

The Evolution of Provenance-Based Intrusion Detection Systems: A Review of Approaches, Limitations, and Future Directions

Su Wang , Zhiliang Wang , Hongbin Sun , Zheyu Jiang , Xia Yin , Dongqi Han , Han Zhang , Xingang Shi , and Jiahai Yang 

ABSTRACT

The provenance graph, due to its ability to fully preserve historical information within a host, has been extensively studied and applied in the field of host intrusion detection. Such research is collectively referred to as provenance-based intrusion detection systems (PIDSeS). However, recent surveys on PIDSeS face limitations such as outdated literature and insufficiently comprehensive classifications. To systematically summarize the evolutionary trajectory and experimental evaluation framework of PIDSeS, this paper conducts a literature review from the perspective of PIDSeS evolution, proposing a more refined framework for host threat detection. For each fundamental step within the framework, representative PIDSeS are reviewed and introduced. Additionally, we qualitatively analyze state-of-the-art (SOTA) PIDSeS from the perspective of experimental evaluation, summarize seven key findings derived from experimental evaluations, and provide an analysis of future research directions addressing some existing limitations in the field of PIDSeS.

INTRODUCTION

As the Internet continues to evolve, cyber attackers increasingly focus on hosts containing critical data. Effectively detecting potential threats on such hosts has become a key research focus in recent years. In 2014, the U.S. Defense Advanced Research Projects Agency (DARPA) launched a project called Transparent Computing, marking the beginning of intrusion detection research using provenance graphs.¹ Subsequently, researchers have published dozens of papers focusing on provenance graphs in leading security conferences and journals. This line of research is collectively referred to as provenance-based intrusion detection systems (PIDSeS) [1]. Compared to other host-based data, provenance graphs offer advantages such as preserving historical information and facilitating causal analysis between host entities. These strengths have demonstrated superior performance in host threat detection across extensive experimental evaluations.

Given the substantial research output in the PIDSeS domain in recent years, some scholars have conducted reviews of related studies. Gong et al. [2] introduced various graph-based intrusion detection approaches from the perspective of graph structures. Zhang et al. [14] focused on intrusion detection methods targeting Advanced Persistent Threat (APT) attacks. However, these reviews have four primary limitations, as outlined below:

- **Classification of Literature.** Certain approaches, such as trace-based methods, are not independently categorized or discussed.
- **Presentation of Literature.** The evolution of PIDSeS is not adequately reflected in these reviews.
- **Experiment Coverage.** There is no systematic qualitative analysis of state-of-the-art (SOTA) literature from an experimental perspective.
- **Literature Selection.** Some SOTA works are not included in the reviews.

This paper provides a literature review from the perspective of the evolution of PIDSeS, proposing a more comprehensive host threat detection framework. For each fundamental step within this framework, representative PIDSeS are selected and reviewed. Unlike other surveys [2], [14], our review incorporates SOTA PIDSeS and reveals their evolution across three distinct categories. Furthermore, we systematically analyze PIDSeS from an experimental standpoint and present our findings in §IV. The primary contributions of this paper are as follows:

- A complete host threat detection framework is proposed, explaining the development of the PIDS research field from an evolutionary perspective.
- To the best of our knowledge, this is the first review to systematically conduct a qualitative analysis of various PIDSeS from an experimental perspective.
- The review includes an introduction to the SOTA studies in PIDS up to the present.

¹ <https://www.darpa.mil/program/transparent-computing>.

Su Wang and Hongbin Sun are with the Zhongguancun Laboratory, Beijing 100084, China; Zhiliang Wang (corresponding author), Zheyu Jiang, Han Zhang, Xingang Shi, and Jiahai Yang are with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China, and also with Zhongguancun Laboratory, Beijing 100084, China (e-mail: wzl@cernet.edu.cn); Xia Yin (corresponding author) is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China, and also with Zhongguancun Laboratory, Beijing 100084, China (e-mail: yxia@tsinghua.edu.cn); Dongqi Han is with the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100084, China.

Digital Object Identifier: 10.1109/MNET.2026.3658071
Date of Current Version: 16 September 2026
Date of Publication: 19 February 2026

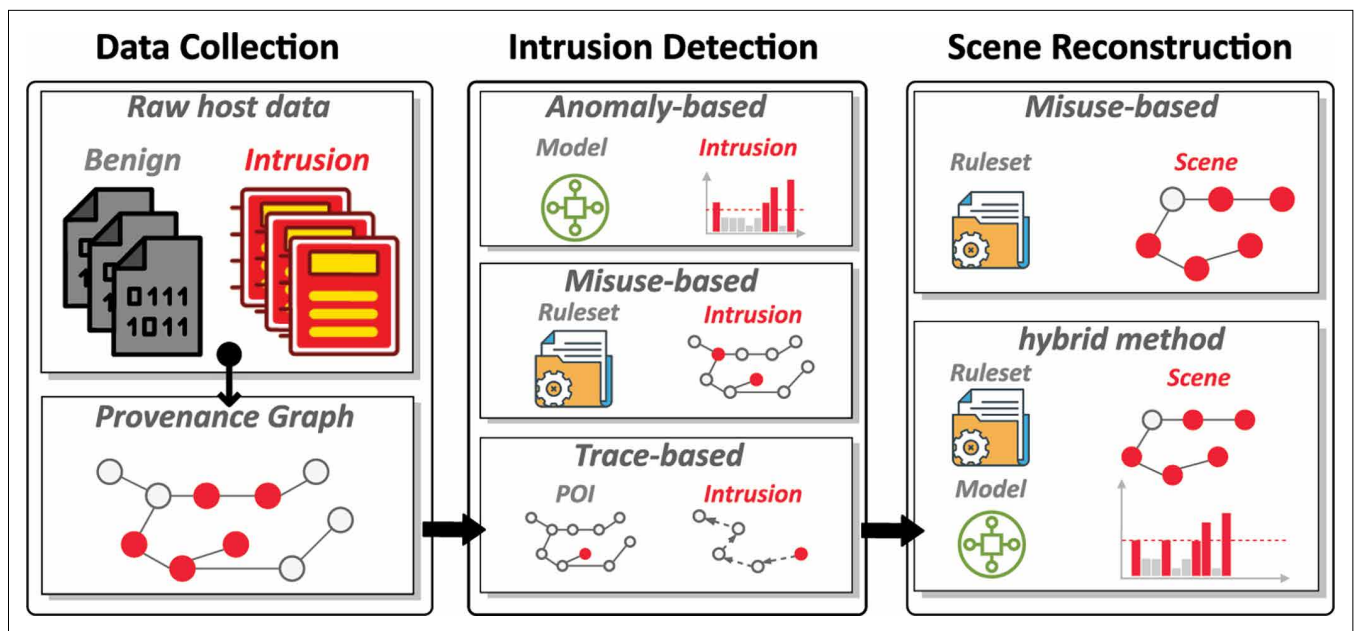


FIGURE 1. The framework of provenance-based intrusion detection.

FRAMEWORK AND PRINCIPLES

We propose a framework encompassing the three fundamental steps of host-based intrusion detection (Fig. 1). Each step in this framework integrates several categories of methods for PIDSes. We introduce the core principles behind each method and the principle guiding our literature review based on this framework.

The first and most fundamental step is data collection. A PIDS processes data in the format of a provenance graph, which is derived from raw data on the host, such as logs and system calls. Typically, a single host maintains a directed acyclic graph, referred to as a whole-system provenance graph. Nodes in the graph represent entities within the system, such as processes and files, while edges denote information flows between entities, such as read, write, or delete operations.

The second step is intrusion detection, aimed at identifying anomalous portions of the provenance graph. These could include the entire graph, specific paths, or individual nodes. This step generally involves three types of methods.

- **Anomaly-Based Methods.** These typically train a model using benign data, such as traditional machine learning models, deep learning models, or graph-based models. During detection, deviations between the model's processing results and expected benign outcomes indicate potential intrusions.
- **Misuse-Based Methods.** These rely on prior knowledge, such as a set of rules, to identify anomalous elements in the provenance graph through rule-matching techniques.
- **Trace-Based Methods.** These begin by identifying points of interest (POIs), which are often generated by the first two types of methods. By tracing backward from POIs, these methods uncover additional information related to the intrusion.

The third step is scene reconstruction, which involves correlating anomalous portions of the provenance graph to reconstruct the complete intrusion narrative. The most common approach is misuse-based, leveraging prior knowledge, such as rulesets, to establish correlations among anomalies. A second category includes hybrid methods, which integrate anomaly detection models to support scene reconstruction.

Using this framework, we conduct a literature review for each category of methods, guided by three principles. First, the selected literature must represent the seminal works in each category. Second, the literature must be published in influential cybersecurity conferences and journals, such as S&P, CCS, USENIX Security (abbreviate it as "USENIX" in the following text), NDSS, TIFS, and TDSC. Finally, the literature must include the latest research advancements.

EVOLUTION OF PIDSes

We introduce the SOTA works in the development and evolution of PIDSes based on three categories: misuse-based, anomaly-based, and trace-based. The overall evolution trend is illustrated in Fig. 2.

Holmes [8] is the first misuse-based method for threat matching on provenance graphs. The team tested their detection method in DARPA's attack-defense exercise. Holmes targets APTs, dividing the kill chain into seven stages (e.g., Initial Compromise, Lateral Movement). For each stage, it uses TTP-based rules with two components: (1) specific behaviors (e.g., "a socket reads a process") and (2) preconditions (e.g., "socket IP not in whitelist"). Matching both in a provenance graph flags suspicious edges (e.g., "Untrusted Read").

Each rule has an anomaly score reflecting threat relevance. Holmes calculates a graph's overall anomaly score by multiplying the highest-scoring rule per stage. Threshold-exceeding graphs are flagged as anomalous. The top-scoring

behaviors across stages reconstruct the attack scenario, enabling both threat detection and reconstruction.

Similar to Holmes, Poirot [7] is also a misuse-based approach. The difference lies in the fact that Holmes uses a set of TTPs-based rules to detect edges in the provenance graph, **whereas Poirot's rules are derived from Cyber Threat Intelligence (CTI) reports and Indicators of Compromise (IOCs)**. Poirot shifts detection from edges to provenance subgraphs. It constructs query graphs representing APT behaviors, formalizing detection as a graph pattern matching (GPM) problem. To address GPM's NP-complete complexity, Poirot uses approximation functions and similarity metrics for efficient matching.

Misuse-based PIDSes like Holmes and Poirot initiated PIDS research, offering intuitive threat description and scalable rule-based detection. However, their reliance on prior knowledge limits unknown threat detection—a critical flaw given attackers' use of zero-day exploits. This drove the development of anomaly-based PIDSes.

Unicorn [4] is a representative work of the anomaly-based PIDSes mentioned earlier. Unlike misuse-based systems, Unicorn does not rely on prior knowledge of historical intrusions. Instead, it constructs a model of normal system behavior using benign provenance graphs. Typically, benign data is much easier to obtain than intrusion-related data.

First, Unicorn uses a graph similarity measurement method (WL kernel) to capture structural patterns from benign provenance graphs, generating feature vectors for each graph. Next, these feature vectors are processed using HistoSketch to ensure uniform dimensionality. Finally, Unicorn uses clustering methods to group the feature vectors of benign graphs, and during intrusion detection, it determines potential intrusions based on deviations across the clusters.

Unicorn also incorporates specialized designs, treating the provenance graph as a dynamically evolving structure. At each stage, it takes snapshot samples to extract feature vectors reflecting temporal changes.

In summary, Unicorn is an anomaly-based method which performs host-based threat detection at the granularity of graph-level analysis. Following Unicorn, more fine-grained detection methods have gradually been proposed.

The graph-level intrusion detection granularity, as represented by Unicorn, has limitations in accurately pinpointing intrusion locations and reconstructing threat scenarios. **ProvDetector [10] is a path-level anomaly-based PIDS that addresses these challenges.**

The first step is to identify valuable paths from the provenance graph for further analysis. Initially, ProvDetector calculates a regularity score for each edge by analyzing a large number of benign provenance graphs. It then computes the regularity score of each path as the product of the scores of its constituent edges. By converting the problem into a search for the K longest paths in the graph, ProvDetector applies a path search algorithm to find the most suspicious paths in the graph.

The second step involves processing these identified paths to prepare them for subsequent operations. ProvDetector accomplishes this by

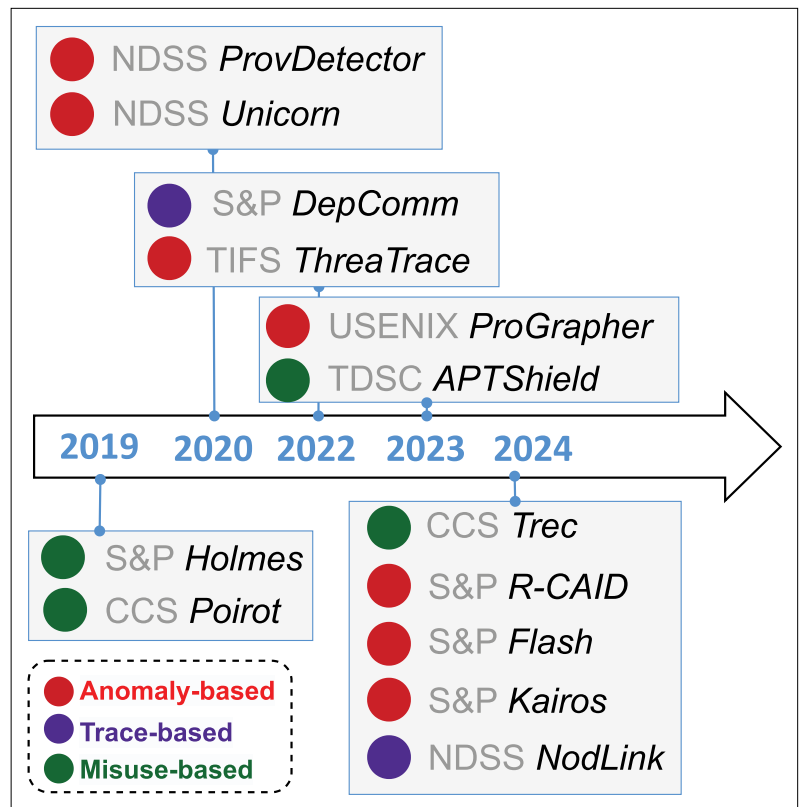


FIGURE 2. The evolution of SOTA PIDSes.

sequentially concatenating the nodes and edges of a path into a “sentence”, which is then converted into feature vectors using the doc2vec algorithm.

In the final step, anomaly detection is performed using the Local Outlier Factor (LOF) model, which identifies intrusion paths within the provenance graph.

The advent of ProvDetector refines the detection granularity of anomaly-based PIDSes from the graph level to the path level, representing a significant advancement. For security analysts, path-level alerts provide more actionable insights, enabling them to better analyze the location and nature of intrusions.

ThreaTrace [11] takes anomaly-based PIDSes a step further by refining the detection granularity to the node level. Unlike traditional methods that rely on clustering or other machine learning models, ThreaTrace employs deep learning models, specifically graph neural networks (GNNs), for learning from benign data and detecting anomalies.

First, ThreaTrace performs real-time detection by processing provenance graphs in a streaming manner. It utilizes the GraphChi C++ library to dynamically manage graph data storage across disk and memory. During the training phase, ThreaTrace uses a graph learning model to gather information from neighboring nodes for each entity in the graph. The model is trained in a self-supervised manner to capture the characteristics of benign data. To ensure comprehensive learning, ThreaTrace adopts a multi-model framework, generating multiple submodels during training and employing a probability ratio algorithm to assist in decision-making.

In the detection phase, ThreaTrace evaluates nodes using this multi-model framework. A node is considered intrusion-related if it is misclassified by all submodels.

Compared to graph-level and path-level detection methods, node-level detection granularity offers distinct advantages. It allows for the precise identification of anomalous entities within the system and enhances the ability to uncover more covert intrusion behaviors.

Trace-based methods represent a third type of intrusion detection method, alongside misuse-based and anomaly-based approaches. These methods require some known intrusion information as the starting point for threat investigation, commonly referred to as points of interest (POI). **DepComm [12] is a representative work in the field of trace-based methods.**

It begins by performing community division on the data source graph. Specifically, DepComm designs a series of novel hierarchical walk schemes to extract features from process nodes. It then uses clustering methods to classify process nodes into different communities and further classifies the resource nodes accessed by these processes into the detected communities, thereby producing process-centric communities.

The second step is community compression. DepComm calculates a process lineage tree for each community and associates each process node with the events that access resource nodes. This tree can be used to identify process-based and resource-based patterns and further merge duplicate nodes and edges to achieve community compression.

In the final step, DepComm generates several infoPaths for each community. InfoPaths are constructed from input-output node pairs within the community. Each infoPath is assigned a priority score, which is higher if the path contains a POI. Based on this approach, DepComm ultimately uncovers intrusion behaviors by identifying and analyzing infoPaths.

So far, we have introduced misuse-based PIDs represented by Holmes [8] and Poirot [7], anomaly-based PIDs represented by Unicorn [4], ProvDetector [10], and ThreaTrace [11] at the graph level, path level, and node level, respectively, as well as trace-based PIDs represented by DepComm [12]. Further on, we will present the latest research advancements in these fields from the past two years to highlight the newest breakthroughs in various types of PIDs.

APTShield [15], similar to Holmes, utilizes the TTPs from the ATT&CK model to design rules for misuse detection. It introduces several improvements in the following aspects. The first one is the data collection module. APTShield is specifically designed as a PID for Linux systems. After evaluating various log collection tools, it identifies Linux Audit as the most lightweight option that satisfies data requirements. The second one is data compaction module. APTShield compresses the collected data source graph, primarily by performing duplicate edge pruning based on causal dependencies and removing completed process nodes that are unrelated to anomalies. The third one is intrusion detection module. APTShield creates labels for processes and files. Process labels include state labels (indicating whether a process

has accessed sensitive information) and behavior labels (indicating whether a process has modified or executed files). File labels include infeasible labels and critical file labels. APTShield defines a series of label propagation rules and top-priority matching rules for various stages of the attack chain. APTShield designs specific prerequisites for certain attack scenarios, enabling effective reconstruction of the attack context.

ProGrapher [13] is an anomaly-based PID with node-level detection granularity. Its core idea is similar to Unicorn [4], starting by treating the data provenance graph as a dynamic graph composed of multiple snapshots. In each snapshot, ProGrapher uses graph2vec to generate multiple low-dimensional subgraphs, called rooted subgraphs (RSGs). It then obtains feature representations for each node on the RSGs and learns graph-level feature representations by maximizing the co-occurrence likelihood between normal snapshots and their corresponding RSGs. These snapshot features are used to train a sequence-learning model called textRCNN. During the detection phase, ProGrapher defines intrusion behaviors based on the deviation of snapshots from the model's predicted values.

Up to this point, ProGrapher and Unicorn share similar methodologies. **The difference lies in that ProGrapher, upon detecting an abnormal snapshot, further ranks the RSGs within that snapshot to identify anomalous entities related to the attack.** As mentioned earlier, ProGrapher extracts graph features by computing the co-occurrence likelihood between snapshots and RSGs. Therefore, it can identify anomalous entities by ranking the probabilities of RSGs.

The distinction between ProGrapher and ThreaTrace is that ProGrapher operates with graph-level granularity during the detection phase and drills down to node-level granularity for intrusion analysis, while ThreaTrace performs intrusion detection directly at the node level.

NodLink [5] is also an anomaly-based PID which detects intrusion in node level. Its most significant contribution and breakthrough is being the first to conduct open-world evaluation on PIDs.

During the feature extraction phase, NodLink uses natural language processing (NLP) techniques to generate vectors from information such as cmdline, filepath, and network IP. Since it focuses solely on process nodes for detection, it aggregates features exclusively for process nodes. The aggregation involves combining the intrinsic features of a process node with those of its adjacent file and network nodes.

In the detection phase, NodLink uses a reconstruction-based model to detect anomalies by comparing original and reconstructed process features.

NodLink also incorporates designs tailored for online detection, such as an importance-oriented greedy algorithm for online optimization that achieves low computational complexity with a bounded competitive ratio and a novel in-memory cache design with a scoring method to prioritize events likely to lead to APT attacks.

NodLink is deployed and evaluated in production environments, successfully detecting seven real-world attacks. Previous works primarily relied

on two evaluation methods: using public datasets or testing in simulated exercises. NodLink stands out as the first PIDS to be evaluated in actual production environments.

Kairos [1] is a node-level anomaly-based PIDS. **Compared to ThreaTrace, it introduces two key advancements: incorporating temporal information aggregation during the model training phase and adding an anomaly investigation step after anomaly detection.** The first one is designed in model training phase. Kairos uses an encoder-decoder architecture to reconstruct edges and aims to minimize the reconstruction error for benign edges. Kairos uses a time-aware graph model to encode historical behaviors, followed by a neural network to reconstruct normal interactions. The second one is designed in anomaly investigation phase. After anomaly detection, Kairos applies standard provenance graph reduction techniques and the Louvain community discovery algorithm to identify the attack scene.

It is worth noting that we do not classify Kairos as a scene reconstruction method in Figure 1 because it does not describe the attack scene. So far, anomaly-based PIDSes can only restore attack scenes but fail to describe them, which we believe renders the restoration incomplete. Achieving comprehensive scene reconstruction without relying on prior knowledge remains a significant research challenge in the PIDS field.

Compared to Kairos, **Flash [9] further incorporates the semantic information of system entities into the scope of anomaly-based PIDS.** Flash introduces three key innovations in anomaly-based PIDS: (1) temporal-aware semantic learning through enhanced word2vec embeddings of system entities (process names, file paths, etc.); (2) scalable graph processing via optimized GNN that focuses on threat-relevant edges and implements an embedding cache, achieving 10x faster traversal; and (3) advanced fine-grained detection with anomaly score ranking and Attack Evolution Graphs (AEGs) for visualizing intrusion progression, significantly extending ThreaTrace's capabilities. These design features collectively enable efficient, precise threat detection while maintaining interpretability.

So far, the anomaly-based PIDSes introduced have all adopted information aggregation-based methods to learn the information in the provenance graph. **R-CAID [3] proposes a new information aggregation method that integrates root cause analysis (RCA).** This paper hypothesizes that preserving root-cause relationships in the data source graph enhances intrusion detection and resilience against mimicry attacks. R-CAID precomputes root causes per node during graph construction and skips common ones (e.g., in GNU C Library) to reduce overhead. It uses a standard GNN for process classification, aggregating both neighbor features and root causes. Training employs self-supervised learning on process execution files.

At this point, research on anomaly-based PIDSes has evolved to node-level detection, while aggregating and analyzing semantic information, neighborhood information, root cause information, and temporal information of entities.

Current anomaly-based PIDSes are limited to identifying whether an entity within the system is

anomalous. **Trec [6] proposes a new concept: can we take a step further to detect the APT tactics / techniques present in the system?** Trec notes that while misuse-based PIDSes can achieve this task, they suffer from high false positives and reliance on predefined rules like blacklists/whitelists. To overcome the challenges of training multi-class models on APT tactics - including scarce threat samples, labeling difficulties, and overlapping benign/malicious data - Trec proposes a three-stage framework: first using anomaly detection to identify Nodes of Interest (NOIs), then applying subgraph sampling to extract APT patterns, and finally employing few-shot learning with a Siamese network to classify tactics using minimal samples.

Trec introduces a novel research direction for PIDSes, integrating both anomaly-based and misuse-based approaches to directly detect APT tactics/techniques in data graphs. This direction holds promise for further research and development in the field.

In the detection issues section of Table 1, we list the functionalities of these PIDSes in terms of specific categorization, detection granularity, scene reconstruction, temporal information, and semantic information.

EVALUATION CONCEPTS

Conducting effective and comprehensive experimental evaluations of PIDSes is a critical issue. We survey the datasets commonly used by SOTA PIDSes and summarize the qualitative evaluations conducted in the aforementioned literature.

The commonly used public datasets are shown in Table 2. We identify eight data sets that have been evaluated by the 13 PIDSes introduced in this paper. Among them, the NODLINK dataset originates from operational production data, while other portions are generated through simulated exercises such as red team/blue team engagements, with the remainder consisting of lab-environment simulations that have been open-sourced. We observe several noteworthy points:

First, the DARPA TC dataset is the most commonly used dataset. This is because it originates from DARPA's Transparent Computing project, which pioneered the use of provenance graphs for threat detection research. Additionally, its simulated environment closely aligns with real-world scenarios.

Second, some of the latest studies have made their laboratory-simulated data open-source, which undoubtedly advances the PIDS field.

Finally, we find that the usage of the StreamSpot dataset surpasses that of Unicorn SC. Upon analysis, we believe the latter surpasses the former in terms of data scale and detection difficulty. Therefore, we recommend that future PIDS-related work consider using more representative datasets for evaluation.

The detailed issues of experiment are listed in the right half of Table 1.

After statistical analysis, we identify 7 characteristics related to the experimental evaluation of the PIDSes included in this study:

- All the PIDSes conduct runtime-related experiments. These experiments typically include tests on runtime performance metrics such as speed, memory usage, and CPU utilization.

PIDS (Published in)	Detection Issues						Experiment Issues				
	Category	Detection Granularity	Scene Reconst.	Temporal Info.	Semantic Info.	Data sets	Runtime Exp.	Robust Exp.	Parameter Exp.	Ablation Study	Real-world Deployment
Holmes (2019 S&P)	Misuse	Node	✓	-	✓	Engagement	✓	✓	✓	-	-
Poirot (2019 CCS)	Misuse	Graph	✓	-	✓	Public + Simulation	✓	✓	✓	-	-
Unicorn (2020 NDSS)	Anomaly	Graph	-	✓	-	Public + Simulation	✓	-	✓	-	-
ProvDetector (2020 NDSS)	Misuse	Path	-	-	✓	Simulation	✓	✓	✓	-	-
ThreaTrace (2022 TIFS)	Misuse	Node	-	-	-	Public	✓	✓	✓	-	-
DepComm (2022 S&P)	Trace	Node	-	✓	-	Public + Simulation	✓	-	✓	-	-
APTShield (2023 TDSC)	Misuse	Node	✓	-	-	Public + Simulation	✓	-	-	-	-
ProGrapher (2023 Usenix)	Anomaly	Node	-	-	-	Public	✓	✓	✓	-	-
NodLink (2024 NDSS)	Anomaly	Node	-	-	-	Public + Simulation	✓	✓	✓	✓	✓
Kairos (2024 S&P)	Anomaly	Node	-	✓	-	Public	✓	✓	✓	-	-
Flash (2024 S&P)	Anomaly	Node	-	✓	✓	Public	✓	✓	✓	✓	-
R-CAID (2024 S&P)	Anomaly	Node	-	-	-	Public	✓	✓	✓	-	-
Trec (2024 CCS)	Misuse	Node	✓	-	-	Simulation	✓	-	✓	✓	-

TABLE 1. The Detection and Experimental Issues of State-of-the-Art PIDSes.

- Except for AptShield, all other PIDSes perform parameter experiments. We believe parameter experiments are necessary as they provide readers and users with insights into how to maximize the detection capabilities of the PIDS.
- Most PIDSes conduct robustness experiments. Robustness against evasion attacks is a crucial criterion for assessing whether a PIDS can be practically deployed. However, there is still room for improvement in robustness research on PIDSes. Among these PIDSes that have evaluated robustness, Holmes, Poirot, ThreaTrace, ProGrapher, NodLink, and Flash are not specifically designed at the methodology level to address adversarial attacks. ProvDetector makes it more difficult and resource-intensive for attackers to mimic behavior due to its path-level detection granularity. Kairos increases the difficulty for attackers to design mimicry attacks while maintaining attack logic by identifying anomalies in both structural and temporal interactions between system entities. R-CAID detects adversarial samples as anomalies due to their association with the root node through its root node embedding approach. Generally speaking, misuse-based methods utilize explicitly defined rules for threat matching, making it difficult for attackers to modify intrusion behaviors to evade rule matching without compromising their attack objectives. In contrast, anomaly-based approaches typically employ closed-box deep learning models, which create opportunities for attackers to make subtle adjustments to their intrusion activities and bypass detection by the model.
- Only three PIDSes perform ablation studies. Ablation studies are typically used to evaluate the effectiveness of individual components of a PIDS. While they are less critical to practical deployment compared to parameter, runtime, and robustness experiments, they still contribute to a more comprehensive evaluation.
- Only NodLink performs actual deployment experiment, which is one of its key contributions. Actual deployment is a critical criterion for assessing whether a PIDS can be adopted in industrial applications. We encourage researchers with the resources to deploy their proposed PIDSes in real-world environments.
- None of the PIDSes explicitly state that their experiments are conducted against specific attack scenarios. Considering that anomaly-based PIDSes [1], [3], [4], [5], [9], [10], [11], [13] only use benign data during the training phase, the threats encountered during the detection phase can be regarded as zero-day attacks. As a result, they are more effective against zero-day attacks compared to misuse-based methods [6], [7], [8], [15] that rely on prior knowledge. When dealing with long-term APT attacks, approaches employing dynamic processing techniques and coordinated disk-memory storage mechanisms [1], [3], [4], [9], [11] demonstrate greater scalability. Additionally, since none of the PIDSes incorporate specialized designs for network interfaces, their effectiveness against insider threats can be assumed to be similar.
- Several PIDSes [1], [3], [4], [9], [11] employ coordinated disk-memory storage technologies to ensure scalability under streaming workloads. This is because if the entire system's provenance graph is processed entirely in memory, the continuously expanding graph size will eventually exhaust memory capacity. In contrast, hard drives offer substantially larger storage space than

Data sets	Year	Size (GiB)	Usage Rate (%)	Authenticity	Benign/ malicious Ratio	Duration	Source Link
StreamSpot	2018	2.8	46	Simulated	4.68	-	https://github.com/sbustreamspot/sbustreamspot-data
Unicorn SC	2020	139	23	Simulated	5.03	-	https://github.com/margoseltzer/shellshock-apt
Darpa TC	2018	1,271	85	Simulated	1.28	14 days	https://github.com/darpa-20/Transparent-Computing/blob/master/README-E3.md
OpTC	2020	831.8	15	Simulated	1980.55	9 days	https://github.com/FiveDirections/OpTC-data
Malware samples	2019	0.0002	8	Simulated	5	-	https://github.com/share-we/malware
ATLAS	2020	1.1	15	Simulated	716.11	-	https://github.com/purseclab/ATLAS
Kellect4APT	2024	18.5	8	Simulated	2239.25	-	https://www.kellect.org/#/kellect-4-aptdataset
NODLINK	2024	0.55	8	Authentic	560.18	5 days	https://github.com/PKU-ASAL/Simulated-Data

Usage Rate: The percentage of the 13 papers included in this study that used this dataset.

TABLE 2. The Overview of the Commonly Used Public Dataset.

memory, thus granting these methods significantly enhanced scalability. Furthermore, approaches that perform detection on dynamically maintained subgraphs [4], [11] or within time-limited windows [1], [9] inherently constrain the upper bound of data stored in memory, thereby also achieving stronger scalability. The PIDSes capable of real-time detection [1], [3], [4], [5], [7], [8], [9], [11], [15] demonstrate superior detection latency compared to other systems limited to offline detection. Regarding the false positive rate (FPR), all evaluated PIDSes exhibit relatively high FPR. Taking the widely used DARPA TC dataset as an example, the average number of false positives exceeds 30,000.

DISCUSSION, LIMITATIONS, AND FUTURE DIRECTIONS

Despite significant breakthroughs in the field of PIDSes in recent years, several unresolved challenges remain. In this chapter, we discuss several important issues, highlight the current limitations of PIDS, and propose some potential future directions for research and development.

Trade-Off Between GNNs and Rules. GNNs have gained widespread popularity in PIDS due to their effectiveness in processing provenance graphs. Compared to traditional PIDSes that rely on rules for threat detection, GNNs offer advantages such as stronger capability in detecting unknown threats and independence from prior knowledge. However, in real-world applications, rule-based matching is often preferred due to its low resource usage and strong interpretability. We recommend supplementing GNNs with rules or model interpretation techniques to enhance their practical applicability. In practice, the academically widely studied explainable AI (XAI) can be applied to improve the transparency of GNN decisions. A commonly used technique involves training interpretable models, such as decision trees, on the same dataset. For hybrid rule-based methods, threat detection can be performed using both rule matching and anomaly detection separately. An algorithm can then be designed to comprehensively analyze the detection results from these two approaches, thereby generating a final detection outcome.

More Realistic Evaluations. Current experimental setups often rely on synthetic or outdated datasets, which may not fully capture the

complexity and diversity of modern attack patterns. Future work should focus on developing or accessing up-to-date, diverse, and realistic datasets to improve the relevance and applicability of PIDSes evaluations. To design new datasets that reflect realistic attack scenarios, we propose two recommendations: First, deploy data collection systems in actual production environments and obtain datasets containing genuine attacks through expert annotation. This approach yields datasets that most accurately mirror real-world scenarios but requires substantial human resources. Second, utilize SOTA technologies such as large language models to simulate realistic attack scenarios as authentically as possible for generating new datasets. While this method demands fewer human resources, it requires more sophisticated technical design.

Difficulty in Real-World Deployment. In real-world analyst workflows, PIDS is typically deployed in the cloud or on large servers due to its high resource usage, receiving data source logs from various hosts for analysis. Upon receiving alerts generated by PIDS, analysts usually adopt two approaches. The first is to directly forward the alerts to the corresponding hosts, leaving the host owners to conduct threat-hunting and determine whether it is a genuine alert. The second approach involves preliminary filtering, where only higher-confidence alerts are forwarded. While many PIDSes approaches demonstrate promising results in controlled environments, their application in real-world settings remains challenging. Factors such as high computational overhead, compatibility with existing systems, and the scalability of detection methods hinder practical deployment. Take ThreaTrace [11] as an example. According to the experimental section, its minimum memory usage during operation exceeds 600 MB. Our survey of several large enterprises reveals that only software with a memory usage rate below 1.6% is permitted to be deployed on internal corporate computers. The substantial resource overhead generated by the GNN model employed by ThreaTrace undoubtedly poses a significant challenge to its practical deployment. Research should prioritize lightweight, efficient, and easily integrable solutions to bridge the gap between theoretical advancements and practical adoption.

Robustness. Current systems often struggle to maintain detection accuracy in the presence of

noise, adversarial attacks, or variations in system behavior. Enhancing the resilience of PIDSEs to such disruptions is essential for ensuring reliability in diverse and dynamic environments. Future directions could include the development of advanced machine learning models that are more resistant to adversarial manipulation and noise.

Threat Model. Since this survey exclusively covers works that employ provenance graphs for host intrusion detection, their threat models share fundamental similarities. We summarize two primary common assumptions across these studies:

- Integrity of provenance graph collection and generation: The system correctly captures and maintains unaltered source data.
- Attack persistence in host activities: Malicious behaviors inevitably leave traces in the host environment.

By addressing these limitations, PIDSEs can move closer to achieving its full potential, enabling more effective and widespread use in combating modern cyber threats.

CONCLUSION

This paper provides a comprehensive review of representative works and recent advancements in the field of provenance-based intrusion detection systems (PIDS). A complete host threat detection framework is proposed, comprising three key steps: data collection, intrusion detection, and scene reconstruction. We systematically introduce these works from an evolutionary perspective, highlighting the developmental trajectory of the PIDS research domain. To the best of our knowledge, this is the first review to conduct a qualitative analysis of various PIDSEs from an experimental perspective, including an in-depth discussion of the datasets commonly used in evaluations and the methodologies employed by SOTA studies up to the present. Finally, we examine the current limitations of PIDSEs and propose key future research directions: developing lightweight, efficient, and integrable solutions for practical adoption; creating up-to-date, diverse datasets to enhance evaluation relevance; and designing robust machine learning models resistant to adversarial attacks and noise.

ACKNOWLEDGMENTS

This work was supported by Zhongguancun Laboratory.

REFERENCES

- [1] Z. Cheng et al., "Kairos: Practical intrusion detection and investigation using whole-system provenance," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2024, pp. 3533–3551.
- [2] Y. Gong et al., "Graph-based insider threat detection: A survey," *Comput. Netw.*, vol. 254, Dec. 2024, Art. no. 110757.
- [3] A. Goyal, G. Wang, and A. Bates, "R-CAID: Embedding root cause analysis within provenance-based intrusion detection," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2024, pp. 3515–3532.
- [4] X. Han et al., "Unicorn: Runtime provenance-based detector for advanced persistent threats," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–19.
- [5] S. Li et al., "NODLINK: An online system for fine-grained APT attack detection and investigation," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2024, pp. 1–18.
- [6] M. Lv et al., "TREC: APT tactic/technique recognition via few-shot provenance subgraph learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Dec. 2024, pp. 139–152.
- [7] S. M. Milajerdi et al., "POIROT: Aligning attack behavior with kernel audit records for cyber threat hunting," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2019, pp. 1795–1812.
- [8] S. M. Milajerdi et al., "HOLMES: Real-time APT detection through correlation of suspicious information flows," in *Proc. IEEE Symp. Security Privacy (S&P)*, May 2019, pp. 1137–1152.
- [9] M. Ur Rehman, H. Ahmadi, and W. Ul Hassan, "Flash: A comprehensive approach to intrusion detection via provenance graph representation learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2024, pp. 3552–3570.
- [10] Q. Wang et al., "You are what you do: Hunting stealthy malware via data provenance analysis," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–17.
- [11] S. Wang et al., "THREATTRACE: Detecting and tracing host-based threats in node level through provenance graph learning," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 3972–3987, 2022.
- [12] Z. Xu et al., "DEPCOMM: Graph summarization on system audit logs for attack investigation," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 540–557.
- [13] F. Yang et al., "Prographer: An anomaly detection system based on provenance graph embedding," in *Proc. 32nd USENIX Secur. Symp. (USENIX Secur.)*, 2023, pp. 4355–4372.
- [14] B. Zhang et al., "A survey on advanced persistent threat detection: A unified framework, challenges, and countermeasures," *ACM Comput. Surveys*, vol. 57, no. 3, pp. 1–36, Mar. 2025.
- [15] T. Zhu et al., "APTSHIELD: A stable, efficient and real-time APT detection system for linux hosts," *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 6, pp. 5247–5264, Nov. 2023.