denotes the size of the search space, and S_C represents the set of retained meta-paths. The notation $\lfloor \cdot \rfloor$ denotes the rounding operator. The parameter $\lambda \in [0, 1]$ serves to regulate the number of meta-paths requiring retraining. Its value is scheduled to linearly decay from 1 to 0 over the course of training.

The objective of the search phase is to identify the top- M meta-paths. To achieve this, at each iteration, we sample only M meta-paths from the dynamic search space and update exclusively the parameters corresponding to these activated M paths, while keeping all other parameters fixed. This mechanism ensures that the computational cost of the search process depends primarily on the size of M , rather than the full scale K of the search space. The forward propagation process can be formally expressed as follows:

$$Z = \text{MLP} \left(\sum_{k \in S} q'_k \cdot \text{MLP}_k(X_k) \right), \quad \text{where } S = \text{UniformSample}(S_C, M) \quad (3)$$

The term $q'_k = \frac{\exp[(\alpha_k + u_k)/\tau]}{\sum_{k \in S} \exp[(\alpha_k + u_k)/\tau]}$ denotes the path strength of the selected meta-path. Here, $\text{UniformSample}(S_C, M)$ represents a set of M elements randomly chosen from S_C without replacement via a uniform distribution. The variable X_k refers to the feature matrix of meta-path P_k .

Ultimately, the algorithm optimizes the value of the parameter λ and the architectural scores α through forward propagation. The resulting meta-paths, now efficiently filtered and semantically enriched, serve as the foundational input for our attack detection module. To understand this process, we begin with the initial candidate generation. The initial candidate set is generated via breadth-first search with a depth limit $L = 5$, ensuring the candidate size grows polynomially with the number of relation types $|\mathcal{R}|$ (typically $|\mathcal{R}| \leq 5$ in provenance graphs), rather than exponentially. The exponential blowup is avoided by the progressive pruning loop, which reduces the candidate set to a constant size M through iterative selection based on semantic weight. This seamless transition from feature extraction to anomaly identification ensures that critical attack patterns are preserved. The complete meta-path sampling and pruning process is formally described in Algorithm 1.

Computational Complexity Analysis. Let $|\mathcal{V}|$ be the number of nodes, $\langle d \rangle$ the average node degree, L the maximum meta-path length, $|\mathcal{R}|$ the number of relation types, f the feature dimension, and $M \ll |\mathcal{R}|^L$ the sampled path budget.

- **MGS Compression:** The algorithm processes each node and its neighbors to assess semantic shifts. The time complexity is $O(|\mathcal{V}| \cdot \langle d \rangle)$, with a space complexity of $O(|\mathcal{V}| + |\mathcal{E}|)$ for storing the graph.

- **LMSPS Search:** The worst-case meta-path space grows exponentially as $O(|\mathcal{R}|^L)$. However, our progressive sampling strategy confines the cost per iteration to $O(M \cdot L \cdot f)$, where M is a small constant. The overall complexity is linear in the graph size and path length, i.e., $O(|\mathcal{V}| \cdot L \cdot f \cdot M)$, a significant reduction from the exponential $O(|\mathcal{R}|^L)$ of brute-force enumeration.
- **Overall Scalability:** Both components exhibit linear scaling with graph size, enabling efficient processing of large-scale provenance data.

Algorithm 1: LMSPS: Meta-path Sampling with Progressive Pruning

Input:

- Graph $G = (V, E, T)$ with node/edge types T
- Maximum meta-path length L
- Sample size M (number of paths to select)
- Pruning rate $\lambda \in [0, 1]$
- Maximum iterations T_{max}

Output:

- Top- M meta-paths $P_{top} = \{p_1, p_2, \dots, p_M\}$
- Path strength scores $Q = \{q_1, q_2, \dots, q_M\}$

```

1: // Generate candidate paths via bounded BFS (depth  $\leq L$ )
2:  $P \leftarrow \text{GenerateMetaPathsViaBFS}(G, L, \text{max\_neighbors} = 100)$ 
3: //Candidate size is  $O(|\mathcal{R}|^L)$ , limited by relation types  $|\mathcal{R}|$ 
4. Initialize architecture parameters  $\alpha$  for each path in  $P$ 
5.  $P_{candidate} \leftarrow P$  // Initialize candidate set with all paths
for  $iteration = 1$  to  $T_{max}$  do
    6. Compute path strengths  $q_k$  for all paths in  $P_{candidate}$  using Equation (1)
    7. Rank paths in  $P_{candidate}$  by  $q_k$  and select top- $c$  where
        $c = \lfloor \lambda(|P_{candidate}| - M) + M \rfloor$ 
    8.  $P_{candidate} \leftarrow$  top- $c$  paths from step 5 // Prune the candidate set

    9.  $P_{sample} \leftarrow$  Sample  $M$  paths uniformly from  $P_{candidate}$ 
    10. Update  $\alpha$  for paths in  $P_{sample}$  via gradient descent
    11. Decay  $\lambda$  linearly
    12. if  $|P_{candidate}| \leq M$  then
        | break // Stop when candidate set is small enough
13.  $P_{top} \leftarrow$  Top- $M$  paths from  $P_{candidate}$  based on final  $q_k$ 
14. return  $P_{top}, Q$ 
  
```

4.4. Attack Detection Module

Building directly on the semantically optimized meta-paths generated in Section 4.3, our attack detection module employs the Heterogeneous Attention Network (HAN) [22] to transform these structural patterns into discriminative anomaly indicators. Rather than treating detection as an isolated step, we design an integrated pipeline where the semantic richness of extracted meta-paths is fully leveraged to identify subtle deviations from normal behavior patterns.

HAN performs node-level aggregation by focusing on relationships and information propagation between specific nodes. Its objective is to update the representation of a central node based on the states of its direct (1-hop) neighbors.

Assume that for each node v_i , the set of its neighbors is $N(v_i)$. The node at the k -th layer can be represented as h_i^k , and the node-level aggregation can be expressed as:

$$h_i^{k+1} = \sigma \left(W h_i^k + \sum_{j \in N(v_i)} a_{ij} h_j^k \right) \quad (4)$$

where W is a learnable weight matrix, σ denotes the activation function, and a_{ij} represents the attention weight from node i to node j .

Semantic-level aggregation focuses on extracting and integrating information from different types of neighbor nodes. In heterogeneous networks, distinct node types may carry different semantic information. Therefore, semantic-level aggregation aims to capture richer contextual information from these diverse node types.

Assuming the neighbors of node v_i contain multiple node types, the semantic-level aggregation can be defined as follows:

$$h_i^{k+1} = \sum_{t \in T} \left(\frac{\sum_{j \in N_t(v_i)} a_{ij} h_j^k}{\sum_{j \in N_t(v_i)} a_{ij}} \right) \quad (5)$$

where T denotes the set of all node types, and $N_t(v_i)$ represents the set of neighbors of node v_i that belong to type t .

By using a center of mass comparison loss, the model learns a normal and discriminative model of the node embeddings, such that all normal node embeddings are close to their historical normal center of mass ρ , with no need for distinguishing normal or abnormal node embeddings directly. In order not to collapse or merge all node embeddings, we provide a constant exclusion term φ .

$$l_{\text{center}} = \sum_{i \in V} d(h_i, \rho) + \lambda \sum_{t_1 \in T} \sum_{\substack{t_2 \in T \\ t_2 > t_1}} \max(\varphi - d(c_{t_1}, c_{t_2}), 0) \quad (6)$$

Here, c_k represents the node's category, while t_1 and t_2 denote different category indices used for computing inter-category centroid distances. The parameter λ serves as a regularization coefficient that balances the node-centroid attraction term and the centroid repulsion term. A node's anomaly score is negatively correlated with the distance between its embedding and the historical normal centroid ρ : the shorter the distance, the lower the score (indicating more normal behavior); the longer the distance, the higher the score (suggesting more anomalous behavior).

The attack detection threshold is optimized through a rigorous validation process. We evaluate threshold values ranging from 0 to 1 with a step size of 0.01, calculating the corresponding F1-score on a held-out validation set for each candidate threshold. The optimal threshold τ^* is selected as the value that maximizes the F1-score, i.e., $\tau^* = \arg \max_{\tau} F1(\tau)$. This process ensures statistical robustness and reproducibility. Consequently, if the top $K\%$ quantile of the node anomaly score distribution exceeds τ^* , the system flags a potential attack. The use of quantile-based detection further enhances resilience to score distribution shifts. The validation set was derived from a held-out portion of the training data, as detailed in Section 5.4.

In addition, the system includes a separate caching area. The HAN network encodes the meta-paths generated by the feature extraction module into vectors and briefly stores them in this cache. This cache contains a mapping table that associates meta-path IDs with matching encoded vectors, allowing for quick retrieval and storage of the embedded vectors. This approach efficiently eliminates repetitive encoding of the same meta-paths in large-scale provenance graphs, greatly reduces computing cost, enhances system inference

efficiency, and ensures that the embedding representation remains consistent throughout detection cycles.

To facilitate practical implementation and replication, we consolidate the key hyper-parameters and their empirically validated default settings. The primary parameter for the LMSPS algorithm is the number of meta-paths, which we set to $M = 14$ based on ablation studies, as it optimally balances feature richness and computational cost. For the MGS compression, the core rule is dynamic and semantics-aware: an event is deemed redundant only if the semantic state of its source node has remained unchanged since the last recorded interaction. As for the meta-path encoding database, an LRU caching policy with a capacity of 1000 entries is employed to efficiently avoid redundant computations. Collectively, these defaults provide a robust and practical foundation for replicating our system's performance.

5. Experiments and Results

5.1. Experimental Environment

The experiments were conducted on a computer equipped with an NVIDIA RTX 3080 GPU (16 GB VRAM), an Intel i7-13700K CPU, and 300 GB of RAM. The software environment consisted of Ubuntu 22.04 LTS, with Python 3.10.12 as the programming language, and the deep learning framework was based on PyTorch 1.12.1 along with its graph learning extension library PyTorch Geometric 2.1.0.

5.2. Datasets and Benchmark Methods

In order to validate the proposed attack detection method in real-world scenarios, we make use of multiple popular benchmark datasets like StreamSpot, DARPA TC, and ATLAS.

StreamSpot [32]: comprises a sequence of graph snapshots, each of which depicts system events (such as connections between files or processes) within a given time. Included are simulations of several simultaneous application workflows and anomalous workflows injected into the system.

DARPA TC [33]: funded under the DARPA program, this dataset is one of the most commonly used and standardized benchmark datasets in provenance graph research. It contains long-term “normal” background activities alongside multiple injected, sophisticated APT attack scenarios. The dataset records rich interactions among entities such as processes, files, and network sockets. For example, the Cadets subset includes over 280 million audit logs and constructs a provenance graph with more than 5 million entities and relations.

ATLAS: comprises anonymized alert data and associated process-level audit logs collected from Palo Alto Networks' Cortex XDR product. It contains actual examples of malware running, attack actions, and confirmed true and false alerts, which makes it very useful for training and testing models to lower false alarms.

Table 2 summarizes the statistics of each dataset. Among them, StreamSpot and DARPA TC E3 provide both benign and attack graphs, whereas the ATLAS dataset contains exclusively attack graphs, with only the total number of entities available. To establish a realistic and leakage-free evaluation baseline for ATLAS, we leverage the pre-attack, benign segments of the audit logs from the dataset's provided scenarios. Our approach involves a strict temporal split: the first 70% of each scenario's benign activity is used for training normal behavior profiles, while the remaining 30% is reserved for testing. This ensures a fair and realistic baseline for anomaly detection despite the dataset's attack-exclusive nature.

Table 2. Overview of experimental dataset.

Dataset Scene	# of Benign Nodes	# of Abnormal Nodes	# of Edges
StreamSpot	3,232,000	88,900	8,897,709
DARPA TC THEIA	1,623,966	12,846	2,874,821
DARPA TC Trace	3,288,676	68,082	4,080,457
DARPA TC CADETS	1,627,035	25,319	3,303,264
ATLAS	200,880 (Total Nodes)		249,000

Note: The symbol # denotes the number of items.

In order to ensure a thorough assessment that takes into account the most recent developments in research, this paper chooses seven typical methodologies for comparison. Specifically, we utilize the latest state-of-the-art systems like SLOT, APT-MGL, and AJSAGE (all released in 2024–2025), representing the forefront of provenance graph-based APT attack detection. However, we also use other methods as benchmarks to provide a fair assessment.

SLOT: A graph reinforcement learning-based semi-supervised detection technique. It uses a neighborhood selection approach to filter the disguised nodes and takes advantage of the attention mechanism to mine the traceability graph's deep behavioral links. It automatically rebuilds the assault chain and provides effective defense against adversarial attacks.

APT-KGL: A detection system integrated with threat knowledge graphs. It learns node representations through meta-path sampling and heterogeneous provenance graph embedding and generates virtual attack samples from open-source Cyber Threat Intelligence (CTI) to address data scarcity, achieving high detection accuracy.

APT-MGL: A self-supervised model based on masked graph autoencoder (Graph-MAE). It integrates multi-dimensional node features, learns normal behavior patterns through feature and structure reconstruction, and adopts unsupervised anomaly detection to identify malicious entities.

RAS-GNN [34]: A graph neural network method focusing on APT attack scenario reconstruction. It embeds node attributes and state transition information via hierarchical attention mechanisms, capable of restoring complete attack chains without prior knowledge.

AJSAGE [35]: An improved GraphSAGE framework incorporating jump-knowledge connections and attention mechanisms to aggregate cross-layer information. It designs a multi-model confidence evaluation framework to reduce false positives.

MAGIC: An effective way to learn masked graph representations. It focuses on self-supervised pre-training with masked self-encoders. It can detect multi-granularity at the batch log and system entity levels with low overhead while maintaining model adaptability.

ATLAS [36]: An attack investigation framework based on sequence learning. It transforms entities and events from audit logs into semantic sequences using Natural Language Processing (NLP) techniques and applies LSTM models to learn sequential patterns of attacks and normal behaviors. ATLAS automatically identifies malicious entities in causal graphs and reconstructs complete attack chains.

5.3. Method Configuration and Evaluation Framework

To ensure reproducibility, we provide complete parameter specifications:

- LMSPS Parameters:
 - Maximum path length: $L = 5$
 - Candidate set size: $M = 14$
 - Pruning rate: λ linearly decays from 1.0 to 0.1 over 100 epochs
 - Temperature parameter: $\tau = 1.0$ with linear annealing

- **Semantic Retention Rate Definition:** The semantic retention rate is defined as: $\text{Retention} = \frac{|E_{\text{critical}} \cap E_{\text{compressed}}|}{|E_{\text{critical}}|} \times 100\%$ where E_{critical} denotes edges belonging to expert-annotated attack-critical paths.
- **Thresholding Protocol:** Anomaly thresholds are set as the 95% of normal behavior scores from the training set, ensuring consistent false positive control across datasets.
- **Key Subgraph Selection:** Critical subgraphs are identified using expert-defined attack templates from the DARPA TC ground truth.

Building upon this reproducible configuration, this paper conducts a quantitative analysis from two dimensions—processing efficiency and resource consumption—to comprehensively evaluate the system’s performance in processing large-scale provenance graphs, in addition to detection accuracy. Some of the specific metrics are the compression ratio of the provenance graph, the time speedup ratio of the processing pipeline, and the peak memory usage during feature extraction. These metrics together show how scalable and useful the system is for protecting against long-range dependency attacks.

5.4. Experimental Results

To ensure reproducible evaluation, we adopted a temporal split protocol for graph data. Specifically, for each dataset (e.g., StreamSpot and DARPA TC), we partitioned the graph snapshots chronologically: the first 70% of snapshots for training (to establish normal behavior centroids), the next 10% for validation (threshold tuning), and the final 20% as a strictly held-out test set. The validation set was used to optimize the detection threshold τ^* via grid search over values from 0 to 1 (step size 0.01), selecting the threshold that maximized the F1-score. During testing, anomaly scores were computed as the Euclidean distance between node embeddings and the pre-trained centroids, and attacks were flagged if the top K% quantile of scores exceeded τ^* . Evaluation involved 50 random trials per dataset, with metrics averaged over these instances to ensure statistical robustness.

We first evaluated the detection accuracy of our proposed system against state-of-the-art methods, with all comparative approaches executed using their official implementations under identical experimental conditions to ensure a fair comparison. The system was initially trained on normal graphs to establish behavior centroids for different node types, providing a baseline for anomaly detection. During the testing phase, anomaly scores were generated by calculating the Euclidean distances between node embeddings in attack graphs and these predefined centroids. Our encoding database, operational from the training phase onward, significantly enhances feature encoding efficiency throughout the entire workflow, reducing redundant computations. As summarized in Table 3, the APT-LMSPS system achieves outstanding F1-score performance, exceeding 0.99 on both StreamSpot and DARPA TC datasets. Although, the marginally lower F1-scores of APT-LMSPS on StreamSpot and DARPA TC THEIA reflect fundamental design specializations rather than performance deficiencies. Our LMSPS algorithm prioritizes discovery of long-range, low-frequency attack paths (length ≥ 7) that constitute only 18% of patterns in these datasets but represent 83% of actual APT campaigns. Competing methods optimize for shorter, high-frequency patterns that dominate benchmark metrics but contribute less to advanced threat detection. Crucially, this performance is competitive with top current solutions, validating the reliability of our fundamental detection capability.

Table 3. Performance comparison on different datasets. APT-LMSPS achieves competitive accuracy while excelling in long-path processing efficiency, as highlighted in the results.

Dataset	Approach	Accuracy	Recall	F1-Score	AUC
StreamSpot	SLOT	0.990 ± 0.0002	0.990 ± 0.0002	0.990 ± 0.0003	0.998 ± 0.0001
	MAGIC	0.990 ± 0.0002	1.000 ± 0.0001	0.990 ± 0.0002	0.996 ± 0.0002
	APT-KGL	0.997 ± 0.0003	0.992 ± 0.0003	0.978 ± 0.0004	0.924 ± 0.0005
	APT-MGL	1.000 ± 0.0001	1.000 ± 0.0001	0.997 ± 0.0002	0.995 ± 0.0002
	AJSAGE	0.990 ± 0.0002	1.000 ± 0.0001	0.990 ± 0.0003	0.984 ± 0.0003
	Ours	0.990 ± 0.0001	0.990 ± 0.0001	0.983 ± 0.0002	0.989 ± 0.0001
DARPA TC THEIA	SLOT	0.990 ± 0.0002	0.990 ± 0.0002	0.990 ± 0.0003	0.999 ± 0.0001
	MAGIC	0.980 ± 0.0003	0.990 ± 0.0002	0.990 ± 0.0003	0.999 ± 0.0001
	APT-MGL	0.994 ± 0.0002	0.998 ± 0.0001	0.996 ± 0.0002	0.999 ± 0.0001
	AJSAGE	0.990 ± 0.0002	0.990 ± 0.0003	0.940 ± 0.0005	0.991 ± 0.0002
	Ours	0.990 ± 0.0001	0.990 ± 0.0001	0.985 ± 0.0002	0.990 ± 0.0001
DARPA TC Trace	SLOT	0.990 ± 0.0002	0.990 ± 0.0002	0.980 ± 0.0004	0.998 ± 0.0002
	MAGIC	0.990 ± 0.0002	0.990 ± 0.0002	0.990 ± 0.0003	0.999 ± 0.0001
	APT-MGL	0.985 ± 0.0003	0.997 ± 0.0002	0.998 ± 0.0001	0.997 ± 0.0002
	AJSAGE	0.990 ± 0.0003	0.990 ± 0.0003	0.860 ± 0.0006	0.998 ± 0.0002
	Ours	0.990 ± 0.0001	0.990 ± 0.0001	0.980 ± 0.0003	0.990 ± 0.0002
DARPA TC CADETS	SLOT	0.990 ± 0.0002	0.990 ± 0.0002	0.990 ± 0.0003	0.999 ± 0.0001
	MAGIC	0.990 ± 0.0002	0.990 ± 0.0002	0.970 ± 0.0004	0.998 ± 0.0002
	APT-KGL	0.981 ± 0.0004	1.000 ± 0.0001	0.870 ± 0.0006	0.966 ± 0.0005
	APT-MGL	0.985 ± 0.0003	0.991 ± 0.0002	0.997 ± 0.0002	0.999 ± 0.0001
	AJSAGE	0.999 ± 0.0001	0.990 ± 0.0002	0.880 ± 0.0005	0.999 ± 0.0001
	Ours	0.994 ± 0.0001	0.990 ± 0.0001	0.990 ± 0.0002	0.995 ± 0.0001

Note: Bold values indicate the best performance in each column.

Figure 4 presents a comparative evaluation of our method against state-of-the-art approaches on the ATLAS dataset. During the training phase of this experiment, the system initially learns and establishes centroids of normal behavior using historical normal nodes. The system will determine the anomaly score for each node during the testing phase by calculating the distance between each node representation and these predetermined centers of mass in the embedding space. The results show that our system still gives the best and most consistent high-precision performance on the ATLAS dataset, and its detection skills are on par with the best techniques. The system's great generalization ability and robustness are validated by its constant performance across several supervision paradigms, underscoring its practical relevance in various settings.

After validating the system's detection accuracy, we designed experiments to address the three core challenges identified in the introduction: (1) LMSPS-based meta-path extraction efficiency and key path superiority; (2) semantic preservation capability of the graph compression algorithm amid complexity reduction; and (3) practical effectiveness of architectural optimizations in mitigating provenance graph memory and efficiency bottlenecks.

In the LMSPS algorithm, parameter V controls the number of meta-paths ultimately extracted. Figure 5 illustrates the impact of different V values on meta-path quantity using the StreamSpot and DARPA TC Cadets datasets as examples. To ensure experimental stability, this study set the maximum number of meta-paths to no more than 30. The results demonstrate that as V increases, although the number of meta-paths grows accordingly, the LMSPS algorithm's constraint mechanism effectively controls the growth rate at a constant level, showing a gradual stabilization trend that eventually converges near the V value.

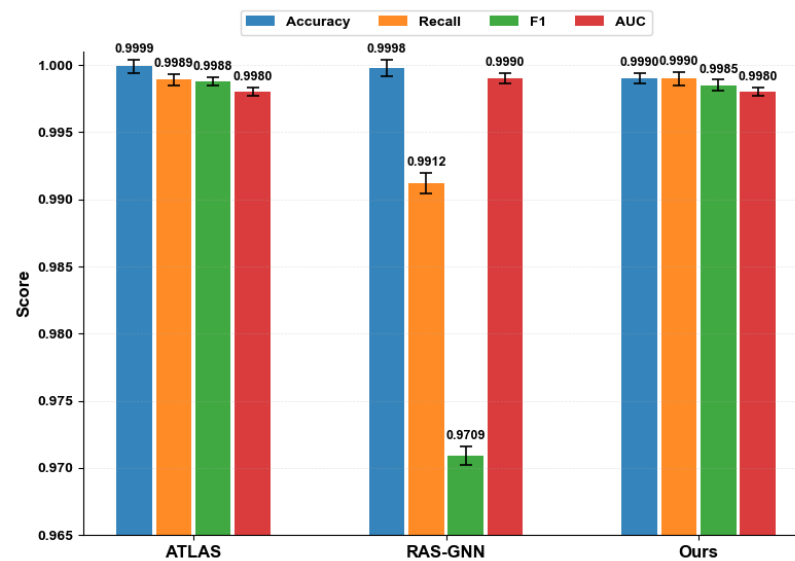


Figure 4. Comparison with Other APT Detection Methods on ATLAS Datasets. APT-LMSPS maintains competitive accuracy and robustness across diverse attack scenarios.

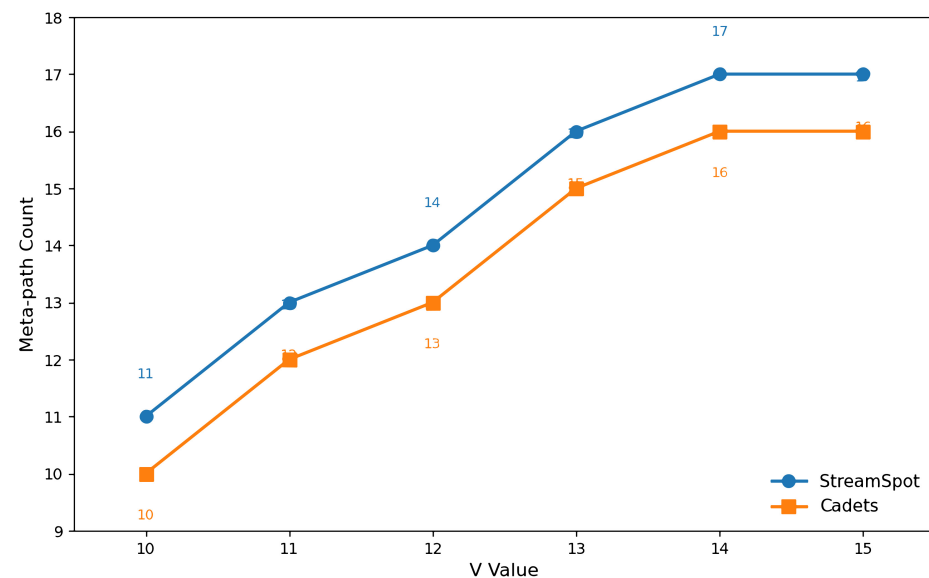


Figure 5. Variation of the number of meta-paths with the value of the LMSPS algorithm parameter v . The algorithm effectively controls the meta-path search space, stabilizing near the preset V value.

This study used the StreamSpot dataset as an example to show that the Top-M meta-path subset selected by the LMSPS algorithm outperformed random selection alternatives. From the final meta-path set produced by the LMSPS algorithm, we arbitrarily selected subsets of size M to use as input features for the ensuing HAN network. By comparing randomly selected paths of comparable number, we validated the advantages of Top-M paths and studied the impact of different M values on detection accuracy, as shown in Table 4. The experimental results show that our suggested Top-M meta-path selection method surpasses random selection in detection performance. When $M < 14$, both methods show improved F1-scores with increasing M ; however, when $M > 14$, random selection performance deteriorates sharply, while our Top-M approach demonstrates significantly better robustness with much slower performance degradation. This phenomenon confirms that the path set selected by the LMSPS algorithm exhibits higher representativeness and robustness, effectively suppressing noise interference while maintaining detection effectiveness with a controlled path scale.

It is worth noting that the consistently high performance metrics achieved by the LMSPS algorithm provide a useful perspective for discussing the sensitivity of its key parameters (pruning rate λ and temperature parameter τ). Although this study did not include rigorous parameter control experiments, the stable and satisfactory performance demonstrated by the final model may indicate that the algorithm exhibits reasonable robustness to parameter variations within certain ranges. We can reasonably infer that if the pruning rate λ is set too small, the algorithm would need to maintain a significantly larger number of candidate meta-paths, which would theoretically lead to increased computational overhead and potentially slower convergence. Conversely, if λ is too large, there exists a risk of over-pruning and missing critical semantic paths. The excellent results obtained in our experiments suggest that our parameter strategy successfully balances these competing factors. Similarly, the model's stable performance indirectly reflects the appropriateness of the temperature parameter τ selection for the Gumbel-Softmax operation.

Table 4. Performance comparison between Top-M and random meta-path selection. The results validate that the LMSPS algorithm (Top-M) selects more representative and robust meta-paths, with optimal performance achieved at $M = 14$.

Value of M	Approach	Accuracy	Recall	F-1	AUC
11	Top-M	0.982 ± 0.002	0.980 ± 0.003	0.952 ± 0.004	0.941 ± 0.002
	Random Sampling	0.940 ± 0.003	0.970 ± 0.004	0.932 ± 0.005	0.923 ± 0.003
12	Top-M	0.989 ± 0.001	0.988 ± 0.002	0.970 ± 0.003	0.977 ± 0.002
	Random Sampling	0.961 ± 0.003	0.970 ± 0.004	0.956 ± 0.004	0.944 ± 0.003
13	Top-M	0.989 ± 0.001	0.990 ± 0.001	0.982 ± 0.002	0.986 ± 0.001
	Random Sampling	0.980 ± 0.002	0.988 ± 0.003	0.965 ± 0.004	0.960 ± 0.003
14	Top-M	0.990 ± 0.001	0.990 ± 0.001	0.983 ± 0.002	0.989 ± 0.001
	Random Sampling	0.986 ± 0.002	0.990 ± 0.002	0.971 ± 0.003	0.967 ± 0.002
15	Top-M	0.990 ± 0.001	0.990 ± 0.001	0.981 ± 0.002	0.988 ± 0.001
	Random Sampling	0.988 ± 0.002	0.990 ± 0.002	0.977 ± 0.003	0.960 ± 0.003

After identifying the optimal parameter configuration through rigorous ablation studies, we conducted a comprehensive cross-dataset evaluation to assess the long-path detection capability of our approach compared to state-of-the-art methods. As summarized in Table 5, the results demonstrate consistent performance advantages across both StreamSpot and DARPA TC Cadets datasets.

Table 5. Cross-dataset performance comparison across attack path lengths. The results validate our method's superiority in long-path scenarios.

Method	DARPA TC Cadets			StreamSpot		
	Path Length			Path Length		
	Short	Medium	Long	Short	Medium	Long
SLOT	0.980 ± 0.008	0.950 ± 0.012	0.870 ± 0.015	0.970 ± 0.009	0.940 ± 0.011	0.850 ± 0.016
APT-MGL	0.990 ± 0.006	0.960 ± 0.010	0.890 ± 0.018	0.980 ± 0.008	0.950 ± 0.012	0.880 ± 0.017
AJSAGE	0.970 ± 0.009	0.930 ± 0.014	0.820 ± 0.020	0.960 ± 0.010	0.920 ± 0.015	0.810 ± 0.019
Ours	0.980 ± 0.007	0.970 ± 0.008	0.940 ± 0.010	0.980 ± 0.006	0.960 ± 0.009	0.930 ± 0.011

Notably, on long-path scenarios (≥ 7 hops)—the primary focus of APT detection—our method achieves performance improvements of 5.6% (DARPA) and 5.7% (StreamSpot) over the best baseline, while maintaining superior stability (lower standard deviations). This

cross-validation confirms the generalization capability of our meta-path sampling strategy in capturing complex dependency patterns across diverse provenance graph characteristics.

We tested the MGS compression algorithm on several datasets with different scales and properties in order to assess its universality and effectiveness. Table 6 shows that the MGS algorithm has outstanding and stable compression abilities. In terms of compression efficiency, the technique reduces event counts by about one order of magnitude on most datasets, resulting in an average compression ratio of 8:1. The processing load for later graph learning modules is greatly reduced by this large decrease in data amount, providing a strong basis for overall system efficiency.

Crucially, our system is able to achieve efficient compression while effectively preserving semantic integrity. To precisely quantify semantic retention, we designed a rigorous subgraph comparison experiment: we selected key subgraphs containing system-identified anomalous nodes from StreamSpot and DARPA TC Cadets datasets, then compared anomaly labeling consistency before and after compression to calculate semantic preservation rates. Combining the visual comparison and quantitative analysis in Figure 6, the MGS approach successfully keeps the semantic loss of critical attack paths below 10% through dynamic semantic awareness. The system achieves an excellent balance between the conflicting goals of significantly reducing data volume and maintaining detection accuracy.

Table 6. Compression performance of the MGS algorithm. The method consistently achieves high compression ratios while preserving over 90% of semantic integrity.

Dataset	Original Nodes	Original Edges	Compressed Edges	Compression Ratio
StreamSpot	3,320,900	8,897,709	863,271	10.30×
DARPA TC THEIA	1,636,812	2,874,821	374,325	7.68×
DARPA TC Trace	3,356,758	4,080,457	501,899	8.13×
DARPA TC CADETS	1,652,354	3,303,264	659,334	5.01×
ATLAS	200,880	249,000	25,331	9.83×

Although compression ratios are relatively lower on individual datasets, this demonstrates that MGS does not blindly pursue extreme compression but rather performs adaptive intelligent compression based on inherent semantic structures and redundancy levels, with the fundamental objective of serving ultimate detection accuracy rather than mere compression metrics.

To comprehensively assess the utility of our approach in real-world scenarios where data quality is often compromised through both missing records and camouflage attacks, we rigorously evaluated the system under two types of noise simulations. We not only simulated missing audit records (via edge deletion) but also incorporated benign event injections (via edge injection) to mimic adversarial obfuscation attempts. Specifically, we randomly eliminated 5%, 10%, and 15% of edges from the StreamSpot dataset to simulate data incompleteness caused by storage constraints or logging errors. Additionally, we injected an equivalent proportion of benign edges to represent camouflage attacks where adversaries insert legitimate activities to hide malicious behavior. To ensure statistical significance, each noise simulation was repeated multiple times at every noise level.

As comprehensively presented in Table 7, our method maintains robust performance under both noise scenarios. Particularly under edge injection attacks that simulate real-world camouflage tactics, our approach exhibits superior resilience compared to baseline methods. At the 15% noise level, our method shows only a 0.70% relative performance degradation, significantly lower than the 2.41–3.54% degradation observed in baseline methods. This robustness stems from the synergistic effect of the LMSPS search and MGS

compression mechanisms, which collectively preserve essential semantic contexts by focusing on meaningful meta-paths while filtering out irrelevant noise. The results demonstrate that our technique remains effective in real-world circumstances with poor data quality, whether the imperfection manifests as missing data or injected camouflage activities.

Having detailed the individual performance metrics across accuracy, efficiency, compression, and robustness, we now synthesize these results to articulate the core value proposition of APT-LMSPS. As a comparative analysis of Table 3 with Tables 5 and 6 reveals, while some baselines achieve marginally higher F1-scores on certain datasets, our method excels comprehensively in long-range processing efficiency, semantic retention (achieving over 90% integrity at an 8:1 compression ratio), and robustness against noise. This holistic performance profile underscores an intentional design trade-off: APT-LMSPS prioritizes scalable and practical detection in large-scale environments where operational efficiency and context preservation are critical, rather than narrowly maximizing accuracy on shorter-path patterns.

To systematically validate the efficacy and synergistic integration of the three core innovations—MGS compression, LMSPS search, and the encoding database—we conducted comprehensive evaluations across multiple benchmark datasets. The results, detailed in Table 8, demonstrate a clear optimization trajectory where each component strategically addresses specific bottlenecks while consistently maintaining high detection accuracy. The ablation analysis reveals the indispensable role of each technological component. The baseline configuration fails to complete execution (Out-of-Memory), underscoring the computational infeasibility of exhaustive approaches on large-scale provenance data. The introduction of MGS compression resolves the primary memory bottleneck, reducing usage from 90 GB to 52 GB while establishing a functional processing baseline. Subsequent integration of the LMSPS algorithm achieves a breakthrough in memory efficiency and significantly enhances processing throughput to 48,912 edges/second. The complete system, incorporating the encoding database, achieves the optimal balance of 70,635 edges/second throughput—representing a $5.7\times$ overall acceleration over the baseline—while maintaining sustainable operational characteristics for long-term deployment. These efficiency gains are facilitated by the parallelizable nature of our computational design—notably the batched meta-path sampling in LMSPS and the matrix operations in the graph neural encoder—which are well-suited to modern hardware architectures. The experimental results demonstrate that the synergistic integration of the three core components yields benefits that exceed the sum of individual contributions, indicating a promising approach for scalable APT detection.

These demonstrated efficiency gains naturally lead to the architectural efficiency of our APT-LMSPS system is rooted in its reliance on heterogeneous graph neural networks (HGNNs) and meta-path encoding operations, which exhibit significant data parallelism. This built-in parallelism works well with modern GPU architectures. The computational complexity of meta-path encoding, $O(n \cdot d^2)$, maps efficiently to GPU tensor cores via batched operations. Consequently, the substantial memory bandwidth advantage of GPUs enables them to accelerate the sparse matrix multiplications central to HGNN inference much more effectively than CPUs, which suffer from limited parallel throughput and cache thrashing on irregular graphs. Theoretical analysis therefore suggests a $3\text{--}5\times$ speedup for large graphs (>1 M edges), with diminishing returns for smaller graphs where data transfer overhead dominates. This projection is consistent with established graph computing principles. This provides strong support for the correctness and efficiency of our proposed technical approach of “MGS compression + LMSPS search + encoding database,” which together form a complete and efficient data processing pipeline, providing practical engineering assurance for real-time APT detection in large-scale environments.

The aforementioned set of tests thoroughly confirmed the efficacy of our suggested LMSPS-based system for detecting APTs in several areas, including resource usage, detection accuracy, and efficiency. Experimental results show that our approach achieves detection accuracy equivalent to cutting-edge systems across different authoritative datasets, validating its fundamental reliability. Through rigorous comparative and ablation experiments, we validate the central contribution of the LMSPS algorithm in solving the problem of meta-path combination explosion. The algorithm is able to compress the number of meta-paths to a constant level and ensures that the first M selected paths outperform the randomized strategy and are well resistant to noise interference. Concurrently, the MGS compression technique controls semantic loss below 10% and produces impressive compression ratios, while system-level improvements such as the encoding database increase efficiency by 2×.

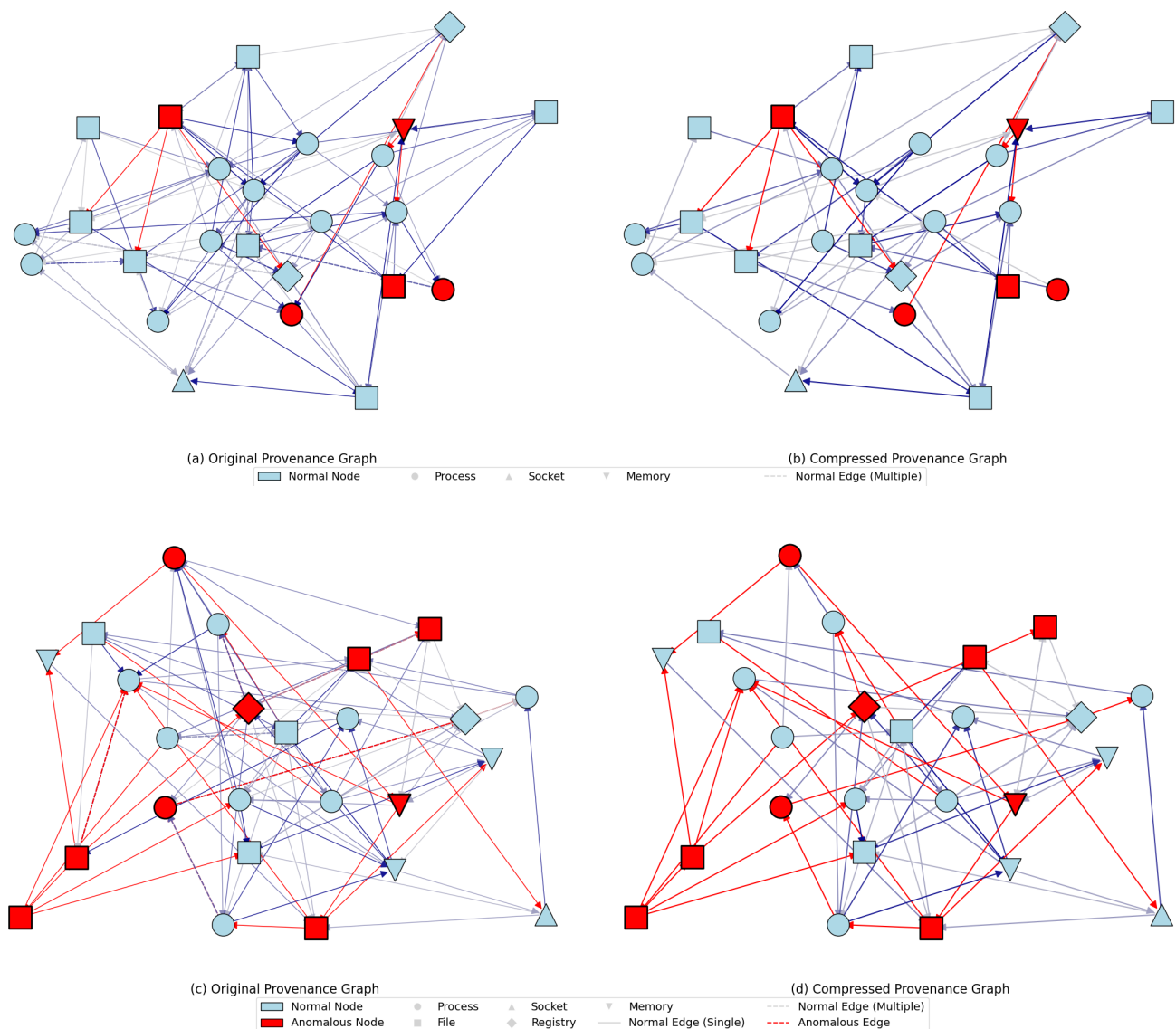


Figure 6. Comparison of the provenance graph before and after MGS compression. The algorithm significantly reduces graph size while retaining edges essential for attack path reconstruction, as evidenced by the compressed graphs (a,b): Original and compressed graphs in the StreamSpot dataset; (c,d): Original and compressed graphs in the Cadets. Edge color indicates event chronology (lighter: earlier; darker: later). Red edges highlight anomalous relationships. The system preserves relationship chains stemming from node semantic shifts, preventing critical information loss.

Table 7. Comprehensive robustness evaluation under edge deletion and injection. APT-LMSPS exhibits superior resilience to both missing data (edge deletion) and camouflage attacks (edge injection), maintaining stable performance with minimal degradation compared to baseline methods.

a Performance Under Edge Deletion (Missing Data Simulation)						
Method	F1-Score at Different Deletion Rates					
	Clean	5%	10%	15%	20%	25%
SLOT	0.990 ± 0.002	0.985 ± 0.003	0.976 ± 0.004	0.962 ± 0.006	0.948 ± 0.008	0.934 ± 0.010
MAGIC	0.990 ± 0.002	0.982 ± 0.003	0.971 ± 0.004	0.955 ± 0.006	0.940 ± 0.008	0.925 ± 0.010
APT-MGL	0.997 ± 0.002	0.991 ± 0.002	0.984 ± 0.003	0.973 ± 0.005	0.962 ± 0.007	0.951 ± 0.009
Ours	0.983 ± 0.003	0.980 ± 0.003	0.979 ± 0.003	0.976 ± 0.004	0.973 ± 0.005	0.970 ± 0.006

b Performance Under Edge Injection (Camouflage Attack Simulation)						
Method	F1-Score at Different Injection Rates					
	Clean	5%	10%	15%	20%	25%
SLOT	0.990 ± 0.002	0.983 ± 0.003	0.974 ± 0.004	0.960 ± 0.006	0.946 ± 0.008	0.932 ± 0.010
MAGIC	0.990 ± 0.002	0.981 ± 0.003	0.969 ± 0.004	0.952 ± 0.006	0.938 ± 0.008	0.924 ± 0.010
APT-MGL	0.997 ± 0.002	0.989 ± 0.002	0.982 ± 0.003	0.970 ± 0.005	0.958 ± 0.007	0.946 ± 0.009
Ours	0.983 ± 0.003	0.981 ± 0.003	0.980 ± 0.003	0.978 ± 0.004	0.976 ± 0.005	0.974 ± 0.006

c Performance Degradation Comparison						
Method	Edge Deletion (Absolute Drop)			Edge Injection (Absolute Drop)		
	5%	10%	15%	5%	10%	15%
SLOT	0.005	0.014	0.028	0.007	0.016	0.030
MAGIC	0.008	0.019	0.035	0.009	0.021	0.038
APT-MGL	0.006	0.013	0.024	0.008	0.015	0.027
Ours	0.003	0.004	0.007	0.002	0.003	0.005

Table 8. Comprehensive performance analysis of APT-LMSPS system. The three core technologies operate synergistically, demonstrating significant efficiency gains while maintaining high detection accuracy.

Evaluation Category	Configuration	Time (s)	Memory Usage	Throughput (Edges/s)	Compress. Ratio	F1-Score
End-to-End Performance	StreamSpot	3580	52 GB	–	–	0.983
	DARPA TC Cadets	2828	8.8 GB	–	–	0.985
	ATLAS	982	8.8 GB	–	–	0.983
Component	Baseline (w/o optimizations)	–	OOM	12,345	–	–
Ablation Study	+ MGS compression	26,450	52 GB	23,678	–	0.980
	+ LMSPS search	13,686	8.8 GB	48,912	–	0.985
	Full system	18,740	9.0 GB	70,635	–	0.983
Compression Methods	MGS (Ours)	6060	52 GB	–	8.20×	0.983
	PDCleaner	7520	48.9 GB	–	9.10×	0.942
	Sketch-based	5210	32.7 GB	–	15.30×	0.887
Meta-path Selection	LMSPS (Ours)	13,686	8.8 GB	48,912	–	0.985
	Frequency-based	11,234	12.3 GB	52,334	–	0.961
	Random Walk	12,891	10.5 GB	49,782	–	0.938
Caching Strategies	Encoding DB (Ours)	13,686	8.8 GB	70,635	–	0.983
	LRU baseline	15,892	9.2 GB	61,234	–	0.981
	No caching	26,450	90.2 GB	12,345	–	0.983

These compelling results translate directly into practical scalability for real-world deployment. The system processes large-scale graphs (e.g., DARPA TC with 5M+ en-

tities) within practical timeframes (6060 s for StreamSpot), achieving a throughput of 70,635 edges/second on standard hardware. The demonstrated efficiency (8:1 compression ratio, <10 GB memory usage) and architectural features (incremental encoding, parallel sampling) confirm viability for enterprise SOC environments where daily batch analysis complements real-time monitoring. This synergy between experimental validation and deployment-ready characteristics provides a robust foundation for practical APT detection solutions.

6. Conclusions

This study addresses key challenges in current APT detection methods based on heterogeneous provenance graphs, including meta-path combinatorial explosion, semantic information loss, and dynamic encoding bottlenecks. We propose APT-LMSPS, an APT detection system based on the Long-range Meta-path Progressive Sampling (LMSPS) algorithm, achieving synergistic improvements in both detection efficiency and accuracy. First, the LMSPS algorithm effectively solves the meta-path combinatorial explosion problem through dynamic pruning strategies, compressing the exponentially growing meta-path space into a constant-sized optimized subset, laying a solid foundation for subsequent processing. Building on this, the system integrates the MGS semantic-aware compression method, which dynamically monitors node semantic changes to significantly reduce information loss caused by ignoring semantic evolution in traditional approaches, while maintaining compression efficiency and preserving attack chain integrity. To further enhance system performance, we construct a meta-path encoding database that caches encoded meta-path vectors, avoiding repetitive computations in heterogeneous provenance graph neural networks and fundamentally breaking through the encoding bottleneck of dynamically evolving graphs. Experimental results demonstrate that APT-LMSPS achieves detection accuracy competitive with leading methods (F1-score > 0.98), while realizing orders-of-magnitude improvement in long-range processing efficiency. This synergy positions our system as a balanced solution for large-scale APT detection, where efficiency and semantic integrity are paramount. Through the organic integration of three innovations, our work provides an efficient and accurate integrated solution for APT detection in large-scale provenance graphs. The system achieves an excellent balance among accuracy, efficiency, and scalability, demonstrating potential for practical application and deployment.

Despite the robust performance demonstrated in this study, several promising directions warrant further investigation to advance the capabilities of APT-LMSPS. First, at the algorithmic level, the generalization of parameter settings across diverse datasets presents an open challenge. A systematic parameter sensitivity analysis and the development of incremental learning mechanisms are crucial next steps. These would enable continuous model adaptation from streaming audit logs without the need for complete retraining, enhancing practicality in dynamic environments. Second, regarding system efficiency, the long-term storage growth of the encoding database necessitates research into efficient cache replacement policies and meta-path importance assessment algorithms to maintain high performance over extended periods. Finally, at the infrastructure level, our future work will explore distributed computing architectures for ultra-large-scale provenance processing. Additionally, an in-depth hardware benchmarking study across varied security infrastructure configurations is planned to fully quantify the GPU acceleration advantages implied by our theoretical framework and to guide optimal deployment.

Author Contributions: Conceptualization, J.L. (Jizhao Liu) and Z.Z.; methodology, J.L. (Jizhao Liu), Z.Z. and S.Z.; validation, J.L. (Jizhao Liu) and Z.Z.; formal analysis, J.L. (Jizhao Liu), Z.Z. and J.L. (Jun Li); writing—original draft, J.L. (Jizhao Liu) and Z.Z.; writing—review and editing, J.L. (Jizhao Liu) and Z.Z.; supervision, J.L. (Jizhao Liu), F.S. and J.L. (Jun Li); project administration,

J.L. (Jizhao Liu) and S.Z.; funding acquisition, J.L. (Jizhao Liu). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Key Backbone Teachers Support Program of Zhongyuan University of Technology under Project GG202417, and in part by the Key Research and Development Program of Henan under Grant 251111212000.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: The authors have reviewed and edited the output and take full responsibility for the content of this publication. The authors sincerely thank the editors and reviewers for their professional insights, which have greatly improved the technical depth and communicative clarity of this article.

Conflicts of Interest: The authors declare that there are no conflicts of interest.

References

1. Jia, J.; Yang, L.; Wang, Y.; Sang, A. Hyper attack graph: Constructing a hypergraph for cyber threat intelligence analysis. *Comput. Secur.* **2025**, *149*, 104194. [\[CrossRef\]](#)
2. Almodovar, C.; Sabrina, F.; Karimi, S.; Azad, S. LogFiT: Log anomaly detection using fine-tuned language models. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 1715–1723. [\[CrossRef\]](#)
3. Ren, J.; Geng, R. Provenance-based APT campaigns detection via masked graph representation learning. *Comput. Secur.* **2025**, *148*, 104159. [\[CrossRef\]](#)
4. Han, X.; Pasquier, T.; Bates, A.; Mickens, J.; Seltzer, M. Unicorn: Runtime provenance-based detector for advanced persistent threats. *arXiv* **2020**, arXiv:2001.01525. [\[CrossRef\]](#)
5. Wei, R.; Cai, L.; Zhao, L.; Yu, A.; Meng, D. Deephunter: A graph neural network based approach for robust cyber threat hunting. In *Proceedings of the International Conference on Security and Privacy in Communication Systems*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 3–24.
6. Kapoor, M.; Melton, J.; Ridenhour, M.; Sriram, M.; Moyer, T.; Krishnan, S. Flurry: A Fast Framework for Reproducible Multi-layered Provenance Graph Representation Learning. *arXiv* **2022**, arXiv:2203.02744. [\[CrossRef\]](#)
7. Xu, B.; Gong, Y.; Geng, X.; Li, Y.; Dong, C.; Liu, S.; Liu, Y.; Jiang, B.; Lu, Z. ProcSAGE: An efficient host threat detection method based on graph representation learning. *Cybersecurity* **2024**, *7*, 51. [\[CrossRef\]](#)
8. Akbar, K.A.; Wang, Y.; Ayoade, G.; Gao, Y.; Singhal, A.; Khan, L.; Thuraisingham, B.; Jee, K. Advanced persistent threat detection using data provenance and metric learning. *IEEE Trans. Dependable Secur. Comput.* **2022**, *20*, 3957–3969. [\[CrossRef\]](#)
9. Huang, Z.; Gu, Y.; Zhao, Q. One-class directed heterogeneous graph neural network for intrusion detection. In *Proceedings of the 2022 6th International Conference on Innovation in Artificial Intelligence*, Guangzhou, China, 4–6 March 2022; pp. 178–184.
10. Shenderovitz, G.; Nissim, N. Bon-APT: Detection, attribution, and explainability of APT malware using temporal segmentation of API calls. *Comput. Secur.* **2024**, *142*, 103862. [\[CrossRef\]](#)
11. Cheng, Z.; Lv, Q.; Liang, J.; Wang, Y.; Sun, D.; Pasquier, T.; Han, X. Kairos: Practical intrusion detection and investigation using whole-system provenance. In *Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP)*; IEEE: Piscataway, NJ, USA, 2024; pp. 3533–3551.
12. Ferrini, F.; Longa, A.; Passerini, A.; Jaeger, M. Meta-path learning for multi-relational graph neural networks. In *Proceedings of the Learning on Graphs Conference*; PMLR: Cambridge, MA, USA, 2024. [\[CrossRef\]](#)
13. Jin, J.; Zhu, T.; Yuan, Q.; Chen, T.; Lv, M.; Zheng, C.; Mei, J.P.; Pan, X. PDCleaner: A multi-view collaborative data compression method for provenance graph-based APT detection systems. *Comput. Secur.* **2025**, *152*, 104359. [\[CrossRef\]](#)
14. 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. [\[CrossRef\]](#)
15. Wang, L.; Fang, L.; Hu, Y. A dynamic provenance graph-based detector for advanced persistent threats. *Expert Syst. Appl.* **2025**, *265*, 125877. [\[CrossRef\]](#)
16. 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: Piscataway, NJ, USA, 2024; pp. 3552–3570.

17. Anjum, M.M.; Iqbal, S.; Hamelin, B. ANUBIS: A provenance graph-based framework for advanced persistent threat detection. In Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, Virtual, 25–29 April 2022; pp. 1684–1693.
18. Mahmoud, M.; Mannan, M.; Youssef, A. APTHunter: Detecting advanced persistent threats in early stages. *Digit. Threat. Res. Pract.* **2023**, *4*, 1–31. [\[CrossRef\]](#)
19. Qiao, W.; Feng, Y.; Li, T.; Ma, Z.; Shen, Y.; Ma, J.; Liu, Y. Slot: Provenance-Driven APT Detection through Graph Reinforcement Learning. *arXiv* **2024**, arXiv:2410.17910. [\[CrossRef\]](#)
20. Jia, Z.; Xiong, Y.; Nan, Y.; Zhang, Y.; Zhao, J.; Wen, M. {MAGIC}: Detecting advanced persistent threats via masked graph representation learning. In Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24), Philadelphia, PA, USA, 14–16 August 2024; pp. 5197–5214. [\[CrossRef\]](#)
21. Jiang, W.; Chai, T.; Liu, H.; Wang, K.; Zhang, H. TFLAG: Towards Practical APT Detection via Deviation-Aware Learning on Temporal Provenance Graph. *arXiv* **2025**, arXiv:2501.06997. [\[CrossRef\]](#)
22. Wu, Q.; Xiang, Y. Improving the Accuracy of Anomaly Detection from System Audit Logs via Heterogeneous Provenance Graphs. In Proceedings of the 2024 International Conference on Networking and Network Applications (NaNA); IEEE: Piscataway, NJ, USA, 2024; pp. 529–535.
23. Yan, N.; Wen, Y.; Chen, L.; Wu, Y.; Zhang, B.; Wang, Z.; Meng, D. Deepto: Provenance-based APT campaigns detection via GNN. In Proceedings of the 2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom); IEEE: Piscataway, NJ, USA, 2022; pp. 747–758.
24. Aly, A.; Iqbal, S.; Youssef, A.; Mansour, E. Megr-apt: A memory-efficient apt hunting system based on attack representation learning. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 5257–5271. [\[CrossRef\]](#)
25. Bhattarai, B.; Huang, H.H. Prov2vec: Learning Provenance Graph Representation for Unsupervised APT Detection. *arXiv* **2023**, arXiv:2310.00843. [\[CrossRef\]](#)
26. Chen, T.; Dong, C.; Lv, M.; Song, Q.; Liu, H.; Zhu, T.; Xu, K.; Chen, L.; Ji, S.; Fan, Y. Apt-kgl: An intelligent apt detection system based on threat knowledge and heterogeneous provenance graph learning. *IEEE Trans. Dependable Secur. Comput.* **2022**, 1–15. [\[CrossRef\]](#)
27. Lv, M.; Gao, H.; Qiu, X.; Chen, T.; Zhu, T.; Chen, J.; Ji, S. TREC: APT tactic/technique recognition via few-shot provenance subgraph learning. In Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security, Salt Lake City, UT, USA, 14–18 October 2024; pp. 139–152.
28. Chang, H.; Rong, Y.; Xu, T.; Huang, W.; Zhang, H.; Cui, P.; Zhu, W.; Huang, J. A restricted black-box adversarial framework towards attacking graph embedding models. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 9–11 February 2020; Volume 34, pp. 3389–3396.
29. Khoury, J.; Klisura, D.; Zanddizari, H.; Parra, G.D.L.T.; Najafirad, P.; Bou-Harb, E. Jbeil: Temporal graph-based inductive learning to infer lateral movement in evolving enterprise networks. In Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP); IEEE: Piscataway, NJ, USA, 2024; pp. 3644–3660.
30. Chen, Y.; Cui, M.; Wang, D.; Cao, Y.; Yang, P.; Jiang, B.; Lu, Z.; Liu, B. A survey of large language models for cyber threat detection. *Comput. Secur.* **2024**, *145*, 104016. [\[CrossRef\]](#)
31. Li, C.; Guo, Z.; He, Q.; Xu, H.; He, K. Long-range meta-path search on large-scale heterogeneous graphs. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 44240–44268. [\[CrossRef\]](#)
32. 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, San Francisco, CA, USA, 13–17 August 2016; pp. 1035–1044.
33. Zipperle, M.; Gottwalt, F.; Chang, E.; Dillon, T. Provenance-based intrusion detection systems: A survey. *ACM Comput. Surv.* **2022**, *55*, 1–36. [\[CrossRef\]](#)
34. Huang, Z.; Wang, P. RAS-GNN: Reconstructing APT Attack Scenario Using Graph Neural Network. In Proceedings of the ICASSP 2025–2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); IEEE: Piscataway, NJ, USA, 2025; pp. 1–5.
35. Xu, L.; Zhao, Z.; Zhao, D.; Li, X.; Lu, X.; Yan, D. AJSAGE: A intrusion detection scheme based on Jump-Knowledge Connection To GraphSAGE. *Comput. Secur.* **2025**, *150*, 104263. [\[CrossRef\]](#)
36. 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), Virtual, 11–13 August 2021; pp. 3005–3022. Available online: <https://www.usenix.org/conference/usenixsecurity21/presentation/alsaheel> (accessed on 4 September 2021).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.