Semantic-Aware Provenance-Based Intrusion Detection for Edge Systems

Qingyu Zeng(✉), Songxuan Liu, Yu Wu, and Yuko Hara

Institute of Science Tokyo, Tokyo 152-8550, Japan
qyzeng@cad.ict.e.titech.ac.jp, liu.s.au@m.titech.ac.jp,
wu.y.e989@m.isct.ac.jp, hara@cad.ict.e.titech.ac.jp

Abstract. As cyberattacks targeting edge devices become increasingly sophisticated, detecting intrusion behaviors becomes increasingly difficult, demanding effective intrusion detection systems (IDS) tailored for resource-constrained environments. Existing provenance-based IDSs show a promising ability to detect benign and attack behaviors, but require a lot of computing resources to build provenance graphs, which is not conducive to edge deployment. To address this gap, we propose a semantic-aware provenance-based IDS, which constructs provenance graphs and prioritizes security-critical events using semantic roles. Through structural and semantic analysis on the DARPA Engagement 3 (CADETS) dataset, we show that role-annotated subgraphs exhibit measurable divergence between benign and attack activities, with certain roles (e.g., `binary-execution`) appearing over 3× more frequently in attack traces, demonstrating the feasibility of semantic priors for lightweight intrusion detection.

Keywords: Intrusion detection · Provenance graph · Semantic aware.

1 Introduction

The proliferation of edge computing devices, ranging from smart gateways to industrial controllers, has significantly expanded the attack surface of modern cyber-physical systems. These devices, often deployed in uncontrolled environments and lacking comprehensive security infrastructure, are increasingly targeted by adversaries for persistent surveillance, lateral movement, and sensitive data exfiltration [5]. The constrained nature of edge platforms, however, poses fundamental challenges for deploying effective intrusion detection systems (IDS) that are both precise and lightweight [8].

Recent advances in provenance-based security monitoring have demonstrated the effectiveness of system-level event graphs in capturing causal dependencies between entities such as processes, files, and network flows [9]. Provenance graphs offer a holistic and high-fidelity view of system behavior, enabling accurate detection and forensic traceability of complex attack chains [7]. However, most existing

provenance-based IDS frameworks assume access to abundant computational resources and rely on heavyweight graph analytics, rendering them unsuitable for edge deployment [3].

In this work, we explore the feasibility of integrating semantic-awareness into provenance graphs as part of an ongoing effort. We present a preliminary modular framework that constructs, filters, and analyzes provenance subgraphs. Our key idea is to associate system events with high-level *semantic roles* (e.g., `binary-execution`) and leverage these roles during training to differentially weight behavior based on its security relevance. This enables a tunable trade-off between accuracy and efficiency, while enhancing the interpretability of alerts. To evaluate our design, we conduct a preliminary analysis using the CADETS dataset, which includes detailed system call-level traces and ground truth. Our results indicate that role-annotated subgraphs exhibit measurable divergence between benign and attack activities. To the best of our knowledge, this is the first work that introduces semantic role annotation into provenance-based intrusion detection system in edge computing scenarios.

The remainder of this paper is organized as follows. Section 2 briefly reviews prior work on provenance-based intrusion detection systems. Section 3 presents the overview of our proposed system architecture. Section 4 presents the semantic-aware provenance graph modeling. Section 5 presents preliminary results. Section 6 concludes our work and discusses future research directions.

2 Related Work

Provenance-based IDSes leverage data provenance, which records the origin and transformation processes of data within a system, to detect malicious activities more effectively than traditional host-based IDSes. Unlike traditional intrusion detection methods that primarily analyze system events in isolation, Provenance-based IDS provide richer semantic context by constructing provenance graphs—directed acyclic graphs recording causal relationships among system entities and events. These systems typically include modules for data collection, graph summarization, intrusion detection, and benchmarking datasets.

Figure 1 illustrates a benign scenario: a common debugging workflow using `LD_PRELOAD`. A user connects to the system via `SSH` and writes a custom shared library, `my_debug_hooks.so`, containing specific debugging code. Subsequently, the `LD_PRELOAD` environment variable is configured with the path of the custom library. Then, the target application (`test`) is launched. Because of the `LD_PRELOAD` setting, the operating system loads the custom library into the application's memory first. Finally, the custom library's code intercepts function calls from the `test` process and writes diagnostic data to a log file (`/tmp/app_debug.log`). This is a routine, non-intrusive debugging practice.

Provenance-based methods can broadly be categorized into anomaly-based, rule-based, and tag-propagation-based approaches. Anomaly-based systems [1] identify intrusions by detecting deviations from established normal behaviors, rule-based methods [6] utilize predefined rules and signatures, and tag-propagation

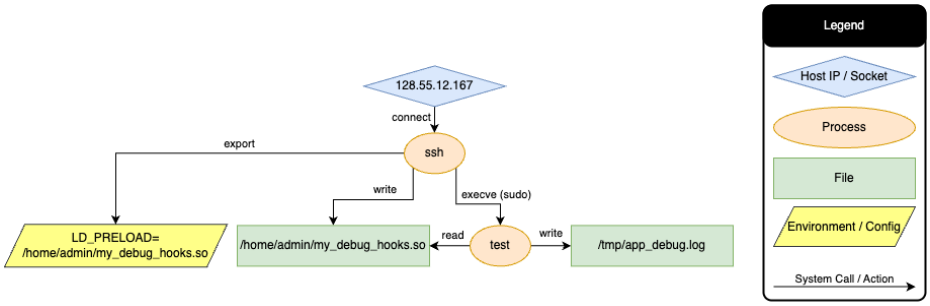

Fig. 1. Illustrative provenance graphs of a benign administrative activity.

methods assign and propagate trustworthiness or risk labels through the provenance graph to highlight suspicious activities.

ALASTOR [1] is a provenance-based IDS framework leveraging graph embedding techniques under the anomaly-based category, tailored specifically for serverless computing environments. Traditional provenance approaches face challenges in these environments due to ephemeral function lifetimes and container reuse, which attackers can exploit to conceal malicious activities from detection. To overcome these limitations, ALASTOR captures detailed provenance data at both the system and application layers, tracing interactions such as system calls and network requests within each serverless function instance. These fine-grained events are aggregated into global provenance graphs, enabling precise reconstruction and analysis of intrusion scenarios. ALASTOR was evaluated through implementation on the OpenFaaS platform, testing complex intrusion scenarios targeting serverless applications. The results indicated that ALASTOR reconstructed detailed attack paths that traditional monitoring methods and several commercially available tools failed to detect.

CAPTAIN [6] is another adaptive provenance intrusion detection framework. Different from ALASTOR, CAPTAIN employs a rule-based approach. Specifically designed to overcome the limitations of traditional static-rule provenance systems, it utilizes a differentiable rule-configuration method based on gradient descent algorithms to dynamically adjust system parameters, including tag initialization, tag propagation rates, and alarm generation thresholds. In contrast to conventional rule-based provenance intrusion detection systems that rely on manually configured static rules, CAPTAIN automatically adjusts these parameters using benign training data, effectively reducing false alarms while preserving detection accuracy against malicious activities. Experimental evaluations conducted on datasets from DARPA Engagements and datasets generated from simulated environments demonstrated substantial performance improvements, including over 90% reduction in false alarms compared with traditional rule-based methods.

Both ALASTOR and CAPTAIN represent significant advancements in the provenance-based intrusion detection. However, these systems are facing limita-

tions in lightweight security for edge devices. Thus, there is an urgent need of lightweight provenance-based intrusion detection system optimized specifically for the resource-constrained edge environments. This could include reducing the overhead associated with provenance data collection and analysis, simplifying provenance graphs, and optimizing graph summarization without compromising critical security insights.

3 System Design

In this work, we introduce the semantic-aware method into provenance-based intrusion detection, specifically optimizing the graph construction phase through semantic-aware filtering.

Goal. This work aims to enhance provenance-based intrusion detection by introducing semantic-awareness into graph construction. We assign high-level semantic roles to system events, focusing on security-critical behaviors.

Assumptions. Provenance traces contain a large volume of semantically irrelevant events, and that meaningful roles can be assigned using rules.

3.1 Architecture Overview

Figure 2 illustrates the architecture of the proposed system. The system operates on a stream of low-level system audit events and produces alerts corresponding to anomaly signatures. It consists of the following main components:

1. **Event Parser:**
   The event parser processes raw system logs from the widely acknowledged public datasets provided by DARPA Engagement, where each line contains a single event record with detailed information about system activities such as file operations, process executions, network communications, and memory operations. The parser extracts key components from these logs including subjects (processes), objects (files, network flows, memory), and events (read, write, execve, clone, etc.), then standardizes them into a unified format that can be used to construct provenance graphs.
2. **Graph Constructor:**
   The graph constructor uses data processed by the event parser to generate a provenance graph. The graph nodes represent the system entities: Subject nodes capture process information including PID, PPID, command line, process name, and parent relationships, while Object nodes represent files, network flows, memory regions, and other system resources with their respective attributes like file paths, IP addresses, and memory addresses. The edges between nodes represent the events and operations that occurred between subjects and objects, each edge containing metadata such as timestamp, event type, and operation parameters. The system maintains a node buffer to track entity states and handles node updates when entity properties change over time. The final graph is serialized into JSON format with separate files

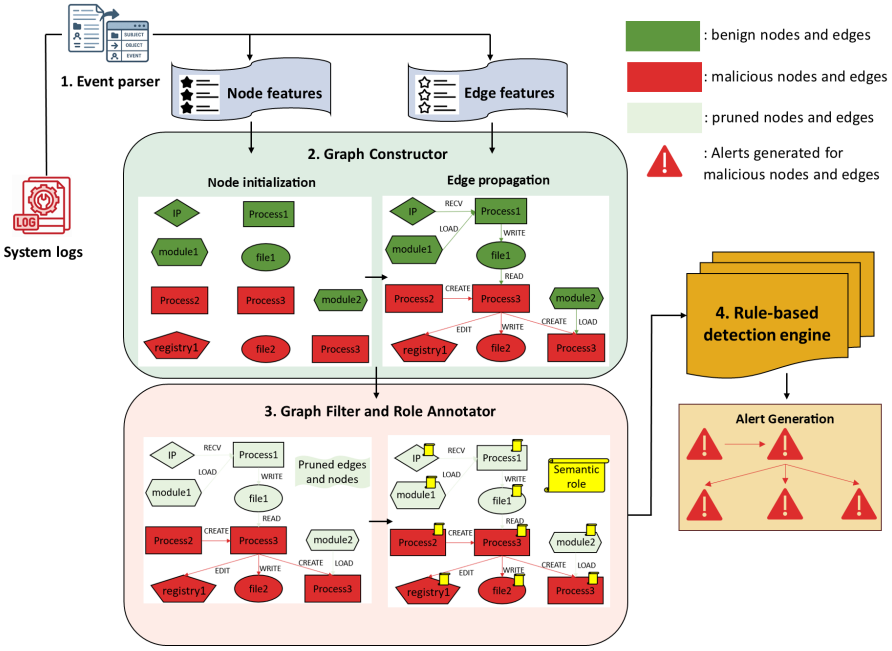

Fig. 2. Architecture overview of the proposed IDS.

for nodes, edges, and principals, creating a comprehensive provenance graph that captures the complete system execution history and can be used for intrusion detection through graph-based analysis and rule-based analysis.

3. **Graph Filter and Role Annotator:**
   The original provenance graph generated by the graph generator contains a large number of edges and nodes that represent low-risk operations. These routine operations in the provenance graph take up a lot of memory, making them unsuitable for edge environments to process. The idea is to apply graph filters and eliminate redundant events, focusing attention on security-critical behaviors. The graph filter prunes the edges representing these low-risk operations, which not only reduces memory usage but also improves the clarity of the risk operations in the provenance graph.
   A role annotator is applied to assign high-level semantic roles that capture the potential impact of each event on system security. We define a set of deterministic mappings based on file paths, system call types, and access patterns. For example, any execution of a binary via `execve`, such as launching tools in `/usr/bin`, is categorized as `binary-execution`, while read operations from low-sensitivity sources like `/dev/tty` are labeled as `routine-read`.
4. **Detection Engine:**
   The detection engine combines rule-based analysis with the Adam optimizer. The detection engine uses a tag-based propagation system. These tags are

initialized based on predefined rules (e.g., untrusted files have lower label vector values) and then propagated through the graph as events occur. The detection engine works by continuously processing system events and updating the value of the tags of each node. After an event happens, the detection engine compares the tag values with learned thresholds to identify attacks. The system can detect various attack types including FileExec (executing untrusted files), FileCorruption (modifying high-integrity files), DataLeak (sending confidential data to untrusted networks), PrivilegeEscalation (gaining root privileges), and MaliciousFileCreation (creating files by untrusted processes). The detection engine uses Adam optimizer to optimize the propagation strength and detection threshold parameters based on benign training data. The training process minimizes false positives by adjusting these parameters to ensure that normal system behavior doesn't trigger alarms while maintaining sensitivity to actual attacks. After comparing the tag values with the detection rules, the alarms would be generated and include information about the suspicious events, entities, and the specific attack type.

3.2 Threat Model

We consider a threat model where edge devices, including smart gateways and industrial controllers, operate in environments that are network-untrusted. The adversary can be local, with remote shell access to the device, and adjacent on the network, capable of interacting with exposed services and injecting traffic. In either case, the attacker can execute code, exploit software vulnerabilities, and misuse legitimate tools.

The attacker aims to execute malicious binaries and perform write operations. We assume that the attacker has no prior control over the monitoring agent and cannot tamper with the system audit logs. The logging mechanism is trusted.

The malicious activity, depicted in the Figure 3, is based on an attack scenario from the DARPA Engagement 5 dataset. The attack sequence demonstrates a classic post-exploitation pattern. It begins with the attacker connecting to the system via `ssh` using stolen credentials. The attack then uses `scp` to upload a malicious toolset, including a client named `mchk` for privilege escalation. After establishing a foothold, the `mchk` client is executed, which employs a local privilege-escalation technique, and leverages `memfd_create` to map an anonymous, in-memory payload before invoking it, while avoiding on-disk detection. Operating with full administrative control, the attacker then proceeds to exfiltrate critical system files, including `/etc/passwd` and `/etc/shadow`, as well as sensitive user documents like `/home/admin/biol`, back to the remote socket.

4 Semantic-Aware Provenance Graph Modeling

System behavior is modeled using a directed provenance graph $G = (V, E, T)$, where V includes heterogeneous entities (e.g., process, files, and network sockets); E encodes interactions between entities (e.g., *read-from*, *write-to*, and *ex-*

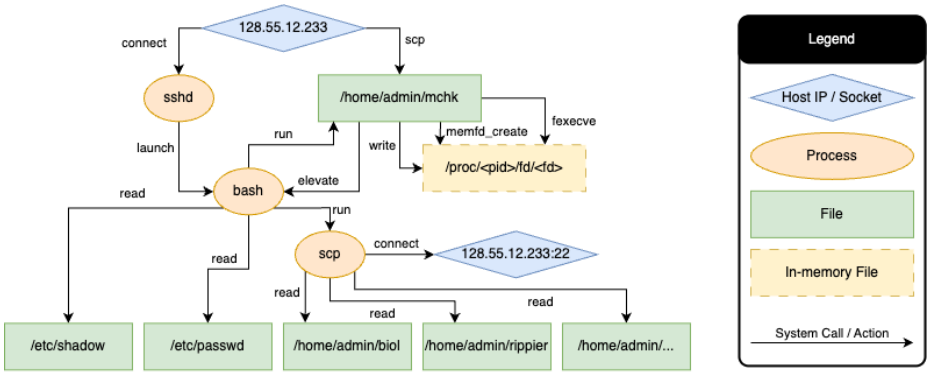

Fig. 3. Illustrative provenance graphs of a multi-stage malicious attack.

ecutes); and T preserves temporal ordering of events. The graph structure captures causal relationships between system activities and forms the basis of behavioral analysis [4].

4.1 Provenance Graph Semantics and Role Annotation

While many provenance systems treat audit events as atomic interactions, this work augments each event with a high-level *semantic role* that captures its functional intent and potential security relevance. These roles are derived using lightweight, rule-based heuristics that analyze object paths, file types, and system call contexts.

Given the volume and redundancy of system-level provenance data, this work focuses on high-value behavioral features.

- **Edge Filtering:** Routine interactions, such as reading configuration files and writing logs, are prevalent in both benign and malicious sessions. We filter edges that correspond to known low-risk actions (e.g., `read-from` on `/etc/motd`) to reduce graph clutter without losing detection sensitivity.
- **Role Annotation:** We apply deterministic labeling rules to assign each event a semantic role, as shown in Table 1. These annotations serve dual purposes: (i) they filter out low-risk behavior during graph construction, and (ii) they modulate loss weights during model training. For instance, misclassifying a `binary-execution` or `staging-write` event may incur a higher penalty than a benign `routine-read`.

4.2 Role-Weighted Training Objective

To reflect the varying semantic importance of different events, we introduce a *role-weighted loss* scheme that scales each event's loss contribution based on its

Table 1. Semantic Role Definitions

Role	Definition
`routine-read`	Reads from non-sensitive locations such as `/dev/tty`, temporary user files, or other benign sources. Common in routine system interactions.
`privileged-read`	Read access to sensitive credentials files, including `/etc/shadow`, SSH private keys, and password files.
`staging-write`	Writes directed at temporary or staging directories, e.g., `/tmp`, `/var/tmp`. Often used for staging payloads and served as attacker preparation zones.
`binary-execution`	Execution of a binary file, as captured by `execve` system calls and equivalent process launch events.
`neutral`	Events that do not match any defined semantic rules. These are often low-risk actions not indicative of security-critical behavior. Used as the default role when no semantic annotation applies.

semantic role. Formally, for each event e with associated role $r \in \mathcal{R}$, the final loss is computed as:

$$\mathcal{L}_{\text{total}} = \sum_{e \in \mathcal{E}} w_r \cdot \mathcal{L}(e), \tag{1}$$

where $\mathcal{L}(e)$ is the base loss (e.g., reconstruction error), and w_r is the role weight assigned to semantic role r.

5 Preliminary Evaluation

5.1 Experimental Setup

All offline experiments are conducted on a machine equipped with an Intel Xeon(R) Silver 4410Y and 256GB RAM. We conduct evaluations on the DARPA Engagement 3 dataset, specifically focusing on the CADETS scenario. This dataset is part of DARPA's Transparent Computing program and was collected from a real-world enterprise environment under controlled attack conditions. It offers system-level event traces in the common data model format, including both benign and malicious activities.

The CADETS trace contains detailed logs of system calls. Each event is timestamped and linked [6], making it suitable for constructing temporal provenance graphs. The dataset includes several realistic attack campaigns such as reverse shell injection, credential scraping, and privilege escalation, embedded within multi-day traces.

5.2 Semantic Subgraph Analysis

To evaluate the feasibility of our approach, we analyzed provenance subgraphs derived from benign and malicious activity windows. We compute a set of structural and semantic metrics on each subgraph to investigate whether malicious

Table 2. Structural comparison of benign vs. attack subgraphs. (mean ± std.)

Metric	Benign	Attack
% of `routine-read` nodes	6.59 ± 16.91%	0.70 ± 5.89%
% of `privileged-read` nodes	0.01 ± 0.57%	0.00 ± 16.10%
% of `binary-execution` nodes	1.49 ± 8.49%	4.93 ± 14.91%
% of `staging-write` nodes	0.03 ± 1.26%	0.70 ± 5.89%

Table 3. Comparison of training efficiency between w/o and w/ role-weighted loss.

Metric	w/o Role-Weighted	w/ Role-Weighted
Epochs to Reach Loss < 10	33	**26**
Epochs to Reach Loss < 1	79	**45**
Min Final Loss (after 100 epochs)	0.00	**0.00**
Avg. Loss (Epochs 20–40)	23.23	**6.71**

behaviors exhibit statistically distinct topological patterns. This allows us to answer the core question: *does the role-annotated subgraph itself encode enough information to separate benign from attack behavior?*

Table 2 compares the proportion of semantically annotated nodes in benign vs. attack subgraphs. We observe that `binary-execution` and `staging-write` roles are more prevalent in attack graphs, reflecting typical behaviors like launching malicious payloads and staging operations in temporary storage. In contrast, `routine-read` appears more in benign traces, consistent with normal startup and interactive operations. This distinction validates the design intuition behind our semantic annotation: although individual events may not be malicious in isolation, their elevated presence in attack subgraphs provides valuable priors for learning. Consequently, assigning higher weights to such roles during training can help amplify model sensitivity to stealthy but structurally deviant behaviors.

5.3 Training Efficiency Comparison

To evaluate the efficiency of our semantic-aware training strategy, we compare the convergence behavior of models trained with and without role-weighted loss. We introduce a semantic-aware loss weighting scheme, where each role is assigned a fixed importance weight reflecting its potential security impact. For example, `binary-execution` and `staging-write` are considered more critical than routine operations.

Our role-weighted loss improves convergence speed by emphasizing security-critical semantic roles and down-weighting routine events during training. Table 3 compares the training efficiency with and without role-weighted loss. The role-weighted model achieves faster convergence, reaching critical loss thresholds significantly earlier than without role-weighted.

Summary. Preliminary results show that semantic role annotations help distinguish attack behaviors through structural patterns and improve model training

efficiency when used in loss weighting. These findings highlight the potential of semantic-aware provenance analysis and provide a promising foundation for future development.

6 Discussion and Future Work

In this work, we introduce a semantic-aware, provenance-based IDS optimized for edge computing environments. By incorporating semantic role annotation, this work aims to balance detection fidelity, interpretability, and resource efficiency. Our current prototype focuses on offline analysis using the CADETS dataset. The results demonstrate that role-annotated subgraphs exhibit measurable structural divergence between benign and attack behavior, validating the effectiveness of our semantic modeling approach.

Several directions remain open for future exploration: 1) Deploying the proposed IDS on actual edge platforms will allow us to evaluate system behavior under real workload conditions. This includes assessing runtime performance, memory usage, and detection latency. 2) Rule-based IDS are lightweight and interpretable, making them well-suited for constrained edge environments. However, their reliance on predefined rules limits their ability to detect zero-day attacks. As part of future work, we plan to integrate lightweight graph neural network (GNN)-based detection models that can identify anomalous provenance graphs. To maintain resource efficiency, we will explore GNN architectures such as GraphSAGE [2] or spatio-temporal GNNs with pruning.

7 Acknowledgment

This work is supported by JST SPRING, Grant Number JPMJSP2180 and JSPS KAKENHI Grant Numbers JP23H00464, JP24KK0184, and JP25K03091.

References

1. Datta, P., et al.: ALASTOR: Reconstructing the provenance of serverless intrusions. In: 31st USENIX Security Symposium. pp. 2443–2460 (2022)
2. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems. vol. 30 (2017)
3. Han, X., Pasquier, T., Seltzer, M.: Provenance-based intrusion detection: opportunities and challenges. In: 10th USENIX Workshop on the Theory and Practice of Provenance (2018)
4. Mei, J., Moura, J.M.: Signal processing on graphs: Causal modeling of unstructured data. IEEE Transactions on Signal Processing **65**(8), 2077–2092 (2016)
5. Siwakoti, Y.R., et al.: Advances in iot security: Vulnerabilities, enabled criminal services, attacks, and countermeasures. IEEE Internet of Things Journal **10**(13), 11224–11239 (2023)
6. Wang, L., et al.: Incorporating gradients to rules: Towards lightweight, adaptive provenance-based intrusion detection. arXiv preprint arXiv:2404.14720 (2024)

7. Xie, Y., et al.: Pagoda: A hybrid approach to enable efficient real-time provenance based intrusion detection in big data environments. IEEE Transactions on Dependable and Secure Computing **17**(6), 1283–1296 (2018)
8. Zarpelão, B.B., et al.: A survey of intrusion detection in internet of things. Journal of Network and Computer Applications **84**, 25–37 (2017)
9. Zipperle, M., et al.: Provenance-based intrusion detection systems: A survey **55**(7), 1–36 (2022)