SProj Report: Provenance-Based Anomaly Detection

Talha Abrar
24100107

Ahmad Shamail
24100016

Mohammad Jaffer Iqbal
24100064

I. ShadeWatcher Recreation

A. Background

System auditing is a foundational tool in cybersecurity, offering a low-level view into system entity interactions which can indicate potential threats. Such auditing tracks how entities like processes, files, and network sockets interact, providing logs (that can be used to build provenance graphs) that can be analyzed for security breaches by tracking the flow of information. However, high quality logs result in an overwhelming volume of data, which is challenging to manage and analyze effectively, especially when trying to efficiently learn a good, generalized representation of what is 'benign' from the data.

B. Introduction

Shadewatcher [2] is an advanced cyber threat analysis tool that leverages the structural similarities between threat detection in cybersecurity and recommendation tasks used in information retrieval systems. By conceptualizing security interactions within a system to user-item interactions in recommendation systems, Shadewatcher predicts the likelihood of system entities engaging in interactions that are out of the ordinary - thus potentially indicating a cyber threat. At its core, Shadewatcher uses a graph neural network to process high-order connectivity data from audit records, which allows it to recognize complex patterns of interactions that may indicate sophisticated attacks. Keeping in line with similar work in this space, Shadewatcher performs unsupervised learning to build a representation of what benign system entity interactions are, such that the dot product of the entity embedding would be higher if the entities are likely to interact. While this approach sounds revolutionary, Shadewatcher does show some 'too good to be true' results which grossly hint at a cherry-picking approach. Since this system was not fully open-sourced, our task was to complete the implementation and check the reproducibility of their results on the open-source DARPA Engagement dataset.

C. Challenges and Approach

With the core model being published online, our first task was to use the provenance graph to perform interaction extraction in a way that was faithful to the approach in the paper. Since these 'interaction pairs' were to be fed into the embedding and GNN models for training, it was important to match their figures exactly to the ones provided in the paper as they directly controlled the feasibility and quality of the training. These interaction pairs needed to be found via forward traversals rooted at data objects, in a way that did not lead to a dataset explosion. Thus, noise reduction and summarization techniques needed to be leveraged. For the DARPA E3 engagement, the numbers quoted in the paper are shown in Table 1:

Dataset	#Node	#Edge	#Behavior	#Interaction
TRACE Dataset	6,109,307	12,661,091	148,335	7,923,332

Table 1: *DARPRA TRACE E3 Dataset Statistics [as reported by ShadeWatcher]*

The first challenge we faced was the extremely slow running time of the existing parser. We used bloom filters to optimize the pipeline, such that it became much more feasible to run. Another challenge to the existing pipeline was that it required the complete provenance graph to be ingested and loaded into memory, which was around 600gb. We fixed the parser to break the provenance graph and ingest it while simultaneously saving the ingested and processed data. The remaining challenge was to match the figures for the number of behaviors and interactions.

Behaviors are traversals rooted at data objects. Behaviors also needed to be merged to manage redundancy. These were then merged, after experimentation, using the following techniques:

(i) Merging traversals rooted at A with B, if the traversal rooted at B has only one incoming edge on a path from A

(ii) Merging traversals rooted at A and B if a traversal rooted at B is already temporally explored during a traversal of A

Using these merging techniques, we were able to bring the number of behaviors down to 400,000.

Despite trying to merge behaviors, the extracted interactions were still in orders of 100 million, which was infeasible to be trained on the GPU. To this end, we tried a few techniques. The first was noise reduction in an effort to reduce irrelevant audit records. This was inspired by [1] and [3], and we implemented (i) Causality Preserving Reduction (CPR), which aims to remove repetitive shadow edges, (ii) Process-centric Causality Approximation Reduction (PCAR), in which we aggregated edges in process nodes. Despite matching the number of edges without PCAR, we still had interaction pairs in orders of 100 million. This was due to the dependency explosion problem, in which long running processes with interleaving edges lead to data dependency with conceptually unrelated data objects. PCAR was particularly helpful as it tackled the issue of an

intense burst of read/write events which are not handled by basic CPR. Through PCAR, we managed to decrease the interaction pairs to 10s of millions. We also experimented with Source Dependence (SD) reduction, but this was not helpful in our case.

The second strategy we used was a continuation of ad-hoc pruning mentioned in the Shadewatcher paper. The paper acknowledges that a forward dfs will definitely lead to a dependency explosion, and thus, the dfs rooted at data objects needs to be terminated preemptively once a 'noisy node' is encountered. In the Watson paper [4] and Shadewatcher codebase, these are limited to a few like 'firefox', '/proc', '/dev', etc., with a vague mention of other noisy nodes without any concrete list in [4]. Using the entities quoted by the authors (without PCAR), we observed that we still had interaction pairs in the order of 100 million, which showed that the authors had employed significant ad-hoc pruning apart from the entities concretely mentioned in the paper. To find out more such entities, we ranked them according to their number of outgoing edges (and thus their potential to suffer from the explosion) and experimented with removing them iteratively to match the number of interactions. To this end, we also employed regex to ensure that if a parent file obj is being removed, then subsequent nested children file obj are also removed. Simultaneously, we also ensured that any entity which is in a malicious path is not removed. Using these, we found that significant entities like config files, admin cache and libraries needed to be pruned from the traversals in order to come close to the quoted number of interactions. Next, we trained the model on extracted entities in the order of 8M, and found the accuracy upon testing to be nowhere near the quoted figures in the paper. The testing set was formulated from the traversal directions provided in the paper for the test set (i.e. specific entities on which the attack path was rooted), and matched with the number of edges quoted in the paper for each attack path in the DARPA E3 dataset. This was also corroborated with a visualization through SPADE.

The final strategy we employed was similar in nature to PCAR and was inspired by [1], in which we aimed to preprocess the provenance graph into a more coarse-grained representation. For example, where previously a single node was uniquely recorded for each firefox process and treated as a separate entity, we tried merging all similar nodes (in this case all firefox nodes) into one generalized firefox node, and then merge the redundant edges. This would decrease the entities and edges, but in turn would also decrease the interaction pairs in the training set. Upon experimental evaluation, we found the model accuracy to be nowhere near the quoted figures on the paper.

D. ProvNinja's Reproduction

ProvNinja's implementation strategy, focusing exclusively on single-hop interactions within the knowledge graph, underscores a significant limitation in capturing the complex interdependencies typical of cybersecurity threats. This approach, as detailed, primarily utilizes direct entity relationships for training its graph neural network (GNN) and TransR embedding model. Such a constrained methodology neglects the broader, indirect interactions that are critical for identifying sophisticated anomalies, potentially leading to a higher rate of false negatives in complex or varied datasets. This issue mirrors broader challenges within the field, as noted in our upcoming project for the ACM REP, which aims to address inefficiencies and gaps in algorithmic choices that can be substantially improved to enhance tool reproducibility and effectiveness.

Moreover, by not incorporating multi-hop interactions, ProvNinja's model fails to effectively track provenance—defined as the comprehensive analysis of data flows through multiple system processes. This oversight not only limits the model's ability to understand the full scope of system interactions but also reduces its effectiveness in real-world applications where threats often manifest across various interconnected nodes. To enhance detection capabilities, integrating multi-hop interactions and adhering more closely to provenance principles would allow for a more nuanced understanding of anomalies and improve the model's overall reliability and predictive accuracy in practical settings. This subtle integration of broader data handling issues, such as those that can arise from unfaithful tool implementations and lead to significant losses in contextual information, aligns with our broader research goals and highlights the critical nature of these methodological enhancements.

II. AirTag Evaluation

A. Background

The current literature on intrusion-based anomaly detection systems employs provenance widely in its models. ShadeWatcher [2] uses concepts from item recommendation systems found online to construct a graph of interacting system entities and learns models on it. Unicorn [5] captures provenance graph snapshots as it's fed to the model in a streaming fashion and learns the difference between the evolution of benign graphs and malicious graphs. Kairos [6] uses the concept of an evolving provenance graph to train a graph neural network that assigns a numerical anomaly score to each system event, i.e., an interaction between a process, file, or network socket.

B. Provenance vs. Raw Audit Logs: A Comparative Analysis

A common assumption in the literature is that structured system audit logs, or provenance, are superior for anomaly detection when compared to simple textual logs. The reasoning is straightforward: a structured format, such as a graph, more effectively captures entity interactions and can be readily utilized by graph neural networks, which perform optimally with structured data.

Despite this prevalent belief, a review of recent papers reveals a lack of empirical evidence demonstrating the definitive superiority of provenance over raw audit logs. AirTag [7], on the other hand, presents a contrary viewpoint, suggesting that raw audit logs may, in fact, be more effective.

AirTag utilizes data from ATLAS [8], a known provenance system, to argue that anomaly detection can achieve higher accuracy with raw data—data from which ATLAS typically constructs provenance graphs. Their method involves training a deep learning model using unsupervised learning techniques on the log texts.

C. Critique of Methodology and Further Experimentation

However, AirTag's methodology is not without its flaws. The primary issue is its comparison against ATLAS, which employs a batch processing approach rather than a streaming approach. This method fails to capture the dynamic evolution of provenance graphs accurately. Moreover, certain instances of data manipulation in the AirTag model raise concerns, particularly as the test data included signals that could distinctively identify malicious logs from benign ones.

Given that the AirTag model is open-source, we decided to reproduce the model. Our objective was to demonstrate that integrating provenance signals into the raw logs before feeding them into the AirTag system enhances the model's ability to detect malicious activities, thereby confirming the effectiveness of provenance.

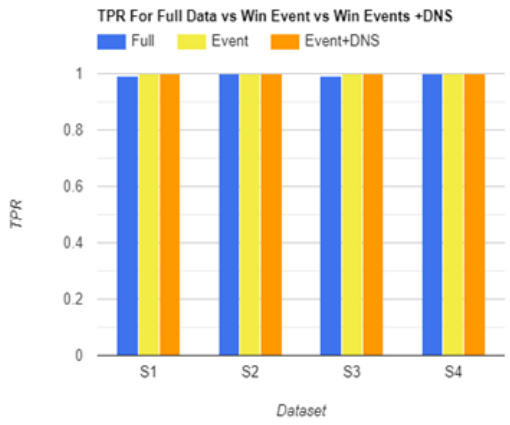
Our approach involved manipulating the parser of the AirTag model to integrate additional fields into the raw log. This allowed AirTag to process this manipulated dataset effectively. These modifications included detailing connections between nodes if an edge existed. This enhanced dataset was derived by feeding the raw text to SPADE [9], which outputs a provenance graph. We then utilized this graph to identify and include edges in the dataset.

Preliminary results show that these provenance signals indeed aid the model, allowing it to achieve higher accuracy in detecting malicious logs. We are currently fine-tuning the data manipulations to optimize accuracy and further substantiate our hypothesis that provenance provides a superior framework for anomaly detection

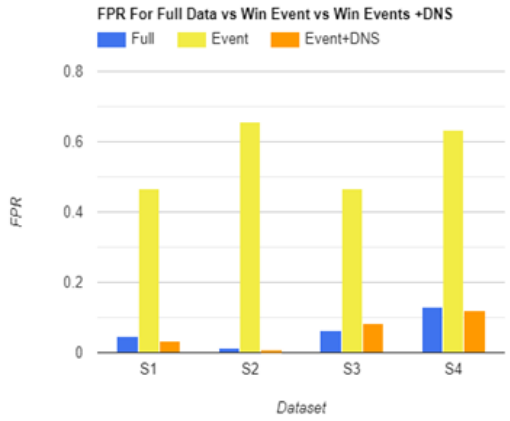
D. Exploring Further with SIGL

Our next experiment involves using the same dataset in two forms: raw and as a provenance graph. The latter will be fed into existing provenance systems like SIGL [10], which are designed to work with graph data. Our hypothesis is that SIGL will demonstrate superior accuracy in anomaly detection compared to the AirTag model using raw data. We began by establishing a baseline accuracy of AirTag using only system event logs and DNS logs from ATLAS data, deliberately excluding browser logs due to their less secure nature. A comparison of metrics between AirTag running on Full ATLAS data, AirTag running on ATLAS event logs only, and AirTag running on ATLAS event logs and DNS logs can be seen in Graph 1 and Graph 2.

We then introduced software installation data that contained DNS logs captured via Wireshark to capture logs in a format consistent with our initial baseline but more suitable for conversion to graph form. The next phase of our research will



Graph 1: *AirTag TPR on Full vs Partial Data*



Graph 2: *AirTag FPR on Full vs Partial Data*

compare the performance of this data when run on a provenance system versus the AirTag model, aiming to demonstrate that provenance systems can indeed leverage structured data for more effective anomaly detection. This part of our study is still in progress and will provide further insights into the potential benefits of provenance-based systems over traditional raw log analysis.

This detailed examination underscores the potential of provenance-based systems in enhancing the accuracy of anomaly detection models, positioning them as a preferable alternative to systems relying on raw audit logs. As this field evolves, further research will undoubtedly refine these approaches, offering more robust and accurate anomaly detection capabilities in cyber security.

III. SIGL - ProvNinja Issue

A. Overview of Misinterpretation

ProvNinja's attempt to reproduce the SIGL model [11] has resulted in a significant misinterpretation of the original methodology. According to ProvNinja's analysis, SIGL purportedly decomposes the provenance graph into a series of

random walks, which are subsequently used to generate embeddings fed into an autoencoder. ProvNinja classifies SIGL as a path-based model, suggesting that it operates primarily on linear sequences derived from the graph, rather than leveraging the full graph structure.

B. SIGL's True Graph-Based Methodology

Contrary to ProvNinja's assertions, SIGL's approach is inherently graph-based. The core mechanism of SIGL involves encoding the entire graph's nodes into embeddings. This process not only captures the isolated characteristics of each node but also preserves the relational dynamics between nodes and their respective edges. Such a holistic view is crucial for the accurate detection of anomalies, as it allows the model to identify unusual patterns in the overall structure of the graph, rather than just in isolated parts.

SIGL's methodology is designed to reconstruct the entire graph's node relationships in the embedding space, which is essential for identifying deviations from typical behavioral patterns seen in system installations and interactions. This reconstruction is pivotal in understanding the complete context of node interactions, which a path-based approach, as misinterpreted by ProvNinja, would fail to achieve.

C. Consequences

The path-based model described by ProvNinja inherently loses significant structural information by focusing on random walks within the graph. This method oversimplifies the complex interdependencies within a provenance graph, which are critical for identifying and understanding the multifaceted nature of cyber threats and anomalies. By not considering the global structure and multiple node connections that define a provenance graph, ProvNinja's reproduced model likely underperforms in scenarios where the relational integrity of the graph elements plays a crucial role in anomaly detection.

D. Implications for Future Research

ProvNinja's misinterpretation underscores the necessity for a deeper understanding and accurate implementation of advanced anomaly detection models like SIGL. It highlights the importance of adhering closely to the intended design principles of such models to fully exploit their capabilities in detecting sophisticated cyber threats. Future attempts to reproduce or modify such models should prioritize a rigorous validation of methodological fidelity to avoid fundamental misinterpretations that could compromise the effectiveness of the anomaly detection process. In conclusion, while ProvNinja's efforts to engage with cutting-edge anomaly detection techniques are commendable, their misinterpretation of SIGL's methodology serves as a cautionary tale. It emphasizes the need for meticulous attention to detail in the reproduction of complex models and the potential pitfalls of deviating from established methodologies.

IV. LoTL Attacks

A. Background and Overview of LoTL Attacks

Living Off the Land (LoTL) attacks constitute a sophisticated cyber threat, where attackers exploit legitimate, local applications and tools already present on a target's system to conduct malicious activities. Unlike traditional malware, which typically involves the downloading and executing of malicious files, LoTL attackers manipulate existing, trusted software to achieve their objectives. This subversive technique minimizes the likelihood of detection by conventional security measures, which are generally designed to identify and react to unrecognized files. LoTL attacks are notably challenging to detect because they utilize tools that are widely trusted and embedded within the target environment, such as system administration tools and scripts [12][13].

The stealth and effectiveness of LoTL attacks make them particularly harsh and increasingly favored among cybercriminals [12]. These attacks often bypass standard antivirus and malware scanners that look for unfamiliar malicious software, thus challenging existing defense mechanisms and necessitating new strategies for detection and prevention [12].

B. Our Contribution to the Field

Our research has been directed towards exploring the intersection of intrusion detection systems that utilize raw logs and those that deploy provenance graphs derived from these logs. Provenance in cybersecurity tracks the origin and history of data within a system, providing a detailed context to raw log data which enhances the detection capabilities of traditional systems. This depth of insight is invaluable in identifying and understanding the sequences and relationships of system activities, particularly in the context of detecting LoTL attacks.

By integrating provenance, we can add a layer of context that is crucial for the detection of LoTL attacks, which are notoriously difficult to identify due to their reliance on legitimate system tools for malicious purposes. To test this idea, we have utilized content from repositories such as GTFO Bins and aim to execute LoTL attacks within a controlled environment, thereby testing and refining our hybrid model of provenance and log data, a similar approach has been highlighted in research done by Adobe [14]. This approach helps us pinpoint the origins and methods of these attacks more effectively and does not represent a shift away from our ongoing research into Software Installation graphs (SIGs) but rather an expansion to include LoTL attacks.

C. Aim of the Project

The primary aim of our project has been to enhance the detection of LoTL attacks through a deep learning approach, focusing on the challenging task of distinguishing between benign administrative operations and malicious actions within a system. We crafted a specialized corpus consisting of command line data, leveraging both benign and malicious examples to train our models. This corpus was carefully curated to include a diverse range of command lines, reflecting the real-world complexity and variability of system operations.

Our deep learning models were trained to recognize patterns and anomalies in this data, using features engineered from the tagged command lines to minimize data sparsity and improve detection accuracy.

D. Project Implementation

In our practical implementation, we utilized the OpTC dataset, which is a comprehensive collection of real-world data including a vast amount of Windows system logs. This dataset is particularly valuable for its extensive command line entries, making it an ideal resource for training models to detect anomalous behavior indicative of LoTL attacks. We extracted approximately 20,000 unique command lines from this dataset and are in the process of gathering more to further enrich our training corpus. The real-world nature of the OpTC dataset enhances the generality and reproducibility of our models, providing a robust foundation for developing effective cybersecurity solutions.

E. Challenges and Future Direction

However, our project is currently on hold as we navigate other aspects of cybersecurity and system reproducibility, motivated by the obstacles we faced over the past year. These challenges have broadened our research scope, leading us to explore additional cybersecurity domains that could influence the effectiveness and reproducibility of our ongoing research. Moving forward, we aim to integrate these insights into a more comprehensive approach to cybersecurity, potentially expanding our methodologies to other forms of cyber threats beyond LoTL attacks. This integrated approach will likely provide a better understanding and capability in tackling modern cyber threats in an increasingly complex digital landscape.

V. Reproducibility of Provenance systems

Accurately reproducing and understanding provenance-based systems like ShadeWatcher [2] and SIGL [10] are critical. However, challenges such as those presented by ProvNinja's [11] interpretations and implementations highlight significant hurdles that need addressing to advance research effectively.

A. Reproducibility Challenges in Cybersecurity Research

ShadeWatcher [2] is designed to analyze cyber threats by examining system entity interactions through advanced techniques like context-aware embeddings and graph neural networks (GNNs). The original design intended to use multi-hop interactions to explore relationships across a broad network. However, ProvNinja's version only used single-hop interactions, which significantly limited its ability to understand complex data flows and interactions across the system. By not looking beyond direct neighbors, ProvNinja's implementation misses out on capturing a full picture of system interactions, which is crucial for tracking and analyzing data flow thoroughly [11].

Furthermore, the issue of incomplete code releases from ShadeWatcher [2], which lacks essential documentation and functionalities, makes it difficult for other researchers to replicate, verify, and build upon the original study. Incomplete or poorly documented code releases create obstacles in achieving accurate replication of research results, which stunts the growth of collective knowledge in any forms of research.

B. Misinterpretation and Misclassification of Systems

The misclassification of SIGL by ProvNinja [11] as a path-based model instead of a graph-based model showcases another issue. SIGL's strategy is to use the entire graph structure to detect anomalies, not just random paths within it [10]. This approach helps preserve important relationships between nodes and edges, crucial for spotting deviations in usual patterns. Misinterpreting this could stem from a lack of understanding of the system or an attempt to simplify its implementation. Such inaccuracies can lead to ineffective security measures and reduce the overall reliability of cybersecurity defenses.

C. Solutions to aid researchers

Our upcoming project for the ACM REP (reproducibility conference) focuses on the significant challenges we've encountered in reproducing provenance or log-based anomaly threat detection tools [15]. The project addresses two high-level issues: first, inefficient algorithmic choices that can be substantially improved, and second, the challenges stemming from unfaithful tool implementations, which include gaps, blind spots, or ambiguous decisions made during the development phase. A key aspect of our research will involve demonstrating the existence of these issues through practical examples. For instance, we plan to show how a pruning approach, intended to reduce dataset size, actually resulted in significant losses in contextual information, leading to poorer results. This example aims to highlight how simplifying data handling can inadvertently strip away crucial information, undermining the tool's effectiveness. The project is an opportunity to engage with the broader community on these common issues, and serve as a useful resource for those embarking on a journey where reproducing systems is a crucial part of their research.

References

[1] N. Michael, J. Mink, J. Liu, S. Gaur, W. U. Hassan, and A. Bates, "On the forensic validity of approximated audit logs," in *Annual Computer Security Applications Conference*, 2020. DOI: 10.1145/3427228.3427272.

[2] J. Zengy, X. Wang, J. Liu, *et al.*, "Shadewatcher: Recommendation-guided cyber threat analysis using system audit records," in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022. DOI: 10.1109/SP46214.2022.9833669.

[3] M. Zipperle, F. Gottwalt, E. Chang, and T. S. Dillon, "Provenance-based intrusion detection systems: A survey," *ACM Computing Surveys*, vol. 55, no. 7, pp. 1–36, 2022. DOI: 10.1145/3539605.

[4] *Watson: Abstracting behaviors from audit logs via aggregation of contextual semantics - ndss symposium*, Jun. 2023. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/watson-abstracting-behaviors-from-audit-logs-via-aggregation-of-contextual-semantics/.

[5] X. Han, T. Pasquier, A. Bates, J. Mickens, and M. I. Seltzer, "Unicorn: Runtime provenance-based detector for advanced persistent threats," in *Network and Distributed System Security Symposium (NDSS'20)*, The Internet Society, 2020.

[6] Z. Cheng, Q. Lv, J. Liang, *et al.*, *Kairos: Practical intrusion detection and investigation using wholesystem provenance*, arXiv preprint arXiv:2308.05034, 2023. [Online]. Available: https://arxiv.org/abs/2308.05034.

[7] H. Ding, J. Zhai, Y. Nan, and S. Ma, "Airtag: Towards automated attack investigation by unsupervised learning with log texts," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 373–390.

[8] A. Alsaheel, Y. Nan, S. Ma, *et al.*, "Atlas: A sequence-based learning approach for attack investigation," in *USENIX Security 2021*, 2021.

[9] A. Gehani and D. Tariq, "Spade: Support for provenance auditing in distributed environments," in *Middleware 2012*, P. Narasimhan and P. Triantafillou, Eds., Berlin: Springer Berlin, 2012, pp. 101–120.

[10] X. Han, X. Yu, T. Pasquier, *et al.*, "Sigl: Securing software installations through deep graph learning," in *Security Symposium (Sec'21)*, USENIX, 2021.

[11] K. Mukherjee, J. Wiedemeier, T. Wang, *et al.*, "Evading provenance-based ml detectors with adversarial system actions," in *USENIX Security*, 2023, pp. 1199–1216.

[12] J. Phipps, *Living off the land attacks: Lotl definition & prevention*, Oct. 2023. [Online]. Available: https://www.esecurityplanet.com/networks/living-off-the-land-attacks/.

[13] *An overview of living off the land (lotl) attack techniques*, Lacework, May 2024. [Online]. Available: https://www.lacework.com/blog/an-overview-of-living-off-the-land-lotl-attack-techniques/.

[14] *Machine learning and feature engineering for detecting living off the land attacks*, GitHub, Adobe, 2024. [Online]. Available: https://github.com/adobe/libLOL.

[15] *Home*, EIG Repro, Nov. 2023. [Online]. Available: https://reproducibility.acm.org/.