Building Next-Generation Datasets for Provenance-Based Intrusion Detection

Qizhi Cai*§, Lingzhi Wang†§, Yao Zhu*, Zhipeng Chen*, Xiangmin Shen‡ and Zhenyuan Li*
*Zhejiang University, †Northwestern University, ‡Hofstra University
{22551204,22321169,22551132,lizhenyuan}@zju.edu.cn,
lingzhiwang2025@u.northwestern.edu, xiangmin.shen@hofstra.edu

***Abstract*—In recent years, provenance-based intrusion detection and forensic systems have attracted significant attention, leading to a rapid growth of related research efforts. However, progress in this area has been hindered by the long-standing lack of updated datasets and benchmarks. Existing datasets suffer from several critical limitations, including outdated attack techniques, short temporal scales, and incomplete or fragmented attack chains. As a result, they fail to capture the characteristics of the latest, real-world Advanced Persistent Threat (APT) attacks. Moreover, the unclear, coarse-grained attack procedures underlying existing datasets make accurate labeling and reliable evaluation difficult. Consequently, the absence of a comprehensive, up-to-date dataset has become a major bottleneck for the progress of this area. To address this, we present our efforts in building a large-scale, diverse, and well-annotated dataset for provenance-based intrusion analysis. Our dataset is generated using an automated attack emulation framework that incorporates recent attack techniques and supports fine-grained ground-truth labeling. Using this dataset, we conduct a comprehensive evaluation of state-of-the-art provenance-based intrusion detection systems, revealing weaknesses that cannot be effectively benchmarked with existing datasets. Our results demonstrate the dataset's value in enabling clearer, more informative evaluations and highlight its potential to advance future research in provenance-based intrusion detection and graph-based security analysis.**

I. Introduction

The past decade has witnessed joint efforts on research in provenance-based intrusion detection, which revealed the growing importance of fine-grained system activity logs for understanding complex attack behaviors. However, the work is hindered by the lack of up-to-date and high-quality datasets. Existing benchmarks, such as DARPA Transparent Computing [1] [2] and OpTC [3], ATLAS [4] and ATLASv2 [5], exhibit several shortcomings. First, some datasets are outdated and mainly record attack scenarios that are no longer representative of contemporary threat landscapes. Second, most of them provide limited information about both attack and expected behaviors, and lack fine-grained ground truth labeling. As a result, it is difficult to conduct a comprehensive analysis and unbiased evaluation of provenance-based intrusion detection systems (PIDS) using these benchmarks.

Additionally, recent studies have pointed out that many ML and GNN-based approaches tend to overfit [6], achieving high performance on these restricted benchmarks but failing to generalize to new or more complex attack scenarios. This overfitting is primarily due to limited scale, incomplete attack coverage, and insufficient benign activity representation in existing datasets. These limitations motivate the need for a modern, high-quality, and comprehensive benchmark that captures diverse attack behaviors and extensive benign activity, enabling more reliable training and evaluation of provenance-based intrusion detection methods.

Constructing a high-quality dataset for PIDS involves several challenges. First, an effective dataset must incorporate up-to-date and realistic attack scenarios, as well as high-fidelity normal user behaviors. This requirement raises two non-trivial issues: how to systematically acquire evolving attack techniques and security knowledge and translate them into attack traces in the dataset, and how to collect or synthesize normal system activities that closely resemble real system behaviors across diverse scenarios. Second, a high-quality dataset should provide sufficient contextual details for both attack and benign activities, along with fine-grained ground truth annotations. Rather than merely enabling the reporting of aggregate performance metrics, such datasets should facilitate in-depth behavioral analysis by offering detailed contextual information at the level of system events or objects. This level of granularity is essential for developers to understand detection outcomes, diagnose failure cases, and iteratively improve PIDS models. Third, practical datasets should support a certain degree of customization to accommodate diverse research objectives. For example, they should allow the inclusion of attacks at different scales and complexities, as well as the adjustment of the intensity and frequency of both malicious and benign behaviors. Such flexibility is crucial for evaluating system robustness under varied and realistic conditions.

This paper presents initial progress toward creating the aforementioned dataset for PIDS. Specifically, we make the following contributions:

- First, we review existing PIDS datasets and summarize their key limitations.

§The first two authors made equal contribution.

Workshop on Attack Provenance, Reasoning, and Investigation for Security in the Monitored Environment (PRISM) 2026
23 February 2026, San Diego, CA, USA
ISBN 978-1-970672-00-8
https://dx.doi.org/10.14722/prism.2026.23021
www.ndss-symposium.org

- Second, we propose two new evaluation metrics to provide real and insightful measurements of the performance. We discuss the corresponding requirements on fine-grained ground truth annotations needed to support these metrics.
- Finally, leveraging a state-of-the-art automated attack emulation framework, we construct a small-scale dataset and evaluate eight representative PIDS on this dataset. Through an analysis of the experimental results, we identify several open challenges and outline directions for future research.

II. Background and Related Work

A. Advanced Persistent Threats

Advanced Persistent Threats (APTs) represent a dominant class of modern cyber attacks, characterized by well-resourced adversaries seeking to infiltrate target systems, maintain long-term access, and exfiltrate sensitive information while evading detection. These attacks typically unfold through multiple stages, such as reconnaissance, privilege escalation, lateral movement, and data exfiltration, whose modular structure allows attackers to adapt their behavior and blend malicious activities into normal system workflows. From a system provenance perspective, APTs manifest as long-range causal chains that span processes, files, network connections, and other system entities over extended periods of time, posing significant challenges for detection and forensic reconstruction. A representative example is the SolarWinds supply-chain attack [7], [8] disclosed in late 2020, in which a malicious backdoor was stealthily distributed through legitimate software updates and remained dormant for an extended period before enabling covert command-and-control and data exfiltration.

B. Provenance-based Intrusion Detection

Provenance, rooted in data lineage analysis [9], aids intrusion detection by tracing data and control flow to identify potential attack paths. It analyzes system entities (e.g., processes, files) and their interactions (e.g., reads, forks). Tools using eBPF [10] or ETW [11] collect audit logs and parse them into *(subject, operation, object, timestamp)* events, which form temporal graphs via shared nodes. Recent work extends this to serverless functions and web applications [12], [13], [14]. Provenance graph-based detection schemes [15], [16], [17] have shown efficacy in semantic attack analysis, behavioral correlation, and interpretability.

Rule-based approaches. SLEUTH [18] and MORSE [19] use tags and rules to condense event histories to track interactions and trigger alarms; HOLMES [20] builds scenario graphs mapped to attack lifecycles. Recent work improves adaptability and semantics of the rules: CAPTAIN [21] introduces differentiable tag propagation with gradient-based tuning, and CONGRAPH [22] combines provenance graphs with process-context attributes for stealthy attack detection.

Embedding-based approaches. These methods analyze the provenance graphs using sketching [23], histogram vectorization [24], [25], self-supervised learning [26], variational autoencoders [17], and GNNs [27], [28], [29]. For example, NODLINK [17] and PROVDETECTOR [30] detect anomalies via graph traversal and rare path feature vectors. KAIROS [31] use a GNN encoder-decoder to model temporal graph evolution and produce fine-grained scores, while R-CAID [28] leverages attention over transitive-closure pseudo-graphs for long-range semantic reasoning. Challenges include embedding computation on large logs. PROV2VEC [32] embeds behaviors via graph kernels, and IDS-HGAT [33] models heterogeneous node/edge semantics based on a heterogeneous graph attention network.

C. Evaluation of Existing Datasets

Early provenance-based datasets with a small scale are basically used for proofs-of-concept and experimental research. For example, the PASS system (2006) [34] focused on storage-level file operations, Hi-Fi Provenance (2012) [35] captured operating system-level events for casual analysis, and SPADE (2012) [36] further extended system-level event capture to support more comprehensive graph construction. However, these datasets were generally small-scale and not publicly available, which limited their utility for evaluating intrusion detection methods in realistic environments. As the research advances, the community has developed larger-scale, publicly available, attack-annotated datasets. With the release of large-scale, publicly available provenance datasets such as DARPA Transparent Computing Engagements, OpTC, and ATLAS, provenance-based intrusion detection research has gained access to standardized evaluation corpora. At the same time, prior analyses have reported several limitations in these datasets that are relevant to their use in experimental evaluation [37].

Specifically, DARPA TC datasets rely heavily on scripted and repetitive benign workloads, which may oversimplify normal system behavior. The OpTC dataset, while more operationally realistic, contains substantial background noise and undocumented distribution shifts between training and testing data. ATLAS and its extensions focus on well-defined attack engagements; however, the conspicuousness of attack behaviors and the granularity of ground-truth labeling vary widely across datasets. More generally, existing datasets often contain experimental artifacts such as logging gaps, workload reconfiguration, and machine downtime, which introduce distributional discontinuities. In addition, ground-truth labeling practices differ across datasets, with varying levels of completeness and precision in the annotation of malicious entities and activities. PROVSYN [38] synthesizes realistic provenance graphs, including structure, semantics, and timing, to augment training data and mitigate class imbalance, improving the robustness of downstream detectors. However, since it only augments the dataset based on the existing data, it cannot incorporate new patterns of attack and benign behaviors.

Some researchers have investigated adversarial attacks against PIDS, in which attackers manipulate system behavior to evade detection. For example, adversaries may inject benign-looking events or structures to distort learned embeddings [39], or construct gadget chains that replace a single suspicious event with a sequence of normal operations [40],

thereby reducing attack saliency. These approaches modify existing datasets to simulate the execution traces produced by adversarial attacks and use them to evaluate attack effectiveness. However, to the best of our knowledge, no dataset currently contains real-world execution traces of adversarial attacks against PIDS.

III. New Metrics

A. Time-Weighted Detection Accuracy for Real-Time APT Detection

Existing metrics for intrusion/APT detection commonly quantify performance via unweighted detection rates, e.g., precision/recall computed over labeled detection targets such as nodes/edges in a provenance graph, time windows, or entire attack instances. A key limitation of these metrics is that they implicitly assign identical importance to all targets, regardless of when they occur in the attack. This assumption is misaligned with real-world APT detection and response, especially in the real-time setting: detecting an attack artifact early often enables containment and prevents downstream damage, whereas a correct alert triggered only in late stages may have substantially lower operational value because the harm has largely materialized. Our insight is that evaluation should reflect this asymmetric value by explicitly discounting late-stage detections and prioritizing early-stage detections. To this end, we propose a time-weighted variant of object-level precision and recall.

Specifically, let $\mathcal{I}$ denote the set of detection targets (e.g., nodes or edges), where each target $i \in \mathcal{I}$ has a timestamp t_i, a binary ground-truth label ($y_i \in \{0,1\}$), and a binary prediction ($\hat{y}_i \in \{0,1\}$). We define a non-increasing weighting function ($w(t_i) \geq 0$) that assigns larger weights to earlier targets and smaller weights to later ones. The time-weighted recall and time-weighted precision are then:

$$R_w = \frac{\sum_{i\in\mathcal{I}} w(t_i)\mathbb{I}[y_i = 1 \wedge \hat{y}_i = 1]}{\sum_{i\in\mathcal{I}} w(t_i)\mathbb{I}[y_i = 1]} \tag{1}$$

$$P_w = \frac{\sum_{i\in\mathcal{I}} w(t_i)\mathbb{I}[y_i = 1 \wedge \hat{y}_i = 1]}{\sum_{i\in\mathcal{I}} w(t_i)\mathbb{I}[\hat{y}_i = 1]} \tag{2}$$

In practice, $w(\cdot)$ can be instantiated using simple, interpretable discounting schemes, such as linear discount $w(t) = 1 - \tau(t)$, where $\tau(t) \in [0,1]$ is a normalized time index, or exponential discount $w(t) = \exp(-\lambda\tau(t))$, which more strongly penalizes late detections and admits a clear "half-life" interpretation for operational response windows. This metric family preserves the familiar semantics of precision/recall while aligning evaluation with the fundamental objective of real-time APT defense: maximizing the utility of early correct detections.

B. TTP Coverage at a Quality Threshold

Prior object-based evaluation metrics for PIDS typically report aggregate precision/recall (or detection rate) over all labeled detection targets (e.g., nodes, edges, or time windows). While useful, such metrics provide little insight into **which** adversary behaviors are being detected reliably. In practice, defenders often prefer detectors with broader detection surface, i.e., systems that accurately cover a wider range of MITRE ATT&CK Tactics, Techniques, and Procedures (TTPs), over systems that perform well only on a narrow subset. Aggregate object-level metrics can mask this limitation: a detector may achieve high overall recall by excelling on a few frequent TTPs while consistently failing on many others, yet this failure remains invisible in a single global score. To explicitly quantify behavioral coverage, we propose *TTP Coverage@threshold*, which measures the fraction of TTPs for which the detector meets a required accuracy criterion. Let $\mathcal{T}$ be the set of ATT&CK techniques or sub-techniques under consideration. For each detection target $i \in \mathcal{I}$, we assume a ground-truth TTP label list $t(i) \in \mathcal{T}$. Let $y_i \in \{0,1\}$ denote whether i is malicious (positive) and $\hat{y}_i \in \{0,1\}$ denote the detector's decision. For each $t \in \mathcal{T}$, we compute a per-TTP quality score Q_t (e.g., $F\beta, t$, or recall/precision), derived from per-TTP precision and recall:

$$P_t = \frac{\sum_{i\in\rangle} \mathbf{1}\{t(i) = t \wedge y_i = 1 \wedge \hat{y}_i = 1\}}{\sum_{i\in\mathcal{I}} \mathbf{1}\{t(i) = t \wedge \hat{y}_i = 1\}} \tag{3}$$

$$R_t = \frac{\sum_{i\in\mathcal{I}} \mathbf{1}\{t(i) = t \wedge y_i = 1 \wedge \hat{y}_i = 1\}}{\sum_{i\in\mathcal{I}} \mathbf{1}\{t(i) = t \wedge y_i = 1\}} \tag{4}$$

Given a user-specified quality threshold $\tau \in [0,1]$, we define TTP Coverage@threshold as the proportion of TTPs whose per-TTP quality meets the threshold:

$$\text{Coverage@}\tau \;=\; \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathbf{1}\{Q_t \geq \tau\}, \text{where } Q_t \in \{P_t,\ R_t\}. \tag{5}$$

Optionally, to prevent degenerate cases where high recall is achieved with unacceptably low precision (or vice versa), we also consider a two-constraint coverage definition:

$$\text{Coverage@}(\tau_P, \tau_R) \;=\; \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathbf{1}\{P_t \geq \tau_P \wedge R_t \geq \tau_R\}. \tag{6}$$

This metric directly captures the detector's effective behavioral surface under a minimum quality bar, complementing aggregate object-level scores and enabling fine-grained analysis of which ATT&CK behaviors are detected reliably versus those that remain blind spots. Knowledge-driven provenance analysis has also shown that ATT&CK-aligned detection can be operationalized by matching behavior templates/subgraphs against execution provenance, producing interpretable evidence for technique-level detection [41].

C. Requirements for the Dataset.

Computing Time-Weighted Detection Accuracy and TTP Coverage@threshold requires more fine-grained dataset labeling. In addition to the object-level ground truth, such as attack-related nodes and edges, for time-weighted evaluation, each object must be associated with a timestamp on the attack timeline. For TTP-aware evaluation, each object must additionally carry a multi-label annotation indicating the corresponding

MITRE ATT&CK behavior(s), i.e., a set of labels $\mathcal{T}_i \subseteq \mathcal{T}$ for each object i. In practice, this implies that the dataset must include (i) a structured representation of attack activity as objects (nodes/edges) with stable identifiers, (ii) per-object temporal metadata, and (iii) per-object ATT&CK label sets. Many existing datasets do not satisfy these requirements, which motivate the need for datasets with fine-grained, temporally ordered, and ATT&CK-aligned ground truth to enable meaningful real-time and behavior-centric evaluation of PIDS.

IV. Dataset Generation

A high-quality dataset for PIDS must satisfy several requirements. First, it should ensure complete and causally consistent event capture. All security-relevant system activities must be recorded without loss, sampling, or filtering, while preserving fine-grained temporal and causal order. Missing or reordered events can break provenance chains, distort graph structures, and significantly impair both detection accuracy and root-cause analysis. Second, a representative dataset must provide realistic, benign workloads and up-to-date attacks. Benign traces should include long-running, heterogeneous system activities, such as user interactions, background services, and application workflows, to reflect natural system variance and avoid overfitting. Meanwhile, attack traces should span multiple stages to better approximate real-world intrusion campaigns. Finally, reliable evaluation requires precise ground truth. Fine-grained labels at the event or entity level are essential for assessing and analyzing detection performance. These requirements together form the basis for constructing and evaluating robust provenance-based intrusion detection systems.

A. Generation of Attack Traces

A high-quality PIDS dataset should include up-to-date and diverse APT attack samples. To this end, we employ AURORA, an automated cyberattack emulation system, to systematically generate and execute such attack scenarios. AURORA collects TTP implementations from multiple sources, including the MITRE ATT&CK knowledge base, threat emulation frameworks such as Atomic Red Team [42], and converts them into structured Attack Actions. It then provides a planning-driven orchestration mechanism that composes individual attack TTPs into coherent attack chains. Specifically, it models an attack action as the minimal executable unit in an attack plan. It uses the Attack Action Linking Model, an extensible set of PDDL predicates, to formally describe the preconditions and effects of each action, thereby enabling the automated linking of actions into valid multi-step attack chains [43]. It supports generating customized attack chains for different emulation scenarios and provides the Python scripts to execute the generated attacks.

B. Generation of Benign Traces

Our benign traces are designed to capture a broad spectrum of realistic day-to-day activities.

We explore two complementary approaches for benign data collection. First, we recruit human volunteers to emulate typical user behaviors in a controlled environment. Beyond common activities such as browsing TikTok, downloading files, visiting webpages, checking emails, and playing video games, participants also perform a variety of everyday user and system workflows, including document editing, software installation and updates, web-based communication, local file management, and media playback. Together, these activities induce diverse process, file, and network interactions, enabling us to construct a representative and robust benign baseline. Second, we leverage a computer-interactive large language model (LLM) agent, referred to as a Computer-Using Agent (CUA) [44], to simulate human-like interactions with the operating system. The CUA autonomously performs routine tasks such as web browsing and office software manipulation, generating benign traces that are consistent yet scalable. This approach complements human-driven traces by providing reproducible and diverse benign behaviors without requiring continuous and expensive human involvement.

C. Data Collection and Formalization

After collecting raw ETW audit logs, we formalize them into a unified provenance representation to support subsequent graph construction and intrusion detection. Specifically, we convert the raw CSV-form ETW records into a line-delimited JSON stream consisting of node and event objects, where nodes represent system entities (e.g., processes, files, registry keys, and network flows) and events represent typed interactions between entities.

We first load the raw ETW CSV records and retain only the entries that correspond to the structured payload format used by our collector (i.e., records tagged as `ProtoBuf`). For each record, we normalize fields into a canonical tuple *(Time, EventType, processID, threadID, Timestamp, EventDetails)*. The `EventType` is derived by mapping the ETW provider/object name and opcode to a semantic event type using an external `format.json`. This step abstracts away ETW-specific identifiers and yields a compact set of security-relevant event types (e.g., `ProcessStart`, `ProcessEnd`, `FileIoRead`, `FileIoWrite`, `ImageLoad`, and network/registry events).

To build a consistent provenance graph, we assign each observed entity a globally unique node identifier and maintain an in-memory index that maps canonical entity identifiers to node IDs. We use type-specific identifiers to deduplicate entities across events: processes are keyed by PID (e.g., `PID:pid`), files are keyed by full path (e.g., `File:path`), registry keys are keyed by key path (e.g., `Key:path`), and network flows are keyed by a 5-tuple derived from the ETW payload (source/destination IP and ports). Whenever an event references a previously unseen entity, we emit a `NODE` record and update the index; otherwise, we reuse the existing node ID. Each node contains a generated UUID and a typed attribute set (e.g., PID/PPID/process name/command line for processes, path for files/keys, and IP/port fields for netflows).

Each normalized ETW record is translated into a provenance EVENT record with the schema *(id, time, type, s, d, d2)*. Here, *s* denotes the subject entity (typically a process), *d* denotes the primary object entity (e.g., a file, a netflow, or a registry key), and *d2* optionally denotes a secondary object when required by the event semantics (e.g., *FileIoRenamePath*, where the event links an old path to a new path). The event time is taken from the ETW timestamp field, and each event is assigned a fresh UUID for traceability. For events indicating that data collection starts on pre-existing entities (e.g., *ProcessDCStart* and *ImageDCStart*), we treat them as graph initialization signals: the former adds a pre-existing process node, while the latter may introduce a pre-existing image file and emits a corresponding *ImageLoad* edge to preserve the implied dependency.

We output the formalized provenance in a streaming, line-delimited JSON representation. Each record is encoded as a *NODE* or *EVENT* object with a corresponding *logData* payload, which provides a compact and append-friendly interface for downstream provenance-graph construction and analysis.

TABLE I
DETECTION RESULTS ON PROVATTACK DATASETS

System	Precision(%)	Recall(%)	TP	FP
PROVATTACK1				
FLASH	0.020	55.00	11	55080
KAIROS	0.000	0.000	0	0
MAGIC	0.080	**100.0**	20	24843
NODLINK	0.011	5.000	1	9219
ORTHRUS	0.000	0.000	0	9
R-CAID	0.000	0.000	0	0
THREATRACE	0.023	**100.0**	20	88235
VELOX	0.000	0.000	0	0
PROVATTACK2				
FLASH	0.266	34.09	15	5632
KAIROS	0.000	0.000	0	0
MAGIC	1.985	**95.46**	42	2074
NODLINK	0.986	29.55	13	1306
ORTHRUS	**36.36**	9.091	**4**	**7**
R-CAID	0.000	0.000	0	0
THREATRACE	0.332	100.0	44	13202
VELOX	0.000	0.000	0	0

V. ANALYSIS OF THE GENERATED DATA

A. Evaluation Setup.

To evaluate provenance-based intrusion detection under controlled yet realistic multi-stage attack scenarios, we construct two experimental datasets, ProvAttack1 and ProvAttack2, derived from the same benign execution environment but injected with different attack chains.

Datasets. Both datasets are generated on identical system configurations and share the same benign workload traces. The only difference lies in the injected attack behavior. Specifically, ProvAttack1 contains the 11th attack chain defined in the Aurora repository, while ProvAttack2 contains the 21st attack chain. Each attack chain is executed once in isolation to avoid interference between attack instances.

Attack Execution. Attack traces are generated using Aurora's automated attack orchestration framework. Aurora executes attack chains defined as ordered compositions of attack techniques, specified in YAML-based attack chain descriptions. These descriptions explicitly encode the sequence of attack actions and their execution dependencies, ensuring deterministic and reproducible attack execution.

Ground Truth Annotation. Ground truth labels are derived directly from the corresponding Aurora attack chain YAML files. For each dataset, we identify all system entities (processes, files, registry keys, and network objects) that are directly involved in the execution of attack actions. These entities are marked as malicious ground truth. Importantly, we do not label entire causal neighborhoods or transitively affected entities; only nodes explicitly participating in the attack chain are annotated. This conservative labeling strategy avoids ambiguity and ensures precise evaluation of detection and root-cause identification.

Detection Systems. To evaluate our dataset under representative provenance-based intrusion detection pipelines, we adopt the open-source evaluation framework provided by Tristan Bilot et al. [6]. Specifically, we integrate our datasets into PIDSMaker, the unified benchmarking framework open-sourced by the authors, which implements a diverse set of state-of-the-art PIDS with standardized preprocessing, graph construction, and evaluation procedures. Following the original study, we evaluate our data using the same eight detection systems including FLASH [27], KAIROS [31], MAGIC [26], NODLINK [17], ORTHRUS [45], R-CAID [28], THREATRACE [46] and VELOX [6].

B. Detection Performance and Analysis

We perform node-level detection; Table I shows the results. The evaluation on PROVATTACK datasets reveals detection behaviors highly consistent with prior work on DARPA and OpTC datasets [6]. We observe that zero precision at node-level detection is not an anomaly but a recurring outcome across multiple provenance-based intrusion detection systems. Although some systems achieve near-perfect recall on the PROVATTACK datasets, their alerts are overwhelmingly dominated by false positives, rendering the detection results impractical for real-world forensic analysis. In contrast, ORTHRUS remains one of the few systems capable of producing compact and high-precision alert sets under node-level evaluation on PROVATTACK. Meanwhile, several systems fail to raise any actionable node-level alerts at all, further highlighting the sensitivity of threshold-based detection pipelines to dataset-specific characteristics.

From the perspective of benchmark construction, the observed zero precision and extremely high false-positive rates can be attributed to multiple factors. First, all systems are evaluated using the default hyperparameters provided in the PIDSMaker framework, without extensive dataset-specific tuning. While this choice ensures consistency and reproducibility across systems, it may lead to suboptimal performance for

specific models whose behavior is highly sensitive to hyperparameter settings. Recent advances have started to explicitly address this sensitivity by making rule parameters differentiable and tuneable with gradient-based optimization, enabling lightweight adaptation without redesigning detection logic [21]. A more thorough hyperparameter optimization could potentially improve detection quality, although such tuning is non-trivial and often requires access to test data, which may introduce data leakage concerns.

Second, the representativeness of the training data plays a critical role in detection performance. If the training data does not sufficiently cover the diversity of benign system behaviors that appear during testing, models may incorrectly assign high anomaly scores to previously unseen but benign activities, resulting in a large number of false positives. This issue is particularly pronounced in provenance-based intrusion detection, where benign behavior spaces are large, evolving, and difficult to capture exhaustively.

This issue highlights the importance of better benign behavior coverage and train–test alignment, which we discuss as a direction for future benchmark extension.

VI. Discussion & Future Work

A. Experimental Design and Evaluation Sensitivity

The current study is subject to limitations related to experimental design and evaluation scope. While this work provides an initial and systematic evaluation of provenance-based intrusion detection systems on the PROVATTACK benchmark, several evaluation dimensions outlined earlier in the paper have not yet been fully realized. In particular, the experiments corresponding to III-A and III-B remain to be designed and executed. As a result, the present evaluation primarily focuses on static, post-hoc detection outcomes, without explicitly assessing detection timeliness or technique-level coverage under practical alerting constraints.

In addition, detection performance can be sensitive to experimental configurations such as hyperparameter settings. In the current evaluation, all systems are assessed using default hyperparameters provided by the PIDSMaker framework, without extensive dataset-specific tuning. While this choice ensures consistency and avoids data leakage, it may prevent certain systems from operating at their optimal detection points.

Together, these factors indicate that some observed detection behaviors may reflect limitations of the current experimental setup rather than fundamental weaknesses of the underlying detection approaches. At the same time, they highlight the challenges of designing comprehensive and fair experimental evaluations for provenance-based intrusion detection systems.

B. Scale and Diversity of Attack Traces

Another limitation of the current benchmark lies in the scale and diversity of attack traces. Both PROVATTACK1 and PROVATTACK2 contain a single attack chain generated using the Aurora framework, and each attack unfolds over a relatively short execution window. This design allows for controlled and interpretable evaluation, but it restricts the scope of attack scenarios represented in the benchmark.

As a result, the current evaluation primarily reflects system behavior under isolated, short-lived attack conditions. It does not assess detection performance in more complex scenarios involving multiple concurrent or temporally overlapping attack chains, nor does it capture long-running or stealthy attacks that evolve gradually over extended periods. Such scenarios are common in real-world APT campaigns and may pose additional challenges for provenance-based detection systems.

C. Benign Behavior Coverage and Data Editing

A further limitation of the current benchmark lies in the representativeness of benign training data. Due to the session-based nature of provenance capture, benign behaviors are often fragmented across isolated execution windows. As a result, training data may not sufficiently cover the full range of normal system behaviors that appear during testing, even when both are collected from the same host. This mismatch can lead detection systems to flag previously unseen but benign activities as anomalous incorrectly.

This issue is inherent to provenance-based datasets, where entity identifiers are session-scoped, and causal chains are frequently truncated by capture boundaries such as reboots or deployment constraints. Simply aggregating benign logs in temporal order does not resolve this problem, as semantic continuity across sessions remains broken.

D. Future Work

To address above limitations, we will extend PROVATTACK along key experimental dimensions as future work. We plan to (i) complete the evaluation of time-weighted detection accuracy and technique-level coverage, and conduct systematic hyperparameter sensitivity analyses using training/validation data only; (ii) incorporate longer and more realistic traces, including multi-stage attack chains and their variants, to assess robustness and scalability beyond isolated instances; and (iii) improve benign behavior coverage via data editing and expansion, such as canonicalizing session-specific entity identifiers into stable identities and selectively merging benign logs from the same host, so that learned normality better reflects long-running cross-session execution and reduces spurious false positives.

References

[1] "Transparent-Computing," 2023, https://github.com/darpa-i2o/Transparent-Computing.

[2] D. T. program, "Transparent Computing Engagement 3 Data Release," 2020. [Online]. Available: https://github. com/darpa-i2o/Transparent-Computing/blob/master/README-E3.md

[3] "Operationally transparent cyber (optc) data release," 2020. [Online]. Available: https://github.com/FiveDirections/OpTC-data

[4] A. Alsaheel, Y. Nan, S. Ma, L. Yu, G. Walkup, Z. B. Celik, X. Zhang, and D. Xu, "Atlas: A sequence-based learning approach for attack investigation." in USENIX Security Symposium, 2021, pp. 3005–3022.

[5] A. Riddle, K. Westfall, and A. Bates, "Atlasv2: Atlas attack engagements, version 2," arXiv preprint arXiv:2401.01341, 2023.

[6] T. Bilot, B. Jiang, Z. Li, N. El Madhoun, K. Al Agha, A. Zouaoui, and T. Pasquier, "Sometimes Simpler is Better: A Comprehensive Analysis of State-of-the-Art Provenance-Based Intrusion Detection Systems," in Security Symposium (USENIX Sec'25). USENIX, 2025.

[7] Microsoft, "Analyzing solorigate, the compromise of solarwinds orion platform," Microsoft Security Response Center, Tech. Rep., 2021, available at https://www.microsoft.com/en-us/security/blog/2020/12/18/analyzing-solorigate-the-compromised-dll-file-that-started-a-sophisticated-cyberattack-and-how-microsoft-defender-helps-protect/.

[8] FireEye, "Highly evasive attacker leverages solarwinds supply chain to compromise multiple global victims with sunburst backdoor," FireEye Threat Research, Tech. Rep., 2020, available at https://cloud.google.com/blog/topics/threat-intelligence/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor/.

[9] R. Ikeda and J. Widom, "Data lineage: A survey," Stanford University Publications. http://ilpubs. stanford. edu, vol. 8090, no. 918, p. 1, 2009.

[10] "eBPF," 2023, https://ebpf.io/.

[11] "Event Tracing for Windows," 2023, https://learn.microsoft.com/en-us/windows-hardware/drivers/devtest/event-tracing-for-windows–etw-.

[12] X. Chen, H. Irshad, Y. Chen, A. Gehani, and V. Yegneswaran, "CLARION: Sound and clear provenance tracking for microservice deployments," in 30th USENIX Security Symposium (USENIX Security 21).

[13] P. Datta, I. Polinsky, M. A. Inam, A. Bates, and W. Enck, "Alastor: Reconstructing the provenance of serverless intrusions," in 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 2443–2460.

[14] W. U. Hassan, M. A. Noureddine, P. Datta, and A. Bates, "Omegalog: High-fidelity attack investigation via transparent multi-layer log analysis," in Network and distributed system security symposium, 2020.

[15] E. Altinisik, F. Deniz, and H. T. Sencar, "Provg-searcher: A graph representation learning approach for efficient provenance graph search," in Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, 2023, pp. 2247–2261.

[16] Z. Li, Q. A. Chen, R. Yang, Y. Chen, and W. Ruan, "Threat detection and investigation with system-level provenance graphs: a survey," Computers & Security, vol. 106, p. 102282, 2021.

[17] S. Li, F. Dong, X. Xiao, H. Wang, F. Shao, J. Chen, Y. Guo, X. Chen, and D. Li, "Nodlink: An online system for fine-grained apt attack detection and investigation," arXiv preprint arXiv:2311.02331, 2023.

[18] M. N. Hossain, S. M. Milajerdi, J. Wang, B. Eshete, R. Gjomemo, R. Sekar, S. Stoller, and V. Venkatakrishnan, "$sleuth$: Real-time attack scenario reconstruction from $cots$ audit data," in 26th USENIX Security Symposium (USENIX Security 17), 2017, pp. 487–504.

[19] M. N. Hossain, S. Sheikhi, and R. Sekar, "Combating dependence explosion in forensic analysis using alternative tag propagation semantics," in IEEE Symposium on Security and Privacy (SP), 2020.

[20] S. M. Milajerdi, R. Gjomemo, B. Eshete, R. Sekar, and V. Venkatakrishnan, "Holmes: Real-time apt detection through correlation of suspicious information flows," in IEEE Symposium on Security and Privacy (SP).

[21] L. Wang, X. Shen, W. Li, Z. Li, R. Sekar, H. Liu, and Y. Chen, "Incorporating gradients to rules: Towards lightweight, adaptive provenance-based intrusion detection," arXiv preprint arXiv:2404.14720, 2024.

[22] L. Li and W. Chen, "Congraph: Advanced persistent threat detection method based on provenance graph combined with process context in cyber-physical system environment," Electronics, vol. 13, no. 5, p. 945, 2024.

[23] E. Manzoor, S. M. Milajerdi, and L. Akoglu, "Fast memory-efficient anomaly detection in streaming heterogeneous graphs," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1035–1044.

[24] X. Han, T. F. J. Pasquier, A. Bates, J. Mickens, and M. I. Seltzer, "Unicorn: Runtime provenance-based detector for advanced persistent threats," in 27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020. The Internet Society, 2020.

[25] D. Yang, B. Li, L. Rettig, and P. Cudré-Mauroux, "Histosketch: Fast similarity-preserving sketching of streaming histograms with concept drift," in 2017 IEEE International Conference on Data Mining (ICDM). IEEE, 2017, pp. 545–554.

[26] Y. N. Zian Jia, Yun Xiong, "Magic: Detecting advanced persistent threats via masked graph representation learning," in 33nd USENIX Security Symposium (USENIX Security 23), 2024, pp. 1–18.

[27] M. U. Rehman, H. Ahmadi, and W. U. Hassan, "Flash: A comprehensive approach to intrusion detection via provenance graph representation learning," in 2024 IEEE Symposium on Security and Privacy (SP). IEEE Computer Society, 2024, pp. 139–139.

[28] A. Goyal, G. Wang, and A. Bates, "R-caid: Embedding root cause analysis within provenance-based intrusion detection," in 2024 IEEE Symposium on Security and Privacy (SP). IEEE, 2024, pp. 3515–3532.

[29] C. Zhang, T. Jia, G. Shen, P. Zhu, and Y. Li, "Metalog: Generalizable cross-system anomaly detection from logs with meta-learning," in 2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE), 2024, pp. 1899–1910.

[30] Q. Wang, W. Hassan, D. Li, K. Jee, X. Yu, K. Zou, J. Rhee, Z. Chen, W. Cheng, C. Gunter, and H. Chen, "You are what you do: Hunting stealthy malware via data provenance analysis," 01 2020.

[31] Z. Cheng, Q. Lv, J. Liang, Y. Wang, D. Sun, T. Pasquier, and X. Han, "Kairos: Practical intrusion detection and investigation using whole-system provenance," in 2024 IEEE Symposium on Security and Privacy (SP). IEEE, 2024, pp. 3533–3551.

[32] B. Bhattarai and H. H. Huang, "Prov2vec: Learning provenance graph representation for anomaly detection in computer systems," in Proceedings of the 19th International Conference on Availability, Reliability and Security, 2024, pp. 1–14.

[33] L. Wu, Y.-L. Xie, S.-X. Zhao, P. Zhou, D. Feng, A. Wildani, and Y.-Y. Wu, "Efficient intrusion detection via heterogeneous graph attention networks and parallel provenance analysis," Computer Networks, p. 111552, 2025.

[34] K.-K. Muniswamy-Reddy, D. A. Holland, U. Braun, and M. I. Seltzer, "Provenance-aware storage systems." in Usenix annual technical conference, general track, 2006, pp. 43–56.

[35] D. J. Pohly, S. McLaughlin, P. McDaniel, and K. Butler, "Hi-fi: collecting high-fidelity whole-system provenance," in Proceedings of the 28th Annual Computer Security Applications Conference, 2012, pp. 259–268.

[36] A. Gehani and D. Tariq, "Spade: Support for provenance auditing in distributed environments," in ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing. Springer, 2012, pp. 101–120.

[37] J. Liu, M. A. Inam, A. Goyal, A. Riddle, K. Westfall, and A. Bates, "What we talk about when we talk about logs: Understanding the effects of dataset quality on endpoint threat detection research," in 2025 IEEE Symposium on Security and Privacy (SP). IEEE, 2025, pp. 112–129.

[38] Y. Huang, W. U. Hassan, Y. Guo, X. Chen, and D. Li, "Provsyn: Synthesizing provenance graphs for data augmentation in intrusion detection systems," arXiv preprint arXiv:2506.06226, 2025.

[39] A. Goyal, X. Han, G. Wang, and A. Bates, "Sometimes, you aren't what you do: Mimicry attacks against provenance graph host intrusion detection systems," in 30th Network and Distributed System Security Symposium, 2023.

[40] K. Mukherjee, J. Wiedemeier, T. Wang, J. Wei, F. Chen, M. Kim, M. Kantarcioglu, and K. Jee, "Evading provenance-based ml detectors with adversarial system actions," in 32nd USENIX Security Symposium (USENIX Security 23), 2023, pp. 1199–1216.

[41] Z. Li, Y. Wei, X. Shen, L. Wang, Y. Chen, H. Xu, S. Ji, F. Zhang, L. Hou, W. Liu et al., "Marlin: Knowledge-driven analysis of provenance graphs for efficient and robust detection of cyber attacks," arXiv preprint arXiv:2403.12541, 2024.

[42] Red Canary, "Atomic red team test repository," https://github.com/redcanaryco/atomic-red-team, accessed: 2025-12-10.

[43] L. Wang, Z. Li, Y. Jiang, Z. Wang, Z. Guo, J. Wang, Y. Wei, X. Shen, W. Ruan, and Y. Chen, "From sands to mansions: Towards automated cyberattack emulation with classical planning and large language models," arXiv preprint arXiv:2407.16928, 2024. [Online]. Available: https://arxiv.org/abs/2407.16928

[44] X. Wang, B. Wang, D. Lu, J. Yang, T. Xie, J. Wang, J. Deng, X. Guo, Y. Xu, C. H. Wu, Z. Shen, Z. Li, R. Li, X. Li, J. Chen, B. Zheng, P. Li, F. Lei, R. Cao, Y. Fu, D. Shin, M. Shin, J. Hu, Y. Wang, J. Chen, Y. Ye, D. Zhang, D. Du, H. Hu, H. Chen, Z. Zhou, H. Yao, Z. Chen, Q. Gu, Y. Wang, H. Wang, D. Yang, V. Zhong, F. Sung, Y. Charles, Z. Yang, and T. Yu, "Opencua: Open foundations for computer-use agents," 2025. [Online]. Available: https://arxiv.org/abs/2508.09123

[45] B. Jiang, T. Bilot, N. El Madhoun, K. Al Agha, A. Zouaoui, S. Iqbal, X. Han, and T. Pasquier, "Orthrus: Achieving high quality of attribution in provenance-based intrusion detection systems," in Security Symposium (USENIX Sec'25). USENIX, 2025.

[46] S. Wang, Z. Wang, T. Zhou, H. Sun, X. Yin, D. Han, H. Zhang, X. Shi, and J. Yang, "Threatrace: Detecting and tracing host-based threats in node level through provenance graph learning," IEEE Transactions on Information Forensics and Security, vol. 17, pp. 3972–3987, 2022.